DOCTORAL THESIS
2016

FAST AND ACCURATE IMAGE REGISTRATION
APPLICATIONS TO ON-BOARD SATELLITE IMAGING

Martin Emilio Rais

Universitat
de les Illes Balears

DOCTORAL THESIS
2016

Doctoral Programme of Mathematics

FAST AND ACCURATE IMAGE REGISTRATION APPLICATIONS TO ON-BOARD SATELLITE IMAGING

Martin Emilio Rais

Advisors Jean-Michel **Morel**
Bartomeu **Coll-Vicens**

in fulfillment of the requirements
for the degrees of
Doctor by the Universitat de les Illes Balears
and
Doctor by the École Normale Supérieure de Cachan

A mi hermano Andrés.

Abstract

English

This thesis starts with an in-depth study of fast and accurate sub-pixel shift estimation methods. A full comparison is performed based on the common shift estimation problems occurring in real-life applications, namely, varying SNR conditions, different displacement magnitudes, non-preservation of the brightness constancy constraint, aliasing, and most importantly, limited computational resources. Based on this study, in collaboration with CNES (the French space agency), two problems that are crucial for the digital optics of earth-observation satellites are analyzed.

Numerical wavefront estimation. We first study the wavefront correction problem in an active optics context. We propose a fast and accurate algorithm to measure the wavefront aberrations on a Shack-Hartmann Wavefront Sensor (SHWFS) device observing the earth. We give here a review of state-of-the-art methods for SHWFS used on extended scenes (such as the earth) and devise a new method for improving wavefront estimation, based on a carefully refined approach based on the optical flow equation. This method takes advantage of the small shifts observed in a closed-loop wavefront correction system, yielding improved accuracy using fewer computational resources. We also propose two validation methods to ensure a correct wavefront estimation on extended scenes. While the first one is based on a numerical adaptation of the (theoretical) lower bounds of image registration, the second method rapidly discards landscapes based on the gradient distribution, inferred from the Eigenvalues of the structure tensor.

Registration-based time delay integration. The second satellite-based application that we address is the numerical design of a new generation of Time Delay Integration (TDI) sensor. In this new concept, active real-time stabilization of the TDI is performed to extend considerably the integration time, and therefore to boost the images SNR. The stripes of the TDI cannot be fused directly by addition because their position is altered by microvibrations. These must be compensated in real time using limited onboard computational resources with high subpixel accuracy. We study the fundamental performance limits for this problem and propose a real-time solution that nonetheless gets close to the theoretical limits. We introduce a scheme using temporal convolution together with online noise estimation, gradient-based shift estimation and a non-conventional multiframe method for measuring global displacements. The obtained results are conclusive on the fronts of accuracy and complexity and have strongly influenced the final decisions on the future configurations of Earth observation satellites at CNES.

Improving image registration by a new RANSAC method. For more complex transformation models, a new image registration method performing accurate robust model estimation through point matches between images is proposed here. The difficulty coming from the presence of outliers causes the failure of traditional regression methods. In computer vision, RANSAC is definitely the most renowned method that overcomes such difficulties. It discriminates outliers by randomly generating minimalist sampled hypotheses and verifying their consensus over the input data. However, its response is based on the single iteration that achieved the largest inlier support, while discarding all other generated hypotheses. We show here that the resulting accuracy can be improved by aggregating all hypotheses. We also propose a simple strategy that allows to rapidly average 2D transformations, leading to an almost negligible extra computational cost. We give practical applications to the estimation of projective transforms and homography+distortion transforms. By including a straightforward adaptation of the locally optimized RANSAC in our framework, the proposed approach improves over every other available state-of-the-art method. A complete analysis of the proposed approach is performed, demonstrating its improved accuracy, stability and versatility.

Français

Cette thèse commence par une étude approfondie des méthodes d'estimation de décalage sous-pixeliques rapides et précises. Une comparaison complète est effectuée prenant en compte problèmes d'estimation de décalage existant dans des applications réelles, à savoir, avec différentes conditions de SNR, différentes grandeurs de déplacement, la non préservation de la contrainte de luminosité constante, l'aliasing et, surtout, la limitation des ressources de calcul. Sur la base de cette étude, en collaboration avec le CNES (l'agence spatiale française), deux problèmes qui sont cruciaux pour l'optique numérique des satellites d'observation de la terre sont analysés.

Estimation numérique du front d'onde. Nous étudions d'abord le problème de correction de front d'onde dans le contexte de l'optique actif. Nous proposons un algorithme rapide et précis pour mesurer les aberrations de front d'onde sur un senseur de type Shack-Hartmann (SHWFS en anglais) en observant la terre. Nous proposons ici une revue de l'état de l'art des méthodes pour le SHWFS utilisé sur des scènes étendues (comme la terre) et concevons une nouvelle méthode pour améliorer l'estimation de front d'onde, en utilisant une approche basée sur l'équation du flot optique soigneusement raffiné. Cette méthode profite des petits décalages observés dans un système de correction de front d'onde en boucle fermée, ce qui améliore la précision tout en minimisant les ressources de calcul. Nous proposons également deux méthodes de validation afin d'assurer une estimation correcte du front d'onde sur les scènes étendues. Tandis que la première est basée sur une adaptation numérique des bornes inférieures (théoriques) pour le recalage d'images, la seconde méthode défausse rapidement les paysages en se basant sur la distribution des gradients.

Intégrateur numérique basé sur le recalage. La deuxième application de satellite abordée est la conception numérique d'une nouvelle génération de senseur du type Time Delay Integration (TDI). Dans ce nouveau concept, la stabilisation active en temps réel du TDI est réalisée pour étendre considérablement le temps d'intégration, et donc augmenter le RSB des images. Les lignes du TDI ne peuvent pas être fusionnées directement par addition parce que leur position est modifiée par des microvibrations. Celles-ci doivent être compensées en temps réel en utilisant des ressources de calcul limitées sur le satellite, avec une précision sous-pixellique. Nous étudions les limites fondamentales théoriques de ce problème et proposons une solution en temps réel qui s'en approche. Nous présentons un système utilisant la convolution temporelle conjointement à une estimation en ligne du bruit de capteur, à une estimation de décalage basée sur les gradients et à une méthode multiimage non conventionnelle pour mesurer les déplacements globaux. Les résultats obtenus sont concluants sur les fronts de la précision et de la complexité et ont fortement influencé les décisions finales sur les futures configurations des satellites d'observation de la Terre au CNES.

Amélioration du recalage d'images par une nouvelle méthode de RANSAC. Pour des modèles de transformation plus complexes, une nouvelle méthode effectuant l'estimation précise et robuste des modèles de mise en correspondance des points d'intérêt entre images est proposée. La difficulté provenant de la présence de fausses correspondances et de mesures bruitées conduit à un échec des méthodes de régression traditionnelles. En vision par ordinateur, RANSAC est certainement la méthode la plus utilisée pour surmonter ces difficultés. RANSAC est capable de discriminer les fausses correspondances en générant de façon alétaoire des échantillons d'hypothèses minimalistes et en vérifiant leur consensus sur les données d'entrée. Cependant, sa réponse est basée sur la seule itération qui a obtenu le consensus le plus large, et elle ignore toutes les autres hypothèses. Nous montrons ici que la précision obtenue peut être améliorée en agrégeant toutes les hypothèses envisagées. Nous proposons également une stratégie simple qui permet de moyenner rapidement des transformations 2D, ce qui réduit le coût supplémentaire de calcul à quantité négligeable. Nous donnons des applications réelles pour estimer les transformations projectives et les transformations homographie + distorsion. En incluant une adaptation simple de LO-RANSAC dans notre cadre, l'approche proposée bat toutes les méthodes de l'état de l'art. Une analyse complète de l'approche proposée est réalisée, et elle démontre un net progrès en précision, stabilité et polyvalence.

Español

La presente tesis comienza con un estudio en profundidad de métodos rápidos y precisos de estimation de translación sub-pixelicos. El rendimiento de los métodos es evaluada ante la presencia de problemas que suelen ocurrir en aplicaciones reales donde se involucra la estimación de translación, en particular, bajo diferentes niveles de señal/ruido, translaciones de diversas magnitudes, ante la no preservación de luminosidad, la presencia de aliasing, y sobre todo, bajo límites estrictos de capacidad de cálculo. Sobre la base de dicho estudio, en colaboración con la CNES (la agencia espacial francesa), dos problemas cruciales para la óptica digital de satélites de observación de la tierra son analizados.

Estimación digital del frente de onda. Estudiamos primero el problema de la corrección del frente de onda en un contexto de óptica activa. Proponemos un método rápido y preciso para medir las aberraciones del frente de onda utilizando un sensor Shack-Hartmann (SHWFS en inglés) observando la tierra. Presentamos una revisión de métodos usados por SHWFS sobre escenas extendidas (como la tierra) y diseñamos un nuevo método que mejora la estimación del frente de onda, utilizando un enfoque cuidadosamente refinado basado en la ecuación de flujo óptico. El método propuesto se aprovecha de las pequeñas translaciones observadas en un sistema de corrección de frente de onda de lazo cerrado para mejorar la precision usando una menor cantidad de recursos computacionales. También proponemos dos métodos de validación para asegurarnos una estimación correcta sobre escenas extendidas. Mientras el primero está basado en una adaptación numérica de los límites inferiores teóricos de registro de imágenes, el segundo descarta rápidamente paisajes basándose en su distribución de gradientes, inferida de los valores propios del tensor de estructura.

Time delay integration basado en el registro de imágenes. La segunda aplicación de satélites abordada es la concepción digital de una nueva generación de sensores de tipo Time Delay Integration (TDI). En este nuevo concepto, se realiza la estabilización en tiempo real del TDI para extender considerablemente el tiempo de integración, y por lo tanto mejorar la relación señal/ruido resultante. Las lineas del TDI no puede ser fusionadas directamente debido a que su posición es alterada por microvibraciones. Estas deben ser compensadas con precisión sub-pixelica en tiempo real utilizando los recursos computacionales limitados existentes en el satélite. Estudiamos los límites fundamentales teóricos de registro de imágenes para este problema y proponemos una solución que los aproxima. El esquema presentado utiliza la convolución temporal conjuntamente con una estimación de ruido en línea, un método de estimación de translación basado en gradientes, y un método no convencional multi-imágen para medir desplazamientos globales. Los resultados son concluyentes tanto en precisión como en cuanto a la complejidad y han fuertemente influenciado las decisiones finales sobre las futuras configuraciones de satélites de observación de la tierra de la CNES.

Mejorando el registro de imágenes mediante una nueva variante de RANSAC. Para transformaciones entre imágenes más complejas, proponemos un nuevo algoritmo de registro de imágenes a partir de pares de puntos coincidentes. La presencia de pares de puntos ajenos al modelo, llamados *outliers*, hacen que los métodos de regresión tradicionales fallen. En visión por ordenador, RANSAC es definitivamente el método más popular para lidiar con *outliers*. Es capaz de discriminarlos generando hipótesis aleatorias tomando muestras minimales y luego verificando su consenso sobre los datos de entrada. Sin embargo, su respuesta está basada en la única iteración que obtuvo el mayor consenso, descartando el resto de las hipótesis generadas. Mostramos aquí que la precisión del método puede ser mejorada agregando todas dichas hipótesis. Proponemos además una simple estrategia que nos permite rápidamente promediar transformaciones 2D, lo cual conduce a un incremento del costo computacional total prácticamente nulo. Damos aplicaciones reales para estimar transformaciones proyectivas y para transformaciones de tipo homografía+distorsión. Incluyendo una adaptación directa del método LO-RANSAC en nuestro esquema, el método propuesto mejora todo método disponible del estado del arte. Realizamos un análisis completo del método, demostrando su precisión, estabilidad y versatilidad.

Català

Aquesta tesi comença amb un estudi en profunditat dels mètodes ràpids i precisos d'estimació de desplaçament subpíxel. Hom realitza una comparació completa de tots els mètodes coneguts sobre els problemes típics que apareixen a les aplicacions; és a dir, diferents nivells de soroll, diferents mides del desplaçament, canvis d'intensitat en les imatges, *aliasing* i, el que és més important, recursos computacionals limitats. A partir d'aquest estudi, en col·laboració amb el CNES (l'agència espacial francesa), hom ha estudiat dos problemes crucials en l'òptica digital dels satèl·lits d'observació terrestre moderns.

Estimació numèrica de fronts d'ona. En primer lloc, estudiem el problema de l'estimació del front d'ona en el context de l'òptica activa. Proposem un algoritme ràpid i precís per mesurar la forma del front d'ona en un dispositiu tipus Shack-Hartmann (SHWFS) dedicat a l'observació de la terra. Fem una revisió dels mètodes actuals per SHWFS utilitzats en les escenes grans i ideem un nou mètode per millorar l'estimació de front d'ona, basat en l'equació de flux òptic. Aquest mètode analitza els petits canvis observats en el sistema de correcció de front d'ona en un bucle tancat, cosa que permet obtenir una millor precisió i utilitzar menys recursos computacionals. També proposem dos mètodes de validació per assegurar una bona estimació global del front d'ona en imatges grans. Mentre que el primer mètode es basa en una adaptació numèrica dels (teòrics) límits inferiors del registrat de la imatge, el segon mètode descarta ràpidament les escenes segons la distribució dels gradients, deduïda dels valors propis del tensor d'estructura.

***Time Delay Integration* basada en registrat.** La segona aplicació que presentem per l'òptica dels satèl·lits és el disseny numèric d'una nova generació d'Integració Temps de retard (TDI) del sensor. En aquest nou concepte, hom realitza una estabilització en temps real de les imatges per tal d'ampliar considerablement el temps d'integració, i per tant per reduir el soroll de les imatges. Les línies de la TDI no es poden fusionar directament mitjançant l'addició, ja que la seva posició s'altera per microvibracions i pel propi desplaçament del satèl·lit. Aquest moviment, doncs, ha de ser compensat en temps real utilitzant els pocs recursos computacionals disponibles a bord, i alhora es requereix una bona precisió sub-pixèlica. Estudiem els límits de rendiment fonamentals d'aquest problema i proposem una solució en temps real que s'acosta a aquests límits teòrics. Per estimar aquest desplaçaments, introduïm un algoritme basat en la convolució temporal juntament amb l'estimació del soroll, l'estimació de canvi basat en la pendent i en l'ús no convencional d'un gran conjunt de línies. Els resultats obtinguts són concloents en els fronts de precisió i complexitat i han influït molt en les decisions finals del CNES sobre les futures configuracions de satèl·lits d'observació de la Terra.

Millora de registrat d'imatges mitjançant un nou mètode RANSAC. Finalment, per models de transformació més complexos, hom proposa un nou mètode d'enregistrat d' imatges a partir de correspondències robustes entre punts d'interès. La dificultat prové de la l'existència habitual d'algunes correspondències incorrectes, que provoquen el fracàs dels mètodes de regressió tradicionals. En visió per ordinador, RANSAC és sens dubte el mètode més conegut que resol aquestes dificultats. És capaç de descartar les correspondències incorrectes mitjançant la generació d'hipòtesis a l'atzar, i verificant llur consens sobre les dades d'entrada. No obstant això, la resposta es basa únicament en la hipòtesi que aconsegueix el major suport, mentre es descarten totes les altres hipòtesis generades. Aquí mostrem com es pot millorar la precisió de RANSAC mitjançant l'agregació ponderada de totes les hipòtesis. També proposem una estratègia simple que permet calcular la mitjana de moltes transformacions 2D, a canvi d'un increment gairebé negligible del cost computacional. Com a aplicació pràctica d'aquesta idea, proposem algoritmes per estimar transformacions projectives amb i sense distorsió òptica radial. Gràcies a la nostra adaptació de RANSAC, els mètodes proposats milloren tots els altres mètodes de l'estat de l'art. Per justificar aquesta millora, fem una anàlisi completa dels resultats que demostra l'increment de precisió, estabilitat i versatilitat.

Acknowledgements

My master's advisor from Buenos Aires, Marta Mejail, together with some good friends of mine, Norberto Goussies and Francisco Gomez Fernandez, introduced me to the world of image processing. It was when I met Jean-Michel Morel that my passion for research started. His hard work, motivation, genius and mostly the love for what he does has been a constant source of inspiration all along my PhD. I don't have enough words to thank him for what he did during the last years. The same goes to my other advisor Bartomeu Coll. Apart from everything he has helped me from an academical point of view, his warmth, human values, simplicity and humility have been very important to me. I have to mention as well the excellent beer he brews.

I would also like to thank the image quality group from CNES, in particular Philippe Kubik, Gwendoline Blanchet, Jean-Marc Delvit, Christophe Latry and Carole Thiebaut, for having proposed several topics of this PhD thesis as well as having collaborated with fruitful discussions and providing image simulations and datasets.

My gratitude goes as well to the thesis reviewers Frédéric Falzon and Pascal Monasse, for their fruitful remarks and for taking the time to read this extensive manuscript. Also, for the members of the jury Vincent Michau, Vasileios Argyriou, Jiří Matas, Coloma Ballester, Jose Luis Lisani and Catalina Sbert for having accepted to be part of the evaluation board.

In the CMLA, I met some wonderful people. I begin by thanking the secretaries Micheline, Veronique and Virginie for all the help they gave me. I would also like to thank my friends at work: Ives, el Matu, Carlitos and Rafa, with whom I spent lots of discussions, funny talks, beers and good times together. I would like to thank as well Carlo, Miguel, Axel and Mauricio, some amazing researchers I found here. And finally, I want to thank particularly two guys that strongly contributed with this dissertation: Gabriele and Enric. Their guidance, patience, methodology and motivation have strongly influenced me along my scientific career, and together with JM, were constant sources of motivation. Your help was invaluable to me.

In the TAMI group, I also met excellent researchers. I would like to begin by thanking Toni for all the help he gave me during this thesis. His perseverance as well as his expertise in the research community have undoubtedly contributed with my academic growth. I would also like to thank the rest of the people in the group: Ana Belen, Joan and Julia.

I would have never been able to finish this PhD without the support of my friends. In France I met some amazing people. Hernan was my partner during my first years here, and a great friend. Aleksandra turned fast into my sister, and together with Claudia and Hernan, were my backbone in the first years of this journey and made my everyday life much happier. I also have to include my Argentinean friends I made here: Alito, la chiqui, la chueka, la peti and Lala.

Talking about friends, I met two other particular individuals with whom I shared my every day life and that will stay in my heart forever. Larucha is the best that Uruguay has ever produced (thanks to la Petruzi, the second best). Indeed, it took less than a year for her to become “my wife”. We have spent so much time together that the nickname seemed just right. She was also part of my backbone, an excellent friend and researcher and probably, one of the reasons I succeeded to live five years abroad. This list wouldn’t be complete without Nicolino. We began to know each other when we travelled to Australia for ICIP, and since then he became more and more important in my life, together with Sandra. Besides being the best ever, he is an amazing person and a great friend. It seems that the cake was not a lie after all.

I would also like to thank my friends from Argentina. This includes Fede, el negro, el blanco, el pelado, camisa, Jr, Jinkis, Jony and Zaina. I regret not being able to be there for some of you in some situations, but even without communicating so often, you are amazing, and own a really important part of my heart. This obviously extends to “las chicas”: brujisina, veci, caskis, queipusi and la primita, and includes Sofi, Aye and Agata as well. You are the best company this manga de boludos will ever find. What can I say about the newcomers? Of course I am dying to share more time with la mumita and malenuchi, I wish I could see them often. Seeing a simple picture of them growing brightens up my day.

I would also like to thank the people that have been with me and my family all along this journey: mi tía Hilda, mi tío Go, Axi y Lean, Miguel y Sarita, Diego, Brenda y Ernest, el tío Eduardo y la tía Elida, Luciana, Hernan, Nati y Pablo, el tio Jorge y la tía Diana.

Finally, I have to thank the five most important people in my life. First is my girlfriend Victoria. Not only for tolerating my snoring, my arriving late because of the thesis, among many other, but also because of all the love and support she gives me every day. Also, to my grandma la Nani, which I missed a lot during all these years. My parents Rita and Jorge were the main reason I was able to start and to finish this thesis. Their support, education, constant caring, ethical values, strong commitment to work, and their love made me what I am today. At last, my brother Andrés, who has been my main inspiration to fight and overcome all difficulties. I miss you a lot. I love you with all my heart.

Last but not least, I would like to thank the Ministerio de Economía y Competitividad for the FPI scholarship associated with the project TIN2011-27539.

Contents

Chapter 1

Introduction

In this thesis, we study the problem of image registration. Two or more images are said to be registered when a unique coordinate system can be used to address the same content for every image. This is equivalent to first designating a reference image or coordinate system, and then align all images of the same scene with respect to it by applying geometric transformations. In particular, we first study the problem when the transformations are 2D translations, by focusing on speed and accuracy. Finally, we expand to other more complex transformation types.

This thesis starts with an in-depth study of fast and accurate sub-pixel shift estimation methods. We focus on two distinct types: gradient-based methods working in the spatial domain and phase correlation methods working in the frequency domain. A full comparison is performed based on the common shift estimation problems occurring in real-life applications, namely, varying SNR conditions, different displacement magnitudes, non-preservation of the brightness constancy constraint, aliasing, and most importantly, limited computational resources. The most important result from Chapter 2 is an in depth review of the most important shift estimation method, together with a practical recipe indicating which methods to use in which circumstances.

Based on this study, in collaboration with CNES (the French space agency), two problems that are crucial for the digital optics of earth-observation satellites were analyzed in chapters 3 and 4, and tentative solutions were proposed for each of them. The major constraint faced in both cases was the need for on-board real-time processes, using few computational resources. Yet optimal numerical solutions must be found for both problems. Indeed both were termed essential to be able to reach high PSNRs in Earth imaging and to avoid the optical aberrations caused by the large size of a light telescope.

We first study in Chapter 3 the wavefront correction problem based on a fast wavefront sensing and on-board correction. We propose a fast and accurate algorithm to measure the wavefront aberrations on a Shack-Hartmann Wavefront Sensor (SHWFS) device observing the earth. This sensor is generally used on the observation of stars to correct aberrated wavefronts caused by rays passing through the earth's atmosphere. We propose in this chapter a review of state-of-the-art methods for SHWFS used instead on extended scenes (such as the earth). We devise a new method for improving wavefront estimation when observing the earth, based on a carefully refined gradient-based approach. This method takes advantage of the small shifts observed in a closed-loop wavefront correction system, yielding an improved accuracy with fewer computational resources. To perform an extensive comparison of the proposed approach with state-of-the-art methods, a simulator was created following the provided CNES specification.

When dealing with extended scenes, it is of vital importance to detect whether the current observed landscape is suitable for accurate numerical wavefront sensing. To this end, two methods are proposed. The first one is based on a numerical adaptation of the (theoretical) lower bounds of image registration obtained by Cramer-Rao Lower Bound (CRLB). The second method is a fast test that discards landscapes based on the gradient distribution, inferred from the eigenvalues of the tensor matrix.

The main result obtained from this chapter is a new method, together with a prevalidation procedure, that improves on the existing state-of-the-art methods used for scene-based SHWFS on both accuracy and processing time.

The second satellite-based application addressed in Chapter 4 is the numerical design of a new generation of smart Time Delay Integration (TDI) images. In this new concept, active real-time stabilization of the TDI is performed to extend considerably the integration time, and therefore to boost the SNR of captured images. The stripes of the TDI cannot be fused directly by addition because their position is altered by microvibrations and other geometric perturbations. These must be compensated using limited onboard computational resources with high subpixel accuracy and in real time. We study the fundamental performance limits for this problem and propose a real-time solution that nonetheless gets close to the theoretical limits. We introduce a scheme using temporal convolution together with online noise estimation, gradient-based shift estimation and a non-conventional multiframe method for measuring global displacements. We compare our results with the theoretical bounds and other state-of-the-art methods. The results are conclusive on the fronts of accuracy and complexity and have strongly influenced the final decisions on the future configurations of Earth observation satellites at CNES.

Finally, during this thesis, the author also worked on multi-scale image fusion of paintings taken by low-cost commercially available cameras. As a spin-off of this project, a new method performing accurate robust model estimation through point matches between images is proposed in Chapter 5. The difficulty coming from the presence of outliers and noisy measurements causes the failure of traditional regression methods. In computer vision, RANSAC is definitely the most used method, able to deal with outliers while performing model estimation. Its success is mainly due to its straightforward implementation, highly accurate results and fast processing times. It is able to discriminate outliers by randomly generating minimalist sampled hypotheses and verifying their consensus over the input data. However, its response is based on the single random sample that achieved the largest inlier support, while discarding all other potentially useful generated hypotheses. In this thesis we show that the resulting accuracy can be improved by aggregating all hypotheses generated in the course of the RANSAC search. This yields RANSAAC, a framework that improves systematically over RANSAC by statistically aggregating hypotheses. We also proposed a simple strategy that allows to rapidly average 2D transformations, leading to an almost negligible extra computational cost. We give practical applications to estimate projective transforms and homography+distortion transforms. By including a straightforward adaptation of the Locally Optimized RANSAC in our framework, the proposed approach improves over every other available state-of-the-art method. A complete analysis of the proposed approach and an extensive comparison with most state-of-the-art algorithms have been performed, demonstrating improved accuracy, stability and versatility.

In this introduction, we give an overview of each of the chapters of this thesis, together with their main results.

1.1 Chapter 2: Review of Global Subpixel Shift Estimation Methods

The problem of global subpixel shift estimation appears in several applications related to image processing. When high accuracy is not required, most authors tend to use the well-known cross-correlation method which usually meets the performance requirements for such tasks. However, under more demanding accuracy constraints, cross-correlation approaches are either not accurate enough or their computational costs are prohibitive. Moreover, problems such as time constraints, varying illumination conditions, aliasing, among several others, complicate the shift estimation task. What is worse, there is little know how about the choice of the adapted shift estimation method.

This chapter offers a detailed analysis on fast and accurate shift estimation methods. Since low computationally intensive methods are required, we focus only on differential methods and phase correlation approaches, offering a complete study on both methodologies. Here, we tackle the case where both images have the same size and are contaminated by white Gaussian noise with the same standard deviation. We focus on the case where the 2D displacement is smaller than one pixel in both dimensions and that the brightness constancy constraint holds, nevertheless we perform tests validating the performance of each when these conditions do not hold. Our objective is therefore to give a practical recipe showing which methods are more suited depending on the conditions of the task. By quantizing the various variations of the conditions, we evaluate on each of them more than a thousand variants of shift estimation methods, and summarize the results here.

Differential methods, also known as Gradient-Based Shift Estimation (GBSE), relate the difference between two successive frames to the spatial intensity gradient of the first image. This relationship is obtained by approximating the shifted image by its first order Taylor approximation. The shift can then be easily estimated using linear least squares minimization, as originally proposed by Lucas and Kanade [97]. However, more elaborate methods can be applied to achieve higher precision. Total least squares minimization [80, 181] slightly improves the results, although it requires the estimation of the singular value decomposition. In fact, the GBSE method is biased [119, 139]. An indirect way to reduce this bias is through iterative procedures [12, 119] or multiscale adaptations [131, 167], which used together, yield excellent results using few computational resources. Other approaches that reduce the influence of the bias are the corrected gradient estimation method [81] and the bidirectional bias correction approach [122]. We also analyzed the influence of gradient estimation and the interpolation method used within GBSE approaches. **We observed that choosing the correct method could yield improvements up to an order of magnitude and are critical for improving the performance of the method.**

Phase correlation methods are based on the fact that most of the information about the relative displacements of objects between two images is contained in the phase of their cross-power spectrum. Several methods exploit this to estimate the displacement both in the Fourier and the spatial domain. These methods require the computation of the discrete Fourier transform (DFT), which could be prohibitive in certain scenarios. Nevertheless, they are able to achieve very accurate results without further computational effort, and they can be used when images are seriously distorted, in either geometry or intensity. Most methods estimate the shift by fitting a function to the phase correlation matrix, obtained by applying the inverse Fourier transform of the cross-power spectrum [1, 8, 57, 69, 132, 134, 162]. Other methods compute the shift directly in the Fourier domain [13, 85, 151, 159].

As a drawback, phase correlation approaches assume circular shifts between images. Since in general, the shift between both images is a simple linear shift rather than a circular shift, windowing functions are employed to mitigate problems related to image edges, by performing an apodization of the input images. For images that have repeating objects, phase correlation may also yield ambiguous results with several correlation peaks, and its use on those cases is not advisable.

Subspace phase correlation methods search for one dimensional representations of the images to then estimate both shifts independently. One of such representations is obtained by taking the image projection in each dimension [5, 138]. Then the shift is estimated either using 1D versions of the phase correlation approach, either by correlating their gradient [135]. Another method, based on the fact that the phase correlation matrix obtained from noiseless images is of rank-1, computes the shift from both its left and right singular vectors [75].

Other studied approaches are based on correlating gradient information. First introduced by Argyriou *et al.* [7], several works followed [135, 178, 179]. In our experiments, gradient correlation approaches yielded more accurate and consistent results than phase correlation methods, and proved to be more robust against noise and aliasing. Nevertheless, they did not achieve as accurate results as GBSE approaches, particularly with small shift magnitudes. Indeed, we observed that GBSE approaches seems to be more accurate and tolerant to noise than phase-correlation methods. They are also computationally cheaper and more stable. However, when the estimated shifts are large, they must be used within pyramidal multiscale approaches which tend to reduce their stability. What is more, when both images to register differ, GBSE approaches proved to be less tolerant than phase correlation methods. We refer the reader directly to the conclusions of sections 2.4.2, 2.4.3, 2.4.4, 2.4.5 and 2.4.6 for a brief summary of this chapter. Nevertheless, in Fig. 1.1 we compare the general performance (by averaging among all discretized shift magnitudes evaluated and all noise levels) of the most representative methods. Methods beginning with LS, TLS, ULS, CLS and MS all refer to GBSE approaches, GC denotes gradient correlation and PC phase correlation.

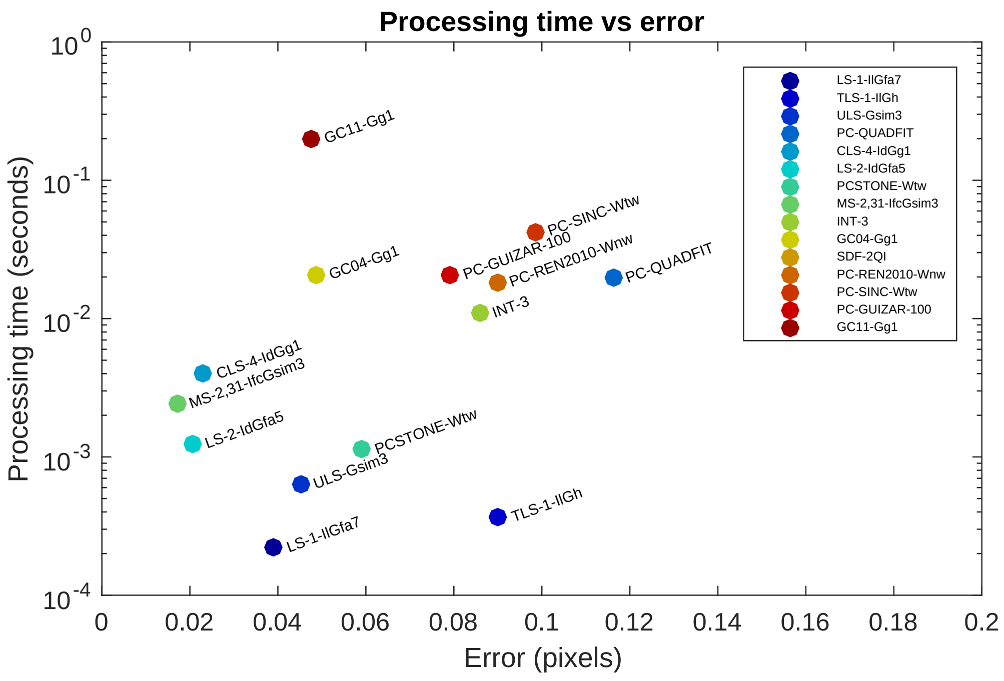


Figure 1.1 – Average execution time (log scale) vs accuracy for some representative methods averaged over the first three shift magnitudes and all noise levels. The meaning of the principal prefixes are: LS = least squares, TLS = total least squares, MS = multiscale, CLS = corrected least squares, ULS = bidirectional bias corrected least squares, GC = gradient correlation, PC = phase correlation.

The analysis made in this chapter is of key importance for the rest of this thesis, particularly for chapters 3 and 4, where two remote sensing applications arising from earth-observation satellites are analyzed.

1.2 Chapter 3: Improving wavefront sensing with a Shack Hartmann device

In order to achieve higher resolutions, current earth-observation satellites use larger light-weight primary mirrors that can deform over time, impacting on image quality [21, 52]. We evaluated the possibility of compensating these deformations directly in the satellite by combining a deformable mirror with a Shack-Hartman wavefront sensor (SHWFS) while observing the earth. The SHWFS uses an array of lenslets to measure the deformation of the incoming wavefront. The shift of each lenslet focal plane image is proportional to the mean slope of the wavefront in the subaperture onto this lenslet. It therefore yields a discrete local approximation of the slope of the wavefront, as observed from Fig. 1.2, which is later used to approximate the actual wavefront.

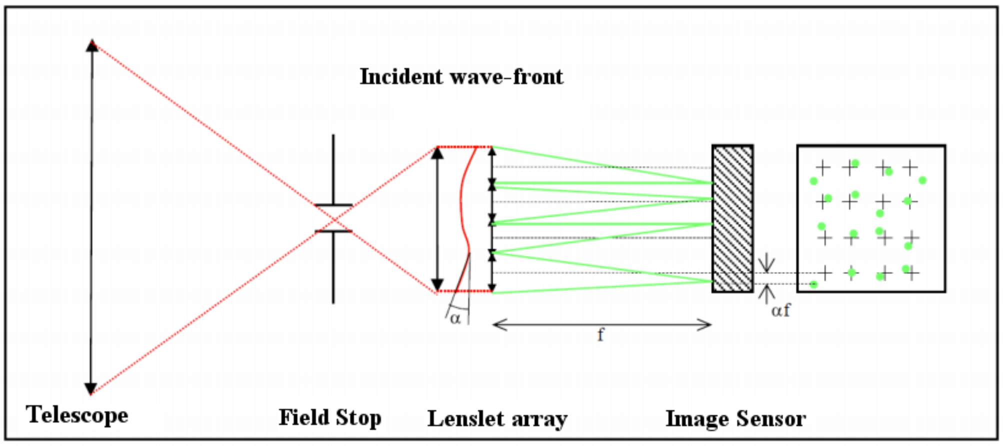


Figure 1.2 – A Shack-Hartmann Wavefront Sensor measures the wavefront by computing the local shifts between the detected spots (in green) and the reference crosses (in black), which would occur if no deformation were present.

Shack-Hartmann wavefront sensors are frequently employed in astronomy to measure the wavefront aberrations produced by the atmospheric turbulence. They work by imaging a point source such as a star, and estimate the relative displacement between a reference image and all other subimages. Their performance is directly linked to the accuracy of the shift estimation algorithm, which must be computationally cheap to be executed on-board.

When observing the earth from space this task becomes more challenging. In particular, the shift estimation task has to be performed using extended scenes. The problem of shift estimation using SHWFS on extended sources has been addressed before, dating as far back as the work of Michau *et al.* in 1993. These approaches can be divided in: correlation methods either in the space [95, 108, 109] or frequency domain [85, 95, 124, 151, 153], phase correlation methods [85, 159], and iterated methods [68, 151, 153], which progressively improve the estimation.

Based on the study performed in Chapter 2, we propose a new shift estimation method that is based on the GBSE approach. This method exploits the fact that the wavefront correction is done in a closed loop so that aberrations on the mirror would translate into small displacements on the SHWFS subapertures. In fact, assuming displacements below half of a pixel, depending on the available computational resources, we propose to use the Lucas and Kanade [97] gradient-based method estimating the image derivatives using the Simoncelli [154] approach. If more resources are available, we propose to iterate the method by resampling the images in the Fourier domain [20].

Since the wavefront sensing is performed on board by observing the earth, it is important to verify if the current landscape observed by the satellite is suitable for performing an accurate measurement of the wavefront. To this end, we propose two methods. The first one is a numerical adaptation of the (theoretical) lower bounds of image registration obtained by Cramer-Rao Lower Bound (CRLB), and the second one is a fast test based on the gradient distribution, inferred from the eigenvalues of the tensor matrix, which if used in conjunction with the GBSE method, does not imply further computational costs. Both proposed tests were able to correctly discard potentially ambiguous cases.

In our experiments, the proposed shift estimation methods proved to be more accurate and stable, as well as less sensitive to noise than all the state-of-the-art approaches, permitting a more precise on-board wavefront estimation. This can be observed from Fig. 1.3, where state-of-the-art methods are compared to both proposed variants using two different test images extracted from a high resolution satellite image of the city of Cannes.

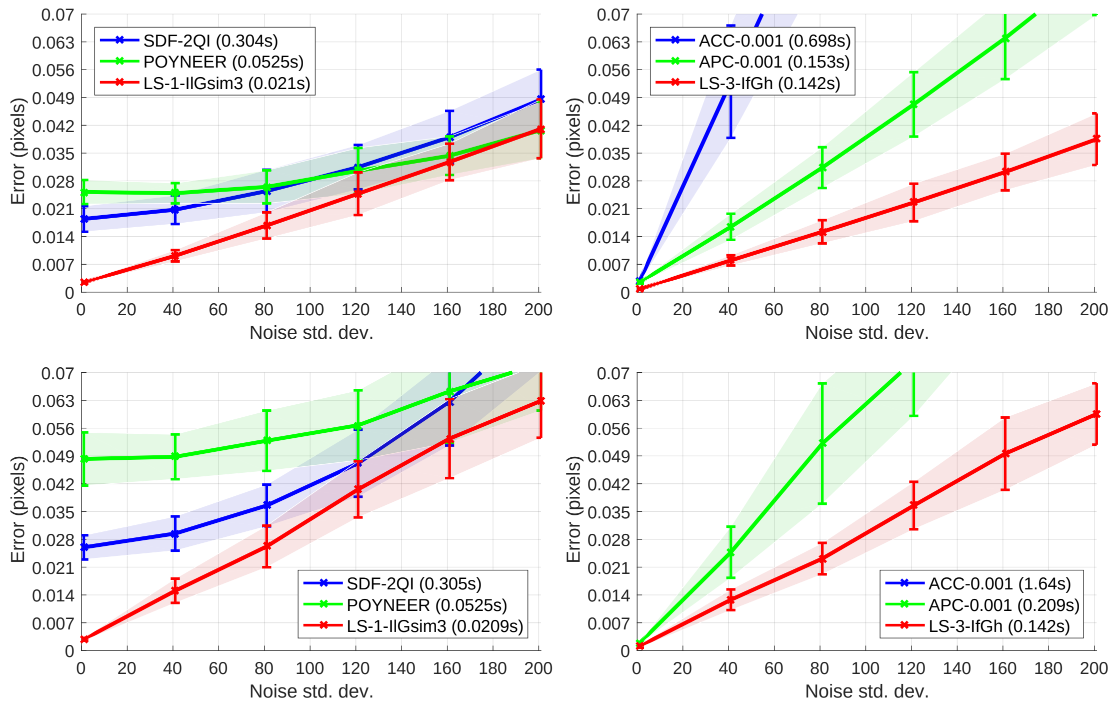


Figure 1.3 – Performance for non-iterative and iterative methods. **Top**: First test image. **Bottom**: Second test image. **Left**: Non-iterative methods. **Right**: Iterative methods.

More important wavefront aberrations translates to larger displacements to estimate in the SHWFS device. In this case, a multiscale approach was adopted for the GBSE method. Again, results in Fig. 1.4 show that our approach improves over the state-of-the-art methods.

Our contribution in this chapter is therefore threefold. We begin with a review of the state-of-the-art on wavefront correction using SHWFS on extended scenes. Second, to estimate the displacements, we propose the use of an iterative global GBSE method which presents several advantages over the conventional correlation methods. Finally, we propose a fast and effective method for scene preselection that adds almost no computational cost.

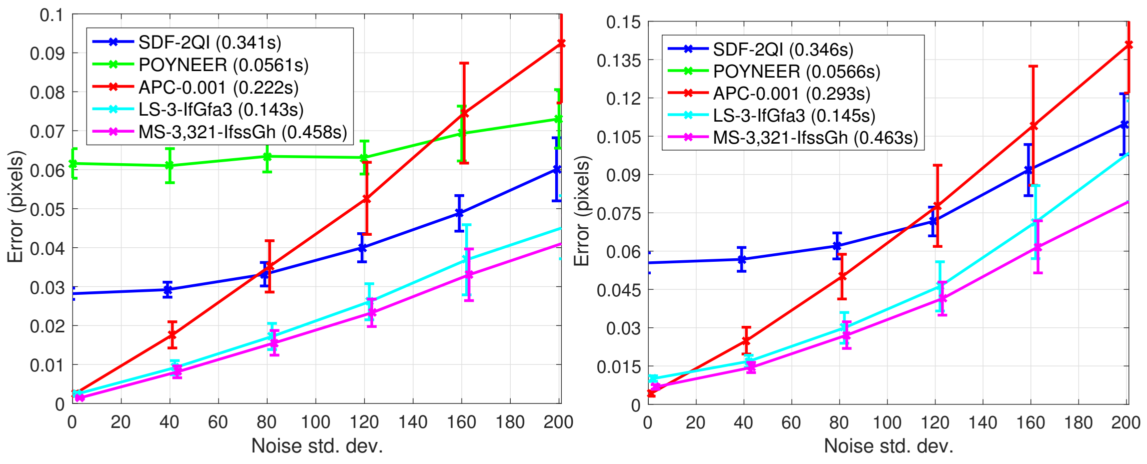


Figure 1.4 – Average error and standard deviations of selected methods and two variants of the GBSE approach when simulating wavefront aberrations yielding displacements of up to 4 pixels. **Left**: First test image. **Right**: Second test image.

1.3 Chapter 4: Stab-Active: stabilizing on board image accumulation

Push broom imagers placed on satellites are used to perform Earth observation at high resolution [19,59]. Because they are close to the Earth ($\approx$ 800 km), they move fast and the image acquisition time is very brief. The solution to avoid motion blur is to use Time Delay Integration (TDI) on the satellite's imaging sensor, which, by synchronizing moving pixels with the motion of the camera or the object, is able to increase the effective exposure time. This sensor works by shifting the partial measurements of its multiple rows to their adjacent rows synchronously with the motion of the image across the array of elements. This synchronous accumulation yields an SNR unobtainable using conventional CCD arrays or single-line-scan devices. However, it places stability constraints on the imaging device. To relax these constraints limiting the accumulation time, we evaluated if performing on board registration and accumulation was possible.

Envisaging the use of a CMOS TDI allows to correct the accumulation procedure by resampling the incoming signal. This permits to avoid motion blur while accumulating many more frames thus obtaining much higher SNRs. Yet this new imaging method raises new technical issues. A prerequisite is that the perturbations of the line of sight must be estimated very accurately. To this aim, little low resolution CMOS sensors set along this line of sight could sense the same landscape underneath the TDI's detection line, possibly under slight perturbations and with a low SNR [107]. The problem is then to perform subpixel image registration, up to the noise limit, of a set of consecutive low SNR images of the same landscape, known as a "TDI ROW" or simply "line". This becomes a non-trivial task due to different external conditions such as pointing errors and satellite vibration. Luckily at the studied time scales (> 500Hz), the satellite micro-vibrations can be considered a linear shift over the image sequence, requiring only to compute a global translation estimate for the whole line (see figures 4.1 and 4.2). Finally, due to limited hardware resources, the desired algorithm must be low demanding in both memory and processing speed.

The optimal bounds for performing shift estimation between two images also have a long history going back at least to 1983 [46]. We shall use their latest developments [139], [119], [146], [3] which will be extended here to a multi-image uniform translation. It so happens that the optimal registration bounds for an image sequence are much more accurate than with only two images, and the challenge is to reach these bounds.

In summary, the Stab Active problem reduces to performing accurate multi-image subpixel shift estimation between N images, where the shift between consecutive images could be assumed constant. What is more, this constant shift is lower than a tenth of a pixel and, due to extremely low exposure times, images have low signal to noise ratios. The requirement of doing this in real-time on-board using low-power satellite-grade hardware makes this an extremely challenging task where most methods had to be discarded beforehand. This study was tested on a simulated dataset provided by CNES containing thousands of image sequences (or lines) which were generated using four possible perturbations to the sequence.

Based on the review of shift estimation methods presented in the first chapter of this manuscript, and due to the limited time constraints, only two types of methods were considered, namely, differential (i.e. gradient-based) and phase-correlation methods. A priori, GBSE method seems the best suited approach due to its low computational cost and its accurate subpixel results. Indeed, phase correlation methods could improve over GBSE when used on images that are severely distorted, in either geometry or intensity [168], however this is not the current case. There were also other reasons that hindered the use of phase correlation approaches. Namely, the necessity to use image apodization which is prohibitive on small images; the fact that images could potentially have periodic patterns that yield several correlation peaks and finally, that the computational time savings by using the FFT are negligible due to the small image sizes dealt here. Nevertheless, we compare our results with those of a phase-correlation-based method [69] to prove our claims.

The key results from this chapter are:

- The introduction of a temporal convolution which proves mandatory to increase the SNR of input images thus helping GBSE methods to achieve more accurate results.
- A validation of the input image sequence by involving the Cramer-Rao Lower Bounds, which we derive here for the case of uniform translation in a multi-image scenario.
- By focusing on the GBSE method, we perform a study on its bias and discuss how to reduce its influence.
- An iterative variant of the Lucas-Kanade GBSE method for shift estimation, where we suggest to use a gradient-estimation method particularly suited for this task, which also requires fewer computational resources than traditional methods.
- A greedy multi-frame scheme to perform shift estimation that is particularly aimed at improving the global shift estimation using the proposed GBSE method. Indeed, this approach proved to improve the precision by up to two orders of magnitude, while reducing the overall computational cost. This could be observed from Table 1.1.
- As a result, four possible algorithms were proposed to CNES. These were designed based on the studied results and CNES' comments in order to be successfully implemented in the satellite. We described and implemented two low cost methods, a third one giving the best cost/performance relationship and a last one focusing on accuracy.
- Most importantly, a solution based on the proposed algorithms is currently being used by CNES in a on-ground demonstrator to register TDI images in real-time and is envisaged to be implemented in the next earth-observation satellites.

1.4 Chapter 5: RANSAAC: RANdom SAmpled Aggregated Consensus

In the other chapters of this thesis, we studied the problem of fast and accurate registration methods when the underlying transformation was a shift, and developed two on

#	Pert.	CRLB	Alg. 6	Alg. 7	Alg. 8	[159]	[69]	[7]
1	P0	0.0004	0.0199	0.0029	**0.0024**	0.0058	**0.0024**	0.0052
2	P1	0.0004	0.0128	**0.0028**	0.0155	0.1724	0.2336	0.0319
3	P2	0.0004	0.0494	**0.0033**	0.0064	0.3420	0.1633	0.0434
4(N)	P3	0.0006	0.3099	**0.0031**	0.0234	3.1045	0.0324	0.0504
5(N)	P3	0.0015	2.8468	**0.0153**	0.0400	1.4983	0.9852	0.0909
6	P0	0.0001	0.0022	0.0011	**0.0009**	0.0014	0.0011	0.0015
7	P1	0.0003	0.0203	**0.0068**	0.0365	0.2074	0.1686	0.0137
8	P2	0.0002	0.0085	**0.0017**	0.0205	1.3908	0.0044	0.0339
9	P3	0.0002	0.0146	**0.0010**	0.0106	3.7202	0.1477	0.0343
10(ON)	P1	0.0019	-	-	-	-	-	-
11(ON)	P1	0.0021	-	-	-	-	-	-
12(PF)	P2	0.0003	0.1244	**0.0036**	0.0506	3.0708	5.2640	0.4162
13(PF)	P2	0.0003	0.1543	**0.0019**	0.0505	2.9960	2.4721	0.2273
Avg.	-	0.0007	0.3239	**0.0040**	0.0234	1.5009	0.8613	0.0862

Table 1.1 – RMSE of each method for some meaningful image sequences. "Pert" refers to the perturbation simulated in the image sequence. CRLB indicates the lower bounds calculated. Alg. 6 returns the average of all shifts estimated from consecutive images. The proposed greedy Alg. 7 estimates the global unique shift by always computing the shift with respect to the first image, which indirectly reduces the bias existing in the used GBSE method. Alg. 8 estimates the shift directly between the first and the last frame, ignoring intermediate images. The rest of the compared approaches are state-of-the-art phase-correlation-based shift estimation methods.

board applications to earth-observation satellites. The presented approaches assumed small displacements between the input images. In this chapter we study the problem of image registration under more complex models. In particular, we focus on feature-based approaches using point matches.

Several applications in computer vision such as image alignment, panoramic mosaics, 3D reconstruction, motion tracking, object recognition, among others, are performed by feature-based approaches. First, characteristic image features are detected on the input images. They can be points with some special characteristic or distinguished image regions. Undoubtedly, SIFT points from Lowe [96] and MSERs from Matas *et al.* [105, 106] are the most recognized feature detectors of each category respectively. The second step is to assign to each feature a unique description. In the case of points, this is done by describing its surroundings [16, 96, 110]. For regions, their silhouettes and the contained texture information are employed [18, 58]. Finally, the descriptors of both images are matched to compute putative matches between them. Ideally, these matches correspond to samples of the underlying model to be estimated. Unluckily, not all detected matches are faithful to the global model, yielding false "clues" to the transformation estimation method.

As explained above, when estimating a single global transformation based on multiple point matches between images, existing methods have to deal with the problem of outliers, i.e., incorrectly detected matches that are not represented by the transformation. Robust to outliers, the Random Sample Consensus (RANSAC) is an iterative method introduced by Fischler and Bolles [55], widely used in computer vision to simultaneously solve the correspondence problem while estimating the implicit global transformation. By randomly generating hypotheses on the transform matching the points, it tries to achieve a maximum consensus in the input dataset in order to deduce the matching points coherent with the transformation, called inliers. RANSAC considers a match to be inlier if the distance between the projected source point and its matching point is lower than a (usually difficult to define) distance parameter δ_d. Once the inliers are discriminated, the parameters of the underlying transformation are estimated using a regression technique on the inliers.

To avoid outliers, instead of using every sample in the dataset to perform the estimation as in traditional regression techniques, RANSAC tests in turn many random sets of sample pairs. RANSAC takes the minimum sample size (MSS) to determine a unique candidate transform, thus incrementing its chances of finding an "all-inlier" sample, i.e., an uncontaminated sample exclusively composed of inliers. This transform is assigned a score based on the cardinality of its consensus set. Finally, the hypothesis that yielded the highest score is saved.

RANSAC has several drawbacks. First, the probability that RANSAC obtains a reasonable result increases with the number of iterations, however it may never reach the optimal solution. What is more, RANSAC results have a high degree of variability for the same input data, and this variability increases with the number of input points and their measurement noise. Second, although robust to outliers, RANSAC is not particularly immune to measurement noise on the input data, as illustrated on Fig. 5.1. Furthermore, the optional final minimization step weighs uniformly the assumed inlier match, ignoring wether they are real inliers or how strongly they may be affected by noise. Third, the maximum tolerable distance parameter δ_d should be sufficiently tight to obtain a precise transformation, however it must also be loose to find enough input samples [33]. Because of such considerations, setting this parameter is a difficult task, even under low measurement noise. Finally and most importantly, **the accuracy of RANSAC is based on the single sample which gave the maximal consensus**. Although this choice may be accurate in some cases, it nevertheless discards other good all-inlier models that may have been generated throughout the iterations.

To make up for this, we present Random Sample Aggregated Consensus (RANSAAC), a simple yet powerful method combining the random sample consensus scheme with a statistical approach. By aggregating the random hypotheses using their consensus set cardinalities, the proposed approach improves systematically on RANSAC. We give practical implementations of this idea on 2D parametric transformation models, by proposing a simple strategy to drastically accelerate the computations. This strategy is based on predefining a MSS of points located on the image extents of the source image, followed by projecting them for every generated hypothesis onto the second image, and finally aggregating independently the projections for each point using the obtained RANSAC scores to obtain a final MSS that can be used to generate the final transformation. This procedure is illustrated in Fig. 1.5, showing the results of several aggregation methods.

Its main benefits over traditional RANSAC are:

- Results are both more accurate and with less variability (in terms of standard deviation of the error). The improvement over the traditional RANSAC approach is on average by a factor of two to three, and is even more important with higher noise, more inliers available, and higher outlier ratios. Moreover, it improves systematically over other state-of-the-art RANSAC extensions.
- The accuracy improvement persists after including the final regression step in RANSAC and its variants, even without using a last-step optimization in RANSAAC.
- As with the original RANSAC method, the accuracy is dramatically improved by adding a local optimization step, which in fact seems suited for our approach because it avoids discarding the generated intermediate models.
- Also, by including this step, the theoretical adaptative stopping criterion becomes more realistic and can be used to effectively stop the iterations without affecting the final accuracy.
- By using the proposed 2D transformation averaging method, the extra computational cost is almost negligible.

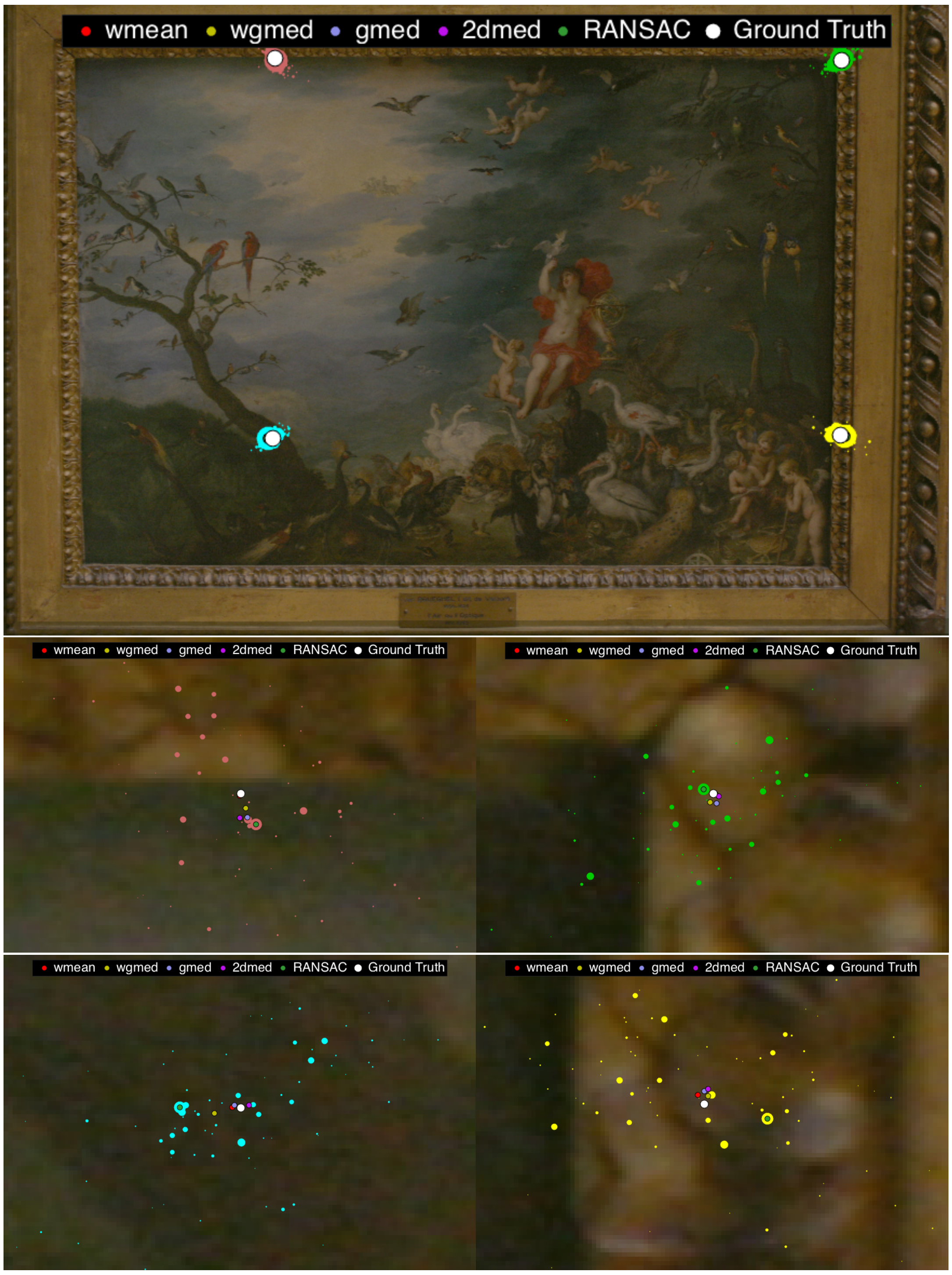
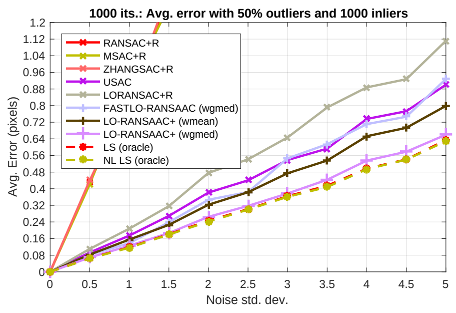


Figure 1.5 – Resulting point distributions resized according to their weights and zoom in on each one of the four corners for 1000 iterations. For visualization purposes, points with low weights were not included. Notation: $wmean$ and $wgmed$ are the weighted mean and the weighted geometric median aggregation respectively. $gmed$ and $2dmed$ represent aggregated results using (unweighted) geometric median and 1D median on both dimensions respectively. The reader is advised to zoom in to see better the positions of the various estimates.

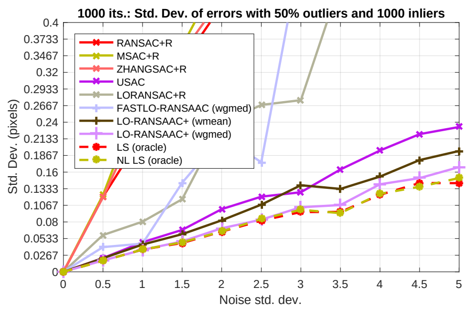
- Robustness against measurement noise is drastically improved for the case of noise distributions with symmetrical pdfs.

In Fig. 1.6 we observe the performance by varying noise conditions of the proposed approach compared to state-of-the-art methods under different inlier/outlier conditions. All compared methods include the last-step least squares minimization. Note that the USAC method is not able to return any results for the case of 75% outliers.

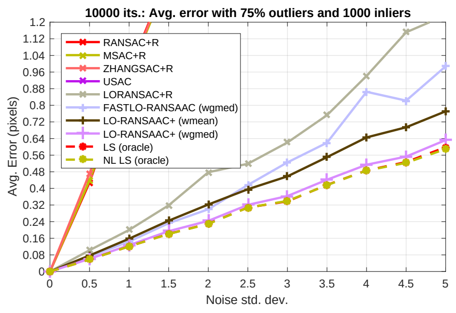
Indeed one of the most remarkable properties of RANSAAC is its robustness against very high outlier ratios. Results proving this are shown in Table 1.2 where we tested the best methods under 90% outliers. One can observe that the straightforward adaptation of the local optimization method into the RANSAAC framework gets results close to what would have been obtained if an oracle that discriminates the inliers had been used, in a scenario where the state-of-the-art USAC method was not able to return any result.



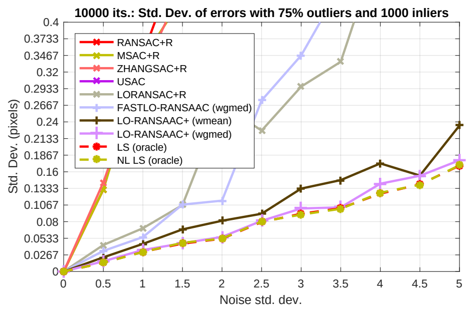
$\bar{E}$: 1000 its with 1000/2000 inlier ratio.



σ_E: 1000 its with 1000/2000 inlier ratio.



$\bar{E}$: 10000 its with 1000/4000 inlier ratio.



σ_E: 10000 its with 1000/4000 inlier ratio.

Figure 1.6 – Avg. errors $\bar{E}$ and their standard deviations σ_E by varying noise for several RANSAC and RANSAAC variants. Experiment: 1000 inliers, different amounts of outliers and doing 1000/10000 iterations. **First row**: 50% outliers. **Bottom row**: 75% outliers and 10k iterations.

σ	LO+ wmean		LO+ wgmed		RANSAC+LS		LO-RANSAC		USAC		LS (oracle)
	10k	20k	10k	20k	10k	20k	10k	20k	10k	20k	-
2	0.53	0.46	**0.36**	**0.31**	49.15	19.90	16.68	1.76	–	–	**0.28**
5	1.35	1.38	**0.94**	**1.15**	23.94	27.69	4.35	6.30	–	–	**0.74**

Table 1.2 – High outlier ratio test: average errors of both aggregation methods for RANSAAC and LO-RANSAAC+, compared with RANSAC with last step least squares minimization (DLT), LO-RANSAC, USAC and computing LS on the inliers with an oracle. For noises $\sigma = 2$ and 5, each method was evaluated with 1000 inliers using both 10000 and 20000 iterations and 90% outliers. The averages are over 50 realizations.

1.5 Summary of contributions

- An in depth study of fast and accurate shift estimation methods, together with a practical recipe indicating when and how to use each approach.
- A new method for shift estimation in the context of Shack-Hartmann Wavefront Sensors used on extended scenes. Using few resources, this method is suitable to be used on board of future earth-observation satellites, and competes with phase-diversity methods for a satellite implementation.
- An accurate multi-image shift estimation method used for image stabilization of CMOS TDI sensors to increase the SNR of Earth satellite images. The proposed method has already been tested by CNES in an on-ground demonstrator to register TDI images and is a strong candidate to be implemented in the next generation of earth-observation satellites.
- A simple but effective modification to the RANSAC algorithm that improves over the state-of-the-art methods by combining the random consensus idea of testing samples with minimal cardinality with a statistical approach performing an aggregation of estimates.
- During this thesis, the author collaborated with Nicola Pierazzo on image denoising. This resulted in
  - a method that by factorizing the patch space, achieved speeding up the method of Levin and Nadler [93] that search for the absolute limits of patch-based image denoising, by a factor of a thousand while maintaining the theoretical claim of optimality.
  - the NLDD algorithm, that improved over the recent DDID method [84] by using it as a post-process denoising approach, combining it with NL-Bayes or BM3D, two state-of-the-art denoising methods.
  - the DA3D method, which also works as a last-step denoising by taking as input the results from other denoising algorithm, and almost always improve them, both in terms of PSNR and particularly with respect to visual quality.
- The author also collaborated with the University of Buenos Aires under the Stic-AmSud program of scientific cooperation. The result of this cooperation was a work on Audio processing that evaluated several features extracted from the audio in order to gain recognition between music, voice or other sounds coming from the environment.

1.6 List of publications

The work in this thesis has led to the following publications:

- Pierazzo, N.; Rais, M., "Boosting “shotgun denoising” by patch normalization" in Image Processing (ICIP), 2013 20th IEEE International Conference on , vol., no., pp.1115-1119, 15-18 Sept. 2013. doi: 10.1109/ICIP.2013.6738230[1].
- Rais, M.; Thiebaut, C.; Delvit, J.-M. and Morel, J.-M., "A tight multiframe registration problem with application to Earth observation satellite design" in Imaging Systems and Techniques (IST), 2014 IEEE International Conference on , vol., no., pp.6-10, 14-17 Oct. 2014. doi: 10.1109/IST.2014.6958436[2].
- Pierazzo, N.; Lebrun, M.; Rais, M.E.; Morel, J.M. and Facciolo, G., "Non-local dual image denoising" in Image Processing (ICIP), 2014 IEEE International Conference on, pp.813-817, 27-30 Oct. 2014. **Top 10% Paper Award.**. doi: 10.1109/ICIP.2014.7025163[3].
- Bengolea, G.; Acevedo, D.; Rais, M.; and Mejail, M., "Feature Analysis for Audio Classification", Lecture Notes in Computer Science, Volume 7042/2014, 239-246, Springer, 2014. doi: 10.1007/978-3-319-12568-8_30[4].
- Rais, M.; Morel, J.-M., Thiebaut, C. and Delvit, J.-M. Stabilization Improvements using optical flows. Minisymposia: Inverse problems in space imaging. Applied Inverse Problems (AIP), May. 2015.
- Pierazzo, N.; Rais, M.; Morel J.-M. and Facciolo, G., “DA3D: Fast and Data Adaptive Dual Domaing Denoising” in Image Processing (ICIP), 2015 IEEE International Conference on, Oct. 2015. doi: 10.1109/ICIP.2015.7350835[5].
- Rais, M.; Morel, J.-M. and Facciolo, G. (2015), Iterative Gradient-Based Shift Estimation: To Multiscale or Not to Multiscale?, in CIARP, volume 9423 of Lecture Notes in Computer Science, page 416-423. Springer, 2015. doi: 10.1007/978-3-319-25751-8_50[6].
- Rais, M., Morel, J.-M., Thiebaut, C.; Delvit, J.-M. and Facciolo, G., "Improving the accuracy of a Shack-Hartmann wavefront sensor on extended scenes", 6th International Workshop on New Computational Methods for Inverse Problems (NCMIP), 2016.
- Rais, M.; Morel, J.-M., Thiebaut, C.; Delvit, J.-M. and Facciolo, G., "Improving wavefront sensing with a Shack-Hartmann device". Accepted in Applied Optics, Optical Society of America, Aug. 2016.
- Rais, M.; Morel, J.-M.; Buades, A.; Coll-Vicens, B.; Meinhardt, E. and Facciolo, G., "Accurate 2D Transformation Estimation through Random Sampled Hypotheses Aggregation". To be submitted.
- Rais, M.; Morel, J.-M.; Facciolo, G., "Review of Fast and Acccurate Subpixel Shift Estimation Methods". To be submitted.
- Rais, M.; Morel, J.-M.; Buades, A., "How to obtain high definition images of paintings using a conventional camera". To be submitted.
- Rais, M.; Morel, J.-M.; Meinhardt, E. and Facciolo, G., "Improving epipolar geometry estimation through Random Sample Aggregated Consensus". To be submitted.

[1] http://dx.doi.org/10.1109/ICIP.2013.6738230
[2] http://dx.doi.org/10.1109/IST.2014.6958436
[3] http://dx.doi.org/10.1109/ICIP.2014.7025163
[4] http://dx.doi.org/10.1007/978-3-319-12568-8_30
[5] http://dx.doi.org/10.1109/ICIP.2015.7350835
[6] http://dx.doi.org/10.1007/978-3-319-25751-8_50

Chapter 2

Review of Global Subpixel Shift Estimation Methods

The problem of global subpixel shift estimation appears frequently on several applications related to image processing. When high accuracy is not required, most authors tend to use the well-known cross-correlation method which usually meets the performance requirements for such tasks. However, under more demanding accuracy constraints, cross-correlation approaches are either not accurate enough or their computational costs are prohibitive. Moreover, problems such as time constraints, varying illumination conditions, aliasing, among several others, make the shift estimation task harder. What is worse, there is little evidence about which shift estimation method to use depending on the circumstances. Our objective in this chapter is therefore to give an in depth study on shift estimation methods, offering a practical recipe showing which methods are more suited depending on the conditions of the task. By discretizing the distinct varying conditions, we evaluate on each of them more than a thousand variants of shift estimation methods, and summarize the results here. We observed that gradient-based shift estimation methods, used in a multi-scale and iterative form, achieve in general the most accurate results, without requiring considerable computational resources. This analysis is of key importance for the rest of this thesis, particularly for Chapters 3 and 4, where two remote sensing applications arising from earth-observation satellites are studied.

2.1 Introduction

In this chapter we study the problem of global sub-pixel shift estimation between images. Given two observations I_1, I_2 of the same image $I(x, y)$, shifted by an unknown displacement $\mathbf{v} = \{v_x, v_y\}$ and affected by noise, the problem of shift estimation is to find this displacement so as to align both images. Applications for shift estimation include medical image registration [76, 91, 101], motion tracking [74], microscopic biology [83], digital image stabilization [51], 3D reconstruction [114], video analysis [9], and are particularly developed in the field of remote sensing [67, 90, 130, 145].

The shift estimation problem presents several challenges. The conditions for this problem to be solved may vary depending on the sensor, the underlying noise model, the effective image size, the magnitude of the displacement, varying illumination between frames, occlusions, aliasing, among others, making shift estimation methods prone to considerable errors depending on their underlying assumptions. Furthermore, the shift estimation task is usually part of a more complex image processing application, which usually constrains its execution time, forcing the shift estimation method to be both fast and accurate.

In this chapter we tackle the case where both images I_1 and I_2 have the same $N \times N$ size, are both contaminated by white Gaussian noise with the same standard deviation σ. We always assume that the 2D displacement is smaller than one pixel in both dimensions and that the brightness constancy constrain holds, i.e.,

$$I_2(x, y) = I_1(x + v_x, y + v_y). \tag{2.1}$$

Observing the behavior of each method under these ideal conditions is sufficient since most problems mentioned above can be reduced to this setting. In particular,

- **Signal dependent noise.** Heteroscedastic noise can be turned homoscedastic by prefiltering the images with a Variance Stabilization Transform (VST) [158]. For example, the well-known Anscombe transform [6], as well as the Generalized Anscombe transform convert Poisson or Poisson + Gaussian noise into approximately standard Gaussian noise making the standard deviation almost constant [158]. Other well-known variance stabilization transformations include the Freeman-Tukey [61], Barlett [15], Curtiss [40] and Eisenhart [48]. In fact, there is a whole family of VSTs for the Poisson distribution, described by Bar-Lev and Enis [14].

- **Varying illumination between frames.** A histogram matching method should be performed previous to the shift estimation. In many contexts, such as satellite imaging, the change in illumination is multiplicative, therefore a fast multiplicative mean equalization should be done. For non-linear changes in illumination, images can be equalized by using the midway equalization method [43]. We refer the reader to further histogram matching approaches [66, 100].

- **Different image sizes.** In template matching applications [26], an image patch of size $N \times N$ has to be aligned with a template or reference image of size $M \times M$ with $N < M$. This is in general a less challenging problem due to the availability of image borders on the reference image. In such situations, any fast method achieving pixel precision [2, 120] could be applied in order to extract a $N \times N$ subimage from the reference, followed by applying a subpixel shift estimation method.

- **Large shift magnitudes.** As in the previous case, large shift magnitudes could be estimated in two steps. After performing an initial pixel-wise shift estimation, any of the methods discussed in this chapter could be used on the intersection of both pixel-aligned images. This includes projection-based methods [2, 120], pyramidal multi-scale approaches [131, 167] or fast periodic cross correlation methods in the Fourier domain [124].

While several methods exist up to date, not many achieve acceptable results on every possible condition. To our understanding, several survey papers exist summarizing these methods [24,101,168,191]. However, none of them offer a performance comparison under the different challenging conditions that may occur, which ends up with the eventual user to pick the wrong method. The contribution of this chapter is therefore twofold. First, an in-depth study of the state-of-the-art shift estimation methods is given, followed by a practical recipe defining which methods to use depending on the particular conditioning of the problem.

2.1.1 Subpixel shift estimation approaches

As mentioned in Tian *et al.* [168], there are mainly four types of shift estimation methods that achieve subpixel accuracy: correlation interpolation, intensity interpolation, differential methods and phase correlation. Recently, feature-based image registration has become extremely popular since the introduction of local feature detection/description methods.

Correlation Interpolation Methods. In order to achieve subpixel accuracy using discrete correlation, an interpolation surface is fitted to samples of this function, and then, the maximum of this surface is searched. When the images are sampled at a high enough frequency, the corresponding discrete correlation function is quite smooth and using a second-order interpolation function can obtain accurate results. This methodology not only implies calculating the discrete correlation between images, which is a resource consuming task, but also to interpolate it.

Intensity Interpolation Methods. Another way to achieve subpixel results is to interpolate selected parts of the input images in order to create a much denser grid. Then, the task is to match these grids between images. This task requires knowing beforehand, which part of the input images to interpolate and then match, something which is not always available. Several searches have to be made in order to find good matches, making this approach time consuming.

Differential Methods. Also known as Gradient-Based Shift Estimation (GBSE), the idea behind this methodology is to relate the difference between two successive frames to the spatial intensity gradient of the first image. This relationship is obtained by approximating the shifted image by its Taylor development up to the first order. Since the higher terms of the Taylor approximation are removed, this relation holds only when the displacement is shorter than a pixel. By performing such an approximation, the shift can then be easily estimated using linear least squares minimization. However, more elaborate methods can be applied to achieve higher precision. This type of methods have linear complexity and are in general considerably faster than both interpolation-based methods. Furthermore, several extensions allow it to become more robust against noise and larger displacements.

Phase correlation Methods. This methodology is based on the fact that most of the information about the relative displacements of objects between two images is contained in the phase of their cross-power spectrum. Several methods exploit this to estimate the displacement both in the Fourier and the spatial domain. These methods require the computation of the discrete Fourier transform (DFT), which could be prohibitive in certain scenarios. Nevertheless, they are able to achieve very accurate results without further computational effort, and they can be used when images are seriously distorted, in either geometry or intensity. Due to this reason, several methods based on this idea have been recently published [7, 8, 57, 69, 132, 134, 135, 159, 162, 178, 179]. As a drawback, phase correlation approaches assume circular shifts between images, i.e. $I_1(x, y) = I_2((x - v_x) \bmod M, (y - v_y) \bmod N)$ (where the images are $M \times N$ in size). In fact, the cross-power spectrum in the frequency domain is the analog of the cross-correlation operator between two signals. Since in general, the shift between both images is a simple linear shift rather than a circular shift, windowing functions are usually employed to mitigate problems related to image edges, by performing apodization of the input images. For images that have repeating objects, phase correlation may also yield ambiguous results with several correlation peaks, and its use on those cases is not advisable.

Feature Based Methods. This approach, as pointed out by [191], is based on the extraction of salient structures or features, that are supposed to be stable. Then, these features are matched between images by either comparing feature descriptors, or similarity measures combined with spatial relationships. Once the matches are calculated, a transformation model is estimated in order to align both images, using the computed matches. The idea for the features is to be distinct, spread all over the image, and efficiently detectable in both images. They are expected to be stable in time to stay at fixed positions during the whole experiment. One algorithm used to describe and match features is the famous SIFT method proposed by Lowe [96]. It gets subpixel accuracy of the features by interpolating the response of the estimated Laplacian over the pixels.

The stability of the features decreases in the presence of noise and in consequence poor registration results are obtained. What is more, although these methods are efficient when they work on a multiscale approach, they are not aimed at achieving accurate subpixel measurements and because of this reason, are not considered in this chapter.

However, in the last chapter of this thesis, we explore this approach and propose a new method that is able to obtain precise models, even with large amounts of incorrectly detected point matches.

2.1.2 Chapter summary

This chapter offers a detailed analysis on fast and accurate shift estimation methods. With few exceptions, we discard every shift estimation method that augments the images resolutions using interpolation to reach sub-pixel accuracy. Although these methods may in practice achieve accurate results, they are computationally expensive. For this reason, in this chapter we focus only on differential methods and phase correlation approaches, offering a complete study on both methodologies.

We observed that gradient-based shift estimation (GBSE) approaches seem to be more accurate and tolerant to noise than phase-correlation methods. They are also computationally cheaper and more stable. However, when the estimated shifts are large, they must be used within pyramidal multiscale approaches which tend to reduce their stability. Although more computationally intensive, gradient correlation methods offer more accurate and consistent results than phase correlation approaches. Their accuracy suffers less under lower SNR scenarios. We refer the reader directly to the conclusions yielded from sections 2.4.2, 2.4.4, 2.4.5 and 2.4.6 for a brief summary of this chapter.

We begin this chapter by giving an overview of the most important fast and accurate shift estimation methods, starting with gradient-based approaches (section 2.2) followed by phase-correlation-based methods (section 2.3). We perform a full evaluation of the analyzed methods under different challenging situations in section 2.4 and give a practical recipe by giving the best approaches to use under those. We conclude and offer some envisaged future work in section 2.5.

2.2 Gradient-Based Shift Estimation methods

Given two images where the brightness constraint holds, namely $I_2(x,y) = I_1(x+v_x, y+v_y)$ where v_x and v_y are the unknown shift coefficients, then using the first order Taylor approximation and assuming the displacement to be small, we have that

$$\begin{aligned} I_2(x,y) - I_1(x,y) &\approx v_x \frac{\partial I_1(x,y)}{\partial x} + v_y \frac{\partial I_1(x,y)}{\partial y}, \\ I_t(x,y) &\approx \nabla I_1(x,y)^T \mathbf{v} \end{aligned} \tag{2.2}$$

where $I_t = I_2 - I_1$ is the discrete temporal derivative, $\nabla I_1(x,y) = \left[\frac{\partial I_1(x,y)}{\partial x}, \frac{\partial I_1(x,y)}{\partial y}\right]$ the image gradient and $\mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}$ corresponds to the unknown shift.

Note that even though in practice the brightness constraint rarely holds exactly, it works remarkably well in real-life applications [56]. Eq. (2.2) relates the difference between the two successive frames to the spatial intensity gradient of the first image for a single pixel. This equation is known as the optical flow equation or the gradient constraint equation. Since there is one equation and two unknowns, the shift cannot be determined, therefore yielding the necessity to add additional constraints. In an optical flow application, the flow for each pixel may be different, therefore many variational approaches based on different possible constraints exist [17, 29, 186]. However, for the shift estimation problem between two images, since all pixels supposedly share the same displacement, each (x,y) location becomes an equation and the problem becomes overdetermined.

2.2.1 Optical Flow equation with Least Squares minimization

The Lucas-Kanade algorithm [97] is probably the most widely known gradient-based method used to estimate the optical flow between two images. Based on Eq. (2.2), it assumes a constant displacement for every pixel around its neighborhood, which allows the construction of an overdetermined system $\mathbf{A}\mathbf{v} = \mathbf{b}$, where $\mathbf{A}$ is composed of spatial intensity derivatives and $\mathbf{b}$ has temporal derivatives

$$\mathbf{A} = \begin{bmatrix} \frac{\partial I_1}{\partial x}(p_1) & \frac{\partial I_1}{\partial y}(p_1) \\ \vdots & \vdots \\ \frac{\partial I_1}{\partial x}(p_N) & \frac{\partial I_1}{\partial y}(p_N) \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} I_t(p_1) \\ \vdots \\ I_t(p_N) \end{bmatrix} \tag{2.3}$$

and p_i with $i = 1 \dots N$ represents the ith pixel and N the number of pixels. The solution $\mathbf{v}$ to this overdetermined system is obtained by performing the linear least squares method, using the Moore-Penrose pseudo-inverse. Then, let I_x, I_y be an abbreviation for $\frac{\partial I_1}{\partial x}$ and $\frac{\partial I_1}{\partial y}$ respectively, the shift is computed by

$$\hat{\mathbf{v}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{2.4}$$

where

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{2.5}$$

is the second moment matrix, and

$$\mathbf{A}^T\mathbf{b} = \begin{bmatrix} \sum I_t I_x \\ \sum I_t I_y \end{bmatrix} \tag{2.6}$$

is a spatio-temporal gradient correlation term. To solve this system, the matrix $\mathbf{A}^T\mathbf{A}$ must be invertible. Although this method was designed to compute the optical flow between two images, this same idea could be directly used as a shift estimation method by simply considering the neighborhood of each pixel as the whole image.

It is not a coincidence that the results of the method depend on the inversion of this second moment matrix since, as it will be shown later in this thesis, the determinant of this matrix is crucial for determining the limits on the estimation performance. A study on this matrix before actually performing the shift estimation could potentially be used to discard ill-posed cases, in which, for example, the gradient occurs on its majority on a single direction and thus we are dealing with a potentially unsolvable situation, known as the aperture problem.

Note that, since the Taylor development is centered at zero, this method performs well only when the translation is subpixel, i.e., estimated shifts larger than 1 would not be correctly estimated. For dealing with cases in which the shift is larger, the estimation must be recursively performed on zoomed-out versions of both images, followed by subsampling on the next scale. This method will be addressed in section 2.2.3.

2.2.2 Optical Flow Equation with Total Least Squares Minimization

To solve the optical flow equation (2.2), the least squares (LS) method can be employed yielding accurate results by using a fast closed form solution. It allows for noise in the independent term $\mathbf{b}$, yet it does not take into account both the noise in the spatial derivative matrix $\mathbf{A}$, coming from the image acquisition and sampling processes, as well as the fact that the derivatives themselves are also approximations (i.e. estimated as differences between neighboring pixels).

In order to deal with noise in the spatial derivatives and to account for approximation errors, an improved technique, named total least squares (TLS) [80,103], can be used. This method assumes perturbations both in the temporal derivative (the independent term $\mathbf{b}$) as well as in the spatial derivatives of the first image (matrix $\mathbf{A}$). To explain this in a more intuitive way, we shall rewrite first the Least Squares (LS) problem as

$$\{\hat{\mathbf{x}}_{LS}, \Delta\mathbf{b}_{LS}\} := \arg\min_{\mathbf{x},\Delta\mathbf{b}} \|\Delta\mathbf{b}\|_2 \quad \text{subject to } \mathbf{Ax} = \mathbf{b} + \Delta\mathbf{b}, \tag{2.7}$$

where the idea is to allow for an error in the independent term $\mathbf{b}$ by minimizing over $\Delta\mathbf{b}$, so that the corrected system of equations $A\mathbf{x} = \hat{\mathbf{b}}, \hat{\mathbf{b}} := \mathbf{b} + \Delta\mathbf{b}$ has an exact solution. Then if $\mathbf{A}^T\mathbf{A}$ is invertible, the unique solution $\mathbf{x}_{LS} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^t\mathbf{b}$ of the optimally corrected system of equations $\mathbf{Ax} = \hat{\mathbf{b}}_{LS}, \hat{\mathbf{b}}_{LS} := \mathbf{b} + \Delta\mathbf{b}_{LS}$ is by definition the least squares approximate solution of the original incompatible system of equations.

As can be seen from the above example, $\mathbf{b}$ is corrected while $\mathbf{A}$ is not, thus ignoring possible errors and noise in the calculation of the derivatives. To deal with them, the total least squares method assumes uncorrelated noise in both the independent term $\mathbf{b}$ as well as in the spatial derivative matrix $\mathbf{A}$. Indeed, total least squares assumes the error in each element of matrix $\mathbf{A}$ and the vector $\mathbf{b}$ to be independent and identically distributed (i.e. the error matrix is white), while performing no assumptions on the distribution of the noise. If however the noise turns out to be correlated, total least squares can actually perform worse than standard least squares. However, TLS finds the true solution when the amount of optical flow equations N goes to infinity, and it out-performs LS estimation as N is sufficiently large [180].

It is well-known that the least-squares approximation is statistically motivated as a maximum likelihood estimator in a linear regression model under zero mean and a normally distributed residual with a covariance matrix that is a multiple of the identity [72]. Similarly, the total least-squares approximation is a maximum likelihood estimator in the errors-in-variables (EIV) model, namely

$$\tilde{\mathbf{A}} = \mathbf{A} + \Delta\tilde{\mathbf{A}}, \quad \tilde{\mathbf{b}} = \mathbf{b} + \Delta\tilde{\mathbf{b}}, \quad \exists\mathbf{x} \text{ such that } \mathbf{Ax} = \mathbf{b}. \tag{2.8}$$

under the assumption that the augmented matrix $[\Delta\tilde{\mathbf{A}}|\Delta\tilde{\mathbf{b}}]$ is a zero mean, normally distributed random vector with a covariance matrix that is a multiple of the identity [103].

Theorem 1. *The solution to the total least squares problem consisting in calculating*

$$\{\hat{\mathbf{x}}_{TLS}, \Delta\mathbf{A}_{TLS}, \Delta\mathbf{b}_{TLS}\} := \underset{\mathbf{x},\Delta\mathbf{A},\Delta\mathbf{b}}{\operatorname{argmin}} \|[\Delta\mathbf{A}\ \Delta\mathbf{b}]\|_F \quad \textit{subject to } (\mathbf{A} + \Delta\mathbf{A})\mathbf{x} = \mathbf{b} + \Delta\mathbf{b} \tag{2.9}$$

where $\|[\Delta\mathbf{A}\ \Delta\mathbf{b}]\|_F$ is the Frobenius norm of the augmented matrix with matrix $\Delta\mathbf{A}$ and the vector $\Delta\mathbf{b}$ side by side, is given by

$$\hat{\mathbf{x}}_{TLS} = -\frac{(v_1, v_2)^T}{v_3} \tag{2.10}$$

where $V = (v_1, v_2, v_3)^T$ is the 3×1 right singular vector associated with the smallest singular value of the augmented matrix $[\mathbf{A}|\mathbf{b}]$.

Proof. Minimizing Eq. (2.9) is the same as minimizing $\|[\Delta\mathbf{A}\ \Delta\mathbf{b}]\|_F$ subject to $\mathbf{b} + \Delta\mathbf{b} \in Range(\mathbf{A} + \Delta\mathbf{A})$. Once the minimum $(\widehat{\Delta\mathbf{A}}, \widehat{\Delta\mathbf{b}})$ is found, any x satisfying

$$(\mathbf{A} + \widehat{\Delta\mathbf{A}})x = \mathbf{b} + \widehat{\Delta\mathbf{b}} \tag{2.11}$$

is said to solve the TLS problem. To seek for a solution, we can generalize the problem for $\mathbf{b}$ to be a matrix $\mathbf{B}$ of $n \times k$ elements. Then Eq. (2.11) can be rewritten as

$$[(\mathbf{A} + \Delta\mathbf{A})\ (\mathbf{B} + \Delta\mathbf{B})] \begin{bmatrix} x \\ -I_k \end{bmatrix} = 0, \tag{2.12}$$

where I_k is the $k \times k$ identity matrix. We shall now show that solving the problem is equivalent to finding $[\Delta\mathbf{A}\ \Delta\mathbf{B}]$ that reduces the rank of $[\mathbf{A}\ \mathbf{B}]$ by k. Define $[U][\Sigma][V]^*$ to be the singular value decomposition of the augmented matrix $[\mathbf{A}\ \mathbf{B}]$

$$[\mathbf{A}\ \mathbf{B}] = [U_A\ U_B] \begin{bmatrix} \Sigma_A & 0 \\ 0 & \Sigma_B \end{bmatrix} \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}^*, \tag{2.13}$$

where V is partitioned into blocks corresponding to the shape of $\mathbf{A}$ and $\mathbf{B}$. Using the Eckart-Young-Mirsky theorem [47,65], the approximation minimizing the Frobenius norm of the error in Eq. (2.9) is such that matrices U and V are unchanged, while the k-smallest singular values are replaced with zeroes. That is

$$[(A + \Delta\mathbf{A})\ (B + \Delta\mathbf{B})] = [U_A\ U_B] \begin{bmatrix} \Sigma_A & 0 \\ 0 & 0_{k\times k} \end{bmatrix} \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}^*, \tag{2.14}$$

so by substracting Eq. (2.13) with Eq. (2.14), we get

$$[\Delta\mathbf{A}\ \Delta\mathbf{B}] = -[U_A\ U_B] \begin{bmatrix} 0_{n\times n} & 0 \\ 0 & \Sigma_B \end{bmatrix} \begin{bmatrix} V_{AA} & V_{AB} \\ V_{BA} & V_{BB} \end{bmatrix}^*. \tag{2.15}$$

We can then remove blocks from the U and Σ matrices, simplifying to

$$[\Delta\mathbf{A}\ \Delta\mathbf{B}] = -U_B\Sigma_B \begin{bmatrix} V_{AB} \\ V_{BB} \end{bmatrix}^* = -[A\ B] \begin{bmatrix} V_{AB} \\ V_{BB} \end{bmatrix} \begin{bmatrix} V_{AB} \\ V_{BB} \end{bmatrix}^*. \tag{2.16}$$

This provides $\Delta\mathbf{A}$ and $\Delta\mathbf{B}$ so that

$$[(A + \Delta\mathbf{A})\ (B + \Delta\mathbf{B})] \begin{bmatrix} V_{AB} \\ V_{BB} \end{bmatrix} = 0. \tag{2.17}$$

Now if V_{BB} is nonsingular, we can then right multiply both sides by $-V_{BB}^{-1}$ to bring the bottom block of the right matrix to the negative identity, giving

$$\begin{aligned} [(A + \Delta\mathbf{A})\ (B + \Delta\mathbf{B})] \begin{bmatrix} -V_{AB}V_{BB}^{-1} \\ -V_{BB}V_{BB}^{-1} \end{bmatrix} &= \\ = [(A + \Delta\mathbf{A})\ (B + \Delta\mathbf{B})] \begin{bmatrix} x \\ -I_k \end{bmatrix} &= 0, \end{aligned} \tag{2.18}$$

and so the solution of the TLS problem for this case is

$$\hat{\mathbf{x}}_{TLS} = -V_{AB}V_{BB}^{-1}. \tag{2.19}$$

Since in our particular problem the matrix B is the vector $\mathbf{b}$ of $N \times 1$, then V_{BB} is a scalar and V_{AB} is a 2×1 vector, therefore the overall cost of this procedure is dominated by the singular value decomposition and the solution becomes

$$\hat{\mathbf{x}}_{TLS} = -\frac{(v_1, v_2)^T}{v_3}. \tag{2.20}$$

□

Optical flow estimation using total least squares instead of the conventional least squares method is not new. Already in 1995, Weber and Malik [185] used total least squares to solve the over-determined optical flow problem. They also used as a reliability measure, the consistency ratio $\frac{\sigma_3}{\sigma_2}$ which is the division between the two smallest singular values of the augmented matrix.

Tsai et al. [175] applied total least squares to estimate stereo optical flow and used σ_3, the smallest singular value of the augmented matrix $[\mathbf{A}\ \mathbf{b}]$, as a reliability measure of the estimates. They also regularized the estimated flow field based on the confidence provided by the value of σ_3. Similarly Bab-Hadiashar and Suter [10, 11] used the Least Median of Squares Orthogonal Distances (LMSOD) to identify the outliers and then total least squares to solve the optical flow problem. Finally, more recently Fashandi et al. [54] have proposed to estimate an optical flow field based on a wavelet decomposition and to use total least squares because of approximations performed on both sides of their over determined equation system.

2.2.3 Bias minimization through iterative and multiscale gradient-based shift estimation

In a least squares problem configuration, the linear least squares produces robust but not very accurate solutions since it ignores the noise in the spatial derivative matrix $\mathbf{A}$. Total least squares takes into account the noise implicitly contained in the matrix $\mathbf{A}$. However, when the noise is not independent or identically distributed or when the system is highly inconsistent (i.e. the determinant of the second order matrix $\mathbf{A}^T\mathbf{A}$ is small), TLS tends to give noisier estimates greatly affecting its robustness [116]. This is because TLS deals with the errors in $\mathbf{A}$ and $\mathbf{b}$ symmetrically. If all the errors in $\mathbf{A}$ and $\mathbf{b}$ are identical and independent, or their ratio can be obtained, then the TLS estimation is asymptotically unbiased [81]. However, since $\mathbf{A}$ is composed of spatial gradients which are estimated using numerical differentiation, the noise we expect on both variables $\mathbf{A}$ and $\mathbf{b}$ is correlated between neighboring pixels, which causes further problems for TLS. In fact, by ignoring the noise, the estimator proposed in (2.4) is systematically biased, meaning that its expected value is not equal to the true shift. Notice, there are no particular assumptions on the noise; it only needs to be symmetric around the true value.

A second source of bias for the estimator (2.4) is the Taylor approximation. Indeed, this method is derived from a Taylor approximation by truncating the Taylor series after the first order derivative. This approximation is accurate only when the second and the higher order derivatives are small. As a result, there is a systematic bias that depends on the image content and the displacement itself. This bias will be analyzed in more detail in section 4.3.2 of this manuscript. Nevertheless, we hereon present differential shift estimation methods that try to reduce its influence indirectly, without explicitly dealing with each of the mentioned reasons.

Instead of dealing with the bias explicitly, it was shown in Pham et al. [119] that both bias sources depend linearly on the shift magnitude. This justifies the use of an iterative method, which is able to significantly reduce the bias, provided an appropriate resampling method is used. This algorithm is described in Alg. 1, where k is the maximum amount of iterations, $findshift$ uses (2.4) to solve for $v[i]$ and $Resample$ resamples an image by interpolation. The selected interpolation algorithm could become a limiting factor to achieve high final accuracy, therefore this decision should not be underrated.

Note that while in the original formulation of Lucas and Kanade, the resampling was performed on the same image from where they computed the gradient (I_1), in the present formulation the gradient of image I_1 is computed only once, while the resampling is always performed on the second image, thus avoiding to recalculate it for every iteration. This method is known as the Inverse Compositional Algorithm [12]. It should also be

Algorithm 1 Iterative GBSE method.

```
1: procedure ILK($I_1, I_2$)                         ▷ Receives a pair of images
2:     $i \leftarrow 0$; $I_2[0] \leftarrow I_2$; $w \leftarrow 0$;
3:     while $i \leq k$ do
4:         $v[i] \leftarrow findshift(I_1, I_2[i])$      ▷ Eq. (2.4)
5:         $w \leftarrow w + v[i]$                   ▷ Accumulate total shift
6:         $I_2[i+1] \leftarrow Resample(I_2, -w)$     ▷ Use original $I_2$
7:         $i \leftarrow i + 1$
8:     end while
9:     return $w$                                  ▷ Return accumulated shift
10: end procedure
```

noted that instead of resampling using the already resampled image, it is always performed taking the original I_2, thus reducing the negative effects of inexact resampling methods and accelerating the process.

It was proven in Pham et al. [119] that this iterative scheme is able to significantly reduce the bias and to make the iterative GBSE estimator practically unbiased. Nevertheless, resampling of highly aliased images could end up violating the brightness constancy constraint, making a single iteration to outperform an iterative scheme, as shown in [122]. However, when a correct resampling is possible and with enough iterations, this method is the only one capable, to the best of our knowledge, to achieve optimal results with no bias.

Another element to consider is when the underlying displacements are larger than one pixel. In this case and as mentioned before, the presented GBSE method fails. However, by building a pyramid representation of the input images, Eq. (2.4) can be applied on each scale to estimate the shift between images, and this estimated shift can in turn be used to resample the second image on the following level of the pyramid [167]. If more accuracy is desired, an iterative scheme (as of Algorithm 1) could be used to better estimate the shift in each scale, although this comes with an increase in computational cost. We used in our implementation a dyadic Gaussian pyramid approximation [28], however we also tried an exact dyadic Gaussian pyramid [112] filtering with $\sigma = 1.4$ before subsampling, but we found out that the results were similar. Starting from the coarse image at scale $s > 1$, the method is presented in Algorithm 2.

Algorithm 2 Multiscale GBSE method.

```
1: procedure MSSE($I_1, I_2, s$)          ▷ Receives a pair of images and amount of scales
2:     $I_1^{1...s} \leftarrow BuildPyramid(I_1, s)$     ▷ Burt&Adelson's Gaussian Pyramid [28]
3:     $I_2^{1...s} \leftarrow BuildPyramid(I_2, s)$     ▷ i.e., IMPYRAMID function from Matlab
4:     $i \leftarrow s$;  $w \leftarrow 0$
5:     while  $i > 0$  do
6:         $v(i) \leftarrow findshift(I_1^i, I_2^i)$        ▷ Eq. (2.4)
7:         or $v(i) \leftarrow ILK(I_1^i, I_2^i)$          ▷ Alg. 1
8:         $w \leftarrow w * 2 + v(i) * 2$
9:         $I_2^{i-1} \leftarrow Resample(I_2^{i-1}, -w)$
10:        $i \leftarrow i - 1$
11:    end while
12:    $v(i) = findshift(I_1^1, I_2^1)$
13: end procedure
```

2.2.4 Minimizing the bias through corrected gradient estimation

Assuming that the noise is Gaussian white noise with variance σ^2 and does not depend on the image intensity, Ji et al. [81] proposed to use a simple and straightforward bias correction technique. In their work, they show that the bias of the least squares estimator amounts to

$$\begin{aligned} bias(\hat{\mathbf{x}}_{LS}) &= lim_{n\to\infty} E(\hat{\mathbf{x}}_{LS} - \mathbf{x}) \\ &= -\sigma^2 \left(lim_{n\to\infty} \left(\frac{1}{N} \mathbf{A}^T \mathbf{A} \right) \right)^{-1} \mathbf{x}, \end{aligned} \tag{2.21}$$

where $\mathbf{A}$ and $\mathbf{x}$ are obtained assuming noiseless images and N is the total amount of pixels in the image. To reduce the influence of the noise on the computations, they propose the corrected least squares estimator (CLS), given by

$$\mathbf{x}_{CLS} = \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} - N\sigma^2 \mathbf{I} \right)^{-1} \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{b}} \right), \tag{2.22}$$

where $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ are the noisy versions of $\mathbf{A}$ and $\mathbf{b}$ respectively, and $\mathbf{I}$ is the 2×2 identity matrix. Thus, provided a correct noise estimation is possible, then by simply subtracting an approximation of the error introduced by the noise from the biased components of the second order matrix, it is possible to attenuate the bias. In practice, however, this is a difficult task because noise in $\tilde{\mathbf{A}}$ is not always Gaussian nor is it uncorrelated with the signal. Furthermore, the noise-corrected second-order matrix in Eq. (2.22) is not necessarily a good approximation of the noiseless matrix $\mathbf{A}^T\mathbf{A}$ when the underlying image is aliased or has been pre-filtered [122]. Last but not least, the noise variance is not always known or easy to estimate.

2.2.5 Bidirectional bias correction for Gradient-Based Shift Estimation

To improve the accuracy of GBSE methods using a single iteration, an approach by Pham and Duggan [122] tries to estimate the unbiased second moment matrix $\mathbf{A}^T\mathbf{A}$ from the noisy images. Let $\tilde{I} = I + n$ be the noisy image where n denotes additive white Gaussian noise. Rewriting the least squares solution of the optical flow equation (2.4) as $\tilde{\mathbf{A}}\tilde{\mathbf{v}} = \tilde{\mathbf{b}}$ where

$$\underbrace{\begin{bmatrix} \sum \tilde{I}_x^2 & \sum \tilde{I}_x \tilde{I}_y \\ \sum \tilde{I}_x I_y & \sum \tilde{I}_y^2 \end{bmatrix}}_{\tilde{\mathbf{A}}} \underbrace{\begin{bmatrix} \tilde{v}_x \\ \tilde{v}_y \end{bmatrix}}_{\tilde{\mathbf{v}}} = \underbrace{\begin{bmatrix} \sum \tilde{I}_t \tilde{I}_x \\ \sum \tilde{I}_t \tilde{I}_y \end{bmatrix}}_{\tilde{\mathbf{b}}}, \tag{2.23}$$

and the tilde indicates the noisy version, the authors point out that, given n_x, n_y the directional derivatives of the noise n, the expected value of matrix $\tilde{\mathbf{A}}$ is given by

$$E[\tilde{\mathbf{A}}] = \begin{bmatrix} \sum I_x^2 + var(n_x) & \sum I_x I_y + cov(n_x, n_y) \\ \sum I_x I_y + cov(n_x, n_y) & \sum I_y^2 + var(n_y) \end{bmatrix} \tag{2.24}$$

therefore $E[\tilde{\mathbf{A}}] \neq \mathbf{A}$ making the computation of the derivative matrix $\mathbf{A}$ to be usually overestimated. Since $E(\tilde{\mathbf{b}}) = \mathbf{b}$, it makes sense to assume $\tilde{\mathbf{b}} \approx \mathbf{b}$, so we have that

$$\tilde{\mathbf{A}}\tilde{\mathbf{v}} = \tilde{\mathbf{b}} \approx \mathbf{b} = \mathbf{A}\mathbf{v} \tag{2.25}$$

from which we conclude that

$$\tilde{\mathbf{A}}\tilde{\mathbf{v}} = \mathbf{A}\mathbf{v}. \tag{2.26}$$

Given that both $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{v}}$ can be estimated using the noisy image and by solving for $\tilde{\mathbf{v}}$ in Eq. (2.23), the method then estimates $\mathbf{A}$ by performing three more shift estimations using shifted versions by an integer translation of the second image. By denoting

$$\mathbf{A} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}, \tag{2.27}$$

and $\tilde{\mathbf{v}}_I = findshift(\tilde{I}_1(x,y), \tilde{I}_2(x,y))$, this yields an overdetermined system with three unknowns and six equations. For example, by assuming $\tilde{v}_x < 0$ and $\tilde{v}_y < 0$, the system is given by

$$\tilde{\mathbf{v}}_{00} = \tilde{\mathbf{v}}_I = findshift(\tilde{I}_1, \tilde{I}_2(x,y)) = \mathbf{M}\tilde{\mathbf{v}} \tag{2.28}$$
$$\tilde{\mathbf{v}}_{10} = findshift(\tilde{I}_1, I_2(x+1,y)) = \mathbf{M}(\tilde{\mathbf{v}}+[1\ 0]^T)$$
$$\tilde{\mathbf{v}}_{01} = findshift(\tilde{I}_1, I_2(x,y+1)) = \mathbf{M}(\tilde{\mathbf{v}}+[0\ 1]^T)$$
$$\tilde{\mathbf{v}}_{11} = findshift(\tilde{I}_1, I_2(x+1,y+1)) = \mathbf{M}(\tilde{\mathbf{v}}+[1\ 1]^T), \tag{2.29}$$

where $\mathbf{M} = \tilde{\mathbf{A}}^{-1}\mathbf{A}$. Then subtracting (2.28) and pre-multiplying both sides by $\tilde{\mathbf{A}}$ gives

$$[p_1, q_1] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{10} - \tilde{\mathbf{v}}_{00}) \approx \mathbf{A}[1\ 0]^T = [a\ b]^T \tag{2.30}$$
$$[p_2, q_2] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{01} - \tilde{\mathbf{v}}_{00}) \approx \mathbf{A}[0\ 1]^T = [b\ c]^T \tag{2.31}$$
$$[p_3, q_3] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{11} - \tilde{\mathbf{v}}_{00}) \approx \mathbf{A}[1\ 1]^T = [a+b\ b+c]^T. \tag{2.32}$$

Finally, the unbiased matrix $\mathbf{A}$ is obtained by weighted least squares, minimizing the following functional

$$\epsilon = w_1(a-p_1)^2 + w_1(b-q_1)^2 + w_2(b-p_2)^2 + w_2(c-q_2)^2 + w_3(a+b-p_3)^2 + w_3(b+c-q_3)^2 \tag{2.33}$$

where the weights are chosen so that smaller shifts are given more importance

$$w_1 = \| |\tilde{\mathbf{v}}_{10}| + |\tilde{\mathbf{v}}_{00}| \|^{-2}, w_2 = \| |\tilde{\mathbf{v}}_{01}| + |\tilde{\mathbf{v}}_{00}| \|^{-2} \text{ and } w_3 = \| |\tilde{\mathbf{v}}_{11}| + |\tilde{\mathbf{v}}_{00}| \|^{-2} \tag{2.34}$$

The justification for the weights is that GBSE center their Taylor development in zero, and therefore the smaller the shift, the more accurate it will be estimated. Finally, the weighted least squares system becomes the solution of

$$\begin{bmatrix} w_1+w_3 & w_3 & 0 \\ w_3 & w_1+w_2+2w_3 & w_3 \\ 0 & w_3 & w_2+w_3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} w_1p_1 + w_3p_3 \\ w_1q_1 + w_2p_2 + w_3(p_3+q_3) \\ w_2q_2 + w_3q_3 \end{bmatrix}, \tag{2.35}$$

were a, b and c were the values of the unbiased matrix $\mathbf{A}$. As can be seen, functional (2.33) as well as the weights definitions came from the assumption that $\tilde{\mathbf{v}}_I < \mathbf{0}$. Therefore, the method begins by estimating $\tilde{\mathbf{v}}_I$, the shift between both original images $\tilde{I}_1$ and $\tilde{I}_2$. Based on this result, four cases arise depending on the sign of both values of $\tilde{\mathbf{v}}_I$, each one yielding different equations which in turn define distinct weighted least squares systems to minimize as well as weight definitions (in the given example, $\tilde{\mathbf{v}}_I = \tilde{\mathbf{v}}_{00}$).

By assuring the subpixel condition of the underlying GBSE method, the other three cases, depending on the initial shift estimation $\tilde{\mathbf{v}}_I = [\tilde{v}_x, \tilde{v}_y]^T$, are

- If $\tilde{v}_x > 0$ and $\tilde{v}_y > 0$, then

$$\tilde{\mathbf{v}}_{00} = findshift(\tilde{I}_1, \tilde{I}_2(x-1, y-1)) = \mathbf{M}(\tilde{\mathbf{v}} - [1\ 1]^T),$$
$$\tilde{\mathbf{v}}_{10} = findshift(\tilde{I}_1, \tilde{I}_2(x-1, y)) = \mathbf{M}(\tilde{\mathbf{v}} + [-1\ 0]^T),$$
$$\tilde{\mathbf{v}}_{01} = findshift(\tilde{I}_1, \tilde{I}_2(x, y-1)) = \mathbf{M}(\tilde{\mathbf{v}} + [0\ -1]^T),$$
$$\tilde{\mathbf{v}}_{11} = \tilde{\mathbf{v}}_I = findshift(\tilde{I}_1, \tilde{I}_2(x,y)) = \mathbf{M}\tilde{\mathbf{v}}. \tag{2.36}$$

- If $\tilde{v}_x > 0$ and $\tilde{v}_y < 0$, then

$$\begin{aligned}
\tilde{\mathbf{v}}_{00} &= findshift(\tilde{I}_1, \tilde{I}_2(x-1,y)) = \mathbf{M}(\tilde{\mathbf{v}} + [-1\ 0]^T),\\
\tilde{\mathbf{v}}_{10} &= \tilde{\mathbf{v}}_I = findshift(\tilde{I}_1, \tilde{I}_2(x,y)) = \mathbf{M}\tilde{\mathbf{v}},\\
\tilde{\mathbf{v}}_{01} &= findshift(\tilde{I}_1, \tilde{I}_2(x-1,y+1)) = \mathbf{M}(\tilde{\mathbf{v}} + [-1\ 1]^T),\\
\tilde{\mathbf{v}}_{11} &= findshift(\tilde{I}_1, \tilde{I}_2(x,y+1)) = \mathbf{M}(\tilde{\mathbf{v}} + [0\ 1]^T).
\end{aligned} \tag{2.37}$$

- If $\tilde{v}_x < 0$ and $\tilde{v}_y > 0$, then

$$\begin{aligned}
\tilde{\mathbf{v}}_{00} &= findshift(\tilde{I}_1, \tilde{I}_2(x,y-1)) = \mathbf{M}(\tilde{\mathbf{v}} + [0\ -1]^T),\\
\tilde{\mathbf{v}}_{10} &= findshift(\tilde{I}_1, \tilde{I}_2(x+1,y-1)) = \mathbf{M}(\tilde{\mathbf{v}} + [1\ -1]^T),\\
\tilde{\mathbf{v}}_{01} &= \tilde{\mathbf{v}}_I = findshift(\tilde{I}_1, \tilde{I}_2(x,y)) = \mathbf{M}\tilde{\mathbf{v}},\\
\tilde{\mathbf{v}}_{11} &= findshift(\tilde{I}_1, \tilde{I}_2(x+1,y)) = \mathbf{M}(\tilde{\mathbf{v}} + [1\ 0]^T).
\end{aligned} \tag{2.38}$$

Once we have the four shift estimates, given by either one of (2.29), (2.36), (2.37) or (2.38), a weighted least squares minimization scheme is used to compute a, b and c, the values of the unbiased matrix $\mathbf{A}$.

Following the same reasoning, the systems to solve for these three cases become

- If $\tilde{\mathbf{v}}_I = \tilde{\mathbf{v}}_{11}$ then

$$w_1 = \big\||\tilde{\mathbf{v}}_{00}|+|\tilde{\mathbf{v}}_{11}|\big\|^{-2}, \quad w_2 = \big\||\tilde{\mathbf{v}}_{01}|+|\tilde{\mathbf{v}}_{11}|\big\|^{-2}, \quad w_3 = \big\||\tilde{\mathbf{v}}_{10}|+|\tilde{\mathbf{v}}_{11}|\big\|^{-2} \tag{2.39}$$

$$[p_1, q_1] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{00} - \tilde{\mathbf{v}}_{11}), \quad [p_2, q_3] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{01} - \tilde{\mathbf{v}}_{11}), \quad [p_3, q_3] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{10} - \tilde{\mathbf{v}}_{11}) \tag{2.40}$$

and the system to solve for a, b and c is

$$\begin{bmatrix} -w_1-w_2 & -w_1 & 0 \\ -w_1 & -2w_1-w_2-w_3 & -w_1 \\ 0 & -w_1 & -w_1-w_3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} w_1 * p_1 + w_2 * p_2 \\ w_1 * (p_1 + q_1) + w_2 * q_2 + w_3 * p_3 \\ w_1 * q_1 + w_3 * q_3 \end{bmatrix}. \tag{2.41}$$

- If $\tilde{\mathbf{v}}_I = \tilde{\mathbf{v}}_{01}$ then

$$w_1 = \big\||\tilde{\mathbf{v}}_{00}|+|\tilde{\mathbf{v}}_{01}|\big\|^{-2}, \quad w_2 = \big\||\tilde{\mathbf{v}}_{10}|+|\tilde{\mathbf{v}}_{01}|\big\|^{-2}, \quad w_3 = \big\||\tilde{\mathbf{v}}_{11}|+|\tilde{\mathbf{v}}_{01}|\big\|^{-2} \tag{2.42}$$

$$[p_1, q_1] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{00} - \tilde{\mathbf{v}}_{01}), \quad [p_2, q_2] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{10} - \tilde{\mathbf{v}}_{01}), \quad [p_3, q_3] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{11} - \tilde{\mathbf{v}}_{01}) \tag{2.43}$$

and the system to solve for a, b and c is

$$\begin{bmatrix} w_2+w_3 & -w_2 & 0 \\ -w_2 & w_1+2w_2+w_3 & -w_2 \\ 0 & w_2 & -w_1-w_2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} w_2 * p_2 + w_3 * p_3 \\ -w_1 * p_1 + w_2 * (q_2 - p_2) + w_3 * q_3 \\ w_1 * q_1 + w_2 * q_2 \end{bmatrix}. \tag{2.44}$$

- If $\tilde{\mathbf{v}}_I = \tilde{\mathbf{v}}_{10}$ then

$$w_1 = \big\||\tilde{\mathbf{v}}_{00}|+|\tilde{\mathbf{v}}_{10}|\big\|^{-2}, \quad w_2 = \big\||\tilde{\mathbf{v}}_{01}|+|\tilde{\mathbf{v}}_{10}|\big\|^{-2}, \quad w_3 = \big\||\tilde{\mathbf{v}}_{11}|+|\tilde{\mathbf{v}}_{10}|\big\|^{-2} \tag{2.45}$$

$$[p_1, q_1] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{00} - \tilde{\mathbf{v}}_{10}), \quad [p_2, q_2] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{01} - \tilde{\mathbf{v}}_{10}), \quad [p_3, q_3] = \tilde{\mathbf{A}}(\tilde{\mathbf{v}}_{11} - \tilde{\mathbf{v}}_{10}) \tag{2.46}$$

and the system to solve for a, b and c is

$$\begin{bmatrix} -w_1-w_2 & w_2 & 0 \\ -w_2 & w_1+2w_2+w_3 & -w_2 \\ 0 & -w_2 & w_2+w_3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} w_1 * p_1 + w_2 * p_2 \\ -w_1 * q_1 + w_2 * (p_2 - q_2) + w_3 * p_3 \\ w_2 * q_2 + w_3 * q_3 \end{bmatrix}. \tag{2.47}$$

Therefore, depending on the initial shift estimation result $\tilde{\mathbf{v}}_I$, one of the four possible cases is used to calculate the values of the unbiased derivative matrix $\mathbf{A}$. At last, the real unbiased shift is calculated as

$$\mathbf{v} = \mathbf{A}^{-1}\tilde{\mathbf{b}}. \tag{2.48}$$

2.2.6 Gradient computation and image prefiltering

By focusing on the design of the filters used to estimate the image gradient under noise, studied in conjunction with prefiltering the input images, Simoncelli [154] was able to reduce the bias by using a gradient filter that approximates the derivatives of the prefilter. The proposed pre-smoothing filters are forced to be symmetric, to guarantee they are linear phase filters, while the gradient filters are selected to be anti-symmetric in order to preserve the property of being a differentiator. Farid and Simoncelli [53] also proposed a set of both prefilters and differentiators obtained by minimizing the errors in the gradient direction, for a fixed size kernel.

This approach was later followed by Elad et al. [50] where they specifically studied the problem for GBSE. In their work, they noted that by designing a set of pre-smoothing filters and gradients filters minimizing the modelling error for each particular image, the estimator performance could be further improved. In fact, each filter is designed based on the spectral form of the first image and the *a priori* knowledge of a maximal motion. Their objective is then to find filter parameters for a set of filters by minimizing a cost function in order to reduce the error of the shift estimates. Although this solution achieves improved accuracy over previous approaches, it does in fact decrease the bias caused by the noise in an indirect way, completely ignoring the statistical performance of the GBSE. Furthermore, the minimization must be done for every image pair, thus radically augmenting the computational burden.

Followed by a precise study on the bias in [139], Robinson and Milanfar proposed designing a gradient filter based on the selection of pre-filters, on the prior knowledge of the image spectrum and some constraint about the shift [140]. Surprisingly, this work proposed to minimize the estimator bias by attacking the approximation error in the data model due to the linear signal approximation performed by the Taylor development, while completely ignoring the noise. In fact, low SNR conditions are discarded even though the bias due to the noise dominates the overall estimator bias [119]. For this reason, they achieve poor results on images with SNR lower than 20dB. Furthermore, none of these previous approaches work under aliased situations or badly sampled images, which is possible (yet undesired) on computer vision problems. To this end, Christmas [35] proposed a differentiator kernel that optimises the fit of the estimator with respect to the ideal differentiator $D(f) = i2\pi f$ at low frequencies, giving less importance to higher frequencies usually more affected by aliasing. This approach is suitable on cases where the image contents appear on the lower frequencies.

Note also that using a large kernel for image derivatives computation imply not only excessive blurring of the image, but also discarding more boundary pixels, thus leaving fewer equations to estimate the shift on (2.4), constraining the gradient kernel to be compact, precise and robust to the presence of noise. In fact, the impact on the accuracy and the robustness to noise of the gradient computation is a key factor to the final performance of the GBSE method. What is more, this computation must be fast in order to accelerate the algorithm.

Emulating the work of Simoncelli [154], in order to increase the accuracy of the method by minimizing noise or aliasing influence, we look for two kernel functions: an asymmetric kernel $\mathbf{d}$ to estimate the image gradients and a symmetric kernel $\mathbf{k}$ to prefilter the

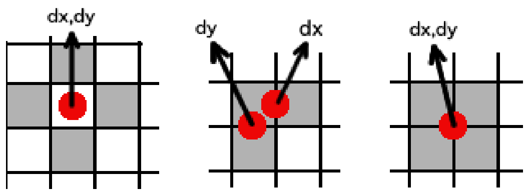


Figure 2.1 – Used pixels (gray background) for fast gradient estimation methods and their exact localizations (red spots). **Left**: Centered differences. **Center**: 1D backward difference for both ∂x and ∂y. **Right**: 2D backward difference.

images. Using both kernels, matrix $\mathbf{A}$ and vector $\mathbf{b}$ from Eq. (2.4) become

$$\mathbf{A} = \begin{pmatrix} (\mathbf{d}_x * I_1)(p_1) & (\mathbf{d}_y * I_1)(p_1) \\ \vdots & \vdots \\ (\mathbf{d}_x * I_1)(p_n) & (\mathbf{d}_y * I_1)(p_n) \end{pmatrix} \text{ and } \mathbf{b} = - \begin{pmatrix} (\mathbf{k} * (I_1 - I_2))(p_1) \\ \vdots \\ (\mathbf{k} * (I_1 - I_2))(p_n) \end{pmatrix}, \tag{2.49}$$

where $*$ denotes convolution.

Because the method should be computationally fast, a straightforward candidate for gradient estimation is the well-known centered differences method using a $[1, 0, -1]$ kernel, however since the central pixel is ignored in the computation, its performance is usually poor under high precision constraints. For this reason, a backward difference method using a $[1, -1]$ kernel would seem more appropriate, however, this derivative corresponds to the center between both pixels and not in the pixels itself, as seen from the middle image of Fig. 2.1.

A more exact gradient estimation method is shown on the right of Fig. 2.1. Using the original image I_1, the derivatives are calculated by convolving it with $\mathbf{d}_x$ and $\mathbf{d}_y$ given by

$$\mathbf{d}_x = \begin{bmatrix} 1/2 & -1/2 \\ 1/2 & -1/2 \end{bmatrix}, \qquad \mathbf{d}_y = \begin{bmatrix} 1/2 & 1/2 \\ -1/2 & -1/2 \end{bmatrix} \tag{2.50}$$

and resampled versions of both images I_1 and I_2 are used to calculate vector $\mathbf{b}$ of (2.3), shifting them by half a pixel to the bottom right using bilinear interpolation, finally taking

$$\mathbf{k} = \begin{bmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{bmatrix}. \tag{2.51}$$

This gradient estimation trick, which we shall call the *hypomode*, despite being simplistic usually improves the accuracy obtained by GBSE methods using finite difference gradient estimation. This is because it slightly blurs the input images, which alleviates both aliasing and noise, and because of its accurate gradient localization.

Another considered smoothing kernel is the 2D Gaussian kernel given by sampling from

$$\mathbf{k} = g(x, y, \sigma_g) = \frac{1}{2\pi\sigma_g^2} \exp\left(-\frac{x^2 + y^2}{2\sigma_g^2}\right) \tag{2.52}$$

and its derivatives

$$\mathbf{d}_x = \frac{\partial g(x, y, \sigma_g)}{\partial x} = -\frac{x}{2\pi\sigma_g^4} \exp\left(-\frac{x^2 + y^2}{2\sigma_g^2}\right), \tag{2.53}$$

$$\mathbf{d}_y = \frac{\partial g(x, y, \sigma_g)}{\partial y} = -\frac{y}{2\pi\sigma_g^4} \exp\left(-\frac{x^2 + y^2}{2\sigma_g^2}\right). \tag{2.54}$$

The kernel support is determined by the σ_g value, which defines the amount of blur applied to each image before performing the computations, therefore higher noise values imply a higher σ_g. While a too low value would be less tolerant to noise, a too high value would imply losing potentially valuable textures which usually aid the shift estimation method. In our experiments, we evaluated using $\sigma_g = \{0.3, 0.6, 1\}$ leading to supports $3, 5$ and 7 respectively.

Other evaluated image gradient estimation methods in the context of GBSE were the 3×3 and the 5×5 kernels from Simoncelli [154], and the 3×3, 5×5 and 7×7 from Farid [53]. As mentioned above, these approaches are the result of finding a smoothing prefilter, which is assumed to be a separable product of identical symmetric 1D functions, together with a derivative antisymmetric filter. Simoncelli minimizes the following energy

$$E(P, D) = \int d\omega W^2(\omega)[j\omega P(\omega) - D(\omega)]^2 \tag{2.55}$$

where $P(\omega)$ and $D(\omega)$ are the Fourier transform of a prefilter and a derivative filter respectively. The weights W are taken to mimic the expected spectral content of natural images

$$W(\omega) = \frac{1}{\sqrt{|\omega|}}. \tag{2.56}$$

Therefore, the method fixes a kernel size and minimizes Eq. (2.55) by defining some constraints on both kernels. On the other side, Farid minimizes

$$E(P, D) = \frac{\int_\omega |j\omega P(\omega) - D(\omega)|^2}{\int_\omega P^2(\omega)} \tag{2.57}$$

in its discrete form

$$E(\mathbf{p}, \mathbf{d}) = \frac{|j\omega F_s\mathbf{p} - F_a\mathbf{d}|^2}{|F_s\mathbf{p}|^2}, \tag{2.58}$$

where $\mathbf{p}$ is the prefilter vector of length $(L+1)/2$, $\mathbf{d}$ the derivative filter vector of size $(L-1)/2$, F_s and F_a are matrices whose columns contain the real and imaginary components of the discrete Fourier basis of size $K >> L$ such that $F_s\mathbf{p}$ gives the DFT of the prefilter and $F_a\mathbf{d}$ gives the DFT of the antisymmetric derivative filter.

The gradient estimators from Christmas [35] were also included in the evaluation. As mentioned before, they are obtained by fitting the kernel with the ideal differentiator $D(f) = i2\pi f$ at low frequencies, avoiding high frequency information. Specifically, for an nth order estimator, they equate the first n derivatives of the estimator frequency response to those of the ideal differentiator. Note that Christmas' approach does not require prefiltering the images, yielding faster processing times, although usually coming with a decrease in accuracy.

All evaluated gradient estimators kernels are shown in table 2.1. Given image I, the gradient along the horizontal direction is estimated by convolving each column of I with the prefilter vector $\mathbf{k}$ and then convolving each row of the result with the differentiator vector $\mathbf{d}$. The vertical gradient is obtained by swapping $\mathbf{d}$ and $\mathbf{k}$ yielding

$$I_x = \mathbf{d} * \mathbf{k}^T * I, \tag{2.59}$$

$$I_y = \mathbf{k} * \mathbf{d}^T * I. \tag{2.60}$$

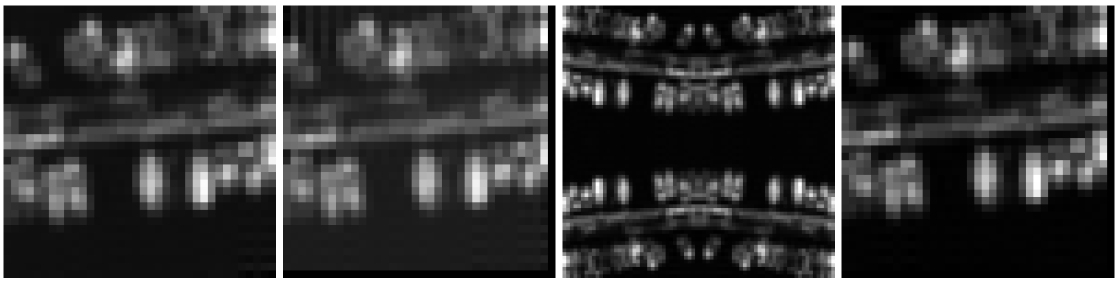
2.2.7 Interpolation methods for image resampling

In order to iterate the algorithm, resampling must be done to shift the image, as indicated in the step 6 of Algorithm 1. To this end, five different interpolation methods were evaluated, namely bilinear, bicubic [82] and cubic spline interpolation [42], together with

Kernel		Sample Number						
		-3	-2	-1	0	1	2	3
Hypomode	**k**				0.5	0.5		
2×2	**d**				1	-1		
Gaussian	**k**			0.003865	0.999990	0.003865		
$\sigma = 0.3$	**d**			0.707110	0.000000	-0.707110		
Gaussian	**k**		0.003645	0.235160	0.943070	0.235160	0.003645	
$\sigma = 0.6$	**d**		0.021915	0.706770	0.000000	-0.706770	-0.021915	
Gaussian	**k**	0.008343	0.101650	0.455560	0.751090	0.455560	0.101650	0.008343
$\sigma = 1$	**d**	0.035436	0.287800	0.644920	0.000000	-0.644920	-0.287800	-0.035436
Simoncelli	**k**			0.224209	0.551580	0.224209		
3×3	**d**			0.455271	0.000000	-0.455271		
Simoncelli	**k**		0.035697	0.248874	0.430855	0.248874	0.035697	
5×5	**d**		0.107662	0.282671	0.000000	-0.282671	-0.107662	
Farid	**k**			0.229879	0.540242	0.229879		
3×3	**d**			0.425287	0.000000	-0.425287		
Farid	**k**		0.037659	0.249153	0.426375	0.249153	0.037659	
5×5	**d**		0.109604	0.276691	0.000000	-0.276691	-0.109604	
Farid	**k**	0.004711	0.069321	0.245410	0.361117	0.245410	0.069321	0.004711
7×7	**d**	0.018708	0.125376	0.193091	0.000000	-0.193091	-0.125376	-0.018708
Christmas	**k**				1			
3×3	**d**			1	0	-1		
Christmas	**k**				1			
5×5	**d**		-1/12	2/3	0	-2/3	1/12	
Christmas	**k**				1			
7×7	**d**	1/60	-3/20	3/4	0	-3/4	3/20	-1/60

Table 2.1 – Gradient estimation kernels evaluated for GBSE methods.

resampling using the Fourier shift theorem [20], which is evaluated with and without image periodization. Image periodization avoids the generation of ringing artifacts due to the discontinuities on the image borders by generating an augmented version mirroring the image. This resulting image has no discontinuities when periodization is assumed, as depicted in Fig. 2.2.



(a) Original image (b) DFT resampling (c) Symmetrized image (d) Symmetrized DFT

Figure 2.2 – Example of FFT resampling with and without image symmetrization. Direct resampling with DFT produces ringing due to the discontinuities at the periodized boundaries. No visible ringing is observed after resampling with symmetrization.

The impact on the results by selecting a correct interpolation method could be important, as it will be shown in the results. The more precise the interpolation, the more accurate the results, however it comes with an increase in the processing cost. A report of interpolation methods for fast image resampling due to a global displacement will be presented in Appendix 1.

2.2.8 Shift estimation by centroid of interpolating kernel

In a recent work by Gilman and Leist [64], the authors propose an illumination-invariant method for fast image registration, for the case of additive contrast changes. Their approach tries to overcome the negative aspects of iterative gradient-based methods by proposing a fast non-iterative solution that does not require performing gradient estimation. Conventionally, iterative GBSE methods perform the following minimization

$$\underset{v_x,v_y}{\operatorname{argmin}} \sum_{x,y} \left(I_2(x,y) - I_1(x+v_x, y+v_y)\right)^2 \tag{2.61}$$

by approximating $I_1(x+v_x, y+v_y)$ by its first order Taylor development. This is followed by an iterative scheme in which the second image is resampled using the result from the shift estimation. By doing this, an interpolation method such as Eq. (2.62) is required. The interpolation kernel h is based on some interpolation basis function with rectangular support between (M^-, N^-) and (M^+, N^+) and its coefficients are samples of the basis function offset by the estimated shift (v_x, v_y). The interpolation is given by the following expression

$$I_1(x+v_x, y+v_y) = \sum_{m=M^-}^{M^+} \sum_{n=N^-}^{N^+} I_1(x+m, y+n) h_{\hat{v}_x,\hat{v}_y}(m,n). \tag{2.62}$$

By combining (2.61) with (2.62), the authors propose to perform the minimization with respect to the interpolating kernel

$$\underset{\mathbf{h}}{\operatorname{argmin}} \sum_{x,y} \left(I_2(x,y) - \sum_{m=M^-}^{M^+} \sum_{n=N^-}^{N^+} I_1(x+m, y+n) h(m,n) \right)^2 \tag{2.63}$$

yielding a linear system $\mathbf{Av} = \mathbf{b}$ where, provided the interpolation kernel is of size $W_k \times H_k$, its main computational cost involves inverting a $W_k^2 \times H_k^2$ matrix. Its implementation is accelerated by simply pre-computing the shifted versions of I_1 and then performing the Hadamard product.

Finally the shift estimates can be obtained by computing

$$\hat{u}_x = \sum_m \sum_n m h(m,n), \qquad \hat{u}_y = \sum_m \sum_n n h(m,n). \tag{2.64}$$

However, the method could easily be made robust against illumination changes. Let A and S be the gain and the offset of a global illumination change given by $I_2(x,y) = AI_1(x,y) + S$, then by normalizing its computation

$$\hat{u}_x = \frac{\sum_m \sum_n m h(m,n)}{\sum_m \sum_n h(m,n)}, \qquad \hat{u}_y = \frac{\sum_m \sum_n n h(m,n)}{\sum_m \sum_n h(m,n)}, \tag{2.65}$$

the method becomes robust against gain A and therefore contrast invariant. The authors also suggest adding the offset variable S into the optimization in Eq. (2.63) to add illumination invariance as well.

2.3 Phase-correlation methods

Introduced in 1975 by Kuglin and Hines [86], the phase correlation method aligns images based on their normalized cross-power spectrum, which is equivalent to computing the normalized circular cross-correlation in the spatial domain. Let I_1 and I_2 be $M \times N$ images such that

$$I_2(x,y) = I_1((x-v_x) \bmod M, (y-v_y) \bmod N) + n(x,y), \tag{2.66}$$

where $\bmod$ is the modulo operator implying a circular shift and $n(x,y)$ denotes the effect of interference terms such as noise, non-overlapping regions, etc. Let F_1 and F_2 be the discrete 2D Fourier transforms of I_1 and I_2 respectively, then by ignoring this last term the Fourier Shift Theorem indicates that

$$F_2(\omega_x,\omega_y) = F_1(\omega_x,\omega_y)\exp\left(-i2\pi\left(\frac{\omega_x v_x}{M} + \frac{\omega_y v_y}{N}\right)\right) \tag{2.67}$$

and by the cross-correlation theorem we have that

$$C = I_1 \star I_2 = I_1^*(-t) \otimes I_2(t) = \mathcal{F}^{-1}\{F_1^* F_2\}, \tag{2.68}$$

where $\star$ and $\otimes$ denote the cross-correlation and the convolution operator respectively, $*$ denotes the complex conjugate and $\mathcal{F}^{-1}$ stands for the discrete inverse Fourier transform. Then by normalizing in Fourier the cross-correlation and based on the Fourier Shift Theorem, the phase correlation matrix, defined by the normalized cross-power spectrum C, is given by

$$\begin{aligned} C(\omega_x,\omega_y) &= \frac{F_2(\omega_x,\omega_y)F_1^*(\omega_x,\omega_y)}{F_1(\omega_x,\omega_y)F_1^*(\omega_x,\omega_y)} \\ &= \exp\left(-i2\pi\left(\frac{\omega_x v_x}{M} + \frac{\omega_y v_y}{N}\right)\right). \end{aligned} \tag{2.69}$$

If both F_1 and F_2 were continuous, then by applying the inverse Fourier transform, we obtain the phase-only correlation (POC) [162] or phase correlation surface (PCS) [135] given by

$$c(x,y) = \mathcal{F}^{-1}\Big\{C(\omega_x,\omega_y)\Big\}(x,y) = \delta(x-v_x, y-v_y), \tag{2.70}$$

where $\delta(x-v_x,y-v_y)$ is a Dirac function centered at (v_x,v_y). Note that by performing normalization, this method becomes robust to affine intensity changes, i.e.

$$I_1(x-v_x, y-v_y) = a \cdot I_2(x,y) + b, \quad a,b \in \mathbf{R}. \tag{2.71}$$

Also note that this method is based on the Fourier shift theorem, which holds when the shift between both images is circular, that is, the part of the image that disappears on one side, appears on its opposite side. Finally, the peak of c is searched to obtain the translation between both images:

$$(\hat{v}_x,\hat{v}_y) = \arg\max_{(x,y)} c(x,y). \tag{2.72}$$

Unless the shift can be exactly described as in Eq. (2.66), due to reasons such as sub-pixel displacements, aliasing, image noise or non-overlapped regions, $c(x,y)$ is not an exact Dirac function, although the location of the peak still permits to accurately compute the displacement. Indeed, the peak value is often lower than one pixel and the surface is frequently noisy, as seen from figures 2.3 and 2.4. Sub-pixel shifts, for example, imply that the energy is distributed between the peak and its adjacent neighbors. Commonly, function fitting methods are used to estimate the peak location to non-integer values. Some other methods try to estimate the shift directly in the Fourier domain, eliminating the need to compute another FFT and thus, decreasing the complexity of the algorithm.

While in this thesis we will describe only the most relevant approaches, complementary information could be found on review articles [4, 133]. Extensions of this technique to estimate scale and rotation transformations also exist [132], although they will not be covered in this review.

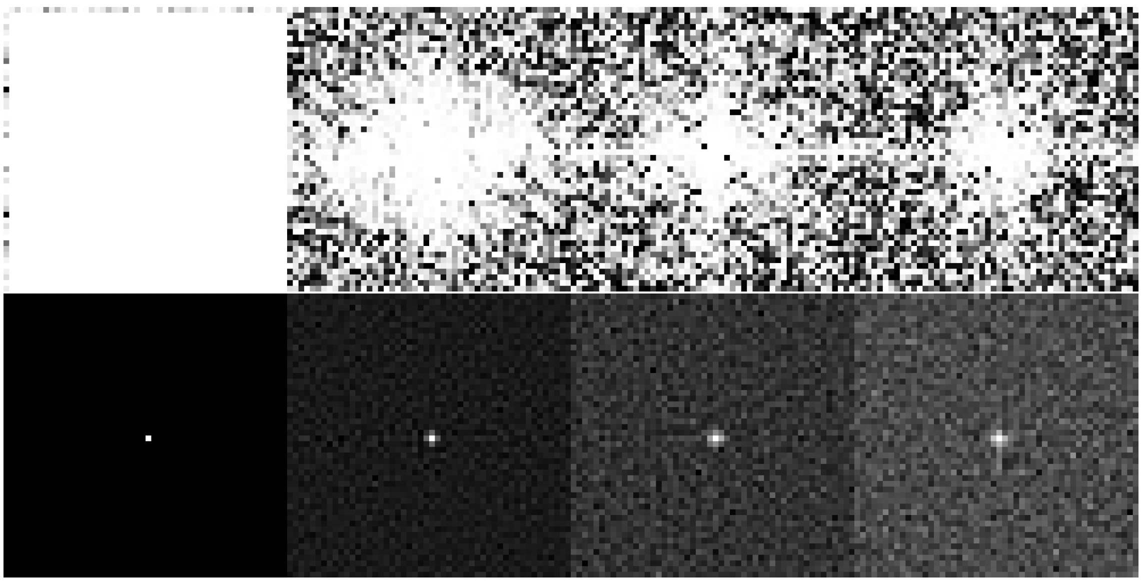

(a) $\sigma = 0$ (b) $\sigma = 75$ (c) $\sigma = 150$ (d) $\sigma = 300$

Figure 2.3 – Cross-power spectrum $C(\omega_x, \omega_y)$ and phase correlation surfaces $c(x, y)$ of two identical images under different random WGN. **Top**: Cross power spectrum. **Bottom**: phase correlation surfaces. Dynamic ranges extended for visualization purposes.

2.3.1 Local function fitting in the spatial domain

Fitting well-known functions in the vicinity of the integer maximum peak of the phase correlation surface can be used to obtain accurate subpixel precision while being computationally efficient.

Quadratic fitting. Abdou [1], working on registration of video frames, proposed to fit a 1D quadratic (parabolic) function on each separate coordinate on the vicinity of the main peak. Let (x_m, y_m) be the integer position of the peak (the integer solution of Eq. (2.72)), then the values used for the fitting on the horizontal and vertical directions are

$$\text{Horizontal: } \{c(x_m - 1, y_m), c(x_m, y_m), c(x_m + 1, y_m)\}, \tag{2.73}$$

$$\text{Vertical: } \{c(x_m, y_m - 1), c(x_m, y_m), c(x_m, y_m + 1)\}. \tag{2.74}$$

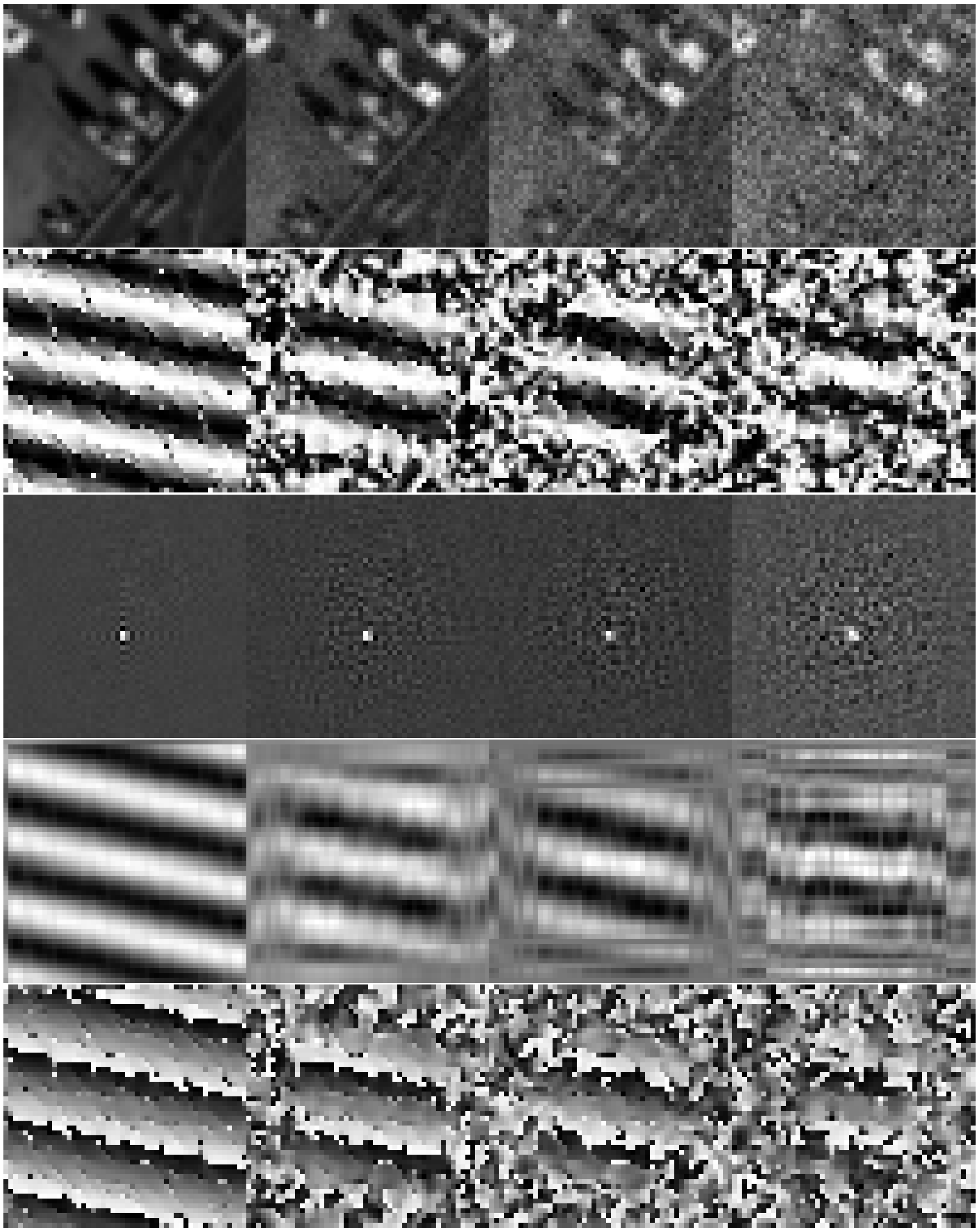
Then fitting a parabolic function yields a closed-form solution

$$\hat{v}_x = \frac{c(x_m + 1, y_m) - c(x_m - 1, y_m)}{2 * c(x_m, y_m) - c(x_m + 1, y_m) - c(x_m - 1, y_m)}. \tag{2.75}$$

Gaussian fitting. In [1] the author also proposed fitting a Gaussian function to the same data, thus yielding the closed form formula

$$\hat{v}_x = \frac{\log \left| \frac{c(x_m+1, y_m)}{c(x_m-1, y_m)} \right|}{\log \left| \frac{c(x_m, y_m)^2}{c(x_m+1, y_m) \cdot c(x_m-1, y_m)} \right|}. \tag{2.76}$$

The vertical shifts are obtained by using the samples from (2.74) instead of (2.73).



(a) $\sigma = 0$ (b) $\sigma = 75$ (c) $\sigma = 150$ (d) $\sigma = 300$

Figure 2.4 – Example of different surfaces used to estimate the displacement of two images shifted by $(-3.5, 0.75)$ pixels under different random WGN ($\sigma = 0, 75, 150$ and 300) assuming 12-bit images. **First row**: First image. **Second row**: Real component of cross-power spectrum $C(\omega_x, \omega_y)$. **Third row**: Phase correlation surfaces $c(x, y)$. **Fourth row**: real component rank-1 approximation of the CPS [75]. **Last row**: Phase difference matrix [159]. Dynamic ranges extended for visualization purposes.

Sinc fitting. Foroosh et al. [57,149] propose to understand the subpixel shift of the input images as if it has been obtained by an integer shift on a higher resolution grid followed by subsampling. Based on this assumption, they analytically determine that the shape of the phase correlation surface corresponds to a Dirichlet kernel. Indeed, the cross-power spectrum of the downsampled frames by factors of M and N along both horizontal and vertical axes is given by

$$C(u,v) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \frac{F(\frac{u+2\pi m}{M},\frac{v+2\pi n}{N})}{\sum_{m'=0}^{M-1}\sum_{n'=0}^{N-1} F(\frac{u+2\pi m'}{M},\frac{v+2\pi n'}{N})} \exp\left(-i\left(\frac{u+2\pi m}{M}v_x, \frac{v+2\pi n}{N}v_y\right)\right) \tag{2.77}$$

and its inverse DFT is

$$\mathcal{F}^{-1}\Big\{C(u,v)\Big\} = c(x,y) = \frac{1}{WH}\frac{\sin(\pi(Mx-v_x))}{\sin(\pi(Mx-v_x)/W)}\frac{\sin(\pi(Ny-v_y))}{\sin(\pi(Ny-v_y)/H)}, \tag{2.78}$$

where W and H are the width and height respectively before downsampling. Based on this study, the authors propose to approximate this Dirichlet kernel by a sinc function

$$sinc(x) = \frac{\sin \pi x}{\pi x} \tag{2.79}$$

yielding

$$c(x,y) \approx \frac{\sin(\pi(Mx-v_x))}{\pi(Mx-v_x)}\frac{\sin(\pi(Ny-v_y))}{\pi(Ny-v_y)} + n(x,y), \tag{2.80}$$

where $n(x,y)$ refers to interference terms such as noise, aliasing, non-overlapped regions, possible compression artifacts, etc.

Therefore, to rapidly recover the shift, based on the fact that for subpixel displacements the signal power in the phase correlation is usually concentrated in a main peak (x_m, y_m) and two side-peaks at $(x_m, y_m \pm 1)$ and $(x_m \pm 1, y_m)$, the authors calculate the shift by linearly weighting the main peak together with one of the side-peaks using only two values to estimate the shift on each dimension. For example, by assuming the side-peaks to be (x_m+1, y_m) and (x_m, y_m+1) and neglecting the interference component $n(x,y)$, this yields

$$\hat{v}_x = \frac{c(x_m+1,y_m)}{c(x_m+1,y_m) \pm c(x_m,y_m)}, \quad \hat{v}_y = \frac{c(x_m,y_m+1)}{c(x_m+1,y_m) \pm c(x_m,y_m)}, \tag{2.81}$$

where both $\hat{v}_x$ and $\hat{v}_y$ are chosen so that they both lie between $[x_m-1, x_m+1]$ and $[y_m-1, y_m+1]$ respectively. Another possibility is to do least squares minimization to fit the 3×3 grid centered on $c(x_m,y_m)$ on each dimension with a 1D sinc function:

$$\hat{v}_x = \text{argmin} \sum_{x_i=\{x_m-1,x_m,x_m+1\}} \left[c(x_i,y_m) - sinc(x_i - C)\right]^2. \tag{2.82}$$

However, due to the non-linearity of the sinc, a non-linear least squares minimization method has to be used. Since this minimization requires only to fit a function over just three values, it does not require much computational time. However, this solution ignores the interference term, thus yielding poor results under low SNR scenarios. Indeed, computing $\hat{v}_x$ of Eq. (2.81) by considering $n(x,y)$ in Eq. (2.80) and taking the limit when $v_x \to 0$ we obtain

$$\lim_{v_x\to 0} \hat{v}_x = \frac{n(1,0)}{(\pi v_y)^{-1}\sin(\pi v_y) + n(1,0) + n(0,0)} \neq 0, \tag{2.83}$$

which proves that the estimator is biased under low SNR situations.

eSinc fitting. More recently, Argyriou and Vlachos [8] proposed a modification of the Sinc function by applying exponential weighting, defined as follows

$$esinc(x) = \exp(-x^2)\frac{\sin \pi x}{\pi x}. \tag{2.84}$$

They claim this modified sinc function is able to better approximate phase correlation surfaces obtained from video data. The esinc is later parametrized to adapt to different magnitudes, scales and shifts changes, searching the parameters of $A\ esinc(B(x-C))$, which leads to the following minimization problem

$$(A, B, C) = \text{argmin} \sum_{x_i=\{x_m-1, x_m, x_m+1\}} \left[c(x_i, y_m) - A\ esinc(B(x_i - C))\right]^2, \tag{2.85}$$

that is again solved using a non-linear optimization method. In this case, C stands for the horizontal shift estimation $\hat{v}_x$; $\hat{v}_y$ is obtained in a similar way, by sampling over the vertical axis. Note that this minimization scheme could also be applied to the sinc fitting case, so therefore will be included in the evaluation.

Using the difference of both side-peaks. Ren et al. [134] claimed that using the difference between both side-peaks of $c(x_m, y_m)$ instead of a single side-peak as in [57] reduces the bias caused by ignoring the interference term as shown in Eq. (2.83). Let $D_x = c(x_m+1, y_m) - c(x_m-1, y_m)$ and $D_y = c(x_m, y_m+1) - c(x_m, y_m-1)$, their method then reduces to calculating

$$\hat{v}_x = \frac{D_x}{c(x_m, y_m) + |D_x|} = \frac{sign(D_x)}{1 + c(x_m, y_m)/|D_x|}, \hat{v}_y = \frac{D_y}{c(x_m, y_m) + |D_y|} = \frac{sign(D_y)}{1 + c(x_m, y_m)/|D_y|}. \tag{2.86}$$

Least squares fitting of modified cross-power spectrum Takita *et al.* [162] drastically improved the accuracy by proposing three important modifications to the method. First, they suggest using least squares fitting involving the frequencies of the neighbourhood around the peak (x_m, y_m) up to 9×9 pixels. Second, a 2D Hanning window should be applied to the images before computing their cross-power spectrum to avoid false edge effects. Finally, the authors propose to filter the cross-power spectrum using low-pass-type filters in order to reduce the influence of corrupted high frequency components. Since this implies modifying the peak shape, they propose a specific fitting model for each low-pass filter. The four filters are

$$H_1(\omega_x, \omega_y) = \begin{cases} 1 & |\omega_x| \leq U_1, |\omega_y| \leq U_2, \\ 0 & otherwise, \end{cases} \tag{2.87}$$

$$H_2(\omega_x, \omega_y) = \frac{1}{MN} H_1(\omega_x, \omega_y) \otimes H_1(\omega_x, \omega_y), \tag{2.88}$$

$$H_3(\omega_x, \omega_y) = \frac{1}{M^2N^2} H_2(\omega_x, \omega_y) \otimes H_1(\omega_x, \omega_y), \tag{2.89}$$

$$H_4(\omega_x, \omega_y) = \exp\left\{2\pi^2\sigma^2(\omega_x^2 + \omega_y^2)\right\}, \tag{2.90}$$

and the models to fit each of them are given by

$$r_1(x,y) = \frac{\alpha}{MN} \frac{\sin\left(\frac{V_1}{M}\pi(x+v_x)\right)}{\sin\left(\frac{\pi}{M}(x+v_x)\right)} \frac{\sin\left(\frac{V_2}{N}\pi(y+v_y)\right)}{\sin\left(\frac{\pi}{N}(y+v_y)\right)}, \tag{2.91}$$

$$r_2(x,y) = \left\{ \frac{\alpha}{MN} \frac{\sin\left(\frac{V_1}{M}\pi(x+v_x)\right)}{\sin\left(\frac{\pi}{M}(x+v_x)\right)} \frac{\sin\left(\frac{V_2}{N}\pi(y+v_y)\right)}{\sin\left(\frac{\pi}{N}(y+v_y)\right)} \right\}^2, \tag{2.92}$$

$$r_3(x,y) = \left\{ \frac{\alpha}{MN} \frac{\sin\left(\frac{V_1}{M}\pi(x+v_x)\right)}{\sin\left(\frac{\pi}{M}(x+v_x)\right)} \frac{\sin\left(\frac{V_2}{N}\pi(y+v_y)\right)}{\sin\left(\frac{\pi}{N}(y+v_y)\right)} \right\}^3, \tag{2.93}$$

$$r_4(x,y) = \frac{1}{2\pi\sigma^2} \exp\left\{ -\frac{x^2+y^2}{2\sigma^2} \right\}, \tag{2.94}$$

where α is a cut-off frequency parameter, $V_1 = 2U_1 + 1$, $V_2 = 2U_2 + 1$ and U_1, U_2 and σ controls the pass-band width.

Increasing accuracy by zero padding A simple strategy to increase phase correlation accuracy is to upsample the cross-correlation grid by zero-padding the cross-power spectrum $C(\omega_x, \omega_y)$. Obviously, this implies increasing the algorithm cost since the inverse DFT has to be computed for a much larger image. However, once an initial estimate of the correlation peak is found, Guizar-Sicairos [69] proposed to refine this value by only computing the DFT values on a small neighborhood around the peak, achieving more precision without using too many computational resources. In fact, the authors propose the use a matrix multiplication implementation of the 2D DFT [156], which performs the upsampling of a 1.5×1.5 pixel neighborhood around the initial estimate without the need to zero-pad the cross-correlation.

Local center of mass on the phase correlation surface Another fast strategy to achieve subpixel accuracy is to compute the center of mass of the phase correlation surface [4,90]. The center of mass could be computed in 1D using

$$\hat{v}_x = \frac{\sum_{x=x_m-w}^{x_m+w} x \cdot c(x, y_m)}{\sum_{x=x_m-w}^{x_m+w} c(x, y_m)}, \quad \hat{v}_y = \frac{\sum_{y=y_m-w}^{y_m+w} y \cdot c(x_m, y)}{\sum_{y=y_m-w}^{y_m+w} c(x_m, y)}, \tag{2.95}$$

where $w \in [1, 2, 3]$ is the support size, or using a 2D approach

$$\hat{v}_x = \frac{\sum_{y=y_m-w}^{y_m+w} \sum_{x=x_m-w}^{x_m+w} x \cdot c(x,y)}{\sum_{y=y_m-w}^{y_m+w} \sum_{x=x_m-w}^{x_m+w} c(x,y)}, \quad \hat{v}_y = \frac{\sum_{y=y_m-w}^{y_m+w} y \cdot c(x,y)}{\sum_{y=y_m-w}^{y_m+w} \sum_{x=x_m-w}^{x_m+w} c(x,y)}. \tag{2.96}$$

2.3.2 Analyzing the phase difference matrix in the frequency domain

Instead of computing the inverse Fourier transform in Eq. (2.70), several methods try to compute the shift directly in the Fourier domain [13, 159]. These methods work by computing the phase of the cross power spectrum

$$\phi(\omega_x, \omega_y) = \arg\left(C(\omega_x, \omega_y)\right) = \frac{\omega_x v_x}{M} + \frac{\omega_y v_y}{N}, \tag{2.97}$$

where the shift (v_x, v_y) can be easily recovered by fitting a plane passing through the origin. The matrix $\phi(\omega_x, \omega_y)$ is usually referred to as the phase difference matrix [13], displayed in the last row of Fig. 2.4.

Robust plane fitting Once ϕ is computed, several approaches recover the shift by a linear least squares plane regression method [159], [85] and [152]. However, due to noise and aliasing on the input images, these methods discard corrupted frequencies in the minimization, yielding the following least squares minimization

$$\hat{v}_x, \hat{v}_y = \underset{v_x, v_y}{\operatorname{argmin}} \sum_{\omega_x, \omega_y} M(\omega_x, \omega_y) \left[2\pi \left(\frac{\omega_x v_x}{M} + \frac{\omega_y v_y}{N} \right) - \phi(\omega_x, \omega_y) \right]^2, \tag{2.98}$$

where $M(\omega_x, \omega_y)$ is the binary mask used to exclude contaminated spectral components from the minimization.

Knutsson *et al.* [85] emphasizes on speed, so its method uses only two or four frequencies of the whole spectra, yielding an extremely fast method that avoids the computation of the DFTs:

$$M(\omega_x, \omega_y) = \begin{cases} 1 & \text{if } (\omega_x, \omega_y) = (1, 0) \vee (\omega_x, \omega_y) = (0, 1), \\ 0 & \text{otherwise.} \end{cases} \tag{2.99}$$

Stone *et al.* [159] analytically studied the influence of aliasing on the phase difference matrix and proposed an algorithm that focuses on reducing its influence on the final accuracy. To this end, their approach first applies a window in the spatial domain in order to eliminate image-boundary effects in the frequency domain. In particular, the authors suggest using either Blackman or Blackman-Harris windows [155]. Finally, the minimization in (2.98) is performed by masking spectral components mostly contaminated by aliasing, namely:

$$M(\omega_x, \omega_y) = \begin{cases} 0 & \text{if } \sqrt{\omega_x^2 + \omega_y^2} > 0.3 \cdot N \vee |F_1(\omega_x, \omega_y)| < \alpha \vee |F_2(\omega_x, \omega_y)| < \alpha, \\ 1 & \text{otherwise.} \end{cases} \tag{2.100}$$

Indeed, they exclude frequencies further away than $0.3 \cdot N$ from the center, where N is the minimum of the number of samples in both dimensions, and frequencies where the energy is below a specified threshold α in any of both images. In the original article, α was chosen by sorting the frequencies by magnitude and retaining the K largest ones. Furthermore, K was set by performing several shift estimations over a range of values and keeping the one where the shift remained stable. In addition to the fact that this procedure is computationally expensive, the range where the shift remained stable was not always straightforward to detect in our empirical testing. A greedy technique that improved over the original method was to discard the frequencies where the magnitudes of both images were lower than a specified percentile. In particular, we observed that taking the percentile 60 systematically improved over the original approach.

Sidick *et al.* [152] use this same approach in the adaptive cross-correlation (ACC) method to perform shift estimation. However, they perform it in an iterative scheme where they first estimate the shift between centered windows half the size of the images followed by resampling the second image in the frequency domain accumulating the estimated shift. The same authors later proposed adaptive periodic correlation (APC) [153] where they changed this shift estimation method for a more robust periodic correlation approach [126]. Both methods from Sidick as well as the periodic correlation method will be given in more detail on the next chapter of this thesis.

Sawtooth cycle count Robust plane fitting approaches estimate the slope of the plane passing through the origin of the phase difference matrix. Not only these methods decay considerably with the noise, but they also suffer from the phase wrapping problem [63]. For this reason, both approaches are limited to estimating displacements up to one pixel, for which the phase will not wrap.

Balci and Foorosh [13] made the observation that the phase difference matrix $\phi(\omega_x, \omega_y)$ defined in Eq. (2.97) is a 2D sawtooth signal (see last row of Fig. 2.4) and that the subpixel displacement between both images can be determined by counting the number of cycles along each axis. Indeed, if no phase unwrapping is performed, these cycles are attributed to the wrapping behaviour of the phase. To compute the number of cycles, they propose to count the peaks in the Hough-transform domain [78]. Each of these peaks provides one linear constraint on both v_x and v_y, yielding an over-determined system which is solved by imposing a regularity contraint where the regularization parameter is obtained using the Generalized Cross Validation (GCV) method.

2.3.3 Subspace phase correlation methods

Projection-based subspace methods Alliney and Morandi [5] showed that the complexity of 2D phase correlation could be significantly reduced by performing two 1D phase correlation estimations on the image projections in each dimension. Indeed, let $F(\omega_x, \omega_y)$ be the Fourier transform of $I(x, y)$, then by defining

$$I_x(x) = \sum_y I(x, y), \quad I_y(y) = \sum_x I(x, y) \tag{2.101}$$

and computing the Fourier transform $F_x(\omega_x)$ of $I_x(x)$ we have that

$$F_x(\omega_x) = \mathcal{F}\Big\{ \sum_y I(x, y) \Big\} = \sum_x \sum_y I(x, y) e^{-i2\pi\omega_x x/M}, \tag{2.102}$$

$$= \sum_x \sum_y I(x, y) e^{-i2\pi(\omega_x x/M + 0\cdot y/N)} = F(\omega_x, 0) \tag{2.103}$$

and similarly for $I_y(y)$

$$F_y(\omega_y) = F(0, \omega_y). \tag{2.104}$$

Based on Eq. (2.67) we know that

$$F_2(\omega_x, 0) = F_1(\omega_x, 0) \exp\left(\text{-}\ i2\pi\omega_x v_x/M\right), \tag{2.105}$$

which yields

$$F_{2x}(\omega_x) = F_{1x}(\omega_x) \exp\left(\text{-}\ i2\pi\omega_x v_x/M\right), \tag{2.106}$$

where F_{1x} and F_{2x} are the Fourier transforms of the x projections for both images I_1 and I_2 respectively. Then the phase correlation method could be applied to each 1D projection to estimate the displacement. In this case, the 1D cross-power spectrum C_x and the 1D PCS c_x for the x direction becomes

$$C_x(\omega_x) = \frac{F_{1x}(\omega_x) F^*_{2x}(\omega_x)}{|F_{1x}(\omega_x) F^*_{2x}(\omega_x)|}, \quad c_x(x) = \mathcal{F}^{-1}\Big\{ C_x(\omega_x) \Big\}(x). \tag{2.107}$$

Equivalently for y, given F_{1y} and F_{2y} the Fourier transforms of y projections for images I_1 and I_2 respectively, both vectors are defined as

$$C_y(\omega_y) = \frac{F_{1y}(\omega_y) F^*_{2y}(\omega_y)}{\Big|F_{1y}(\omega_y) F^*_{2y}(\omega_y)\Big|}, \quad c_y(y) = \mathcal{F}^{-1}\Big\{ C_y(\omega_y) \Big\}(y). \tag{2.108}$$

The authors of [5] suggest to compute both shift estimates using 1D versions of the phase correlation methods presented so far, by windowing the input images. Nevertheless, they do so for integer displacements. In Fig. 2.5 one can see two images shifted by $(-3.5, 0.75)$ pixels, under different noise intensities. One observes that the noise does not affect so much the projections, however differences between the projections appear due to objects entering and leaving the scene. This indeed affects the performance of projection-based methods.

Robinson and Milanfar [138] generalized the notion of projection by using the Radon transform at an angle θ defined on the continuous image $f(x,y)$ as,

$$\mathcal{R}(p,\theta)[f(x,y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(p - x\cos(\theta) - y\sin(\theta))\,dx\,dy, \tag{2.109}$$

where p is the perpendicular distance from a line to the origin and θ is the angle formed by the distance vector. Then by using two perpendicular angles $\theta_i, i = 1,2$, they estimate the 2D shift by first calculating both 1D shifts u_{θ_1} and u_{θ_2} to finally compute

$$\begin{bmatrix} \hat{v}_x \\ \hat{v}_y \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) \\ \cos(\theta_2) & \sin(\theta_2) \end{bmatrix}^{-1} \begin{bmatrix} u_{\theta_1} \\ u_{\theta_2} \end{bmatrix}. \tag{2.110}$$

Each 1D shift u_{θ_i} is computed using a line fitting approach on the unwrapped phases of the 1D normalized cross-power spectrums $C_\theta(\omega_x)$, using a masking procedure resembling the approach of Stone [159], as presented in section 2.3.2. The shift in the θ direction is therefore estimated by computing

$$\underset{u_\theta}{\operatorname{argmin}} \sum_{\omega} M_\theta(\omega)\left[2\pi\omega u_\theta/M - \phi_\theta(\omega)\right]^2, \tag{2.111}$$

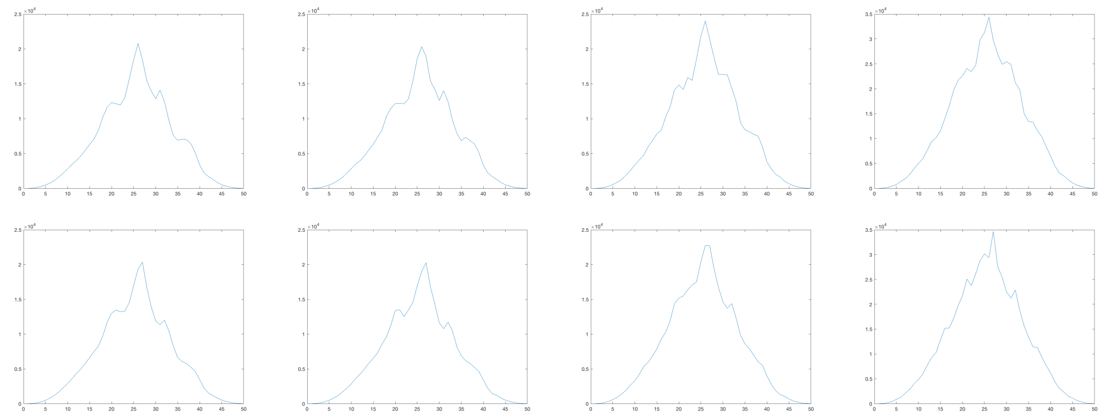
where $\phi_\theta(\omega) = \arg\left(C_\theta(\omega)\right)$ and $M_\theta(\omega)$ is the 1D equivalent of the binary weighting mask used to reduce contaminated frequencies. The shift in the y directions is calculated using the same strategy. To keep the computational complexity low, they again reduce to the setup presented above by choosing $\theta_1 = 0$ and $\theta_2 = \pi/2$, so both C_{θ_1} and C_{θ_2} are defined by Eqs. (2.107) and (2.108) respectively. Finally, they conclude that using a projection-based approach achieves dramatic savings in computation with essentially no degradation in final accuracy. In Fig. 2.6 we see the incidence of noise on the unwrapped phases of the cross-power spectrums of two images shifted by $(-3.5, 0.75)$. When the noise is low, a line fitting algorithms seems suitable, however under lower SNR scenarios, it seems the method accuracy will suffer.

Ren *et al.* [135] use the gradients of the projected components to improve even further the accuracy and robustness of the method to the ignored interference terms in Eq. (2.66). Indeed, let $I'_{2x}(x) = I_{2x}(x+1) - I_{2x}(x)$ then

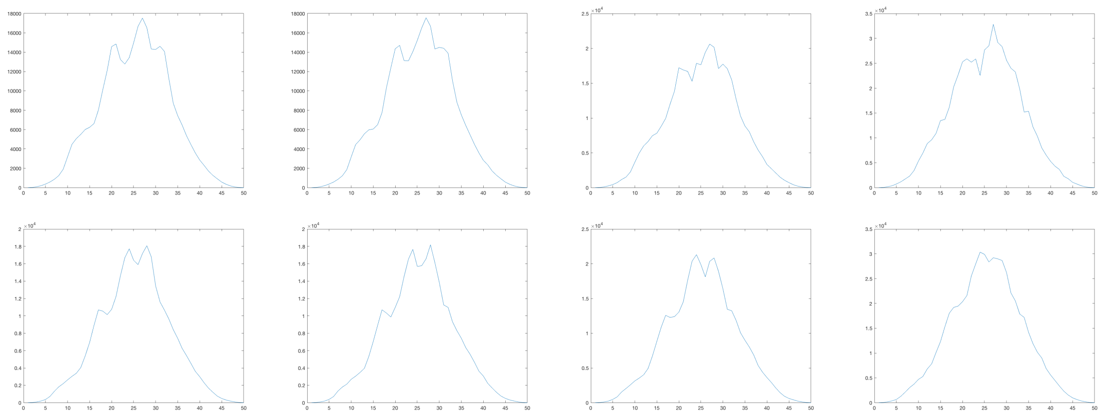
$$\begin{aligned} I'_{2x}(x) &= I_{2x}(x+1) - I_{2x}(x) && (2.112)\\ &= \sum_y I_2((x+1) \bmod M, y) - \sum_y I_2(x,y) && (2.113)\\ &= \sum_y \left[I_1((x+1-v_x) \bmod M, (y-v_y) \bmod N) + n((x+1) \bmod M, y)\right] \\ &\quad - \sum_y \left[I_1((x-v_x) \bmod M, (y-v_y) \bmod N) + n(x,y)\right] && (2.114)\\ &= I'_{1x}(x - v_x) + n'_x(x), && (2.115) \end{aligned}$$

where $n'_x(x) = n_x(x+1) - n_x(x)$ and $n_x(x) = \sum_y n(x,y)$. When N is large enough, the authors point out that $n'_x(x) = 0$, thus yielding $I'_{2x}(x) = I'_{1x}(x - v_x)$. This means that the gradients of the projections are not affected by the interference terms, therefore the method becomes more robust to them, even under non-zero-mean noise.

Then the PCS between both gradients is computed and the subpixel shifts are estimated by fitting a Gaussian around the peak values. To increase robustness while also accelerating the computations, the peak value is only searched around the interval $\left[\frac{7M}{16}, \frac{9M}{16}\right]$, using one-eight of the samples.

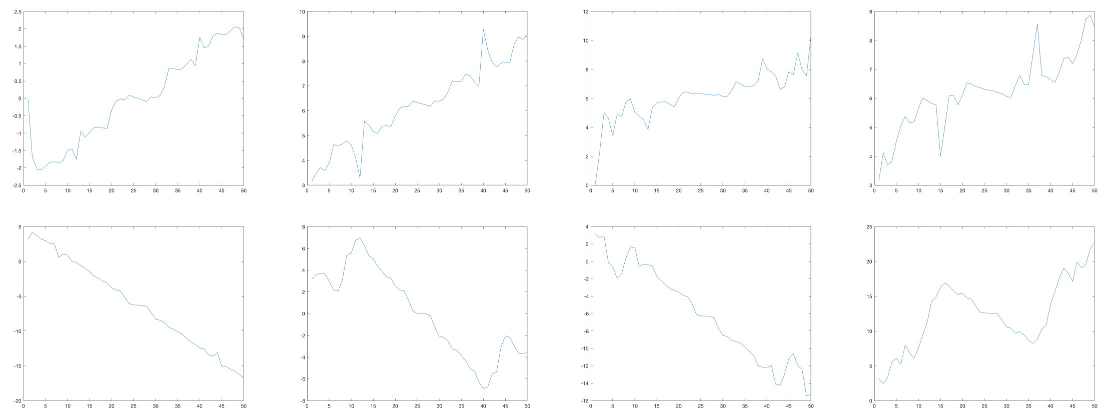


(a) I_{1y} and I_{2y}, $\sigma=0$ (b) I_{1y} and I_{2y}, $\sigma=75$ (c) I_{1y} and I_{2y}, $\sigma=150$ (d) I_{1y} and I_{2y}, $\sigma=300$



(e) I_{1x} and I_{2x}, $\sigma=0$ (f) I_{1x} and I_{2x}, $\sigma=75$ (g) I_{1x} and I_{2x}, $\sigma=150$ (h) I_{1x} and I_{2x}, $\sigma=300$

Figure 2.5 – Projections of two images shifted by $(-3.5, 0.75)$ pixels under different random WGN ($\sigma = 0, 75, 150$ and 300) assuming 12-bit images. **Top two rows**: Vertical direction of both images (shift 0.75). **Last two rows**: Horizontal direction of both images (shift -3.5).



(a) $\sigma = 0$ (b) $\sigma = 75$ (c) $\sigma = 150$ (d) $\sigma = 300$

Figure 2.6 – Unwrapped phases obtained from the normalized cross-power spectrum of two images shifted by $(-3.5, 0.75)$ pixels under different random WGN ($\sigma = 0, 75, 150$ and 300) assuming 12-bit images. Both images were pre-filtered using a Blackman window. **Top**: Vertical direction (shift 0.75). **Bottom**: Horizontal direction (shift -3.5).

The rank-one approach A straightforward but usually overlooked aspect about the phase correlation matrix of Eq. (2.69) is that, in a noiseless context, each element could be written as the product of two complex exponentials,

$$C(\omega_x, \omega_y) = \exp\left(i2\pi\omega_x v_x/M\right) \cdot \exp\left(i2\pi\omega_y v_y/N\right) = q_x(\omega_x)q_y(\omega_y). \tag{2.116}$$

This means that matrix $C(\omega_x, \omega_y)$ can be written as the outer product of non-zero vectors, implying it has rank one. By making this observation, Hoge [75] proposed to compute the singular value decomposition (SVD) of the phase correlation matrix C to obtain a rank-1 approximation, followed by applying linear least squares on unwrapped versions of both $\arg(q_x)$ and $\arg(q_y)$, the left and right dominant singular vectors. Since both are 1D signals, the unwrapping is straightforward and the computational cost of the minimization is significatively lower. To gain robustness against aliasing and edge effects, the unwrapped 1D phases are masked by only including frequencies lying on a specific range, namely $|\omega| \in [2, 0.6(s/2)]$ where s is the length of the array. Fig. 2.4 puts in evidence the effects of computing the rank-1 approximation of the matrix $C(\omega_x, \omega_y)$. While the original cross-power spectrum is noisy, its rank-1 approximation yields a much cleaner version. Its effects under low SNR are however not stable, affecting the accuracy. Interestingly, the phase difference matrix gets considerably denoised using the rank-1 approximation, as seen in Fig. 2.7. The unwrapped versions of the 1D phase components for both left and right singular vectors are also displayed in this figure. While on the vertical direction, the line fitting seems straightforward in most cases, the same does not occur on the other direction due to the high amount of outliers. A robust line fitting method such as RANSAC could be used in this task. Indeed, this is the main idea behind the recent article of Tong *et al.* [170].

2.3.4 Gradient correlation methods

The phase correlation (PC) methods discussed above are able to estimate image displacements because the phase component of images holds structural information. Therefore by correlating the phase components between two images, it is possible to estimate the displacement between them. Indeed, phase information is invariant with respect to uniform variations of illumination, has a strong response to edges and usually yields high peak localizaion accuracy, which makes phase-correlation approaches suitable for several shift estimation tasks. Nevertheless, other shift estimation approaches, which resemble PC methods, are based on correlating the image gradients [7, 178, 179].

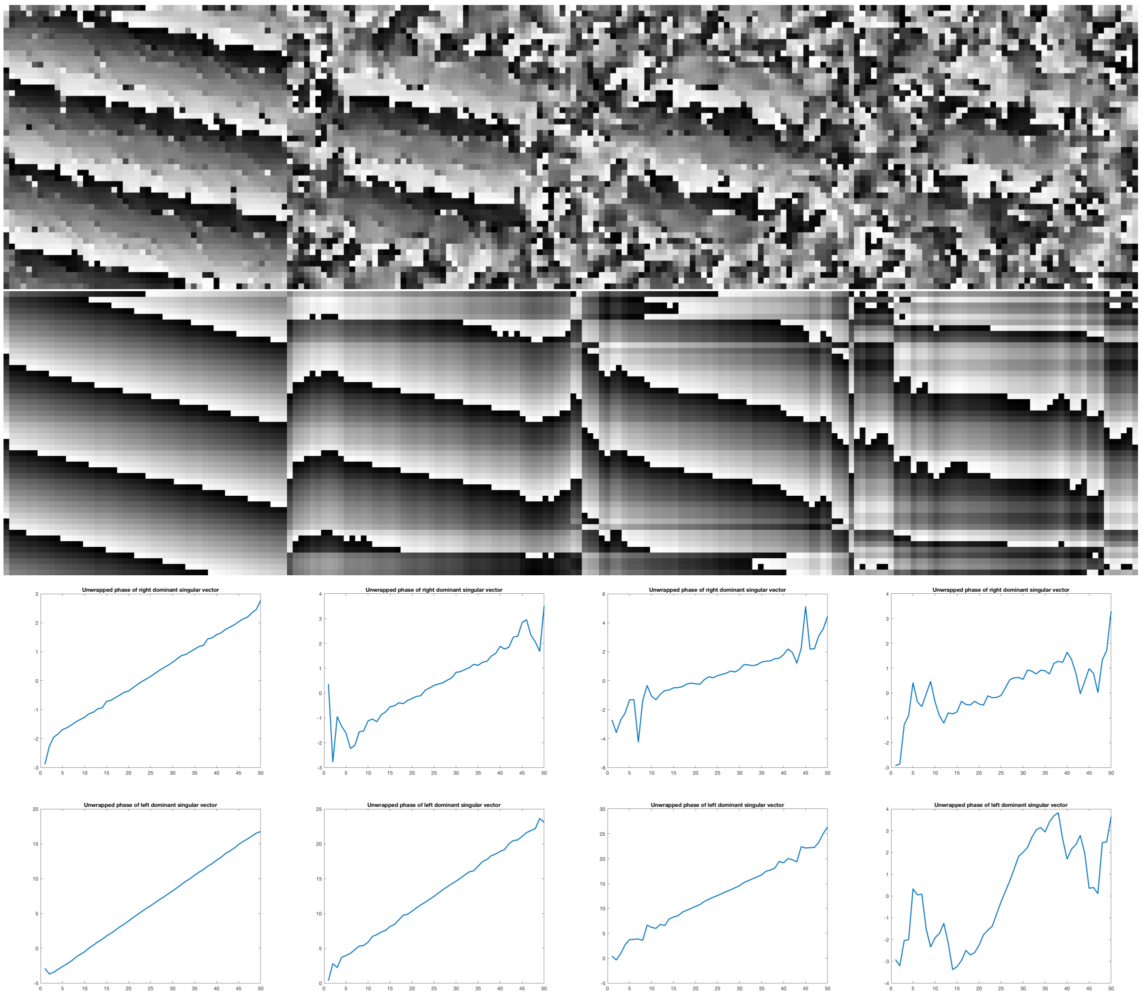
For each input image, the gradient correlation (GC) approach [7] combines both horizontal and vertical gradients into a new image using complex numbers. Then, it computes the cross-correlation of these two complex images in the frequency domain to finally estimate the peak. Formally, images g_1 and g_2 are given by

$$g_i(\mathbf{x}) = \frac{\partial I_i}{\partial x}(\mathbf{x}) + j\frac{\partial I_i}{\partial y}(\mathbf{x}), \quad i = 1, 2. \tag{2.117}$$

If $G_1 = \mathcal{F}\{g_1(\mathbf{x})\}$ and $G_2 = \mathcal{F}\{g_2(\mathbf{x})\}$ are their respective Fourier representations, $|G_1| = \mathcal{F}\{|g_1(\mathbf{x})|\}$ and $|G_2| = \mathcal{F}\{|g_2(\mathbf{x})|\}$ the Fourier representations of their magnitudes, then the gradient correlation (GC) and the normalized gradient correlation NGC are as follows

$$GC(\mathbf{x}) = g_1(\mathbf{x}) \star g_2(\mathbf{x}) = \mathcal{F}^{-1}\{G_1^* G_2\}(\mathbf{x}) \tag{2.118}$$

$$NGC(\mathbf{x}) = \frac{\mathcal{F}^{-1}\{G_1^* G_2\}}{\mathcal{F}^{-1}\{|G_1|^* |G_2|\}}. \tag{2.119}$$



(a) $\sigma = 0$ (b) $\sigma = 75$ (c) $\sigma = 150$ (d) $\sigma = 300$

Figure 2.7 – Phase difference matrix computed directly (first row) and by using a rank-1 approximation (second row) of the cross-power spectrum of two images shifted by $(-3.5, 0.75)$ pixels under different random WGN ($\sigma = 0, 75, 150$ and 300) assuming 12-bit images. Third and fourth rows display the phase unwrapped left an right dominant singular vectors of the cross-power spectrum respectively. Both images were pre-filtered using a Blackman window. Dynamic ranges extended for visualization purposes.

The images are previously zero padded to avoid undesired edge effects in the correlation. Finally, to achieve sub-pixel accuracy, the same quadratic fitting proposed in Abdou [1] is used around the peak value. An example of a gradient correlation surface is shown on the first row of Fig. 2.8.

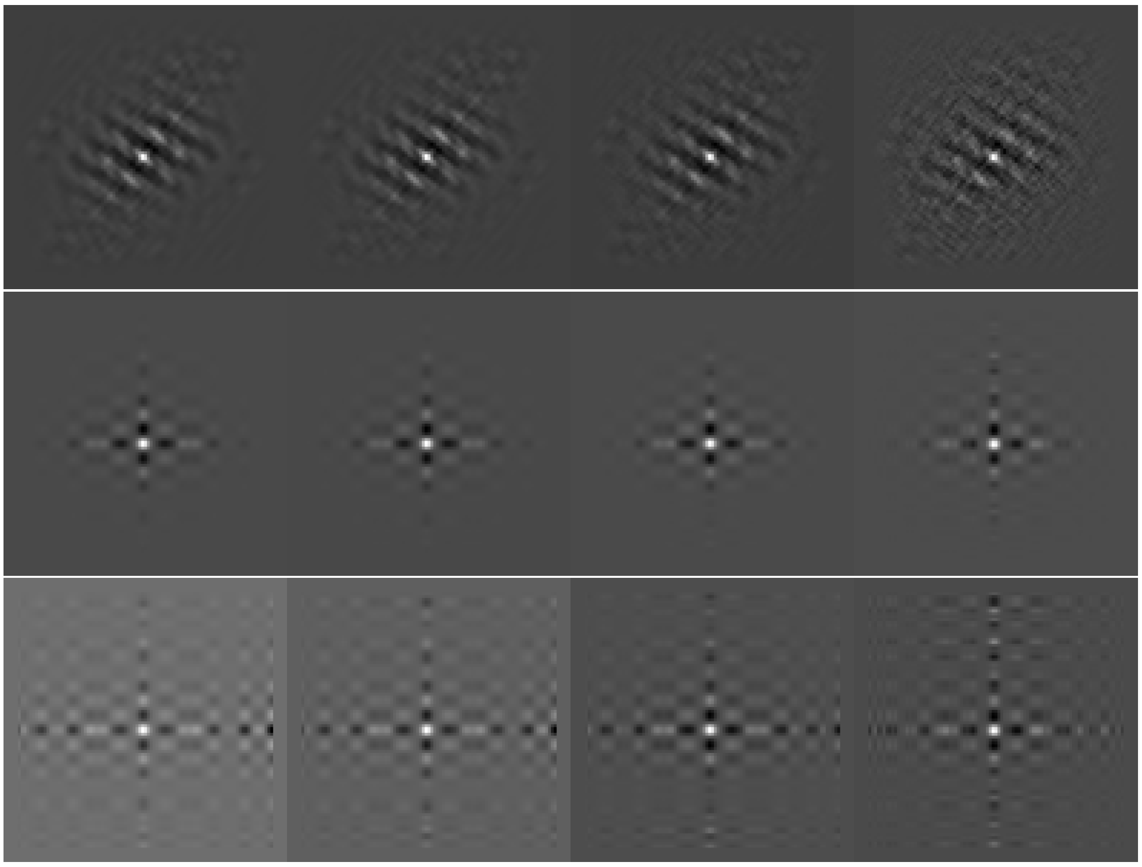
In [179] the method is improved by fitting a 1D kernel, based on the mexican hat wavelet, to the left and right singular vectors of the SVD of the (normalized) gradient correlation grid. The fitting is performed through least squares minimization and resolved using the Levenberg-Marquardt [92] algorithm. For the minimization to converge rapidly, the method is initialized with the preliminary result of applying quadratic fitting to the rank-1 approximation of the (normalized) gradient correlation (Fig. 2.8, middle row). Therefore, given either $GC(\mathbf{x})$ or $NGC(\mathbf{x})$, the method first computes its rank-1 approximation $GC_1 = \lambda_1 U_1 V_1^T$ using its SVD, and uses GC_1 to compute an initial shift v_x by linearly fitting Eq. (2.75) to its peak value. To refine the results, it fits the following kernel using $2R+1$ samples around the peak on each U and V

$$K_{1D}(x) = p_1 \left\{1 - (p_2(x - v_x))^2\right\} \frac{1}{\sqrt{2\pi}p_3} \exp\left\{-\frac{(x-v_x)^2}{2p_3^2}\right\} \tag{2.120}$$

where $[v_x, p_1, p_2, p_3]$ are the kernel parameters being optimized. As initial values for the optimization, they propose to use $p_1 = p_2 = p_3 = 1$ and $R = 10$. It should be noted that the authors evaluate two possible normalized gradient correlation rank-1 approximations. Either they compute directly the SVD of Eq. (2.119) as already explained, either they divide the rank-1 approximations of both numerator and denominator of Eq. (2.119), followed by taking again the rank-1 of the quotient, as in

$$NGC(\mathbf{x}) = \text{rank1}\left\{\frac{\text{rank1}(\mathcal{F}^{-1}\left\{G_1^* G_2\right\})}{\text{rank1}(\mathcal{F}^{-1}\left\{|G_1|^* |G_2|\right\})}\right\}. \tag{2.121}$$

This last approach, whose NGC grid is shown in the last row of Fig. 2.8, seems to offer slightly better results, although the difference is not significative.



(a) $\sigma = 0$ (b) $\sigma = 75$ (c) $\sigma = 150$ (d) $\sigma = 300$

Figure 2.8 – Gradient correlation without (top) and by using a rank-1 approximation (center) and the normalized gradient correlation after forcing rank-1 (bottom) of two images shifted by $(-3.5, 0.75)$ pixels under different random WGN ($\sigma = 0, 75, 150$ and 300) assuming 12-bit images. Both images were zero-padded up to $2N - 1$ pixels [178]. Dynamic ranges extended for visualization purposes.

2.4 Method evaluation

In this section we perform a full evaluation of the presented methods, under different parametrizations, on several shift-estimation tasks. For experimentation, and due to the applications elaborated in this thesis and presented in the following chapters, all experiments were done using a high-resolution satellite image of the city of Cannes 2.9. In a simulated environment, I_1 was generated by taking a 50×50 subimage from a random location, while I_2 was obtained by shifting the high-resolution image in the Fourier domain using Eq. (2.67) followed by taking a subimage with the same size and from the same location. All shift estimation errors were computed as the root mean squared error (RMSE) and are in pixels. This implies that if $\mathbf{v}$ and $\hat{\mathbf{v}}$ are the real shift and the estimated shift, then the error obtained is

$$E(\hat{\mathbf{v}}) = \sqrt{\frac{(v_x - \hat{v}_x)^2 + (v_y - \hat{v}_y)^2}{2}}. \tag{2.122}$$

Results presented in this evaluation are, in every case, obtained by averaging 100 realizations.

Figure 2.9 – Input image used for simulation.

Apart from the most important shift-estimation methods introduced in the previous section, five more approaches coming from the remote sensing community were also evaluated. These five methods originate from wavefront aberration estimation using Shack-Hartmann devices on extended scenes and will be thoroughly explained in the next chapter. Three of these methods (SDF, ADF and ADF2 [95]) are loosely based on correlating both images followed by fitting a conic section in the 2D neighbourhood around the peak, while ACC [151] and APC [153] are two iterative methods, the former being based on the method of Stone [159] while the latter is based on a periodic correlation technique.

Due to the extensive amount of evaluated methods and variants, we defined a nomenclature to refer to each of them. In Table 2.2 we observe all evaluated methods together with their parameters. For every method, parameter ***i*** refers to the amount of iterations (***i*** $\in [1, 2, 3, 4]$ in our experiments), ***gr*** to a gradient estimation method presented in section 2.2.6 (Table 2.3), ***int*** to an interpolation method presented in section 2.2.7 (Table 2.5), ***sp*** to the support size, ***up*** to the upsampling factor, ***win*** to the apodization window used (Table 2.4) and ***d*** to the dimensionality of the solution (***d*** $\in [1, 2]$).

For the multiscale GBSE method (MS), we first put the amount scales followed by a coma. Then we specify its configuration by putting for each scale the amount of iterations ***i*** followed by the interpolation method ***int*** used in each scale from the finest to the coarsest scale. For example, MS-3,321-IdssGh represents three scales, with a single iteration and spline interpolation on the coarsest scale, followed by two iterations and spline interpolation in the intermediate scale, and finally doing three iterations and DFT interpolation with symmetrization in the original scale, and where the image gradients are always computed using the hypomode.

Code	Parameters	Method	Reference
LS	LS-***i***-I***int***-G***gr***	Least Squares GBSE	Alg. 1
TLS	TLS-***i***-I***int***-G***gr***	Total Least Squares GBSE	Eq. (2.10)
CLS	CLS-***i***-I***int***-G***gr***	Corrected Least Squares GBSE	Eq. (2.22)
ULS	ULS-G***gr***	Bidirectional bias correction GBSE	[122]
MS	MS-***s***,***i***-I***int***G***gr***	Multiscale GBSE	Alg. 2
INT	INT-***sp***	Interpolation method	Sec. 2.2.8
PC-GUIZAR	PC-GUIZAR-***up***	Zero-fitting the cross-power spectrum	[69]
PCSTONE	PCSTONE-W***win***	Robust plane fitting on the phase difference matrix	[159]
PC-QUADFIT	PC-QUADFIT-W***win***	Quadratic fitting on the phase correlation surface	[1]
PC-GAUSSFIT	PC-GAUSSFIT-W***win***	Gaussian fitting on the phase correlation surface	[1]
PCFOO	PCFOO-W***win***	Max. of sinc approx. to phase correlation surface	[57]
PC-SINC	PC-SINC-W***win***	Sinc fitting on the phase correlation surface	[8]
PC-ESINC	PC-ESINC-W***win***	E-Sinc fitting on the phase correlation surface	[8]
PC-LCM	PC-LCM-***d***D***sp***	***d***-dimensional local center of mass of the PCS	[4]
PC-REN2010	PC-REN2010-W***win***	Difference between both side-peaks of the PCS	[134]
SS-HOGE	SS-HOGE-W***win***	Subspace method by Rank-1 approximation	[75]
SS-ROBINSON	SS-ROBINSON-W***win***	Projection-based subspace phase correlation	[138]
SS-REN2014	SS-REN2014-W***win***	Projection-based subspace gradient correlation	[135]
GC04	GC04-G***gr***	Gradient Correlation	[7]
GC11	GC11-G***gr***	Kernelized Gradient Correlation	[179]
NGC04	NGC04-G***gr***	Normalized Gradient Correlation	[7]
NGC11	NGC11-G***gr***	Kernelized Normalized Gradient Correlation	[179]
SDF	SDF-2QI	Sum of squared differences	[95]
ADF	ADF-2QI	Sum of absolute differences differences	[95]
ADF2	ADF2-2QI	Sum of squared absolute differences differences	[95]
ACC	ACC-***i***It	Adaptive Cross-Correlation	[151]
APC	APC-***i***It	Adaptive Periodic-Correlation	[153]

Table 2.2 – Method references for this evaluation section.

Code	Name
h	Hypomode
g0.3	3×3 Gaussian derivative $\sigma = 0.3$
g0.6	5×5 Gaussian derivative $\sigma = 0.6$
g1	7×7 Gaussian derivative $\sigma = 1$
sim3	3×3 Simoncelli derivative
sim5	5×5 Simoncelli derivative
sim7	7×7 Simoncelli derivative
fa3	3×3 Farid derivative
fa5	5×5 Farid derivative
fa7	7×7 Farid derivative
ch1	1st order Christmas derivative
ch2	2nd order Christmas derivative
ch3	3rd order Christmas derivative

Table 2.3 – Evaluated gradient estimation method codes ***gr***.

Code	Name
nw	No Window
ex	Image zero-padding
bm	Blackman
bh	Blackman-Harris
bl	Bartlett
bw	barthann
cw	Chebyshev
gw	Gaussian
hw	Hamming
tw	Tukey
ft	Flat-top

Table 2.4 – Window codes ***win*** used for apodization.

Code	Name
l	Bilinear interpolation
c	Bicubic interpolation
s	3rd order spline interpolation
f	FFT interpolation
d	FFT with symmetrization int.

Table 2.5 – Evaluated interpolation method codes ***int***.

As mentioned in section 2.1, when evaluating the methods in the presence of noise, we assumed white Gaussian noise uncorrelated with the signal. Simulating Ren *et al.* [134], we discretized the noise into five distinct levels (five different values for the noise standard deviation σ_N), shown in Table 2.6, and used this categorization to evaluate the methods.

Level	1	2	3	4	5
σ_N	0.000	0.005	0.015	0.025	0.055

Table 2.6 – Std. dev. of white Gaussian noise for each noise level injected to the simulated images.

In the same manner, all evaluated displacements were grouped into four categories. The reason behind this distinction is that some methods are more suited than others for each designated category. The ranges for each shift magnitude level are as shown in Table 2.7. In the first category, extremely small sub-pixel shifts were considered, as they could exist in several applications, for example the Stab-Active problem presented in Chapter 4 of this thesis. We also include as a fourth category, magnitudes larger than 1.1 pixels. By design, all non-multiscale GBSE methods as well as the phase correlation method from Stone [159] fail under this category. Due to this reason, although being shown for completeness, this shift magnitude is excluded from the general averages on every table.

Level	1	2	3	4
Range	$\|\mathbf{v}\| \leq 0.1$	$0.1 < \|\mathbf{v}\| \leq 0.5$	$0.5 < \|\mathbf{v}\| \leq 1.1$	$\|\mathbf{v}\| > 1.1$

Table 2.7 – Shift magnitude ($\|\mathbf{v}\|$) ranges for each evaluated category.

We begin our evaluation by studying the impact of gradient estimation methods on GBSE approaches.

2.4.1 Influence of gradient estimation on GBSE methods

Since GBSE methods are gradient-based, their performance obviously depends on the method used to estimate the image gradient. To this end, we evaluated several gradient estimation methods using the original least-squares approach as explained in section 2.2.6. Emulating Ren *et al.* [134], we used all possible shifts obtained by taking two values from $[-0.875, -0.75, -0.5, -0.25, -0.125, -0.07, -0.02, 0, 0.03, 0.125, 0.25, 0.5, 0.75, 0.875]$. In total, this represented 196 shifts, from which for each experiment, a random sub image was obtained and shifted in the frequency domain using the Fourier shift theorem. The average error obtained for all 196 shifts is shown in Fig. 2.10 as the noise increases (left) or by varying shift magnitude (right) , using the five noise levels shown in Table 2.6 and the four shift magnitudes of Table 2.7.

Several conclusions are drawn following the preceding experiment.

- First, the most accurate gradient estimation method for GBSE proved to be the method of Farid [53], followed by the approach of Simoncelli [154].
- Second, the kernel support size should be set larger (7×7) under higher noise scenarios, and shorter (3×3) when the SNR is high.
- Third, using $\sigma = 0.6$ for the Gaussian kernel always obtained the best results, however the performance using $\sigma = 1$ seems less affected by higher noise situations, and should be considered in those cases. A more in-depth study on Gaussian gradient estimation performance for GBSE methods is given in the following section.
- Finally, even though computationally cheap, the hypomode gradient estimation should be avoided when shift estimation is done on all possible shift magnitudes between the [-1,1] interval. Nevertheless, under low shift magnitudes and high noise, the hypomode becomes the most accurate method, as seen in Table 2.8, where

the top 10 gradient estimation methods are shown depending on both the shift magnitude and the noise.

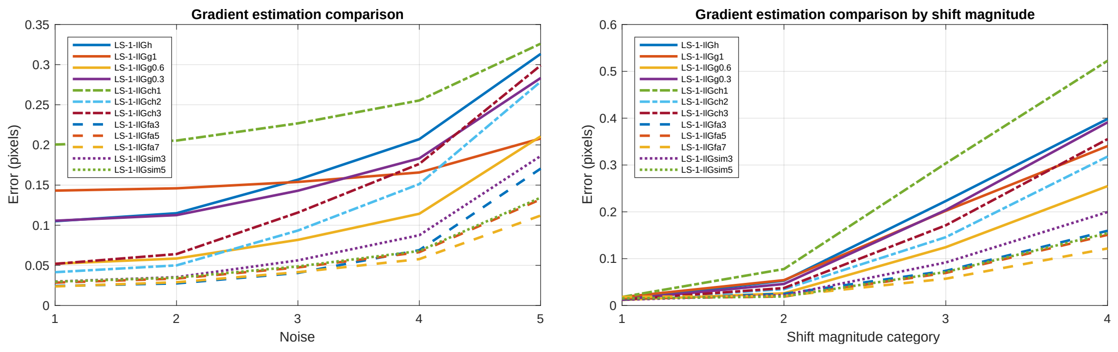


Figure 2.10 – Comparison of accuracy obtained by using different gradient estimation methods on the GBSE algorithm. **Left**: By noise level. **Right**: By shift magnitude category.

A study on Gaussian derivatives

An experiment was performed in order to study the influence of prefiltering using a Gaussian blur and computing the Gaussian derivatives to estimate the gradient. The objective was to search for the standard deviation $\hat{\sigma}$ of the Gaussian kernel h that minimized the registration error

$$\hat{\sigma} = \underset{\sigma}{\operatorname{argmin}} \left\| (A^T A)^{-1} A^T b - \mathbf{v} \right\| , \tag{2.123}$$

where

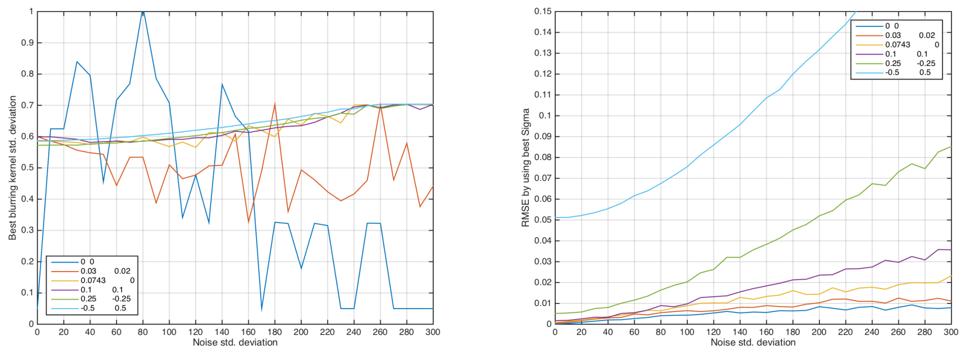
$$\mathbf{A} = \begin{pmatrix} (h_\sigma^x * I_1)(p_1) & (h_\sigma^y * I_1)(p_1) \\ \vdots & \\ (h_\sigma^x * I_1)(p_n) & (h_\sigma^y * I_1)(p_n) \end{pmatrix}, \qquad \mathbf{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}, \qquad \mathbf{b} = -\begin{pmatrix} (h_\sigma * I_t)(p_1) \\ \vdots \\ (h_\sigma * I_t)(p_n) \end{pmatrix}. \tag{2.124}$$

In the previous expression, $*$ stands for convolution and h_σ^x, h_σ^y are the derivatives of the Gaussian kernel on both dimensions, that depend on the σ parameter. By performing simulation, two images shifted by $\mathbf{v}$ were generated and WGN was added. The standard GBSE algorithm was used by computing the Gaussian derivative from the original I_1 image, and by blurring I_t with a Gaussian with the same standard deviation, as explained in section 2.2.6. The minimization was performed using the Levenberg-Marquardt nonlinear least squares optimization method. In Fig. 2.11, the evolution of $\hat{\sigma}$ is shown for each shift, as the noise of the input image increases. While for extremely small displacements (or no displacement), the $\hat{\sigma}$ varied in a random manner, for displacements already as small as $(0.0743, 0)$ and larger, the obtained values tended to be along 0.6 up to 0.7 as the noise increased.

To better understand the importance of choosing the correct $\hat{\sigma}$, in Fig. 2.12, we performed two other experiments. In Fig. 2.12(a) the average standard deviation of the top 10 best RMSEs obtained by varying the σ parameter is displayed, for each displacement. This implies to test each one of the values for the σ parameter between 0 and 1 with step size of 0.05, and obtain for each the resulting RMSE to finally compute the standard deviation of the lowest 10. For consistency, this experiment was averaged several times. As seen from the figure, the gain gets higher with the displacement magnitude. However, when the shift magnitude is under 0.1 pixels, this gain is negligible, suggesting in this case, a more relaxed selection of the σ value.

	$\|\mathbf{v}\| \leq 0.1$	$0.1 < \|\mathbf{v}\| \leq 0.5$	$0.5 < \|\mathbf{v}\| \leq 1.1$	$\|\mathbf{v}\| > 1.1$	Avg.
σ 0.000	0.0000 LS-1-IlGfa7	0.0010 LS-1-IlGfa5	0.0245 LS-1-IlGfa3	0.0733 LS-1-IlGfa3	0.0286 LS-1-IlGfa7
	0.0001 LS-1-IlGsim5	0.0010 LS-1-IlGfa7	0.0290 LS-1-IlGfa7	0.0843 LS-1-IlGfa7	0.0291 LS-1-IlGfa3
	0.0001 LS-1-IlGfa5	0.0012 LS-1-IlGsim5	0.0333 LS-1-IlGsim3	0.1025 LS-1-IlGfa5	0.0346 LS-1-IlGfa5
	0.0007 LS-1-IlGg0.6	0.0046 LS-1-IlGg0.6	0.0347 LS-1-IlGfa5	0.1054 LS-1-IlGsim5	0.0357 LS-1-IlGsim5
	0.0015 LS-1-IlGsim3	0.0052 LS-1-IlGsim3	0.0362 LS-1-IlGsim5	0.1163 LS-1-IlGsim3	0.0391 LS-1-IlGsim3
	0.0020 LS-1-IlGch3	0.0067 LS-1-IlGch3	0.0420 LS-1-IlGch2	0.1717 LS-1-IlGg0.6	0.0592 LS-1-IlGch2
	0.0033 LS-1-IlGch2	0.0128 LS-1-IlGch2	0.0583 LS-1-IlGch3	0.1785 LS-1-IlGch2	0.0603 LS-1-IlGg0.6
	0.0037 LS-1-IlGfa3	0.0150 LS-1-IlGfa3	0.0641 LS-1-IlGg0.6	0.2094 LS-1-IlGch3	0.0691 LS-1-IlGch3
	0.0045 LS-1-IlGg0.3	0.0216 LS-1-IlGg0.3	0.1305 LS-1-IlGg0.3	0.2725 LS-1-IlGh	0.1079 LS-1-IlGh
	0.0048 LS-1-IlGh	0.0231 LS-1-IlGh	0.1315 LS-1-IlGh	0.2952 LS-1-IlGg0.3	0.1130 LS-1-IlGg0.3
σ 0.005	0.0039 LS-1-IlGg0.6	0.0049 LS-1-IlGfa5	0.0260 LS-1-IlGfa3	0.0783 LS-1-IlGfa3	0.0311 LS-1-IlGfa3
	0.0041 LS-1-IlGsim3	0.0050 LS-1-IlGsim5	0.0314 LS-1-IlGfa7	0.0867 LS-1-IlGfa7	0.0323 LS-1-IlGfa7
	0.0043 LS-1-IlGch3	0.0057 LS-1-IlGfa7	0.0374 LS-1-IlGfa5	0.1051 LS-1-IlGfa5	0.0380 LS-1-IlGfa5
	0.0045 LS-1-IlGsim5	0.0061 LS-1-IlGsim3	0.0376 LS-1-IlGsim3	0.1080 LS-1-IlGsim5	0.0391 LS-1-IlGsim5
	0.0045 LS-1-IlGfa5	0.0068 LS-1-IlGch3	0.0390 LS-1-IlGsim5	0.1216 LS-1-IlGsim3	0.0423 LS-1-IlGsim3
	0.0049 LS-1-IlGch2	0.0072 LS-1-IlGg0.6	0.0508 LS-1-IlGch2	0.1774 LS-1-IlGg0.6	0.0644 LS-1-IlGg0.6
	0.0054 LS-1-IlGfa7	0.0111 LS-1-IlGch2	0.0691 LS-1-IlGg0.6	0.1923 LS-1-IlGch2	0.0648 LS-1-IlGch2
	0.0055 LS-1-IlGfa3	0.0145 LS-1-IlGfa3	0.0716 LS-1-IlGch3	0.2267 LS-1-IlGch3	0.0774 LS-1-IlGch3
	0.0059 LS-1-IlGg0.3	0.0238 LS-1-IlGg0.3	0.1372 LS-1-IlGg0.3	0.2863 LS-1-IlGh	0.1151 LS-1-IlGh
	0.0063 LS-1-IlGh	0.0264 LS-1-IlGh	0.1415 LS-1-IlGh	0.3025 LS-1-IlGg0.3	0.1173 LS-1-IlGg0.3
σ 0.015	0.0105 LS-1-IlGch2	0.0131 LS-1-IlGsim3	0.0400 LS-1-IlGfa3	0.0993 LS-1-IlGfa7	0.0435 LS-1-IlGfa7
	0.0108 LS-1-IlGch3	0.0142 LS-1-IlGfa5	0.0428 LS-1-IlGfa7	0.1134 LS-1-IlGfa3	0.0452 LS-1-IlGfa3
	0.0109 LS-1-IlGg0.6	0.0143 LS-1-IlGsim5	0.0510 LS-1-IlGfa5	0.1224 LS-1-IlGfa5	0.0502 LS-1-IlGfa5
	0.0110 LS-1-IlGsim3	0.0156 LS-1-IlGfa3	0.0527 LS-1-IlGsim5	0.1256 LS-1-IlGsim5	0.0515 LS-1-IlGsim5
	0.0110 LS-1-IlGg0.3	0.0161 LS-1-IlGfa7	0.0622 LS-1-IlGsim3	0.1563 LS-1-IlGsim3	0.0606 LS-1-IlGsim3
	0.0118 LS-1-IlGh	0.0176 LS-1-IlGg0.6	0.0960 LS-1-IlGg0.6	0.2137 LS-1-IlGg0.6	0.0845 LS-1-IlGg0.6
	0.0119 LS-1-IlGfa3	0.0178 LS-1-IlGch2	0.1080 LS-1-IlGch2	0.2705 LS-1-IlGch2	0.1017 LS-1-IlGch2
	0.0132 LS-1-IlGsim5	0.0232 LS-1-IlGch3	0.1382 LS-1-IlGch3	0.3113 LS-1-IlGch3	0.1209 LS-1-IlGch3
	0.0134 LS-1-IlGfa5	0.0362 LS-1-IlGg0.3	0.1741 LS-1-IlGg0.3	0.3223 LS-1-IlGg1	0.1434 LS-1-IlGg0.3
	0.0154 LS-1-IlGg1	0.0432 LS-1-IlGh	0.1905 LS-1-IlGg1	0.3524 LS-1-IlGg0.3	0.1445 LS-1-IlGg1
σ 0.025	0.0161 LS-1-IlGch2	0.0226 LS-1-IlGfa3	0.0592 LS-1-IlGfa7	0.1250 LS-1-IlGfa7	0.0589 LS-1-IlGfa7
	0.0162 LS-1-IlGg0.3	0.0241 LS-1-IlGfa5	0.0718 LS-1-IlGfa5	0.1552 LS-1-IlGfa5	0.0679 LS-1-IlGfa5
	0.0165 LS-1-IlGg0.6	0.0242 LS-1-IlGsim5	0.0738 LS-1-IlGsim5	0.1589 LS-1-IlGsim5	0.0694 LS-1-IlGsim5
	0.0166 LS-1-IlGsim3	0.0246 LS-1-IlGsim3	0.0744 LS-1-IlGfa3	0.1735 LS-1-IlGfa3	0.0720 LS-1-IlGfa3
	0.0168 LS-1-IlGch3	0.0267 LS-1-IlGfa7	0.1001 LS-1-IlGsim3	0.2139 LS-1-IlGsim3	0.0888 LS-1-IlGsim3
	0.0170 LS-1-IlGh	0.0313 LS-1-IlGg0.6	0.1351 LS-1-IlGg0.6	0.2720 LS-1-IlGg0.6	0.1137 LS-1-IlGg0.6
	0.0175 LS-1-IlGfa3	0.0398 LS-1-IlGch2	0.1816 LS-1-IlGch2	0.3436 LS-1-IlGg1	0.1516 LS-1-IlGch2
	0.0201 LS-1-IlGch1	0.0488 LS-1-IlGch3	0.2032 LS-1-IlGg1	0.3689 LS-1-IlGch2	0.1556 LS-1-IlGg1
	0.0205 LS-1-IlGg1	0.0533 LS-1-IlGg0.3	0.2146 LS-1-IlGch3	0.4136 LS-1-IlGch3	0.1735 LS-1-IlGch3
	0.0205 LS-1-IlGsim5	0.0552 LS-1-IlGg1	0.2242 LS-1-IlGg0.3	0.4194 LS-1-IlGg0.3	0.1783 LS-1-IlGg0.3
σ 0.055	0.0261 LS-1-IlGh	0.0512 LS-1-IlGfa5	0.1235 LS-1-IlGfa7	0.2137 LS-1-IlGfa7	0.1092 LS-1-IlGfa7
	0.0265 LS-1-IlGg0.3	0.0515 LS-1-IlGsim5	0.1530 LS-1-IlGfa5	0.2693 LS-1-IlGfa5	0.1277 LS-1-IlGfa5
	0.0274 LS-1-IlGch2	0.0535 LS-1-IlGfa7	0.1560 LS-1-IlGsim5	0.2742 LS-1-IlGsim5	0.1296 LS-1-IlGsim5
	0.0274 LS-1-IlGch1	0.0569 LS-1-IlGfa3	0.2051 LS-1-IlGfa3	0.3593 LS-1-IlGfa3	0.1626 LS-1-IlGfa3
	0.0276 LS-1-IlGg0.6	0.0616 LS-1-IlGsim3	0.2262 LS-1-IlGsim3	0.3900 LS-1-IlGsim3	0.1765 LS-1-IlGsim3
	0.0281 LS-1-IlGsim3	0.0696 LS-1-IlGg0.6	0.2548 LS-1-IlGg1	0.4156 LS-1-IlGg1	0.1942 LS-1-IlGg1
	0.0281 LS-1-IlGch3	0.0735 LS-1-IlGg1	0.2582 LS-1-IlGg0.6	0.4400 LS-1-IlGg0.6	0.1989 LS-1-IlGg0.6
	0.0292 LS-1-IlGfa3	0.0927 LS-1-IlGch2	0.3468 LS-1-IlGch2	0.5801 LS-1-IlGch2	0.2618 LS-1-IlGch2
	0.0329 LS-1-IlGg1	0.0951 LS-1-IlGg0.3	0.3530 LS-1-IlGg0.3	0.5861 LS-1-IlGg0.3	0.2652 LS-1-IlGg0.3
	0.0369 LS-1-IlGsim5	0.1008 LS-1-IlGch3	0.3733 LS-1-IlGch3	0.6156 LS-1-IlGch3	0.2794 LS-1-IlGch3
Avg.	0.0119 LS-1-IlGg0.6	0.0191 LS-1-IlGfa5	0.0572 LS-1-IlGfa7	0.1218 LS-1-IlGfa7	0.0545 LS-1-IlGfa7
	0.0122 LS-1-IlGsim3	0.0192 LS-1-IlGsim5	0.0696 LS-1-IlGfa5	0.1509 LS-1-IlGfa5	0.0637 LS-1-IlGfa5
	0.0124 LS-1-IlGch3	0.0206 LS-1-IlGfa7	0.0715 LS-1-IlGsim5	0.1544 LS-1-IlGsim5	0.0651 LS-1-IlGsim5
	0.0125 LS-1-IlGch2	0.0221 LS-1-IlGsim3	0.0740 LS-1-IlGfa3	0.1596 LS-1-IlGfa3	0.0680 LS-1-IlGfa3
	0.0128 LS-1-IlGg0.3	0.0249 LS-1-IlGfa3	0.0919 LS-1-IlGsim3	0.1996 LS-1-IlGsim3	0.0815 LS-1-IlGsim3
	0.0132 LS-1-IlGh	0.0261 LS-1-IlGg0.6	0.1245 LS-1-IlGg0.6	0.2549 LS-1-IlGg0.6	0.1044 LS-1-IlGg0.6
	0.0135 LS-1-IlGfa3	0.0348 LS-1-IlGch2	0.1459 LS-1-IlGch2	0.3181 LS-1-IlGch2	0.1278 LS-1-IlGch2
	0.0150 LS-1-IlGsim5	0.0373 LS-1-IlGch3	0.1712 LS-1-IlGch3	0.3405 LS-1-IlGg1	0.1441 LS-1-IlGch3
	0.0152 LS-1-IlGfa5	0.0460 LS-1-IlGg0.3	0.2022 LS-1-IlGg1	0.3553 LS-1-IlGch3	0.1537 LS-1-IlGg1
	0.0180 LS-1-IlGg1	0.0530 LS-1-IlGh	0.2038 LS-1-IlGg0.3	0.3911 LS-1-IlGg0.3	0.1634 LS-1-IlGg0.3

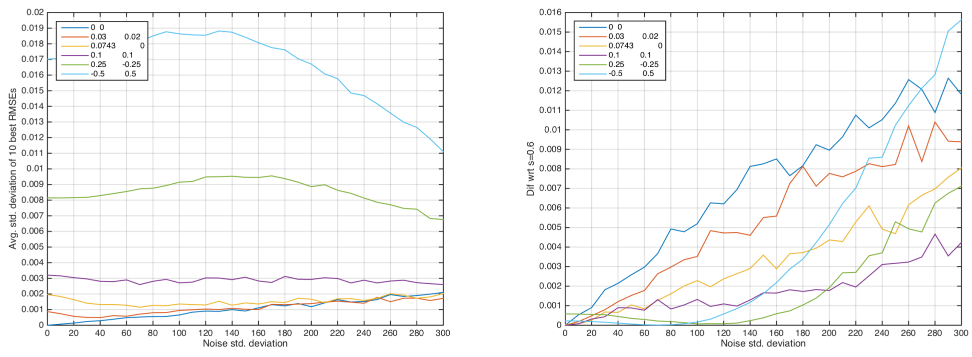
Table 2.8 – Average error per shift and magnitude of top 10 evaluated Least Squares GBSE methods using all gradient estimation approaches. **Rows**: five noise levels. **Columns**: four shift magnitudes. Note that the last column displays the top ten methods of the average for each noise level excluding the last shift magnitude level.



(a) $\hat{\sigma}$ per noise (b) RMSE obtained using $\hat{\sigma}$

Figure 2.11 – $\hat{\sigma}$ values obtained based on noise and shift. Standard deviation of noise is considering 12-bit images.

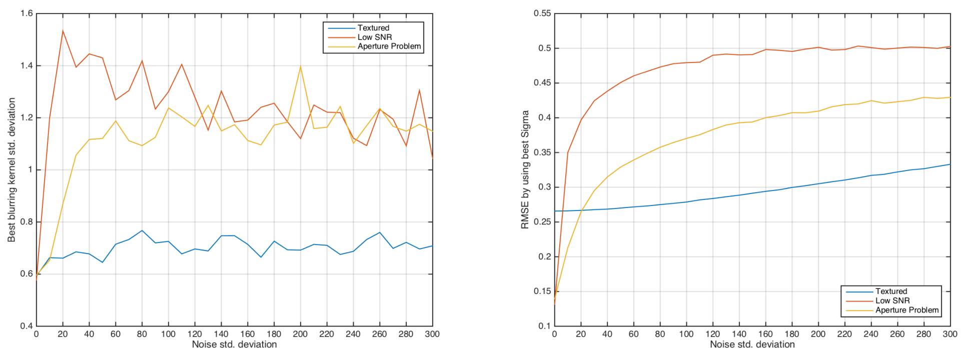
Finally, since $\sigma = 0.6$ usually achieved good results, a comparison between the best $\hat{\sigma}$ and $\sigma = 0.6$ was made in Fig. 2.12(b). Interestingly, by fixing $\sigma = 0.6$, higher accuracy gains could be obtained when the displacements are of small magnitude (lower than 0.1 pixels). However, when the displacement is high enough (in our case half a pixel in magnitude), under low SNR values (noise std. dev. bigger than 150), the accuracy gain by selecting the correct $\hat{\sigma}$ could be as high as 0.016 pixels. This implies that for low displacement magnitudes, lower σ values should be used (around $\sigma = 0.45$); for medium shift magnitudes (between 0.1 and 0.3); $\sigma = 0.6$ should be used, and for high displacements, the decision to increase the σ relies on the noise level.



(a) Average standard deviation of ten best RMSEs per shift (b) Difference between RMSE obtained using $\hat{\sigma}$ and using the $\sigma = 0.6$

Figure 2.12 – Influence of using the correct σ for the Gaussian kernel on the results

Another experiment was performed to confirm the $\hat{\sigma}$ values obtained under different image conditions. Three image types were selected, namely a textured image, a low SNR image, and an image having the well-known aperture problem (the gradient direction is mostly uniform across the image). Also, larger shifts up to $\|v\| = 1$ were included for this test. Results averaging all evaluated shifts are displayed in Fig. 2.13. It seems that for well-textured images, values of σ between 0.6 and 0.8 achieved the best performance, however for more complicated images, higher values achieved better results. This makes sense, since in a case where high precision is not possible, considerably blurring the image decreases the maximum shift returned and therefore the maximum error. This can be seen in Fig. 2.13(b) where the average errors reach almost half the magnitude and the largest evaluated shift. It has to be noted that for each noise amount, each problematic

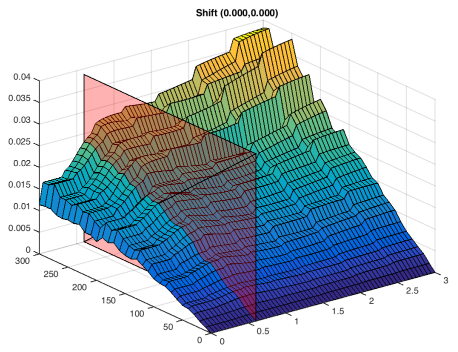


(a) $\hat{\sigma}$ per noise (b) RMSE of methods using $\hat{\sigma}$

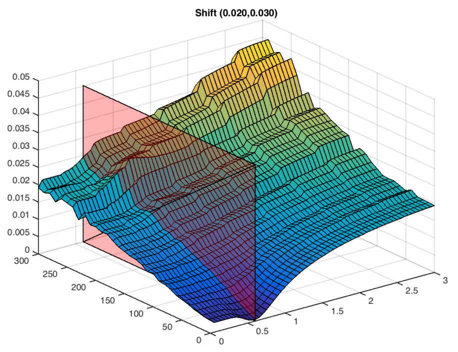
Figure 2.13 – $\hat{\sigma}$ values obtained based on problematic case and noise.

case and each simulated shift, a total of 1000 experiments were performed. The $\hat{\sigma}$ displayed on the figures is the median while the RMSE shown is the average RMSE of these experiments.

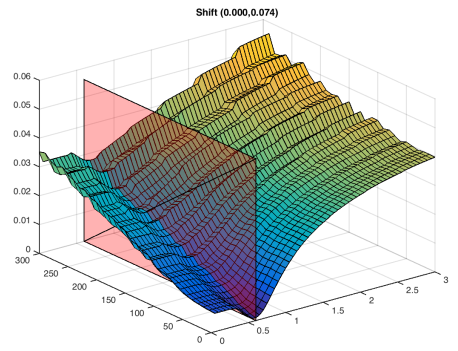
Finally, to give more insight about the progression of the error based on the blurring kernel standard deviation used and on the noise, Fig. 2.14 shows the RMSE, in pixels, depending on the underlying shift to be estimated. As it can be seen from the graphs, as the noise increases, also does the error, as expected, and the $\hat{\sigma}$ that minimizes this error seems to be around 0.6, growing as high as 0.7 when the noise becomes high. Another interesting result is that as the displacement grows, the fact that values of $\hat{\sigma}$ around 0.6 are the best becomes more apparent. However, when the shift is small enough (lower than 0.1 pixel), other $\hat{\sigma}$ values should be used, as remarked in the previous test. Furthermore, for the case of high shift magnitude and high noise, the estimation error becomes higher, so accurately setting σ in these cases could be justified by the accuracy gain.



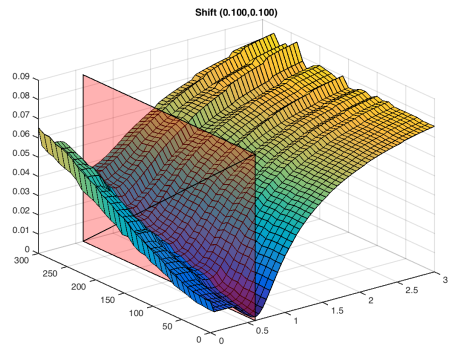
(a) Shift (0,0)



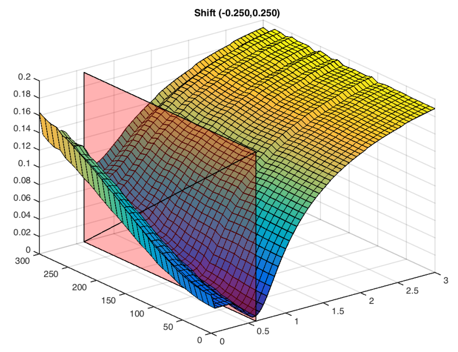
(b) Shift (0.02, 0.03)



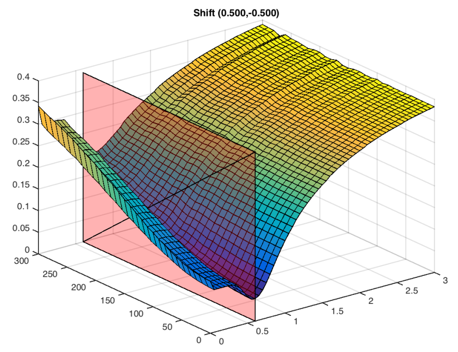
(c) Shift (0, 0.074)



(d) Shift (0.1, 0.1)



(e) Shift (-0.25, 0.25)



(f) Shift (0.5, -0.5)

Figure 2.14 – Evolution of RMSEs obtained based on $\hat{\sigma}$ values (from 0 to 3) and noise (from 0 to 300] for each shift. The red rectangle shows, as a reference, $\sigma = 0.6$.

2.4.2 Robustness to noise

One of the most important difficulties in shift estimation is related to the presence of noise on the input images. In this section, we evaluate the effects of noise on each method in a simulated environment. For each error value shown, an average of 100 experiments was performed, each time generating a new noise realization and using a different random subimage. Due to the randomness of the experiment, potential content-less images or aperture problem cases could exist. However, by using both validation methods presented in the next chapter, namely the Cramer-Rao lower bound and the Eigenratio score, these cases were excluded from the results, and the computed averages were based on valid cases only. Note also that by selecting random subimages, two experiments with injected noise sharing the same standard deviation could still have different SNRs. This was on purpose, to evaluate the versatility of each method under potentially plausible situations.

As proved by both Robinson [139] and Pham [119], the negative effect of noise is stressed as the displacement to estimate gets larger. For this reason, we evaluated the performance of every approach by varying noise and displacement magnitude conditions, based on the noise levels shown in Table 2.6. Figure 2.15 shows an example of two landscapes, the first with high photon count and the second with average photon count, under each simulated noise level. While in the first landscape, the effects of the noise start to be noted from the third noise level, for the second case it is already noticeable on the second level.

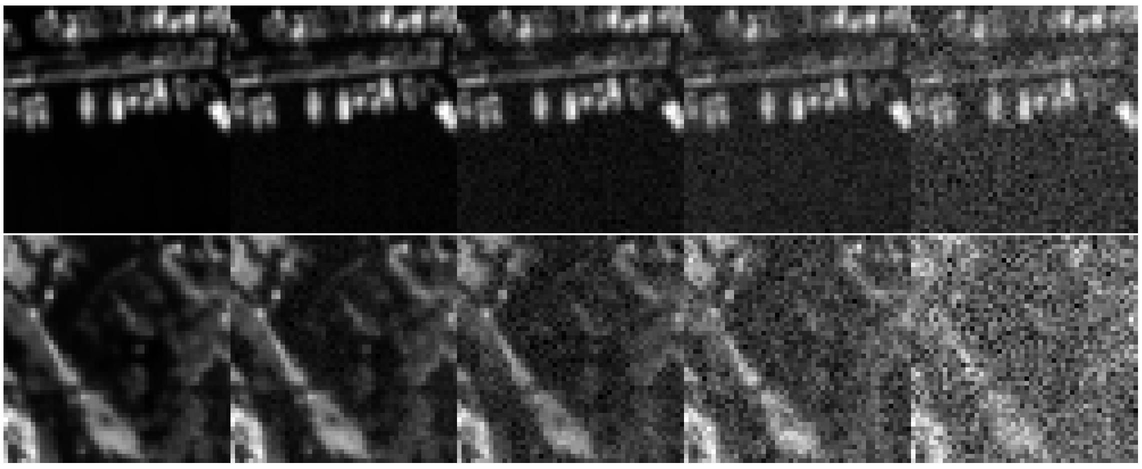

(a) Level 1 (b) Level 2 (c) Level 3 (d) Level 4 (e) Level 5

Figure 2.15 – Sample images for the five analyzed noise levels. **Top**: high constrasted image (high SNR). **Bottom**: intermediate contrasted image (average SNR). Dynamic ranges extended for visualization purposes.

To summarize results, we display in Table 2.9 the top 10 methods together with their average errors for each evaluated shift magnitude category and every noise level analyzed. Also, the averages by noise level and by shift magnitude were included in the results. Since in general each top 10 is dominated by the different variants from a single method, this table does not seem too informative. However, some interesting conclusions could be inferred. First, the best methods are always gradient-based, and the amount of iterations and/or scales depends on the shift magnitude. For the lowest shift magnitude studied, using a single or two iterations of the original LS or the TLS method are optimal. For the second analyzed magnitude category, the best results were obtained without requiring a multiscale approach, however several iterations should be performed (three or four). Finally, for the remaining shift magnitudes, a multiscale approach is mandatory and the configuration that seems to achieve the best results is to start with a single iteration on the broadest scale and increasing the amount of iterations by one for each subsequent scale progressively. Also, for the finest scale, the best interpolation scheme should be used (FFT with or without symmetrization or spline). Finally, for short shift-magnitudes, the gradients with the smallest supports are recommended. In particular, the hypomode stands out from the rest. As the shift magnitude increases, both 3×3 Farid or Simoncelli gradient estimation methods attain better scores under lower SNR scenarios. As a special case, under infinite SNR situations, larger gradient estimation kernels seem to achieve the best accuracies.

To give a better comparison between the studied methods, we took the best performing variant for each of them and compared them in the same manner. Therefore, the top 10 for the best variant of each method for each shift magnitude/noise case is displayed in Table 2.10. From this table several conclusions arise. First and most importantly, GBSE methods systematically improve over other approaches, and the difference in performance could be up to factor of five, and is more remarked on lower SNR scenarios. Secondly, for shifts magnitudes between 0.1 and 1.1, the bidirectional bias corrected method of Pham *et al.* makes the top 10 by using a single iteration and no resampling, yielding itself as the best candidate under limited computational time constraints. For shifts lower than 0.1, the total least squares method with one iteration is recommended for fast processing times. Third, gradient correlation approaches improve over phase-correlation-based methods, particularly under lower SNR where in general phase-correlation methods tend to fail. Finally, under low noise, apodization methods improved phase correlation approaches systematically and Hamming, Blackman or Tukey windows achieved the best results. However, as the noise gets higher, some methods achieved higher accuracies without apodization. This is because noise affects more the final accuracy than the fake edges generated by the false periodicity assumption. Therefore, having more pixels from which to estimate the shift, due to avoiding windowing the images in the spatial domain, helps reduce the noise influence, yielding improved results.

Predefined set of methods

While for practical reasons, it is interesting to know the top 10 methods for each case, we get no knowledge of the performance for each method for all situations. To this end, we preselected 13 methods, and evaluated their performance for every noise/shift magnitude case. Three fast single iteration methods were selected, namely the traditional least squares approach and the total least squares both using one iteration and the hypomode gradient estimation, together with the bidirectional bias correction method of Pham [122] using the 3×3 Farid gradient estimation. Also, both the least squares and the total least squares methods using four iterations with the same gradient estimation method were included in the comparison. As for GBSE multiscale approaches we included two methods. One using two scales, one iteration on the coarse scale and three on the original scale (3,1 iteration pattern), using spline interpolation on the last scale, and another method using three scales and a (3,2,1) iteration pattern for which the employed interpolation method in the final scale was FFT with symmetrization. Both methods used the 3×3 Farid gradient estimation, which proved before to be the most robust against noise. For phase-correlation methods, we included in the evaluation the approach proposed by Guizar-Sicairos [69] using an upsampling value of 2000, the method of Stone [159] and the Sinc fitting approach, both by applying a hamming window for apodization purposes, and the method of Hoge [75] with a Tukey window. Finally, two gradient correlation approaches were included, both using Gaussian derivatives with $\sigma = 0.6$. They are the original method [7] and the more recent approach by Tzimiropoulos [179].

In Table 2.11 we show the results for each predefined method on each condition

	$\|\mathbf{v}\| \leq 0.1$	$0.1 < \|\mathbf{v}\| \leq 0.5$	$0.5 < \|\mathbf{v}\| \leq 1.1$	$\|\mathbf{v}\| > 1.1$	Avg.
	0.0000 MS-3,321-IdssGfa7	0.0000 MS-2,31-IdcGfa7	0.0000 MS-3,321-IdssGfa7	0.0000 MS-3,321-IdssGfa7	0.0000 MS-3,321-IdssGfa7
	0.0000 MS-2,21-IdcGfa7	0.0000 MS-2,21-IdcGfa7	0.0000 MS-2,31-IdcGfa7	0.0000 MS-3,321-IfssGfa7	0.0000 MS-2,31-IdcGfa7
	0.0000 MS-2,31-IdcGfa7	0.0000 MS-3,321-IdssGfa7	0.0000 MS-2,21-IdcGfa7	0.0000 MS-3,321-IdssGfa5	0.0000 MS-2,21-IdcGfa7
	0.0000 CLS-3-IdGfa7	0.0000 LS-3-IdGfa7	0.0000 LS-4-IdGfa7	0.0000 MS-3,321-IdssGsim5	0.0000 LS-4-IdGfa7
σ	0.0000 LS-3-IdGfa7	0.0000 TLS-3-IdGfa7	0.0000 TLS-4-IdGfa7	0.0000 MS-3,321-IfssGfa5	0.0000 TLS-4-IdGfa7
0.000	0.0000 TLS-3-IdGfa7	0.0000 TLS-4-IdGfa7	0.0000 LS-3-IdGfa7	0.0000 MS-3,321-IfssGsim5	0.0000 LS-3-IdGfa7
	0.0000 CLS-4-IdGfa7	0.0000 LS-4-IdGfa7	0.0000 TLS-3-IdGfa7	0.0001 MS-3,321-IdssGg0.6	0.0000 TLS-3-IdGfa7
	0.0000 LS-4-IdGfa7	0.0000 MS-2,31-IdcGsim5	0.0000 MS-2,31-IfcGfa7	0.0005 MS-3,321-IsssGfa7	0.0000 MS-2,31-IfcGfa7
	0.0000 TLS-4-IdGfa7	0.0000 MS-2,21-IdcGsim5	0.0000 MS-2,21-IfcGfa7	0.0006 MS-3,321-IdssGch3	0.0000 MS-2,21-IfcGfa7
	0.0000 LS-2-IdGfa7	0.0000 MS-2,21-IdcGfa5	0.0000 MS-3,321-IdssGsim5	0.0007 MS-3,321-IfssGg0.6	0.0000 MS-2,31-IdcGsim5
	0.0031 LS-2-IsGh	0.0035 CLS-4-IdGh	0.0035 MS-3,321-IdssGh	0.0044 MS-3,321-IfssGfa3	0.0034 TLS-4-IdGh
	0.0032 TLS-2-IsGh	0.0035 TLS-4-IdGh	0.0035 MS-3,321-IdssGch2	0.0045 MS-3,321-IdssGfa3	0.0034 MS-3,321-IdssGh
	0.0032 LS-2-IfGh	0.0035 LS-4-IdGh	0.0035 TLS-4-IdGh	0.0045 MS-3,321-IfssGsim3	0.0035 MS-2,31-IdcGh
	0.0032 CLS-2-IsGh	0.0036 MS-3,321-IdssGh	0.0035 MS-2,31-IdcGch2	0.0045 MS-3,321-IdssGsim3	0.0035 MS-3,321-IdssGch2
σ	0.0032 LS-2-IdGh	0.0036 CLS-3-IdGh	0.0035 MS-2,31-IdcGg0.3	0.0046 MS-3,321-IdssGg0.3	0.0035 MS-2,31-IdcGch2
0.005	0.0032 TLS-2-IfGh	0.0037 MS-2,31-IdcGh	0.0035 MS-2,21-IdcGch2	0.0046 MS-3,321-IsssGfa3	0.0035 MS-2,21-IdcGch2
	0.0032 CLS-2-IfGh	0.0037 LS-3-IdGch2	0.0035 MS-3,321-IdssGg0.3	0.0046 MS-3,321-IdssGg0.6	0.0036 LS-4-IdGch2
	0.0032 CLS-2-IdGh	0.0037 CLS-3-IdGch2	0.0035 MS-2,31-IdcGh	0.0047 MS-3,321-IsssGsim3	0.0036 LS-4-IdGh
	0.0032 TLS-2-IdGh	0.0037 LS-4-IdGch2	0.0035 LS-4-IdGch2	0.0048 MS-3,321-IfssGg0.6	0.0036 LS-4-IdGg0.3
	0.0032 LS-3-IdGh	0.0037 CLS-4-IdGch2	0.0036 LS-4-IdGg0.3	0.0050 MS-3,321-IsssGg0.6	0.0036 MS-3,321-IdssGg0.3
	0.0092 LS-2-IsGh	0.0105 TLS-4-IdGh	0.0118 MS-3,321-IdssGh	0.0220 MS-3,321-IfssGfa3	0.0113 TLS-4-IdGh
	0.0092 LS-2-IdGh	0.0105 CLS-4-IdGh	0.0118 MS-3,321-IfssGh	0.0221 MS-3,321-IdssGfa3	0.0113 MS-3,321-IdssGh
	0.0093 LS-2-IfGh	0.0106 TLS-4-IfGh	0.0121 MS-3,321-IsssGh	0.0221 MS-3,321-IfssGsim3	0.0114 TLS-4-IfGh
	0.0095 TLS-2-IdGh	0.0107 CLS-3-IdGh	0.0123 MS-3,321-IdssGch2	0.0221 MS-3,321-IdssGsim3	0.0114 MS-3,321-IfssGh
σ	0.0095 TLS-2-IfGh	0.0107 CLS-4-IfGh	0.0125 MS-3,321-IdssGg0.3	0.0222 MS-3,321-IsssGfa3	0.0116 MS-3,321-IdssGg0.3
0.015	0.0095 TLS-2-IsGh	0.0107 TLS-3-IsGh	0.0126 MS-3,321-IfssGch2	0.0222 MS-3,321-IsssGsim3	0.0117 MS-3,321-IdssGch1
	0.0095 LS-2-IcGch1	0.0107 CLS-4-IdGg0.3	0.0127 MS-3,321-IdssGch1	0.0246 MS-3,321-IdssGg0.3	0.0117 MS-2,31-IdcGh
	0.0095 LS-3-IdGh	0.0107 TLS-3-IdGh	0.0127 MS-3,321-IsssGch2	0.0247 MS-3,321-IfssGg0.3	0.0117 MS-3,321-IsssGh
	0.0096 LS-2-IlGch1	0.0107 MS-3,321-IdssGg0.3	0.0127 MS-3,321-IfssGg0.3	0.0250 MS-3,321-IsssGg0.3	0.0117 MS-2,31-IdcGg0.3
	0.0096 CLS-2-IcGch1	0.0107 LS-4-IfGch2	0.0128 LS-4-IfGfa3	0.0269 MS-3,321-IdssGch2	0.0117 MS-2,31-IdcGch2
	0.0145 TLS-1-IlGh	0.0179 MS-3,321-IfssGh	0.0208 MS-3,321-IfssGh	0.0254 MS-3,321-IsssGsim3	0.0200 MS-3,321-IfssGh
	0.0147 LS-2-IdGh	0.0180 MS-3,321-IdssGh	0.0209 MS-3,321-IdssGh	0.0256 MS-3,321-IfssGsim3	0.0200 MS-3,321-IdssGh
	0.0148 LS-2-IfGh	0.0183 MS-3,321-IsssGh	0.0211 MS-3,321-IsssGh	0.0256 MS-3,321-IdssGsim3	0.0203 MS-3,321-IsssGh
	0.0148 LS-2-IsGh	0.0188 MS-2,31-IscGh	0.0220 MS-2,31-IdcGsim3	0.0258 MS-3,321-IsssGfa3	0.0209 MS-3,321-IdssGch1
σ	0.0149 LS-1-IlGg0.3	0.0189 MS-2,31-IdcGh	0.0220 MS-2,31-IfcGsim3	0.0261 MS-3,321-IfssGfa3	0.0209 MS-3,321-IfssGch1
0.025	0.0150 LS-2-IsGch1	0.0190 MS-2,31-IfcGh	0.0220 MS-2,31-IfcGfa3	0.0261 MS-3,321-IdssGfa3	0.0211 MS-2,31-IfcGsim3
	0.0150 LS-2-IfGch1	0.0190 TLS-4-IsGh	0.0220 MS-2,31-IdcGfa3	0.0278 MS-3,321-IsssGg0.3	0.0211 MS-2,31-IdcGsim3
	0.0150 LS-2-IdGch1	0.0191 MS-3,321-IdssGch1	0.0220 MS-2,31-IscGsim3	0.0283 MS-3,321-IdssGg0.3	0.0212 MS-2,31-IfcGfa3
	0.0150 LS-2-IcGch1	0.0192 TLS-4-IdGh	0.0220 MS-3,321-IdssGch1	0.0284 MS-3,321-IfssGg0.3	0.0212 MS-2,31-IdcGfa3
	0.0150 LS-2-IlGch1	0.0193 MS-3,321-IfssGch1	0.0221 MS-2,31-IscGfa3	0.0330 MS-3,321-IdssGch2	0.0212 MS-2,31-IscGsim3
	0.0219 LS-1-IlGh	0.0476 LS-4-IsGfa3	0.0461 MS-2,31-IscGfa3	0.0687 MS-3,321-IfssGfa3	0.0475 LS-4-IfGfa3
	0.0225 TLS-1-IlGh	0.0476 LS-4-IfGfa3	0.0462 MS-2,31-IfcGfa3	0.0688 MS-3,321-IdssGfa3	0.0475 LS-4-IdGfa3
	0.0230 LS-1-IlGch1	0.0476 LS-4-IdGfa3	0.0462 MS-2,31-IdcGfa3	0.0689 MS-3,321-IsssGfa3	0.0476 MS-2,31-IfcGsim3
	0.0232 LS-2-IfGh	0.0477 LS-4-IlGsim3	0.0463 MS-2,31-IscGsim3	0.0706 MS-3,321-IfssGsim3	0.0476 MS-2,31-IdcGsim3
σ	0.0232 LS-2-IdGh	0.0477 LS-4-IsGsim3	0.0464 MS-2,31-IfcGsim3	0.0707 MS-3,321-IsssGsim3	0.0476 MS-2,31-IscGsim3
0.055	0.0232 LS-2-IsGh	0.0478 LS-4-IlGfa3	0.0464 MS-2,31-IdcGsim3	0.0707 MS-3,321-IdssGsim3	0.0477 LS-4-IsGfa3
	0.0233 LS-2-IlGh	0.0478 LS-4-IfGsim3	0.0470 MS-3,321-IsssGfa3	0.0889 MS-3,321-IsssGfa5	0.0478 MS-2,31-IfcGfa3
	0.0234 LS-2-IcGh	0.0478 LS-4-IdGsim3	0.0471 MS-3,321-IsssGsim3	0.0891 MS-3,321-IfssGfa5	0.0478 MS-2,31-IdcGfa3
	0.0236 LS-1-IlGg0.3	0.0479 LS-4-IcGsim3	0.0472 MS-3,321-IfssGfa3	0.0891 MS-3,321-IdssGfa5	0.0478 MS-2,31-IscGfa3
	0.0241 CLS-1-IlGch1	0.0480 TLS-4-IlGsim3	0.0472 MS-3,321-IdssGfa3	0.0892 MS-3,321-IsssGsim5	0.0483 LS-4-IfGsim3
	0.0101 LS-2-IsGh	0.0167 LS-4-IdGfa3	0.0170 MS-2,31-IdcGfa3	0.0282 MS-3,321-IfssGfa3	0.0170 LS-4-IdGfa3
	0.0102 LS-2-IdGh	0.0168 LS-4-IdGsim3	0.0170 MS-2,31-IdcGsim3	0.0282 MS-3,321-IdssGfa3	0.0170 MS-2,31-IdcGsim3
	0.0102 LS-2-IfGh	0.0168 LS-4-IfGfa3	0.0170 MS-2,31-IfcGfa3	0.0284 MS-3,321-IfssGsim3	0.0170 MS-2,31-IfcGsim3
	0.0110 LS-3-IdGh	0.0168 LS-4-IfGsim3	0.0170 MS-2,31-IfcGsim3	0.0284 MS-3,321-IdssGsim3	0.0170 LS-4-IfGfa3
Avg.	0.0110 LS-3-IfGh	0.0171 LS-3-IdGfa3	0.0172 MS-3,321-IdssGh	0.0286 MS-3,321-IsssGfa3	0.0171 MS-2,31-IdcGfa3
	0.0110 LS-2-IlGh	0.0171 MS-2,31-IfcGsim3	0.0173 MS-2,31-IscGsim3	0.0289 MS-3,321-IsssGsim3	0.0171 MS-2,31-IfcGfa3
	0.0111 LS-2-IcGh	0.0171 MS-2,31-IdcGsim3	0.0174 MS-3,321-IdssGsim3	0.0331 MS-3,321-IsssGsim5	0.0171 LS-4-IdGsim3
	0.0111 TLS-1-IlGh	0.0172 MS-2,31-IdcGh	0.0174 MS-3,321-IdssGfa3	0.0331 MS-3,321-IdssGsim5	0.0172 LS-4-IfGsim3
	0.0111 LS-2-IcGch1	0.0172 LS-3-IfGfa3	0.0174 MS-2,31-IscGfa3	0.0332 MS-3,321-IfssGsim5	0.0172 MS-2,21-IdcGsim3
	0.0112 LS-3-IsGh	0.0172 MS-2,31-IdcGg0.3	0.0174 MS-3,321-IfssGsim3	0.0332 MS-3,321-IsssGfa5	0.0173 MS-2,21-IfcGsim3

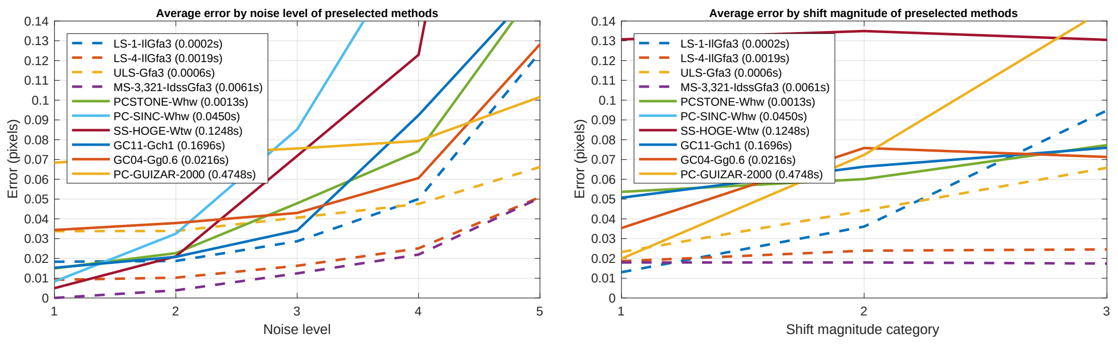
Table 2.9 – Average error per shift and magnitude of top 10 evaluated methods using all variants. **Rows**: five noise levels. **Columns**: four shift magnitudes. Note that the last column displays the top ten methods of the average for each noise level excluding the last shift magnitude level.

	$\|\mathbf{v}\| \leq 0.1$	$0.1 < \|\mathbf{v}\| \leq 0.5$	$0.5 < \|\mathbf{v}\| \leq 1.1$	$\|\mathbf{v}\| > 1.1$	Avg.
	0.0000 MS-3,321-IdssGfa7	0.0000 MS-2,31-IdcGfa7	0.0000 MS-3,321-IdssGfa7	0.0000 MS-3,321-IdssGfa7	0.0000 MS-3,321-IdssGfa7
	0.0000 CLS-3-IdGfa7	0.0000 LS-3-IdGfa7	0.0000 LS-4-IdGfa7	0.0196 PC-SINC-Whw	0.0000 LS-4-IdGfa7
	0.0000 LS-3-IdGfa7	0.0000 TLS-3-IdGfa7	0.0000 TLS-4-IdGfa7	0.0231 SS-HOGE-Wbl	0.0000 TLS-4-IdGfa7
	0.0000 TLS-3-IdGfa7	0.0000 CLS-4-IdGfa5	0.0011 CLS-4-IdGg0.6	0.0835 GC04-Gch3	0.0004 CLS-4-IdGg0.6
σ	0.0009 GC11-Gch3	0.0040 SS-HOGE-Wbw	0.0068 SS-HOGE-Wbw	0.1340 GC11-Gch3	0.0044 SS-HOGE-Wbw
0.000	0.0012 PCSTONE-Whw	0.0047 PC-SINC-Wbw	0.0083 PC-SINC-Wbw	0.4176 SDF-2QI	0.0049 PC-SINC-Wbw
	0.0012 SS-ROBINSON-Wnw	0.0119 GC11-Gch3	0.0276 GC11-Gch3	0.4607 ADF-2QI	0.0135 GC11-Gch3
	0.0014 PC-SINC-Wtw	0.0126 PCSTONE-Whw	0.0311 PCSTONE-Whw	0.5105 ADF2-2QI	0.0150 PCSTONE-Whw
	0.0016 NGC11-G5	0.0318 ULS-Gfa7	0.0476 GC04-G6	0.5868 CLS-4-IdGfa7	0.0324 ULS-Gfa7
	0.0038 INT-3	0.0399 INT-3	0.0576 ULS-Gg0.3	0.5933 PCFOO	0.0333 GC04-Gg1
	0.0031 LS-2-IsGh	0.0035 CLS-4-IdGh	0.0035 MS-3,321-IdssGh	0.0044 MS-3,321-IfssGfa3	0.0034 TLS-4-IdGh
	0.0031 MS-2-LS2-IsGh	0.0035 TLS-4-IdGh	0.0035 TLS-4-IdGh	0.0458 SS-HOGE-Whw	0.0034 MS-3,321-IdssGh
	0.0032 TLS-2-IsGh	0.0035 LS-4-IdGh	0.0035 LS-4-IdGch2	0.0477 PC-SINC-Whw	0.0036 LS-4-IdGch2
	0.0032 CLS-2-IsGh	0.0035 MS-2-LS4-IdGh	0.0060 CLS-4-IdGg0.6	0.0895 GC04-Gch3	0.0047 CLS-4-IdGg0.6
σ	0.0063 PC-GUIZAR-100	0.0159 GC11-Gch3	0.0225 SS-HOGE-Wtw	0.1545 GC11-Gch3	0.0192 GC11-Gch3
0.005	0.0063 GC11-Gch2	0.0189 PCSTONE-Whw	0.0277 PC-SINC-Wtw	0.6072 CLS-4-IdGfa7	0.0210 SS-HOGE-Wtw
	0.0063 INT-3	0.0199 SS-HOGE-Wtw	0.0355 GC11-Gch3	0.6188 PCFOO	0.0227 PCSTONE-Whw
	0.0074 GC04-Gg1	0.0301 ULS-Gfa7	0.0367 PCSTONE-Whw	1.3427 APC-2It	0.0249 PC-SINC-Wtw
	0.0082 NGC04-Gg1	0.0304 PC-SINC-Wtw	0.0558 GC04-G6	1.8574 ACC-3It	0.0327 ULS-Gsim5
	0.0082 PCSTONE-Wnw	0.0372 INT-3	0.0564 ULS-Gg0.3	1.9387 INT-5	0.0376 GC04-Gg1
	0.0092 LS-2-IsGh	0.0105 TLS-4-IdGh	0.0118 MS-3,321-IdssGh	0.0220 MS-3,321-IfssGfa3	0.0113 TLS-4-IdGh
	0.0092 MS-2-LS2-IsGh	0.0105 CLS-4-IdGh	0.0128 LS-4-IfGfa3	0.0886 GC04-Gch2	0.0113 MS-3,321-IdssGh
	0.0095 TLS-2-IdGh	0.0107 MS-3,321-IdssGg0.3	0.0135 TLS-4-IfGh	0.1002 PC-ESINC-Wtw	0.0121 LS-4-IdGfa3
	0.0096 CLS-2-IcGch1	0.0107 LS-4-IfGch2	0.0198 CLS-4-IfGg0.6	0.1300 SS-HOGE-Wtw	0.0146 CLS-4-IdGg0.6
σ	0.0113 PC-GUIZAR-2000	0.0266 GC11-Gg0.3	0.0546 GC11-Gg0.6	0.1893 GC11-Gch3	0.0337 GC11-Gg0.6
0.015	0.0136 GC04-G6	0.0400 PC-REN2010-Wex	0.0593 ULS-Gch1	0.5603 PCFOO	0.0406 ULS-Gfa3
	0.0160 PCSTONE-Wex	0.0419 PCSTONE-Wtw	0.0599 PCSTONE-Whw	0.6411 CLS-4-IdGfa7	0.0430 GC04-Gg0.6
	0.0171 ULS-Gfa3	0.0419 ULS-Gfa5	0.0634 GC04-Gg0.6	1.8438 ACC-3It	0.0479 PCSTONE-Whw
	0.0172 INT-3	0.0432 INT-3	0.0792 SS-HOGE-Wtw	2.0047 INT-5	0.0624 SS-HOGE-Wnw
	0.0191 GC11-Gg0.3	0.0490 SS-HOGE-Wnw	0.0911 PC-ESINC-Wtw	2.1902 PCSTONE-Wbw	0.0662 INT-3
	0.0145 TLS-1-IlGh	0.0179 MS-3,321-IfssGh	0.0208 MS-3,321-IfssGh	0.0254 MS-3,321-IsssGsim3	0.0200 MS-3,321-IfssGh
	0.0147 LS-2-IdGh	0.0190 TLS-4-IsGh	0.0237 LS-4-IdGfa3	0.1044 GC04-Gch3	0.0214 LS-4-IdGfa3
	0.0147 MS-2-LS2-IdGh	0.0196 CLS-4-IdGg0.3	0.0273 TLS-4-IfGh	0.1051 PC-ESINC-Wnw	0.0215 TLS-4-IdGh
	0.0153 CLS-2-IfGch1	0.0201 LS-4-IfGsim3	0.0334 CLS-4-IfGg1	0.1694 GC11-Gch2	0.0262 CLS-4-IdGg0.6
σ	0.0196 PC-GUIZAR-2000	0.0437 ULS-Gfa5	0.0715 ULS-Gsim3	0.2217 SS-HOGE-Wtw	0.0475 ULS-Gfa3
0.025	0.0225 PCSTONE-Wex	0.0509 PC-REN2010-Wex	0.0793 GC11-G6	0.6126 PCFOO	0.0606 GC04-Gg0.6
	0.0268 GC04-Gg1	0.0559 GC11-Gg0.6	0.0806 GC04-Gg1	0.6499 CLS-4-IdGfa7	0.0649 PCSTONE-Wtw
	0.0268 ULS-Gfa3	0.0562 PCSTONE-Wtw	0.0915 PCSTONE-Wtw	2.0982 ACC-3It	0.0678 GC11-Gg0.6
	0.0311 INT-3	0.0598 INT-3	0.1292 PC-ESINC-Wtw	2.1405 INT-5	0.0794 PC-GUIZAR-2000
	0.0316 SS-HOGE-Wex	0.0670 GC04-Gg0.6	0.1303 SS-HOGE-Wnw	2.3039 PCSTONE-Wnw	0.0856 SS-HOGE-Wex
	0.0219 LS-1-IlGh	0.0476 LS-4-IsGfa3	0.0461 MS-2,31-IscGfa3	0.0687 MS-3,321-IfssGfa3	0.0475 LS-4-IfGfa3
	0.0219 MS-2-LS1-IlGh	0.0476 MS-2-LS4-IsGfa3	0.0519 LS-4-IfGsim5	0.2764 GC04-Gg0.6	0.0475 MS-2-LS4-IfGfa3
	0.0225 TLS-1-IlGh	0.0480 TLS-4-IlGsim3	0.0562 CLS-4-IfGfa5	0.3321 PC-GUIZAR-1000	0.0546 CLS-4-IdGfa5
	0.0241 CLS-1-IlGch1	0.0574 CLS-4-IcGfa5	0.0791 ULS-Gh	0.3879 GC11-Gg1	0.0622 TLS-4-IfGh
σ	0.0329 PCSTONE-Wex	0.0621 ULS-Gsim3	0.0835 GC04-Gg1	0.6946 SS-HOGE-Wnw	0.0662 ULS-Gsim3
0.055	0.0483 INT-3	0.0996 PC-GUIZAR-100	0.0954 TLS-4-IfGh	0.9116 PCFOO	0.0834 GC04-Gg1
	0.0523 PC-SINC-Wex	0.1011 PCSTONE-Wnw	0.0955 GC11-Gg1	1.0141 CLS-4-IdGfa7	0.0984 GC11-Gg1
	0.0556 GC04-Gg1	0.1110 GC04-Gg1	0.1341 PCSTONE-Wtw	2.2279 ACC-3It	0.1016 PC-GUIZAR-1000
	0.0560 ULS-Gfa3	0.1157 GC11-Gg1	0.1495 PC-GUIZAR-1000	2.2333 LS-1-IlGfa7	0.1044 PCSTONE-Wnw
	0.0740 SS-HOGE-Wex	0.1238 INT-3	0.2533 SS-HOGE-Wex	2.4070 PCSTONE-Whw	0.1510 SS-HOGE-Wex
	0.0101 LS-2-IsGh	0.0167 LS-4-IdGfa3	0.0170 MS-2,31-IdcGfa3	0.0282 MS-3,321-IfssGfa3	0.0170 LS-4-IdGfa3
	0.0101 MS-2-LS2-IsGh	0.0167 MS-2-LS4-IdGfa3	0.0186 LS-4-IdGfa3	0.1467 GC04-Gg0.3	0.0170 MS-2-LS4-IdGfa3
	0.0111 TLS-1-IlGh	0.0180 TLS-4-IdGh	0.0265 CLS-4-IdGg1	0.1614 PC-SINC-Wtw	0.0198 TLS-4-IdGh
	0.0117 CLS-2-IcGch1	0.0190 CLS-4-IdGg0.3	0.0281 TLS-4-IdGh	0.2319 SS-HOGE-Wtw	0.0214 CLS-4-IdGg0.6
	0.0181 PCSTONE-Wex	0.0434 ULS-Gfa5	0.0632 GC11-Gg0.6	0.2373 GC11-Gch1	0.0444 ULS-Gfa3
Avg.	0.0198 PC-GUIZAR-100	0.0524 PCSTONE-Wtw	0.0658 ULS-Gsim3	0.6593 PCFOO	0.0520 GC11-Gg1
	0.0214 INT-3	0.0544 GC11-Gg1	0.0685 GC04-Gg1	0.6998 CLS-4-IdGfa7	0.0538 GC04-Gg1
	0.0218 GC04-Gg1	0.0550 PC-REN2010-Wex	0.0773 PCSTONE-Whw	1.9667 ACC-3It	0.0581 PCSTONE-Wtw
	0.0232 ULS-Gfa3	0.0608 INT-3	0.1095 PC-SINC-Wtw	2.0953 INT-5	0.0795 PC-GUIZAR-2000
	0.0283 SS-HOGE-Wex	0.0676 SS-HOGE-Wex	0.1280 SS-HOGE-Wnw	2.2549 PCSTONE-Whw	0.0802 SS-HOGE-Wex

Table 2.10 – Average error per shift and magnitude of top 10 evaluated methods using the best variants for each approach. **Rows**: five noise levels. **Columns**: four shift magnitudes. Note that the last column displays the top ten methods of the average for each noise level excluding the last shift magnitude level.

sorted by accuracy. From this, we can observe the versatileness of multiscale gradient-based approaches. Also, the bidirectional bias correction method (ULS) proved to work well in most cases, and it usually improves over other more expensive methods as the noise increases. Another thing to point out is the instability of the iterative total least squares method, particularly under low SNR conditions. Although this method usually achieves more accurate results than the original least squares approach, its usage must be verified [175, 185] under uncontrolled scenarios. Also, the poor tolerance to noise of phase-correlation-based methods was made evident from these results. Nevertheless, the Guizar-Sicairos approach seems to be less affected by upsampling the phase-correlation surface around its peak. Finally, as expected, every non-multiscale GBSE method failed when shift magnitudes are larger than 1.1.

To get a better understanding of the robustness of the methods against noise or shift magnitude, Fig. 2.16 shows how the error evolves by varying both noise or shift magnitude categories. To be fair, shifts with magnitude larger than 1.1 were excluded from both figures. Once again, the improved accuracy of the multiscale GBSE approach is evidenced in both tests, and improves over using a non-multiscale approach when the shifts get larger [131]. Also, the least squares approach with one iteration is an excellent candidate when the shift magnitude is short or the SNR is high. When this is not the case and a fast method is required, the bidirectional bias correction approach proved to be the best candidate. Once again, gradient correlation methods were more accurate than phase correlation-based approaches, particularly as the noise increases.



(a) Top ten methods by noise level (b) Top ten methods by shift magnitude category

Figure 2.16 – Average error for some variants of preselected methods. **Left**: varying noise averaging over all shift magnitudes. **Right**: varying shift magnitude averaging over all noise levels.

A short study on multiscale gradient-based shift estimation

We studied whether it is more convenient to iterate the standard GBSE approach in a direct fashion, i.e. using only one scale, or in a multiscale approach. To this end, both methodologies described in section 2.2.3 were evaluated under different noise conditions, shifts and gradient estimators. To show the most representative results, four SNR conditions were considered: noiseless, low noise ($\sigma = 75$), medium noise ($\sigma = 150$) and high noise ($\sigma = 300$). Noise values are according to 12 bit images. Each table is organized in groups of four lines corresponding to each of these four noise configurations. Also, the four most significative shifts in terms of results are shown: a big shift $(0.5, -0.9)$, a medium shift $(0.2, -0.2)$, a small shift $(0.024, 0.052)$ and no shift.

The performance of each algorithm under each condition was evaluated by simulation using the same setup employed throughout this section. The results shown were later validated using the Cramer-Rao bound (see chapter 4), verifying that both $var(\hat{v_x})$

	$\Vert\mathbf{v}\Vert \leq 0.1$	$0.1 < \Vert\mathbf{v}\Vert \leq 0.5$	$0.5 < \Vert\mathbf{v}\Vert \leq 1.1$	$\Vert\mathbf{v}\Vert > 1.1$
σ 0.000	0.0000 MS-3,321-IdssGfa3	0.0000 MS-3,321-IdssGfa3	0.0001 MS-3,321-IdssGfa3	0.0196 PC-SINC-Whw
	0.0003 MS-2,31-IscGfa3	0.0017 MS-2,31-IscGfa3	0.0016 MS-2,31-IscGfa3	0.0197 MS-3,321-IdssGfa3
	0.0011 GC11-Gg0.6	0.0044 SS-HOGE-Wtw	0.0085 SS-HOGE-Wtw	0.0317 SS-HOGE-Wtw
	0.0012 PCSTONE-Whw	0.0087 PC-SINC-Whw	0.0097 TLS-4-IlGfa3	0.0911 GC04-Gg0.6
	0.0018 PC-SINC-Whw	0.0126 PCSTONE-Whw	0.0097 LS-4-IlGfa3	0.1756 GC11-Gg0.6
	0.0020 SS-HOGE-Wtw	0.0140 GC11-Gg0.6	0.0145 PC-SINC-Whw	0.2735 PC-GUIZAR-2000
	0.0035 LS-4-IlGfa3	0.0143 TLS-4-IlGfa3	0.0311 PCSTONE-Whw	0.7801 MS-2,31-IscGfa3
	0.0035 TLS-4-IlGfa3	0.0143 LS-4-IlGfa3	0.0364 GC11-Gg0.6	2.1838 PCSTONE-Whw
	0.0036 TLS-1-IlGh	0.0363 ULS-Gfa3	0.0482 GC04-Gg0.6	15.7022 ULS-Gfa3
	0.0036 LS-1-IlGh	0.0388 TLS-1-IlGh	0.0578 ULS-Gfa3	Inf LS-1-IlGh
	0.0059 GC04-Gg0.6	0.0403 LS-1-IlGh	0.1174 TLS-1-IlGh	Inf TLS-1-IlGh
	0.0061 PC-GUIZAR-2000	0.0490 GC04-Gg0.6	0.1364 PC-GUIZAR-2000	Inf LS-4-IlGfa3
	0.0073 ULS-Gfa3	0.0627 PC-GUIZAR-2000	0.1708 LS-1-IlGh	Inf TLS-4-IlGfa3
σ 0.005	0.0037 MS-3,321-IdssGfa3	0.0040 MS-3,321-IdssGfa3	0.0039 MS-3,321-IdssGfa3	0.0045 MS-3,321-IdssGfa3
	0.0038 MS-2,31-IscGfa3	0.0042 MS-2,31-IscGfa3	0.0042 MS-2,31-IscGfa3	0.0477 PC-SINC-Whw
	0.0050 TLS-1-IlGh	0.0144 TLS-4-IlGfa3	0.0102 LS-4-IlGfa3	0.0552 SS-HOGE-Wtw
	0.0052 LS-1-IlGh	0.0144 LS-4-IlGfa3	0.0104 TLS-4-IlGfa3	0.0981 GC04-Gg0.6
	0.0062 TLS-4-IlGfa3	0.0180 GC11-Gg0.6	0.0225 SS-HOGE-Wtw	0.1881 GC11-Gg0.6
	0.0062 LS-4-IlGfa3	0.0189 PCSTONE-Whw	0.0367 PCSTONE-Whw	0.2415 PC-GUIZAR-2000
	0.0065 GC11-Gg0.6	0.0199 SS-HOGE-Wtw	0.0432 PC-SINC-Whw	2.1241 PCSTONE-Whw
	0.0068 PC-GUIZAR-2000	0.0345 PC-SINC-Whw	0.0434 GC11-Gg0.6	2.6144 LS-1-IlGh
	0.0077 GC04-Gg0.6	0.0350 ULS-Gfa3	0.0564 GC04-Gg0.6	13.5597 ULS-Gfa3
	0.0090 ULS-Gfa3	0.0411 TLS-1-IlGh	0.0575 ULS-Gfa3	Inf TLS-1-IlGh
	0.0124 PCSTONE-Whw	0.0449 LS-1-IlGh	0.1203 TLS-1-IlGh	Inf LS-4-IlGfa3
	0.0197 PC-SINC-Whw	0.0495 GC04-Gg0.6	0.1439 PC-GUIZAR-2000	Inf TLS-4-IlGfa3
	0.0207 SS-HOGE-Wtw	0.0671 PC-GUIZAR-2000	0.1799 LS-1-IlGh	Inf MS-2,31-IscGfa3
σ 0.015	0.0099 TLS-1-IlGh	0.0121 MS-3,321-IdssGfa3	0.0130 MS-2,31-IscGfa3	0.0221 MS-3,321-IdssGfa3
	0.0104 LS-1-IlGh	0.0123 MS-2,31-IscGfa3	0.0131 MS-3,321-IdssGfa3	0.1171 GC04-Gg0.6
	0.0113 PC-GUIZAR-2000	0.0185 LS-4-IlGfa3	0.0167 LS-4-IlGfa3	0.1300 SS-HOGE-Wtw
	0.0121 MS-2,31-IscGfa3	0.0185 TLS-4-IlGfa3	0.0215 TLS-4-IlGfa3	0.1764 PC-SINC-Whw
	0.0121 MS-3,321-IdssGfa3	0.0272 GC11-Gg0.6	0.0546 GC11-Gg0.6	0.2345 GC11-Gg0.6
	0.0136 TLS-4-IlGfa3	0.0431 ULS-Gfa3	0.0599 PCSTONE-Whw	0.2796 PC-GUIZAR-2000
	0.0136 LS-4-IlGfa3	0.0432 PCSTONE-Whw	0.0616 ULS-Gfa3	0.6751 MS-2,31-IscGfa3
	0.0137 GC04-Gg0.6	0.0519 GC04-Gg0.6	0.0634 GC04-Gg0.6	2.2229 PCSTONE-Whw
	0.0171 ULS-Gfa3	0.0534 TLS-1-IlGh	0.0792 SS-HOGE-Wtw	2.6654 LS-1-IlGh
	0.0194 GC11-Gg0.6	0.0627 PC-GUIZAR-2000	0.1115 PC-SINC-Whw	17.7786 ULS-Gfa3
	0.0405 PCSTONE-Whw	0.0689 LS-1-IlGh	0.1527 PC-GUIZAR-2000	Inf TLS-1-IlGh
	0.0645 SS-HOGE-Wtw	0.0700 PC-SINC-Whw	0.1541 TLS-1-IlGh	Inf LS-4-IlGfa3
	0.0742 PC-SINC-Whw	0.0718 SS-HOGE-Wtw	0.2446 LS-1-IlGh	Inf TLS-4-IlGfa3
σ 0.025	0.0145 TLS-1-IlGh	0.0199 MS-2,31-IscGfa3	0.0221 MS-2,31-IscGfa3	0.0261 MS-3,321-IdssGfa3
	0.0150 LS-1-IlGh	0.0203 MS-3,321-IdssGfa3	0.0229 MS-3,321-IdssGfa3	0.2217 SS-HOGE-Wtw
	0.0196 PC-GUIZAR-2000	0.0246 LS-4-IlGfa3	0.0274 LS-4-IlGfa3	0.2261 PC-SINC-Whw
	0.0218 MS-2,31-IscGfa3	0.0267 TLS-4-IlGfa3	0.0716 ULS-Gfa3	0.2354 GC11-Gg0.6
	0.0227 MS-3,321-IdssGfa3	0.0442 ULS-Gfa3	0.0845 GC04-Gg0.6	0.2685 PC-GUIZAR-2000
	0.0234 LS-4-IlGfa3	0.0559 GC11-Gg0.6	0.0851 GC11-Gg0.6	0.6882 MS-2,31-IscGfa3
	0.0268 ULS-Gfa3	0.0627 PCSTONE-Whw	0.0924 PCSTONE-Whw	2.3369 PCSTONE-Whw
	0.0304 GC04-Gg0.6	0.0670 GC04-Gg0.6	0.1439 SS-HOGE-Wtw	2.7269 LS-1-IlGh
	0.0623 GC11-Gg0.6	0.0694 PC-GUIZAR-2000	0.1492 PC-GUIZAR-2000	42.2713 ULS-Gfa3
	0.0674 PCSTONE-Whw	0.0773 TLS-1-IlGh	0.2029 TLS-1-IlGh	Inf TLS-1-IlGh
	0.1206 SS-HOGE-Wtw	0.1037 LS-1-IlGh	0.2205 PC-SINC-Whw	Inf LS-4-IlGfa3
	0.1666 PC-SINC-Whw	0.1043 SS-HOGE-Wtw	0.3185 LS-1-IlGh	Inf TLS-4-IlGfa3
	Inf TLS-4-IlGfa3	0.1683 PC-SINC-Whw	Inf TLS-4-IlGfa3	Inf GC04-Gg0.6
σ 0.055	0.0219 LS-1-IlGh	0.0478 LS-4-IlGfa3	0.0461 MS-2,31-IscGfa3	0.0688 MS-3,321-IdssGfa3
	0.0225 TLS-1-IlGh	0.0498 MS-2,31-IscGfa3	0.0472 MS-3,321-IdssGfa3	0.2764 GC04-Gg0.6
	0.0463 LS-4-IlGfa3	0.0534 MS-3,321-IdssGfa3	0.0589 LS-4-IlGfa3	0.3321 PC-GUIZAR-2000
	0.0475 MS-2,31-IscGfa3	0.0621 ULS-Gfa3	0.0804 ULS-Gfa3	0.4214 GC11-Gg0.6
	0.0512 MS-3,321-IdssGfa3	0.0998 PC-GUIZAR-2000	0.0964 GC11-Gg0.6	0.7207 SS-HOGE-Wtw
	0.0555 PC-GUIZAR-2000	0.1451 TLS-1-IlGh	0.1039 GC04-Gg0.6	0.7655 PC-SINC-Whw
	0.0560 ULS-Gfa3	0.1617 GC04-Gg0.6	0.1495 PC-GUIZAR-2000	0.7991 MS-2,31-IscGfa3
	0.1191 GC04-Gg0.6	0.1631 PCSTONE-Whw	0.1662 PCSTONE-Whw	2.4070 PCSTONE-Whw
	0.1468 PCSTONE-Whw	0.1669 GC11-Gg0.6	0.3531 TLS-1-IlGh	2.8674 LS-1-IlGh
	0.4459 SS-HOGE-Wtw	0.1783 LS-1-IlGh	0.3983 SS-HOGE-Wtw	8.1707 ULS-Gfa3
	0.5200 PC-SINC-Whw	0.4742 SS-HOGE-Wtw	0.4705 LS-1-IlGh	Inf TLS-1-IlGh
	Inf TLS-4-IlGfa3	0.5693 PC-SINC-Whw	0.5198 PC-SINC-Whw	Inf LS-4-IlGfa3
	Inf GC11-Gg0.6	Inf TLS-4-IlGfa3	Inf TLS-4-IlGfa3	Inf TLS-4-IlGfa3

Table 2.11 – Average error of preselected methods per shift and magnitude. **Rows**: five noise levels. **Columns**: four shift magnitudes.

and $var(\hat{v_y})$ are lower than 0.01, so that the averaged values contain only valid shift estimations.

Shift (px)	**Noise**	**IT2Gh**	**IT2Gg1**	**IT2Gg0.3**	**MS2Gh**	**MS2Gg1**	**MS2Gg0.3**
(0.5000,-0.9000)	$\sigma = 0$	0.0514	0.0818	0.0472	0.0387	0.1600	**0.0316**
	$\sigma = 75$	0.1375	0.1053	0.1103	0.0744	0.1808	**0.0582**
	$\sigma = 150$	0.2875	0.1414	0.2305	0.1267	0.2130	**0.1009**
	$\sigma = 300$	0.4927	0.2327	0.4292	0.2319	0.2872	**0.1909**
(0.0000,0.0000)	$\sigma = 0$	**0.0000**	0.0000	0.0000	0.0000	0.0000	0.0000
	$\sigma = 75$	**0.0114**	0.0188	0.0132	0.0164	0.0188	0.0175
	$\sigma = 150$	**0.0168**	0.0330	0.0219	0.0300	0.0326	0.0313
	$\sigma = 300$	**0.0191**	0.0539	0.0307	0.0527	0.0549	0.0573
(0.2000,-0.2000)	$\sigma = 0$	0.0115	0.0117	0.0154	**0.0115**	0.0360	0.0159
	$\sigma = 75$	0.0249	0.0267	0.0223	**0.0192**	0.0470	0.0225
	$\sigma = 150$	0.0652	0.0424	0.0487	0.0360	0.0591	**0.0358**
	$\sigma = 300$	0.1295	0.0765	0.1103	0.0738	0.0899	**0.0694**
(0.0240,0.0520)	$\sigma = 0$	0.0040	**0.0019**	0.0052	0.0039	0.0073	0.0054
	$\sigma = 75$	**0.0122**	0.0181	0.0138	0.0156	0.0189	0.0166
	$\sigma = 150$	**0.0198**	0.0326	0.0231	0.0296	0.0326	0.0311
	$\sigma = 300$	**0.0300**	0.0543	0.0354	0.0547	0.0564	0.0581
Avg.	$\sigma = 0$	0.0167	0.0238	0.0170	0.0135	0.0508	**0.0132**
	$\sigma = 75$	0.0465	0.0422	0.0399	0.0314	0.0664	**0.0287**
	$\sigma = 150$	0.0973	0.0624	0.0811	0.0556	0.0843	**0.0498**
	$\sigma = 300$	0.1678	0.1043	0.1514	0.1033	0.1221	**0.0939**

Table 2.12 – Estimation error (in pixels) per shift of every method using **two iterations** and **bicubic** interpolation from valid estimations. For each shift and estimation method, four SNR conditions were tested. The first three columns are for the iterative method (IT) while the last three are for the multiscale approach (MS) with a single iteration per scale. In each case, three gradient estimation methods were used: backward difference and Gaussian derivative with $\sigma = 1$ and with $\sigma = 0.3$ respectively.

In tables 2.12 and 2.13 results are shown for two iterations and bicubic interpolation, and for three iterations with spline interpolation respectively. From these results several conclusions can be drawn.

- First, as expected, the multiscale method is much more robust when the shift magnitude is high. In fact, even at a shift as low as (0.2,-0.2) it is recommendable to use the multiscale method instead of the standard iterative version.
- Second, when no shift or a small shift is present, the single-scale methods achieve much better accuracies. Apparently, the multiscale algorithms are not suited for such small shifts since their poor performance on lower scales results in less accurate results. This result contradicts several state-of-the-art methods and is worth remarking.
- Third, regarding the amount of iterations/scales to use, in presence of high noise, performing more iterations in the original scale or using more scales in the multiscale approach gives worse results in terms of accuracy. When dealing with a noisy situation, the resampling operation proved to be negative for the shift estimation algorithm. This result is more accentuated for the multiscale approach.
- Finally, the multiscale algorithm proved to be a better contender when dealing with noise in general, although this factor is greatly influenced by the shift magnitude.

Shift (px)	Noise	IT3Gh	IT3Gg1	IT3Gg0.3	MS3Gh	MS3Gg1	MS3Gg0.3
(0.5000,-0.9000)	$\sigma = 0$	0.0156	0.0238	**0.0065**	0.0114	0.1007	0.0086
	$\sigma = 75$	0.0646	0.0437	0.0437	0.0293	0.1194	**0.0251**
	$\sigma = 150$	0.1869	0.0727	0.1326	0.0533	0.1488	**0.0497**
	$\sigma = 300$	0.4092	0.1480	0.3337	0.1093	0.2143	**0.1001**
(0.0000,0.0000)	$\sigma = 0$	**0.0000**	0.0000	0.0000	0.0000	0.0000	0.0000
	$\sigma = 75$	**0.0116**	0.0199	0.0133	0.0261	0.0330	0.0260
	$\sigma = 150$	**0.0194**	0.0357	0.0246	0.0468	0.0547	0.0467
	$\sigma = 300$	**0.0250**	0.0621	0.0391	0.0980	0.1065	0.1001
(0.2000,-0.2000)	$\sigma = 0$	**0.0027**	0.0045	0.0043	0.0206	0.0205	0.0235
	$\sigma = 75$	0.0201	0.0229	**0.0182**	0.0304	0.0430	0.0334
	$\sigma = 150$	0.0475	0.0395	**0.0373**	0.0483	0.0593	0.0489
	$\sigma = 300$	0.1096	**0.0718**	0.0913	0.0982	0.1195	0.0977
(0.0240,0.0520)	$\sigma = 0$	**0.0007**	0.0009	0.0013	0.0068	0.0041	0.0080
	$\sigma = 75$	**0.0120**	0.0188	0.0132	0.0219	0.0283	0.0225
	$\sigma = 150$	**0.0207**	0.0348	0.0248	0.0471	0.0548	0.0464
	$\sigma = 300$	**0.0313**	0.0622	0.0412	0.1011	0.1056	0.1001
Avg.	$\sigma = 0$	0.0047	0.0073	**0.0030**	0.0097	0.0313	0.0100
	$\sigma = 75$	0.0271	0.0263	**0.0221**	0.0269	0.0559	0.0268
	$\sigma = 150$	0.0686	**0.0457**	0.0548	0.0489	0.0794	0.0479
	$\sigma = 300$	0.1438	**0.0860**	0.1263	0.1017	0.1365	0.0995

Table 2.13 – Estimation error (in pixels) per shift of every method using **three iterations** and **spline** interpolation from valid estimations. Table configuration is the same as in Table 2.12.

However, except when the shift magnitude is lower than a fifth of a pixel, its use is recommended. Moreover, its computational cost is lower than the iterative counterpart since the resampling is performed on lower resolution images.

Conclusions

To summarize, the conclusions for this section are:

- The ***best overall method*** proved to be the ***multiscale GBSE*** approach using three scales. The best iteration/resampling configuration on the scales was to do a single iteration on the coarsest scale, followed by performing two iterations in the middle scale using spline interpolation, and applying three iterations on the finest (original) scale using FFT with symmetrization as the interpolation method.
- The best method type in general is gradient-based, and ***the amount of iterations and/or scales depends on the shift magnitude to estimate***.
- The ***derivative*** for GBSE methods that proved to be the ***most robust to noise*** was the one proposed by ***Farid [53] using a*** 3×3 ***support***, although the 3×3 variant of the Simoncelli derivative [154] achieved similar results.
- For ***low shift magnitudes*** (category 1 in Table 2.7) the best methods are again gradient-based and the recommended gradient estimation method to use was the ***hypomode***.
- For ***shift magnitudes larger than 0.5***, a ***multiscale approach*** is mandatory.
- ***Phase correlation methods suffer from low SNR scenarios***.
- In general, ***gradient correlation-based methods improve over phase correlation methods***, and this difference becomes more important as the noise increases.
- For phase-correlation-based methods, the ***best windows*** used for apodization proved to be the ***Hamming*** window, the ***Blackman*** window and the ***Tukey*** window. Also, in several cases, ***avoiding apodization*** yielded the best results.
- The TLS GBSE method proved in general to be more accurate than both the standard LS and the CLS variant. For the TLS approach, a singular value decomposition is required, which may be expensive in some cases. In contrast, the CLS

method requires the knowledge of the noise standard deviation, which may not be available in some situations. ***If the TLS*** method is used, ***validating*** using existing approaches [175, 185] ***must be done*** since it proved to be unstable on many situations, particularly under lower SNRs.

- On shifts with ***magnitudes between 0.1 and 1.1***, the ***bidirectional bias corrected method*** of Pham *et al.* [122] achieves good results without requiring interpolation nor an iterative procedure, thus singling it out as the ***best overall low-cost method***. As an exception, if the shift magnitude is *a-priori* known to be lower than 0.1, a single iteration of the total least squares method gets the best results among the fast methods.
- For ***large shift magnitudes*** (last category in Table 2.7), the ***multiscale GBSE*** method always achieved the best results. However, it should be noted that the amount of scales used depends on the maximum possible shift magnitude to estimate. Since in the evaluated setup, the maximum displacement was known *a-priori*, the amount of scales was set according to it. If however this information is not known, phase/gradient correlation-based methods should be used. In particular, for high SNR scenarios, either fitting a sinc on the PCS [57] or using Hoge's subspace method [75] yielded the best results. Under ***lower SNR***, ***gradient correlation*** approaches should be employed.

2.4.3 Robustness to violation of the brightness constancy constraint

While GBSE methods rely on the brightness constancy constraint, the main strength of phase-correlation methods is their higher tolerance against image changes. Therefore, we performed an experiment, using the same setup described in this section, where a square of size $K \times K$ was removed (i.e., set to white) from the center of the first image before estimating the shift. We tested with $K = [3, 5, 7, 9, 11]$, which, given that the shift is estimated using 50×50 images, this implies removing up to 5% of the image. The results of this are displayed in Fig. 2.17 for the five values of K. All methods were evaluated using two noise levels: low noise ($\sigma = 0.005$) or high noise ($\sigma = 0.055$).

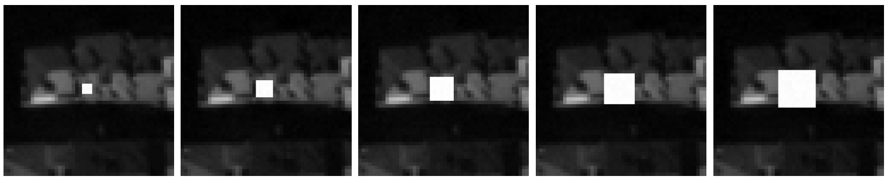

(a) $K = 3$ (b) $K = 5$ (c) $K = 7$ (d) $K = 9$ (e) $K = 11$

Figure 2.17 – Example evaluated subimages for the five different values of K.

In Table 2.14 we observe the top 5 methods averaged over shift magnitudes one to three. Although a decrease in performance is observed for GBSE methods, they still improve over phase correlation methods under shift magnitudes lower than one pixel. However, when evaluating shift magnitudes higher than one pixel (category four from Table 2.7), the situation is different, as seen from Table 2.15. In fact, GBSE methods still achieve the best accuracies when the size of the removed square is $K = 3$ or, in case of high noise, when $K \leq 7$. On every other case, GBSE methods are not recommended. For example, with $K = 11$, the best results were obtained using phase correlation methods as expected, and most importantly, under high SNR scenarios, the difference between them and the best performing GBSE variant was up to four times better in terms of the measured error in pixels. What is worse, since the deleted square was removed from the center of each subimage, then the windowing procedure applied before computing the phase correlation removes almost all the image information (this explains why the best performing PC methods did not perform apodization). Since the difference between both images does not necessarily have to be in the center, this implies phase correlation-based methods should obtain even better results accentuating the improvement with respect to GBSE methods.

	$K = 3$	$K = 5$	$K = 7$	$K = 9$	$K = 11$
σ 0.005	0.0231 MS-3,321-IfssGh	0.0376 LS-4-IfGch2	0.0632 LS-4-IfGch2	0.0762 LS-4-IfGch2	0.0940 PC-SINC-Wnw
	0.0234 LS-4-IdGch2	0.0376 MS-2-LS4-IfGch2	0.0632 MS-2-LS4-IfGch2	0.0762 MS-2-LS4-IfGch2	0.0944 LS-4-IfGch2
	0.0241 TLS-4-IdGg0.3	0.0428 TLS-4-IdGch1	0.0739 CLS-4-IdGch1	0.0883 CLS-4-IdGch1	0.0944 MS-2-LS4-IfGch2
	0.0279 CLS-4-IfGg0.6	0.0505 CLS-4-IfGg0.6	0.0766 TLS-4-IdGch1	0.0967 PC-SINC-Wnw	0.1474 CLS-4-IfGfa3
	0.0565 ADF2-2QI	0.0730 ULS-Gch1	0.0991 PC-SINC-Wnw	0.1196 ULS-Gh	0.1506 PCSTONE-Wnw
σ 0.055	0.0645 MS-2,31-IscGfa3	0.0815 CLS-4-IdGg0.6	0.1016 LS-4-IfGfa3	0.1221 LS-4-IfGfa3	0.1486 LS-4-IfGch2
	0.0664 CLS-4-IfGg0.6	0.0831 LS-4-IdGfa3	0.1016 MS-2-LS4-IfGfa3	0.1221 MS-2-LS4-IfGfa3	0.1486 MS-2-LS4-IfGch2
	0.0665 LS-4-IdGfa3	0.0831 MS-2-LS4-IdGfa3	0.1030 CLS-4-IfGg0.6	0.1380 CLS-4-IfGfa3	0.1546 CLS-4-IdGch1
	0.0796 TLS-4-IfGh	0.1117 TLS-4-IdGh	0.1221 ULS-Gh	0.1559 PCSTONE-Wnw	0.1651 PCSTONE-Wnw
	0.0933 ULS-Gh	0.1207 ULS-Gch2	0.1460 TLS-4-IdGh	0.1622 ULS-Gg0.3	0.1814 TLS-4-IcGch1

Table 2.14 – Average error of top 5 evaluated methods per noise and K value for shift magnitudes one to three. **Rows**: two noise levels. **Columns**: five K values. Only the best variant per method is displayed.

	$K = 3$	$K = 5$	$K = 7$	$K = 9$	$K = 11$
σ 0.005	0.0324 MS-4,4321-IfssssGh	0.0525 ADF2-2QI	0.1121 PCSINC-Wnw	0.1511 PCSINC-Wnw	0.1888 PCSINC-Wnw
	0.0484 ADF2-2QI	0.0781 MS-5,54321-IfssssGch1	0.1190 PCESINC-Wnw	0.1593 PCESINC-Wnw	0.1935 PCESINC-Wnw
	0.0874 ADF-2QI	0.0868 ADF-2QI	0.1284 PCREN2010-Wnw	0.1669 PCREN2010-Wnw	0.2018 PCREN2010-Wnw
	0.1286 GC04-Gch3	0.1274 PCSINC-Wnw	0.2022 MS-4,4321-IfssssGh	0.3735 PCQUADFIT	0.5107 PCQUADFIT
	0.1935 GC11-Gg0.3	0.1359 PCESINC-Wnw	0.3194 PCQUADFIT	0.4677 MS-3,321-IsssGfa3	0.6405 PCGAUSSFIT
	0.2012 PCSINC-Wnw	0.1461 PCREN2010-Wnw	0.4191 SS-HOGE-Wnw	0.5036 PCGAUSSFIT	0.7407 SS-HOGE-Wnw
	0.2098 PCESINC-Wnw	0.3391 SS-HOGE-Wnw	0.4525 PCGAUSSFIT	0.5449 SS-HOGE-Wnw	0.7662 MS-3,321-IsssGfa3
σ 0.055	0.1434 MS-4,4321-IssssGh	0.1512 MS-5,54321-IsssssGch1	0.4121 MS-3,321-IsssGfa3	0.6444 PCESINC-Wnw	0.6636 PCESINC-Wex
	0.4550 PCREN2010-Wnw	0.4168 PCESINC-Wnw	0.5034 PCREN2010-Wnw	0.6476 PCREN2010-Wnw	0.6654 PCREN2010-Wex
	0.4589 PCESINC-Wnw	0.4176 PCREN2010-Wnw	0.5051 PCESINC-Wnw	0.6496 PCSINC-Wnw	0.6717 PCSINC-Wex
	0.4649 PCSINC-Wnw	0.4228 PCSINC-Wnw	0.5117 PCSINC-Wnw	0.6640 MS-3,321-IsssGfa3	0.7000 PCQUADFIT
	0.6876 GC04-Gch3	0.6585 PCQUADFIT	0.7304 PCQUADFIT	0.8098 PCQUADFIT	0.7557 PCGAUSSFIT
	0.7474 PCQUADFIT	0.6693 GC04-G5	0.7952 PCGAUSSFIT	0.8154 GC11-Gch2	0.7859 MS-3,321-IsssGfa3
	0.8013 PCGAUSSFIT	0.7211 PCGAUSSFIT	0.9658 PCFOO	0.8858 PCGAUSSFIT	0.9326 PCFOO

Table 2.15 – Average error of top 7 evaluated methods per noise and K value for the fourth shift magnitude. **Rows**: two noise levels. **Columns**: five K values. Only the best variant per method is displayed.

Conclusions from these experiments

In this section, we studied the behaviour of shift estimation methods when both images differ. This difference could be caused for example, by occlusions, different information appearing on each spectra on multi-spectral images or because both images were taken at different moments in time. In these cases, when the underlying displacement is lower than one pixel (categories one to three from Table 2.7), GBSE methods yield the best results. However, with larger displacement magnitudes, phase correlation approaches should be considered. In particular, with high SNR, phase correlation methods that use local function fitting in the spatial domain, described in section 2.3.1, yield the best results and dramatically improve over GBSE methods. Under lower SNR scenarios, the improvement of PC over GBSE methods becomes less significative, and both approaches should be considered.

2.4.4 Robustness to aliased sample processes

We evaluated the robustness of each approach to aliasing. To simulate aliasing, we first applied a Gaussian low-pass filter of parameter σ_A to a high-resolution image, yielding

$\bar{I} = g_{\sigma_a} * I$ where g denotes the Gaussian kernel and $*$ the convolution operator. Then we took both images

$$\tilde{I}_1 = subsample(\bar{I}) \quad \tilde{I}_2 = subsample(shift(\bar{I}, d)) \tag{2.125}$$

where $shift$ is an integer resampling method (only moves the pixels), d is 16 times the desired subpixel displacement and $subsample$ subsamples the image by regularly sampling 1 out of 16 pixels. We evaluated $d = [-7, -4, 0, 8] \times [-7, -4, 0, 8]$, where in this case $\times$ corresponds to the cartesian product. In total, this implied estimating 16 subpixel shifts where for each dimension the shifts were $[-0.4375, -0.25, 0, 0.5]$. To test different levels of aliasing, we used five values for σ_A displayed in Table 2.16.

Level	1	2	3	4	5
σ_A	8	6.4	4	2.8	0.8

Table 2.16 – Values of the standard deviation of the Gaussian used to prefilter the image before subsampling to generate different aliasing levels

All experiments were performed using the same two subimages extracted from the same location of both $\tilde{I}_1$ and $\tilde{I}_2$ respectively, by adding a mild noise of $\sigma_N = 75$ which is equivalent to noise level 3. Both images are displayed in Fig. 2.18 for the five simulated levels of aliasing for the shift $(-0.25, 0.5)$ which proved to be troublesome for several methods. The difference between compensated images $\tilde{I}_1(x, y) - \tilde{I}_2(x + 0.25, y - 0.5)$, shown on the bottom row of the figure, illustrates the effects of aliasing on the shift estimation problem. As the aliasing level increases, more differences appear on the edges of the objects, and these differences gets larger in value. These differences may also disorient shift estimation methods generating non-existing secondary motions which alter the final results.

Nevertheless, results in Fig. 2.19 prove that gradient-based methods are still able to deal with aliasing even though they break the brightness constancy constraint on which they are based. To generate this figure, one candidate variant from each evaluated shift estimation algorithm was selected, particularly the variant obtaining the highest average error among all aliasing levels and evaluated shifts. The figure then displays the top ten of these variants. As seen, GBSE approaches using spline interpolation and Gaussian derivatives with $\sigma = 0.6$ gave the best results on average using few computational resources. Among the phase correlation methods, the improved gradient correlation approach of [179] seems the least affected by aliasing, particularly when using short supported filters for gradient calculation (either Gausian with $\sigma = 0.3$ or the first order Christmas kernel). The Stone method [159] also achieved accurate results without performing apodization of the input images, and should be considered, thanks to its fast implementation (in our computer, among these methods, it turned out to be the fastest). Finally, the Pham bidirectional approach [122] offers a good balance between speed and robustness against aliasing.

Since aliasing is in general unpredictable and may impact more or less the accuracy depending on the sampling, averaging over all shifts may hide interesting cases. Indeed, for most evaluated displacements, the average accuracy of the top 10 approaches never exceed 0.04 pixels of error, however, we observed two situations for which the error was considerably higher. Results for two diferent shifts $(-0.437, 0.5)$ and $(-0.25, 0.5)$ displayed in Fig. 2.20 show a radical change on the methods obtaining the best results. Particularly in those cases, the variants presented by Tzimiropoulos *et al.* [179] based on gradient correlation obtained the best results, which allows us to conclude that their method is the most robust against aliasing. As for differential methods, the ULS approach again turned out to outperform its class. From these figures two important conclusions should

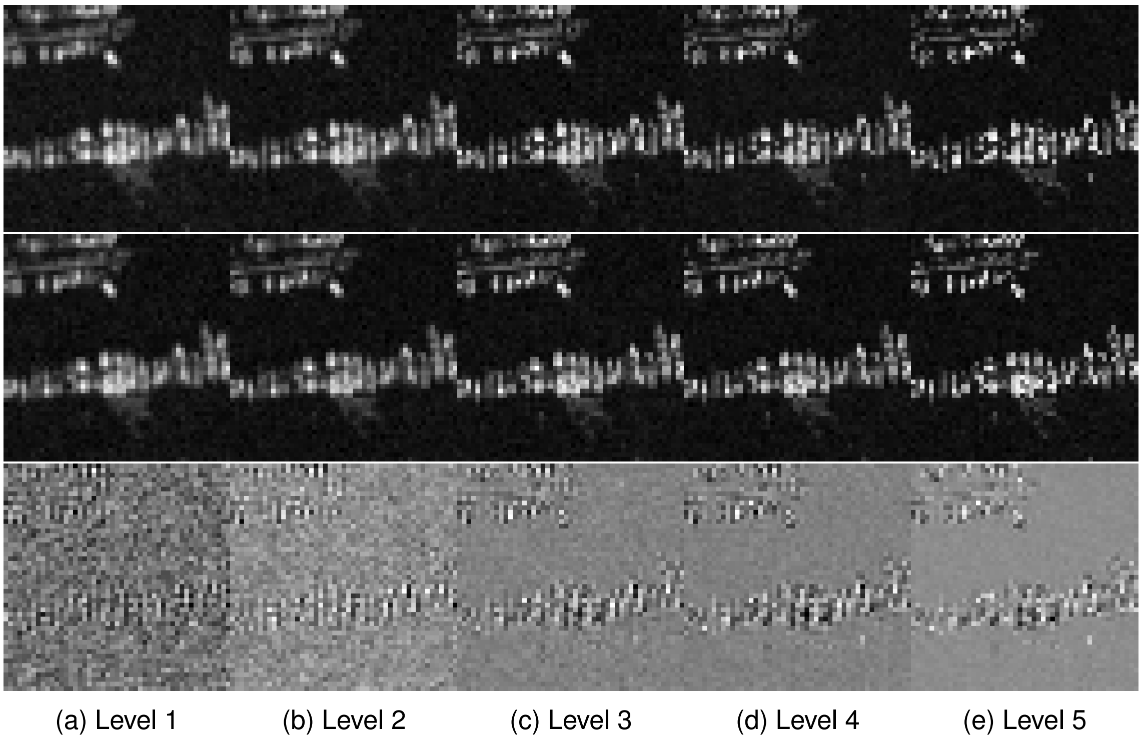


Figure 2.18 – Input images shifted by $(-0.25, 0.5)$ used for testing aliasing influence for each aliasing level. **Top**: $\tilde{I}_1$. **Center**: $\tilde{I}_2$. **Bottom**: Difference between compensated images (aliasing + noise effects). Dynamic ranges extended.

be made. First, that indeed the claims of aliasing robustness of the Christmas derivatives (of first and second order) are true and should be considered when high aliasing exists. And secondly, as expected, that iterative approaches should avoid resampling in the frequency domain, due to a potential increase of artifacts appearing from working with images sampled below the Nyquist rate.

On the other side, an example of the top 10 methods usually resulting for every tested displacement is shown in Fig. 2.21. GBSE approaches in general proved to be quite robust against low to moderate aliasing scenarios, while also achieving the most accurate results using the fewest computational resources.

Conclusions from these experiments

Three conclusions were drawn from these experiments when dealing with aliasing. First, that GBSE methods (in particular CLS or ULS) offer the best results in terms of accuracy over time consumed. Second, that gradient estimation should be performed using either the kernels proposed by Christmas [35] or Gaussian derivatives with $\sigma = 0.3$ or $\sigma = 0.6$. Finally, under extreme aliasing scenarios, the slightly more expensive gradient correlation method proved to offer the best results.

2.4.5 Computational cost comparison

To provide an idea of the computational resources required for each method, we averaged execution times for 1000 executions using several shift magnitudes and noises. The processing time of each method was measured using non-optimized Matlab implementations on an Intel Xeon E5-2650 CPU. Results, shown in Fig. 2.22, are orientative and should not be taken as a definitive measurement. Indeed, some displayed times are evidently not representative. As an example, the subspace method of Robinson (SS-

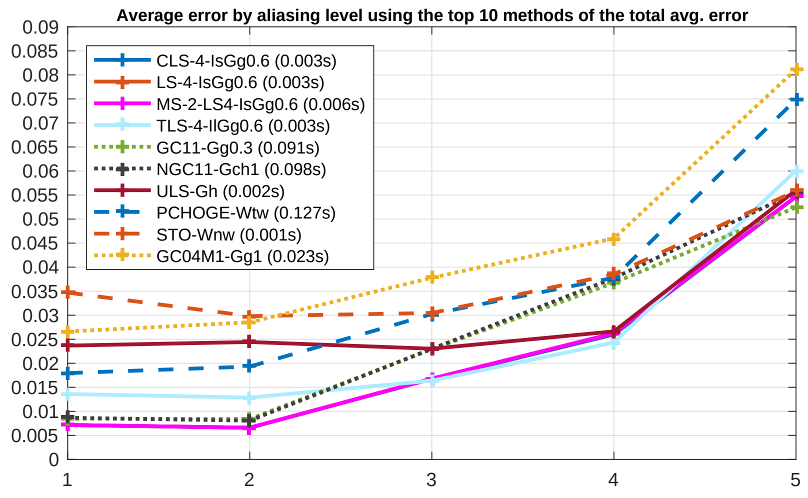


Figure 2.19 – Average error by aliasing level of top ten overall methods

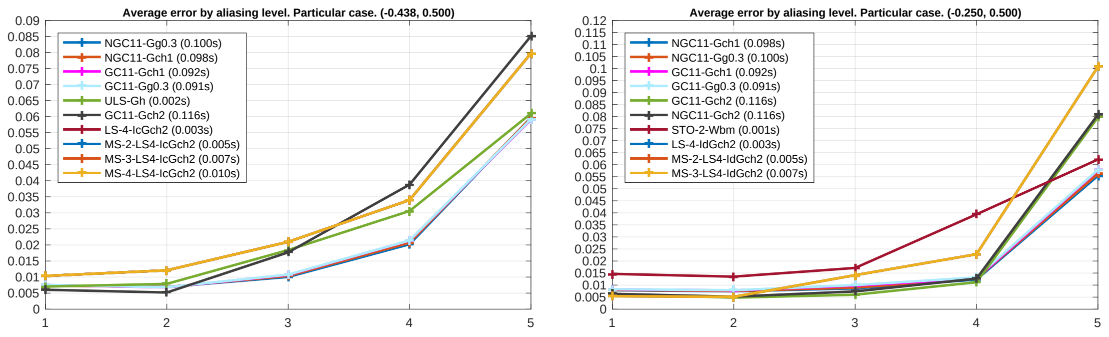


Figure 2.20 – Average error by aliasing level of top 10 for two particular shifts

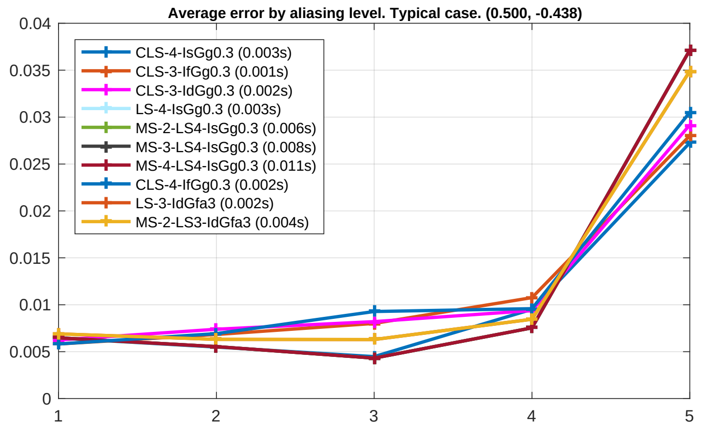


Figure 2.21 – Average error by aliasing level of top 10 methods for a typical shift

ROBINSON-Wtw) should reflect lower processing time given its low complexity. Both Sinc fitting (PC-SINC-Wtw) together with gradient correlation methods based on non-linear kernel fitting (GC11) employ considerable time due to the optimization performed. The optimization is done by means of the $lsqcurvefit$ function in Matlab which besides the optimization procedure, also does an important amount of overhead work used to initialize internal components that would not exist if a more direct implementation were available.

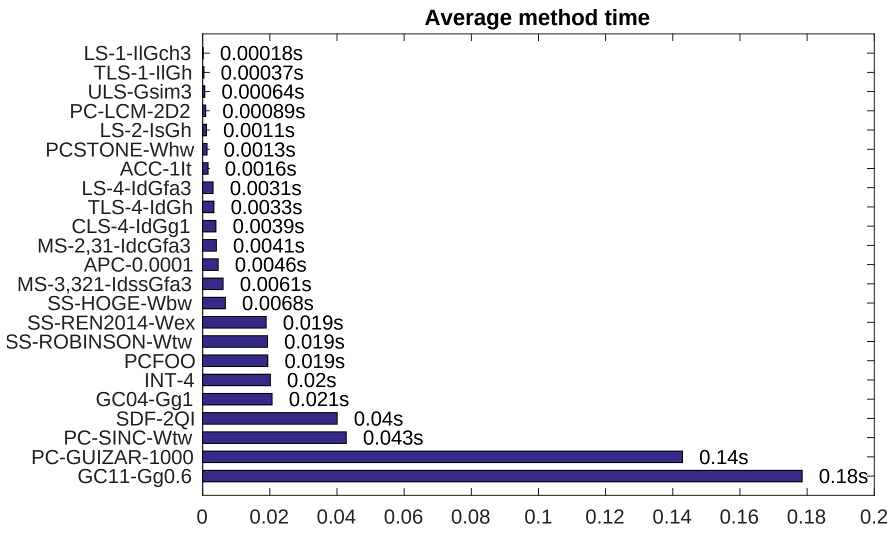


Figure 2.22 – Average execution time for some representative methods

To contrast processing time with precision, we averaged the errors for each method over all noise levels and over the first three categories of shift magnitudes. We also made this comparison for each shift magnitude individually. Results displayed in Fig. 2.23 and Fig. 2.24 represent a general summary of this review.

Based on the provided figures, we conclude that gradient-based methods not only achieve more accurate results but also require fewer computational resources. When the shift estimation must cope with severe computational constraints, single iterated GBSE approaches are recommended, and the best option, as seen before, depends on the shift size. If the underlying shift magnitudes are small, LS or TLS methods using short gradient estimation kernels support should be selected. If shifts are uniformly distributed along the [-1,1] interval, the bidirectional bias corrected (ULS) approach proves to be the best choice. For larger displacements, the recommended approach is multiscale GBSE, the method of Hoge [75] or the approach of Stone [159]. Under less demanding time constraints, gradient-correlation-based methods should be considered.

2.4.6 Evaluation on real MRI images

We evaluated every method against a set of real MRI images of a grapefruit. This set, provided by Hoge [75] together with its ground truth, is used in several articles of the state-of-the-art to contrast existing approaches. It consists of five different images shifted. To offer a more complete evaluation, we tested all shift estimation methods under five different noise levels injected to the images. Again, we assumed additive white Gaussian noise and the standard deviation for each noise level is shown in Table 2.6. In Fig. 2.25 we show the original image along with two other versions generated by manually adding white Gaussian noise.

For this experiment and due to the length of the underlying shifts to estimate, we included two multiscale GBSE methods using four and five scales, with increasing it-

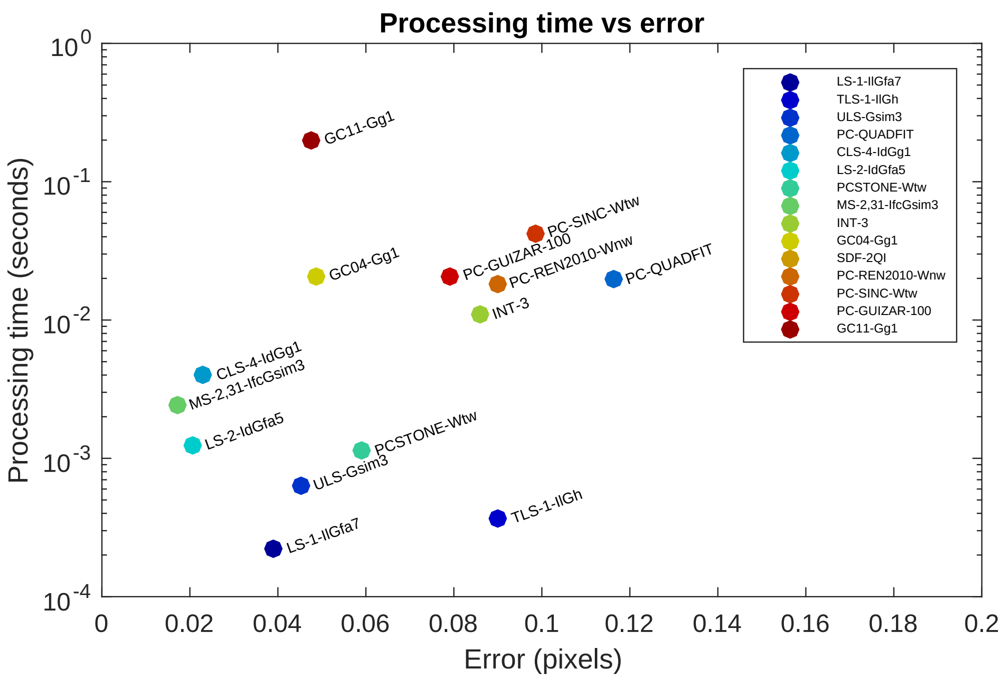
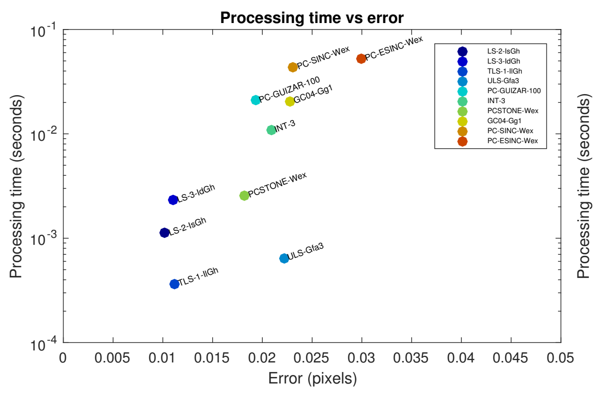
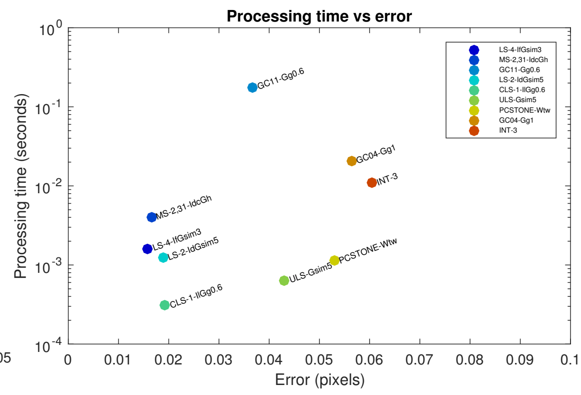


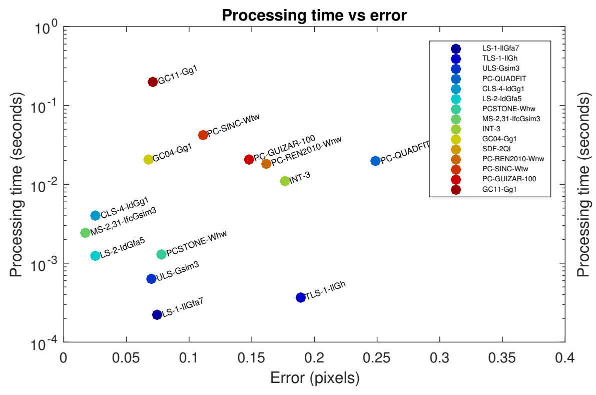
Figure 2.23 – Average execution time (log scale) vs accuracy for some representative methods averaged over the first three shift magnitudes and all noise levels



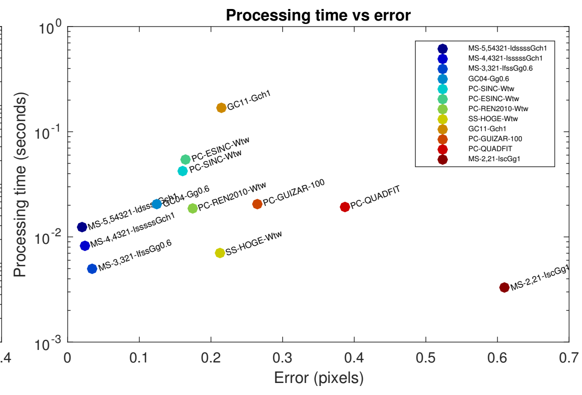
(a) Shift magnitude category 1



(b) Shift magnitude category 2



(c) Shift magnitude category 3



(d) Shift magnitude category 4

Figure 2.24 – Average execution time (log scale) vs accuracy for some representative methods for each shift magnitude category averaged over all noise levels

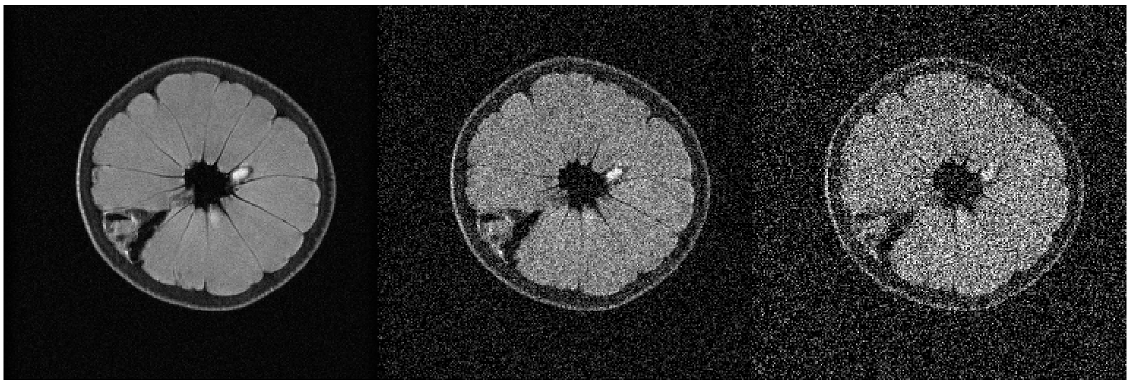


Figure 2.25 – First image of the MRI set provided by Hoge [75] under different noise levels. **Left**: Original image (level 1). **Middle**: $\sigma = 0.015$ (level 3). **Right**: $\sigma = 0.055$ (level 5)

erations across scales, spline interpolation and hypomode gradient, denoted MS-4 and MS-5 respectively. Results in Table 2.17 show that errors were considerably higher than what was obtained in the previous simulated experiments. To explain this, we used the ground truth to resample each image with respect to the other. In Fig. 2.26 we observe the absolute difference between an image and another aligned with it using the ground truth. As can be noted, for both left and middle images, the difference is considerably high on the image edges, which proves the images are not correctly registered, even though the ground truth was used. For the third case, the images are better registered and what is observed is mainly (signal dependent) noise and some minor aliasing. Therefore, this explains the significative errors achieved by most methods. This fact is indeed of major importance, since several articles compared their methods using this dataset [75, 134, 135, 179]. Based on this result, the analysis of performance for every method for the whole dataset is not feasible, thus ommited. However, as an example, we evaluated every method by estimating the shift between the fourth and the fifth images of the dataset, for which the provided ground truth appeared to be faithful (by observing the difference between both aligned images). Results appearing in Table 2.18 again prove that the multiscale GBSE approach is the best overall option, however, by a shorter margin in this case. The next best competitor is again gradient-correlation-based methods [7, 179]. The iterative periodic correlation method [153] also gathered acceptable results, however the obtained error was too high for the lowest evaluated SNR. As for phase-correlation approaches, the esinc [8] almost always appear as the best option for this images. The top 10 methods computed by taking the averages over all noise levels for these two images is shown in Table 2.19. We observe that only six method types achieved acceptable results for all noise levels.

$\sigma = 0.000$	$\sigma = 0.005$	$\sigma = 0.015$	$\sigma = 0.025$	$\sigma = 0.055$
0.1753 SS-HOGE-Wbh	0.1742 PC-LCM-2D2	0.1913 NGC11-Gch3	0.2021 GBSE-MS-4	0.2132 GBSE-MS-5
0.1815 PC-LCM-1D2	0.1856 SS-HOGE-Wbh	0.1927 PC-LCM-2D1	0.2075 APC-0.0001	0.2137 NGC11-Gg0.6
0.1850 APC-0.0001	0.1908 APC-0.0001	0.1941 APC-0.0001	0.2119 NGC11-G6	0.2210 GC11-Gg0.6
0.1894 NGC04-Gg0.6	0.1935 NGC11-G6	0.1953 GC11-Gch3	0.2121 GC11-G6	0.2241 GC04-Gg1
0.1942 NGC11-G5	0.1938 GC11-G6	0.1985 GC04-G6	0.2142 GC04-Gg1	0.2403 ADF-2QI
0.1942 GC11-G5	0.1948 MS-4	0.2039 GBSE-MS-5	0.2161 PC-ESINC-Wbh	0.2403 ADF2-2QI
0.1962 MS-4,4321-IssssGch3	0.2005 GC04-Gg1	0.2065 SDF-2QI	0.2270 SDF-2QI	0.2461 CFI-2QI
0.2025 GC04-G6	0.2046 CFI-2QI	0.2068 CFI-2QI	0.2279 CFI-2QI	0.2472 SDF-2QI
0.2043 SDF-2QI	0.2046 SDF-2QI	0.2155 ADF2-2QI	0.2304 ADF2-2QI	0.2493 PC-GUIZAR-10
0.2045 CFI-2QI	0.2115 ADF2-2QI	0.2160 ADF-2QI	0.2307 ADF-2QI	0.2549 APC-0.0001

Table 2.17 – Error values for top 10 best variants of each method averaged over all measurements for the grapefruit MRI dataset by injecting different amounts of additive white Gaussian noise. Note that these results cannot be relied upon, as the ground truth provided for this dataset appears to be incorrect.

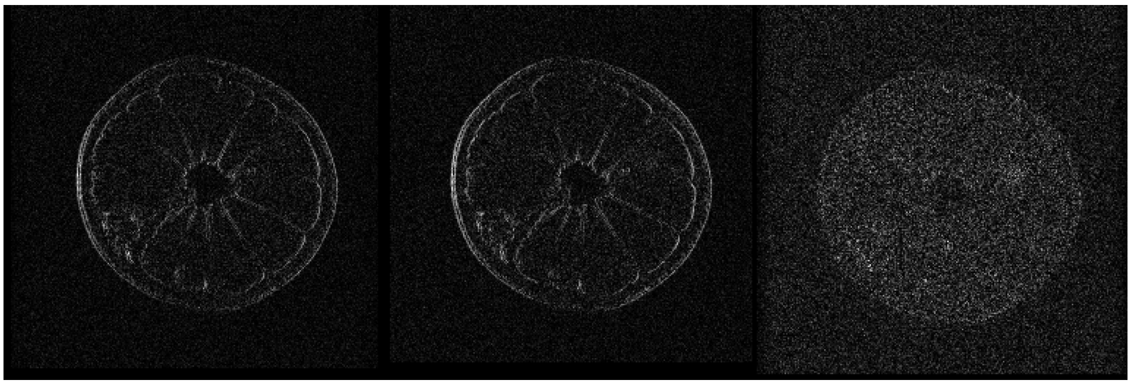


Figure 2.26 – Absolute value of the difference between aligned images based on the ground truth information provided by Hoge [75] using the original images. **Left**: Between the first and the third image. **Middle**: Between the first and the fifth image. **Right**: Between the third and the fifth image. Dynamic ranges extended for improved visual perception.

$\sigma = 0.000$	$\sigma = 0.005$	$\sigma = 0.015$	$\sigma = 0.025$	$\sigma = 0.055$
0.0024 MS-3,321-IfssGg1	0.0136 MS-3,321-IsssGsim3	0.0229 MS-3,321-IsssGsim5	0.0296 MS-3,321-IsssGfa3	0.0587 MS-3,321-IdssGfa5
0.0056 PC-ESINC-Wex	0.0162 GC11-Gch1	0.0264 GC11-Gg0.3	0.0357 GC11-Gg0.6	0.0597 GC11-Gg1
0.0062 NGC04-Gg1	0.0167 PC-GUIZAR-10	0.0275 NGC11-Gch1	0.0377 NGC11-Gg0.6	0.0634 NGC11-Gg1
0.0095 SS-HOGE-Wtw	0.0175 NGC11-Gch2	0.0307 GC04-Gg1	0.0420 GC04-Gg1	0.0672 GC04-Gg1
0.0115 NGC11-Gg1	0.0199 APC-3It	0.0394 APC-2It	0.0625 APC-0.01	0.1260 PC-ESINC-Wtw
0.0118 APC-0.0001	0.0245 GC04-Gg1	0.0423 PC-ESINC-Wtw	0.0690 PC-ESINC-Wtw	0.1458 APC-2It
0.0127 GC11-Gch3	0.0269 SS-HOGE-Wtw	0.1838 SS-HOGE-Whw	0.3425 PCFOO	0.4030 PCFOO
0.0243 GC04-Gg1	0.2594 PCFOO	0.2930 PCFOO	0.3645 SS-REN2014-Wtw	0.9873 ACC-2It
0.0842 ACC-0.0001	0.3413 ACC-3It	0.4506 ACC-3It	0.7081 ACC-2It	1.5331 SS-REN2014-Whw
0.1370 PCFOO	3.5684 CLS-4-IdGfa7	4.2313 CLS-4-IfGfa7	4.2080 LS-4-IdGfa7	4.7611 LS-4-IsGfa7

Table 2.18 – Average error values for top 10 best variants of each method by estimating the shift between the fourth and the fifth images of the grapefruit MRI dataset (ground truth shift: $(7.68, 0)$) by injecting different amounts of additive white Gaussian noise.

Avg
0.0282 MS-4,4321-IsssssGsim3
0.0317 GC11-Gg0.6
0.0332 NGC11-Gg0.6
0.0377 GC04-Gg1
0.0558 PC-ESINC-Wtw
0.0563 APC-3It
0.2870 PCFOO
0.5266 ACC-3It
0.5683 SS-REN2014-Whw
4.0268 CLS-4-IdGfa7

Table 2.19 – Average error values for all levels of noise for top 10 best variants of each method by estimating the shift between the fourth and the fifth images of the grapefruit MRI dataset (ground truth shift: $(7.68, 0)$).

2.5 Concluding Remarks

In this chapter we performed an in-depth review of fast and accurate shift estimation methods. Although most recent work on shift estimation is based on phase-correlation methods, GBSE approaches proved to be more accurate, stable and run faster than phase correlation approaches, although they are limited to subpixel estimation or required to be used in a pyramidal multiscale approach in order to estimate larger shifts. On the other side, gradient-correlation-based methods showed an improvement over phase correlation methods, particularly under higher noise or highly aliased scenarios, although they usually require more computational resources. Throughout this chapter, we found out that if *a-priori* knowledge is available regarding the maximum shift magnitude to estimate, then better estimation accuracies can be achieved by selecting a more suitable variant for each method. We provided the reader with such fine-tuning tips. Another interesting discovery is that one of the most used ground truth data used for comparison among state-of-the-art methods is not precise and should be avoided. Finally and most importantly, based on this study and on the characteristics of the underlying shift estimation problem, a practical recipe is offered to the community in order to achieve fast and accurate shift estimation.

Despite the extensive length of this review, we plan to include other useful information. First, some interesting methods were excluded from the final comparison due to the complexity of their implementations or high computational times. These were the methods of Takita [162], Balci [13], Leprince [90] and the recent method of Tong *et al.* [170]. We plan to add them in the future. Another considered task, is to test the robustness of each method under varying illumination conditions between the images. We believe that in these cases, if no histogram equalization is possible, GBSE methods should be avoided and phase or gradient correlation approaches would become first options. Also, we plan on combining different presented approaches to obtain a more general method yielding good results on every situation. Finally, we plan to provide publicly a ground truth dataset to be used for comparison of shift estimation methods, considering all difficulties usually faced on such task.

Acknowledgements

I would like to thank Prof. Vasileios Argyriou for providing several source codes helping the development of this review. We also thank Dr. Tuan Q. Pham for fruitful comments and discussions.

Chapter 3

Improving wavefront sensing with a Shack-Hartmann device

In order to achieve higher resolutions, current earth-observation satellites use larger light-weight primary mirrors that can deform over time, impacting on image quality. We evaluated the possibility of compensating these deformations directly in the satellite by combining a deformable mirror with a Shack-Hartman wavefront sensor (SHWFS). The performance of the SHWFS depends entirely on the accuracy of the shift estimation algorithm employed, which should be computationally cheap to be executed on-board. We analyzed the problem of fast accurate shift estimation in this context and propose a new algorithm, based on a global optical flow method that estimates the shifts in linear time. In our experiments, our method proved to be more accurate and stable, as well as less sensitive to noise than all current state-of-the-art methods, permitting a more precise on-board wavefront estimation.

3.1 Introduction

Adaptive optics (AO) is a well-known technology to sense and correct wavefront distortions. This technology is used in astronomy to produce sharper images from heavily aberrated wavefronts originated by atmospheric turbulence. This correction is usually performed through a deformable mirror which adapts to the measured wavefront correcting the distortion [189]. Since AO also allows to improve the performance of aberrated optical systems, it is widely used in several other contexts such as ophthalmology [188], microscopy [22] and free-space laser communication systems [177] among others [176].

A key component of an adaptive optics system is the wavefront sensing mechanism, i.e., the device used to precisely measure the distortion. A Shack-Hartmann wavefront sensor (SHWFS) is one such device. It uses an array of lenslets to measure the deformation of the incoming wavefront. The shift of each lenslet focal plane image is proportional to the mean slope of the wavefront in the subaperture onto this lenslet. It yields a discrete local approximation of the slope of the wavefront (Fig. 3.1). This deformation is usually measured by imaging a point source such as a star, and estimating the relative displacement between a reference image and all other subimages to compute the local gradient of the wavefront. The measured slopes are then used to approximate the actual wavefront.

Recently, the community evaluated the possibility of correcting wavefront deformations on earth-observation satellites [21,52,124] caused by the deformation of the primary mirror. In this setting, the problem of atmospheric turbulence is negligible. Indeed, in astronomical observations from the earth, the angle of view is extremely narrow. As a result, the light wavefront crosses a narrow solid angle of atmosphere and its perturbations due to turbulence have a great impact on image quality. In earth observation from a satellite, however, the angle of view is much larger, so the perturbations due to turbulence are relatively much smaller.

However, the correction of optical aberration is becoming more and more important for high resolution earth-observation satellites. Indeed, in order to increase the spatial

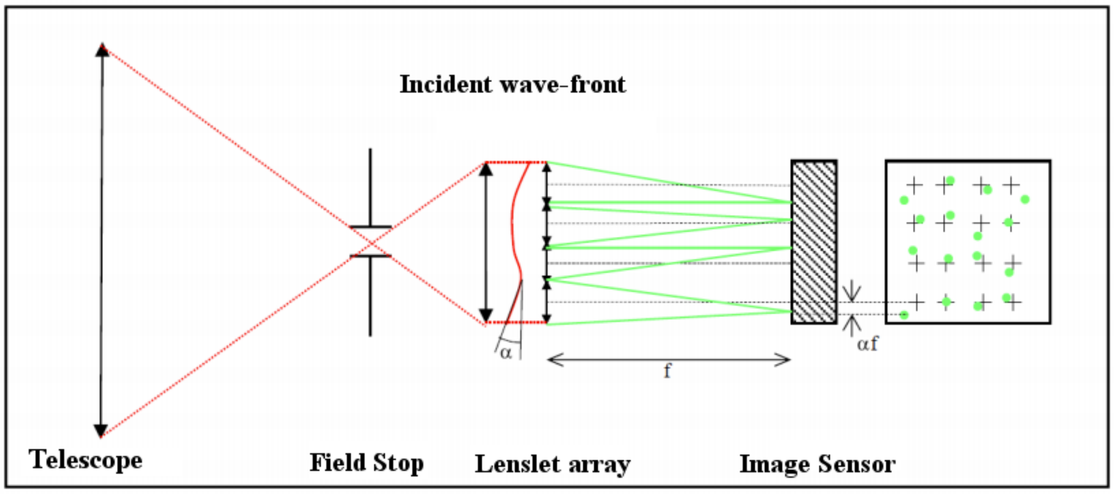


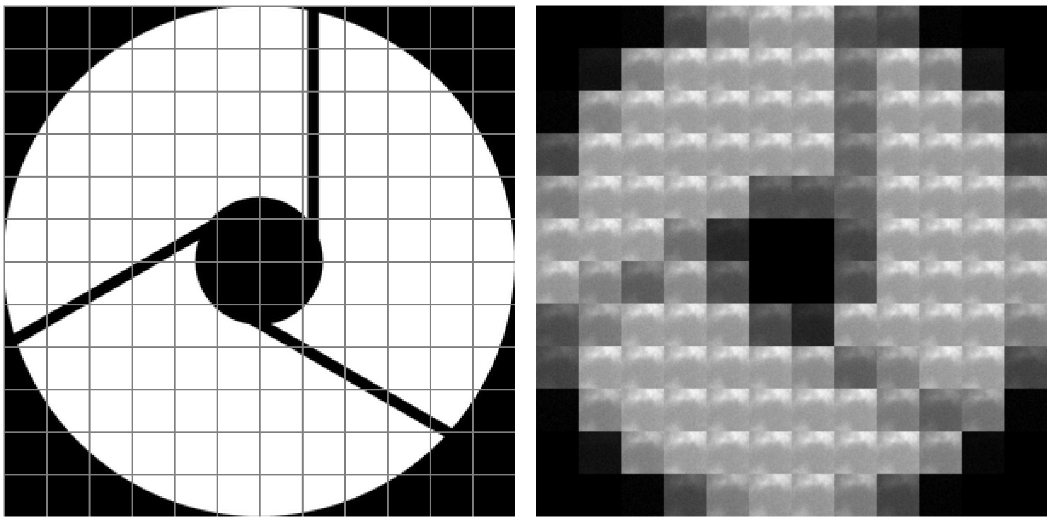
Figure 3.1 – A Shack-Hartmann Wavefront Sensor measures the wavefront by computing the local shifts between the detected spots (in green) and the reference crosses (in black), which would occur if no deformation was present.

resolution of satellite images, i.e., its ground sample distance, a larger primary mirror is required to gain a higher angular resolution. Large mirrors must be thicker to avoid deformations which increase dramatically their weight and render launching costs prohibitive. For this reason large yet lighter mirrors have to be considered. Their drawback is that time-varying deformations due to thermal effects and vibration severely deteriorate the image quality [30]. To correct these deformations, a SHWFS device could be used to measure them by observing the earth, together with a deformable mirror to compensate for these deformations. As opposed to adaptive optics where the distortion has to be compensated in real-time due to the fast changes in the atmosphere, in our case the compensation for the mirror deformation is less frequent, roughly on timescales of seconds.

Coming from control theory, two schemes exist for performing wavefront correction. While in an *open loop* adaptive optics system the wavefront error is measured before it has been corrected by the wavefront corrector, in a *closed loop* system the measured wavefront deformation is the residual error after the correction of the previous estimation has been performed. The difference between both operational modes is important because in a *closed loop* system, the wavefront aberrations measured will be small, permitting to assume a maximum shift between images up to a few pixels. In this chapter we assume a *closed loop* system.

Another important factor that affects the accuracy of wavefront sensing is the phenomenon of scintillation and phase anisoplanatism, which result in more complex patterns than simple global shifts between the subapertures. The influence of this phenomenon was widely studied in the context of adaptive optics [126, 137, 182]. However, in remote sensing, because the phase aberration is produced at the telescope pupil, all parts of the image are affected by it in the same way, neglecting its incidence [126].

Finally, once the shift estimation is performed and all wavefront slopes have been estimated, several methods can be used to reconstruct the wavefront from its local gradient estimations, namely the iterative zonal method by Southwell [157], the vector-matrix-multiply (VMM) method [73] and the Fourier Transform Reconstruction (FTR) method [125]. The latter is recommended when the number of actuators is high, however, the VMM method obtains more accurate results under a SHWFS configuration using less than 12×12 subapertures [141].



(a) Occlusion schema in a 12×12 SHWFS. (b) Images obtained when observing an extended landscape.

Figure 3.2 – Example of a SHWFS occlusion scheme under a Korsch telescope. Both the secondary mirror and the arms used to hold it are clearly visible. For each lenslet, the decrease of the incoming signal is proportional to its occlusion.

3.1.1 Wavefront sensing from earth-observation satellites

As mentioned before, wavefront sensing in astronomy is usually performed using the stars. When observing the earth from space this task becomes more challenging. Indeed, there are some key differences between performing wavefront sensing for earth-observation satellites and for astronomical observation, namely,

Extended scene vs point source observation. For an earth observation satellite, the SHWFS is used on extended scenes instead of point sources such as stars. This setup is called scene-based wave-front sensing (SBWFS), or extended-scene wavefront sensing (ESWFS). Because the scene is extended, a field stop has to be installed in front of the SHWFS, as seen in Fig. 3.1, so that the images given by the lenslet array do not overlap in the lenslet focal plane [108]. This yields a grid of images, each one corresponding to one sub-aperture, which are shifted versions of the same scene. Accurately measuring these shifts permits to estimate the gradient of the wavefront. As a drawback, since we are dealing with landscapes larger than the captured image, achieving high accuracy on the shift estimation task gets challenging. Worse still, in most wavefront sensors, the extent of the source object normally reduces the contrast of the signal, thwarting accurate shift measurements [144].

Difference in subimage SNRs due to pupil occlusions. For long focal length telescopes commonly used on earth-observation satellites, the Korsch concept is the most common. The pupil of a Korsch telescope is generally occluded in the center by a secondary mirror. The arms used to hold this mirror also occlude the pupil. In these regions, the lenslets suffer a loss in the incoming signal, proportional to their percentage of occlusion. This configuration is depicted in Fig. 3.2(a). An example of a SHWFS output in the CRT sensor is shown in Fig. 3.2(b). From this figure, it can be seen how the SNR on the partially occluded lenslets is significantly lower.

Limited on-board computational capacity. An important distinction when performing wavefront sensing from earth-observation satellites is its limited computational capacity. Due to this constraint several shift estimation methods proposed for SHWFS are not suitable to be implemented on-board due to their high complexity.

Unusable observations. Another difference when observing extended scenes, as opposed to the use of SHWFS with point sources, is the need to predict if the current scene permits to accurately estimate the wavefront aberration. Scenes such as clouds, sea or any textureless landscape can thwart all shift measurement methods, leading to poor wavefront estimation.

Based on these differences, we present in this chapter a new shift estimation method in the context of SHWFS used on extended scenes. Our contribution is threefold. We shall start with a review of the state-of-the-art on wavefront correction using SHWFS on extended scenes. Second, we propose the use of an iterative global optical flow method for shift estimation which presents several advantages over the conventional correlation methods. Third, we propose a fast and effective method for scene preselection that adds almost no further computational cost to the overall estimation using the proposed algorithm.

This chapter is organized as follows. In Section 3.2 we review the state-of-the-art on shift estimation applied for SHWFS on extended scenes. In Section 3.3 we present our optical flow method and its usage for wavefront correction from earth-observation satellites. In Section 3.4 we focus on scene pre-selection methods. In Section 3.5 we study the influence of the parameters on the proposed method, and thoroughly compare its performance with the state-of-the-art. We involve ground truth simulations provided by CNES and our own simulator. We conclude and present some future work in Section 3.6.

3.2 State-of-the-art review

Since the Shack-Hartmann wavefront sensor was introduced in the late 1960s [123], several algorithms were proposed to estimate the shifts using point sources such as stars. However, only few authors have studied the problem of an extended source, occuring when observing the earth from space. Broadly, they can be categorized in correlation methods working in the spatial [95, 108, 109] or in the frequency domain [85, 95, 124, 151, 153], phase correlation methods that estimate the shift directly in the frequency domain [85, 159], iterative methods [151, 153] that improve on previous approaches by progressively estimating the shift and maximum likelihood (ML) approaches [68] that incorporate a specific noise model to the problem and compute ML estimate as the solution of an optimization problem.

3.2.1 Correlation-based Methods

These methods compute a correlation score on a grid $C(i, j)$ and interpolate it to determine the subpixel location of the peak. The methods mainly differ in the choices of correlation score and interpolation strategy.

Spatial domain correlation-based methods. Michau *et al.* [109] were among the first to propose an experimental implementation for using a SHWFS on extended sources larger than the wavefront sensor field of view. Their method computes the discrete cross-correlation between the images and a reference subimage chosen from the central region of the wavefront sensor image pattern. To estimate the subpixel shifts, the correlation peak location is computed as the centroid of the pixels with a correlation higher than half the maximal observed correlation. However, for asymmetric peaks in the correlation function, the peak position obtained by this method differs from the true maximum, leading to considerable errors. In particular, this often occurs in low SNR scenarios, frequent for earth-observation satellites.

In another publication, Michau *et al.* [108] compared several methods of shift estimation for extended sources. Assuming Gaussian noise and scenes where the extent of the source is smaller than the field stop, the maximum likelihood estimator (MLE) is computed by maximizing a correlation-type function between both images. However, the authors show that using the cross-correlation method for shift estimation is biased for extended scenes. They also stress that for Poisson noise, the MLE maximizes the correlation between the first image and the logarithm of the reference image, yet this estimator is also biased when working with extended scenes.

Löfdahl [95] tested several shift estimation algorithms for Shack-Hartmann wavefront sensors observing the sun. By testing with several possible sources of errors such as noise, blur and bias mismatch, he evaluated five different correlation-based methods to obtain the correlation score $C(i, j)$. Among the evaluated correlation algorithms, the best all-around performer proved to be the classical least squares approach or squared difference function (SDF), given by

$$SDF : C(i,j) = \sum_{m,n} \left[I_1(m,n) - I_{ref}(m+i, n+j)\right]^2 . \tag{3.1}$$

Another correlation score can be calculated as the Covariance Function in the Image domain

$$CFI : C(i,j) = -\sum_{m,n} \check{I}_1(m,n) \cdot \check{I}_{ref}(m+i, n+j), \tag{3.2}$$

where $\check{I}$ is the trend-corrected version of I where a fitted plane has been subtracted. Indeed, Smithson & Tarbell [163] showed that a linear trend in intensity shifts the covariance peak from the correct position, so before computing the method, a fitted plane has to be subtracted from both images. In practice for the SHWFS case, the authors simply subtracted the mean value for each image yielding

$$\check{I}(x,y) = I(x,y) - \frac{1}{N} \sum_{x,y} I(x,y). \tag{3.3}$$

They also tested other two methods based on the absolute difference between both images (ADF), and its square (ADF^2) in order to better locate the minimum at the sub-pixel level,

$$ADF = C(i,j) = \sum_{m,n} |I_1(m,n) - I_{ref}(m+i, n+j)|, \tag{3.4}$$

$$ADF^2 = C(i,j) = \left(\sum_{m,n} |I_1(m,n) - I_{ref}(m+i, n+j)| \right)^2 . \tag{3.5}$$

To achieve subpixel precision, they evaluated four interpolation strategies to look for the minimum value on the correlation grid $C(i_{min}, j_{min})$. The four algorithms they evaluated can be described as fitting a conic section,

$$f(x,y) = a_1 + a_2 x + a_3 x^2 + a_4 y + a_5 y^2 + a_6 xy \tag{3.6}$$

to the 3×3-element submatrix s centered in the sample minimum $C(i_{min}, j_{min})$.

$$s_{i,j} = C(i + i_{min}, j + j_{min}); \qquad i,j = -1, 0, 1. \tag{3.7}$$

The evaluated algorithms differ on the number of pixels used and on whether the fitting is done in 2D or in each dimension separately. The 1D quadratic interpolation is given by setting

$$\begin{aligned} &a_1 = s_{0,0}, \quad a_2 = (s_{1,0} - s_{-1,0})/2, \quad a_3 = (s_{1,0} - 2s_{0,0} + s_{-1,0})/2, \\ &a_4 = (s_{0,1} - s_{0,-1})/2 \text{ and } a_5 = (s_{0,1} - 2s_{0,0} + s_{0,-1})/2. \end{aligned} \tag{3.8}$$

The 2D version of this algorithm is obtained by fitting the 2D conic of (3.6) into the 3×3 values, yielding the same parameters as before, in addition to

$$a_6 = (s_{1,1} - s_{-1,1} - s_{1,-1} + s_{-1,-1})/4, \tag{3.9}$$

which logically implies using the values of the corners.

They also evaluated a 1D Least Square (1LS) method, separately for both dimensions, which is performed by averaging the 3 values on one dimension and then performing the least squares on the other. The parameters a_1 to a_5 are then the same, however, in a_2 and a_3, instead of taking $s_{1,0}, s_{0,0}$ and $s_{-1,0}$, they use $\langle s_{1,j} \rangle_j, \langle s_{0,j} \rangle_j$ and $\langle s_{-1,j} \rangle_j$, and in a_4 and a_5, instead of $s_{0,1}, s_{0,0}$ and $s_{0,-1}$, they use $\langle s_{i,1} \rangle_i, \langle s_{i,0} \rangle_i$ and $\langle s_{i,-1} \rangle_j$. In this setting, $\langle \cdot \rangle_i$ denotes averaging over index $i \in \{-1, 0, 1\}$. For the 2D version, 2LS has the same parameters as 1LS and the parameter a_6 is identical to (3.9).

Finally, the 1D minimum is given by

$$\hat{v}_x = i_{min} - \frac{a_2}{2a_3}, \qquad \hat{v}_y = j_{min} - \frac{a_4}{2a_5} \tag{3.10}$$

and the 2D minimum is

$$\hat{v}_x = i_{min} + \frac{2a_2a_5 - a_4a_6}{a_6^2 - 4a_3a_5}, \qquad \hat{v}_y = j_{min} + \frac{2a_3a_4 - a_2a_6}{a_6^2 - 4a_3a_5}. \tag{3.11}$$

Frequency domain correlation-based methods. Poyneer [124] studied the wavefront estimation problem using a SHWFS by observing the Earth from space using lightweight optics. By assuming periodicity on the input images, the author points out that minimizing the MSE between both images becomes equivalent to maximizing their periodic convolution, which is efficiently computed in the frequency domain using the cross-correlation theorem

$$C(i,j) = \mathcal{F}^{-1}\{\hat{I}_{ref}^* \hat{I}_1\}(i,j), \tag{3.12}$$

where $\mathcal{F}^{-1}$ stands for the inverse Fourier transform, $\hat{I}_1$ and $\hat{I}_{ref}$ are the Fourier representations of images I_1 and the reference image I_{ref} respectively, and $\hat{I}_{ref}^*$ denotes the complex conjugate of $\hat{I}_{ref}$. To get subpixel precision a parabola is fitted and its maximum is computed as

$$\hat{v_x} = i_{max} + \frac{\frac{1}{2}\left(C(i_{max}-1, j_{max}) - C(i_{max}+1, j_{max})\right)}{C(i_{max}-1, j_{max}) + C(i_{max}+1, j_{max}) - 2C(i_{max}, j_{max})},$$

where $C(i_{max}, j_{max})$ is the maximum value in integer coordinates of the correlation grid. The estimate for $\hat{y_0}$ is obtained in an analog way. This algorithm suffers from errors due to the wraparound of pixels and the inherent non-periodicity of the input images.

The computational complexity of the entire method is $O(SN^2 \log N)$ where S is the amount of subapertures and N the image size. The author compared this method with the $O(N^4)$ MSE computation for all possible overlaps, and showed that for high-quality images, the difference is negligible. This is no more true when images are small and with poor SNR conditions, in which case the method achieves less accurate results.

Löfdahl [95] also evaluated a frequency-domain method which, similarly to the method of Poyneer, computes the covariance in the Fourier domain (CFF). However, both images are previously normalized to zero mean and windowed with a 2D Hamming window or a flat-top window to avoid ringing caused by the periodization, by calculating

$$CFF : C(i,j) = -\mathcal{F}^{-1}\left\{\mathcal{F}\{w(m,n)\check{I}(m,n)\}\mathcal{F}^{*}\{w(m,n)\check{I}_{ref}(m,n)\}\right\}. \tag{3.13}$$

Again, the subpixel maximum is obtained by fitting a parabola to the grid.

3.2.2 Phase Correlation

We give here a quick recall about the phase-correlation method, which was explained in detail in Chapter 2 of this manuscript. As already mentioned, the phase-correlation method was widely studied in the image processing domain [57, 159, 162]. Let $I(i,j)$ be an $M \times N$ image, due to the Fourier shift theorem we know that

$$\mathcal{F}\{I(i-\Delta_x, j-\Delta_y)\} = \hat{I}(\omega_x,\omega_y)\exp\left(-j2\pi\left(\frac{\omega_x v_x}{M}+\frac{\omega_y v_y}{N}\right)\right) \tag{3.14}$$

then by computing the cross-power spectrum between both images

$$\hat{C}(u,v) = \frac{\hat{I}_{ref}(\omega_x,\omega_y)\hat{I}_1^{*}(\omega_x,\omega_y)}{|\hat{I}_{ref}(\omega_x,\omega_y)\hat{I}_1^{*}(\omega_x,\omega_y)|} = \exp\left(j2\pi\left(\frac{\omega_x v_x}{M}+\frac{\omega_y v_y}{N}\right)\right) \tag{3.15}$$

and extracting the phase for each frequency of this function, the matrix ϕ, called the phase difference matrix, is given by

$$\arg(\hat{C})(\omega_x,\omega_y) = \phi(\omega_x,\omega_y) = 2\pi\left(\omega_x v_x/M + \omega_y v_y/N\right). \tag{3.16}$$

Classic phase correlation estimates the shift in the spatial domain as the peak of the inverse Fourier transform of $\hat{C}(\omega_x,\omega_y)$. However, the shifts can be computed directly in the frequency domain by fitting a plane passing through the origin of the phase difference matrix $\phi(\omega_x,\omega_y)$. By assuming square images ($M = N$), the shift is estimated by calculating

$$\hat{v_x} = \frac{M}{2\pi}\frac{b_p a_{qq} - b_q a_{pq}}{a_{pp}a_{qq} - a_{pq}^2}, \quad \hat{v_y} = \frac{M}{2\pi}\frac{b_q a_{pp} - b_p a_{pq}}{a_{pp}a_{qq} - a_{pq}^2}, \tag{3.17}$$

where

$$a_{pp} = M\sum_{\omega_x=-\frac{M}{2}}^{\frac{M}{2}-1}\omega_x^2, \quad a_{qq} = M\sum_{\omega_y=-\frac{M}{2}}^{\frac{M}{2}-1}\omega_y^2, \quad a_{pq} = \sum_{\omega_x=-\frac{M}{2}}^{\frac{M}{2}-1}\sum_{\omega_y=-\frac{M}{2}}^{\frac{M}{2}-1}\omega_x\omega_y,$$

$$b_p = \sum_{\omega_x=-\frac{M}{2}}^{\frac{M}{2}-1}\sum_{\omega_y=-\frac{M}{2}}^{\frac{M}{2}-1}\phi(\omega_x,\omega_y)\omega_x \text{ and } b_q = \sum_{\omega_x=-\frac{M}{2}}^{\frac{M}{2}-1}\sum_{\omega_y=-\frac{M}{2}}^{\frac{M}{2}-1}\phi(\omega_x,\omega_y)\omega_y. \tag{3.18}$$

Due to aliasing, some of these frequencies may be heavily corrupted, considerably worsening the shift estimation. Therefore, Knutsson *et al.* [85] discard most corrupted frequencies from Eqs. (3.17) and (3.18), yielding an extremely fast algorithm that uses only two frequency values:

$$\hat{v_x} = \frac{M}{2\pi}\phi(u_1,0), \quad \hat{v_y} = \frac{M}{2\pi}\phi(0,v_1). \tag{3.19}$$

In particular, from [159], the estimate based on $u_1 = v_1 = 1$ has the advantage of being the least sensitive to aliasing and therefore, the most reliable. By using only these two frequencies and assuming the shift between both images is less than half the field of view, then no phase unwrapping is required [63]. Moreover, there is no need to compute the whole FFT since just two frequencies are required. However, using two frequencies may not be enough for accurate shift estimation, therefore the authors of [85] proposed to use four frequency components ($\phi(1,-1), \phi(1,0), \phi(1,1)$ and $\phi(0,1)$), for which they showed an improvement on irregular images. Even under these circumstances, as pointed out by Poyneer [124], the accuracy suffers considerably under low SNR situations.

3.2.3 Iterated Estimation

Since correlation-based shift estimation has a bias proportional to the shift magnitude [119], then compensating the shift by resampling one of the images and iterating should progressively reduce this bias (see section 3).

Sidick et al. [151] proposed Adaptive Cross-Correlation (ACC) that estimates the shift using a similar approach to the one in Knutsson [85] and uses this estimation to resample the second image in the frequency domain to iterate the procedure. Due to potential ringing artifacts on the image boundaries after the resampling procedure, the shift is estimated using the central part of both images with size $N/2$, involving a fourth of the image pixels in the computation. To make the shift estimation more accurate, they use eight frequency components $\phi(\omega_x, \omega_y)$ with $0 \leq \omega_x, \omega_y \leq 2$ to perform the least-squares fitting, excluding the center. Therefore, given a SHWFS configuration of $N \times N$ sub-images where N is taken preferably to be a power of 2, the algorithm that estimates the shift between I_k and I_{ref} does the following:

1. Take the central $N' \times N'$ pixel part of $I_{ref}(i,j)$, where $N' < N$ (usually $N' = N/2$) and compute its FFT $\hat{I}'_{ref}(\omega_x, \omega_y)$. Do the same with $I_k(i,j)$ obtaining $\hat{I}'_k(\omega_x, \omega_y)$.

2. Compute the cross-correlation in the Fourier domain: $C_k(\omega_x, \omega_y) = [\hat{I}'_{ref}(\omega_x, \omega_y)]^* \hat{I}'_k(\omega_x, \omega_y)$ where * denotes the complex-conjugate. Then obtain the complex phase $\phi_k(\omega_x, \omega_y)$ of $C_k(\omega_x, \omega_y)$.

3. Estimate the shift $(\hat{v_x}, \hat{v_y})$ performing least squares fitting by using Eq. (3.17). In the minimization, use the eight frequency components given by taking $0 \leq \omega_x, \omega_y \leq 2$ and excluding the $\phi(0,0)$ frequency. Based on [85], these frequencies have the advantage of being less sensitive to aliasing, leaving the low-spatial-frequency, high-contrast components in the image but rejecting high-spatial-frequency low SNR completely from the shift estimate. This approach is similar to the one used by Stone *et al.* [159].

4. Resample $I_k(\omega_x, \omega_y)$ in the frequency domain by computing the FFT $\hat{I}_k(\omega_x, \omega_y)$ of the whole $N \times N$ image, followed by its multiplication by the estimated ω_x and ω_y slopes and taking the inverse FFT of this product:

$$I_k(i + \Delta_x, j + \Delta_y) = \mathcal{F}^{-1}\left\{\hat{I}_k(i,j) \exp\big(-j2\pi(\hat{v_x}\omega_x + \hat{v_y}\omega_y)\big)\right\}. \tag{3.20}$$

This procedure is made iterative by accumulating both $\hat{v_x}$ and $\hat{v_y}$ shift estimates, until a predefined number of iterations is reached or until $\Delta = [\hat{v_x}^2 + \hat{v_y}^2]^{1/2}$, the increase of both shift estimates, is lower than a predefined tolerance value. This technique is then applied for every sub-aperture $I_k(i,j)$.

This method greatly resembles to the one proposed in [159]. Its lack of robustness is compensated by making the algorithm iterative. However, since Fourier resampling is done, the underlying images must validate the Nyquist criterion, otherwise ringing artifacts may appear that could lead to a poor shift estimation.

This algorithm was evaluated on simulated scenes by varying the shift between them [151] and by using a real SHWFS configuration testbed [152] produced in the Jet Propulsion Laboratory at NASA [113]. By relaxing the tolerance parameter of their iterative scheme, it achieves errors of the order of 0.05 pixels by using between 3 and 6 iterations.

The method presents two problems. Since it discards half of each image to perform the shift estimation, it usually fails when image contents appear away from the center. Also, as in [85], the shift estimation step of the method suffers severely in low SNR scenarios. Due to this reason, the authors updated the method in [153] by replacing this step with the periodic-correlation technique of Poyneer [124]. The resulting method, named Adaptive Periodic Correlation (APC), proved to be more tolerant to noise than ACC.

3.2.4 Maximum Likelihood Estimator

Gratadour et al. [68] performed an in depth study of the performance of the MLE within an adaptive optics context. Assuming that the two images I_0 and I_1 are unknown and contaminated with Gaussian noise their method reduces to minimizing the following energy with respect to the shift x_1

$$-\,ln(\mathcal{L}(x_1)) = \textstyle\sum_k \frac{1}{4\sigma^2(k)}|I_1(k) - [I_0(x-x_1)]_{III}(k)|^2, \tag{3.21}$$

where $\sigma^2(k)$ is the noise variance at pixel k and $[]_{III}$ a sampling operator. In practice, to sample from the shifted image $[I_0(x-v_x)]_{III}(k)$, they perform the shift using (3.14) by first applying a low-pass filter to reduce the effect of noise and high frequencies approximation as

$$[I_0(x-v_x)]_{III}(k) = \mathcal{F}^{-1}\left[\tilde{I}_0 \times \Pi_{u_c}(\omega) \times e^{-2i\pi\omega v_x}\right], \tag{3.22}$$

where $\tilde{I}_0$ denotes the Fourier transform of image I_0 and $\Pi_{u_c}(u)$ is a low-pass mask in the Fourier domain. The cut-off is chosen to be the diffraction cut-off frequency of the telescope. Finally, this minimization is performed numerically using a conjugate gradient method. Although this algorithm improves over the typical cross-correlation, it has high computational cost prohibiting its implementation on satellites [108, 124].

As a second contribution, the authors of [68] propose a joint maximum likelihood approach in which they estimate simultaneously a non-noisy reference image and the shift parameters for all the subapertures of the SHWFS. Interestingly, the reference image that maximizes this joint likelihood is found to be the mean of all shifted images. Despite its high computational cost, this joint estimation is shown to improve over pairwise estimation, in particular under lower SNR conditions.

3.3 Accurate shift estimation using optical flow in the context of a SHWFS

In order to estimate the shift between two sub-apertures and based on the results of Chapter 2 of this thesis, we propose to use a gradient-based shift estimator (GBSE) based on the optical flow equation. For the sake of completeness, we recall this method in this chapter which includes some repetitions from the former explanation, however we refer the reader to section 2.2 for a full description of the method. We include in this section some new experiments which are tested particularly on SHWFS configurations.

The idea behind GBSE, as proposed originally by Lucas and Kanade [97], is to relate the difference between two successive frames to the spatial intensity gradient of the first image. Given the two images $I_2(x,y) = I_1(x-v_x, y-v_y)$ where v_x and v_y are the unknown shift coefficients, the first order Taylor approximation yields

$$I_1(x,y) - I_2(x,y) \approx v_x \frac{\partial I_1(x,y)}{\partial x} + v_y \frac{\partial I_1(x,y)}{\partial y}. \tag{3.23}$$

To estimate the global optical flow between I_1 and I_2, the Lucas-Kanade algorithm assumes a constant flow for every pixel, which allows the construction of an overdetermined system of constraint equations $\mathbf{A}\mathbf{v} = \mathbf{b}$, where $\mathbf{A}$ is composed of spatial intensity derivatives and $\mathbf{b}$ has temporal derivatives. Emulating Simoncelli [154], in order to increase the accuracy of the method by minimizing noise or aliasing influence, we looked for two kernel functions: an anti-symmetric kernel $\mathbf{d}$ to estimate the image gradients and a symmetric kernel $\mathbf{k}$ to prefilter the images. Using both kernels, matrix $\mathbf{A}$ and vector $\mathbf{b}$ become

$$\mathbf{A} = \begin{pmatrix} (\mathbf{d}_x * I_1)(p_1) & (\mathbf{d}_y * I_1)(p_1) \\ \vdots & \vdots \\ (\mathbf{d}_x * I_1)(p_n) & (\mathbf{d}_y * I_1)(p_n) \end{pmatrix}, \mathbf{b} = \begin{pmatrix} (\mathbf{k} * (I_1 - I_2))(p_1) \\ \vdots \\ (\mathbf{k} * (I_1 - I_2))(p_n) \end{pmatrix} \tag{3.24}$$

where p_i with $i = 1 \dots n$ represents the ith pixel and n the image size. The shift is obtained by Moore-Penrose pseudo-inversion

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}^{-1}}_{\mathbf{A}^\mathbf{T}\mathbf{A}} \underbrace{\begin{bmatrix} \sum I_t I_x \\ \sum I_t I_y \end{bmatrix}}_{\mathbf{A}^\mathbf{T}\mathbf{b}} \tag{3.25}$$

where I_x, I_y stands for $\mathbf{d}_x * I_1$ and $\mathbf{d}_y * I_1$ respectively, and $I_t = \mathbf{k} * (I_1 - I_2)$ is the derivative over time.

It is no coincidence that the results of the method depend on the inversion of this second moment matrix. As will be shown later, the determinant of this matrix is crucial for determining the limits on the estimation performance. This matrix will be used to discard ill-posed cases before actually performing the shift estimation. The classic rejection case is when the gradient is mostly oriented in a single direction. This unsolvable situation is known as the aperture problem. Its detection by using the Cramer-Rao lower bounds is detailed in Section 3.4.

It is important to remark that by centering the Taylor development at zero and taking up to the first order term, the method gets systematically biased and the method becomes less precise as the shift gets higher, i.e., estimated shifts larger than one pixel would not be correctly estimated. Furthermore, the noise in the input images is completely ignored by the algorithm, which also impacts on its performance. Both these bias sources are studied in detail in [119, 139] and will be addressed next.

3.3.1 Iterative gradient-based shift estimator

Instead of dealing with the bias explicitly, it was shown in Pham et al. [119] that both bias sources depend linearly on the shift magnitude. This justifies the use of an iterative method, which is able to significantly reduce the bias, provided an appropriate resampling method is used. Algorithm 3 performs k iterations, computing the shift $v[i]$ by solving (3.25) ($findshift$), and iterates by reinterpolating the original image I_2 ($Resample$) using the total accumulated shift w. It is a variant of the Lucas-Kanade algorithm [97] known as Inverse Compositional Algorithm [12], in which the gradients of the reference image are static throughout the iterations, so $(\mathbf{A}^\mathbf{T}\mathbf{A})^{-1}$ is only computed once.
In the next sections we will see that the gradient discretization used by $findshift$ and the choice of interpolation method for image resampling are crucial to the precision of the GBSE method.

Algorithm 3 Iterative GBSE method.

```
1:  procedure ILK($I_1, I_2$)                          ▷ Receives a pair of images
2:      $i \leftarrow 0$; $I_2[0] \leftarrow I_2$; $w \leftarrow 0$;
3:      while $i \leq k$ do
4:          $v[i] \leftarrow findshift(I_1, I_2[i])$                ▷ (3.25)
5:          $w \leftarrow w + v[i]$                     ▷ Accumulate total shift
6:          $I_2[i+1] \leftarrow Resample(I_2, -w)$           ▷ Use the original $I_2$
7:          $i \leftarrow i + 1$
8:      end while
9:      return $w$                              ▷ Return accumulated shift
10: end procedure
```

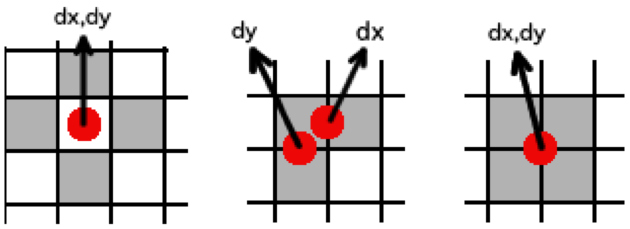


Figure 3.3 – Used pixels (gray background) for fast gradient estimation methods and their exact localization (red dots). **Left**: Centered differences. **Center**: 1D backward difference for both ∂x and ∂y. **Right**: 2D backward difference.

3.3.2 Gradient computation and image prefiltering

Gradient computation for optical flow methods has been thoroughly studied [50, 140, 154]. Since our shift estimation is focused on accuracy and the image size on SHWFS is typically small (usually smaller than 50×50 pixels), using a large kernel for computing image derivatives implies losing valuable boundary values. This constrains the kernel to be compact yet precise and robust to noise. In fact, the impact on the accuracy and the robustness to noise of the gradient computation is a key factor to the final performance of the GBSE method.

Because the method should be computationally fast, only simple schemes could be afforded. A straightforward candidate is the centered differences kernel $[1, 0, -1]$, however since the central pixel is ignored in the computation, its precision could be improved by taking contiguous pixels. For this reason, a backward difference kernel $[1, -1]$ would seem more appropriate. However, the corresponding center of this derivative differs for each component as seen from the middle image of Fig. 3.3.

A more precise gradient estimation method, illustrated on the right of Fig. 3.3, computes the derivatives by performing convolution with $\mathbf{d}_x$ and $\mathbf{d}_y$ given by

$$\mathbf{d}_x = \begin{bmatrix} 1/2 & -1/2 \\ 1/2 & -1/2 \end{bmatrix}, \qquad \mathbf{d}_y = \begin{bmatrix} 1/2 & 1/2 \\ -1/2 & -1/2 \end{bmatrix}, \tag{3.26}$$

and computes the vector $\mathbf{b}$ in (3.24) by prefiltering both I_1 and I_2 using the half pixel bilinear shift kernel

$$\mathbf{k} = \begin{bmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{bmatrix}. \tag{3.27}$$

This gradient estimation procedure called *hypomode* [142], despite being simplistic usually improves the accuracy obtained by GBSE methods using finite difference gradient estimation. This is because it slightly blurs the input images, which alleviates both aliasing and noise.

We also considered the normalized 2D Gaussian smoothing kernel $\mathbf{k} = g(x, y, \sigma_g)$ with standard deviation σ_g, and its derivatives $\mathbf{d}_x = -\frac{x}{\sigma_g^2} g(x, y, \sigma_g)$ and $\mathbf{d}_y = -\frac{y}{\sigma_g^2} g(x, y, \sigma_g)$. The parameter σ_g controls the amount of blur, thus using a small value would be less tolerant to noise, while a high value could remove textures useful for the shift estimation. In our experiments, we evaluated using $\sigma_g = \{0.3, 0.6, 1\}$ with supports of $3, 5$ and 7 pixels respectively. Finally, we included gradient estimation kernels from Simoncelli [154] with taps 3 and 5, Farid [53] with taps $3, 5$ and 7 and the Christmas kernels [35] with taps $3, 5$ and 7. We refer the reader to Section 2.2.6 for a more complete description of these approaches.

In Fig. 3.4 we show the best performing variant for each approach on each case. As seen from both Fig. 3.4(a) and Fig. 3.4(b), the most accurate gradient estimator for small shifts (≈ 0.1) proved to be the *hypomode*. Therefore it should be considered when the underlying deformations are small enough (which could happen on a closed loop system with frequent mirror corrections). However, as the maximum shift caused by the wavefront aberration gets larger, the *hypomode* resulted the worst approach while computing the 3×3 Simoncelli derivatives offered the best balance between accuracy and tolerance against noise, as seen from figures 3.4(c), 3.4(d). Finally, the larger 5×5 Simoncelli gradient estimation method achieved the best accuracies under shifts larger than half a pixel. It should be noted that displayed times do not include the computation of the derivative, reflecting only the processing time of the GBSE method.

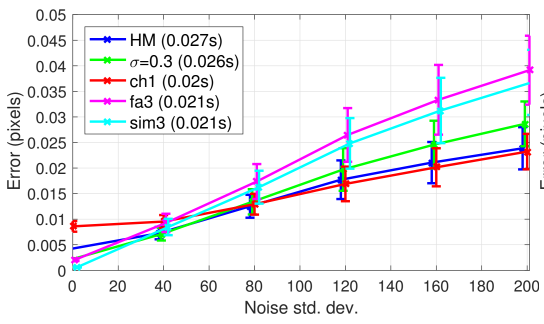
3.3.3 Interpolation methods for image resampling

In order to iterate the method, the second image needs to be shifted, as indicated in step 6 of Algorithm 3. To this end, five different interpolation methods were evaluated, namely bilinear, bicubic [82] and cubic spline [42], together with resampling using the Fourier shift theorem [20], which is evaluated with and without image periodization. Image periodization prevents the generation of ringing artifacts due to the discontinuities at the image boundaries. It amounts to resampling with DFT a mirrored version of the image which has no discontinuities when periodization is assumed (see Fig.3.5).

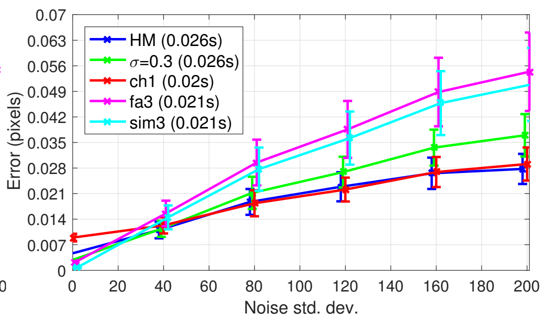
The choice of interpolation method is critical for the overall method accuracy. However precision comes at a higher processing cost. We evaluated the different interpolations by simulating a landscape with noise of $\sigma_n = 80$, displacements between sub-apertures of up to 2 pixels, and computing gradients by using Gaussian derivatives with $\sigma_g = 0.6$. For each method we computed the average error and standard deviation over 200 independent random realizations, using the GBSE method with 2 and 3 iterations. Results in Fig. 3.6 show a non-negligible performance difference between the methods. In terms of average error, both DFT-based methods achieve the best results, however no significant improvement is obtained by performing periodization, although a minimal reduction in the standard deviation is observed. The splines interpolation yields slightly less accurate results, however it should be considered if fast DFT hardware cannot be installed on board. Finally, both bicubic and bilinear interpolation perform significantly worse than the rest, although bicubic interpolation results had less variability with respect to other approaches. Lastly using three iterations does not improve considerably over using two, not justifying the increase in computational cost. However, although not shown in this experiment, this difference gets higher as the noise increases, which justifies its use.

3.3.4 Image intensities equalization

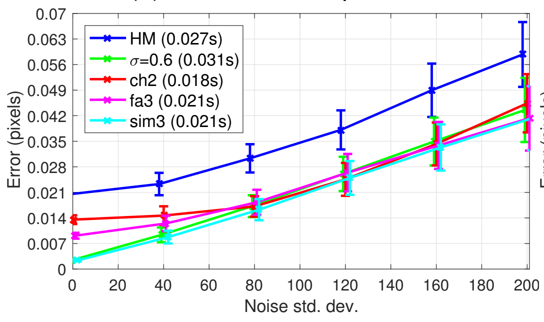
One requirement of the GBSE algorithm is to work on images with similar intensities. Since occluded sub-apertures receive less light, their intensities differ from the reference



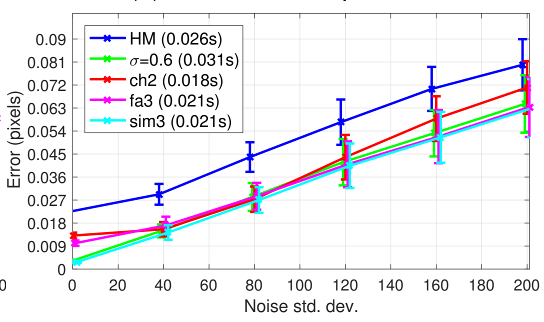
(a) Max. shift: 0.1 pixels.



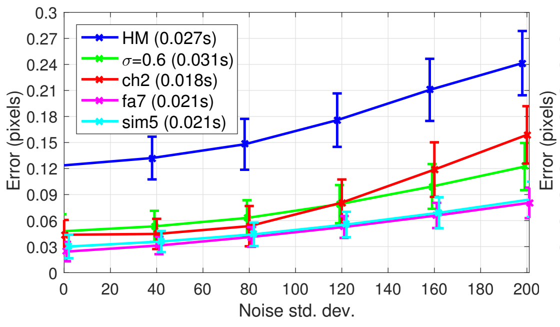
(b) Max. shift: 0.1 pixels.



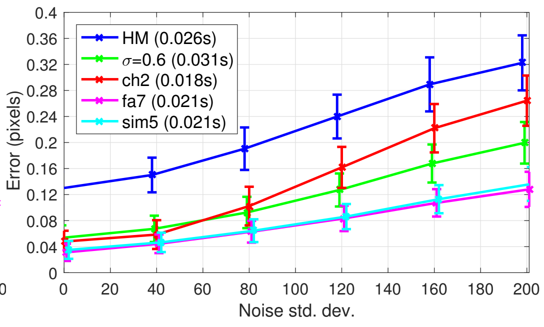
(c) Max. shift: 0.5 pixels.



(d) Max. shift: 0.5 pixels.

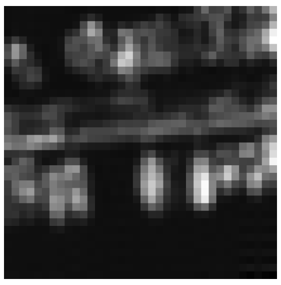


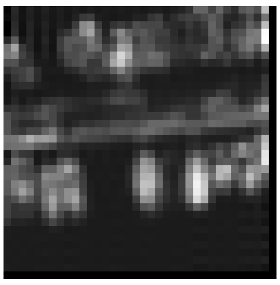
(e) Max. shift: 2 pixels.

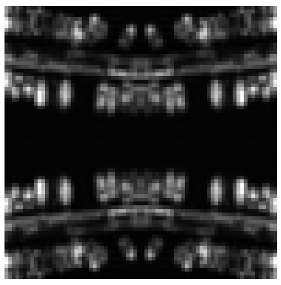


(f) Max. shift: 2 pixels.

Figure 3.4 – Gradient estimation methods comparison for three different maximum shifts (0.1, 0.5 and 2 pixels) in a SHWFS context. **Left**: Highly contrasted image (simple case). **Right**: Gradients mainly distributed on a single direction (challenging case). Noise standard deviation according to 12-bit images.


(a) Original image


(b) DFT resampling


(c) Symmetrized image

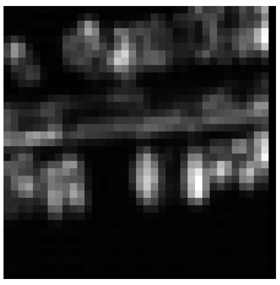

(d) Symmetrized DFT

Figure 3.5 – Example of FFT resampling with and without image symmetrization. Resampling with DFT produces ringing due to the discontinuities at the periodized boundaries. No visible ringing is observed after resampling with symmetrization.

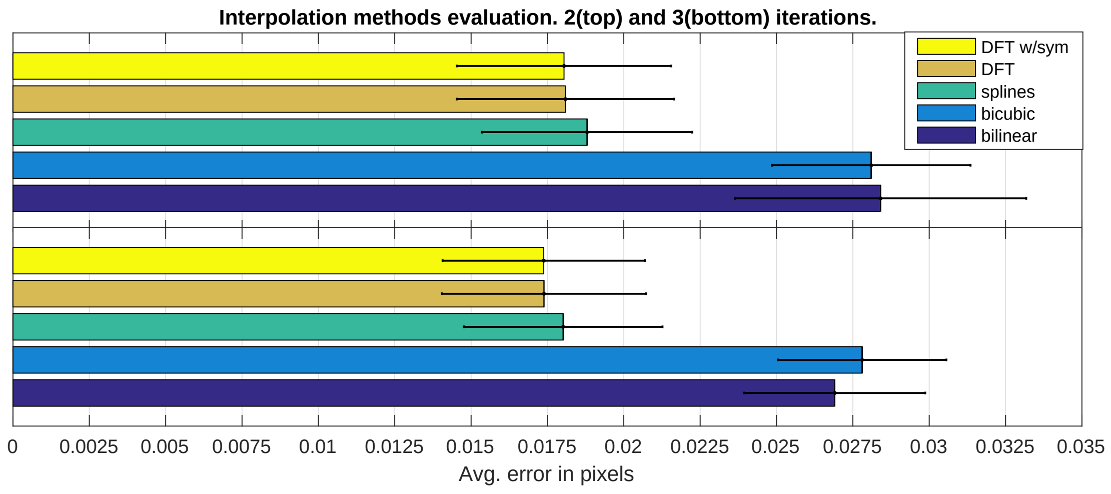


Figure 3.6 – Comparison of evaluated interpolation methods using 2 and 3 iterations and Gaussian derivatives with $\sigma_g = 0.6$. Image noise was fixed to $\sigma_n = 80$ and the maximum displacement was set to 2 pixels.

image. To this end, all subimages are equalized by normalizing their mean with the highest mean among all subapertures, computing

$$I_k(x,y) = I_k(x,y)\frac{maxMean}{mean_k}, \tag{3.28}$$

where $mean_k$ is the average value of image k and $maxMean = \max_k(mean_k)$.

Another strategy when estimating the shift between any sub-aperture and the reference image, as suggested in [21], is to first convolve each of them with the point spread function (PSF) of the other. This procedure is useful when the PSF of each sub-aperture is known beforehand, which may not be the case. However, as the authors point out, each PSF could be estimated using a star once the satellite is in orbit. Nevertheless, the convolution cost is not negligible since it has to be done for every shift estimation. Here, we chose to perform the equalization by the simple procedure explained above since it is computationally cheaper, manages to obtain excellent results and our focus is on the shift estimation method. However this other normalization procedure should be considered to improve the results even further.

3.3.5 Multiscale implementation

The GBSE algorithm can be easily adapted to work in an open loop environment where aberrations are potentially larger, leading to larger shifts between subapertures. As mentioned before, if the shift is larger than one pixel, the GBSE method fails. However, by building a pyramid representation of the input images, (3.25) can be applied on each scale to estimate the shift between images, and this estimated shift can in turn be used to resample the second image on the following level of the pyramid. If more accuracy is desired, then Algorithm 3 can be used to better estimate the shift at each scale. We compute the pyramid using an approximate dyadic Gaussian pyramid [28]. Starting from the coarse image at scale $n > 1$, the method is summarized in Algorithm 4.

3.4 Scene preselection and robustness estimation

Before evaluating the deformation of the incoming wavefront, it is crucial to determine if the current landscape, i.e., what the satellite is observing at the moment, is suitable for

Algorithm 4 Multiscale GBSE method.

```
1: procedure MSSE($I_1, I_2$)                                   ▷ Receives a pair of images
2:     $I_1^{1 \ldots n} \leftarrow BuildPyramid(I_1, n)$         ▷ Burt&Adelson's Gaussian Pyramid [28]
3:     $I_2^{1 \ldots n} \leftarrow BuildPyramid(I_2, n)$         ▷ i.e., IMPYRAMID function from Matlab
4:     $i \leftarrow n;\ \ w \leftarrow 0$                        ▷ $n$: number of scales, $w$: accumulated shift
5:     while $i > 1$ do
6:         $v(i) \leftarrow ILK(I_1^i, I_2^i)$ or $findshift(I_1^i, I_2^i)$    ▷Alg. 3 or (3.25)
7:         $w \leftarrow 2w + 2v(i)$
8:         $I_2^{i-1} \leftarrow Resample(I_2^{i-1}, -w)$
9:         $i \leftarrow i - 1$
10:    end while
11:    $v(1) = findshift(I_1^1, I_2^1)$
12:    return $w + v(1)$                                        ▷ Return accumulated shift
13: end procedure
```

performing wavefront sensing. This problem has been studied by Sidick *et al.* [152] and by Poyneer [124] who proposed several fail-safe criteria to determine if a SH image is acceptable.

In this context, we propose to perform two distinct validations to ensure an accurate wavefront correction. The best achievable accuracy, given the current landscape and the underlying noise, can be estimated by the Cramer-Rao lower bound. In the last decade, three main works have addressed the calculation of the error on the estimated shift caused by the noise. Robinson and Milanfar [139] used the Cramer-Rao lower bound (CRLB) to derive a performance limit for image registration. The CRLB imposes a lower limit on the mean squared error for any estimate of a deterministic parameter. Pham et al. [119] continued on the same idea to derive a theoretical limit for image registration, followed by a study of the bias for the gradient based 2D shift estimation. Recently, Aguerrebere *et al.* [3] performed an in depth study of performance limits within a multi-image alignment context and derived several Cramer-Rao lower bounds depending on the conditioning of the problem.

By assuming that the noise is independent, homoscedastic, Gaussian and with the same variance σ_n^2 on both images, and denoting by S the set of all pixels in the image, the Cramer-Rao bound for any unbiased 2D shift estimator is

$$var(v_x) \geq \frac{\sigma_n^2 \sum_S I_y^2}{Det}, \qquad var(v_y) \geq \frac{\sigma_n^2 \sum_S I_x^2}{Det}, \tag{3.29}$$

where $Det = \sum_S I_x^2 \sum_S I_y^2 - \left(\sum_S I_x I_y\right)^2$. We can therefore define a parameter Δ_{CRLB}, the maximum allowed error in pixels, that determines whether the current landscape is accepted for performing wavefront estimation, by verifying

$$\left(var(v_x) + var(v_y)\right)^{1/2} \leq \Delta_{CRLB}. \tag{3.30}$$

Yet, I_x, I_y represent the gradient obtained from the unknown noiseless image I. Thus this bound is only useful for a theoretical study. In practice, however, the required values can be approximated using the method of [122], detailed in Section 2.2.5, where they estimate the second moment matrix of image I_1 from noisy versions $\tilde{I}_1$ and $\tilde{I}_2$. Another possibility would be to approximate these sums by computing the expected value using the derivative definition and a noise variance estimation. For example set $I_x(i,j) = I(i+1,j) - I(i,j)$ and let $\tilde{I} = I + n$ be the observed noisy image where n has distribution $\mathcal{N}(0, \sigma_n^2)$. Since all $n(i,j)$ are independent, by the law of large numbers we have

$$\sum_S \tilde{I}_x^2 = \sum_S (I_x + n_x)^2 = \sum_S I_x^2 + \sum_S I_x n_x + \sum_S n_x^2 \simeq \sum_S I_x^2 + 2|S|\sigma_n^2.$$

It follows that we can estimate $\sum_S I_x^2$ from the noisy image $\tilde{I}$ by

$$\sum_S I_x^2 \simeq \sum_S \tilde{I}_x^2 - 2|S|\sigma_n^2. \tag{3.31}$$

The other terms of Eq. (3.29) can be computed using similar calculations.

The CRLB defined in (3.29) expresses a direct relationship between the accuracy of the estimation and the SNR ratio, measured as a ratio of the noise to the square of the gradient. Furthermore, it follows from the form of the denominator of these expressions that if there is a strong correlation between I_x and I_y, the vertical and horizontal partial derivatives, then Det will be zero or very close to zero. The formulas show that this entails a high variance for the estimation of the translation. This is the so called Aperture Problem [150]. In that case the true motion is irrecoverable.

Another interesting interpretation could be done regarding this bound. As can be noted, the Fisher information matrix for the particular problem of shift estimation is in fact the structure tensor with weights equal to one. Then, since the determinant of the structure tensor is the product of its corresponding eigenvalues, an analysis of this bound could be made through their magnitudes. If both eigenvalues of the structure tensor are small, it means the image gradients have very small magnitudes possibly suggesting the image is flat. Achieving good shift estimation accuracy is made more difficult in this case, in particular under a low SNR situation. Also, if one eigenvalue is high but the other is particularly small, it means the gradients on the image are distributed in one direction only, implying the shift estimation in the other direction will not be possible. Both of these situations are reflected by the Cramer-Rao bound as the variance of the estimators will be high. This information will be considered by the Eigenratio test detailed next.

It must be noted that the Cramer-Rao bounds are derived for an unbiased 2D shift estimation. For the case of biased estimators, the bound is even higher. Also, the hypothesis made about the noise being white Gaussian and homoscedastic does not hold in a SHWFS context used from earth-observation satellites. However, the noise model for the sensor is usually known beforehand, therefore allowing its variance to be stabilized by applying a variance stabilization transform (VST) such as the well-known Anscombe transform [6].

Finally, for the case of SHWFS, since shifts between images are independent, the limit on the registration accuracy is imposed by estimating the shift between two frames, given by Eq. (3.29). The CRLB for a certain landscape could be defined either by the maximum or the average CRLB of all subapertures. Once determined, the test in Eq. (3.30) must be verified for the landscape to be processed.

Eigenratio test: measuring distribution of the gradient through the ratio of the eigenvalues of the structure tensor As noted by [3], the CRLB is less precise where it is most needed, namely when the signal is dominated by noise. Indeed, in that case the numerator and denominator of (3.29) cannot be estimated reliably, being both the difference of equivalent terms. This is why we will be forced to recur to more robust safety computations.

We propose to rapidly discard a landscape for a shift estimation by involving the Eigenratio score of the second moment matrix $\tau = \mathbf{A}^\mathbf{T}\mathbf{A}$ of (3.25), which as previously noted, is also the structure tensor with unit weights associated to the image I_1, and is given by

$$\tau = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}, \tag{3.32}$$

where I_x and I_y are the image gradients in both horizontal and vertical directions. The eigenvalues λ_1, λ_2 of τ and their corresponding eigenvectors e_1, e_2 summarize the distribution of the image gradient $\nabla I = (I_x, I_y)$. Namely, if $\lambda_1 > \lambda_2$, then e_1 (or $-e_1$) is the direction that is maximally aligned with the gradient within the image. More generally, the value of λ_k , for $k \in \{1, 2\}$, is the average of the square of the directional derivative of I along e_k. The relative discrepancy between the two eigenvalues is an indicator of the degree of anisotropy of the gradient in the image, i.e. how strongly is it biased towards a particular direction (and its opposite). Then by calculating the ratio λ_2/λ_1, we obtain a number between 0 and 1, characterizing the dominance of a particular direction for the gradients of the image. Empirically it was verified that a ratio $\lambda_2/\lambda_1 < 0.2$ effectively degrades the shift estimation task, and this metric could be used to discard badly conditioned situations.

To compute this ratio, since $\tau = \mathbf{A}^T\mathbf{A}$ is positive definite and symmetric, then both of its eigenvalues λ_1 and λ_2 are real and non-negative and its computation is straightforward. Based on the fact that the trace t of matrix τ is the sum of their eigenvalues and that its determinant Det is their product, we define

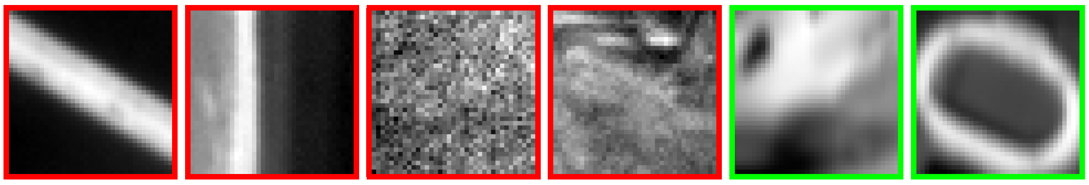
$$\lambda_1 = \frac{t}{2} + \sqrt{\frac{t^2}{4} - Det}, \qquad \lambda_2 = \frac{t}{2} - \sqrt{\frac{t^2}{4} - Det}. \tag{3.33}$$

where $t = \sum I_x^2 + \sum I_y^2$ and $Det = (\sum I_x^2)(\sum I_y^2) - (\sum I_x I_y)^2$.

3.5 Results

3.5.1 Scene pre-selection evaluation

Figures 3.7(a) and 3.7(b) display two examples of SH landscapes that do not validate the EigenRatio test. In these images there is a strong dominant gradient direction that complicates the shift estimation. The aperture problem is visually obvious in both cases. Figures 3.7(c) and 3.7(d) present a high CRLB. The first one corresponds to a project where the images are composed by only noise, while the second example has a higher SNR, however the noise is still considerable. Finally, two examples of valid landscapes are shown in figures 3.7(e) and 3.7(f) which were validated by both proposed tests.



(a) $0.021, 0.041$ (b) $0.057, 0.042$ (c) $0.115, 0.343$ (d) $0.063, 0.491$ (e) $0.004, 0.437$ (f) $0.002, 0.616$

Figure 3.7 – CRLB and eigenratios respectively for six examples of landscapes. Green represents valid and red implies failure.

3.5.2 Comparison with state-of-the-art methods

The CNES simulated database

CNES (French space agency) provided realistic simulations of several SHWFS landscapes obtained from earth-observing satellites with their corresponding ground truth. The provided images were 37×37 pixels from a 12×12 grid following the occlusion schema of Fig. 3.2(a). The simulated wavefront aberrations translated into displacements no larger than half a pixel. For each provided landscape, 3 different SNR settings were available. Fig. 3.8 displays some example landscapes.

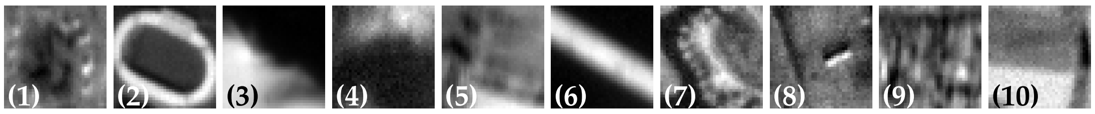


Figure 3.8 – Examples of landscapes from the CNES database.

Every state-of-the-art algorithm and our proposed method using different interpolation/gradient estimation approaches was tested against the whole dataset. In Table 3.1 the errors are displayed for the best performing methods of each computational cost range.

The errors obtained for figures 3.8.1 and 3.8.2 are close to the theoretical limits predicted by their CRLB. On the other side, the error obtained with Fig. 3.8.3 and its CRLB implies that a strong improvement is possible. Line 4, corresponding to Fig. 3.8.4, is an example where every method fails to achieve accurate results, possibly due to the noise and the lack of texture, which is revealed by a lower bound considerably higher than for most other cases. Fig. 3.8.5 shows an example of a landscape where the Poyneer method [124] as well as APC [153] are clearly improved by the proposed approach. Fig. 3.8.6 proves that the aperture problem hinders most shift estimation methods, as evidenced by their resulting errors. The APC algorithm particularly failed under the landscape present in Fig. 3.8.7 although the reason is not evident. Finally, line 9 (Fig. 3.8.9) shows a case where the SDF-2QI method was outperformed by all variants of the proposed approach. It should be remarked that the non-iterative variant of the GBSE method using the 3×3 Simoncelli kernel achieved almost similar results than more expensive iterative variants (requiring up to 10 times less time to compute), which makes it the most suitable choice for this dataset.

Table 3.1 – CRLB, EigenRatio (ER) and errors for selected landscapes of the CNES database. [a]

	CRLB	ER	SDF2QI [95]	MICHAU [109]	POYNEER [124]	APC [153]	LS-1-IlGh	LS-1-IlGsim3	LS-2-IcGh	LS-3-IsGh
(1)	0.014	0.66	0.019	0.121	0.026	0.054	0.030	0.018	0.017	**0.016**
(2)	0.006	0.44	0.044	0.106	0.038	0.158	0.014	0.009	0.008	**0.008**
(3)	0.006	0.19	0.043	0.096	0.105	0.092	**0.035**	0.036	0.036	0.036
(4)	0.041	0.59	0.124	0.113	0.090	0.794	0.070	0.070	**0.064**	0.069
(5)	0.011	0.77	0.054	0.113	0.069	0.070	0.026	0.016	**0.016**	**0.016**
(6)	0.030	0.01	0.072	0.099	0.083	0.081	0.053	0.043	0.043	**0.040**
(7)	0.019	0.69	0.025	0.111	0.048	0.134	0.034	0.022	**0.021**	0.021
(8)	0.026	0.63	0.052	0.114	0.039	0.083	0.048	0.033	0.031	**0.031**
(9)	0.010	0.34	0.026	0.113	0.029	0.042	0.028	0.016	0.015	**0.014**
(10)	0.027	0.61	0.085	0.112	0.051	0.645	0.046	0.040	**0.038**	0.043
Avg. Err.	0.0289	132	0.231	0.111	0.491	0.478	0.047	0.041	**0.039**	0.039
Valid avg.	0.0106	64	0.029	0.110	0.048	0.057	0.027	0.019	0.017	**0.017**
Avg. Time (s)			0.309	0.015	0.055	0.139	0.029	0.022	0.107	0.186

[a]Rows in green represent landscapes considered valid (CRLB < 0.02 and ER > 0.2), while invalid landscapes are shown in red. Bold indicates lowest error. Averages correspond to the whole dataset, and its ER column displays the amount of landscapes processed. The Valid row represents the averages over all valid landscapes. The first column links each row with the landscapes shown in Fig. 3.8. LS-1-IlGh, LS-2-IcGh and LS-3-IsGh represent Alg. 3 using the *hypomode* derivative with 1 , 2 (bicubic interpolation) and 3 (spline interpolation) iterations respectively. The column LS-1-IlGsim3 stands for 1 iteration and the 3×3 Simoncelli kernel for gradient estimation [154].

Simulated experiments

Experimental Setup. We used a simulator to evaluate the performance of the proposed method. Given a 12-bit input image, a set of images was generated matching a SHWFS configuration provided by CNES. A 12×12 lenslet grid occluded as shown in Fig. 3.2(a) was assumed. For each lenslet a ground truth shift was randomly generated. Since we were simulating in a closed loop environment, the maximum displacement was kept to half a pixel. Each sub-aperture image was then obtained by DFT resampling the input image, extracting a 37×37 pixels subimage, multiplying by the occlusion factor, and lastly adding noise.

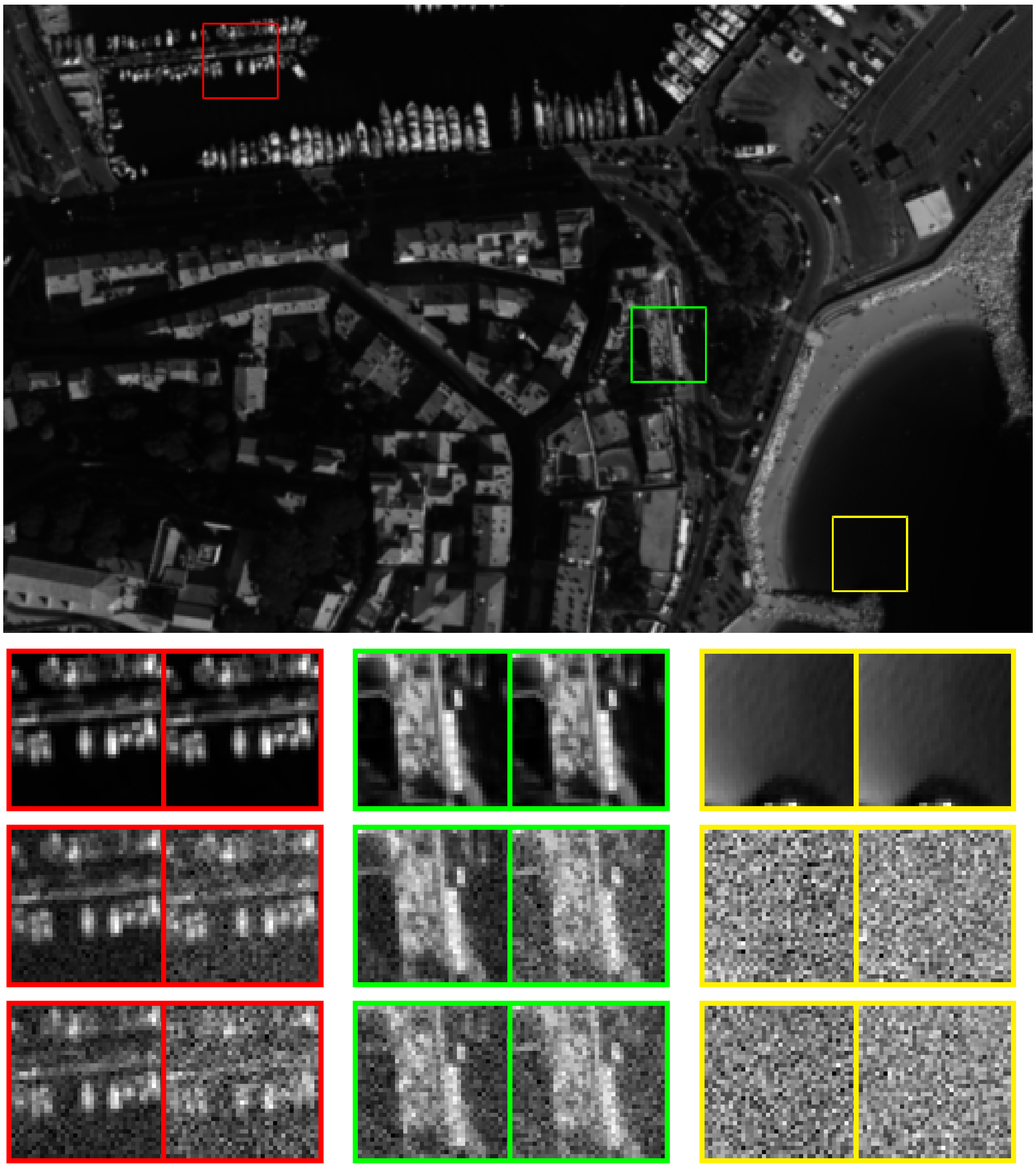


Figure 3.9 – **Top**: Input image used for the simulations. **Bottom**: For each level of noise $\sigma_n = \{1, 100, 200\}$ (vertically separated), two different sub-apertures are shown: no occlusion and 57% occluded (horizontally separated). The dynamic ranges were stretched for viewing purposes.

The error for each method was computed as the mean error for all valid sub-apertures. A sub-aperture was considered invalid and therefore discarded if it was occluded on more than 60% of its surface. The error for each sub-aperture was the Euclidean distance between the measured and the ground truth shift. To evaluate the robustness to image content, we used three subimages: a highly contrasted one that should not present any difficulty for shift estimation, a slightly more challenging one with its gradient mainly distributed on a single direction, and one from the sea with almost no signal (there are just a few pixels of land on the bottom). Different amounts of additive white Gaussian noise were simulated (with standard deviation $\sigma_n \in [1, ..., 200]$), and all methods were evalu-

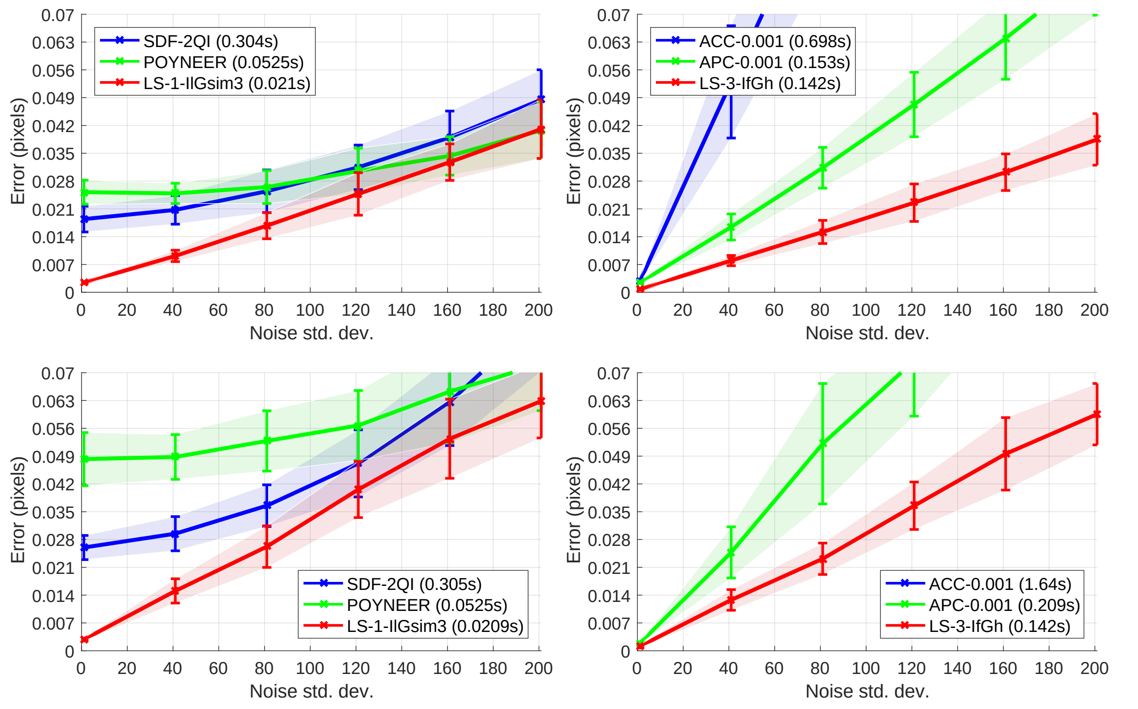


Figure 3.10 – Performance for non-iterative and iterative methods. **Top**: First test image. **Bottom**: Second test image. **Left**: Non-iterative methods. **Right**: Iterative methods.

ated on 100 noise realizations. Fig. 3.9 shows the subimages affected by three noise levels. Signal dependent noise was handled by application of a variance stabilization transform. Finally, the processing time of each method was measured using a non-optimized Matlab implementation on an Intel Xeon E5-2650 CPU.

We tested several variants of the proposed GBSE method. Five interpolation methods for resampling (see Section 3.3.3), all gradient estimation methods mentioned in Section 3.3.2, and using up to three iterations per shift estimation. Also, variants using the multiscale approach were evaluated (Section 3.3.5) up to 3 scales, where for each scale, an iterative GBSE algorithm was used with up to 3 iterations. From these methods we retained the best performing non-iterative method ($k = 1$ and no resampling) which uses the 3×3 kernel from the Simoncelli gradient computation method [154], and the best with 3 iterations which uses Fourier resampling and the *hypomode* gradient estimation. We compared them with the best state-of-the-art algorithms, namely, ACC [151], APC [153], the Poyneer method [124] and Löfdahl's SDF-2QI algorithm [95]. All other presented methods were also evaluated but excluded due to lower accuracies.

Results in a closed loop system. We compared iterative and non-iterative algorithms separately. In each case, the most accurate variant of the proposed solution is compared against the two most accurate methods of the state-of-the-art. The average errors and the standard deviations for non-iterating methods on the first image type are shown in Fig. 3.10 (top left), together with the processing time. The non-iterative GBSE version using the 3×3 Simoncelli derivatives outperforms both Poyneer and SDF-2QI (which perform similarly in this image). GBSE is also more stable (less variability) and faster (it took half the time of the Poyneer method). In Fig. 3.10 (top right) both iterative methods ACC and APC were compared with GBSE using 3 iterations, Fourier resampling and the *hypomode* derivative. Again the proposed method proved to be the most accurate and stable. Even more, this method was the best also compared to non-iterated approaches. Notice the high impact of noise on the ACC method. It is more precise than both non-iterating methods for low noise values ($\sigma \leq 20$). However, it diverges for stronger noise. This problem is well-known for phase-correlation methods [64, 124].

The results for the second image type (bottom of Fig. 3.10) are similar to the first one, although the average errors are higher. While the SDF-2QI performs slightly worse, the Poyneer method is considerably worse. This is due to discrepancies at the image boundaries resulting from the periodicity assumption of the periodic convolution used by Poyneer. It also explains why both ACC and APC methods behave so poorly on this image. This also implies that computing the shift in the spatial domain is usually more stable than doing so in the frequency domain, unless some windowing is performed, which is prohibitive on such small images where the objects are sometimes close to the image boundaries. Yet again, the proposed methods are more precise, stable and with less variability than the state-of-the-art. The third image type did not pass the verification step and all the methods failed, so its results are omitted.

Remark about the interpolation of the correlation function. It is worth noting in Fig. 3.10 that the precision of both correlation based methods, SDF-2QI and Poyneer, flat-out for small levels of noise. A possible explanation for this behavior is that these methods don't account for the alias introduced during the computation of the correlation. Szeliski and Scharstein [161] and Sabater et al. [146] have shown that in order to properly compute and interpolate the correlation cost between two images (or equivalently the SDF), the images and the cost function must be oversampled by at least a factor two. In a nutshell, this is because the squared difference of two band-limited images has twice the bandwidth of the input images. In that case, perfect band-limited interpolation (sinc filter) could be used to interpolate the cost function from the samples taken at half-pixel disparity steps, allowing to accurately compute the disparity with arbitrarily subpixel precision. Not accounting for this aliasing phenomenon affects the precision of the interpolated maximum of correlation, which could in part explain the poor performance of the SDF-2QI and Poyneer methods under low noise levels.

Results under larger displacements. The same experiment was performed by simulating wavefront aberrations having displacements of up to 4 pixels, to evaluate robustness against high frequency aberrations. In Fig. 3.11 the average error together with the standard deviations are shown for the most representative methods. Due to the larger underlying displacements, multiscale approaches achieve better results as the noise increases, reproducing the results of [131]. In fact, a multiscale approach becomes mandatory if fewer than two iterations are performed, and the best performing method found used 3 scales (where a single iteration was used in the coarsest scale, two iterations with spline interpolation in the second scale and 3 iterations with DFT interpolation in the final finer scale) and the *hypomode* gradient estimation. Although this method proved to be the most accurate in our tests, it also requires more computational resources as evidenced by its running time. However, using the iterative single-scale approach with 3 iterations, DFT interpolation and the 3×3 gradient estimation kernel proposed by Farid [53] showed comparable results while requiring less processing time. Nevertheless, the method proposed by Poyneer should be considered when lower processing times are required. Indeed, as expected, non-iterated versions of the GBSE algorithm failed when estimating the shift with large underlying displacements. Let us add the caveat that when measuring the aberration using the second slightly challenging landscape, the Poyneer method obtained inaccurate results even under high SNR conditions.

3.6 Concluding Remarks

A new method for accurate sub-pixel shift-estimation, based on an iterative global optical flow algorithm, was proposed in the context of a SHWFS on extended scenes for aberration correction on-board earth-observation satellites. Using a telescope simulator developed by CNES, we showed that the proposed algorithm is more accurate, stable,

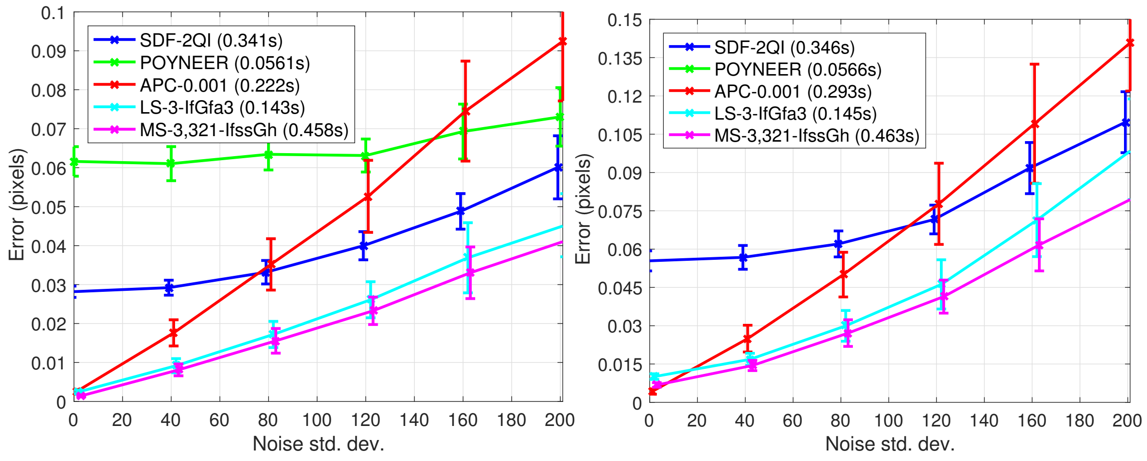


Figure 3.11 – Average error and standard deviations of selected methods and two variants of the GBSE approach when simulating wavefront aberrations yielding displacements of up to 4 pixels. **Left**: First test image. **Right**: Second test image.

robust to noise, and with lower variability than the current state-of-the-art, permitting a more precise wavefront estimation. This opens the way to cheaper high resolution Earth observation satellites, by permitting on-board real time active optics on large but far lighter mirrors.

Chapter 4

Stab-Active: stabilizing on board image accumulation

Image registration can play a key role to increase the SNR of Earth satellite images by fusing many successive noisy thin image stripes in real time, before transmitting to the ground the final fused image. Yet these stripes cannot be fused directly by addition because their position is altered by microvibrations and other geometric perturbations. These must be compensated using limited onboard computational resources with high subpixel accuracy and in real time. In this chapter we study the fundamental performance limits for this problem and propose a real time solution that nonetheless gets close to the theoretical limits. We introduce a scheme using temporal convolution together with online noise estimation, gradient-based shift estimation and a non-conventional multiframe method for measuring global displacement. Finally, we compare our results with the theoretical bounds and other state-of-the-art methods. The results are conclusive on the fronts of accuracy and complexity.

4.1 Introduction

Push broom imagers placed on satellites are used to perform Earth observation at high resolution [19, 59]. Because they are close to the Earth ($\approx$ 800 km), they move fast and the image acquisition time is very brief. The solution to avoid motion blur is to use Time Delay Integration (TDI) on the satellite's imaging sensor, which, by synchronizing the pixels with the motion of the camera or the object, is able to increase the effective exposure time. This sensor works by shifting the partial measurements of its multiple rows to their adjacent rows synchronously with the motion of the image across the array of elements. This synchronous accumulation yields an SNR unobtainable using conventional CCD arrays or single-line-scan devices. However, it places stability constraints on the imaging device. To relax these constraints limiting the accumulation time, we evaluated if performing on board registration and accumulation was possible.

Envisaging the use of a CMOS TDI allows to correct the accumulation procedure by resampling the incoming signal. This permits to avoid motion blur while accumulating many more frames thus obtaining much higher SNRs. Yet this raises new technical issues. A prerequisite is that the perturbations of the line of sight must be estimated very accurately. To this aim, little low resolution CMOS sensors set along this line of sight could sense the same landscape underneath of the TDI's detection line, possibly under slight perturbations and with a low signal to noise ratio (SNR) [107]. The problem is then to perform subpixel image registration, up to the noise limit, of a set of consecutive low SNR images of the same landscape, known as a "TDI ROW" or simply "line". This becomes a non-trivial task due to different external conditions such as pointing errors and satellite vibration. Luckily at the studied time scales ($>$ 500Hz), the satellite micro-vibrations can be considered a linear shift over the image sequence, requiring only to compute a global translation estimate for the whole line (see figures 4.1 and 4.2). Finally, due to limited hardware resources, the desired algorithm must be low demanding in both memory and processing speed.

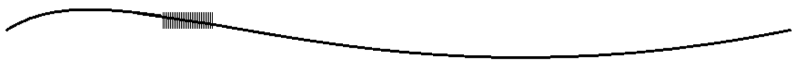


Figure 4.1 – In the studied time scales, the satellite micro-vibrations are locally linear.

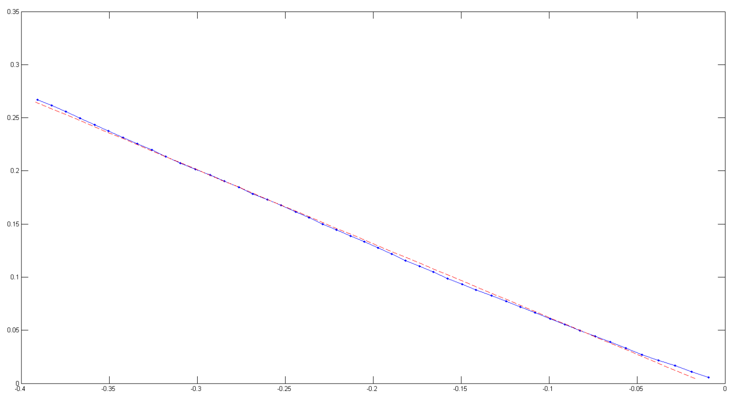


Figure 4.2 – Accumulated displacements estimated using algorithm 23 for a random TDI row in the provided dataset. It can be clearly seen that accumulated displacements follow a linear law.

Based on this task, several interesting questions arise. Is it possible to predict the theoretical optimal performance of the registration method to fix a theoretical bound? Since this will imply an accurate noise estimate, can the noise be estimated at low cost on board? Can the smoothness of the trajectory be used to make an accurate and very low complexity registration? How can the results be validated? All these questions will be addressed in this study.

This chapter is then organized as follows. We begin in Section 4.2 by briefly discussing which shift estimation approaches were more suitable for our particular problem. We continue by giving the optimal lower bounds for the current multi-image registration problem in Section 4.3, together with a study on the bias of the GBSE method. Section 4.4 describes in detail the proposed algorithms. Finally, the methods are evaluated in Section 4.5 and conclusions are given afterwards.

4.2 Method analysis

In summary, the Stab Active problem reduces to performing accurate multi-image subpixel shift estimation between N images, where the shift between consecutive images could be assumed constant. What is more, this constant shift is lower than a tenth of a pixel and, due to extremely low exposure times, images have low signal to noise ratios. The requirement of doing this in real-time on-board using low-power satellite-grade hardware makes this an extremely challenging task where most methods have to be discarded beforehand.

Subpixel shift estimation between two images is an old subject for which several very good reviews exist over the years [168], [191], and particularly, in Chapter 2 of this manuscript. An iterative method to achieve subpixel accuracy was already proposed in 1981 [97]. A classic alternative involves FFT-based phase correlation [132], which has been improved to get subpixel accuracy in [57], [162] and [69]. The optimal bounds for performing registration between two images also have a long history going back at least to 1983 [46]. We shall use their latest developments [139], [119], [146] which will be extended here to a multi-image uniform translation. It so happens that the optimal registration bounds for an image sequence are much more accurate than with only two images, and the challenge is to reach these bounds.

Based on the review of shift estimation methods presented in Chapter 2 of this manuscript, and due to the limited time constraints, only two types of methods were considered, namely, differential (i.e. gradient-based) and phase-correlation methods. A priori, GBSE method seems the best suited approach due to its low computational cost and its accurate subpixel results. Indeed, phase correlation methods could improve over GBSE when used on images that are severely distorted, in either geometry or intensity [168], however this is not the current case. Other reasons that allowed to discard phase correlation approaches were:

- The shift between images is a simple linear shift, rather than a circular shift, which implies performing image apodization thus discarding potentially useful textures. This is prohibitive on small image sizes.
- The phase-correlation method does not perform well in presence of an aperture problem, as will be shown in our experimental section.
- For periodic images (that have repeating objects, a potential situation that could happen in our current problem), phase correlation may also yield ambiguous results with several correlation peaks in the resulting output.
- Last but not least, the computational time saved using the DFT is more significant if the images to be registered are large, which is not the case of our particular problem.

Nevertheless, we compare our results with those of a phase-correlation-based method [69] in our experiments thus proving our claims.

4.3 Influence of Noise on the Registration

4.3.1 Noise estimation

Since the translation between consecutive frames is assumed to be small with respect to the pixel size (less than a tenth of a pixel), by subtracting both frames we can easily estimate the noise. Setting $I_t^* = I_2^* - I_1^*$ where I_2^* and I_1^* are contaminated by additive white Gaussian noise with standard deviation σ, implies that

$$\frac{1}{|S|}\sum_{x,y}(I_t^*(x,y))^2 \approx 2\sigma^2, \tag{4.1}$$

where $|S|$ is the image size. Thus, in order to make the estimation more precise, this calculation can be done for each pair of consecutive frames in the line and its average will be a better estimation for the variance σ^2. This toy example can become reality by applying a variance stabilizing transform (VST), like for example the Anscombe transform [6], which transforms a Poisson noise into an approximately white Gaussian noise. A possible implementation of the VST is discussed in Appendix B. We verified that when running the algorithms with no Anscombe VST, there is a small decrease in performance. Also, applying the VST to the images has the merit of aligning the numerical calculations with the theoretical calculations, which are usually made for white noise. Nevertheless, the noise in the dataset is not white Gaussian. It is the classic sum of a Poisson noise and of a thermal approximately white Gaussian noise. Nevertheless, the noise remains independent at each pixel. Thus, we estimate the noise $\hat{\sigma}$ by

$$\hat{\sigma}^2 := \frac{1}{N-1}\sum_{i=1}^{N-1}\left[\frac{1}{2|S|}\sum_{x,y}(I^*_{i+1}(x,y) - I^*_i(x,y))^2\right] \tag{4.2}$$

where N is the number of images to register, $|S|$ is the number of pixels of each image and I^*_i represents each noisy VST transformed image from I_i. We shall see that this estimate is crucial in the whole study of the influence of noise on the accuracy of motion estimation.

4.3.2 Predicting the error on the estimated shift caused by the noise

In the last decade, four main works have handled the calculation of the error on the estimated shift caused by the noise. Robinson and Milanfar [139] used the Cramer-Rao bound (CRB) to derive a performance limit for image registration. The CRB bound imposes a lower limit on the mean squared error for any estimate of a deterministic parameter. Then, Pham et al. [119] continued on the same idea to derive a theoretical limit for image registration, followed by a study of the bias for the 2D GBSE method of Lucas and Kanade [97]. Sabater et al. [146] have studied how accurate the disparity computed from a rectified stereo pair of images with small baseline can be, and a formula for the variance of the disparity error caused by the noise was proven. Contrarily to CRB bounds, the error formula of this last paper is not a bound but a real evaluation of the variance. It can be applied to our case if the direction of the velocity is approximately known. Recently, Aguerrebere *et al.* [3] performed an in depth study of performance limits within a multi-image alignment context and derived several Cramer-Rao lower bounds depending on the conditioning of the problem. Interestingly enough, assuming for example that the block matching is given in the direction of x, the obtained formal error estimate for the variance is just the double of the Cramer Rao lower bound given in (4.3). Thus, the Cramer Rao lower bounds below are tight in the sense that they give the correct order of magnitude for the error caused by the noise. Furthermore, if the motion direction is approximately known (as it actually is), then (4.3) gives the correct estimate of the error caused by the noise.

In this section we assume that after the VST, the noise is additive, white and Gaussian. Two dimensional shift estimation searches for a translational vector $\mathbf{v} = [v_x, v_y]^T$ between two images. Then, given σ^2 the variance of the noise, and denoting by S the set of all pixels in the image, the Cramer-Rao bound [119] for any unbiased 2D shift estimator is

$$var(v_x) \geq \mathbf{F}^{-1}_{11} = \frac{\sigma^2\sum_S I_y^2}{Det(\mathbf{F})}, \tag{4.3}$$

$$var(v_y) \geq \mathbf{F}^{-1}_{22} = \frac{\sigma^2\sum_S I_x^2}{Det(\mathbf{F})}, \tag{4.4}$$

where $I_x = \frac{\partial I}{\partial x}$, $I_y = \frac{\partial I}{\partial y}$, $\mathbf{F}$ is called the Fisher information matrix (FIM):

$$\mathbf{F}(\mathbf{v}) = \frac{1}{\sigma^2}\begin{bmatrix} \sum_S I_x^2 & \sum_S I_xI_y \\ \sum_S I_xI_y & \sum_S I_y^2 \end{bmatrix} \tag{4.5}$$

and $Det(\mathbf{F}) = \sum_S I_x^2 \sum_S I_y^2 - (\sum_S I_xI_y)^2$.

These formulas express a direct relationship between the accuracy of the estimation and the SNR ratio, measured as a ratio of the noise to the square of the gradient. Furthermore, it follows from the form of the denominator of these expressions that if there is a strong correlation between I_x and I_y, the vertical and horizontal partial derivatives, then

$Det(\mathbf{F})$ will be zero or very close to zero. The formulas show that this entails a high variance for the estimation of the translation. This is the so called Aperture Problem [150]. In that case the true motion is irrecoverable. The Cramer Rao lower bounds are therefore critical to the detection of the aperture problem.

Based on this, we propose two complementary tests for performing robust shift estimation. The first one, to be executed on-board thanks to its low computational cost, checks that the SNR is sufficient. The second one, more computationally expensive, is obviously given by the Cramer-Rao bounds (4.3) and (4.4). It has to be noted that this bound cannot be directly computed since it uses information from noiseless images, which we do not have. To estimate this bound we follow the methodology mentioned in section 3.4 of this manuscript.

Note that the Cramer-Rao bounds are derived for an unbiased 2D shift estimation, however the gradient-based method presented in section 2.2 is indeed biased. An analysis of this bias is given next, which resembles the study performed by Pham *et al* [119].

Analysis of bias for the GBSE method

Following Eq. (2.2), the GBSE method minimizes the Minimum Squared Error (MSE) to obtain both translations v_x, v_y resulting in

$$\begin{bmatrix} \sum_S I_x^2 & \sum_S I_x I_y \\ \sum_S I_x I_y & \sum_S I_y^2 \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum_S I_x I_t \\ \sum_S I_y I_t \end{bmatrix}, \tag{4.6}$$

which gives the solution

$$v_x = \frac{1}{Det}\left(\sum_S I_x I_t \sum_S I_y^2 - \sum_S I_y I_t \sum_S I_x I_y\right), \tag{4.7}$$

$$v_y = \frac{1}{Det}\left(\sum_S I_y I_t \sum_S I_x^2 - \sum_S I_x I_t \sum_S I_x I_y\right), \tag{4.8}$$

where $Det = \sum_S I_x^2 \sum_S I_y^2 - (\sum_S I_x I_y)^2$. This method, as shown in (3.23), is derived from a Taylor approximation by truncating the Taylor series after the first order derivative. This approximation is accurate when the second and the higher order derivatives are small or when the displacement is much smaller than 1 pixel. Also, equation (3.23) completely ignores the noise in the input images. As a result, there is a systematic bias that depends on the image content and the displacement itself. In fact, the bias is a combined effect of multiple modeling errors such as truncation of Taylor series and noise intensity. The bias due to the Taylor series truncation can be easily derived by putting the correct formula $I_t = v_x I_x + v_y I_y + \varepsilon$ in the equations above. It is enough to replace I_t by $I_t - \varepsilon$ in them. Then the additional corrective term due to higher orders of derivatives of I_1 is

$$bias_x^T = \frac{1}{Det}\left(\sum_S \varepsilon I_x \sum_S I_y^2 - \sum_S \varepsilon I_y \sum_S I_x I_y\right), \tag{4.9}$$

$$bias_y^T = \frac{1}{Det}\left(\sum_S \varepsilon I_y \sum_S I_x^2 - \sum_S \varepsilon I_x \sum_S I_x I_y\right). \tag{4.10}$$

Developments made in [119] prove that the even-order terms in the expression of $\sum_S \varepsilon I_x$ and $\sum_S \varepsilon I_y$ vanish due to Parseval's theorem:

$$\sum_S \varepsilon I_x = \frac{1}{6} v_x^3 \sum_S I_{xxx} I_x + \frac{1}{2} v_x^2 v_y \sum_S I_{xxy} I_x + \frac{1}{2} v_x v_y^2 \sum_S I_{xyy} I_x + \frac{1}{6} v_y^3 \sum_S I_{yyy} I_x + \ldots \tag{4.11}$$

This bias is cubic on the displacement function. Since we are dealing with a sub-pixel displacement between frames, of the order for example of $\frac{1}{10}$ of a pixel, this bias would be of the order of $\frac{1}{1000}$. This justifies the truncation of the Taylor series, since these values are negligible in comparison with the bias caused by noise, as will be confirmed experimentally.

A direct calculation omitted here but easy to reproduce, proves that the bias due to noise in the horizontal direction (the vertical direction is similar and thus omitted), for the GBSE method, is the following:

$$
\begin{aligned}
bias_x^N &= v_x^* - v_x && (4.12)\\
&= \frac{\sum_S I_x I_t \left(\sum_S I_y^2 + \sum_S n_y^2\right) - \sum_S I_y I_t \sum_S I_x I_y}{\left(\sum_S I_x^2 + \sum_S n_x^2\right)\left(\sum_S I_y^2 + \sum_S n_y^2\right) - \left(\sum_S I_x I_y\right)^2} && (4.13)\\
&= \frac{\sum_S I_x I_t \sum_S n_y^2}{Det + N} + v_x \frac{N}{Det + N}, && (4.14)
\end{aligned}
$$

where v_x^* is the estimated shift computed from (4.7) using derivatives of the noisy image $I_1^* = I_1 + n_1$, n_x and n_y are derivatives of the noise n_1 in horizontal and vertical dimensions respectively, and N is the over-estimation of the Hessian matrix's determinant if the gradient information of the noisy image is used:

$$
N = \sum_S n_x^2 \sum_S I_y^2 + \sum_S n_y^2 I_x^2 + \sum_S n_x^2 \sum_S n_y^2. \tag{4.15}
$$

4.3.3 Cramer-Rao lower bounds for image registration of several images with uniform translation

In equations (4.3) and (4.4), the given Cramer-Rao bounds are based on performing registration using a pair of images. However, in our particular problem, we deal with a set of $N = 64$ images, so the bounds should be modified by taking this into account. For this reason more precise Cramer-Rao lower bounds had to be derived. Due to the proper nature of the problem, the assumption of a global uniform translation (v_x, v_y) per line was considered. This assumption was backed up by the empirical results obtained in Fig. 4.2 where the accumulated translation, calculated for each line in the provided dataset, showed an almost perfect linear shape.

Then, assuming that $I_0, \ldots, I_{N-1}$ are uniformly translated sample images of a noiseless image I, each being corrupted by additive white Gaussian noise $n_0, \ldots, n_{N-1}$ with variance σ^2, then

$$
\begin{aligned}
I_0(x,y) &= I(x,y) + n_0(x,y),\\
I_1(x,y) &= I(x - v_x, y - v_y) + n_1(x,y),\\
I_2(x,y) &= I(x - 2v_x, y - 2v_y) + n_2(x,y),\\
&\vdots\\
I_{N-1}(x,y) &= I(x - (N-1)v_x, y - (N-1)v_y) + n_{N-1}(x,y).
\end{aligned}
$$

Let $\hat{v} = (\hat{v_x}, \hat{v_y})$ be an estimator for the global shift (v_x, v_y), then the probability of the unknown scene I given $\hat{v}$ is:

$$
Pr(I|\hat{v}) = \prod_{i=0}^{N-1} \prod_S \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(I_i(x,y) - I(x - iv_x, y - iv_y)\right)^2}{2\sigma^2}\right). \tag{4.16}
$$

Then, $logPr(I|\hat{v})$ becomes

$$logPr(I|\hat{v}) = -\frac{1}{2\sigma^2}\sum_{i=0}^{N-1}\sum_{S}\left(I_i(x,y) - I(x-iv_x, y-iv_y)\right)^2 + const. \tag{4.17}$$

Then, taking the partial derivatives with respect to v_x and v_y, we get

$$\frac{\partial}{\partial v_x}logPr(I,\hat{v}) = \frac{1}{\sigma^2}\sum_{i=0}^{N-1}\sum_{S}\left[I_i(x,y) - I(x-iv_x, y-iv_y)\right]\frac{\partial}{\partial v_x}I(x-iv_x, y-iv_y), \tag{4.18}$$

$$\frac{\partial}{\partial v_y}logPr(I,\hat{v}) = \frac{1}{\sigma^2}\sum_{i=0}^{N-1}\sum_{S}\left[I_i(x,y) - I(x-iv_x, y-iv_y)\right]\frac{\partial}{\partial v_y}I(x-iv_x, y-iv_y), \tag{4.19}$$

Finally, taking the second derivatives, we get

$$\frac{\partial^2}{\partial v_x^2}logPr(I,\hat{v}) = \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left[n_i(x,y)\frac{\partial^2}{\partial v_x^2}I(x-iv_x, y-iv_y) - \left(\frac{\partial}{\partial v_x}I(x-iv_x, y-iv_y)\right)^2\right], \tag{4.20}$$

$$\frac{\partial^2}{\partial v_y^2}logPr(I,\hat{v}) = \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left[n_i(x,y)\frac{\partial^2}{\partial v_y^2}I(x-iv_x, y-iv_y) - \left(\frac{\partial}{\partial v_y}I(x-iv_x, y-iv_y)\right)^2\right], \tag{4.21}$$

$$\begin{aligned}\frac{\partial^2}{\partial v_x v_y}logPr(I,\hat{v}) &= \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left[n_i(x,y)\frac{\partial^2}{\partial v_x v_y}I(x-iv_x, y-iv_y)\right] \\ &\quad - \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left[\frac{\partial}{\partial v_x}I(x-iv_x, y-iv_y)\frac{\partial}{\partial v_y}I(x-iv_x, y-iv_y)\right].\end{aligned} \tag{4.22}$$

At last, the 2×2 Fisher Information Matrix (FIM) F given by $-E\left[\frac{\partial^2}{\partial\hat{v}}\log Pr(I|\hat{v})\right]$ for our problem becomes

$$-E\left[\frac{\partial^2}{\partial v_x^2}\log Pr(I|\hat{v})\right] = \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left(\frac{\partial}{\partial v_x}I(x-iv_x, y-iv_y)\right)^2, \tag{4.23}$$

$$-E\left[\frac{\partial^2}{\partial v_y^2}\log Pr(I|\hat{v})\right] = \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\left(\frac{\partial}{\partial v_y}I(x-iv_x, y-iv_y)\right)^2, \tag{4.24}$$

$$-E\left[\frac{\partial^2}{\partial v_x v_y}\log Pr(I|\hat{v})\right] = \frac{1}{\sigma^2}\sum_{S}\sum_{i=0}^{N-1}\frac{\partial}{\partial v_x}I(x-iv_x, y-iv_y)\frac{\partial}{\partial v_y}I(x-iv_x, y-iv_y). \tag{4.25}$$

Using the chain rule, we know that

$$\frac{\partial I(x-iv_x, y-iv_y)}{\partial v_x} = \frac{\partial I}{\partial x}\frac{\partial(x-iv_x)}{\partial v_x} + \frac{\partial I}{\partial y}\frac{\partial(y-iv_y)}{\partial v_x} = -\frac{\partial I}{\partial x}i \tag{4.26}$$

and similarly,

$$\frac{\partial I(x-iv_x, y-iv_y)}{\partial v_y} = -\frac{\partial I}{\partial y}i. \tag{4.27}$$

Then, the Fisher Information Matrix for our particular problem becomes

$$\mathbf{F}(\mathbf{v}) = \frac{1}{\sigma^2}\left(\sum_{i=1}^{N-1} i^2\right)\left[\begin{array}{cc} \sum_S I_x^2 & \sum_S I_x I_y \\ \sum_S I_x I_y & \sum_S I_y^2 \end{array}\right], \tag{4.28}$$

or, what is the same,

$$\mathbf{F}(\mathbf{v}) = \frac{1}{\sigma^2}\left(\frac{N(N-1)(2N-1)}{6}\right)\left[\begin{array}{cc} \sum_S I_x^2 & \sum_S I_x I_y \\ \sum_S I_x I_y & \sum_S I_y^2 \end{array}\right], \tag{4.29}$$

so that the Cramer-Rao bounds for performing registration on a set of $N = 64$ consecutive images is

$$var(v_x) \geq \mathbf{F}_{11}^{-1} = \frac{\sigma^2 \sum_S I_y^2}{Det(\mathbf{F})}\left(\sum_{i=1}^{N-1} i^2\right)^{-1}, \tag{4.30}$$

$$var(v_y) \geq \mathbf{F}_{22}^{-1} = \frac{\sigma^2 \sum_S I_x^2}{Det(\mathbf{F})}\left(\sum_{i=1}^{N-1} i^2\right)^{-1}. \tag{4.31}$$

where $Det(\mathbf{F}) = \sum_S I_x^2 \sum_S I_y^2 - (\sum_S I_x I_y)^2$. This result is remarkable and sligthly puzzling. Assuming that the Cramer-Rao estimates are tight, they mean that with N image samples we can divide the variance by an order of magnitude of N^3 and not N as would be expected for a simple average of N independent noisy estimates.

If we imagine now that a more reliable algorithm could match each image to each, and that the number of "independent" shift estimate would therefore be $N(N+1)/2$, still, the variance of an average of all estimates obtained in that way would be, under the (inexact) independence assumption of all of these estimates, of the order of $1/N^2$ times the variance of a single estimate. But here the Cramer Rao factor is still more favorable, of the order of $1/N^3$.

4.3.4 Conditions based on the SNR of the images

In order for the algorithm to correctly register a line, its images must have a good SNR. There must be significantly more information than noise. If this does not hold, it implies there are actually no objects to register, or that they are dominated by the noise. In order to measure efficiently the SNR and to ensure reliable estimates for the GBSE method and for the bias, it appears that getting a reliable value of the energies of the image $\sum_x I_x^2$ and $\sum_x I_y^2$ is crucial. The evaluation of these quantities becomes reliable if and only the following ratios are high enough:

$$\theta_x = \frac{\sum_{x,y} \tilde{I}_x(x,y)^2}{2|S|\hat{\sigma}^2}, \tag{4.32}$$

$$\theta_y = \frac{\sum_{x,y} \tilde{I}_y(x,y)^2}{2|S|\hat{\sigma}^2}, \tag{4.33}$$

where $|S|$ is the total amount of pixels of the computed gradient and $\hat{\sigma}^2$ is the average variance of the estimated noise, given by Eq. (4.2). Notice that $\theta_x, \theta_y \geq 1$ and that a value strictly larger that 1 entails the presence of signal while a value close to 1 entails that there is only noise in the observed image. We fix later a lower safety threshold for these values at 10, but we shall see that any value between 5 and 50 seems to work as well to discriminate the reliable lines from those which are not (it is probably more important to have a very sure threshold to discard problematic lines than to have it tight).

4.4 Implementation

This section describes all proposed algorithms. The main algorithm works by first validating the line, followed by computing the temporal convolution and performing the multi-image shift estimation. We propose two different variants for the multi-image shift estimation, a naive implementation and a variant that computes the global shift using a smart method that indirectly reduces the inherent bias of the shift estimation method. We also include a third variant which focuses on speed. While we describe the main algorithms together with their underlying ideas here, we include for the sake of completion the rest of the algorithms used in Appendix C. We refer the reader to them for a more complete understanding of the proposed methods.

4.4.1 Gradient-based shift estimation algorithm for noisy images

We begin this section with the presentation of Algorithm 5. This algorithm describes the general proposed stab-active method showing how each line is processed. We propose to first estimate the noise variance, which is later used for validation purposes. The validation algorithm, taking into account the measured noise, proposes a value indicating a suitable number of images to perform temporal accumulation. Once this is done, the global shift is estimated using the temporally convolved set.

Algorithm 5 Optical Flow for noisy images

```
Input: $I(t), 1 \leq t \leq 64$ images, $p \in \mathbb{N}$, $itQty$ amount of iterations, $interpMethod$ interpolation method used.
Output: $offset \in \mathbb{R}^2$ global estimated frame to frame displacement.

$\sigma^2 \leftarrow EstimateNoise(I)$                                      ▷ Algorithm 20
$[validLine, p] \leftarrow VerifySNR(I, \sigma^2, th = 10, maxP = 4)$       ▷ Algorithm 9
if $validLine$ then
    $I_{TC} \leftarrow TemporalConvolution(I, p)$                           ▷ Algorithm 22
    $offset \leftarrow ComputeOffset(I_{TC}, itQty, interpMethod)$          ▷ Algorithm 6
else
    $I_R \leftarrow \emptyset$
end if
```

Algorithm 20, available in Appendix C, estimates the noise mean variance $\hat{\sigma}$ by averaging over all noise estimations between consecutive frames. Algorithm 22 describes the (obvious) temporal convolution which reduces the noise before flow estimation. Finally, Algorithm 23 computes the optical flow between two consecutive frames by using the gradient-based shift estimation approach. The gradient estimation method used is explained in Algorithm 21.

This shift estimation algorithm was made iterative so that, after computing a first estimation of the shift between both frames, it interpolates the second frame and recalculates the shift again, in order to improve performance. In fact, this iterative motion corrects the algorithm's bias without needing to estimate it (see Section 2.4.2). Also, since it only uses the derivative of the first image, the iterative process does not recalculate it, thus permitting a faster implementation. However, obviously, the iterative process implies performing interpolation on the images, which, depending on the interpolation method, may imply heavy computation costs. Three interpolation methods were considered: bilinear, bicubic and cubic spline interpolation. Fast implementations for these methods along with other alternative approaches are analyzed in Appendix A of this thesis. While bilinear interpolation is the fastest method, its results are not acceptable, so it will not be considered in this study. Interpolation methods working in the frequency domain were also discarded due to the impossibility of installing dedicated FFT hardware based on the constraints posed by CNES.

Two methods were proposed to perform the multi-image shift estimation. Algorithm 6 describes the calculation of the offset of each pair of consecutive frames. Then the average is computed on the set of calculated offsets to obtain the final shift estimate for the whole line. A diagram depicting its behaviour is shown in Fig. 4.3.

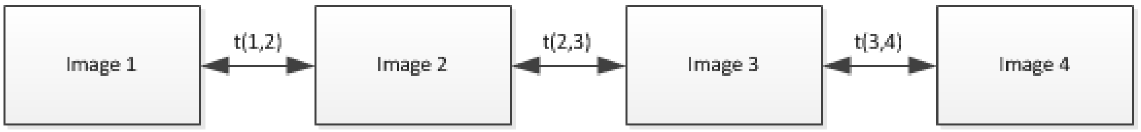


Figure 4.3 – Behaviour of the naive multi-image shift estimation method of Algorithm 6 with four images. All shifts are estimated between consecutive frames and averaged.

Algorithm 6 ComputeOffset (First Version)

```
Input: $I(t), 1 \leq t \leq 64 - 2p$ temporal convolved images, $itQty$ amount of iterations, $interpMethod$ interpolation method used.
Output: $offset$ global estimated frame to frame transition.

for $i = 1$ to $64 - 2p$ do
    $[I_x, I_y] \leftarrow ComputeGradient(I(i))$    ▷ Algorithm 21
    $offsets(i) \leftarrow EstimateShift(I(i), I(i+1), I_x, I_y, itQty, interpMethod)$    ▷ Algorithm 23
end for
$offset \leftarrow$ Mean$(offsets)$
```

Note that Algorithm 6 estimates the offsets between frames independently, followed by taking the mean of these estimates. Then if the shift estimation method is biased (as we have observed in Section 4.3.2), then this bias will be accumulated along all estimates yielding less accurate results. Indeed, the gradient-based shift estimation method underestimates the shift under high noise or large displacement magnitudes. Then Algorithm 6 will also underestimate the global shift.

To avoid this, we propose Algorithm 7, an alternative that estimates the shift by compensating the bias. In this new method, the shift is estimated between the first image and every other in sequential order. However, as we get further away from the first image, the shift to be estimated becomes larger, jeopardizing the GBSE method when the distance surpasses one pixel. To amend this, the previous estimated shift could be used so that as soon as it exceeds half a pixel, the following image is resampled back, thus shifting it towards the first image. Even more, this procedure could be done pixel-wise requiring no further computational cost, while still keeping both images closer than one pixel. The behaviour of this method is shown in Fig. 4.4.

This second algorithm therefore calculates the offsets for the whole line with respect to the first frame. Thus, it avoids accumulating the calculated offsets in order when calculating the final average. Furthermore, the gradient is only calculated for the first image, thus considerably accelerating the implementation.

Finally, a third variant is proposed. Its idea is to directly estimate the global translation using only the first and the last frames and averaging over the amount of images. The problem with this approach is that the GBSE, as explained before, does not perform well under displacements larger than one pixel, so a multiscale strategy must be adopted

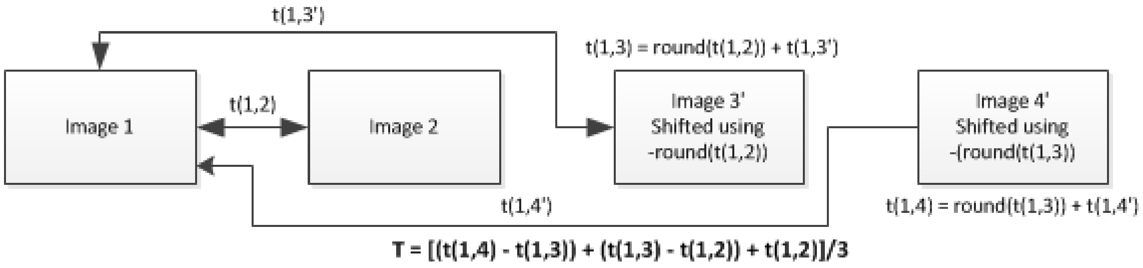

t(1,3')
t(1,3) = round(t(1,2)) + t(1,3')
Image 1
t(1,2)
Image 2
Image 3' Shifted using -round(t(1,2))
Image 4' Shifted using -(round(t(1,3))
t(1,4')
t(1,4) = round(t(1,3)) + t(1,4')
T = [(t(1,4) - t(1,3)) + (t(1,3) - t(1,2)) + t(1,2)]/3

Figure 4.4 – Behaviour of the proposed multi-image shift estimation method of Algorithm 7 with four images. First, the shift estimated between the first two images is used to resample image 3, obtaining image 3'. The shift assigned to the third image is then the sum of the accumulated previous shift plus the shift estimated between image 1 and 3'. This assigned shift is used to resample the fourth image, yielding image 4', and the process repeats. At the end, the returned shift is given by the average of all deaccumulated shifts. Note that the resampling is at the pixel level therefore no interpolation is required.

Algorithm 7 ComputeOffset (Version 2)

Input: $I_{TC}(k), 1 \leq k \leq 64 - 2p$, $p \in \mathbb{N}$, $itQty$ amount of iterations, $interpMethod$ interpolation method used.
Output: $offset$ global estimated transition.

```
$I_x, I_y \leftarrow ComputeGradient(I_{TC}(1))$    ▷ Algorithm 21
$previous \leftarrow [0, 0]$
for $i = 1$ to $64 - 2 * p - 1$ do
    $[v_x, v_y] \leftarrow round(previous)$
    $I_{TC}(i+1) \leftarrow PixelwiseResampling(I_{TC}(i+1), -v_x, -v_y)$    ▷ Shift the image pixelwise $[-v_x, -v_y]$ pixels
    $acum(i) \leftarrow [v_x, v_y] + EstimateShift(I_{TC}(1), I_{TC}(i+1), I_x, I_y, itQty, interpMethod)$    ▷ Alg. 23
    $previous \leftarrow acum(i)$
end for
$offsets \leftarrow DeacumulateOffsets(acum)$    ▷ subtract each element with its previous from the last until the first
$offset \leftarrow$ Mean$(offsets)$
```

so that for a particular scale, the displacement gets lower than this value. In our case, we chose to zoom out both images three times, thus reducing the size of the image, for each zoom out, to half its original size. To follows that the maximum allowed displacement would be of 8 pixels, which suits the requirements of our problem. In fact, for the provided CNES dataset, the maximum displacement per line is of 4.5 pixels. In order to perform the zoom out, a Gaussian filter is applied before subsampling the image, in order to avoid undesired aliasing effects. Finally, once the Gaussian pyramid is built, the displacement is calculated at each scale and used to resample the next scale. Again, to speed up the method, integer resampling is performed to avoid the heavy costs induced by interpolating the image. This algorithm is described in Alg. 8.

Algorithm 8 CalculateExtendedTransition

```
Input: $I_1, I_2$ input images, $I_x, I_y$ gradient of the first image, $itQty$ amount of iterations.
Output: $offset$ estimated transition.

$[I_1^1, I_1^2, I_1^3, I_1^4] \leftarrow BuildGaussianPyramid(I_1, 4)$    ▷ Burt&Adelson's Gaussian Pyramid [28]
$[I_2^1, I_2^2, I_2^3, I_2^4] \leftarrow BuildGaussianPyramid(I_2, 4)$    ▷ Burt&Adelson's Gaussian Pyramid [28]
$acumShift \leftarrow [0, 0]$
for $i = 4$ to $2$ do
    $I_x^i, I_y^i \leftarrow ComputeGradient(I_1^i)$    ▷ Algorithm 21
    $shift \leftarrow EstimateShift(I_1^i, I_2^i, I_x^i, I_y^i, itQty, bicubic)$    ▷ Force bicubic interpolation for speed
    $acum \leftarrow acum * 2 + shift * 2$    ▷ Resample image in the next scale
    $[I_2^{i-1}, x, y] \leftarrow PixelwiseResampling(I_2^{i-1}, -acum)$    ▷ Shift the image pixel-wise $[-v_x, -v_y]$ pixels
    if $i == 2$ then
        $acum \leftarrow [x, y]$    ▷ For the last scale, keep the shift used for resampling.
    end if
end for
$offset \leftarrow acum + EstimateShift(I_1^1, I_2^1, I_x, I_y, itQty, bicubic)$    ▷ Using algorithm 23
$offset \leftarrow offset/64 - 2 * p - 1$
```

4.4.2 Automatic calculation of p for each line

A question that arises from this study is the choice of the right value for the parameter p, which counts the number of accumulated frames in the temporal convolution. This issue is related to the SNR verification for every line, which in Alg. 5, is performed using the $VerifySNR$ method. There are several external constraints that make this election non-trivial, namely the amount of noise in the input images, the total translation of the line, and the desired algorithm's precision and computation cost. As an example, under low noise conditions, it is immediate that accumulating more images would not improve results. Also, if the total displacement is large enough and objects are moving too fast within the sequence, setting a high value for p would possibly imply accumulating the same object in two different positions, thus implying a lower final precision. On the other side, if the line has plenty of noise, more accumulation is needed otherwise the GBSE method would probably fail. Last but not least, the SNR verification could be performed by averaging every frame of the line, or on specific frames, depending on the desired speed of the algorithm. For example, in Algorithm 8, where only two frames are used to compute the global offset, it is better to verify just these two frames.

A simple idea that combines all these constraints is implemented in Alg. 9, where $p = 2$ is first evaluated, and is incremented exponentially until proper SNR values are found or until a maximum threshold for p is achieved, implying the line does not have enough information to be processed (i.e. the signal is dominated by noise). Note that this verification, for speed reasons, is performed on the first and the last frame of the line only. However, another more accurate variant of this algorithm was implemented that performs the verification on every frame, however this algorithm was omitted since it adds no further information. Note that Algorithm 9 calculates the gradient for the first time-convolved image and this is the only place where it is actually being computed throughout the whole process if the second version of $ComputeOffset$ algorithm is used.

Algorithm 9 VerifySNR

```
Input: $I$ input images, $\sigma^2$ estimated noise for the line, $th = 10$ desired threshold for $\theta$ SNR value (algorithm parameter), $MAXP = 4$ indicates the maximum value for $p$ ($p \leq 2^{MAXP} = 16$)
Output: $verification$ verifies if line has enough SNR to be processed, $p$ final value for the parameter, $I_F, I_L$ first and last temporal convoluted images, $I_x, I_y$ gradient of $I_F$.
$p \leftarrow 1$
$j \leftarrow 1$
$verification \leftarrow false$
while $j \leq MAXP$ and $\neg(verification)$ do
    $p \leftarrow p * 2$
    $I_F \leftarrow \sum_{i=1}^{2*p+1} I(i)$
    $I_L \leftarrow \sum_{i=64-2p}^{64} I(i)$
    // Update the noise variance after temporal convolution
    $final\sigma^2 \leftarrow \sigma^2 * (2 * p)$
    $[I_x, I_y] \leftarrow ComputeGradient(I_F)$ // Using algorithm 21
    $[\theta_1, \theta_2] \leftarrow CalculateSNR(I_x, I_y, final\sigma^2)$ // Using equations (4.32) and (4.33)
    $[I_{xLast}, I_{yLast}] \leftarrow ComputeGradient(I_L)$ // Using algorithm 21
    $[\theta_3, \theta_4] \leftarrow CalculateSNR(I_{xLast}, I_{yLast}, final\sigma^2)$; // Using equations (4.32) and (4.33)
    $verification \leftarrow (\theta_1 > th$ and $\theta_2 > th$ and $\theta_3 > th$ and $\theta_4 > th)$
    $j = j + 1$
end while
```

4.4.3 Complexity of the analyzed algorithms

Due to the demanding computational constraints, a full analysis on the cost of the described algorithms will be given next. Interpolation intended for visualization purposes will be skipped. For every algorithm, the complexity is linear in the number of pixels.

First, Algorithm 5 will be analyzed. There are three operations per pixel to compute $\hat{\sigma}$ in Algorithm 20. For the SNR validation, the algorithm takes at most one operation to calculate I_F and another for I_L. Since the gradient is calculated on just these two images and not the whole set, its cost is negligible. The same happens when calculating the θ values, and since the algorithm iterates at most $maxP = 4$ times, this yields a cost between 2 and 8 operations per pixel. The complexity of Algorithm 22 is $2p + 1$ operations per pixel, implying that in lines with low noise conditions, 5 ($p = 2$) operations are performed, however under heavy noisy situation, 33 ($p = 16$) operations could be used.

For the ComputeOffset (Version 1) algorithm, the gradient for every image is computed as in Algorithm 21, taking 6 operations per pixel. This is followed by the estimation of the translation using the optical flow (Algorithm 23). In this algorithm, four operations are needed when averaging every four neighboring pixels, and two more for determining each sI_x^2, sI_y^2 and sI_xI_y. That makes 14 operations before the loop. Then inside the loop, another four neighboring pixels average is performed taking again 4 operations, one more to compute I_t and 4 more to compute both sI_xI_t and sI_yI_t, yielding 9 operations per pixel. The cost of performing interpolation should be added if necessary. We shall call $InterpCost$ this cost. Then Algorithm 23 takes $14 + 9 * itQty + (itQty - 1) * InterpCost$ operations. If only one iteration is used, then no interpolation is done and the method would take 23 operations per pixel. Finally, the total number of operations using Algorithm 5 with Algorithm 6 is between $\mathbf{26 + 9 * itQty + (itQty - 1) * InterpCost}$ and $\mathbf{60 + 9 * itQty + (itQty - 1) * InterpCost}$. Again, if just one iteration is used, this would be between 35 and 69 operations per pixel depending on the SNR of the line. The real amount of operations per pixel is in fact lower . Indeed, when a high value for p is used, less offsets are calculated. However the given amount of operations per pixel is a rough approximation and should be taken as a reference and not as an exact value.

When the second version of ComputeOffset is used, described in Algorithm 7, the number of operations is reduced by five, since the gradient for each image does not have to be computed for every frame (just for the first frame). However an operation is needed to perform the integer shift.

Finally, for the CalculateExtendedTransition algorithm 8, the number of operations per pixel is indeed less than 2, since only the first and the last convolved images are used, ignoring the rest. The value 2 is because to create both images, at most it requires one operation per pixel for each. However, it has to be noted that due to interpolation, subsampling and other operations performed in this algorithm, its computational cost, as it will be seen in the results, is not considerably lower than the cost of Algorithm 5.

4.5 Results

To evaluate the performance of the proposed method, a simulated test set together with its ground truth was provided by CNES (French space agency). It simulated arrays of noisy images (with Poisson noise) obtained from a noiseless aerial raw image, which was finely resampled by high order splines to simulate subpixel motion. In it, there were approximately 3000 lines, each containing 64 frames, where each frame is a 50×50 image having either 20cm per pixel of 40cm per pixel resolution. Noise varied between lines, while the total shift for the N images on the horizontal plane (δ_x) could range between 0 and 4.5 pixels. On the vertical plane, the provided simulated images did not contain any shift (i.e. $\delta_y = 0$). To measure how the algorithms performed, the Euclidean distance with respect to the true global shift $\left(\frac{\delta_x}{N-1}, \frac{\delta_y}{N-1}\right)$ was used.

In order to compare several alignment methods, a significant subset of lines were chosen so as to represent each type of problem and/or perturbation found in the provided dataset. In total, 13 different lines were chosen: one for each perturbation for both provided resolutions (20cm and 40cm), two lines corresponding to very noisy situations, two lines corresponding to situations where no objects could be recognized (only noise was visible), and two lines where phase-correlation-based methods fail, due to their implicit assumption of periodicity.

4.5.1 Impact of temporal convolution on image noise

To obtain a good estimation of the shifts between images, the noise is attenuated by applying a temporal convolution over the frames in the line. In Fig. 4.5 two consecutive frames are shown along with the absolute value of their difference. This experiment illustrates the dominance of noise over signal, or rather the elimination of signal by making the difference. It also shows that without previous denoising a correct estimation of displacement is unlikely.

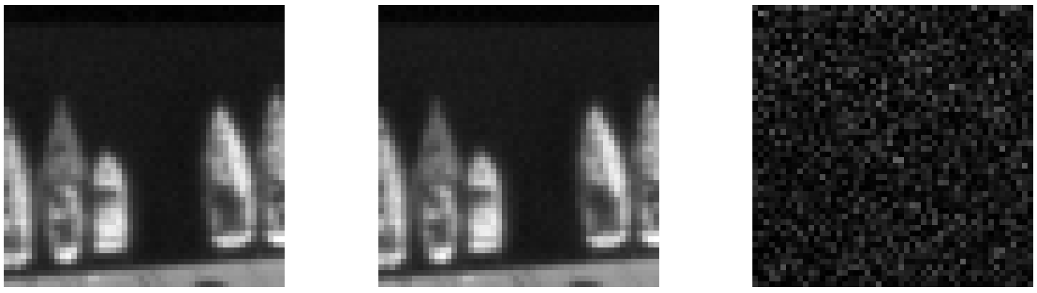

(a) Frame k (b) Frame k+1 (c) $|k+1-k|$

Figure 4.5 – Difference between consecutive frames. This difference should be in principle used to compute I_t, the time derivative. Noise dominates so much in it that it can be used for estimating the noise. However, for the GBSE method, this experiment shows that a temporal smoothing will be necessary.

For our noise estimation method, it has to be noted how in some specific cases it is not precise. This happens with high SNRs when the shift between frames is large. For these cases, as can be seen in Fig. 4.6, when calculating the difference between frames, parts of the moving objects can be perceived, which is later mistaken with noise by our noise estimation method. However, also in these cases, it is certain that I_t, the time derivative, is able to better describe the movement of the scene, which implies that the GBSE method is better conditioned, suggesting that different values for parameter p of the temporal convolution should be used.

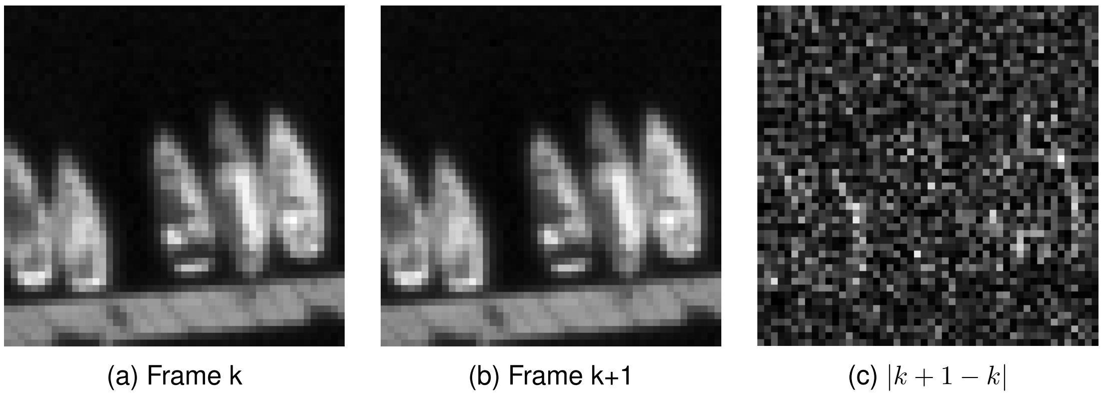


Figure 4.6 – Difference between consecutive frames for line 'SIMU40\P3\L0100'. One can perceive parts of the ships in the difference, which for this specific case, leads to overestimating the noise.

The results of applying a temporal convolution to both lines is shown in figures 4.7 and 4.8. In the first case, using $p = 8$, almost all noise was successfully removed. In the second case, a comparison using several values of p for temporal convolution is given. It can be seen that using $p = 2$ considerable noise can still be perceived, implying that a higher value for p is needed. On the contrary, when using $p = 16$, practically all the noise was removed, however a significant translation of more than one pixel can be seen in the difference between consecutive temporally convolved frames, decreasing the final precision of the gradient estimation method.

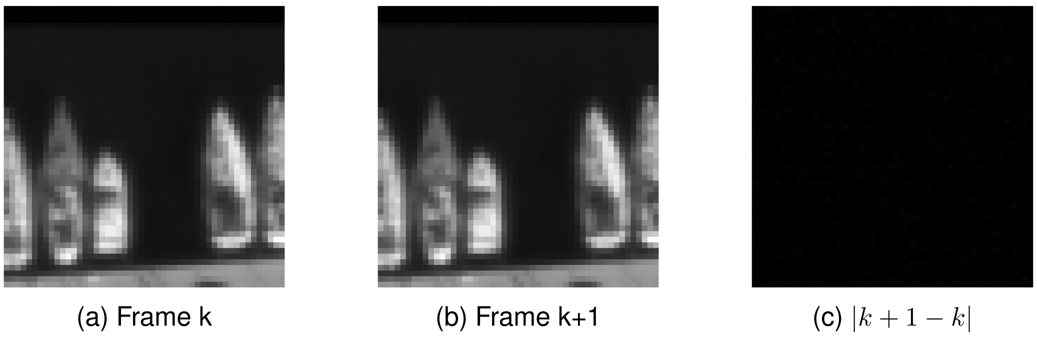


Figure 4.7 – Difference between consecutive frames after applying temporal convolution.

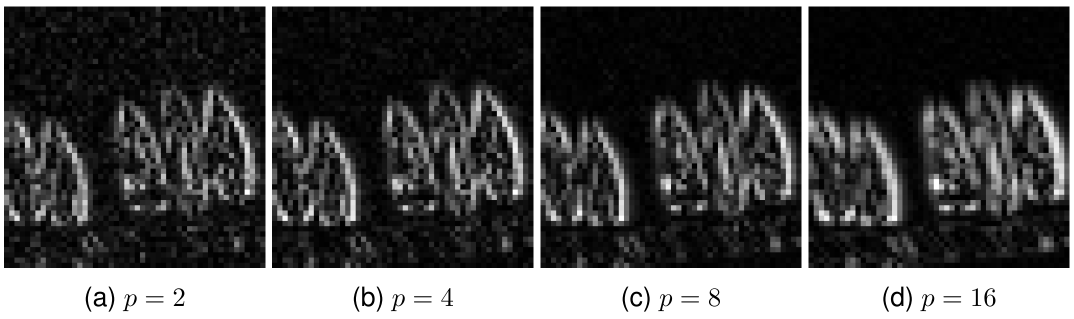
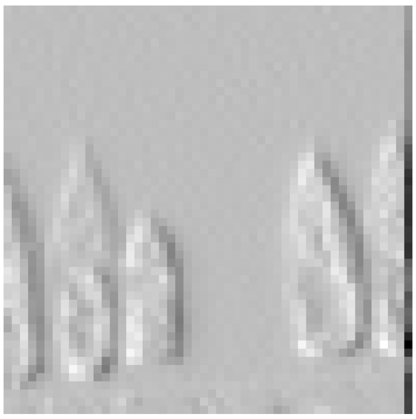


Figure 4.8 – Difference between consecutive frames after applying temporal convolution using different values for parameter p.

Finally, figures 4.9 and 4.10 shown the impact of a temporal convolution on the image gradient. From these results, it seems straightforward that the temporal convolution is necessary for achieving more accurate results.


(a) $\frac{\partial I_1}{\partial x}$

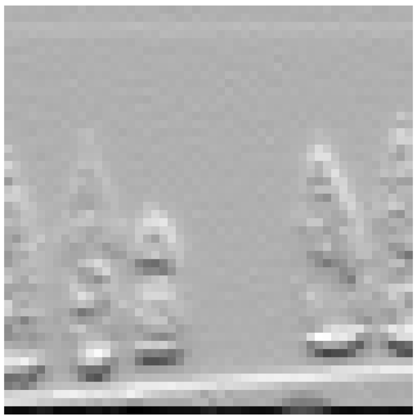
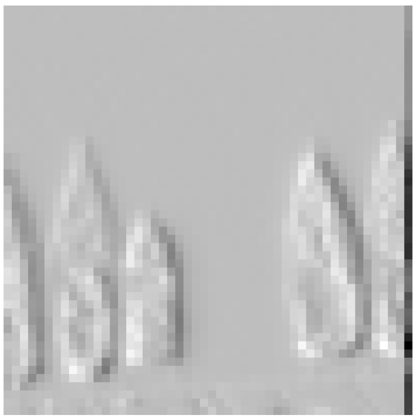

(b) $\frac{\partial I_1}{\partial y}$

Figure 4.9 – Image of the two gradient components before temporal convolution.


(a) $\frac{\partial I_1}{\partial x}$

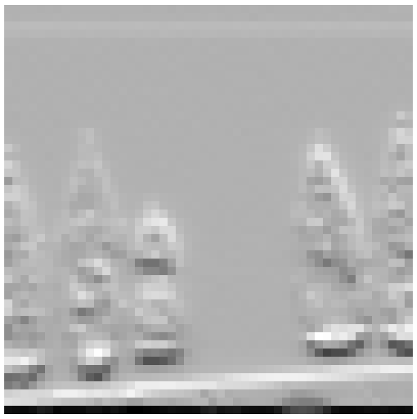

(b) $\frac{\partial I_1}{\partial y}$

Figure 4.10 – Image of the two gradient components after temporal convolution.

4.5.2 Numerical Results

Noise estimation and thresholds

Table 4.1 displays the standard deviation for the estimated noise, the Cramer-Rao bounds of equations (4.30) and (4.31), and both θ values. Note that the θ values are calculated after performing temporal convolution to know if a line can or cannot be properly aligned by our algorithm, or if it should be discarded. Both displayed lower bounds are calculated with respect to the global shift (v_x, v_y) of a line as given in equations (4.30) and (4.31). The prefixes have a meaning: "N" stands for Noisy (images with low SNR), "ON" for Only Noise (images without any visible object) and "PF" for Periodic Failure, meaning that if images are assumed periodic, as in Fourier based methods, results, as it will be shown, drastically fail.

It seems advisable to put a conservative lower threshold on θ_x and θ_y of the order of 10 to avoid these obvious cases where noise dominates signal. This will be further studied in Section 4.5.4. However, it is worth noting that images that have an SNR value below 1.5 have no content at all and should be definitely discarded.

Shift estimation results

To evaluate the performance of the shift estimation between two frames, we used the provided ground truth. For each type of perturbation, the total displacement, uniformly

#	σ	θ_x	θ_y	$\sqrt{var(v_x)}$	$\sqrt{var(v_y)}$
1	4.0043	56.15	45.57	2.8308e-4	2.4028e-4
2	4.1989	45.62	57.77	3.1852e-4	2.6549e-4
3	4.2083	42.54	62.28	3.0110e-4	2.4548e-4
4(N)	5.2102	24.14	8.69	4.0049e-4	4.7340e-4
5(N)	7.9257	2.39	5.80	1.1255e-3	1.0494e-3
6	1.2208	611.88	509.79	8.2380e-5	7.9108e-5
7	1.2925	250.57	110.53	1.6302e-4	2.1596e-4
8	1.2584	253.74	101.26	1.5327e-4	1.5164e-4
9	1.3269	230.06	150.20	1.1277e-4	1.2312e-4
10(ON)	16.9681	1.06	1.05	1.3341e-3	1.3563e-3
11(ON)	17.6862	1.22	1.25	1.5227e-3	1.3984e-3
12(PF)	1.3062	183.17	63.49	1.7067e-4	2.2495e-4
13(PF)	1.2918	114.36	37.52	1.5451e-4	2.0374e-4

Table 4.1 – Estimated standard deviation of the noise σ, θ SNR values, and Cramer-Rao bounds for the selected lines. The low values of θ permit a reliable detection of lines (lines 5, 10, 11) that indeed will prove to be problematic.

distributed between a line, is shown in Table 4.2.

Perturbation	Δ_x	Δ_y	$\frac{\Delta_x}{64}$	$\frac{\Delta_y}{64}$
P0	0	0	0	0
P1	-1.5	0	-0.0234375	0
P2	-3	0	-0.046875	0
P3	-4.5	0	-0.0703125	0

Table 4.2 – Provided ground truth for the database. It shows the displacement, in pixels, for each perturbation type for a whole line.

As a way of measuring how the algorithms performed, the root mean squared error has been used. Then, being $\hat{v}$ the algorithm's estimation of the global shift for a particular line, and Δ the total shift for that line, as given by Table 4.2, then the measured error is

$$RMSE(\hat{v}, \Delta) = \sqrt{(\hat{v_x} * 63 - \Delta_x)^2 + (\hat{v_y} * 63 - \Delta_y)^2}. \tag{4.34}$$

It has to be taken into account that this measurement will in fact calculate the overall error performed by aligning the 64 frames, instead of the error performed by calculating the single global shift $(\hat{v_x}, \hat{v_y})$.

Six algorithms were compared, namely Algorithm 5 on its two variants, Algorithm 8, all of them based on the GBSE method, the fast phase-correlation method of Stone [159], the slower but more more accurate method of Guizar-Sicairos *et al.* [69] and the gradient-correlation approach of Argyriou *et al.* [7] using the second order Christmas gradient estimation. To address the multi-image shift estimation problem, we tested each method under different schemes, namely by estimating the shift between the first and the last frames, and by accumulating the shifts for all consecutive images, and kept the scheme that gave the best results. We also included the global Cramer-Rao bounds given by Eqs. (4.30) and (4.31) by computing $CRLB = \sqrt{var(v_x)^2 + var(v_y)^2}$. It should be noted that all methods were evaluated after the temporal convolution step, so that the conditions were the same for every case.

Using $p = 4$ for the temporal convolution, $itQty = 2$ and spline interpolation, Table 4.3 show the performance for each method.

#	Pert.	CRLB	Alg. 6	Alg. 7	Alg. 8	[159]	[69]	[7]
1	P0	0.0004	0.0199	0.0029	**0.0024**	0.0058	**0.0024**	0.0052
2	P1	0.0004	0.0128	**0.0028**	0.0155	0.1724	0.2336	0.0319
3	P2	0.0004	0.0494	**0.0033**	0.0064	0.3420	0.1633	0.0434
4(N)	P3	0.0006	0.3099	**0.0031**	0.0234	3.1045	0.0324	0.0504
5(N)	P3	0.0015	2.8468	**0.0153**	0.0400	1.4983	0.9852	0.0909
6	P0	0.0001	0.0022	0.0011	**0.0009**	0.0014	0.0011	0.0015
7	P1	0.0003	0.0203	**0.0068**	0.0365	0.2074	0.1686	0.0137
8	P2	0.0002	0.0085	**0.0017**	0.0205	1.3908	0.0044	0.0339
9	P3	0.0002	0.0146	**0.0010**	0.0106	3.7202	0.1477	0.0343
10(ON)	P1	0.0019	-	-	-	-	-	-
11(ON)	P1	0.0021	-	-	-	-	-	-
12(PF)	P2	0.0003	0.1244	**0.0036**	0.0506	3.0708	5.2640	0.4162
13(PF)	P2	0.0003	0.1543	**0.0019**	0.0505	2.9960	2.4721	0.2273
Avg.	-	0.0007	0.3239	**0.0040**	0.0234	1.5009	0.8613	0.0862

Table 4.3 – RMSE of each method for the selected lines.

As can be seen, the second variant (Algorithm 7) gives the best results in almost every case and clearly improves over the naive approach given by Alg. 6. When there is no displacement along the whole line, Alg. 8 achieved the best results, although the difference with Alg. 7 is negligible. Other conclusions drawn from this experiment are the poor tolerance to noise of the Stone method, as we have already foreseen in Chapter 2 of this manuscript, the higher tolerance to noise of gradient-correlation methods, and the poor performance of all correlation-based methods with images having periodic patterns that yield several peaks in their correlation grid. Indeed, an important advantage of gradient-based methods is that they are local in nature, therefore they are less affected by periodic patterns or even the aperture problem. Finally, it should be mentioned that even by using only two iterations, Algorithm 7 gets close to the global Cramer-Rao bounds, being only an order of magnitude below on every evaluated case.

Analysis of the proposed multi-image scheme for gradient-based shift estimation

Table 4.3 shows an evident improvement between the naive sequential shift estimation approach described in Alg. 6 and the proposed multi-image scheme of Alg. 7 which computes the shifts always with respect to the first image. This improvement is not coincidental, indeed the proposed scheme indirectly reduces the bias from the gradient-based shift estimation method proposed. The reason behind this is not apparent since in fact, estimating the shift between images that are farther away using the GBSE approach adds more bias in the results [119].

We will explain this with a practical example. In every case, a non-iterated GBSE method is used to measure the displacements and the displacements shown are always from the horizontal axis (the displacement on the vertical axis was always zero in the provided dataset therefore its analysis is discarded). For visual purposes, we just show the shift estimates obtained from the first 30 frames to prove our claims. First, in Fig. 4.11 we observe the shifts estimated using the naive method from Alg. 6. The shifts between consecutive frames were always underestimated. This is because of the bias in the GBSE method used, analyzed in depth in Section 4.3.2, influenced mainly by the noise on the frames.

The proposed multi-image shift estimation results between each frame with respect to the first are displayed in Fig. 4.12. While the blue bars indicate the estimated shift,

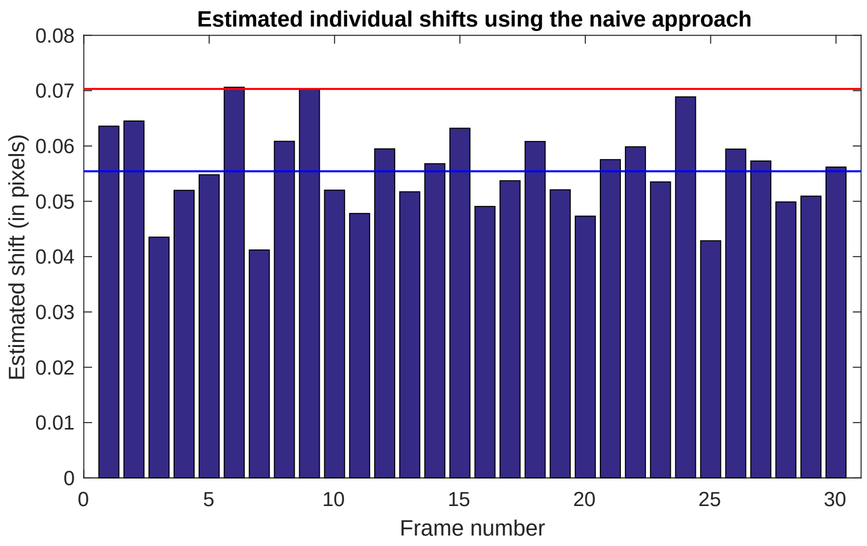


Figure 4.11 – Shifts estimated using the naive multi-image approach of Alg. 6. Each bar represents the estimated shift between both consecutive frames. The underestimation of the shift due to the bias is evident. The red line denotes the ground truth while the blue line represents the resulting average from all estimations.

the yellow bars denote the correct shift that should have been obtained. As mentioned previously, as the processed frame gets farther away from the first one, the ground truth displacement gets larger and so does the bias. This justifies the bigger errors obtained. However, an interesting event occurs when the last estimated shift goes larger than half a pixel. Indeed, the algorithm resamples the next image one pixel in the opposite direction. Then the next shift, which again is underestimated because of the bias, is done so in the opposite direction. Then, when accumulating this underestimated shift, the bias produced in all the previous estimations is compensated. This becomes more evident by observing the accumulated shift from Fig. 4.13. In this example, until the 10th frame, the shift was always underestimated and the error got larger in every frame. Then when the image is resampled one pixel to the left, the next estimated shift will be -1 plus the next (underestimated) shift. This yields a result which is an overestimation that compensates the previous accumulated errors, and that again, because of the bias, will eventually become an underestimation again. In the 25th frame, this procedure is repeated and the accumulated error is compensated.

It should be noted that this procedure succeeds in better estimating the shifts in this particular case, when all shifts between consecutive frames are supposed to be the same and in the same direction. If this would not be the case, then this method would make no sense and should not be considered.

Evaluating the performance on the whole database

In order to evaluate the influence on the results of the parameters, an evaluation of our three gradient-based proposed algorithms was performed on the whole provided database. First, the incidence of the Anscombe VST on the results is shown in tables 4.4, 4.5 and 4.6. From this study it can be concluded that using Anscombe VST is indeed recommended since it leads to significantly improve the precision of the method, specially for the second and the third proposed algorithms. It has to be noted, as can be seen in Table 4.6, that by using the VST computation proposed in Appendix B, it is possible to achieve similar results without applying the square root operation.

This was followed by studying the influence of the parameter $itQty$. Taking as ref-

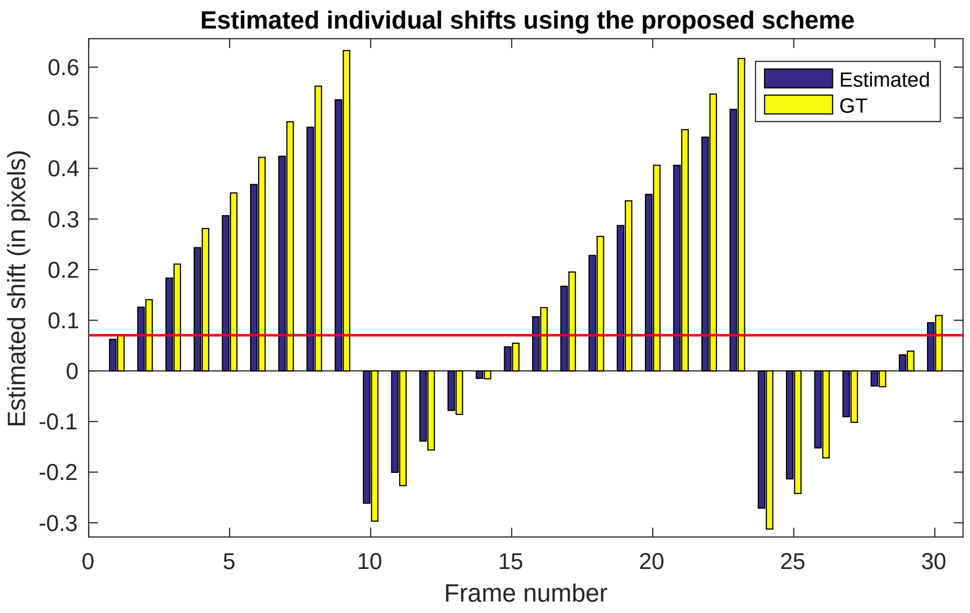


Figure 4.12 – Shifts estimated using the proposed multi-image scheme of Alg. 7. Each bar represents the estimated shift with respect to the first frame. The ground truth bars in yellow represent the correct shift to be estimated based on the displacement between both frames. The red line denotes the ground truth while the blue line represents the resulting average from all estimations.

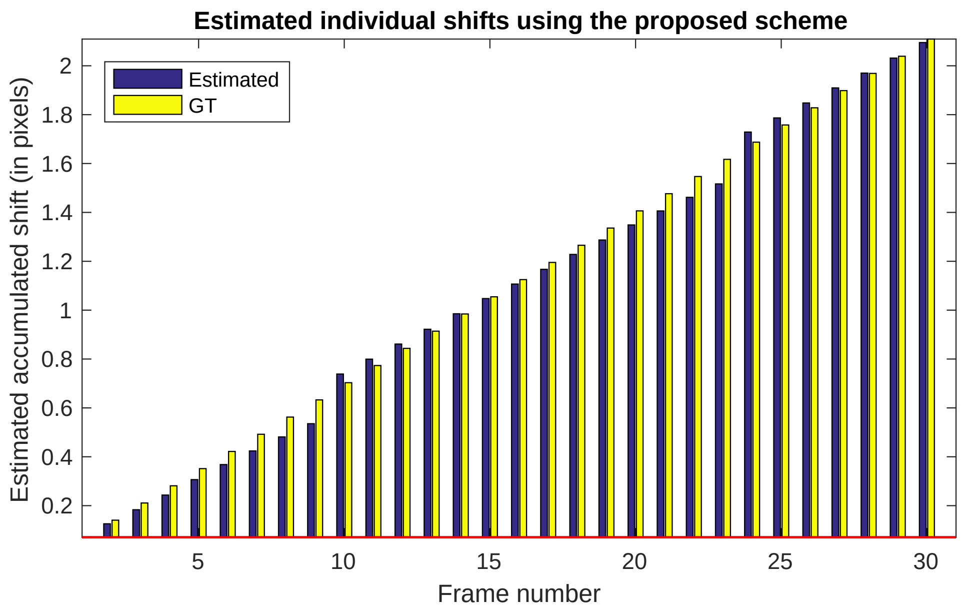


Figure 4.13 – Shifts estimated using the proposed multi-image scheme of Alg. 7. Each bar represents the *accumulated* estimated shifts with respect to the first frame. The ground truth bars in yellow represent the correct shift to be estimated based on the displacement between both frames.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0061	0.0057	0.0122
20	P1	0.3053	0.0502	0.0379
20	P2	0.6661	0.0179	0.1338
20	P3	1.0055	0.0497	0.4511
40	P0	0.0012	0.0013	0.0031
40	P1	0.3492	0.0786	0.0349
40	P2	0.6781	0.0790	0.0834
40	P3	0.9367	0.0964	0.3860
Avg		0.4935	0.0474	0.1428

Table 4.4 – Average $RMSE$ for the provided dataset without applying the Anscombe VST. Parameters: $p = 4, itQty = 1$.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	**0.0058**	**0.0050**	**0.0098**
20	P1	**0.2965**	0.0504	**0.0265**
20	P2	**0.6287**	**0.0155**	**0.0560**
20	P3	**0.9451**	**0.0217**	**0.1656**
40	P0	0.0015	0.0015	**0.0031**
40	P1	**0.3145**	**0.0590**	**0.0282**
40	P2	**0.6123**	**0.0267**	**0.0633**
40	P3	**0.8729**	**0.0447**	**0.2146**
Avg		**0.4596**	**0.0280**	**0.0709**

Table 4.5 – Average $RMSE$ for the provided dataset by applying the Anscombe VST. Parameters: $p = 4, itQty = 1$. Results in bold indicate better performance than in Table 4.4.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	**0.0058**	**0.0049**	**0.0097**
20	P1	**0.2961**	0.0503	**0.0264**
20	P2	**0.6276**	**0.0156**	**0.0561**
20	P3	**0.9453**	**0.0217**	**0.1660**
40	P0	0.0015	0.0014	**0.0031**
40	P1	**0.3146**	**0.0590**	**0.0282**
40	P2	**0.6125**	**0.0268**	**0.0633**
40	P3	**0.8731**	**0.0448**	**0.2104**
Avg		**0.4596**	**0.0281**	**0.0704**

Table 4.6 – Average $RMSE$ for the provided dataset by applying the Anscombe VST using algorithm described in Appendix B. Parameters: $p = 4, itQty = 1$. Results in bold indicate better performance than in Table 4.4.

erence the single-iteration algorithm whose results are shown in Table 4.6, one can see in tables 4.7 and 4.8 the results of the proposed algorithms by varying the number of iterations. It can be seen from the results how by increasing the number of iterations, the algorithm's bias is corrected leading to better results. It is worth remarking that the best overall improvement can be appreciated in the first proposed algorithm since, as explained in Section 4.4.1, this method accumulates the frame-by-frame estimation error, therefore the lower each individual error, the lower the global error. However, Algorithm 7 tries to compensate this bias in a frame to frame basis, so to process the next frame, it does not rely solely on the previous error.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0070	0.0061	**0.0087**
20	P1	**0.0873**	**0.0138**	**0.0165**
20	P2	**0.1936**	**0.0099**	0.0644
20	P3	**0.3098**	**0.0100**	**0.1543**
40	P0	0.0018	0.0018	**0.0027**
40	P1	**0.0846**	**0.0136**	**0.0125**
40	P2	**0.1683**	**0.0062**	0.0704
40	P3	**0.2372**	**0.0062**	**0.1842**
Avg		**0.1362**	**0.0084**	**0.0642**

Table 4.7 – Average $RMSE$ for the provided dataset. Parameters: $p = 4, itQty = 2$ using spline interpolation. Results in bold indicate better performance than reference Table 4.6.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0074	0.0065	**0.0084**
20	P1	**0.0336**	**0.0074**	**0.0139**
20	P2	**0.0790**	**0.0082**	0.0667
20	P3	**0.1402**	**0.0087**	**0.1462**
40	P0	0.0019	0.0019	**0.0027**
40	P1	**0.0252**	**0.0052**	0.0092
40	P2	**0.0541**	**0.0039**	0.0722
40	P3	**0.0862**	**0.0052**	**0.1714**
Avg		**0.0534**	**0.0059**	**0.0613**

Table 4.8 – Average $RMSE$ for the provided dataset. Parameters: $p = 4, itQty = 3$ using spline interpolation. Results in bold indicate better performance than using $itQty = 2$ (Table 4.7).

Another interesting comparison performed was the influence on the results of parameter p, which indicates how many frames should be accumulated in the temporal convolution. Results are shown in tables 4.9, 4.10 and 4.11, including as well Table 4.6 with $p = 4$. For Algorithm 6, the bigger the parameter p, the better the performance, however, this does not occur with the other methods. Quite the contrary, for Algorithm 7, the lower the p, the better the results. However, this does not hold on every case. When the input line has considerable noise, increasing the number of accumulated frames usually improves the results, implying a direct relationship between the average noise of the line and the best possible p value. Also, a too high value for p in situations where it is not needed, namely, with high SNR values, decreases the precision. This study justifies the idea of the proposed Algorithm 9.

At last, the influence of the interpolation method was evaluated when the algorithms were made iterative, i.e. when the number of iterations was larger than one. Using the same parameters as in Table 4.7, we evaluated the iterative algorithms this time using

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0073	0.0044	0.0105
20	P1	0.3486	0.0292	0.0303
20	P2	0.7483	0.0393	0.0583
20	P3	1.1295	0.0298	0.1738
40	P0	0.0014	0.0013	0.0039
40	P1	0.3213	0.0386	0.0309
40	P2	0.6330	0.0475	0.0676
40	P3	0.9243	0.0588	0.2455
Avg		0.5142	0.0311	0.0776

Table 4.9 – Average $RMSE$ for the provided dataset. Parameters: $p = 2, itQty = 1$.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0059	0.0057	0.0079
20	P1	0.2576	0.0851	0.0288
20	P2	0.5301	0.0408	0.0479
20	P3	0.7677	0.0270	0.1101
40	P0	0.0017	0.0017	0.0028
40	P1	0.3062	0.0913	0.0333
40	P2	0.5697	0.0438	0.0485
40	P3	0.7604	0.0277	0.1457
Avg		0.3999	0.0404	0.0531

Table 4.10 – Average $RMSE$ for the provided dataset. Parameters: $p = 8, itQty = 1$.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0057	0.0057	0.0072
20	P1	0.2258	0.0643	0.0642
20	P2	0.4200	0.0105	0.0988
20	P3	0.5496	0.0503	0.1277
40	P0	0.0016	0.0016	0.0026
40	P1	0.2867	0.1787	0.0709
40	P2	0.4603	0.0118	0.1043
40	P3	0.5462	0.0466	0.1311
Avg		0.3120	0.0462	0.0759

Table 4.11 – Average $RMSE$ for the provided dataset. Parameters: $p = 16, itQty = 1$.

bicubic interpolation instead of spline interpolation. Results shown in Table 4.12 show a slight decrease in performance compared to spline interpolation, however the difference in execution time, which for spline interpolation usually involves almost three times the computational cost, seems to justify this precision loss.

Res.	Pert.	Alg. 6	Alg. 7	Alg. 8
20	P0	0.0079	0.0066	0.0087
20	P1	0.0978	0.0132	0.0165
20	P2	0.1896	0.0095	0.0644
20	P3	0.3107	0.0097	0.1543
40	P0	0.0023	0.0020	0.0027
40	P1	0.1083	0.0121	0.0125
40	P2	0.2314	0.0059	0.0704
40	P3	0.2945	0.0078	0.1842
Avg		0.1553	0.0083	0.0642

Table 4.12 – Average $RMSE$ for the provided dataset. Parameters: $p = 4, itQty = 2$. Interpolation: bicubic.

4.5.3 Visual Results

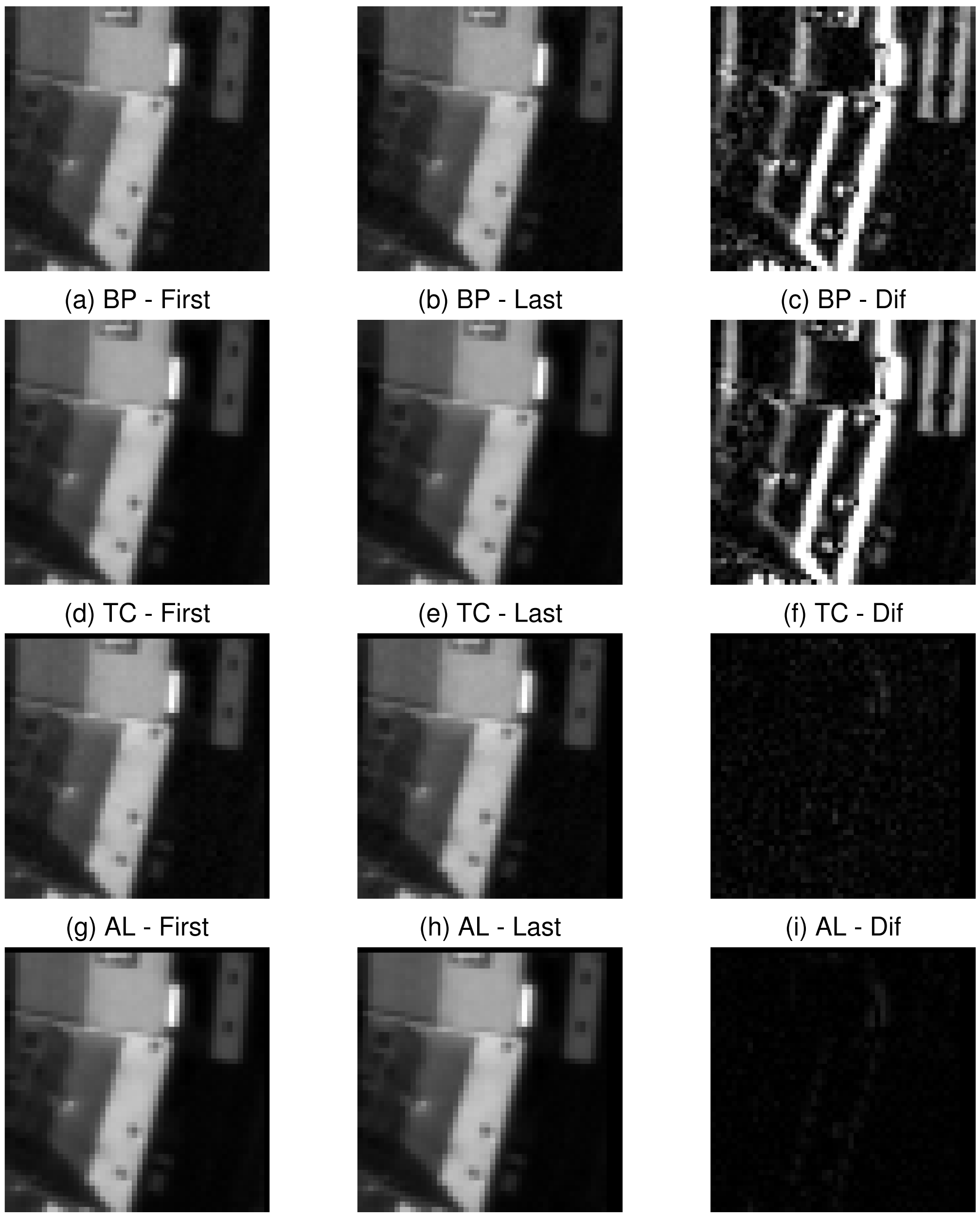
In order to analyze the results visually, the registration process was divided in 4 distinct steps defined by the following four acronyms:

- BP: Before Processing, the original images of the line which are the raw input to the algorithm.
- TC: after Temporal Convolution (in order to get rid of noise and possible perturbations).
- AL: after ALigning the frames.
- FI: FInal result after performing a temporal convolution on the aligned frames.

For obvious space constraints, three images will be shown for each step: the first frame (out of the 64), the last frame, and the absolute value of their difference. Since the values for the differences are so small that they may not be recognizable by the human eye, a multiplicative factor has been applied.

Figures 4.14 and 4.15 show the results for lines SIMU40/P2/L0804 and SIMU40/P2/L0900. It can be seen that the methods drastically align the frames, reducing considerably the difference. In figures 4.16 and 4.17, two noisy lines are evaluated. One can observe how registration results for these lines are not as good, due to the high noise. In these cases, using more frames for the temporal convolution improves the results. However, this implies estimating the shifts using less frames, which, if there is a lot of movement going on, usually leads to poor results. Despite these caveats, the results are still acceptable using our registration method, since most of the perceivable differences in the final result are attributable to noise.

The fact that the methods still work under these conditions is mainly due to the performed temporal convolution, which gets rid of noise in order to perform a better estimation. However, the number of frames used to do this convolution is crucial, since a too low value will lead to poor results, as shown in Figure 4.18. In this case, a total of 8 frames were accumulated, which leads to a wrong registration, while using 16 frames yielded an acceptable performance.



(a) BP - First (b) BP - Last (c) BP - Dif

(d) TC - First (e) TC - Last (f) TC - Dif

(g) AL - First (h) AL - Last (i) AL - Dif

(j) FI - First (k) FI - Last (l) FI - Dif

Figure 4.14 – Results for line SIMU40/P2/L0804. BP, before processing, TC, after temporal convolution, AL, after alignment, FI, temporal convolution applied after alignment.

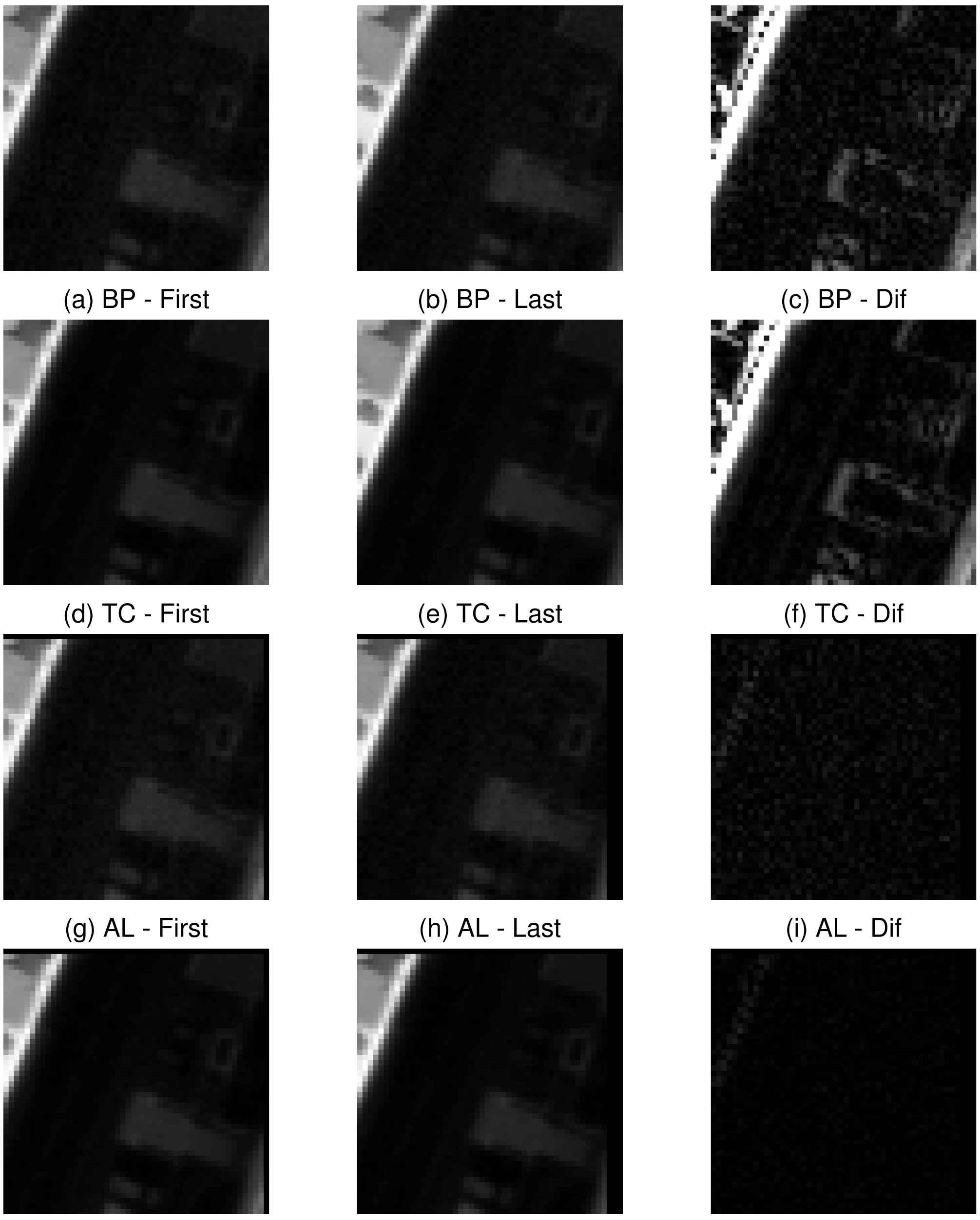


Figure 4.15 – Results for line SIMU40/P2/L0884. BP, before processing, TC, after temporal convolution, AL, after alignment, FI, temporal convolution applied after alignment.

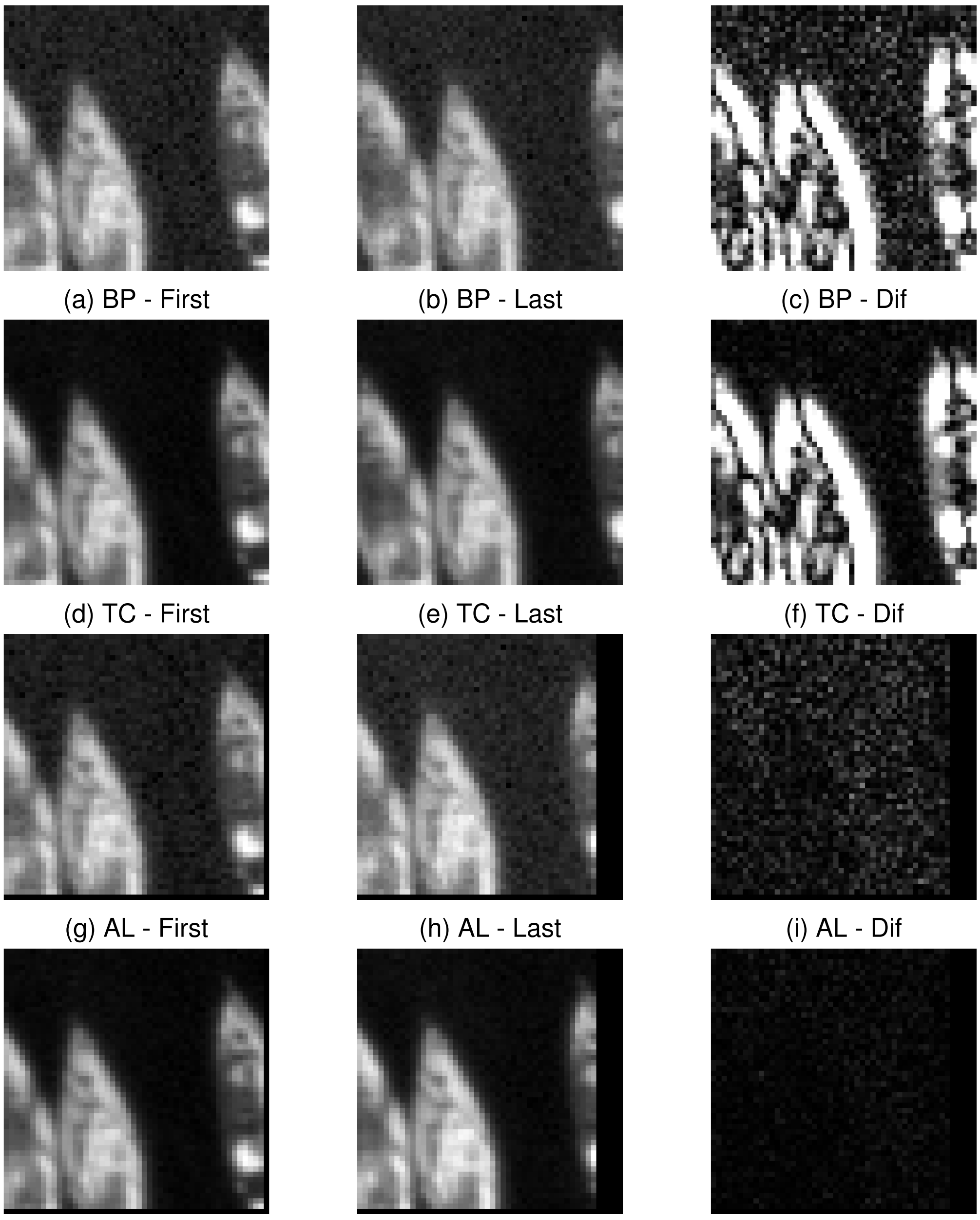

(a) BP - First (b) BP - Last (c) BP - Dif
(d) TC - First (e) TC - Last (f) TC - Dif
(g) AL - First (h) AL - Last (i) AL - Dif
(j) FI - First (k) FI - Last (l) FI - Dif

Figure 4.16 – Results for line SIMU20/P3/L0228. BP, before processing, TC, after temporal convolution, AL, after alignment, FI, temporal convolution applied after alignment.

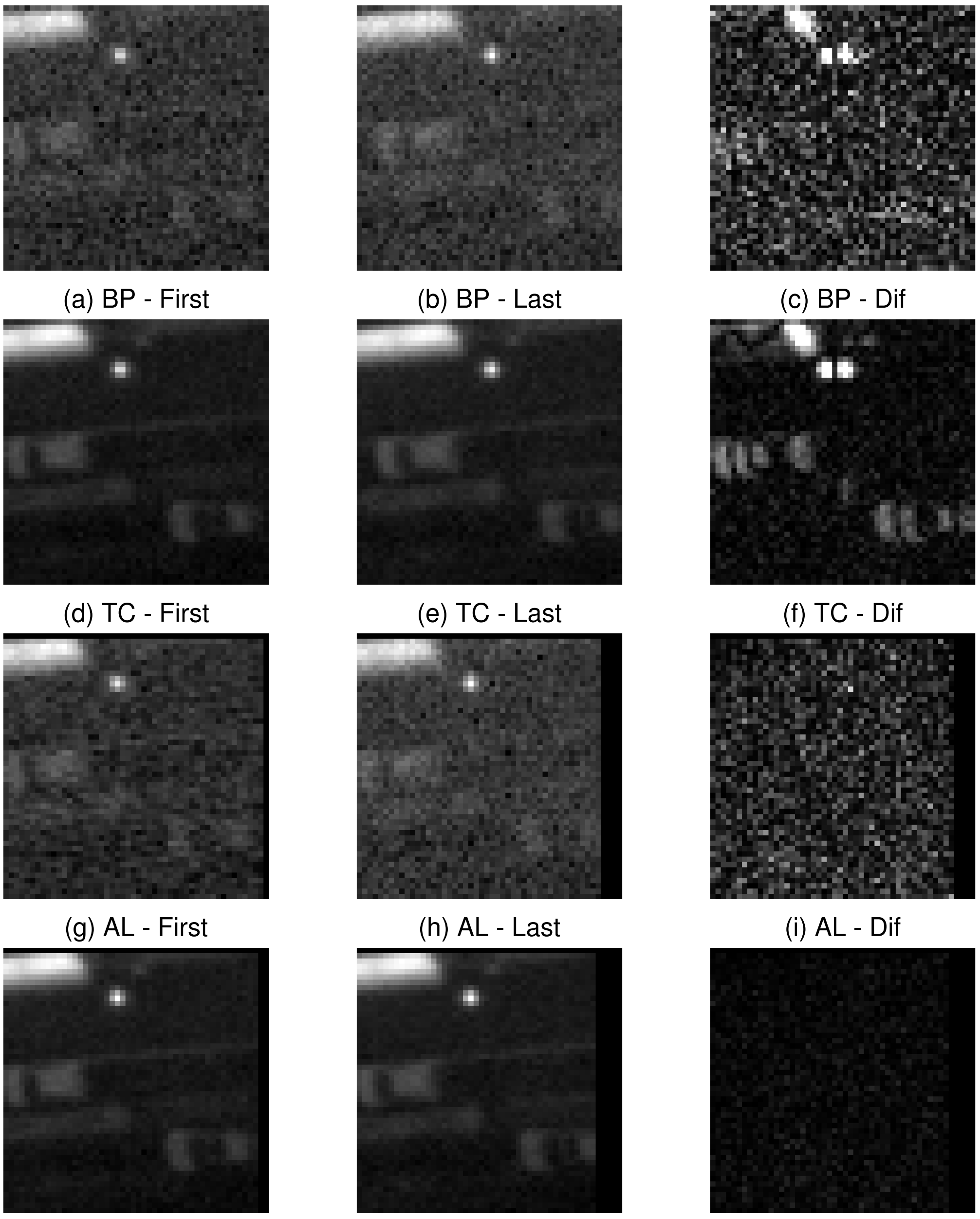
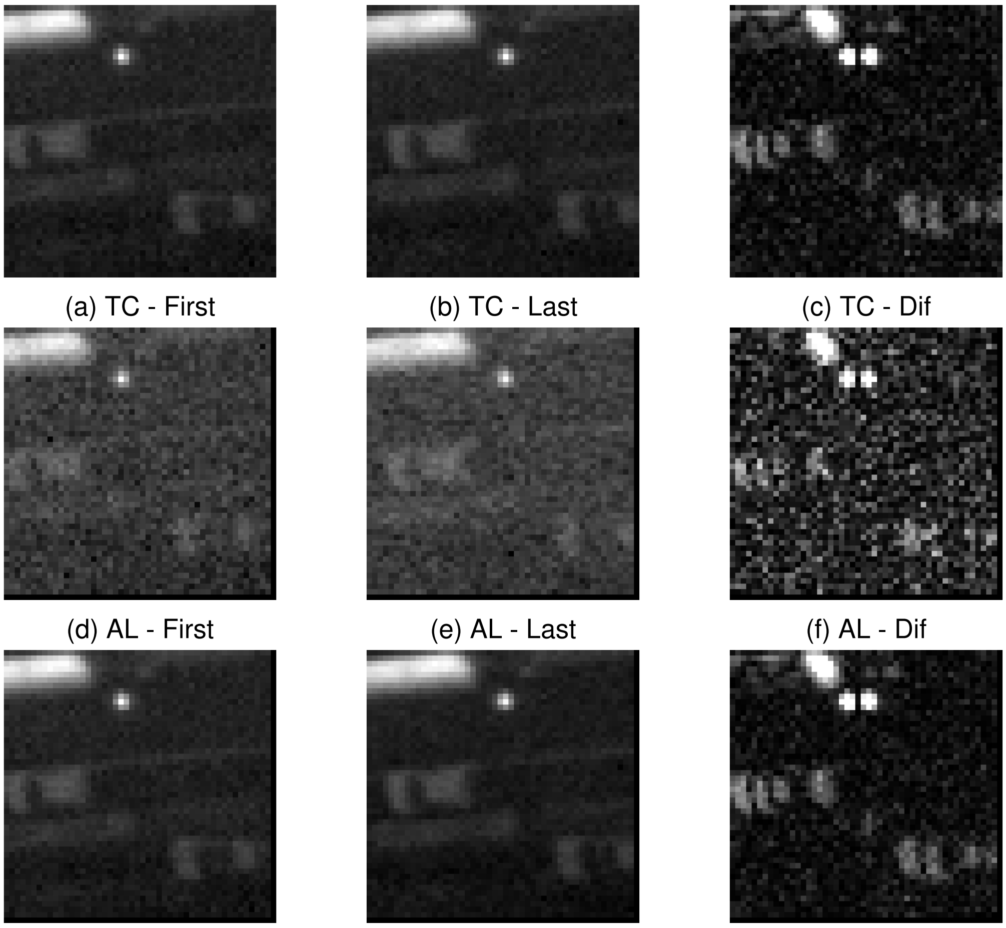

(a) BP - First (b) BP - Last (c) BP - Dif
(d) TC - First (e) TC - Last (f) TC - Dif
(g) AL - First (h) AL - Last (i) AL - Dif
(j) FI - First (k) FI - Last (l) FI - Dif

Figure 4.17 – Results for line SIMU20/P3/L0648. BP, before processing, TC, after temporal convolution, AL, after alignment, FI, temporal convolution applied after alignment.



(a) TC - First (b) TC - Last (c) TC - Dif

(d) AL - First (e) AL - Last (f) AL - Dif

(g) FI - First (h) FI - Last (i) FI - Dif

Figure 4.18 – Results for line SIMU20/P3/L0648 using $p = 4$ for temporal convolution. The registration failed because of its low SNR, as predicted by the SNR measurements for this line in Table 4.1. BP, before processing, TC, after temporal convolution, AL, after alignment, FI, temporal convolution applied after alignment.

4.5.4 Proposed Algorithms

In order to simplify the task, four possible algorithms are proposed based on the above results. Two fast methods, based on both algorithms 8 and 5, which aim at obtaining acceptable results while focusing on having a low computational cost, one medium method, which gives the best trade-off between precision and execution time, and a high precision algorithm, which is intended to achieve the best results. All algorithms systematically apply the Anscombe VST using the method described in Appendix B.

The first evaluated fast method, denoted Fast, is based on Algorithm 8, and uses the SNR verification described in Algorithm 9, just validating good SNR values in the first and the last frames of each line. The interpolation used is bicubic, which gives a good trade-off between precision and speed, and the $itQty$ parameter is set to 2.

The second fast algorithm presented, denoted Fast2, is based on Algorithm 5 using the second version of the $ComputeOffset$ method, described in Algorithm 7. The SNR validation is again performed using Algorithm 9. The interpolation method chosen is bicubic, however it is only used to generate the resulting video, since the number of iterations ($itQty$) is set to 1, implying no need to perform interpolation to calculate the global translation.

The third algorithm, denoted Medium, is based again on Algorithm 5 using Algorithm 7 to compute the global offset and verifying the SNR using Algorithm 9. The only difference between this algorithm and the previous one is that it does bias correction using $itQty = 2$ implying the necessity to interpolate intermediate results. To do this, bicubic interpolation is used. We found that this algorithm gives considerably better results while taking few computational resources.

Finally the fourth method named HighPrec, which is as well based on Algorithm 7 to compute the global offset, is the most expensive method presented. The SNR verification is performed on every frame of the line, and if any frame does not have enough SNR using a high enough value for parameter p, then the whole line is discarded. The number of iterations in this case is three, and cubic splines are used as the interpolation method, which was proven to give the best results.

The four proposed methods were analyzed and compared in order to obtain a clear idea of their performance. Their execution times using a non-optimized Matlab implementation on an Intel Core i7 computer are given in Table 4.13, in which we can appreciate the strong difference in cost between the fourth method and the rest.

Method 1 (Fast)	Method 2 (Fast2)	Method 3 (Medium)	Method 4 (HighPrec)
0.143s	0.152s	0.216s	0.530s

Table 4.13 – Average execution time, in seconds, for every proposed algorithm on every line.

After evaluating each method with every line in the dataset, Table 4.14 was obtained. The SNR threshold used for the first method was 20, while for the rest, a threshold of 10 had to be verified. A higher threshold was set for the first method because of its lower tolerance to noise which made the method to fail. Results show that the first method, indeed, has a good precision when there is enough information in the first and last frames of the line. However, in most situations, both medium and high precision methods not only are able to correctly estimate the displacement on more lines, but also obtain much better results.

An interesting measure is how each method performs under lines with varying SNR values. We combined both θ values into one using its Euclidean norm, that is $\sqrt{\theta_x^2 + \theta_y^2}$. Then, depending on this value, the precision of each method was studied. Since both provided resolutions have distinct SNR conditions, the comparison was made for each

Res.	Pert.	Method 1 (Fast)	Method 2 (Fast2)	Method 3 (Medium)	Method 4 (HighPrec)
20	P0	0.00293	0.00356	0.00469	0.00459
20	P1	0.00846	0.04702	0.01472	0.00688
20	P2	0.00829	0.01384	0.00725	0.00571
20	P3	0.00531	0.02193	0.00875	0.00786
40	P0	0.00253	0.00133	0.00180	0.00172
40	P1	0.02814	0.03759	0.00424	0.00296
40	P2	0.02918	0.04809	0.00824	0.00354
40	P3	0.05914	0.03560	0.00682	0.00483
Avg		0.01800	0.02612	0.00706	0.00476

Table 4.14 – Average error, in pixels, for the provided dataset for every proposed algorithm.

perturbation and for each resolution. Then, for each method, we set the minimum SNR threshold to be 2 (since a lower threshold simply makes no sense because there is no possible alignment in just random noise), and we evaluated the whole dataset.

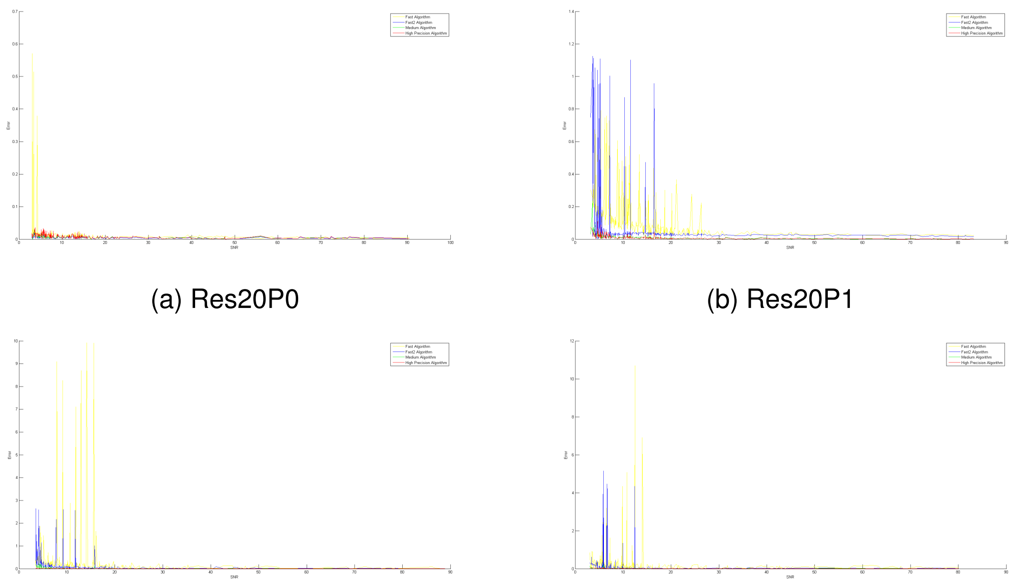
The obtained precision results are displayed in Table 4.15. In this table it can be seen how the Fast method in fact does not perform well in low SNR conditions, while both medium and high precision methods seem to keep obtaining good results. Interestingly, this experiment proves the importance of iterating the algorithm, since as can be inferred from the tables, under lower SNR scenarios, the gain obtained by performing more iterations becomes evident.

Res.	Pert.	Method 1 (Fast)	Method 2 (Fast2)	Method 3 (Medium)	Method 4 (HighPrec)
20	P0	0.01434	0.00667	0.00961	0.00612
20	P1	0.11981	0.09840	0.02485	0.00807
20	P2	0.28216	0.15589	0.02091	0.00839
20	P3	0.36107	0.17349	0.01945	0.00903
40	P0	0.00336	0.00136	0.00183	0.00172
40	P1	0.07907	0.04270	0.00859	0.00305
40	P2	0.10854	0.05106	0.01062	0.00359
40	P3	0.26639	0.07973	0.00816	0.00498
Avg		0.15434	0.07616	0.01300	0.00562

Table 4.15 – Average error, in pixels, for the provided dataset for every proposed algorithm using low SNR thresholds.

For resolution 20, the error of each algorithm is contrasted in Fig. 4.19, while the same is done for resolution 40 in Fig. 4.20. In every case, when dealing with low SNR values, the methods fluctuate more, however both fast methods presented cases in which they drastically fail to perform the registration or where a high error was obtained. When dealing with higher SNR values, the error for every method gets stable and the four algorithms usually converge. It can also be seen how the high precision method presented performs better than the rest on most cases. In fact, when there is no displacement between frames, as the case of perturbation zero (P0) in the dataset, every method performs well, and this is the only perturbation in which there is not a method that performs clearly better than the rest.

In order to better understand how each algorithm performs under varying SNRs, the accumulated error produced by each method for both resolutions is also presented in Fig. 4.21. From this figure, we can better see how each method performs in terms of precision, being the high precision algorithm the obvious winner of the lot, while the fast algorithms accumulating a high amount of error for low SNR values and then stabilizing to perform better. Again, as mentioned before, it has to be noted that when there is no displacement (P0), the best algorithm is not the fourth one, however the accumulated error for the whole dataset is considerably low, even taking into account low SNR lines. Finally, it has to be remarked that the difference in precision between both high precision algorithms is not justifiable, taking into account their difference in computational costs.

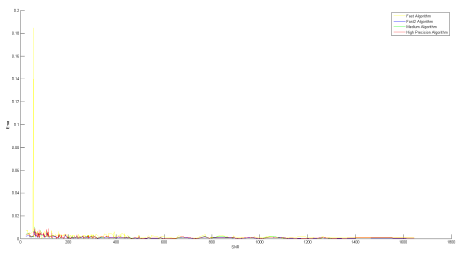


(a) Res20P0 (b) Res20P1

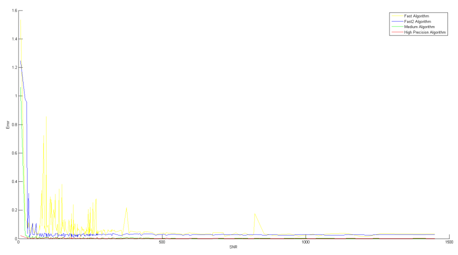
(c) Res20P2 (d) Res20P3

Figure 4.19 – Estimation error produced by each method on the provided Res 20 dataset, depending on the SNR of the line. Better visualization is achieved by zooming in on the digital document.

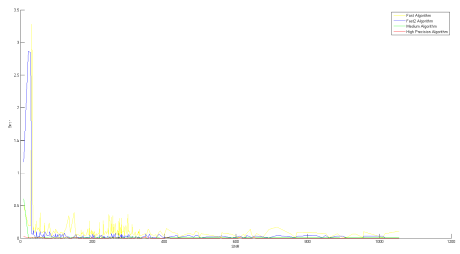
After observing these results, since there is a considerable difference in precision between fast methods and high precision ones, we compared them separately. However, this time, we decided to combine results of every perturbation type, for each resolution. Figure 4.22 shows the error and the accumulated error for both fast algorithms on both resolutions in the first two rows followed by both Medium and High Precision algorithms in the last two rows. In them, we can now appreciate how the precision of both fast and high precision algorithms are compared to each other, allowing us to select a good value for the SNR threshold. In the case of fast algorithms, the accumulated error, specially obtained when low SNR lines found, gets unacceptably high, suggesting that if any of these algorithms is used, the correct value for the threshold should be set to 20 for resolution 20 and around 250 for resolution 40. However, if the Fast2 algorithm is used, for resolution 40 a value of 70 for the threshold should be enough. In the case of High Precision algorithms, for both resolutions, a value of 20 for third algorithm and a value of 10 for the fourth one seems to be good.



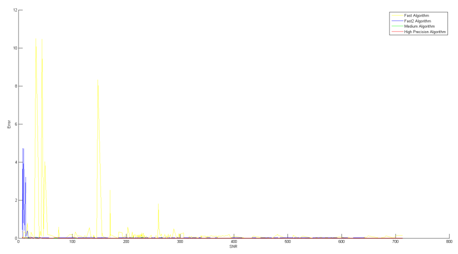
(a) Res40P0



(b) Res40P1

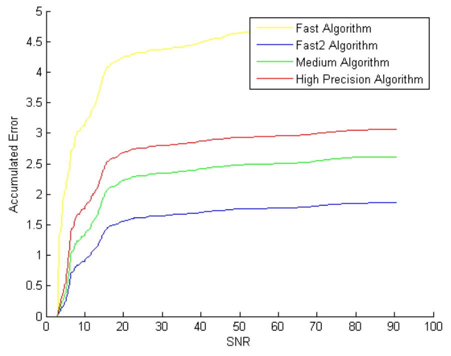


(c) Res40P2

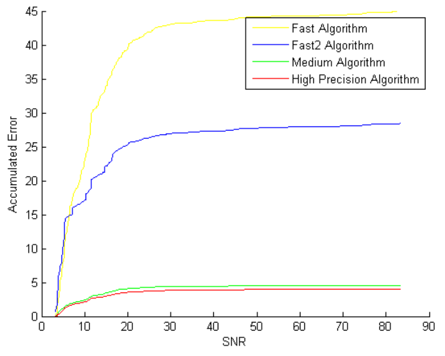


(d) Res40P3

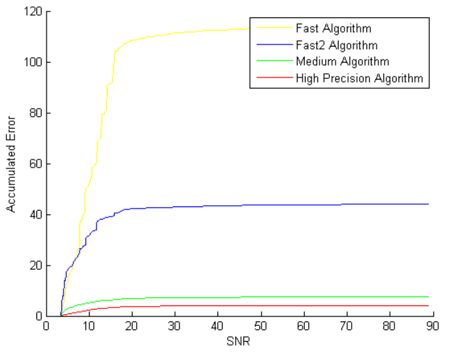
Figure 4.20 – Estimation error produced by each method on the provided Res 40 dataset, depending on the SNR of the line. Better visualization is achieved by zooming in on the digital document.



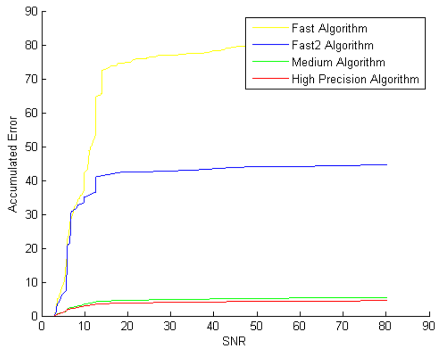
(a) Acum Res20P0



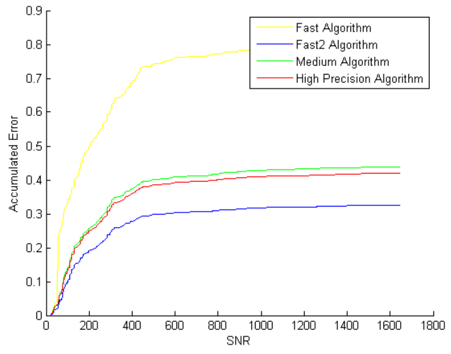
(b) Acum Res20P1



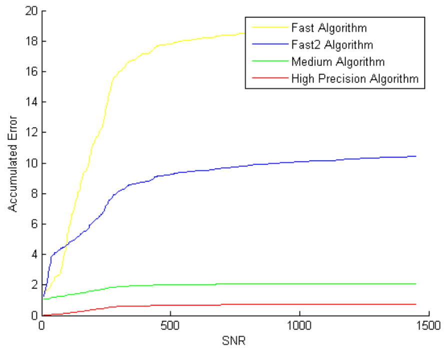
(c) Acum Res20P2



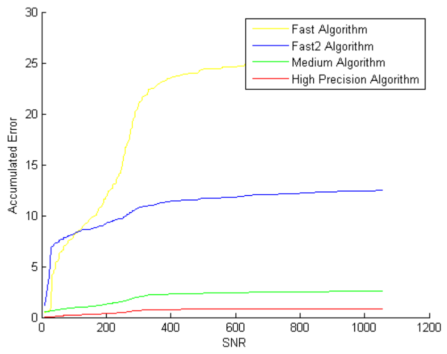
(d) Acum Res20P3



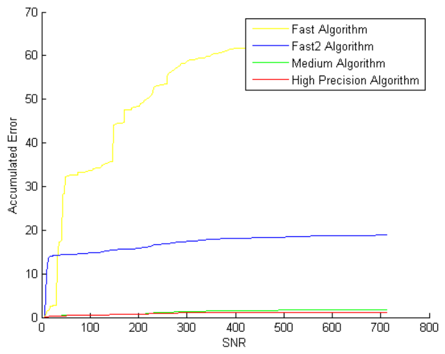
(e) Acum Res40P0



(f) Acum Res40P1

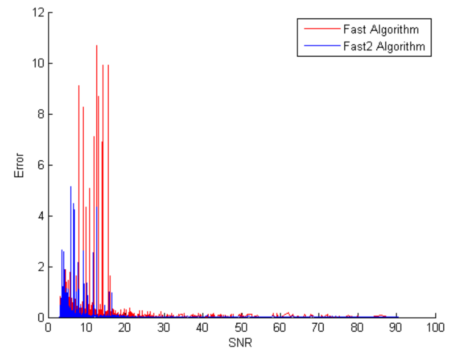


(g) Acum Res40P2

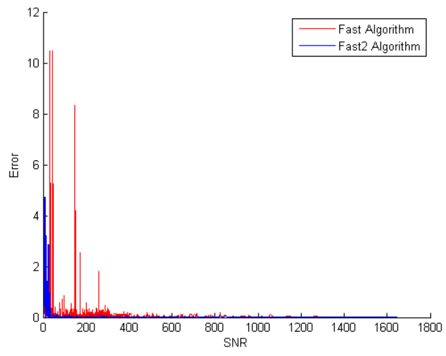


(h) Acum Res40P3

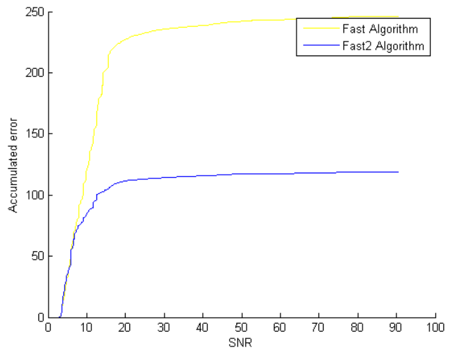
Figure 4.21 – Accumulated estimation error produced by each method on both provided datasets, depending on the SNR of the line.



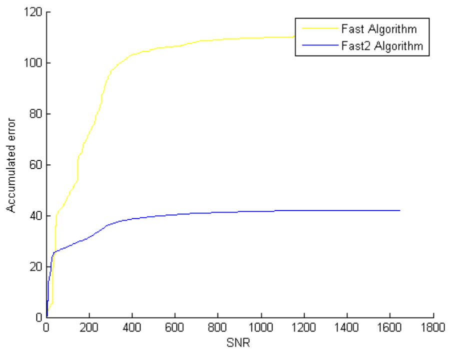
(a) Res20



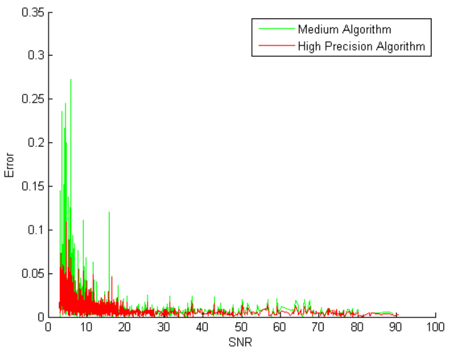
(b) Res40



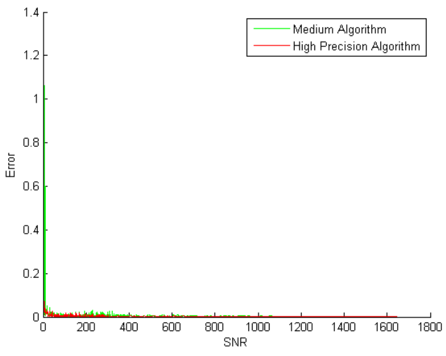
(c) Acum Res40



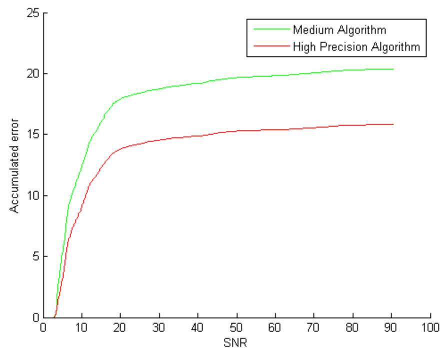
(d) Acum Res40



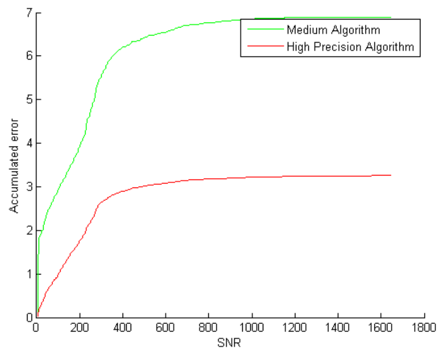
(e) Res20



(f) Res40



(g) Acum Res40



(h) Acum Res40

Figure 4.22 – Comparison of performance of both fast and high precision methods for each resolution, combining all perturbations. **Top two rows:** Fast methods. **Bottom two rows:** High precision methods.

4.6 Concluding Remarks

In this chapter we studied the problem of performing multi-image shift estimation based on a problematic posed by CNES (French space agency) which envisages to perform on board stabilization to reduce the influence of micro-vibrations on the satellite limiting the amount of accumulated frames in the TDI sensor. If fast and accurate shift estimation is proved feasible, then the amount of accumulated frames on the CMOS TDI device could be increased thus ameliorating the resulting SNR of the final image.

We began this study without any assumption on the displacements between the frames and with only the simulated dataset provided by CNES. By analyzing the obtained results, the first conclusion was the observation that the motion can be assumed to be linear on such a small time span. This fact, that has anyways obvious physical reasons, was not considered at the beginning of this study, and was later validated by CNES.

Our conclusions are:

- The best method in terms of velocity and accuracy proved to be the iterative gradient-based shift estimation method. It is the only adapted method for such small images and small motions, while instead Fourier-based methods (phase correlation, correlation peak detection, gradient correlation) fail, or are not implementable on board due to their higher computational costs.
- Performing temporal convolution before computing the shift estimation proved to be important, particularly under low SNR scenarios. Indeed, under linear motion, this temporal convolution does not affect the motion estimation, increases the SNR of the image sequence and decreases the influence of the other potential perturbations, like for example aliasing. All of these assumptions were confirmed by experiments.
- An automatic way to compute the number of frames needed for temporal convolution is proposed, based on obtained results. This algorithm depends on the SNR of the input line and on the desired thresholds.
- We proposed an alternative multi-image scheme particularly suited for the current problem, that improves over the naive sequential frame to frame shift estimation by reducing the influence of the bias of GBSE methods. The improvement obtained by using this approach was up to two orders of magnitude.
- We observed that some of the lines may contain only noise, and must therefore be automatically discarded. A fast way to discard them is proposed by computing a SNR through the image derivatives, which proved to be accurate enough for this objective.
- We have studied the lower achievable bounds on image registration for a pair of frames and expanded this concept for our particular case, when several images needs to be registered using a global displacement.
- The number of operations per pixel of the proposed algorithms are, depending on the parameters, between

  $$\mathbf{26+9*itQty+(itQty-1)*InterpCost} \text{ and } \mathbf{60+9*itQty+(itQty-1)*InterpCost}$$

  operations/pixel for the estimation of the global shift for a line, where $itQty$ is the amount of iterations used to estimate the shift between a pair of frames and $InterpCost$ is the cost of performing interpolation, if needed. If just a single iteration is used, then in total, between 35 and 69 operations/pixel are required, depending on the SNR of the line. If more iterations are performed, the interpolation cost is not negligible and should be taken into account. For example, for bicubic interpolation the approximate cost is 16 operations per pixel, nevertheless, adding a second iteration was able to reduce the bias significantly in some cases.

- It is crucial to decide if a line is reliable or not, and to estimate the error. We have fixed reasonable SNR thresholds that guarantee that the method works. Nevertheless, these thresholds give no guarantee in presence of an aperture problem, namely the presence of a very dominant gradient direction in the image. We found no such example in the provided dataset, but this may nevertheless happen and must be accounted for.
- We found reasonable (but perhaps not necessary) to apply to the images the Anscombe transform to make the noise nearly uniform before estimating the shifts. We observed that precision is decreased if no Anscombe transform is applied. We also studied how to apply the Anscombe transform using a computational trick that approximates the square root of a number without needing to calculate it. We found out that using this trick we are able to achieve similar results than with the normal square root.
- The precision of the method is increased if more than one iteration is applied in the calculation of the shift between two frames, however it implies performing interpolation for every iteration step. We observed, however, that the gains of applying an iterative version of the algorithm are much higher compared to the computational cost needed, if a simple interpolation is used, such as bicubic interpolation. Using a more complex interpolation method, such as spline interpolation, achieves improved results, however its increased execution time does not justify its use.
- The literature on the subject was thoroughly explored and we can reasonably sustain that the best method has been found and that the best way to estimate the error knowing the noise is also at hand.
- Four possible algorithms were proposed to the CNES as a result. These were designed based on the studied results and the CNES comments in order to be successfully implemented in the satellite. We described and implemented two low cost methods, a third one giving the best cost/performance relationship and a last one focusing on accuracy. A brief summary of them is shown in Table 4.16. The last two algorithms can successfully deal with more lines, without requiring to increase the SNR threshold parameters, thus being more tolerant to noise.
- Most importantly, the proposed algorithm is currently being used by the CNES in a on-ground demonstrator to register TDI images in real-time and is envisaged to be implemented in the next earth-observation satellite OTOS.

Method	Ops/Pixel	Avg. Error	Th.	Comments
Fast1	$< (2 + 2 * GP)$*	0.052 (from 1824 out of 2976 lines) (th=20)	20/250	Uses just first and last frames. Uses two iterations for shift estimation. Applies Anscombe transform. Validates SNR threshold on first two frames. Does bicubic interpolation for multi-scale.
Fast2	35–69	0.02 (from 1824 out of 2976 lines) (th=20)	20/70	Computes shift with respect to first frame. Uses one iteration for shift estimation. Applies Anscombe transform. Validates SNR threshold on first two frames. No interpolation needed.
Medium	BI+(44–78)	0.007 (from 2149 out of 2976 lines) (th=10)	10/20	Computes shift with respect to first frame. Uses two iterations for shift estimation. Applies Anscombe transform. Validates SNR threshold on first two frames. Bicubic interpolation used.
HighPrec	SI+(53–87)	0.005 (from 2365 out of 2976 lines) (th=5)	5/5	Computes shift with respect to first frame. Uses three iterations for shift estimation. Applies Anscombe transform. Validates SNR threshold on every frame. Spline interpolation used.

Table 4.16 – Summary of the proposed algorithms. The second column shows the range of required operations per pixel of the method, the third displays the average error measured on the provided dataset, the fourth column suggests convenient thresholds for the SNR on each resolution (20/40), and the fifth column displays key information about each algorithm. BI stands for Bicubic Interpolation cost, SI for Spline Interpolation cost and GP represents the cost of building the Gaussian Pyramid.
* The number of operations per pixel is smaller than two plus the cost of building the Gaussian pyramid for two images up to 4 resolutions. This implies performing interpolation and subsampling operations which makes the total execution time to be not considerably than the one from the Fast2 algorithm.

Chapter 5

RANSAAC: RANdom SAmple Aggregated Consensus

In the previous chapters, we studied the problem of fast and accurate subpixel shift estimation, focusing on two applications to be used on earth-observation satellites. Another frequent image registration problem is to find a model relating two images based on point matches between them. Indeed, accurate 2D transformation estimation through point matches between images is a well-known problem in computer vision. The difficulty coming from the presence of outliers and noisy measurements makes traditional regression methods usually fail. RANSAC discriminates outliers by randomly generating minimalistic sampled hypotheses and verifying their consensus over the input data. While discarding all other trials, it bases its response on a single iteration having the largest inlier support. In this chapter we show that the resulting accuracy can be improved by using all generated hypotheses. This yields RANSAAC, a framework that improves systematically over RANSAC and its state-of-the-art variants by statistically aggregating hypotheses. To this end, we also propose a simple strategy that allows to rapidly average 2D transformations, leading to an almost negligible extra computational cost. We give practical applications on both projective transforms as well as homography+distortion models and show the improvement in both cases.

5.1 Introduction

In the previous chapters of this thesis, we studied the problem of fast and accurate registration methods when the underlying transformation was a shift, and gave two example applications to be implemented on-board of earth-observation satellites. The presented approaches assumed small displacements between the input images. In this chapter we study the problem of image registration under more complex models. In particular, we focus on feature-based approaches using point matches, as mentioned in Chapter 2 of this thesis.

Several applications in computer vision such as image alignment, panoramic mosaics, 3D reconstruction, motion tracking, object recognition, among others, are performed by feature-based approaches. First, characteristic image features are detected on the input images. These features should meet some repeatability criteria to ensure they can be detected on images, regardless of their location, pose, illumination conditions, scale, etc. They could be points with some special characteristic or distinguished image regions. Undoubtedly, SIFT points [96] and MSERs [105,106] are the most recognized feature detectors of each category respectively. The second step is to assign to each feature a unique description. In the case of points, this is done by describing its surroundings [16,96,110]. For regions, their silhouettes and the contained texture information are employed [18,58]. Finally, the descriptors of both images are matched to compute putative matches between them. Ideally, these matches correspond to samples of the underlying model to be estimated. Unluckily, not all detected matches are faithful to the global model, yielding false “clues” to the transformation estimation method.

As explained above, when estimating a single global transformation based on multiple point matches between images, existing methods have to deal with the problem of outliers, i.e., incorrectly detected matches that are not represented by the transformation. Robust to outliers, the Random Sample Consensus (RANSAC) is an iterative method introduced by Fischler and Bolles [55], widely used in computer vision to simultaneously solve the correspondence problem while estimating the implicit global transformation. By randomly generating hypotheses on the transform matching the points, it tries to achieve a maximum consensus in the input dataset in order to deduce the matching points belonging to the transformation, usually called inliers. Once the inliers are discriminated, the parameters of the underlying transformation are usually estimated using a regression technique on the inliers. RANSAC achieves good accuracy when observations are noiseless, even with a significant number of outliers in the input data.

Instead of using every sample in the dataset to perform the estimation as in traditional regression techniques, RANSAC tests in turn many random sets of sample pairs. Since the probability of picking an outlier increases exponentially with the size of the sample, RANSAC takes the minimum sample size (MSS) m to determine a unique candidate transform, thus incrementing its chances of finding an "all-inlier" sample, i.e., an uncontaminated sample exclusively composed of inliers. This transform is assigned a score based on the cardinality of its consensus set. Finally the method could return the hypothesis that achieved the highest consensus. However, as suggested by the authors, a last-step minimization is performed taking its inliers to compute the parameters of the transformation.

More formally, given a transformation ϕ with parameters θ and a dataset of pairs X_i, Y_i consisting of $i = 1, \dots, N$ samples, RANSAC computes

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \rho\Big(dist(\phi_\theta(X_i), Y_i)\Big), \tag{5.1}$$

where $dist$ is usually the squared L_2 distance given by $dist(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$ and the cost function ρ is defined as

$$\rho(e) = \begin{cases} 1 & \text{if } e \leq \delta_d \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

The parameter δ_d is a critical parameter of the algorithm. It is set proportional to the assumed measurement noise of the input matches with different factors depending on the type of estimation problem [70]. Finally, the optional last-step minimization refines the previous result by computing

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \left\| \phi_\theta(\tilde{X}_i) - \tilde{Y}_i \right\|^2 \tag{5.3}$$

where the inlier matches $(\tilde{X}_i, \tilde{Y}_i)$ are defined by

$$\{(\tilde{X}_i, \tilde{Y}_i) \mid \tilde{X}_i \in X, \tilde{Y}_i \in Y, \rho(dist(\phi_{\hat{\theta}}(\tilde{X}_i), \tilde{Y}_i)) = 1\}. \tag{5.4}$$

5.1.1 RANSAC iterations and model error

It was shown in [55] that by drawing k samples, with

$$k \geq \frac{\log(1 - \eta_0)}{\log(1 - \epsilon^m)}, \tag{5.5}$$

where m is the MSS, one can ensure with confidence η_0 that at least one outlier-free sample is obtained from a dataset having inlier rate ϵ. Therefore, the total number of iterations could be set according to Eq. (5.5). However, since the inlier rate ϵ is in general not known beforehand, the number of iterations cannot be defined *a priori*. Nevertheless, this value could be updated every time a better hypothesis is found, by using Eq. (5.5) and taking ϵ as the current inlier ratio, given by the ratio between the amount of inliers and the total amount of matches.

However, as noted by Chum *et al.* [39], the number of iterations of Eq. (5.5) is overly optimistic, so that an outlier-free sample does not guarantee an accurate estimation either. This is due not only to the noise on the position of the input points (measurement noise), but also to the underlying "noise" on the model hypotheses themselves [127, 171]. This other "noise" is caused by several factors such as limited numerical precision, poor conditioning of the model estimation method, or by other sometimes non-avoidable reasons such as the rounding of pixel intensities or point coordinates. In practice, the number of samples required in RANSAC should be set typically to two or three times the theoretical number [129].

5.1.2 Distance parameter

The parameter δ_d depends on the measurement noise of the m input samples. Assume that these are 2D points with Gaussian noise of zero mean and standard deviation σ. Then their error, computed as $d(\mathbf{x}'_i, H\mathbf{x}_i)^2$, i.e. the distance between the projected point on the first image onto the second image and its match, is the sum of squared Gaussian variables which thus follows a χ^2_m distribution with m degrees of freedom. The probability that this error is lower than a given value k can be computed from the cumulative χ^2_m distribution F_m. Capturing a fraction α of the inliers is ensured by setting $\delta_d = F_m^{-1}(\alpha)\sigma^2$, which means that a true inlier will be incorrectly rejected $1 - \alpha$ percent of the time. Then if the error of the estimated transformation H for a particular set of matches $(\mathbf{x}, \mathbf{x}')$ is measured as the direct transfer error, given by

$$Err(H, \mathbf{x}, \mathbf{x}') := \sum_i d(\mathbf{x}'_i, H\mathbf{x}_i)^2, \tag{5.6}$$

this implies setting $\delta_d = 5.99\sigma^2$ for $\alpha = 0.95$ [70]. However, if the error is given by the symmetric transfer error, namely

$$Err(H, \mathbf{x}, \mathbf{x}') := \sum_i d(\mathbf{x}'_i, H\mathbf{x}_i)^2 + d(\mathbf{x}_i, H^{-1}\mathbf{x}'_i)^2 \tag{5.7}$$

then the χ^2_m distribution has 4 degrees of freedom (i.e. $m = 4$). Therefore, one must set $\delta_d = 9.4877\sigma^2$ for $\alpha = 0.95$ or $\delta_d = 13.2767\sigma^2$ for $\alpha = 0.99$. However, since in practice, the standard deviation of the noise is not available, this parameter is usually set by experimentation. Throughout this chapter, unless otherwise specified, this parameter was always set using $\alpha = 0.99$ and assuming the error measured as in Eq. (5.7).

5.1.3 Final refinement step

As already mentioned, RANSAC reduces the effect of measurement noise by re-estimating the transform using a last-step minimization on the resulting inliers. Depending on the underlying model, both linear and non-linear minimization methods exist [23, 70], with the latter one usually achieving better results with a higher computational cost. Although this final stage does in general improve RANSAC results, it gives equal importance to each match for the minimization. Then if RANSAC is not able to completely discriminate the outliers within the input data, this introduces a bias in the computation commonly resulting in a drop in performance.

Throughout this chapter, we will refer to RANSAC as the method described by Eq. (5.1) without performing the last refinement step, and RANSAC+M or more directly RANSAC with last-step minimization to the full approach following Eq. (5.3).

5.1.4 RANSAC weaknesses

RANSAC has several drawbacks. First, the probability of RANSAC obtaining a reasonable result increases with the number of iterations, however it may never reach the optimal solution. What is more, RANSAC results have a high degree of variability for the same input data, and this variability increases with the amount of input points and their measurement noise. Second, although robust to outliers, RANSAC is not particularly immune to measurement noise on the input data, as illustrated on Fig. 5.1. Furthermore, the optional final minimization step weighs uniformly the assumed inlier match, ignoring wether they are real inliers or how strongly they may be affected by noise. Third, the maximum tolerable distance parameter δ_d should usually be sufficiently tight to obtain a precise transformation, however it must also be loose to find enough input samples [33]. Because of such considerations, setting this parameter is a difficult task, even under low measurement noise. Finally and most importantly, **the accuracy of RANSAC is based on the single iteration where the best model was found**. Although this may be accurate in some cases, it nevertheless discards other good all-inlier models that may have been generated throughout the iterations.

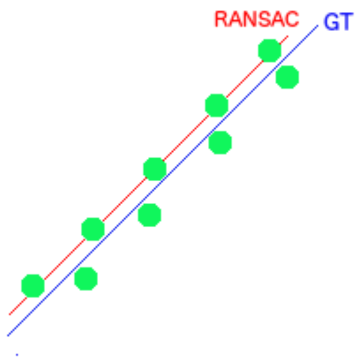


Figure 5.1 – Toy example where RANSAC is not able to detect the correct model due to measurement noise. If the distance parameter is set too small, RANSAC will adjust its model to the noisy estimates, therefore fitting the line on the upper five points (in red), whereas the ground truth line passed through the middle (in blue).

To make up for this, we present Random Sample Aggregated Consensus (RANSAAC), a simple yet powerful method combining the random sample consensus scheme with a statistical approach. By aggregating the random hypotheses using their consensus set cardinalities, the proposed approach improves systematically on RANSAC. We give practical implementations of this idea on 2D parametric transformation models, by proposing a simple strategy to drastically accelerate the computations. This allows to rapidly aggregate 2D parametric transformations using negligible computational resources.

Its main benefits over traditional RANSAC are:

- Results are both more accurate and with less variability (in terms of standard deviation of the error). The improvement over the traditional RANSAC approach is on average by a factor of two to three, and is even more important with higher noise, more inliers available, and higher outlier ratios. Moreover, it improves systematically over other state-of-the-art RANSAC extensions.
- The accuracy improvement persists after including the final regression step in RANSAC and its variants, even without using a last-step optimization in RANSAAC.

- As with the original RANSAC method, the accuracy is dramatically improved by adding a local optimization step, which in fact seems suited for our approach because it could avoid discarding the generated intermediate models.
- Also, by including this step, the theoretical adaptative stopping criterion signaled by Eq. (5.5) becomes more realistic and could be used to effectively stop the iterations without affecting the final accuracy.
- By using the proposed 2D transformation averaging method, the extra computational cost is almost negligible.
- Robustness against measurement noise is drastically improved for the case of noise distributions with symmetrical pdfs.

The rest of this chapter is organized as follows. We begin by giving a review on RANSAC variants in section 5.2. In section 5.3 we detail the proposed algorithm. We thoroughly evaluate it in section 5.4 to finally conclude and give some remarks regarding the future work in section 5.5.

5.2 RANSAC Alternatives or Improvements

One of the aims of robust statistics is to reduce the impact of outliers. In the field of computer vision, a robust estimator must be able to correctly distinguish the true information in the case where the outliers and the noise occupy a high percentage of the data. In the context of regression analysis, the Least-Median-of-Squares (LMedS) and the Least Trimmed Squares (LTS) are well-known classical methods and still widely used. Both approaches resemble RANSAC by generating hypotheses from minimalistic samples. However, after computing the residuals for each data point, instead of discarding the outliers, both methods sort the input points based on their residual error. The LMedS then returns the hypothesis for which the residual of the median point on this ordering is the lowest. The LTS method returns instead the model for which the sum of a fixed percentage of best ranked input data is minimized [143]. Although the LMeds method does not require to specify the δ_d parameter of RANSAC, it implicitly assumes at least 50% of inliers in the dataset. This method was successfully used to estimate the epipolar geometry between two images [190].

Since its introduction and due to the massive proliferation of panorama generation [25], 3D estimation [70] and invariant feature points [96, 136], several modifications of RANSAC were proposed [34, 127, 128], mainly addressing three major aspects: computational cost, accuracy and robustness against degenerate cases, i.e., cases where it is not possible to recognize an incorrect hypothesis by simply scoring the drawn minimal sample. Interestingly, these techniques can also be grouped based on the strategy they adopt.

One of such strategies is to modify the random sampling process of RANSAC to generate better hypotheses, improving both speed and accuracy. In this group, NAPSAC [115] segments the sample data by assuming that inliers tend to be spatially closer to one another than outliers, thus picking points laying within a hypersphere of a specific radius. The PROSAC algorithm [36] exploits the ordering structure of the set of tentative correspondences and bases its sampling on the similarity computed on local descriptors. GroupSAC [117] combines both strategies by segmenting the input data based on feature characteristics.

Another strategy is to perform a simple test by partially evaluating the dataset to discard hypotheses before computing the score, thus reducing the overall computational cost. The $T_{d,d}$ test [104] checks whether d randomly drawn points are inliers to the hypothesis, discarding it otherwise. In [37], the authors propose to use a sequential probability ratio test (SPRT) to discard "bad" hypotheses, based on Wald's theory of sequential testing [183]. Finally, instead of evaluating one model at a time, Preemptive RANSAC [118] filters hypotheses by scoring several of them in parallel over the same subset of samples, permitting real-time applications.

To improve on accuracy, instead of computing the cardinality of the set of sample points having smaller residuals than a specified threshold, several methods modify how samples are scored by weighting inliers based on their residual error. The MSAC method [172], a simple redescending M-estimator [79], takes the maximum between the error and the threshold to score each datum to finally minimize over their sum. Conversely, MLESAC [173] maximizes the log likelihood of the given data by assuming the inlier error has Gaussian distribution while the outlier error follows an uniform law.

One strategy used to improve both accuracy and speed is to perform a local optimization step after a "good" enough model has been found. The locally optimized RANSAC (LO-RANSAC) [39], as well as its variants [88,129], performs a fixed amount of RANSAC iterations by taking non-minimal samples of the inliers of each newly found best hypothesis (approx. 12/14 matches for an homography), in a so-called *inner*-RANSAC procedure. For each model generated by taking a non-minimal sample, a larger distance threshold is used to compute its inliers and an iterative procedure is performed to refine this model by progressively shrinking the threshold. The recently published Optimal RANSAC method by Hast *et al.* [71] also exploits this technique, however they add small modifications such as dataset pruning to remove outliers among iterations along with adaptive termination of the method when the same inlier set is found again after the *inner*-RANSAC procedure.

To increase the robustness of RANSAC against degenerate data in an epipolar geometry estimation context, DEGENSAC [38] performs a test aimed at identifying hypotheses where five or more correspondences in the minimal sample are related by a homography, to avoid bad solutions on images containing a dominant plane in the scene. In cases where most of the data samples are degenerate and do not provide a unique solution, QDEGSAC [60] performs several RANSAC executions, iteratively, by adding constraints to the inliers of the previous results.

Based on these approaches, Raguram *et al.* [127] studied several RANSAC extensions and proposed an algorithm that intelligently combined them into a single modular method called Universal RANSAC (USAC). This method obtains state-of-the-art results using few computational resources. It does so by combining ideas from LO-RANSAC, DEGENSAC, the SPRT test and PROSAC.

More recently, the community has been focusing towards two distinct objectives. The work of Suter and his collaborators [32,121,164] aims at improving the sampling process. By taking random non-minimal subsets of samples, they show it considerably reduces the bias caused by taking minimal size samples over noisy input data [121,164]. Another approach [32], also used in [99], is to perform an initial, not so extense round of random samples together with their hypotheses to build some prior knowledge, followed by guiding the subsequent sampling using the residuals of these hypotheses, i.e. the residual sorting information.

A second recent trend is to approach the robust estimation problem in an inverse fashion [98,99,165,169]. Instead of searching for model parameters that minimize a predefined residual function, the idea is to use, as in the guided sampling context, the residuals for each data point obtained from several random models, in order to distinguish between inliers and outliers in a clustering procedure. To this end, these methods build a matrix where its columns are the input points and its rows represent each sampled hypothesis, and that each value is the evaluation of each data point (column) on the hypothesis (row). This matrix is usually referred to as the preference matrix. Then by reordering both the rows and the columns in order to place correct model hypotheses together with the data points belonging to this model, the preference matrix will have a block-diagonal structure, where each block is composed by inliers of a specific model in the input data.

The methods found in the literature mainly differ in how to sample, define and cluster this matrix. While J-Linkage [169] uses a binary method to define the relationship between each data point and each hypothesis in the matrix, T-Linkage [98] uses a continuous thresholded value, conceptually similar to the difference between RANSAC and MSAC scores. To perform the clustering, both methods rely on an agglomerative clustering technique, proceeding in a bottom-up manner: starting from all singletons, each iteration of the algorithm merges the two clusters with the smallest distance. J-Linkage uses the Jaccard distance metric while T-Linkage relies on the Tanimoto distance (hence both names J-Linkage and T-Linkage).

The approach of Tepper and Sapiro [165] performs bi-clustering on the preference matrix by using non-negative matrix factorization (NMF). An interesting advantage of this technique is that an object is allowed to belong to multiple bi-clusters, which may occur when some data points belong to two different models, such as the intersecting points of two planes. To accelerate the method, they add an *a-contrario* statistical approach to discard "bad" hypothesis before computing the bi-clustering. A similar approach is used in [99], where they combine Robust PCA with Symmetric NMF to cluster the preference matrix. It should be noted that, opposite to RANSAC where a single model is estimated, these approaches target multimodel estimation and therefore are not the focus of the present work.

Finally, two techniques address one of the major problems of RANSAC: the selection of the δ_d parameter. ORSA [111] introduces an *a contrario* criterion to avoid setting hard thresholds for inlier/outlier discrimination. On the other hand, Raguram [129] estimates the covariance of the underlying transform and propagates this error due to the noise by modifying the regions for which each data point is considered an inlier.

5.3 Random Sample Aggregated Consensus

As explained above, the traditional RANSAC algorithm as well as its extensions discard potentially useful hypotheses entirely basing its results on a randomly generated hypothesis that better fits the data. What is more, due to the underlying measurement noise, there may be no hypothesis that, by randomly drawing points over the input data, would generate the correct model, therefore justifying the use of least squares regression over the (tentative) inliers [39,127]. However, hypotheses that obtained lower scores than the best, and were therefore discarded, could potentially be useful to obtain a more precise estimation of the underlying model. In fact, if hypotheses generated by all-inlier noisy samples are similar enough, i.e., their variance is bounded, then it makes sense to aggregate them to improve the accuracy of the method. The idea behind the proposed approach is to take advantage of every generated hypothesis, using its score as a confidence measure that allows not only to discard outliers but also to give more importance to better models. To successfully apply the proposed approach, one requirement should be met: the possibility to perform operations directly on the models, such as averaging, computing the geometric median or being able to estimate geometrically meaningful distances between different models. These models could be directly aggregated in their manifold, by understanding its geodesics [94]. However, for the case of 2D transformations mapping points in one image to points on another, instead of aggregating on the topological space, a simple strategy based on pre-selecting points could be used virtually adding no cost to the overall computation.

5.3.1 Aggregation of 2D parametric transformations

Take, for example, the task of averaging homographic transformations. First, notice that the set of all homographies is not a vector space, so that a convenient notion of nonlinear averaging is required. By selecting a proper parametrization, it is possible to define the average, however the result will depend on the choice of the parametrization selected. For example, all the following parametrizations are common to homographies:

1. A 3×3 matrix (represents all homographies, uses 9 parameters).
2. A 3×3 matrix normalized so that a chosen entry is 1 (represents some homographies using 8 params).
3. A 3×3 matrix normalized to have norm 1 (represents all homographies using 9 params).
4. The displacement vectors of 4 previously chosen points on the 4 image corners (represents all homographies using 8 params).

Notice that parametrizations (2) and (3) are problematic. Parametrization (2) is not able to represent all homographies, and the values of parametrization (3) are not a vector space themselves. Between (1) and (4), we contend that it is better to use (4) because only in (4) all parameters are measured with the same units. While the average of matrix representations has no direct geometric interpretation, independently averaging point locations has a direct interpretation.

In the context of RANSAC, we shall first illustrate this idea through a simple but classic example. Assume again that two images are related by a homography, each with some detected feature points and a certain amount of matching pairs between both images. Let us consider a single feature $\mathbf{x}_0$ from the first image that has its corresponding point $\mathbf{y}_0$ on the second image (the match $(\mathbf{x}_0, \mathbf{y}_0)$ does not necessarily need to be a detected match). At each iteration RANSAC randomly picks matches, generates a hypothesized homography H_k and computes its score using the whole dataset. This hypothesis in fact yields a tentative match $\hat{\mathbf{y}}_\mathbf{0} = H_k(\mathbf{x}_0)$. If the matches randomly selected were all inliers, then, with high probability, $\hat{\mathbf{y}}_\mathbf{0}$ will be close to the real matching point $\mathbf{y}_\mathbf{0}$ on the second image. Thus, after several lucky all-inlier iterations, many estimates close to $\mathbf{y}_\mathbf{0}$ are actually found.

All of these estimates are affected by two noise sources: the underlying "noise" of the model hypothesis as explained before, and the measurement noise of the input points. Provided both of them have zero mean and a symmetric distribution function, aggregating these estimates should yield values closer to the true point $\mathbf{y}_\mathbf{0}$. However, since there may be outliers in the input samples, it is better to assign a weight for each generated hypothesis reflecting its reliability. Luckily, RANSAC already computes this measure, which is its so-called score. Therefore, by weighting each of the resulting estimated points using the RANSAC score and aggregating them by, for example, taking their weighted mean or weighted geometrical median, the projected point will, with a high probability, result closer to its true value $\mathbf{y}_\mathbf{0}$.

When estimating a 2D transformation mapping points from one image into another, all the information to be transformed lies within the first image, i.e., it is not important what occurs to pixels lying outside it since they are not available. Although this may seem obvious, it permits to pre-select points that can fully determine the extent of the underlying transformation, minimizing the errors caused by quantification and the so-called model "noise". By using these points, it is possible to generate different estimates for every hypothesis. These estimates could also be weighted by how "good" the hypothesis is, i.e., the RANSAC score.

More formally, let X_i, Y_i be matching points where $X_i \in \mathcal{C}_1$, $Y_i \in \mathcal{C}_2$, $i \in 1, \ldots, N$, $\mathcal{C}_1, \mathcal{C}_2 \subset \mathcal{R}^2$ and $x_j \in \mathcal{C}_1$, $j \in 1, \ldots, n$ with $n \ll N$ predefined points on the first image. For each iteration k with hypothesis $\mathcal{H}_k = \phi_{\theta_k}$ RANSAAC computes both the projected predefined points

$$\hat{y}_j^k = \phi_{\theta_k}(x_j) \text{ for } 1 \leq k \leq n \tag{5.8}$$

and the RANSAC score

$$w_k = \sum_i^N \rho\Big(dist(\phi_{\theta_k}(X_i), Y_i)\Big), \tag{5.9}$$

where $dist$ is usually the squared L_2 distance $dist(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$ and ρ is the cost function. After K iterations, for each pre-selected point x_j, RANSAAC aggregates the different estimations $\hat{y}_j^k$ using the computed weights w_k, resulting in trustworthy matches $(x_j, \hat{y}_j)$

$$\hat{y}_j = aggregate(\hat{y}_j^i, w_i) \tag{5.10}$$

with $i \in \{1, \ldots, K\}$. This procedure, in fact, could be regarded as a way to generate "denoised" inlier matches, since x_j could be any point on the first image, while $\hat{y}_j$ will be placed, with a high probability, close to the true matching point y_j.

The *RANSAAC* algorithm adds a few lines to the traditional RANSAC method and an almost negligible computational cost. In Algorithm 10, the traditional RANSAC method is shown with yellow background. As seen within the lines with white background, RANSAAC first verifies for each iteration if the hypothesis is acceptable, and then applies it to the source points while storing its scores as weights. Finally, after iterating, if no acceptable transform was found, it returns the traditional RANSAC result. However, if several hypotheses were considered valid (i.e. had more inliers than the $minValidScore$ parameter), the computed projected points are aggregated using their weights and the algorithm returns the model that best fits these estimates.

Predefined Source Points

The source points that are a parameter for the method obviously depend on the global transform being estimated. The cardinality of this basis should be at least equal to the minimum number m of matches required to generate a single hypothesis, also known as the minimum sample size (MSS). We simply pick points which can uniquely determine the desired transform and minimize the noise inferred by model generation, as depicted in Fig. 5.2. Any model that could be correctly estimated by sampling points on its extents is a candidate model for which this strategy could be applied. For a homography, for example, the four points on the corners of the image should be selected. For a transformation involving a homography and two radial distortion parameters that requires either 5 or 6 matches to be computed, such as the one recently proposed by Kukelova et al. [87], a point on the center of the top and/or on the left image border should be added.

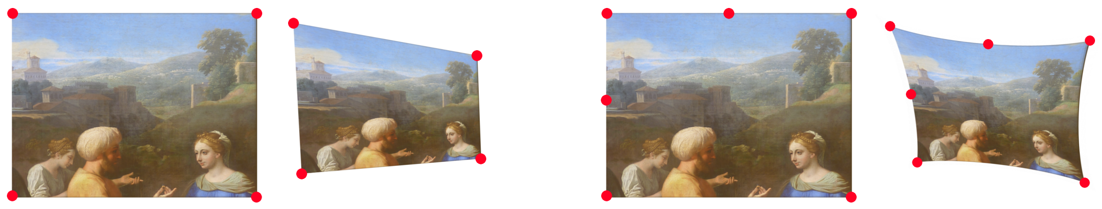


Figure 5.2 – Suggested predefined source points (in red). Left: for homography estimation. Right: for homography + distortion.

Algorithm 10 Computing a 2D transformation using the RANSAAC algorithm

Require: $X_1, X_2 \in \mathbb{N}^2 \times N$: N matching points, $minSamples \in \mathbb{N}$: minimum amount of matches to compute the transform, $srcPts \in \mathbb{N}^2 \times K$: vector with $K = minSamples$ input points, $\delta_d \in \mathbb{R}$: RANSAC distance threshold, $minValidScore \in \mathbb{N}$: minimum score to consider an iteration valid, $iters \in \mathbb{N}$: amount of iterations, $p \in \mathbb{R}$: aggregation parameter.

```
1:  $maxScore \leftarrow 0$
2:  for $it = 1$ to $iters$ do
3:      $Xss_1, Xss_2 \leftarrow GetRandomSample(X_1, X_2, minSamples)$
4:      $h \leftarrow GenerateHypothesis(Xss_1, Xss_2)$
5:      $inliers \leftarrow EvaluateHypothesis(h, X_1, X_2, \delta_d)$
6:      if $\#inliers > maxScore$ then
7:          $maxScore \leftarrow \#inliers$
8:          $ransacRes \leftarrow h$
9:      end if
10:     if $score > minValidScore$ then
11:         $Add(dstPts, Project(srcPts, h))$
12:         $Add(weights, \#inliers)$
13:     end if
14: end for
15: if $\#dstPts > 0$ then
16:     $resPts \leftarrow Aggregate(dstPts, weights, p)$ // See 5.3.2
17:     return $GenerateHypothesis(srcPts, resPts)$
18: else
19:     return $ransacRes$
20: end if
```

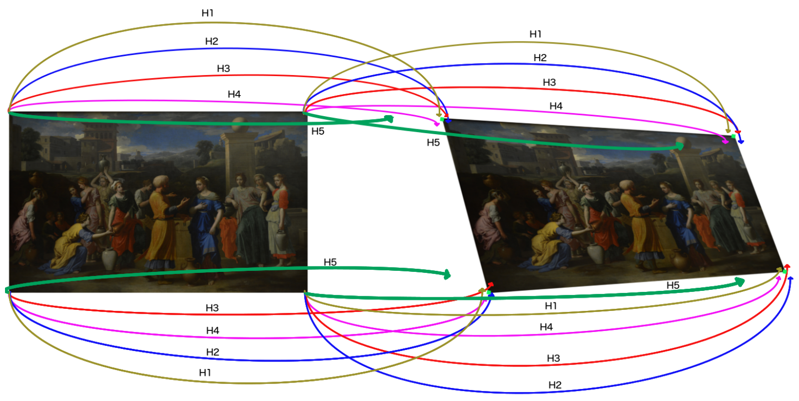


Figure 5.3 – Two images related by a homography. Each of the points on the corner of the first image are evaluated with 5 hypotheses. These destination points are aggregated to compute the final transform.

Note that since the final aggregated points should be all inliers to the model, instead of considering a minimal set of points for characterizing the final transform, it is possible to sample more points. However, this strategy requires more per-iteration computational resources therefore it is discarded.

5.3.2 Aggregation of Estimates

For the case of 2D parametric transformations, instead of simply returning the best hypothesis, RANSAAC gathers, for each point of the predefined basis, a set of estimates of its location on the second image together with a weight computed from the attained RANSAC score of the underlying hypothesis that generated each estimate. These form a weighted cluster, each point being associated the score of the hypothesis from which it was derived. All of these estimates can then be aggregated to compute a more accurate location for each point of the basis. This yields a minimum sample that permits to recover the transform. In the example shown in Fig. 5.3, five possible hypotheses are shown. Since the fifth hypothesis H_5 (green arrow) obtained a lower score, it has less influence on the resulting aggregation.

Two aggregation techniques were considered, namely the weighted mean and the weighted geometric median [45]. To aggregate the points x_i in the set $\mathcal{C}$, with weights w_i, with $i \in \{1, \dots, |\mathcal{C}|\}$, the weighted mean computes

$$wmean(\mathcal{C}) = \left(\sum_{i=1}^{|\mathcal{C}|} w_i^p x_i \right) \Bigg/ \left(\sum_{i=1}^{|\mathcal{C}|} w_i^p \right), \tag{5.11}$$

where p is a parameter of the aggregation procedure.

The weighted geometric median is defined as the point in the set achieving the minimum sum of distances to the others, namely

$$wgmed(\mathcal{C}) = \underset{y \in C}{arg\,min} \sum_{i=1}^{|\mathcal{C}|} w_i^p \left\| x_i - y \right\|_2 . \tag{5.12}$$

It can be calculated using the Weiszfeld's algorithm [187], which is a form of iteratively re-weighted least squares [41] which iteratively computes

$$y_{i+1} = \left(\sum_{j=1}^{|\mathcal{C}|} \frac{w_j^p x_j}{\|x_j - y_i\|} \right) \Bigg/ \left(\sum_{j=1}^{|\mathcal{C}|} \frac{w_j^p}{\|x_j - y_i\|} \right), \tag{5.13}$$

where again p is a parameter of the method.

By using weighted aggregation schemes based on the RANSAC score, the $minValueScore$ parameter becomes less important, and setting it as $minValueScore = minSamples$ proved to be a good choice for projective transformations.

5.3.3 Local Optimization and its variants

One of the major improvements of the original RANSAC method, in terms of accuracy, is to include an intermediate step every time a new "best" hypothesis has been found. This step, known as the local optimization step, achieves improved results by using the fact that models with "good enough" inlier supports are probably not distant from the true underlying model. Therefore, by taking non-minimal samples from the inliers of this new "best" model followed by a greedy strategy to refine the results, the LO-RANSAC algorithm [39] and its variants [88, 127] improve systematically over RANSAC.

The local optimization step is shown in Algorithm 11. The method begins by taking non-minimal samples of the inliers of the new best hypothesis found, and computes the model by performing a least squares fit ($GetModelLS$ method). Then, it computes the inliers of this model for the whole dataset using a larger distance threshold ($\delta_d \cdot m_{\delta_d}$) and uses these inliers to calculate a second model by least squares fitting. This second model is later refined in an iterative re-weighted least squares fashion, by progressively shrinking the distance threshold until its original value is met. The refined model with the largest amount of inliers is then returned.

In practice, if an all-inlier sample is picked during a RANSAC iteration, this method obtains excellent approximations to the optimal solution. However, since RANSAC only keeps the best model defined by its inlier support, the last may not necessarily be the optimal model, as a better model may have appeared during these inner iterations and was subsequently discarded. This suggests that RANSAAC may significantly benefit from this procedure, and it turns out that it does. Therefore, saving these intermediate models is the only extra work needed to adapt this method to RANSAAC, as seen by the lines with white background in Algorithm 11.

RANSAAC variants for Local Optimization

In its original formulation, the local optimization (Algorithm 11) was executed every time a new best hypothesis is found during RANSAC iterations. However, what differs between implementations is how the best values are updated. Both LO-RANSAC [39] and its recent improved version LO^+ [88] work as an "algorithm within an algorithm", meaning that the local optimization step does not update RANSAC's internal *best-so-far* variables ($maxScore$ and $ransacRec$ in Algorithm 10), but has its own variables. This implies that the local optimization step is run every time a new best hypothesis is found, which is in the order of the logarithm of the number of samples [88]. On the other side, in USAC [127] the LO step updates the current best RANSAC values, which in general yields a single execution of this procedure accelerating the overall RANSAC algorithm.

As for RANSAAC, two other possible schemes exist when performing aggregation. One possibility is to aggregate only the models generated during the local optimization step, which in general are closer to the optimal model. A second possibility, is to combine these models with the models generated during the RANSAC iterations. This makes sense since worse models will be given less importance based on their inlier score, therefore the resulting aggregation accuracy should not suffer.

Based on these two possibilities for using the LO step, and the two possible ways to perform aggregation, four different versions of RANSAAC will be evaluated, namely LO, LO+, FAST LO and FAST LO+, where FAST indicates the USAC approach while the

Algorithm 11 Local Optimization

```
Require: $X_1, X_2 \in \mathbb{N}^2 \times N$: $N$ matching points, $Xi_1, Xi_2 \in \mathbb{N}^2 \times K$: $K$ inlier matches, $\delta_d \in \mathbb{R}$: RANSAC distance threshold, $s_{is} \in \mathbb{N}$: size of inner sample, $m_{\delta_d} \in \mathbb{R}$: threshold multiplier, $reps \in \mathbb{N}$: inner sample repetitions, $lsIters \in \mathbb{N}$: least squares iterations,
1:  if $K < 2s_{is}$ then
2:      return 0
3:  end if
4:  $\Delta_{\delta_d} = (m_{\delta_d} \cdot \delta_d - \delta_d)/lsIters$
5:  $maxScore \leftarrow K$
6:  $bestModel \leftarrow []$
7:  $models \leftarrow []$
8:  $weights \leftarrow []$
9:  for $it = 1$ to $reps$ do
10:     $Xss_1, Xss_2 \leftarrow GetRandomSample(Xi_1, Xi_2, s_{is})$    ▷ Get non-minimal sample
11:     $h_{nonMSS} \leftarrow GetModelLS(Xss_1, Xss_2)$    ▷ Compute model using Least Squares
12:     $inliers \leftarrow EvalH(h_{nonMSS}, X_1, X_2, \delta_d \cdot m_{\delta_d})$
13:     $Add(models, h_{nonMSS})$    ▷ Save computed model
14:     $Add(weights, \#inliers)$    ▷ and its weight
15:     $Xii_1, Xii_2 \leftarrow GetInliers(X_1, X_2, inliers)$    ▷ Select the inliers
16:     $h_{RLS} \leftarrow GetModelLS(Xii_1, Xii_2)$    ▷ Compute model using Least Squares
17:     for $j = 1$ to $lsIters$ do
18:         $inliers \leftarrow EvalH(h_{RLS}, X_1, X_2, \delta_d \cdot m_{\delta_d} - j \cdot \Delta_{\delta_d})$
19:         $Add(models, h_{RLS})$    ▷ Save computed model
20:         $Add(weights, \#inliers)$    ▷ and its weight
21:         $Xii_1, Xii_2 \leftarrow GetInliers(X_1, X_2, inliers)$    ▷ Select the inliers
22:         $h_{RLS} \leftarrow GetModelLS(Xii_1, Xii_2)$    ▷ Compute model using Least Squares
23:     end for
24:     $inliers \leftarrow EvalH(h_{RLS}, X_1, X_2, \delta_d)$
25:     $Add(models, h_{RLS})$    ▷ Save computed model
26:     $Add(weights, \#inliers)$    ▷ and its weight
27:     if $\#inliers > maxScore$ then
28:         $maxScore \leftarrow \#inliers$
29:         $bestModel \leftarrow h_{RLS}$
30:     end if
31: end for
32: return $maxScore, bestModel, models, weights$
```

+ sign indicates that only the hypotheses generated during the LO steps are aggregated. In algorithm 12 we see the four possible variants. Each variant is understood by choosing between yellow and green (FAST LO and LO) and between red and grey (all hypotheses and only LO hypotheses) and keeping only lines in those colors together with the white background lines. These variants are compared in section 5.4.2 and, as evidenced from the results, the LO+ approach achieves the best results overall.

5.4 Experiments

We compared the proposed method and its variants with the traditional approaches both qualitatively and quantitatively. Two applications were considered, namely the projective transform estimated using the DLT algorithm [160], and a homographic model that allows distortions on both images. The latter was estimated using the H_5 method of Kukelova et al. [87], which, given five matches, computes one radial distortion coefficient per image (using the division model and assuming the distortion center is on the center of the image), and a homography. Since robust homography estimation is, along with fundamental matrix estimation, the most frequent global model estimation problem in computer vision, we focus our evaluation on it. Nevertheless, we show results proving that the proposed approach also works on the other transform type as well, and could be potentially generalized to any parametric model obtained from points provided the possibility of averaging those models is available.

This section is summarized as follows. Focusing on projective transformations, we first show in section 5.4.1 some qualitative results comparing RANSAAC with the original RANSAC method by registering images of paintings. Then a detailed quantitative evaluation is performed by simulation in section 5.4.2. By fixing a ground-truth homography and randomly adding outliers and noise, we begin this evaluation by studying how the aggregation weight parameter p influences the results. Using the best found values, we analyze the accuracy evolution of our approach along the iterations, compared to RANSAC. This last test is interesting to gain insight about how fast the proposed approach improves over the standard method. Before comparing RANSAAC to the state-of-the-art methods, we test all its variants together with all proposed aggregation schemes to obtain the best candidate. From this experiment, the LO+ variant using weighted geometric median (wgmed) aggregation proves to outperform the rest on almost every test. Since some state-of-the-art methods such as USAC [127] use adaptive termination to save computational time, we study the influence of performing adaptive termination on both RANSAC and RANSAAC and compare it against USAC. We then perform a comparison between the proposed method and the state-of-the-art approaches without and including a last-step minimization. We also evaluate the influence of performing a last-step minimization on the inliers of RANSAAC, proving it is only necessary when not enough "good" models where sampled during the iterations. We also test the performance of RANSAAC under extreme conditions of 90% outliers and medium/high noise and observed some striking results. We finish the evaluation by testing the versatility of the method on randomly generated homographies, with and without adaptive termination.

5.4.1 Qualitative Evaluation: Paintings Registration with Projective Transformations

We first evaluated both RANSAC and RANSAAC with weighted mean aggregation on pictures of paintings taken in a museum under poor lighting conditions, which resulted in considerable noise visible on the output images. The objective was then to register each

Algorithm 12 Computing a 2D transformation using the (FAST) LO(+)-RANSAAC algorithm. The four possible variants are represented by only taking lines with specific background colors: FAST LO / LO and using all hypotheses / using only LO hypotheses (+) .

Require: $X_1, X_2 \in \mathbb{N}^2 \times N$: N matching points, $minSamples \in \mathbb{N}$: minimum amount of matches to compute the transform, $srcPts \in \mathbb{N}^2 \times K$: vector with $K = minSamples$ input points, $\delta_d \in \mathbb{R}$: RANSAC distance threshold, $minValidScore \in \mathbb{N}$: minimum score to consider an iteration valid, $iters \in \mathbb{N}$: amount of iterations, $p \in \mathbb{R}$: aggregation parameter.

```
$maxScore \leftarrow 0$
$maxScore_{LO} \leftarrow 0$
for $it = 1$ to $iters$ do
    $Xss_1, Xss_2 \leftarrow GetSample(X_1, X_2, minSamples)$
    $h \leftarrow GenerateHypothesis(Xss_1, Xss_2)$
    $inliers \leftarrow EvalH(h, X_1, X_2, \delta_d)$
    if $\#inliers > maxScore$ then
        $maxScore \leftarrow \#inliers$
        $ransacRes \leftarrow h$
        $scoreBest_{LO}, hBest_{LO}, hs_{LO}, w_{LO} \leftarrow LO(X_1, X_2, inliers, \delta_d)$
        $Add(dstPts_{LO}, Project(srcPts, hs_{LO}))$    ▷ Adds projected points for each hypothesis
        $Add(weights_{LO}, w_{LO})$    ▷ and their corresponding weights
        if $scoreBest_{LO} > maxScore$ then
            $maxScore \leftarrow scoreBest_{LO}$
            $ransacRes \leftarrow hBest_{LO}$
        end if
        if $scoreBest_{LO} > maxScore_{LO}$ then
            $maxScore_{LO} \leftarrow scoreBest_{LO}$
            $ransacRes_{LO} \leftarrow hBest_{LO}$
        end if
    end if
    if $score > minValidScore$ then
        $Add(dstPts, Project(srcPts, h))$
        $Add(weights, \#inliers)$
    end if
end for
if $\#dstPts > 0$ then
    $finalDstPts \leftarrow dstPts_{LO}$
    $finalWeights \leftarrow weights_{LO}$
    $finalDstPts \leftarrow dstPts \cup dstPts_{LO}$
    $finalWeights \leftarrow weights \cup weights_{LO}$
    $resPts \leftarrow Aggregate(finalDstPts, finalWeights, p)$ // See 5.3.2
    return $GenerateHypothesis(srcPts, resPts)$
else
    return $ransacRes$
    return $ransacRes_{LO}$
end if
```

photograph of a painting detail taken from a shorter range, with the whole painting [27]. By using SIFT features and matching its descriptors, we proceeded to register both images using both methods. An example can be seen in Fig. 5.4, where the camera distortion was previously corrected and the images are related by a pure homography. After registering both images, the resulting difference is mainly noise, however RANSAAC performs better, as evidenced in the flamingo on the bottom part of the painting (middle row of Fig. 5.4). When measurement noise is added to the input points on both images ($\sigma = 10$), the difference in the registration performance becomes much more evident. Edges due to misalignments appear all over the difference image in RANSAC, but are dimmer on the RANSAAC results (bottom row of Fig. 5.4).

To get a better understanding of the behaviour of the method, Fig. 5.5 shows the point distribution with their sizes according to their obtained weights using $p = 5$. As can be seen, the RANSAC algorithm always returns the model given by the points with the highest weight, ignoring every other estimation. By incorporating other weighted estimations and aggregating them, a closer point to the ground truth is obtained for every corner, therefore yielding a more precise homography.

5.4.2 Quantitative Evaluation by a Simulated Ground Truth. Application 1: Estimating Projective Transforms.

To perform in depth experimentation we built a valid ground truth, composed of point matches and a transform. In order to obtain a valid real-life transform, we used the traditional SIFT algorithm, followed by the application of RANSAC with 100000 iterations between two images. To generate the matches, we randomly selected a fixed set of points from the first image as inliers and we matched them to the points obtained by applying the ground truth transform to them. Finally, to incorporate outliers in the dataset, we randomly picked other points from the first image and matched them with random positions on the second image.

On this dataset, we compared several state-of-the-art methods and all variants of RANSAAC. To measure the error, we averaged the symmetric transfer error in pixels for every inlier. If ϕ_θ is the obtained model, X_i is an inlier point in the first image, Y_i its match and K the amount of inliers, then the error E is defined as

$$E = \frac{1}{K}\sum_{i=1}^{K} \frac{\|\phi_\theta(X_i) - Y_i\|_2 + \left\|\phi_\theta^{-1}(Y_i) - X_i\right\|_2}{2}. \tag{5.14}$$

By varying the number of inliers, the percentage of outliers, the input noise and the number of iterations performed, we computed for each experiment the average error $\bar{E}$ and the standard deviations σ_E over 100 evaluations.

Note that for all presented results, the very same random samples for each iteration were used for RANSAC, LO-RANSAC and RANSAAC, thus ensuring equal chance for those methods. We used the implementation of USAC available from the author [127].

We evaluated the performance of the proposed approach under distinct conditions. Since the noise level was known beforehand, the RANSAC δ_d parameter was set according to section 5.1.2, and for RANSAAC this parameter was fixed to $2*\delta_d$ due to the higher tolerance of the method with respect to this parameter. We took $minValidScore = 4$ on every situation and unless explicitly mentioned, the parameter p was set depending on the amount of iterations performed and on the used aggregation scheme, indicated by the results obtained in the following section.

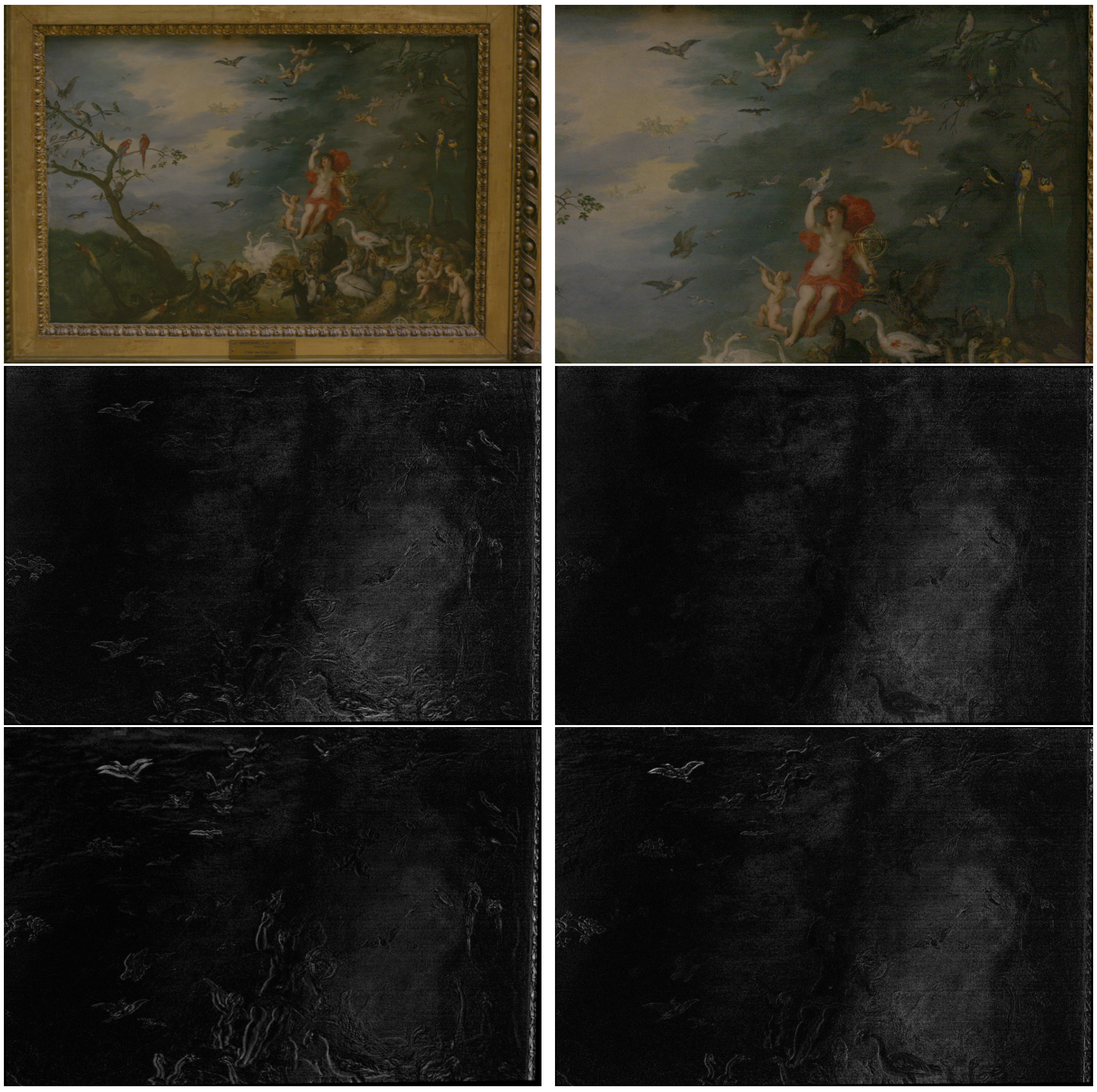


Figure 5.4 – RANSAC (left) and RANSAAC (right) registration results. Dynamic ranges were stretched to highlight differences. Top: input images. Middle: registration difference using input images. Bottom: registration difference by adding measurement noise of $\sigma = 10$ to the keypoint positions on both images in pixels. Edges due to misalignments can be perceived in the difference image, particularly by using the traditional RANSAC method. Painting: L'Air, ou L'Optique. Jan I BRUEGHEL, 1621. Louvre Museum.

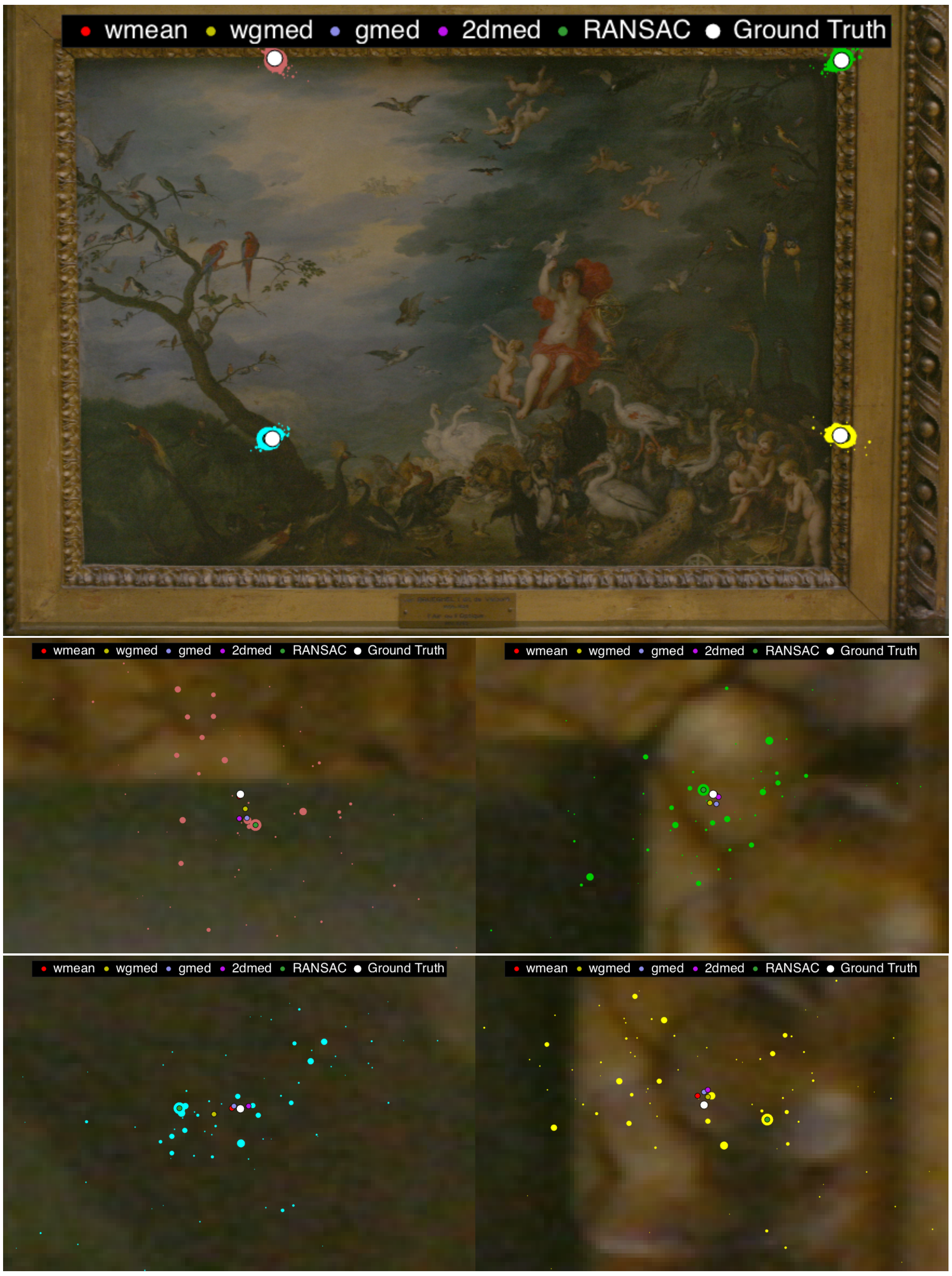


Figure 5.5 – Resulting point distributions resized according to their weights and zoom in on each one of the 4 corners for 1000 iterations. For visualization purposes, points with low weights were not included. Notation: $wmean$ and $wgmed$ are the weighted mean and the weighted geometric median aggregation respectively. $gmed$ and $2dmed$ represent aggregated results using (unweighted) geometric median and 1D median on both dimensions respectively. The reader is advised to zoom in to see better the positions of the various estimates.

Aggregation Weights

Since there are outliers in the input data, the weighting of the hypotheses on the aggregation is mandatory. The weight parameter p defines how much a hypothesis is taken into account in the aggregation process. As the parameter p increases, the weighting penalizes the transforms with lower scores, while increasing the influence of higher scored ones. Indeed, increasing the value p leads to eliminate the influence of poor hypotheses, but a too high value will imply using too few transforms for the final aggregation. Eventually, this leads the method to only use the best hypothesis, thus behaving like the traditional RANSAC.

To this end, an experiment was performed to measure the impact of the parameter p (and therefore how much poor hypotheses should be discriminated) for the cases of 20% and 50% outliers for both the standard RANSAAC method as well as its LO-RANSAAC+ variant. For the case of the original approach, it turns out that the best value of p depends on both the outlier percentage, but most importantly, on the number of hypotheses performed. Indeed, as more hypotheses are drawn, they should be better discriminated, suggesting higher values of p. Errors for both weighted mean and weighted geometric median aggregations for the standard RANSAAC while varying the weight exponent p for projective transforms are displayed in the first two rows of Fig. 5.6. For this method, we conclude that for the weighted geometric median aggregation, the exponent p should be set to 2 and 3 for both 1000 and 10000 iterations respectively. The best values of p for the weighted mean range between 4 to 8. We chose to use $p = 5$ for 1000 iterations and $p = 7$ for 10000 iterations. We observed that as soon as the value of p is high enough so that it allows to correctly discriminate between inliers and outliers, then the resulting accuracy does not vary significatively. This observation becomes more important for the weighted mean aggregation, particularly because the weighted geometric median aggregation is more robust against outliers.

We also performed the same test for the LO-RANSAAC+ variant of our approach and its results are shown in the third and fourth row of Fig. 5.6. It turns out that for the weighted geometric median aggregation, varying the p parameter does not affect the resulting accuracy. This is due to the fact that the geometric median is more robust to outliers than the mean, therefore not requiring to completely avoid them in the aggregation. For the weighted mean aggregation, although the accuracies did not vary much (note that the error ranges are significantly lower between the first two rows and the second two rows), the same pattern of higher values of p depending on the iteration quantity was observed. For this method, we selected $p = 11$ for 1000 iterations and $p = 15$ for 10000, although any p value higher than 7 offers similar performance.

Accuracy evolution along iterations

To give insight about the evolution of the error under RANSAC and under the proposed approach, their accuracy was computed while iterating. To this end, we simulated input points from a predefined real homography (i.e. computed from two images of a landscape), corrupted them with white Gaussian Noise of $\sigma = 5$ and added 50% outliers. Using this input, we run RANSAC and our method 1000 times and calculated the error for every iteration (for RANSAAC, this implied performing aggregation) up to 20000 iterations. Finally, we computed the average error over all experiments per iteration. In Fig. 5.7 left, we observe how RANSAC does not only converge slower in comparison to both aggregation schemes of RANSAAC, but also that it never reaches their accuracy. However, as evidenced in the figure on the right where the first 500 iterations are shown, RANSAC usually achieves better results at the beginning, while RANSAAC usually requires more *all-inlier* samples to produce more accurate results. This is ex-

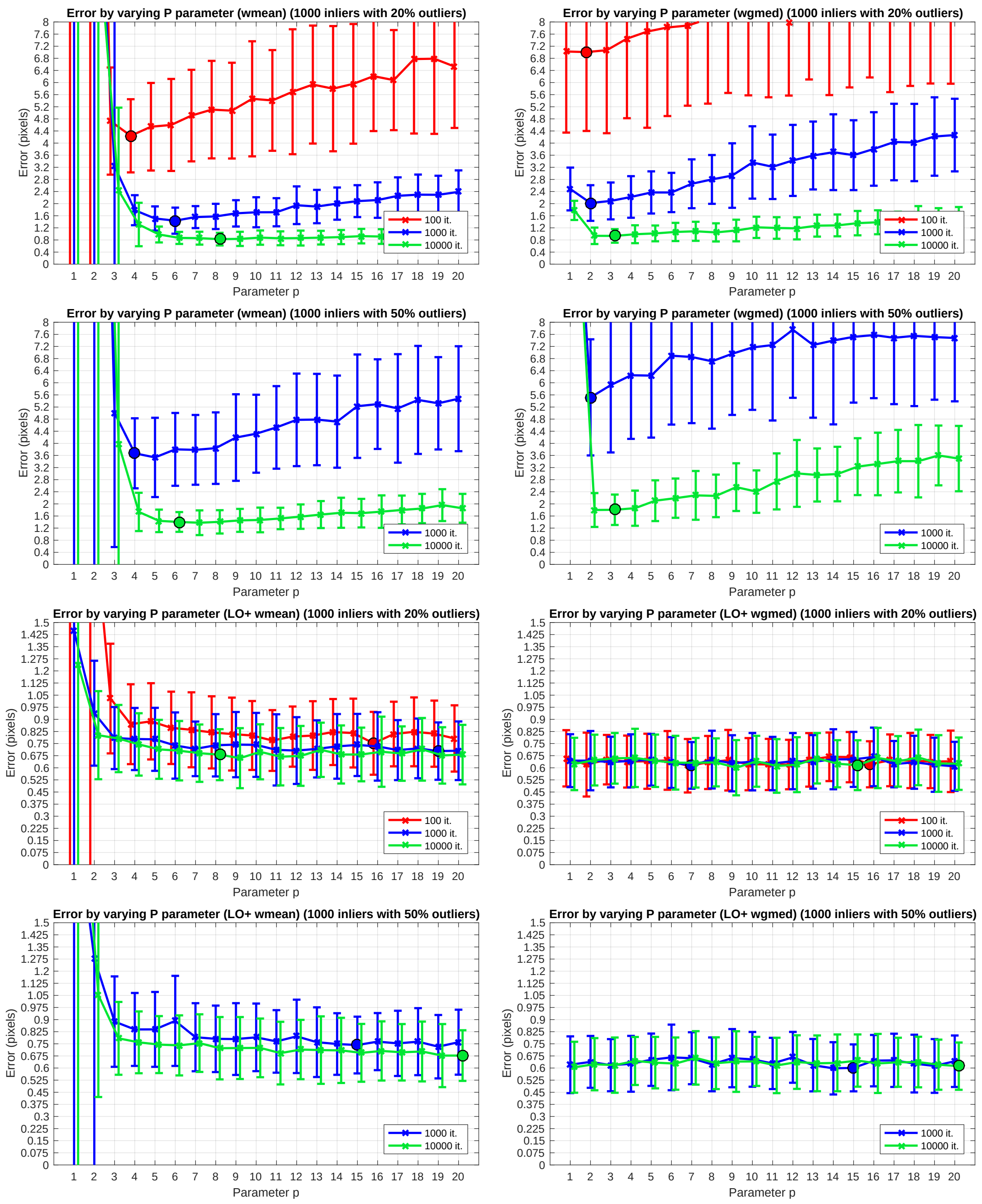


Figure 5.6 – Comparison of the error (in pixels) and the standard deviations by varying the p exponent. Results are drawn for 100, 1000 and 10000 iterations. Model: projective transformations. Method: LO-RANSAAC+. Inlier qty.: 1000. Noise: $\sigma = 5$. Averaged over 100 realizations. Big dots indicate lowest errors. First and second rows are results of standard RANSAAC with 20% and 50% outliers respectively, while third and four rows are results of using LO-RANSAAC+ under the same outlier configuration. **Left**: Weighted mean aggregation. **Right**: Weighted geometric median aggregation.

pected, since RANSAAC gains from aggregating all generated samples, thus only having a single "good" hypothesis and several "bad" ones will always benefit a method such as RANSAC that just considers the single hypothesis with the highest obtained score. However, as more *all-inlier* hypotheses are sampled, RANSAAC starts improving over traditional RANSAC (in the figure this happens at 125 iterations).

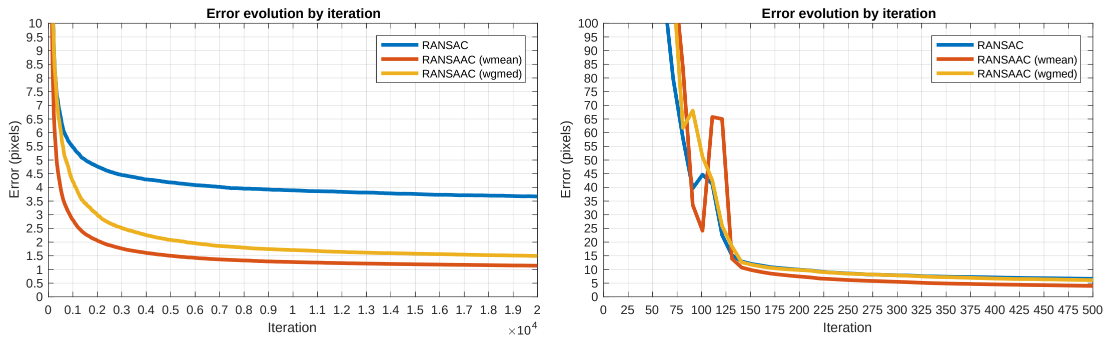


Figure 5.7 – Comparison of the error by iteration (in pixels) between RANSAC and the proposed algorithm using $wmean$ and $wgmed$ aggregation. Model: projective transformations. Noise: $\sigma = 5$. 100 inliers and 50% outliers. Averaged over 1000 realizations. **Left**: Up to 20000 iterations. **Right**: First 500 iterations.

Local Optimization and Aggregation Methods Evaluation

We evaluated both aggregation variants for the four proposed versions of RANSAAC by varying the number of inliers, the percentage of outliers and the number of noise iterations performed. Table 5.1 shows results for both the FAST LO and the FAST LO+ variants under three different noise levels ($\sigma = 0.5, 2$ or 5). For each method and noise level, both 1000 and 10000 iterations were tested, each row corresponding to situations presenting either 0%, 5%, 20% or 50% outliers respectively.

Several conclusions could be inferred from these results. As can be seen, by using more inliers, every aggregated method improved, and in every case, the best performing aggregation scheme was the weighted geometric median. Interestingly, this did not occur when no local optimization steps were performed.

The reason behind this is evident, as both the outlier ratio and the number of inliers increase, the probability of obtaining good models by random sampling becomes lower, therefore averaging these not-so-close-to-the-optimal models will achieve better results than taking their median, which will be far away from the optimal solution. However, when better (i.e. closer to the optimal solution) hypotheses are available, then computing their median is not only more accurate but also drastically reduces the possible bias caused by including outliers in the estimation. Indeed, performing LO steps provides excellent models, which justifies why $wgmed$ aggregation improves over $wmean$. Another remark is that as there are more inliers in the data, including the RANSAC hypotheses in the aggregation improves over taking only the models generated on the LO steps. This improvement however, is almost negligible when using the LO version of the method, as observed from table 5.2. Indeed, both versions result in practically the same accuracy, implying that the least computationally demanding version should be used (LO+). Again, using the weighted geometric median aggregation improves over the weighted mean, particularly when there are more inliers.

Finally, results for the original RANSAAC approach together with both their best locally optimized versions are shown in table 5.3. From this table we can infer that the LO+ method with weighted geometric median aggregation gives the best results in terms of

Table 5.1 – Average errors and standard deviations for the different versions of RANSAAC using both proposed aggregation schemes. For noises $\sigma = 0.5, 2$ and 5, each method was evaluated with 100 and 1000 inliers using both 1000 and 10000 iterations. The four errors represent four outlier ratios: 0%, 5%, 20% and 50%. Bold denotes the best performers.

σ	FASTLO (wmean)		FASTLO+ (wmean)		FASTLO (wgmed)		FASTLO+ (wgmed)	
	1k	10k	1k	10k	1k	10k	1k	10k
	100 inliers							
0.5	0.21±0.06	**0.21**±0.05	0.22±0.06	0.22±0.06	**0.20**±0.05	0.21±0.05	**0.20**±0.05	**0.20**±0.05
	0.22±0.06	**0.21**±0.05	0.23±0.06	0.23±0.06	**0.21**±0.05	**0.21**±0.06	**0.20**±0.05	**0.21**±0.05
	0.22±0.06	**0.21**±0.06	0.23±0.06	0.23±0.06	**0.21**±0.05	**0.20**±0.05	**0.21**±0.05	**0.21**±0.06
	0.23±0.07	0.23±0.06	0.23±0.07	0.24±0.06	**0.21**±0.06	**0.22**±0.05	**0.21**±0.06	**0.22**±0.05
2	0.80±0.22	0.85±0.22	0.85±0.24	0.91±0.23	0.77±0.22	0.85±0.24	**0.76**±0.22	**0.83**±0.23
	0.82±0.19	0.84±0.23	0.87±0.21	0.91±0.25	**0.78**±0.21	0.84±0.25	**0.78**±0.21	**0.81**±0.23
	0.92±0.23	0.86±0.21	0.96±0.25	0.93±0.23	**0.85**±0.23	**0.84**±0.19	**0.84**±0.23	**0.84**±0.20
	0.93±0.24	0.90±0.23	0.94±0.24	0.93±0.25	**0.84**±0.21	**0.84**±0.20	**0.84**±0.21	**0.84**±0.21
5	2.06±0.55	2.01±0.55	2.20±0.58	2.21±0.60	**1.92**±0.52	2.09±0.59	**1.92**±0.52	**1.95**±0.56
	2.15±0.57	2.16±0.56	2.25±0.61	2.30±0.61	**2.06**±0.54	2.17±0.62	**2.05**±0.55	**2.11**±0.54
	2.23±0.57	2.16±0.60	2.32±0.57	2.31±0.65	2.08±0.57	2.11±0.60	**2.07**±0.55	**2.05**±0.55
	2.30±0.58	2.11±0.52	2.31±0.58	2.18±0.54	2.06±0.56	1.98±0.47	**2.04**±0.55	**1.97**±0.48
	1000 inliers							
0.5	0.10±0.03	**0.07**±0.02	0.14±0.04	0.13±0.04	**0.08**±0.02	**0.07**±0.02	**0.07**±0.03	**0.07**±0.03
	0.11±0.03	**0.07**±0.02	0.14±0.04	0.13±0.04	**0.08**±0.02	**0.07**±0.02	**0.08**±0.05	**0.07**±0.04
	0.12±0.03	0.09±0.02	0.14±0.04	0.13±0.04	**0.07**±0.03	**0.07**±0.02	**0.07**±0.03	**0.07**±0.03
	0.14±0.04	0.11±0.03	0.15±0.05	0.13±0.04	**0.08**±0.04	**0.07**±0.02	**0.08**±0.05	**0.07**±0.03
2	0.39±0.10	0.30±0.08	0.53±0.18	0.53±0.13	**0.27**±0.08	**0.28**±0.08	**0.28**±0.15	0.30±0.11
	0.45±0.12	0.30±0.08	0.56±0.15	0.53±0.16	**0.29**±0.09	**0.28**±0.07	**0.30**±0.11	0.30±0.12
	0.48±0.12	0.33±0.08	0.55±0.16	0.50±0.13	**0.29**±0.12	**0.27**±0.07	**0.30**±0.14	0.28±0.09
	0.57±0.17	0.46±0.12	0.60±0.25	0.53±0.16	**0.35**±0.22	**0.29**±0.10	**0.36**±0.25	0.30±0.15
5	1.09±0.26	0.75±0.20	1.44±0.36	1.31±0.39	**0.73**±0.20	**0.66**±0.18	0.77±0.38	0.71±0.30
	1.10±0.24	0.79±0.22	1.42±0.37	1.31±0.40	**0.73**±0.20	**0.72**±0.22	0.77±0.38	0.77±0.32
	1.16±0.33	0.86±0.22	1.36±0.46	1.25±0.37	**0.73**±0.27	**0.69**±0.21	0.81±0.55	**0.69**±0.25
	1.48±0.51	1.17±0.28	1.53±0.54	1.38±0.43	**0.93**±0.61	**0.78**±0.33	0.99±0.73	0.88±0.67

Table 5.2 – Average errors and standard deviations for the different versions of RANSAAC using both proposed aggregation schemes. For noises $\sigma = 0.5, 2$ and 5, each method was evaluated with 100 and 1000 inliers using both 1000 and 10000 iterations. The four errors represent four outlier ratios: 0%, 5%, 20% and 50%. Bold denotes the best performers.

σ	LO (wmean)		LO+ (wmean)		LO (wgmed)		LO+ (wgmed)	
	1k	10k	1k	10k	1k	10k	1k	10k
	100 inliers							
0.5	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05	**0.19**±0.05	**0.19**±0.05	**0.19**±0.05	**0.19**±0.05	**0.20**±0.05
	0.20±0.05	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05	**0.20**±0.05
	0.21±0.05	**0.20**±0.05	**0.21**±0.06	**0.20**±0.05	**0.21**±0.06	**0.20**±0.05	**0.21**±0.06	**0.20**±0.05
	0.21±0.06	**0.21**±0.05	**0.21**±0.06	**0.21**±0.05	**0.21**±0.06	**0.21**±0.05	**0.21**±0.06	**0.21**±0.05
2	**0.75**±0.19	**0.81**±0.22	**0.75**±0.20	**0.81**±0.22	**0.75**±0.20	**0.81**±0.23	**0.75**±0.21	**0.81**±0.22
	0.76±0.19	**0.81**±0.22	**0.76**±0.19	**0.81**±0.21	**0.76**±0.20	**0.81**±0.23	**0.76**±0.20	**0.81**±0.22
	0.83±0.22	**0.83**±0.20	**0.83**±0.22	**0.83**±0.20	**0.83**±0.22	**0.83**±0.20	**0.83**±0.22	**0.82**±0.20
	0.84±0.19	0.82±0.21	0.84±0.19	0.82±0.21	**0.82**±0.19	**0.81**±0.21	**0.82**±0.19	**0.81**±0.21
5	1.90±0.52	**1.91**±0.55	1.90±0.52	**1.90**±0.56	**1.88**±0.51	**1.90**±0.55	**1.88**±0.50	**1.91**±0.55
	2.02±0.49	2.07±0.51	**2.02**±0.49	**2.07**±0.51	**2.02**±0.51	**2.06**±0.53	**2.01**±0.51	**2.06**±0.52
	2.05±0.53	2.03±0.56	2.05±0.53	**2.01**±0.57	**2.03**±0.53	2.03±0.55	**2.03**±0.54	2.03±0.55
	2.01±0.56	**1.94**±0.46	2.01±0.55	**1.94**±0.46	**1.98**±0.52	1.96±0.48	**1.98**±0.52	1.97±0.48
	1000 inliers							
0.5	**0.07**±0.02	**0.07**±0.02	**0.07**±0.02	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02	**0.07**±0.02	**0.06**±0.02
	0.07±0.02	**0.06**±0.02	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02	**0.06**±0.02	**0.06**±0.02	**0.06**±0.02
	0.07±0.02	**0.07**±0.02	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02	**0.06**±0.02	**0.06**±0.02	**0.06**±0.02
	0.08±0.02	**0.07**±0.02	0.08±0.02	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02	**0.07**±0.02	**0.06**±0.02
2	0.27±0.07	**0.26**±0.07	0.27±0.07	0.27±0.07	**0.24**±0.06	**0.26**±0.07	**0.24**±0.06	**0.26**±0.07
	0.28±0.09	0.27±0.07	0.28±0.08	0.28±0.07	**0.26**±0.07	**0.26**±0.07	**0.26**±0.07	**0.26**±0.07
	0.28±0.07	0.26±0.07	0.28±0.07	0.27±0.06	**0.25**±0.06	**0.25**±0.06	**0.25**±0.06	**0.25**±0.06
	0.32±0.08	0.28±0.07	0.32±0.08	0.28±0.07	**0.26**±0.07	**0.25**±0.06	**0.26**±0.07	**0.25**±0.06
5	0.71±0.18	0.66±0.17	0.71±0.18	0.68±0.19	**0.63**±0.14	**0.61**±0.17	**0.63**±0.14	**0.62**±0.17
	0.73±0.16	0.69±0.21	0.74±0.16	0.70±0.21	**0.65**±0.15	**0.66**±0.20	**0.65**±0.16	**0.67**±0.19
	0.70±0.18	0.68±0.17	0.70±0.18	0.69±0.17	**0.63**±0.15	**0.64**±0.17	**0.63**±0.15	**0.64**±0.17
	0.80±0.20	0.70±0.19	0.80±0.19	0.70±0.19	**0.66**±0.17	**0.63**±0.18	**0.66**±0.17	**0.63**±0.18

Table 5.3 – Average errors and standard deviations for the different versions of RANSAAC using both proposed aggregation schemes. For noises $\sigma = 0.5, 2$ and 5, each method was evaluated with 100 and 1000 inliers using both 1000 and 10000 iterations. The four errors represent four outlier ratios: 0%, 5%, 20% and 50%. Bold denotes the best performers.

σ	wmean		FASTLO (wmean)		LO+ (wmean)		wgmed		FASTLO (wgmed)		LO+ (wgmed)	
	1k	10k	1k	10k	1k	10k	1k	10k	1k	10k	1k	10k
	100 inliers											
0.5	0.23±0.06	0.21±0.05	0.21±0.06	0.21±0.05	**0.20**±0.05	**0.19**±0.05	0.25±0.07	0.23±0.06	**0.20**±0.05	0.21±0.05	**0.19**±0.05	**0.20**±0.05
	0.24±0.06	0.21±0.05	0.22±0.06	**0.21**±0.05	**0.20**±0.05	**0.20**±0.05	0.27±0.08	0.23±0.06	**0.21**±0.05	**0.21**±0.06	**0.20**±0.05	**0.20**±0.05
	0.26±0.07	0.21±0.05	0.22±0.06	**0.21**±0.06	**0.21**±0.06	**0.20**±0.05	0.29±0.09	0.23±0.06	**0.21**±0.05	**0.20**±0.05	**0.21**±0.06	**0.20**±0.05
	0.45±0.14	0.26±0.06	0.23±0.07	0.23±0.06	**0.21**±0.06	**0.21**±0.05	0.61±0.22	0.30±0.08	**0.21**±0.06	**0.22**±0.05	**0.21**±0.06	**0.21**±0.05
2	0.89±0.23	0.87±0.23	0.80±0.22	0.85±0.22	**0.75**±0.20	**0.81**±0.22	1.02±0.27	0.92±0.27	0.77±0.22	0.85±0.24	**0.75**±0.21	**0.81**±0.22
	0.91±0.23	0.86±0.23	0.82±0.19	0.84±0.23	**0.76**±0.19	**0.81**±0.21	1.01±0.28	0.91±0.27	0.78±0.21	0.84±0.25	**0.76**±0.20	**0.81**±0.22
	1.03±0.27	0.89±0.20	0.92±0.23	0.86±0.21	**0.83**±0.22	**0.83**±0.20	1.21±0.34	0.97±0.24	0.85±0.23	0.84±0.19	**0.83**±0.22	**0.82**±0.20
	1.76±0.48	1.03±0.23	0.93±0.24	0.90±0.23	0.84±0.19	0.82±0.21	2.39±0.74	1.23±0.36	0.84±0.21	0.84±0.20	**0.82**±0.19	**0.81**±0.21
5	2.20±0.63	2.07±0.57	2.06±0.55	2.01±0.55	1.90±0.52	**1.90**±0.56	2.42±0.66	2.29±0.67	1.92±0.52	2.09±0.59	**1.88**±0.50	**1.91**±0.55
	2.39±0.61	2.22±0.56	2.15±0.57	2.16±0.56	**2.02**±0.49	**2.07**±0.51	2.72±0.73	2.40±0.73	2.06±0.54	2.17±0.62	**2.01**±0.51	**2.06**±0.52
	2.58±0.69	2.25±0.62	2.23±0.57	2.16±0.60	2.05±0.53	**2.01**±0.57	2.95±0.84	2.47±0.77	2.08±0.57	2.11±0.60	**2.03**±0.54	2.03±0.55
	4.60±1.55	2.47±0.63	2.30±0.58	2.11±0.52	2.01±0.55	**1.94**±0.46	6.00±1.87	2.98±0.89	2.06±0.56	1.98±0.47	**1.98**±0.52	1.97±0.48
	1000 inliers											
0.5	0.11±0.03	0.07±0.02	0.10±0.03	0.07±0.02	**0.07**±0.02	**0.07**±0.02	0.15±0.04	0.09±0.02	0.08±0.02	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02
	0.12±0.03	0.07±0.02	0.11±0.03	0.07±0.02	**0.07**±0.02	**0.07**±0.02	0.16±0.04	0.09±0.02	0.08±0.02	**0.07**±0.02	**0.06**±0.02	**0.06**±0.02
	0.18±0.04	0.08±0.02	0.12±0.03	0.09±0.02	**0.07**±0.02	**0.07**±0.02	0.22±0.06	0.10±0.03	**0.07**±0.03	**0.07**±0.02	**0.06**±0.02	**0.06**±0.02
	0.42±0.14	0.15±0.04	0.14±0.04	0.11±0.03	0.08±0.02	**0.07**±0.02	0.54±0.17	0.22±0.07	0.08±0.04	**0.07**±0.02	**0.07**±0.02	**0.06**±0.02
2	0.47±0.12	0.30±0.08	0.39±0.10	0.30±0.08	0.27±0.07	0.27±0.07	0.57±0.17	0.35±0.11	0.27±0.08	0.28±0.08	**0.24**±0.06	**0.26**±0.07
	0.52±0.13	0.31±0.08	0.45±0.12	0.30±0.08	0.28±0.08	0.28±0.07	0.64±0.19	0.36±0.09	0.29±0.09	0.28±0.07	**0.26**±0.07	**0.26**±0.07
	0.67±0.17	0.33±0.09	0.48±0.12	0.33±0.08	0.28±0.07	0.27±0.06	0.92±0.28	0.40±0.12	0.29±0.12	0.27±0.07	**0.25**±0.06	**0.25**±0.06
	1.68±0.50	0.61±0.15	0.57±0.17	0.46±0.12	0.32±0.08	0.28±0.07	2.21±0.76	0.83±0.27	0.35±0.22	0.29±0.10	**0.26**±0.07	**0.25**±0.06
5	1.18±0.34	0.75±0.20	1.09±0.26	0.75±0.20	0.71±0.18	0.68±0.19	1.54±0.49	0.82±0.22	0.73±0.20	0.66±0.18	**0.63**±0.14	**0.62**±0.17
	1.25±0.35	0.80±0.22	1.10±0.24	0.79±0.22	0.74±0.16	0.70±0.21	1.55±0.48	0.92±0.24	0.73±0.20	0.72±0.22	**0.65**±0.16	**0.67**±0.19
	1.66±0.47	0.88±0.21	1.16±0.33	0.86±0.22	0.70±0.18	0.69±0.17	2.15±0.67	1.04±0.28	0.73±0.27	0.69±0.21	**0.63**±0.15	**0.64**±0.17
	4.22±1.25	1.55±0.48	1.48±0.51	1.17±0.28	0.80±0.19	0.70±0.19	5.50±1.91	2.15±0.73	0.93±0.61	0.78±0.33	**0.66**±0.17	**0.63**±0.18

accuracy and stability on every analyzed case. It is remarkable how the $wgmed$ aggregation benefits from local optimization, evidenced by the increase in accuracy up to an order of magnitude for 1000 iterations, 1000 inliers, 50% outliers and $\sigma = 5$. Finally, performing more RANSAC iterations does not improve the results as considerably when performing local optimization. Indeed, results could become slightly worse when performing more than enough iterations since there are more possible bad hypotheses that have to be correctly discriminated. For those cases, a higher value of the parameter p becomes necessary. In conclusion, adding local optimization to RANSAAC does not only improve results but also enables the method perform fewer iterations, making it less computationally demanding.

Similar conclusions are drawn by observing the performance as the noise increases in Fig. 5.8 for 1000 iterations and Fig. 5.9 for 10000 iterations. From these, we observe that non locally optimized RANSAAC aggregation methods require many more iterations to become more accurate, and even so, they still do not improve over their locally optimized counterparts. Also, averaging the models proved to be more dependent on the amount of iterations than computing their geometric median, which seems more stable and with faster convergence.

Results by using adaptive termination

We compared the LO-RANSAAC+ algorithm using both proposed aggregation schemes together with USAC, RANSAC and RANSAC followed by performing least squares minimization on the results. On every case, a maximum of 1000 iterations was allowed for all methods, although the methods that included adaptive termination varied this pa-

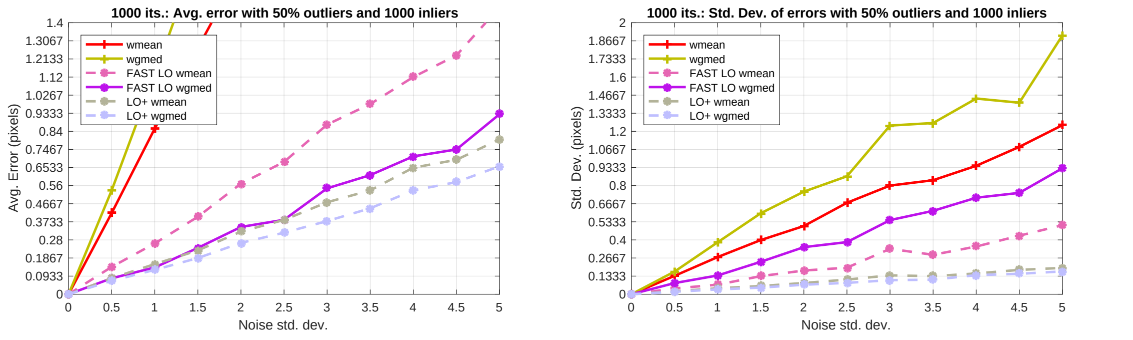


Figure 5.8 – Performance comparison through noise between the different aggregation and LO variants for RANSAAC by performing 100 experiments. Scenario: 50% outliers, 1000 inliers and 1000 iterations. **Left:** Average error. **Right:** Standard deviation of the error.

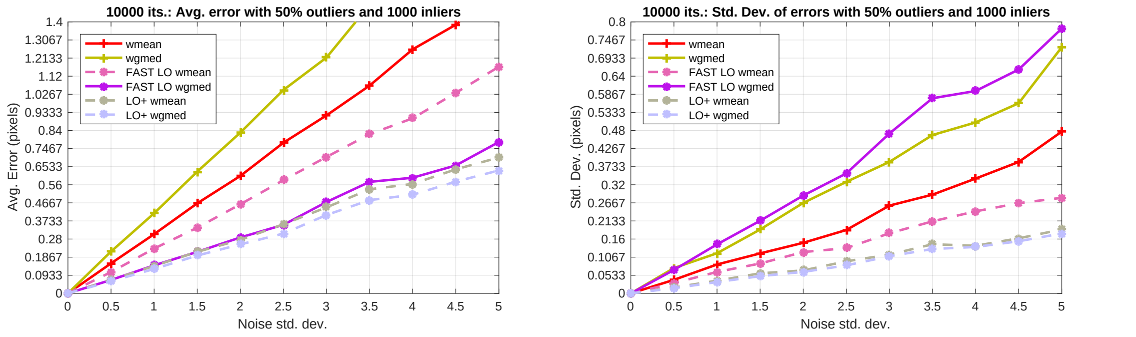


Figure 5.9 – Performance comparison through noise between the different aggregation and LO variants for RANSAAC by performing 100 experiments. Scenario: 50% outliers, 1000 inliers and 10000 iterations. **Left:** Average error. **Right:** Standard deviation of the error.

rameter dynamically during the execution, based on Eq. (5.5) using $\eta_0 = 0.99$. Results show average errors for 100 executions, where the noise value was fixed to $\sigma = 5$ and the proper distance values were set for every method according to section 5.1.2. USAC results are in fact duplicated since adaptive termination was always activated, however it is shown in this figure for comparison purposes.

Several conclusions are drawn from results shown in Fig. 5.10. First, while the weighted mean aggregation slightly suffers from using adaptive termination, the weighted geometric median aggregation in fact slightly benefits from it. The reason for this is again the same as mentioned before, and is related with the fact that making more iterations implies also having to better discriminate between samples. Also, when fewer inliers are available, both LO+ aggregation methods beat USAC, however with adaptive termination, the weighted mean achieves slightly worse results than USAC when adaptive termination is used. This does not occur using $wgmed$ aggregation.

Comparison with state-of-the-art methods.

We compared the proposed approach with several state-of-the-art methods. As representatives of RANSAAC, we included both aggregation methods for the most accurate LO-RANSAAC+ algorithm, and we added the faster FASTLO-RANSAAC with weighted geometric median aggregation. They were compared to all methods implemented in the GML RANSAC Matlab Toolbox [102], although the two best performing were kept, namely MSAC [173] and ZHANGSAC [190]. The recent USAC algorithm [127] was also included in the comparison, together with two minimization methods applied directly on the inliers using an oracle: the linear least squares methods using the DLT algorithm

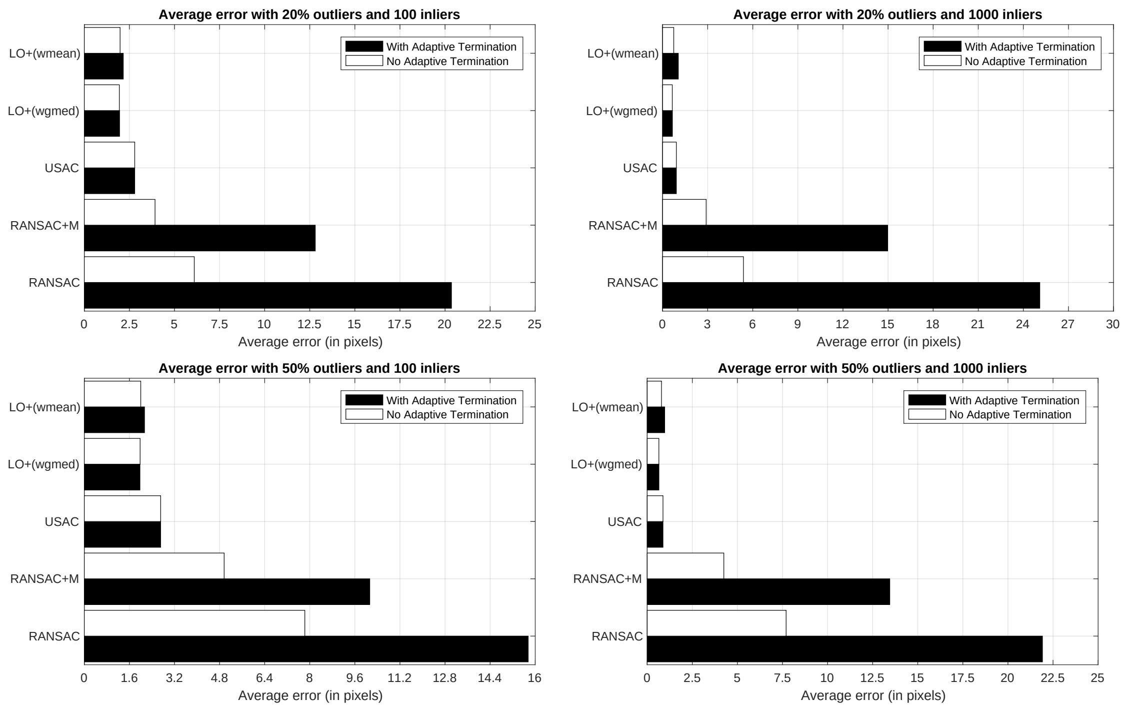


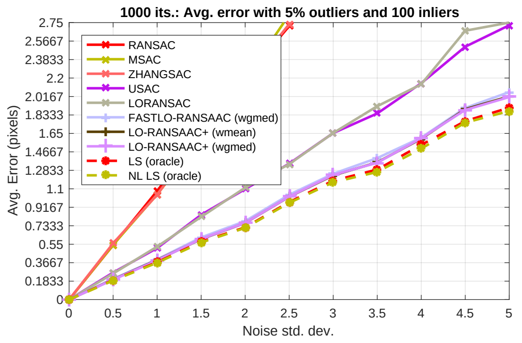
Figure 5.10 – Comparison by using adaptive termination to the proposed approach. **Left**: 100 inliers. **Right**: 1000 inliers. **Top**: 20% outliers. **Bottom**: 50% outliers.

and the non-linear method which minimizes Sampson's approximation to geometric reprojection error [70].

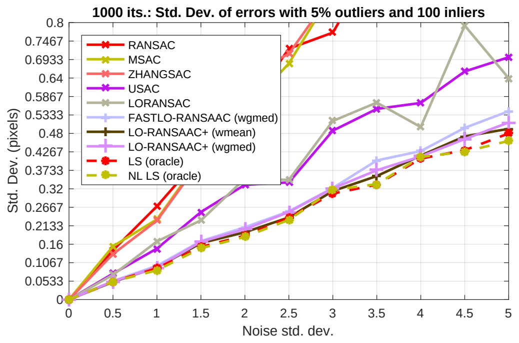
With only 100 inlier matches available, the average errors together with their standard deviations are shown in Fig. 5.11 for 5%, 20% and 50% outlier ratios. In these experiments, USAC did not show any reasonable gain with respect to the LO-RANSAC method. On the other side, the three compared variants of RANSAAC improve considerably over USAC, particularly as the noise gets higher. Furthermore, their accuracies were close to the methods that used an oracle to discriminate between inliers and outliers. Between the three variants, the LO-RANSAAC+ using $wgmed$ aggregation seems to be slightly better, however this difference is almost negligible. Finally, any RANSAC variant that does not include a local optimization step was not able to achieve acceptable results.

When there are more points in the input data, the discrimination between inliers and outliers becomes more challenging. However, provided it is possible to do so correctly, the errors achieved are lower due to the larger amount of matches, as seen in Fig. 5.12. Again, all non locally optimized approaches fail in this context. The LO-RANSAC approach is able to attain good results, although they are always worse than the state-of-the-art USAC method, contrary to the previous scenario. Again, the three RANSAAC variants improve over USAC, although by a lower margin. Nevertheless, LO-RANSAAC+ with $wgmed$ aggregation does still beat USAC by a wide margin and is again close in performance to the methods that use an oracle. What is more, when there are 75% outliers, the USAC approach failed, therefore is not shown in the figure, while the LO-RANSAAC+ method still achieves highly accurate and stable results.

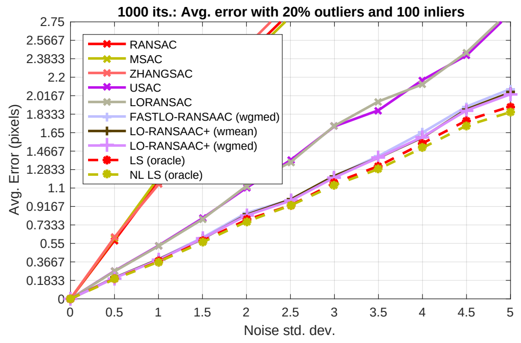
Interestingly, these RANSAC modifications, along with many others, could easily fit into the proposed RANSAAC algorithm. For example, both the $T_{d,d}$ test as well as the SPRT test could be used to early discard bad hypotheses, the MSAC scoring could be incorporated in the weighting of the input samples, and any improved sampling technique such as PROSAC, NAPSAC or GroupSAC would provide better hypotheses to perform the aggregation.



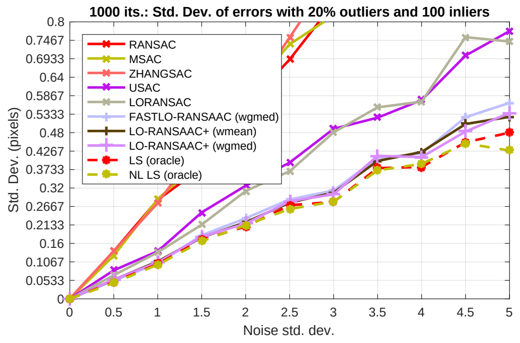
$\bar{E}$: 1000 its with 100/105 inlier ratio.



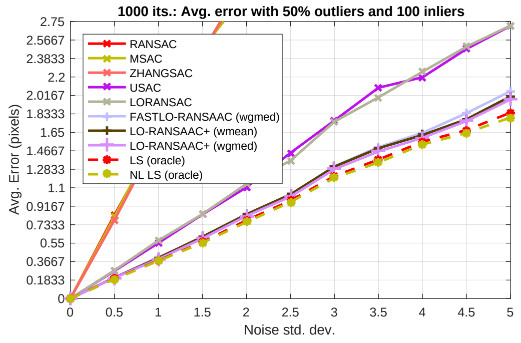
σ_E: 1000 its with 100/105 inlier ratio.



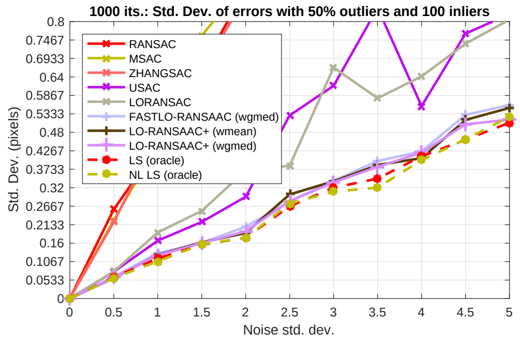
$\bar{E}$: 1000 its with 100/125 inlier ratio.



σ_E: 1000 its with 100/125 inlier ratio.

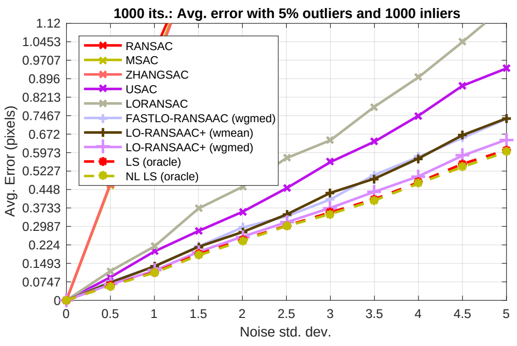


$\bar{E}$: 1000 its with 100/200 inlier ratio.

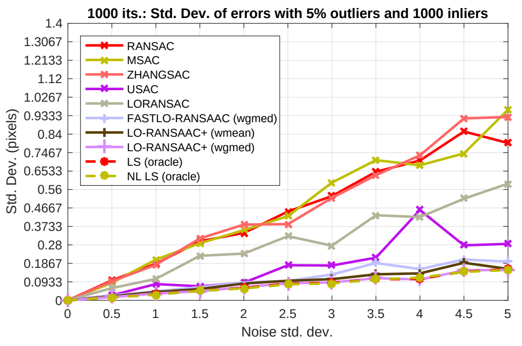


σ_E: 1000 its with 100/200 inlier ratio.

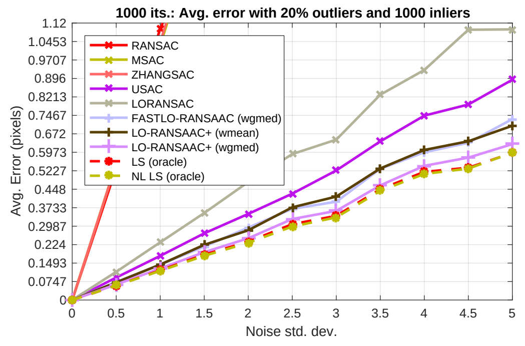
Figure 5.11 – Avg. errors $\bar{E}$ and their std. dev. σ_E by varying noise for several RANSAC and RANSAAC variants. Experiment: 100 inliers, 1000 iterations and different amounts of outliers. **First row**: 5% outliers. **Second row**: 20% outliers. **Third row**: 50% outliers.



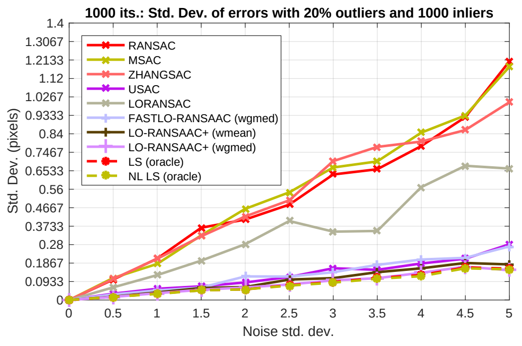
$\bar{E}$: 1000 its with 1000/1053 inlier ratio.



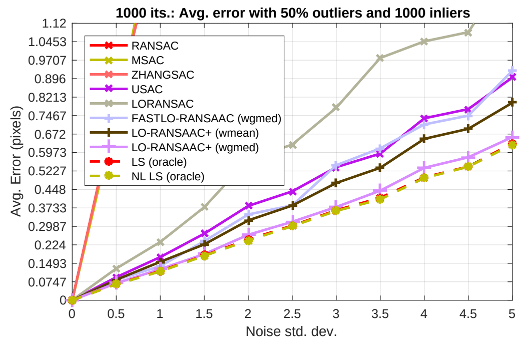
σ_E: 1000 its with 1000/1053 inlier ratio.



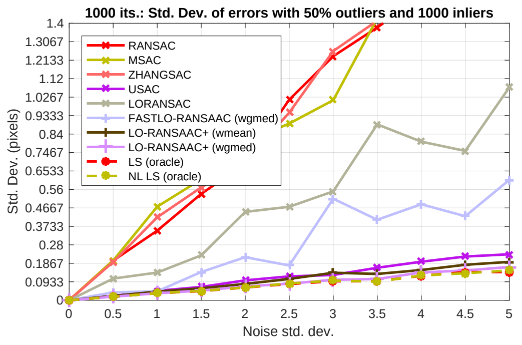
$\bar{E}$: 1000 its with 1000/1250 inlier ratio.



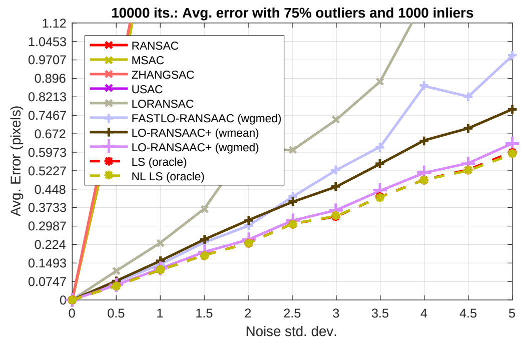
σ_E: 1000 its with 1000/1250 inlier ratio.



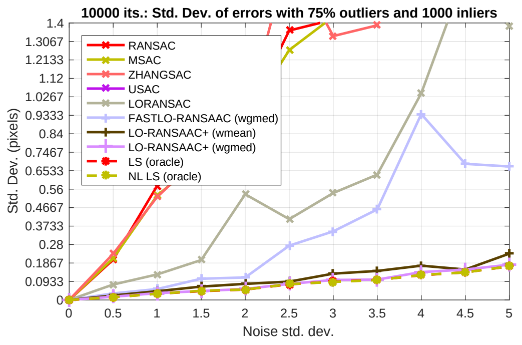
$\bar{E}$: 1000 its with 1000/2000 inlier ratio.



σ_E: 1000 its with 1000/2000 inlier ratio.



$\bar{E}$: 10000 its with 1000/4000 inlier ratio.



σ_E: 10000 its with 1000/4000 inlier ratio.

Figure 5.12 – Avg. errors $\bar{E}$ and their standard deviations σ_E by varying noise for several RANSAC and RANSAAC variants. Experiment: 1000 inliers, different amounts of outliers and doing 1000/10000 iterations. **First row**: 5% outliers. **Second row**: 20% outliers. **Third row**: 50% outliers. **Last row**: 75% outliers and 10k iterations.

Comparison by adding last-step minimization.

Despite the fact that RANSAC ignores the measurement noise on the input estimates, the authors suggest to perform an *a-posteriori* minimization on the calculated inliers after finishing the iterations. To this end, we compared every considered algorithm by applying this final minimization step, together with the already mentioned variants of our approach. For the simple sake of comparison, the minimization was always performed using the DLT algorithm, although results computed by minimizing the Sampson approximation to the geometric reprojection error showed no considerable difference. Note that for the USAC method the results got worse after performing the last step minimization, therefore the original method is included in the comparison.

Results shown in both figures 5.13 and 5.14 respectively do not differ considerably with the corresponding results when omitting the last step. This is because the most accurate methods all include the local optimization step, which already performs least squares minimization, although in a heuristic approach which improves the overall precision of the method. Therefore, even though we can observe that all methods not including this LO step improve considerably with respect to not applying last step minimization, they are still not able to compete with LO approaches. On the other end, LO-RANSAC sees a non-negligible improvement both in accuracy and stability by adding the final minimization, even though it still does not improve over USAC.

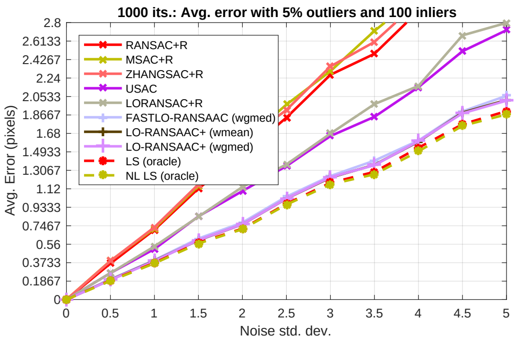
What happens with RANSAAC if adding a last-step least squares minimization.

We studied the performance of RANSAAC by adding a last-step least squares minimization using the inliers, computed by performing an extra evaluation of the dataset and counting the input points in a similar way as the standard RANSAC algorithm, using the δ_d parameter.

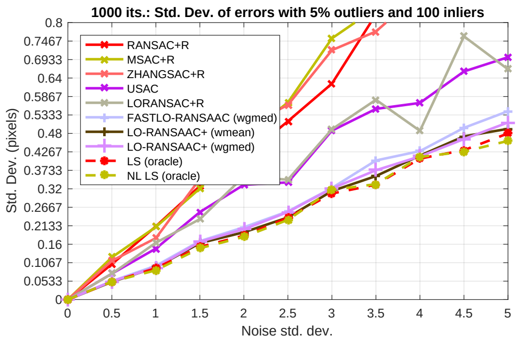
We show in Fig. 5.15 the difference between the error obtained using the original approach and the error after performing the last-step minimization. This means that when the values go below zero, performing the last step affects negatively the accuracy of the method. On the contrary, when values go higher than zero, it justifies the minimization. Several RANSAAC alternatives were evaluated for different outliers and iterations scenarios.

We deduce from the figures that when "enough" iterations are performed, the minimization should be avoided. Also, the number of iterations considered as "enough" depends on the evaluated variant of RANSAAC. For the case of 100 inliers the last-step minimization should definitely be avoided since 1000 iterations seem to be adequate for almost every method. The exception for this is the standard non-locally optimized RANSAAC method where for high outlier percentages, it shows a notable accuracy gain after performing this last step.

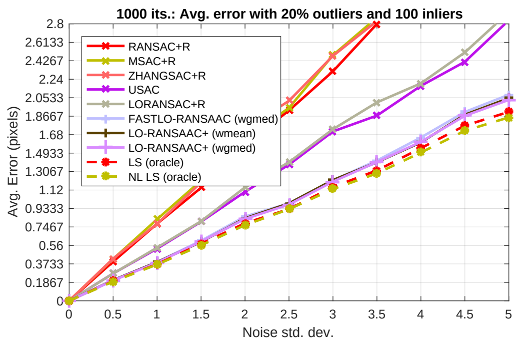
With more inliers in the input data, the use of a last-step minimization becomes more justified, particularly with high outlier percentages and lower amounts of iterations. However, it should be noted that both LO+ variants, which as shown before usually achieve the best results, seem to suffer by using this last step. This is even more evident for the $wgmed$ aggregation, the most accurate method. Therefore, to conclude, the efficiency of the method determines the amount of iterations to be considered as "enough", and when this is the case, performing a last-step least squares minimization should be avoided. In our results, the LO-RANSAAC+ method with $wgmed$ aggregation proved to be the most efficient with respect to the amount of iterations needed. In fact, although not practical, if it were possible to check whether applying this step improves over not doing it, then this would indicate if more iterations are required.



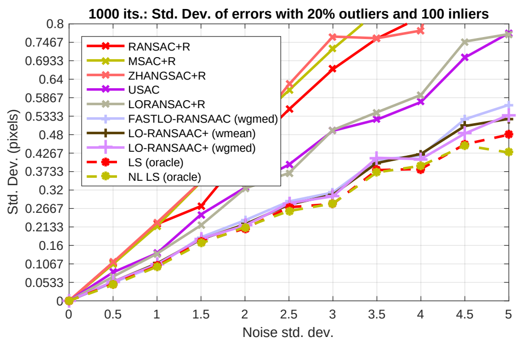
$\bar{E}$: 1000 its with 100/105 inlier ratio.



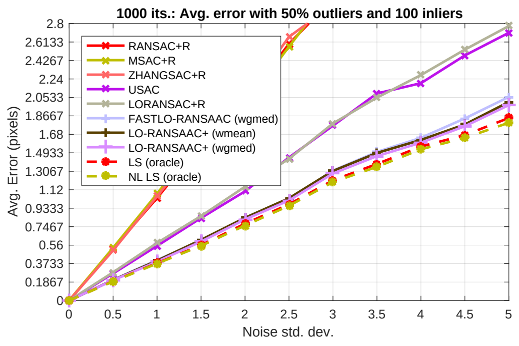
σ_E: 1000 its with 100/105 inlier ratio.



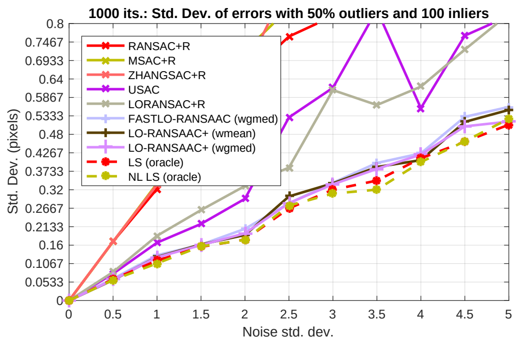
$\bar{E}$: 1000 its with 100/125 inlier ratio.



σ_E: 1000 its with 100/125 inlier ratio.

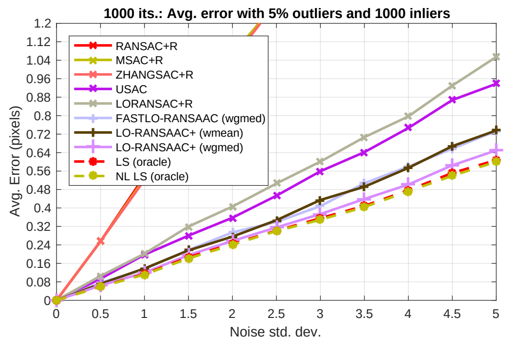


$\bar{E}$: 1000 its with 100/200 inlier ratio.

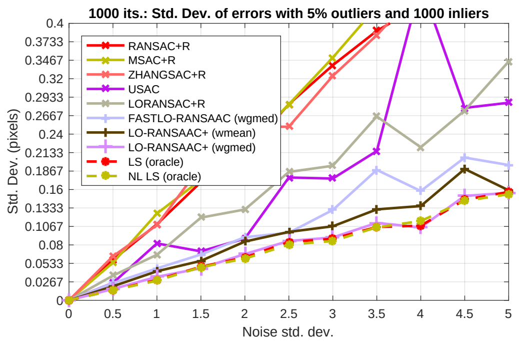


σ_E: 1000 its with 100/200 inlier ratio.

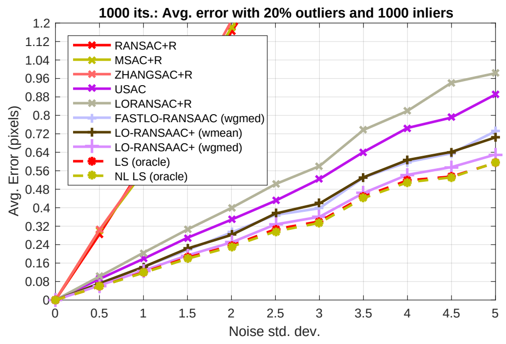
Figure 5.13 – Avg. errors $\bar{E}$ and their std. dev. σ_E by varying noise for several RANSAC and RANSAAC variants. Experiment: 100 inliers, 1000 iterations and different amounts of outliers. **First row**: 5% outliers. **Second row**: 20% outliers. **Third row**: 50% outliers.



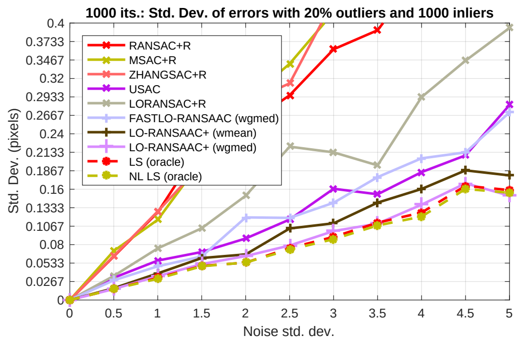
$\bar{E}$: 1000 its with 1000/1053 inlier ratio.



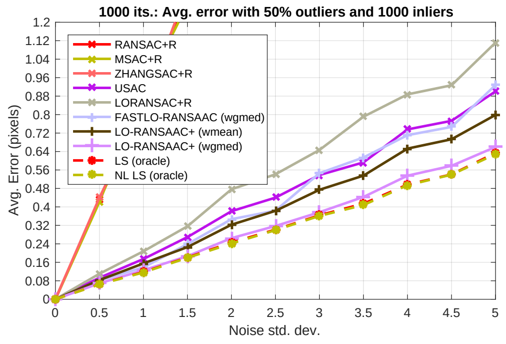
σ_E: 1000 its with 1000/1053 inlier ratio.



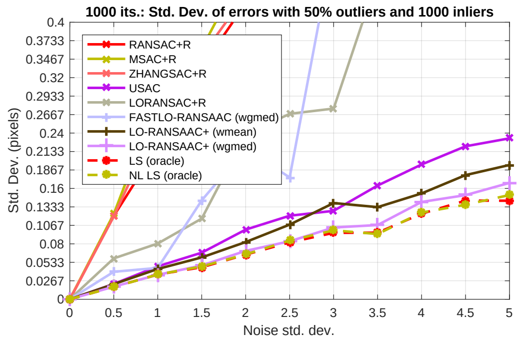
$\bar{E}$: 1000 its with 1000/1250 inlier ratio.



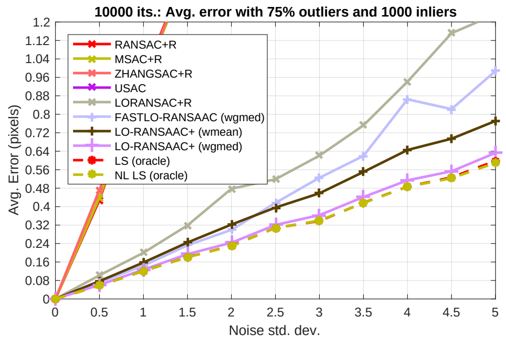
σ_E: 1000 its with 1000/1250 inlier ratio.



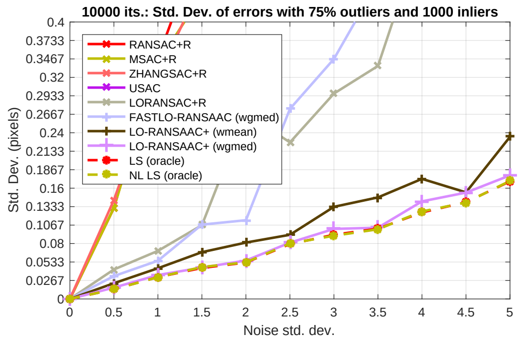
$\bar{E}$: 1000 its with 1000/2000 inlier ratio.



σ_E: 1000 its with 1000/2000 inlier ratio.



$\bar{E}$: 10000 its with 1000/4000 inlier ratio.



σ_E: 10000 its with 1000/4000 inlier ratio.

Figure 5.14 – Avg. errors $\bar{E}$ and their standard deviations σ_E by varying noise for several RANSAC and RANSAAC variants. Experiment: 1000 inliers, different amounts of outliers and doing 1000/10000 iterations. **First row**: 5% outliers. **Second row**: 20% outliers. **Third row**: 50% outliers. **Last row**: 75% outliers and 10k iterations.

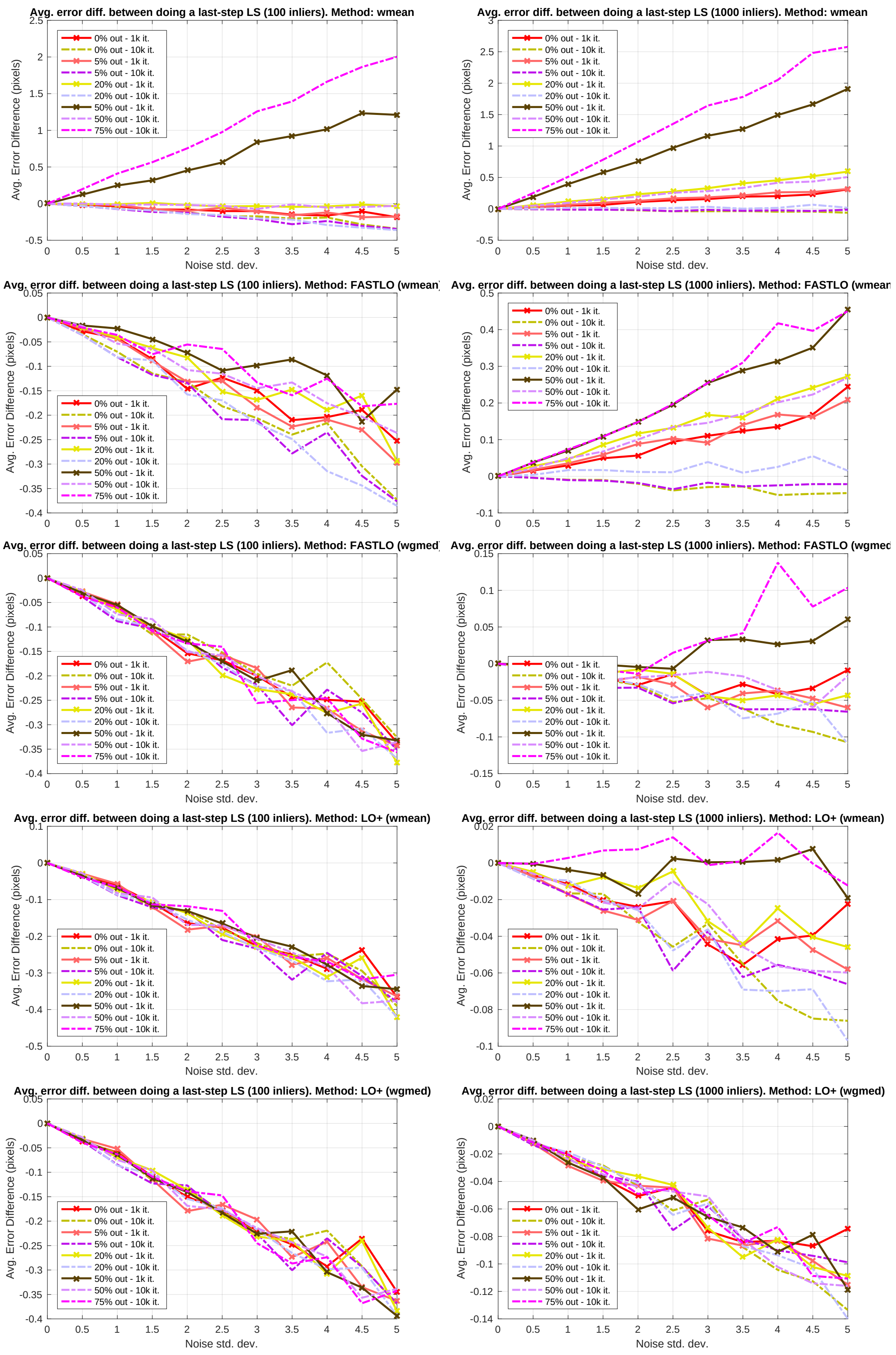


Figure 5.15 – Average difference of errors between different versions and aggregations for RANSAAC by adding a last step least squares minimization. Each row represents a different RANSAAC version. In order of appearance, they are: RANSAAC with $wmean$ aggregation, FASTLO with $wmean$, FASTLO with $wgmed$, LO+ with $wmean$ and finally LO+ with $wgmed$ aggregation. **Left**: 100 inliers. **Right**: 1000 inliers. Values lower than zero implies performance degradation after last-step minimization.

Performance under high outlier ratios

We tested our method under extreme situations with high outlier ratios. We evaluated the accuracy of different algorithms with 1000 inliers and 9000 outliers, using both 10000 and 20000 iterations. In table 5.4 results are shown under two noise levels: mild/moderate noise ($\sigma = 2$) or high noise ($\sigma = 5$). Even under these extreme conditions, while RANSAAC failed, as evidenced by both $wmean$ and $wgmed$ columns on the table, the LO-RANSAAC+ method was still able to achieve extremely low error values. On the contrary, the RANSAC method failed considerably, even after applying the last step minimization, while LO-RANSAC did not achieve errors of under a pixel in average even by using 20000 iterations. Furthermore, USAC did not return any results. Interestingly, the achieved precision of the LO-RANSAAC+ is close to the accuracy obtained by using an oracle, although there is still some room left for improvement.

σ	wmean		LO+ wmean		wgmed		LO+ wgmed		RANSAC+LS		LO-RANSAC		USAC		LS (oracle)
	10k	20k	10k	20k	10k	20k	10k	20k	10k	20k	10k	20k	10k	20k	-
2	621.70	87.13	0.53	0.46	303.63	26.60	**0.36**	**0.31**	49.15	19.90	16.68	1.76	–	–	**0.28**
5	45.61	48.48	1.35	1.38	33.16	42.18	**0.94**	**1.15**	23.94	27.69	4.35	6.30	–	–	**0.74**

Table 5.4 – High outlier ratio test: average errors of both aggregation methods for RANSAAC and LO-RANSAAC+, compared with RANSAC with last step least squares minimization (DLT), LO-RANSAC, USAC and computing LS on the inliers with an oracle. For noises $\sigma = 2$ and 5, each method was evaluated with 1000 inliers using both 10000 and 20000 iterations and 90% outliers. The averages are over 50 realizations.

Performance evaluation using random homographies

We also tested the robustness of the proposed approach to different homographies. We used the RANSAC toolbox of Zuliani available online [192] to simulate random homographies. These homographies H are calculated as

$$H = \begin{bmatrix} C & C\gamma - S & T_x \\ S & S\gamma - C & T_y \\ P_x & P_y & 1 \end{bmatrix}, \tag{5.15}$$

where

$$\begin{aligned} C &= s\cos(\phi), S = s\sin(\phi), \\ s &= 1 + (0.5 \cdot (rand - 0.5)), \\ \phi &= 30\pi(rand - 0.5), \\ T_x &= randn, T_y = randn \\ \gamma &= 0.05 \cdot randn, \\ P_x &= 1e^{-7} \cdot randn, P_y = 1e^{-7} \cdot randn, \end{aligned}$$

$rand$ is a random number uniformly sampled in the $[0, 1]$ interval and $randn$ is a random number sampled using the standard normal distribution. Note that both $rand$ and $randn$ refer to different random values and should not be interpreted as the same variable/number.

Then, we evaluated both LO-RANSAAC+ aggregation methods together with USAC [127] which proved to be the best state-of-the-art approach during our previous tests and again, aided by an oracle, the least squares minimization of the Sampson approximation to the geometric reprojection error. The experiment again consisted in averaging the error over 100 trials by varying the noise and the inlier/outlier configuration. The amount of iterations was set to 1000. Differing from the previous experiments where both the homography and the inliers were fixed, a new random homography and random inliers sampled from it were generated for each trial. Then, as before, outliers were injected in the input data by picking random positions on both images, and finally Gaussian white noise was added to the final points.

Results confirmed the improvement of LO-RANSAAC+ over USAC as shown in Fig. 5.16, validating the versatility of the approach. It should be noted that with low inlier counts, $wmean$ obtains more accurate and stable results (i.e. with lower standard deviation) than $wgmed$. This is because for smaller datasets, the RANSAC approach tends to make less mistakes when discriminating the inliers from the outliers, and so, averaging only good hypotheses achieves better results than using their median. Indeed, as the amount of matches increases, the probability of including outliers in the aggregation gets higher, therefore taking the median becomes more robust improving over the average.

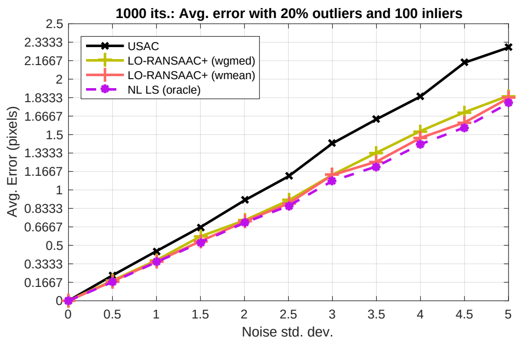
As a final test, instead of fixing the amount of iterations performed, we evaluated the same methods using adaptive termination (see sec. 5.1.1). Again the maximum number of iterations was set to 1000, however in practice the algorithm performed around 12 and 85 iterations for 20% and 50% outliers respectively, together with a single LO step. Impressively, even by taking such low amount of iterations, the method still managed to improve over USAC. However, this did not occur using $wmean$ aggregation for the case of 1000 inliers, due to its already observed necessity to get more models in order for the average to be closer to the optimal solution. A possible workaround for this, validated empirically, was to double the theoretical number of iterations given by Eq. 5.5. On the contrary, the LO-RANSAAC+ method with $wgmed$ aggregation obtained more accurate results than USAC on every case.

Lastly, we observed some instability for our approach when using 100/200 inlier ratio and $\sigma = 5$. We believe this peak must have been caused by some homography which was not correctly averaged. In fact, some degenerate cases of homographies would not be correctly aggregated by using the approach described in section 5.3.1. By considering the horizon of an homography as all points lying in the line where the scalar product of the third row of the homography matrix by any of these points gives zero, then data points close to this line will be projected extremely far away from the second image, therefore their average will not be precise. What is worse, because of the noise on the input data, each "good" homography, i.e. sampled from all inliers, will be noisy. Then it is possible for one of the preselected points to lie close but on different sides of the horizon line for two different valid homographies, so when projected, they end up on opposites sides of the image and their average does not follow the geodesics of the space. A more in-depth study on this subject is envisaged in the near future. Nevertheless, it should be pointed out that for most "real life" homographies, the probability of occurrence of such degenerate situations is very low.

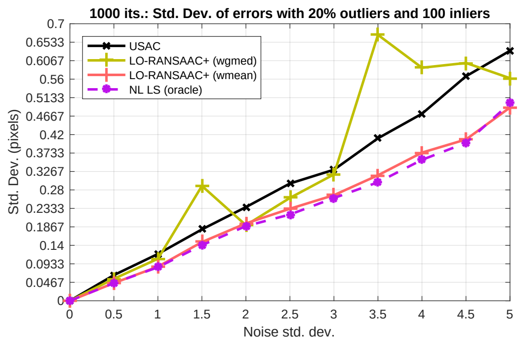
5.4.3 Application 2: Estimating Homography+Distortion

The recently proposed method [87] to estimate an homography and a distortion coefficient for each image was evaluated to test the robustness of the proposed approach against different model "noises". In this case, the available algorithm to compute such model assumes only 5 input matches, and it is not possible to perform a least squares like minimization based on the authors supplied code. Therefore, this is an interesting case in which, evidently, no local optimization is possible as is, and no last step minimization could be applied. Therefore, we restricted ourselves to evaluating the original RANSAAC approach using $wmean$ aggregation, comparing it with the RANSAC method, as used in the author's work.

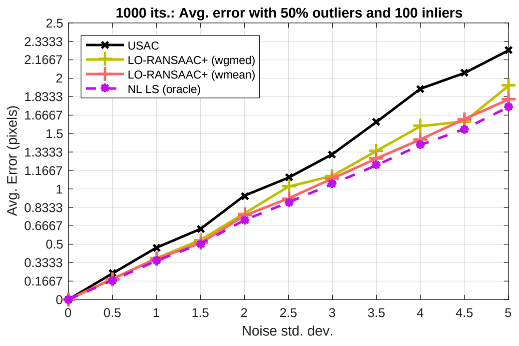
As seen in Fig. 5.18, the method outperforms RANSAC in every evaluated experi-



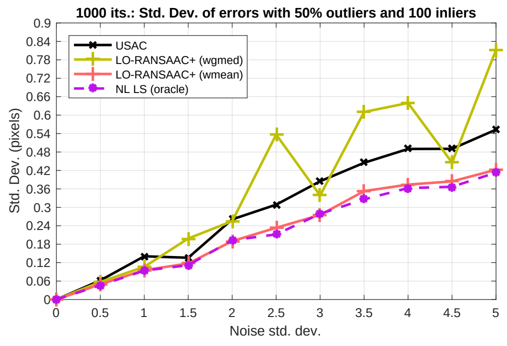
$\bar{E}$: 1000 its with 100/125 inlier ratio.



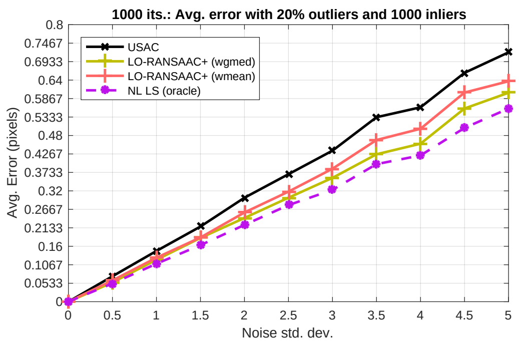
σ_E: 1000 its with 100/125 inlier ratio.



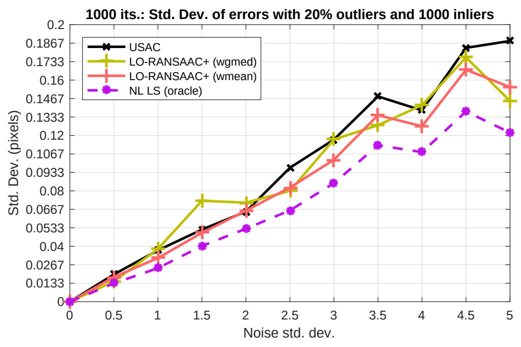
$\bar{E}$: 1000 its with 100/200 inlier ratio.



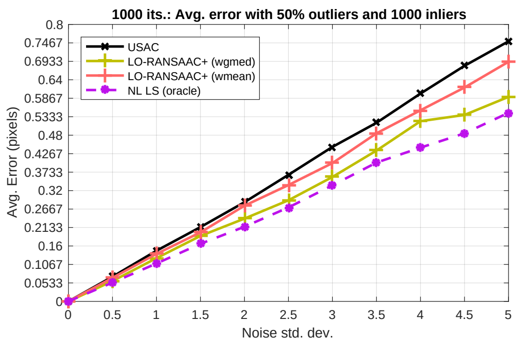
σ_E: 1000 its with 100/200 inlier ratio.



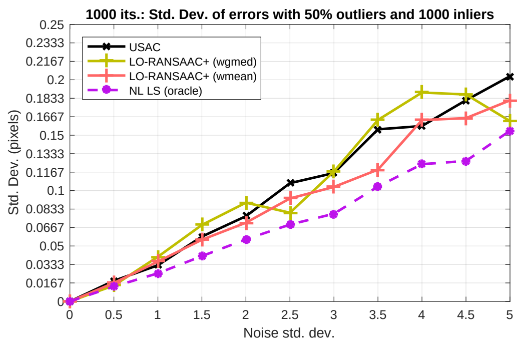
$\bar{E}$: 1000 its with 1000/1250 inlier ratio.



σ_E: 1000 its with 1000/1250 inlier ratio.

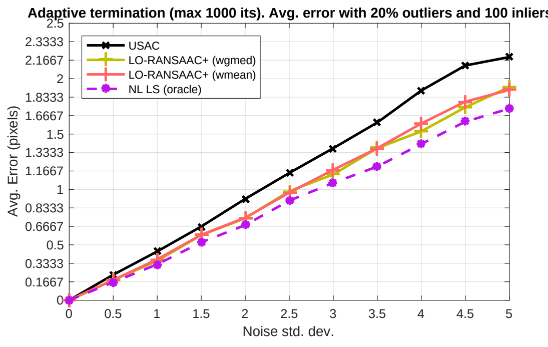


$\bar{E}$: 1000 its with 1000/2000 inlier ratio.

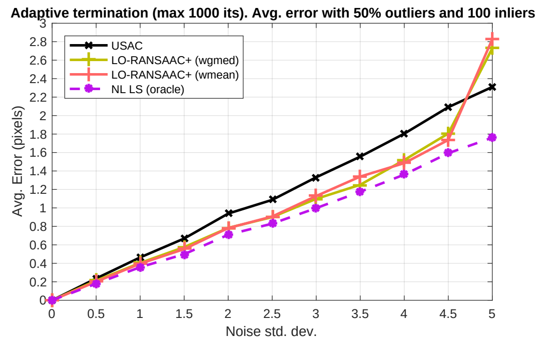


σ_E: 1000 its with 1000/2000 inlier ratio.

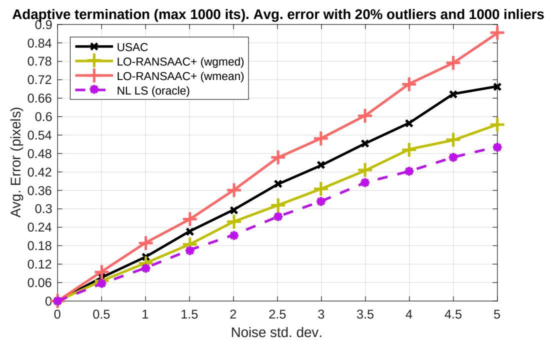
Figure 5.16 – Avg. errors $\bar{E}$ and their standard deviations σ_E by varying the homography and the input matches for each trial, using 1000 iterations. **First row**: 20% outliers and 100 inliers. **Second row**: 50% outliers and 100 inliers. **Third row**: 20% outliers and 1000 inliers. **Fourth row**: 50% outliers and 1000 inliers.



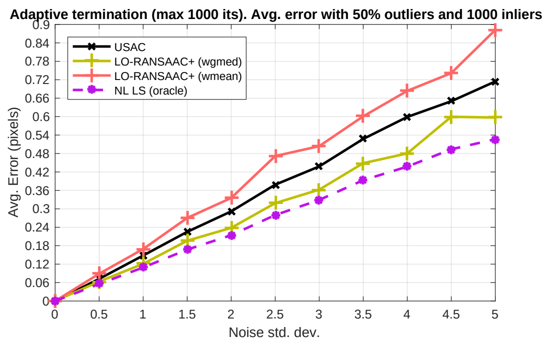
$\bar{E}$: 1000 its with 100/120 inlier ratio.



$\bar{E}$: 1000 its with 100/200 inlier ratio.



$\bar{E}$: 1000 its with 1000/1200 inlier ratio.



$\bar{E}$: 1000 its with 1000/2000 inlier ratio.

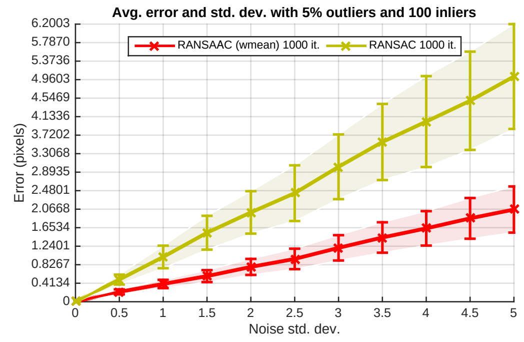
Figure 5.17 – Avg. errors $\bar{E}$ by varying the homography and the input matches for each trial, using adaptive termination. **Top**: 100 inliers. **Bottom**: 1000 inliers. **Left**: 20% outliers. **Right**: 50% outliers.

ment on both accuracy and stability, and the difference in performance gets higher as the amount of iterations increase, particularly for higher outlier percentages. This experiment proves the versatility of our approach compared to other more complex models.

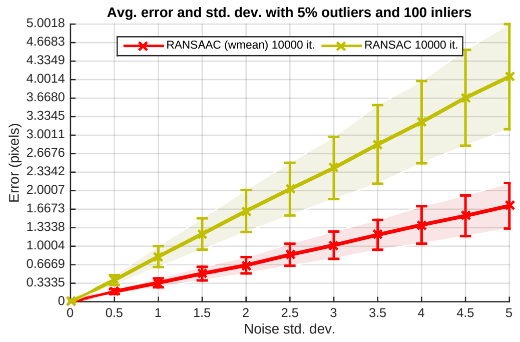
5.5 Concluding Remarks

In this chapter we introduced a simple, yet powerful method that clearly improves on other RANSAC strategies in presence of measurement noise, by combining the random consensus idea using samples with minimal cardinality with a statistical approach performing an aggregation of estimates. This comes with an almost negligible extra computational cost. Due to its statistical nature, it also facilitates the always difficult task of parameter selection. What is more, most of RANSAC enhancements easily fit into the proposed method. By adding local optimization to RANSAAC, the resulting accuracy and stability proved to be top-notch surpassing every state-of-the-art algorithm. Moreover, it succeeded in accurately estimating models under 90% outliers, a situation where most current state-of-the-art approaches fail. Nevertheless, other RANSAC extensions could still be used to improve even further the results. We propose the following. First, to modify the evaluation step by using the method proposed in [129] or to replace the hard thresholding-based scoring with a more flexible approach such as the used in MSAC [172]. Second, to modify the random sampling strategy by using the PROSAC approach [36] or the more recent proposed guided sampling methods such as residual sorting [32] or higher than minimal subset sampling [164]. Finally, to accelerate the method by discarding hypotheses before computing their score, by using, for example, the randomized RANSAC approach [37]. This leaves space for further improvement.

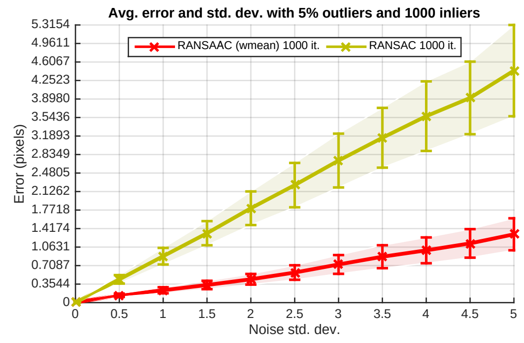
As a future work, a more thorough evaluation of the proposed 2D transformation



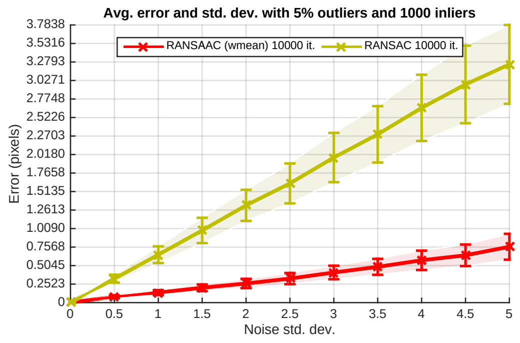
1000 its - 100/105 inlier ratio



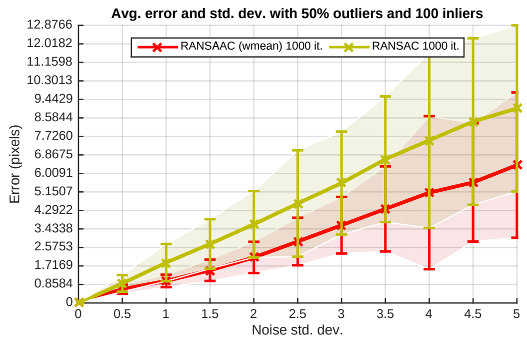
10000 its - 100/105 inlier ratio



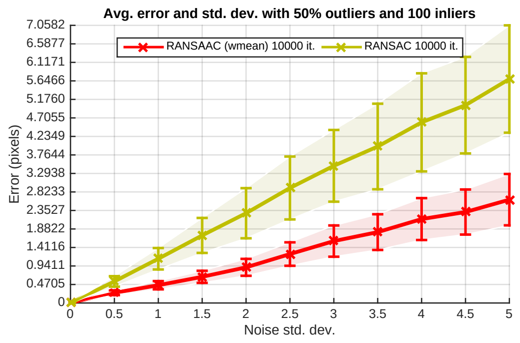
1000 its - 1000/1050 inlier ratio



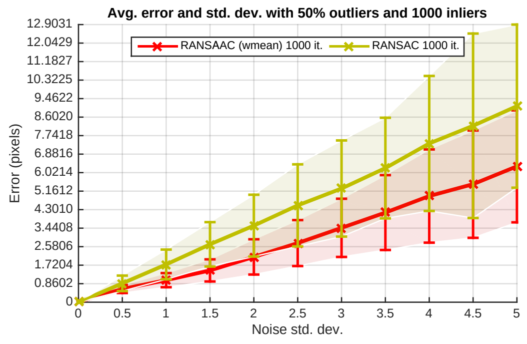
10000 its - 1000/1050 inlier ratio



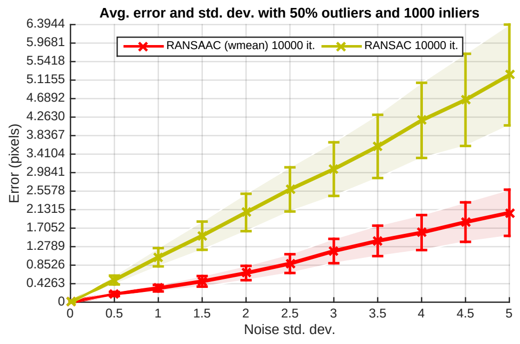
1000 its - 100/200 inlier ratio



10000 its - 100/200 inlier ratio



1000 its - 1000/2000 inlier ratio



10000 its - 1000/2000 inlier ratio

Figure 5.18 – Avg. errors $\bar{E}$ and their std. dev. σ_E by varying noise for RANSAC and RANSAAC (weighted mean) with different iterations (1000 and 10000), quantity of inliers (100 and 1000) for the H_5 transformation [87]. **Top**: 5% outliers. **Bottom**: 50% outliers.

averaging technique is required. As seen in the experiments, the approach seems to fail for some degenerate homographies, and a workaround should be explored. Particularly, since the problem appears to be caused by averaging models which project points far away from the second image, or on inverted positions, it seems appropiate in this case to avoid the aggregation on that particular data point or to treat these cases exceptionally.

Finally, we plan to extend the method to the case of epipolar geometry estimation as well as to investigate its statistical justification. We already tested our approach for essential matrix estimation, by averaging on the essential manifold [174], and achieved improved results over RANSAC. Another objective would be to adapt this method to multi-model detection, which is envisaged using local aggregation in a mean-shift [31] fashion.

Chapter 6

Conclusions

This thesis studied a major problem in image processing: image registration. Aligning noisy images of the same scene with high accuracy is often required in several applications, particularly in remote sensing. Capturing high resolution images of Earth from satellites orbiting at 700km in space under harsh conditions is a challenging task that begun as early as the 1960s. Thanks to the evolution of technology, the Internet, GPS navigation and commercial applications such as Google Earth, Earth-observation through satellites has flourished in the last decade, increasing the demand for higher resolutions and better image quality. In this thesis we proposed effective numerical solutions for two possible applications that aim at meeting these demands using currently available on board hardware, which enables them to be implemented on future Earth-observation satellites. Indeed, both proposed solutions have been considered by CNES, are now implemented in hardware and are being thoroughly evaluated.

From a scientific point of view, it was noted throughout this thesis that a shift estimation method based on image gradients, dating back to the 80s, had been unjustly discarded or ignored by the remote sensing community. This method is proved by the present dissertation to be extremely precise if carefully refined, improving over every state-of-the-art method used for the studied topics. Furthermore, it is computationally cheap, which makes it an outstanding candidate to be implemented on low processing power satellite hardware.

This method was indeed proposed for measuring the wavefront aberrations, caused by the deformation of the main mirror in Earth-observation satellites, through a Shack-Hartmann wavefront sensor. It achieved results that improve over the current state-of-the-art by profiting from the small displacements present in a closed-loop system. As future work, it would be interesting to evaluate the presented method in an adaptive optics context, in which the wavefront aberrations are caused by the light passing through the earth's atmosphere. In such conditions, these aberrations are typically of higher magnitudes, implying larger displacements that complicate the use of gradient-based shift estimation approaches. However, since this application is ground-based, the available computational resources are considerably higher, which would allow to implement the proposed solution by increasing the amount of scales and/or iterations.

The last contribution of this thesis is related to RANSAC, undoubtedly the most popular computer vision method to robustly estimate transformations under the heavy presence of outliers and measurement noise. This iterative method generates multiple random hypotheses and returns the one which gives the maximum consensus among the input data. Each hypothesis is generated using minimalistic samples taken randomly. Therefore, often many hypotheses are drawn using inlier samples, thus generating models that approximate the one being sought. However, only the hypothesis with the largest consensus is kept, discarding all others. This observation led to RANSAAC, a framework that by aggregating such hypotheses, systematically improves over RANSAC. Furthermore, aggregating 2D transformations present in real-life applications adds a negligible cost to the overall computational cost of RANSAC. Therefore, by combining the random sampled consensus idea with a statistical approach that aggregates models, a new variant of RANSAC was proposed that improves over every state-of-the-art method. Envisaged for the future is the application of this framework in the estimation of other models. In particular, to improve the estimation of the essential/fundamental matrix which relate corresponding points in stereo images. To this end, possible methods for essential/fundamental matrix aggregation are being studied.

Appendix A

Fast Interpolation Methods. Focus on Image Resampling for Shift Estimation on Satellite Images.

A full study of interpolation methods was presented in this appendix. Fourier interpolation gives the best results without a heavy computational cost, although it is still more expensive than other methods such as nearest neighbor, bilinear and biquadratic. Among low cost methods, the one which gives the best results seems to be Schaum2. Other approaches that prefilter the data before interpolating usually achieve improved results. Between them, oMoms3 and BSpline2 or BSpline3 seems to be a good trade between cost and precision. If low computational cost is required, both bicubic and biquadratic methods should be considered. Bilinear interpolation usually produced blurred results, and its use is not advised unless extreme speed constraints. Finally, all interpolation methods presented here can be evaluated online on the IPOL Journal [62].

A.1 Introduction

Interpolation is the process of estimating the intermediate values of a continuous function from discrete finite samples. It is frequently used in image processing to magnify or reduce images and to correct spatial distortions. It is also used in problems where sub-pixel precision is required, such as our case of study. In order to perform image registration, an interpolation method is generally also required to resample one of the images on the grid of another. However, as it is required that the algorithm is run on a satellite in real-time, the complexity of the interpolation algorithm should be carefully taken into account.

In this report, we study several methods for image interpolation, and their implementations in order to achieve good interpolation results while minimizing the computational cost. A complete panorama of interpolation methods will be given, with their respective complexities and interpolation results.

A.1.1 Problem statement

Given an input image f with uniformly-sampled pixels $f_{i,j}$, the goal of interpolation is to find a function $g(x, y)$ satisfying

$$f_{i,j} = g(i, j) \;\; \forall i, j \in \mathbb{Z} \tag{A.1}$$

such that g approximates the underlying function from which v was sampled. Another way to interpret this is that v was created by subsampling, and interpolation attempts to invert this process.

We will first discuss linear methods for interpolation, including nearest neighbor, bilinear and bicubic interpolation. Then we will discuss splines and finally we will explain sinc interpolation and its approximations. We focus on separable interpolation, so most of what is said to one-dimensional interpolation applies as well to N-dimensional separable interpolation. For example, figure A.1 shows how a $1D$ interpolation method is extended in $2D$, by first interpolating in the horizontal direction and then interpolating each of these results in the vertical direction.

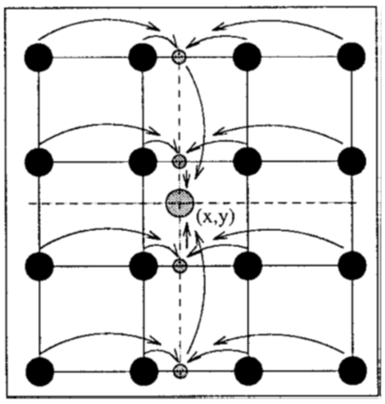


Figure A.1 – Interpolation of the point (x, y) in a 4×4 neighborhood.

A.1.2 Interpolation function and interpolation kernels

For equally spaced data in 1-D, if f is a sampled function and g is the corresponding interpolation function, then many interpolation functions can be written in the form

$$g(x) = \sum_k c_k u\left(\frac{x - x_k}{h}\right) \tag{A.2}$$

where h represents the sampling increment, the x_k's are the interpolation nodes and u is called the *interpolation kernel*. The c_k's, as will be explained later, are parameters that depend upon the sampled data f_m, and are selected so that the interpolation condition (A.1) is satisfied. For our particular problem, we will assume $h = 1$ and that the x_k's are the integers, which leads to the following interpolation function

$$g(x) = \sum_{k \in \mathbb{Z}} c_k u(x - k) \tag{A.3}$$

A.1.3 Interpolation kernel characteristics

The interpolation kernel in (A.3) converts discrete data into continuous functions by a convolution-like operation. It has a significant impact on the numerical behavior of interpolation functions and due to their influence on accuracy and efficiency, they can be effectively used to create new interpolation algorithms.

Interpolating function

Among the requirements of an interpolating function, the most important one is that it has to be symmetric, in order to ensure invariance to mirror images. Another desirable property is that $u(0) = 1$ and $u(k) = 0$ when k is any nonzero integer. If this is the case, then $u(x - k)$ is zero unless $x = k$, then

$$g(j) = \sum_{k \in \mathbb{Z}} c_k u(j - k) = c_j$$

and since the interpolation condition requires that $g(j) = f_j \ \forall j \in \mathbb{Z}$, then $c_j = f_j$. In other words, the c_k's in (A.3) are replaced by the sampled data, which implies a substantial computational improvement over interpolation schemes such as B–splines, where the interpolation kernel is not zero for nonzero integers and thus, the c_k's must be determined either by solving a tridiagonal matrix problem or by pre-filtering the input data as it will be explained later in this report. If the interpolation kernel in (A.3) holds this property, it is said to be an interpolating function, and can be written as:

$$g(x) = \sum_{k \in \mathbb{Z}} f_k u(x - k) \tag{A.4}$$

If the interpolation kernel does not hold this property, then it is called approximator [89].

Kernel support

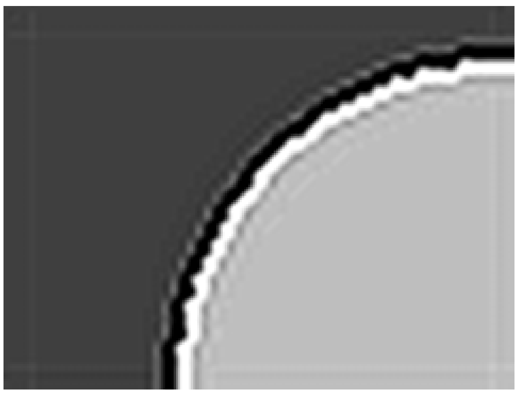
The support of the kernel refers to how many data samples it requires in order to generate a result. The larger the support, the slower the interpolation. However, including more samples in the calculations generates improved interpolation results, as will be shown. So if S is the support of the kernel, then

$$g(x) = \sum_{k=-\infty}^{\infty} c_k u(x - k) = \sum_{k=-\frac{|S|}{2}}^{\frac{|S|}{2}} c_k u(x - k)$$

implying fewer calculations to perform.

Frequency response and ringing artifacts

As it will be seen in the results, kernels with considerable side lobes in the spatial domain and high support usually produce visible ringing artifacts, in particular around high frequency edges of the image, as illustrated in figure A.2. This is caused by a brutal frequency cutoff as the one obtained by convolution with the Sinc kernel. To avoid ringing effects, the filter must decay slowly and tend to zero as the frequency increases. So it must have a reverted bell shape like, for example, the Butterworth filter. This ringing artifacts are simply explained by the Gibbs phenomenon [77].


(a) Image showing ringing artifacts. 3 levels on each side of transition: overshoot, first ring, and (faint) second ring.

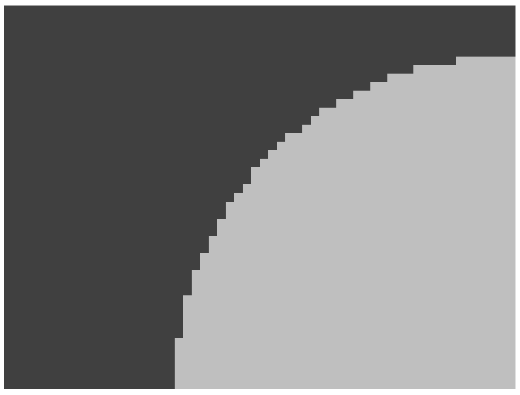

(b) Original image

Figure A.2 – Ringing artifacts example

Figure A.3 illustrates why smaller side lobes in the spatial domain are desired when strong intensity changes occur in an non-bandlimited signal. Since there is a strong discontinuation in the signal, when the interpolation kernel has significant side lobes, the influence of those side lobes increases the interpolation error, while, for example, for the simple linear interpolation method, since no side lobes in the kernel exist (also due to its compact support), smaller errors are introduced in the results. This example suggests two possible solutions: either limiting the support of the interpolation kernel (which may lead to worse interpolation results, specially if we try to interpolate high order polynomials), either minimizing the side lobes of the interpolation kernel.

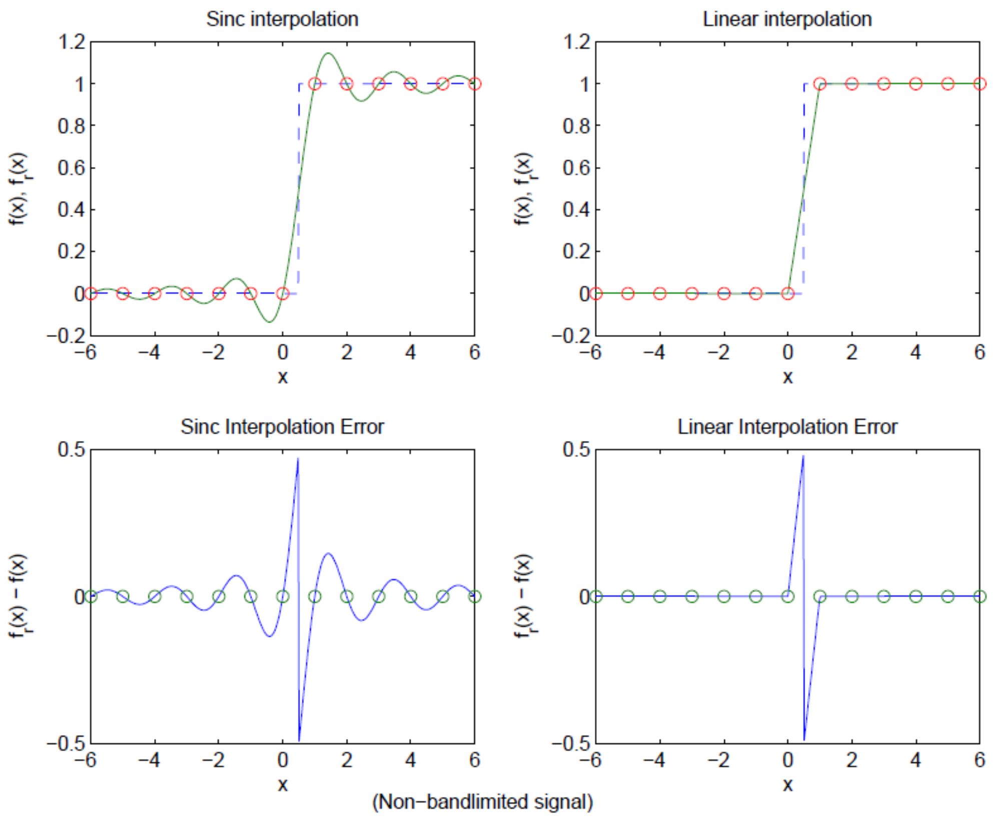


Figure A.3 – Side lobes impact on a non-bandlimited signal (Heaviside function) by comparing Sinc interpolation (left) to Linear interpolation (right). Red circles represent the samples, the dotted lines show the original signal and the green lines are the interpolation results. The strong discontinuity in the signal implies ringing on the results using sinc interpolation.

Approximation order

An interpolation method has approximation order J if it reproduces polynomials up to degree $(J-1)$.

Let g be the interpolated function and u be the interpolation kernel defined as in the previous section. The Strang–Fix conditions [166] provide the following equivalent conditions to determine the approximation order:

- u has approximation order J; i.e., there exist weights $p_n[k]$ such that

$$x^n = \sum_{k\in\mathbb{Z}} p_n[k]u(x-k)$$

,

- $\hat{u}(0) = 1,\ \hat{u}^{(j)}(2\pi k) = 0\ \forall k \in \mathbb{Z}_*, j = 0, \dots, J-1$
- $\sum_{k\in\mathbb{Z}}(x-k)^j u(x-k) = \mu_j\ \forall x \in \mathbb{R}, j = 0, \dots, J-1$

where μ_j depends on j only, $\hat{u}^{(j)}$ denotes the jth derivative of $\hat{u}$ and $\mathbb{Z}_* := \mathbb{Z}\backslash\{0\}$.

Two-Step Interpolation

Besides being a desirable property due to speed reasons, that u in (A.3) must be interpolating can be an inconvenient restriction. It complicates the problem of finding a kernel that also satisfies other design objectives. It can also be an obstacle computationally. In some cases such as spline interpolation, u has infinite support, yet the interpolant can be expressed as a linear combination of compact support functions (the B–splines).

Suppose again that a one-dimensional sequence of samples $(f_m)_m$ is to be interpolated and that the basis function $\varphi(t)$ is the B–spline kernel. Then,

$$g(x) = \sum_{k\in\mathbb{Z}} c_k \varphi(x-k) \tag{A.5}$$

where the coefficients c_k are selected such that

$$f_m = \sum_{k\in\mathbb{Z}} c_k \varphi(m-k) \tag{A.6}$$

This means that interpolation is a two-step procedure: first, one finds the coefficients c_k satisfying (A.6) c_k and second, the interpolant is constructed as $\sum c_k\varphi(t-k)$.

To solve for the coefficients, equation (A.6) is nothing but a linear system of equations in terms of the unknown coefficients c_k. Based on equation (A.6), this system seems to have infinite unknowns (since $k \in \mathbb{Z}$) and infinite equations (because all arguments m are considered). However, in practice, m is limited to the size of the image, and k is as well limited because the interpolation kernel has finite support. Then the resulting problem has the form of $c = P^{-1}f$ and involves inverting the matrix P, which is the discretization of the interpolation kernel φ so that $P_k = \varphi(k)$. In fact, in two dimensions P is block tridiagonal and some specific methods are available for inverting this type of matrices, which often imply an elevated computational cost. Note that there is one c_k per pixel, implying the rank of the matrix to be equal to the size of the image.

Another strategy is to notice that $f_m = \sum c_k\varphi(m-k)$ is equivalent to a discrete convolution $f_m = (c * p)_m$ where $p_m = \varphi(m)$, that is, the discretization of the continuous function φ. Since convolution is nothing else but filtering, discrete filtering can be an alternative solution to matrix inversion for the determination of the coefficients c_k. This suggests solving for c_k under the Z-transform as

$$f(z) = c(z)p(z) \Rightarrow c(z) = \frac{1}{p(z)} f(z). \tag{A.7}$$

Let $(p)^{-1}$ denote the convolution inverse of p, which is the sequence such that $p * (p)^{-1} = \delta$ where δ is the unit impulse ($\delta_0 := 1$ and $\delta_m := 0$ otherwise). Using $(p)^{-1}$ we obtain c_k by prefiltering as

$$c = (p)^{-1} * f. \tag{A.8}$$

A unique convolution inverse exists in many practical cases of interest. If it exists and φ is real and symmetric, which is the case for every considered interpolation kernel analyzed in this report, then $(p)^{-1}$ can be factored into pairs of recursive (infinite impulse response) filters, allowing for an efficient in-place calculation.

The freedom to select an interpolation kernel is much larger after the interpolation constraint has been removed; this opens up the use of kernels that offer much better performance (for a given computational cost) than kernels that hold the interpolation constraint.

Kernel Normalization

Another basic and desirable quality of an interpolation method is that it reproduces constants, which is equivalent to

$$\sum_{k\in\mathbb{Z}} u(x-k) = 1 \;\; \forall x \in \mathbb{R} \tag{A.9}$$

If the kernel does not reproduce constants, it can be the normalized to fix this by setting

$$\tilde{u}(x) = \frac{u(x)}{\sum_{m\in\mathbb{Z}} u(x-m)} \tag{A.10}$$

provided that the denominator does not vanish. Normalization ensures that the interpolation weights sum to 1. Interpolation with the normalized kernel reproduces constants: suppose $f_k = c$, then

$$g(x) = \sum_{k\in\mathbb{Z}} f_k \tilde{u}(x-k) = \sum_{k\in\mathbb{Z}} c\tilde{u}(x-k) = \frac{\sum_{k\in\mathbb{Z}} cu(x-k)}{\sum_{m\in\mathbb{Z}} u(x-m)} = c$$

A.2 Interpolation Methods

A.2.1 Ideal Interpolation

Following the sampling theory, the scanning of a continuous image $f(x,y)$ yields infinite repetitions of its continuous spectrum $F(u,v)$ in the Fourier domain, which do not overlap since the Nyquist criterion is satisfied. If this is so, and only then, the original image $f(x,y)$ can be reconstructed perfectly from its samples $f_{i,j}$ by multiplication of an appropriate rectangular prism in the Fourier domain. The 1D ideal interpolation equals the multiplication with a rectangle function in the Fourier domain and can be realized in the spatial domain by a convolution with the sinc function:

$$u^{Ideal}(x) = \frac{\sin(\pi x)}{\pi x} = sinc(x). \tag{A.11}$$

The kernel $u^{Ideal}(x)$, also known as the kernel of the Whittaker–Shannon interpolation [148], has infinite support and since $u^{Ideal}(0) = 1$ and $u^{Ideal}(k) = 0$ when k is any nonzero integer, it is called an interpolator kernel. The interpolation is such that $g(x)$, the resulting interpolation function, is bandlimited which means its Fourier transform is zero for frequencies outside of $[-\frac{1}{2}, \frac{1}{2}]$. This interpolation is also often called "sinc interpolation" or "Fourier zero-padding", since to perform it in the frequency domain requires padding the image with zeros outside its Fourier spectrum.

The powerful property of sinc interpolation is that it is exact for bandlimited functions, that is, if $f_{m,n} = f(m,n)$ with

$$\int\int f(x,y)e^{-2\pi i(x\xi+y\nu)}dxdy = 0 \; \forall \; |\xi| \text{ or } |\nu| \geq \frac{1}{2}, \tag{A.12}$$

then the interpolation reproduces f.

However, although it provides an exact reconstruction of the original $f(x,y)$, it is spatially unlimited and cannot therefore be performed in the spatial domain unless some periodicity is assumed on the input image. Besides, the main disadvantage of sinc interpolation is that in aliased images it produces significant ripple artifacts (due to the so called Gibbs phenomenon) in the vicinity of image edges. This is because $sinc(x)$ decays slowly, at a rate of $1/x$, so the damage from meeting an edge is spread throughout the image. This is justified by the fact that bandlimitedness cannot be expected on every real-word image. As will be explained later, several methods try to limit the ripple artifacts by windowing the kernel.

In some ways, sinc interpolation is the ultimate interpolation. It is exact for bandlimited functions, so the method is very accurate on smooth data. Additionally, Fourier zero-padding avoids staircase artifacts, it is effective in reconstructing features at different orientations.

A.2.2 Nearest Neighbor Interpolation

The easiest and fastest way to perform interpolation using a spatially limited kernel is by the nearest neighbor method. The value $g(x)$ is given by the sample that is the closest to position x, which is $f_{[x]}$ where [.] denotes rounding to the nearest integer. For this reason, nearest neighbor interpolation is sometimes called "pixel duplication". The interpolation kernel for nearest neighbor is

$$u(x) = \begin{cases} 1, & 0 \leq |x| \leq 0.5 \\ 0, & elsewhere. \end{cases} \tag{A.13}$$

This kernel requires $N = 1$ supporting points and does reproduce constants.

A.2.3 Bilinear Interpolation

For separated bi-linear interpolation, the values of both direct neighbors are weighted by their distance to the opposite point of interpolation. Therefore, the linear approximation of the sinc function follows the triangular function

$$u(x) = \begin{cases} 1 - |x|, & 0 \leq |x| < 1 \\ 0, & elsewhere. \end{cases} \tag{A.14}$$

Within each cell $[m, m+1] \times [n, n+1]$, the interpolation is a convex combination of the samples located at the cell corners $v_{m,n}, v_{m+1,n}, v_{m,n+1}, v_{m+1,n+1}$. Because this combination is convex, the interpolation is bounded between $\min v$ and $\max v$ and does not produce overshoot artifacts. Bilinear interpolation reproduces affine functions: if $v_{m,n} = am+bn+c$ then $g(x, y) = ax+by+c$. The triangular function in (A.14) corresponds to a modest low–pass filter in the frequency domain, which results in the attenuation of high-frequency components. Due to its conception, this kernel is both interpolating and reproduces constants, and it requires $N = 2$ supporting points in $1D$ and $N = 4$ in 2D respectively.

Bilinear interpolation is arguably the simplest possible separable method that produces a continuous function. It is extremely efficient and on many platforms available in hardware, making it practical for real-time applications.

A.2.4 Quadratic Interpolation

In the work of Neil Dodgson [44], a quadratic interpolator is introduced claiming that its visual quality is close to that of cubic interpolation, however, it only requires sixty percent of the computation time. Due to this reason, it becomes a serious candidate for performing interpolation in the satellite. Its interpolation kernel is given by

$$u(x) = \begin{cases} -2|x|^2+1, & |x| \leq \frac{1}{2} \\ |x|^2 - \frac{5}{2}|x| + \frac{3}{2}, & \frac{1}{2} < |x| < \frac{3}{2} \\ 0, & elsewhere. \end{cases} \tag{A.15}$$

Note that this kernel requires $N = 3$ supporting points, which sum to $N = 9$ in 2D. It is obtained by forcing linear phase (so it does not introduce phase distortions), C^0-continuity and that it interpolates the data points. As it can be perceived from equation A.15, this kernel is not C^1, however a C^1 version of it is given in the original article of Dogson [44], but by imposing C^1-continuity, it looses the interpolation property and thus becomes an approximator. This kernel also reproduces constants. The interpolating quadratic produces a piecewise reconstruction where each parabolic piece is constrained to pass through a data point and the two adjacent midpoints. Finally, a reconstructor with linear phase obeys the rule that $u(x - x') = u(-(x - x'))$ for some constant x'. That is to say, the function is symmetric around the line $x = x'$. For it to have zero phase, $x' = 0$ and $u(x) = u(-x)$, thus it is obtained by forcing u to be an even function.

A.2.5 Bicubic Interpolation

Bicubic interpolation tries to fit cubic polynomials into the sample data points, in order to approximate the sinc function. Bicubic interpolation can be performed using distinct support sizes: $N = 2, 4, 6$ and 8 support points.

Two-Point Interpolation

By forcing C^0 and C^1 continuity and $u(0) = 1, u(k) = 0$ for $k \neq 0$, for $N = 2$ we obtain the following kernel:

$$u(x) = \begin{cases} 2|x|^3 - 3|x|^2 + 1, & 0 \leq |x| \leq 1 \\ 0, & elsewhere. \end{cases} \tag{A.16}$$

This kernel does also reproduce constants by definition and the resulting curves are similar to those obtained by linear interpolation, but the pieces fit C^1-continuosly in the spatial domain.

Four-Point Interpolation

Again, by forcing C^0 and C^1 continuity and $u(0) = 1, u(k) = 0$ for $k \neq 0$, then seven of the eight coefficients are determined leaving one extra free parameter a:

$$u(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1, & 0 \leq |x| < 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a, & 1 \leq |x| < 2 \\ 0, & elsewhere. \end{cases} \tag{A.17}$$

The values -1, -0.75, and -0.5 have been proposed for a, motivated by various notions of optimality [89]. The choice $a = -0.5$, proposed by Keys [82], is particularly compelling since it is third-order accurate using $a = -0.5$ and only first-order accurate for any other a. Furthermore, $a = -0.5$ is optimal in a sense of matching the sinc kernel and is also optimal in terms of the interpolation error. Keys determined the constant a by forcing the Taylor series expansion of the sampled sinc function to agree in as many terms as possible with the original signal. In fact, when using $a = -0.5$, the first three terms of the Taylor series expansion of the input signal agree with the interpolated function. Thus, cubic interpolation with $a = -0.5$ can reconstruct any second-degree polynomial. This so-called Keys interpolation kernel, also known as the Catmull–Rom cubic is:

$$u(x) = \begin{cases} (3/2)|x|^3 - (5/2)|x|^2 + 1, & 0 \leq |x| < 1 \\ -(1/2)|x|^3 + (5/2)|x|^2 - 4|x| + 2, & 1 \leq |x| < 2 \\ 0, & elsewhere. \end{cases} \tag{A.18}$$

Six-Point and Eight Point Interpolation

Using third degree polynomials but increasing the interpolation kernel support N improves the quality of resampling, although the execution time is dramatically decreased. Cubic interpolation using $N = 6$ support points is given by

$$u(x) = \begin{cases} (6/5)|x|^3 - (11/5)|x|^2 + 1, & 0 \leq |x| < 1 \\ -(3/5)|x|^3 + (16/5)|x|^2 - (27/5)|x| + 14/5, & 1 \leq |x| < 2 \\ (1/5)|x|^3 - (8/5)|x|^2 + (21/5)|x| - 18/5, & 2 \leq |x| < 3 \\ 0, & elsewhere. \end{cases} \tag{A.19}$$

while using $N = 8$ support points, the kernel becomes

$$u(x) = \begin{cases} (67/56)|x|^3 - (123/56)|x|^2 + 1, & 0 \leq |x| < 1 \\ -(33/56)|x|^3 + (177/56)|x|^2 - (75/14)|x| + 39/14, & 1 \leq |x| < 2 \\ (9/56)|x|^3 - (75/56)|x|^2 + (51/14)|x| - 45/14, & 2 \leq |x| < 3 \\ -(3/56)|x|^3 + (33/56)|x|^2 - (17/7)|x| + 18/7, & 3 \leq |x| < 4 \\ 0, & elsewhere. \end{cases} \tag{A.20}$$

A.2.6 Windowed Sinc Approximations

As explained before, the sinc interpolation produces ripple artifacts in aliased images due to the so-called Gibbs phenomenon [77]. A solution to limiting the ripple artifacts of the sinc kernel is to approximate it with a compactly-supported function

$$u(x) = w(x) sinc(x), \tag{A.21}$$

where w is a window function.

Lanczos

A popular choice in image processing is the Lanczos window,

$$w(x) = \begin{cases} sinc(x/n) & \text{if } |x| < n, \\ 0 & elsewhere. \end{cases} \tag{A.22}$$

where n is a positive integer usually set to 2, 3 or 4. The Lanczos kernel does not reproduce constants exactly, but can be normalized to fix this as described in section A.1.3. Each interpolated value is the weighted sum of $2n$ consecutive input samples. Thus, by varying the $2n$ parameter one may trade computation speed for improved frequency response. The parameter also allows one to choose between a smoother interpolation or a preservation of sharp transients in the data. For image processing, the tradeoff is between the reduction of aliasing artifacts and the preservation of sharp edges.

There are many other possibilities for the window, for example Hamming, Kaiser, and Dolph-Chebyshev windows to name a few, each making different tradeoffs in frequency characteristics. The parameter n indeed defines the support of the kernel. When $n = 2$ the kernel has support $N = 4$ in 1D and $N = 16$ for images. $n = 3$ implies a support $N = 6$ in 1D and $N = 36$ in 2D, while $n = 4$ implies a big support $N = 8$ in 1D and $N = 64$ in 2D, usually making the interpolation method time consuming.

A.2.7 B-Splines

Basis splines (B–splines) are on of the most commonly used family of spline functions. They can be derived by several self-convolutions of a so called basis function. In fact, taking β^0 as the indicator function of $\left[-\frac{1}{2}, \frac{1}{2}\right]$ and B_k as the convolution product

$$\beta^k := \beta^{k-1} * \beta^0,\ k = 1, 2, \ldots$$

then B_k are called (centered) cardinal B–splines. It has compact support in $\left[-\frac{k+1}{2}, \frac{k+1}{2}\right]$ and is an even function. There is a whole family of basis functions made of B–splines. Its general formula is given by

$$\beta^k(x) = \sum_{j=0}^{k+1} \frac{(-1)^j (k+1)}{(k+1-j)!\, j!} \left(\frac{n+1}{2} + x - k\right)_+^k \quad \forall x \in \mathbb{R}, \forall k \in \mathbb{N} \tag{A.23}$$

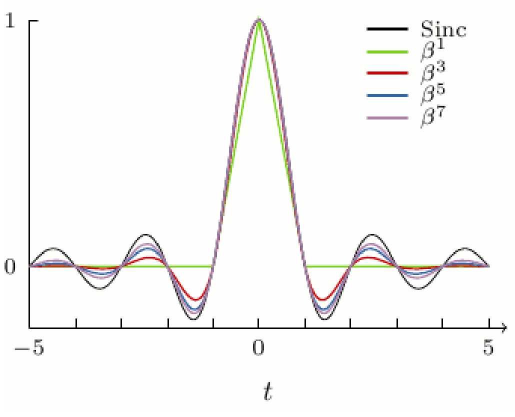
where $(x)_+^k$ is the one-sided power function

$$x_+^n = \begin{cases} 0, & n = 0 \wedge x < 0 \\ \frac{1}{2}, & n = 0 \wedge x = 0 \\ 1, & n = 0 \wedge x > 0 \\ x_+^0 x^n, & n > 0. \end{cases} \tag{A.24}$$

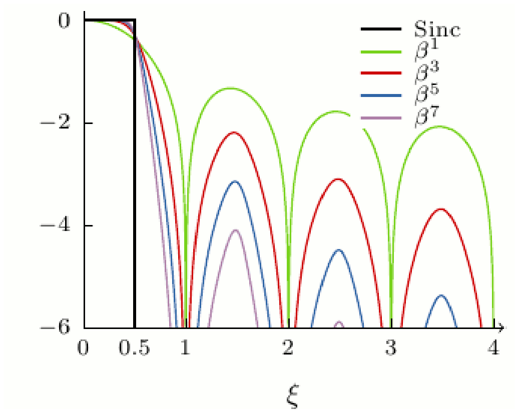
The B–spline functions are optimal in the sense that, among all piecewise polynomials with uniformly spaced knots, the B–splines have the maximal approximation order and are maximally continuous for a given support.

The B–spline β^{j-1} has approximation order j. For $J > 2$, β^{j-1} is not interpolating, so prefiltering must be applied as described in section A.1.3 where β^{j-1} takes the role of f.

As $J \to \infty$, B–spline interpolation converges to Whittaker–Shannon interpolation in a strong sense: the associated interpolation kernel u converges to the sinc both in the spatial and Fourier domains in L^p for any $1 \le p < \infty$. This convergence is illustrated in figure A.4 with $u(x)$ and its Fourier transform for degrees 1, 3, 5, and 7.



(a) $u(t)$



(b) $\log_{10}|\hat{u}(\xi)|$

Figure A.4 – Comparison of the B-spline interpolation kernels and the sinc function.

The most commonly used B–spline is β^3, the cubic B–spline, which has support $N = 4$ in 1D and $N = 16$ in 2D:

$$u(x) = \begin{cases} (2/3) - (1/2)|x|^2(2 - |x|), & 0 \le |x| < 1 \\ (1/6)(2 - |x|)^3, & 1 \le |x| < 2 \\ 0, & elsewhere. \end{cases} \tag{A.25}$$

A.2.8 o-Moms

The family of functions that enjoy Maximal Order and Minimal Support is called Moms (also called splines of minimal support). It can be shown that any of these functions can be expressed as the weighted sum of a B–spline and its derivatives, such that the distribution of the decomposition theorem has a vanishing support.

$$Moms^n(x) = \beta^n(x) + \sum_{m=1}^{n} c_m \frac{d^m}{dx^m}\beta^n(x). \tag{A.26}$$

The B-splines are a special case within the class of Moms. The B-splines are the Moms with maximum regularity, they are $(J-2)$ times continuously differentiable. Within the Moms, the "o-Moms functions" are the Moms defined by minimizing the quantity

$$\frac{1}{J!}\sqrt{\sum_{k\in\mathbb{Z}_*} |\hat{u}^{(J)}(k)|^2}$$

The o-Moms functions have the same approximation order and support as the B-splines. So compared to the B-splines, the advantage of o-Moms is that they have lower asymptotic L^2 interpolation error. On the other hand, o-Moms are less regular than the B-splines.

The first few even-order o-Moms are

$$\begin{array}{cc}
\text{Order J} & f(t) \\
2 & \beta^1(t) \\
4 & (1+\frac{1}{42}\frac{d^2}{dx^2})\beta^3(t) \\
6 & (1+\frac{1}{33}\frac{d^2}{dx^2}+\frac{1}{7920}\frac{d^4}{dx^4})\beta^5(t) \\
8 & (1+\frac{1}{30}\frac{d^2}{dx^2}+\frac{1}{4680}\frac{d^4}{dx^4}+\frac{1}{3603600}\frac{d^6}{dx^6})\beta^7(t) \\
10 & (1+\frac{2}{57}\frac{d^2}{dx^2}+\frac{7}{25840}\frac{d^4}{dx^4}+\frac{1}{1627920}\frac{d^6}{dx^6}+\frac{1}{3047466240}\frac{d^8}{dx^8})\beta^9(t) \\
12 & (1+\frac{5}{138}\frac{d^2}{dx^2}+\frac{1}{3220}\frac{d^4}{dx^4}+\frac{1}{1101240}\frac{d^6}{dx^6}+\frac{1}{1078334208}\frac{d^8}{dx^8}+\frac{1}{4151586700800}\frac{d^{10}}{dx^{10}})\beta^{11}(t)
\end{array} \tag{A.27}$$

A note on numbering: Usually, with a piecewise polynomial method, its approximation order is L but its highest degree is $(L-1)$. We refer to methods by degree, for instance "o-Moms 3" and "β^3" are methods that are locally cubic polynomial and have approximation order 4.

$$o - Moms^3(x) = \beta^n(x) + \frac{1}{42}.\frac{d^2}{dx^2}\beta^3(x) \tag{A.28}$$

$$= \begin{cases} \frac{1}{2}|x|^3 - |x|^2 + \frac{1}{14}|x| + \frac{13}{21}, & 0 \le |x| < 1 \\ -\frac{1}{6}|x|^3 + |x|^2 - \frac{85}{42}|x| + \frac{29}{21}, & 1 \le |x| < 2 \\ 0, & elsewhere. \end{cases} \tag{A.29}$$

A.2.9 Schaum

The splines proposed by Schaum [147] are also within the class of Moms. Schaum splines have the property that they are interpolating, so prefiltering is not needed. Like the o-Moms, the pseudo-Lagrangian basis functions proposed by Schaum can also be represented as a weighted sum of B-splines and of their even-order derivatives. They have same order and same support as B-splines and o-Moms. Their main interest is that they are interpolating. They are discontinuous for even degrees, and are C_0 for odd degrees.

Quadratic Schaum function is given by:

$$sch^2(x) = \begin{cases} 1 - x^2, & |x| < \frac{1}{2} \\ \frac{9}{16}, & |x| = \frac{1}{2} \\ 1 - \frac{3}{2}x + \frac{1}{2}x^2, & \frac{1}{2} < |x| < \frac{3}{2} \\ -\frac{1}{16}, & |x| = \frac{3}{2} \\ 0, & |x| > \frac{3}{2} \end{cases} \tag{A.30}$$

Cubic Schaum function expressed using third degree B-spline function is:

$$sch^3(x) = \beta^3(x) - \frac{1}{6}\frac{d^2}{dx^2}\beta^3(x) \tag{A.31}$$

$$= \begin{cases} \frac{1}{2}|x|^3 - |x|^2 - \frac{1}{2}|x| + 1, & 0 \leq |x| < 1 \\ \frac{1}{6}(2 - |x|)^3 + \frac{1}{6}|x| - \frac{1}{3}, & 1 \leq |x| < 2 \\ 0, & elsewhere. \end{cases} \tag{A.32}$$

A.2.10 Summary

Table A.1 summarizes the above described interpolation methods that we shall evaluate.

Name	Support radius	Req. Norm. (Sec. A.1.3)	# pairs	Scale factor
nearest	0.5	No	0	1
bilinear	1	No	0	1
quadInterp	1.5	No	0	1
bicubic	2	No	0	1
lanczos2	2	Yes	0	1
lanczos3	3	Yes	0	1
lanczos4	4	Yes	0	1
schaum2	1.5	No	0	1
schaum3	2	No	0	1
bspline2	1.5	No	1	8
bspline3	2	No	1	6
bspline5	3	No	2	120
bspline7	4	No	3	5040
bspline9	5	No	4	362880
bspline11	6	No	5	39916800
omoms3	2	No	1	21/4
omoms5	3	No	2	7920/107
omoms7	4	No	3	675675/346

Table A.1 – Summary of every considered interpolation method. The first column indicates the method name, the second the kernel's support radius, the third column specifies if the kernel requires normalization, the fourth column indicates the number of filter pairs used for prefilting, while the last column indicates the constant scale factor to use with prefiltering. A further explanation about this last two values and the prefiltering process is given in section A.3.3.

A.3 Implementation

In this section, the main algorithm will be explained and further optimized. Two optimizations will be given, in order to improve interpolation speed. The first one relies on the fact that separable kernels are being used, which allows to perform less computations. The second one is based on the fact that we are translating a whole image, thus implying that every pixel will be shifted using the same δ_x, δ_y, allowing us to precompute the kernel values for both x and y positions.

Algorithm 13 describes how to translate an image performing interpolation using a non-separable kernel. It can be seen how the kernel is evaluated S^2 times where S is the support of the kernel. The first proposed optimization is given by using separable kernels.

A.3.1 Separable kernel

Digital image interpolation occurs in all digital photos at some stage. It happens any time you resize or remap your image from one pixel grid to another. The interpolation results can vary significantly depending on the interpolation algorithm. It is only an approximation, therefore an image will always lose some quality each time interpolation is performed (with the exception of Fourier invertible interpolation or oversampling where the initial image can be recovered by the inverse process). The general formula for 2-D

Algorithm 13 2D non-separable interpolation

```
Input: $Kernel$: 2D non-separable interpolation kernel, $KernelWidth$: Kernel support,
$\delta_x, \delta_y$: translation parameters, $Src$: source image, $SrcHeight, SrcWidth$: source image size,
Output: $Dest$: resulting interpolated image

$NumSamples \leftarrow SrcWidth * SrcHeight$ // Amount of samples to perform interpolation
$KernelRadius \leftarrow \frac{KernelWidth}{2}$ // Radius of the kernel
for $k = 0$ to $NumSamples - 1$ do
    // $X_I, Y_I$ is the position on the image that we are interpolating for sample $k$
    $X_I \leftarrow mod(k, SrcWidth) + \delta_x$
    $Y_I \leftarrow \lfloor k/SrcWidth \rfloor + \delta_y$
    // $IndexX0, IndexY0$ is the top left position of where the kernel should be placed on the image
    $IndexX0 \leftarrow \lceil X_I - KernelRadius \rceil;$
    $IndexY0 \leftarrow \lceil Y_I - KernelRadius \rceil;$
    $Sum \leftarrow 0$ // Initialize the sum
    // Compute the interpolated value at (X[k], Y[k]) iterating over the whole kernel
    // To deal with boundaries, Extension method corrects the given position if is outside the limits based on the assumed boundary handling
    for $n = 0$ to $KernelWidth - 1$ do
        $IndexY \leftarrow Extension(SrcHeight, IndexY0 + n)$ // Vertical position of the image
        $SrcRowOffset \leftarrow SrcWidth * IndexY$ // Vector index of the current row
        for $m = 0$ to $KernelWidth - 1$ do
            $IndexX \leftarrow Extension(SrcWidth, IndexX0 + m)$ // Horizontal position of the image
            // Multiply the value of the image by the value of the kernel and accumulate the result
            $Sum \leftarrow Sum + Src[IndexX + SrcRowOffset] * Kernel(X_I - (IndexX0 + m), Y_I - (IndexY0 + n))$
        end for
    end for
    $Dest[k] \leftarrow Sum$ // Finally, assign the accumulated value and repeat for next sample
end for
```

image interpolation is

$$g(x, y) = \sum_{m,n \in \mathbb{Z}} f_{m,n} K(x - m, y - n) \tag{A.33}$$

where $f_{m,n}$ is the image to be interpolated, and $K(x, y)$ is the 2-D interpolating kernel. If the kernel K is separable, it implies that $K(x, y) = K(x)K(y)$, permitting the formula above to be calculated as

$$g(x, y) = \sum_n \sum_m f_{m,n} K(x - m)K(y - n) \tag{A.34}$$

$$= \sum_n K(y - n) \sum_m f_{m,n} K(x - m) \tag{A.35}$$

$$= \sum_n K(y - n)u_n(x) \tag{A.36}$$

where

$$u_n(x) = \sum_m f_{m,n} K(x - m) \tag{A.37}$$

In figure A.1, a kernel of support 4×4 is used to perform interpolation. Actually, for that particular example, as the kernel has support 4, then instead of evaluating it 16 times, we first need to compute $u_n(x)$, that is, to evaluate the kernel horizontally 4 times followed by interpolating the results again, adding another 4 kernel evaluations. Then the amount of kernel evaluations is halved for this case. Actually, if S is the support size, the difference between the required amount of evaluations for a non-separable kernel and a separable one is $S^2 - 2 * S$. Also, by using separable kernels, the performance is increased even further since evaluating a bi-dimensional kernel usually involves more operations than evaluating its one-dimensional version. Algorithm 14 describes this optimization by precomputing in arrays the kernel values used to perform interpolation.

A.3.2 Image translation algorithm

In our particular case, we need to shift a whole image using the global translation parameters δ_x, δ_y, thus translating every pixel always using the same distance. Further advantage can be taken by precomputing the kernel not in a per pixel basis, by globally, reducing dramatically the amount of calculations. Algorithm 15 describes this optimization. The kernel is evaluated $2 * S$ times for the whole image, compared to $2 * S * NumSamples$ as in the separable case or $S^2 * NumSamples$ as in the original algorithm, where S accounts for the kernel's support.

Finally, as described in section A.1.3, some kernels require normalization in order to reproduce constants. Algorithm 16 finally depicts this straightforward procedure, and should be used only with kernels requiring normalization.

A.3.3 Image Prefiltering

In the general interpolation scheme, when the interpolation kernel is not an interpolating function, as described in section A.1.3, the coefficients c_k have to be computed in order for the method to interpolate the samples. Both splines and o-Moms are not interpolating kernels, and thus they require, as explained in section A.1.3, to obtain the c_k coefficients by performing prefiltering in the following way:

$$c = (p)^{-1} * f. \tag{A.38}$$

Algorithm 14 2D separable interpolation

```
Input: $Kernel$: 1D separable interpolation kernel, $KernelWidth$: Kernel support,

$\delta_x, \delta_y$: translation parameters, $Src$: source image, $SrcHeight, SrcWidth$: source image size,
Output: $Dest$: resulting interpolated image


$NumSamples \leftarrow SrcWidth * SrcHeight$ // Amount of samples to perform interpolation
$KernelRadius \leftarrow \frac{KernelWidth}{2}$ // Radius of the kernel
for $k = 0$ to $NumSamples - 1$ do
    // $X_I, Y_I$ is the position on the image that we are interpolating for sample $k$
    $X_I \leftarrow mod(k, SrcWidth) + \delta_x$
    $Y_I \leftarrow \lfloor k/SrcWidth \rfloor + \delta_y$
    // $IndexX0, IndexY0$ is the top left position of where the kernel should be placed on the image
    $IndexX0 \leftarrow \lceil X_I - KernelRadius \rceil$;
    $IndexY0 \leftarrow \lceil Y_I - KernelRadius \rceil$;
    $Sum \leftarrow 0$ // Initialize the sum
    // Precompute 1D kernel results by evaluating it for sample $k$
    for $m = 0$ to $KernelWidth - 1$ do
        $KernelXBuf[m] \leftarrow Kernel(X_I - (IndexX0 + m))$
        $KernelYBuf[m] \leftarrow Kernel(Y_I - (IndexY0 + m))$
    end for
    // Compute the interpolated value at (X[k], Y[k])
    for $n = 0$ to $KernelWidth - 1$ do
        $IndexY \leftarrow Extension(SrcHeight, IndexY0 + n)$ // Vertical position of the image
        $SrcRowOffset \leftarrow SrcWidth * IndexY$ // Vector index of the current row
        for $m = 0$ to $KernelWidth - 1$ do
            $IndexX \leftarrow Extension(SrcWidth, IndexX0 + m)$ // Horizontal position of the image
            // Multiply the value of the image by the value of the 1D horizontal kernel and by the value of the 1D vertical kernel and accumulate the result
            $Sum \leftarrow Sum + Src[IndexX + SrcRowOffset] * KernelXBuf[m] * KernelYBuf[n]$
        end for
    end for
    $Dest[k] \leftarrow Sum$ // Finally, assign the accumulated value and repeat for next sample
end for
```

Algorithm 15 Optimized Image Translation

```
**Input**: $Kernel$: 1D separable interpolation kernel, $KernelWidth$: Kernel support,

$\delta_x, \delta_y$: translation parameters, $Src$: source image, $SrcHeight, SrcWidth$: source image size,
**Output**: $Dest$: resulting interpolated image


$NumSamples \leftarrow SrcWidth * SrcHeight$ // Amount of samples to perform interpolation
$KernelRadius \leftarrow \frac{KernelWidth}{2}$
// Precompute the kernel values by evaluating it (for the fixed translation)
for $m = 0$ to $KernelWidth - 1$ do
    $KernelXBuf[m] \leftarrow Kernel(\delta_x - m - \lceil \delta_x - KernelRadius \rceil)$
    $KernelYBuf[m] \leftarrow Kernel(\delta_y - m - \lceil \delta_y - KernelRadius \rceil)$
end for
// Iterate all the image samples and perform the resampling
for $k = 0$ to $NumSamples - 1$ do
    // $X_I, Y_I$ is the position on the image that we are interpolating for sample $k$
    $X_I \leftarrow mod(k, SrcWidth) + \delta_x$
    $Y_I \leftarrow \lfloor k/SrcWidth \rfloor + \delta_y$
    // $IndexX0, IndexY0$ is the top left position of where the kernel should be placed on the image
    $IndexX0 \leftarrow \lceil X_I - KernelRadius \rceil$;
    $IndexY0 \leftarrow \lceil Y_I - KernelRadius \rceil$;
    $Sum \leftarrow 0$ // Initialize the interpolation sum
    // Compute the interpolated value at (X[k], Y[k])
    for $n = 0$ to $KernelWidth - 1$ do
        $IndexY \leftarrow Extension(SrcHeight, IndexY0 + n)$ // Vertical position of the image
        $SrcRowOffset \leftarrow SrcWidth * IndexY$ // Vertical index of the current row
        for $m = 0$ to $KernelWidth - 1$ do
            $IndexX \leftarrow Extension(SrcWidth, IndexX0 + m)$ // Horizontal position of the image
            // Multiply the value of the image by the value of the 1D horizontal kernel and by the value of the 1D vertical kernel and accumulate the result
            $Sum \leftarrow Sum + Src[IndexX + SrcRowOffset] * KernelXBuf[m] * KernelYBuf[n]$
        end for
    end for
    $Dest[k] \leftarrow Sum$ // Finally, assign the accumulated value and repeat for next sample
end for
```

Algorithm 16 Optimized image translation normalizing kernel

```
Input: $Kernel$: 1D separable interpolation normalization requiring kernel ,
$KernelWidth$: kernel support, $\delta_x, \delta_y$: translation parameters, $Src$: source image, $SrcHeight, SrcWidth$: source image size,
Output: $Dest$: resulting interpolated image

$NumSamples \leftarrow SrcWidth * SrcHeight$ // Amount of samples to perform interpolation
$KernelRadius \leftarrow \frac{KernelWidth}{2}$
// Precompute the kernel values by evaluating it (for the fixed translation)
for $m = 0$ to $KernelWidth - 1$ do
    $KernelXBuf[m] \leftarrow Kernel(\delta_x - m - \lceil \delta_x - KernelRadius \rceil)$
    $KernelYBuf[m] \leftarrow Kernel(\delta_y - m - \lceil \delta_y - KernelRadius \rceil)$
end for
// Iterate all the image samples and perform the resampling
for $k = 0$ to $NumSamples - 1$ do
    // $X_I, Y_I$ is the position on the image that we are interpolating for sample $k$
    $X_I \leftarrow mod(k, SrcWidth) + \delta_x$
    $Y_I \leftarrow \lfloor k/SrcWidth \rfloor + \delta_y$
    // $IndexX0, IndexY0$ is the top left position of where the kernel should be placed on the image
    $IndexX0 \leftarrow \lceil X_I - KernelRadius \rceil$;
    $IndexY0 \leftarrow \lceil Y_I - KernelRadius \rceil$;
    $Sum \leftarrow 0$ // Initialize the interpolation result value
    $Acum \leftarrow 0$ // Initialize the weights accumulator used for normalization
    // Compute the interpolated value at (X[k], Y[k])
    for $n = 0$ to $KernelWidth - 1$ do
        $IndexY \leftarrow Extension(SrcHeight, IndexY0 + n)$ // Vertical position of the image
        $SrcRowOffset \leftarrow SrcWidth * IndexY$ // Vertical index of the current row
        for $m = 0$ to $KernelWidth - 1$ do
            $IndexX \leftarrow Extension(SrcWidth, IndexX0 + m)$ // Horizontal position of the image
            $Weight \leftarrow KernelXBuf[m] * KernelYBuf[n]$ // Calculate the kernel weight
            // Multiply the value of the image by the value of the 1D horizontal kernel and by the value of the 1D vertical kernel and accumulate the result
            $Sum \leftarrow Sum + Src[IndexX + SrcRowOffset] * Weight$
            $Acum \leftarrow Acum + Weight$ // Acumulate the weights for normalization
        end for
    end for
    $Dest[k] \leftarrow \frac{Sum}{Acum}$ // Finally, assign the normalized accumulated value and repeat for next sample
end for
```

where $p_k = \varphi(k)$ depends on the interpolating kernel φ, that may be a B-spline or an o-Moms function. Due to the fact that $\varphi(k)$ is symmetric, this makes the filter symmetric. Also, this filtering is recursive, that means that to calculate the output for the current pixel, the results of previous pixels is used. This type of filters usually receive the name of infinite impulse response (IIR) filters. Since any symmetrical recursive filter may be decomposed into a cascade of elementary symmetrical exponential filters, which themselves can be separated into two complementary causal and anticausal components, the implementation of this filter can be fast and simple. This decomposition is explained in the following way. Let us consider the z transform of a generic symmetrical stable recursive filter of order $2N$:

$$H_{2N}(z) = \frac{c_0}{[z^N + z^{-N}] + \left(\sum_{k=1}^{N-1} a_k[z^k + z^{-k}]\right) + a_0} \tag{A.39}$$

where c_0 and $a_k, 0 \leq k \leq N-1$ are constant coefficients related to the choice of interpolating kernel. A polynomial $P_{2N}(z)$ can be defined by multiplying the denominator of (A.39) by z^N. Clearly, since $H_{2N}(z) = H_{2N}(z^{-1})$, this polynomial satisfies the equation: $z^{-N}P_{2N}(z) = z^N P_{2N}(z^{-1})$. It follows that the zeros of $P_{2N}(z)$ occur in reciprocal pairs. These roots, which are assumed to lie outside the unit circle, are denoted by (r_i, r_i^{-1}) with $|r_i| < 1, 1 \leq i \leq N$. Consequently, $H_{2N}(z)$ can be factored as

$$H_{2N}(z) = c_0 \prod_{i=1}^{N} H(z; r_i) \tag{A.40}$$

where

$$H(z; r_i) = \underbrace{\left(\frac{1}{1 - r_i z^{-1}}\right)}_{Causal} \underbrace{\left(\frac{-r_i}{1 - r_i z}\right)}_{Anti-causal} \tag{A.41}$$

allowing the prefiltering to be performed as a cascade of first-order recursive filters. This is done using first a causal filter, and performing an anti-causal filter on the results, as shown in equation (A.42).

$$\begin{cases} y^+(k) = x(k) + r_i y^+(k-1) & k = 1, \ldots, K-1 \\ y(k) = r_i(y(k+1) - y^+(k)) & k = K-2, \ldots, 0 \end{cases} \tag{A.42}$$

where $x(k)$ is the input signal, $y(k)$ is the output and r_i are the roots of the kernel basis. For the left endpoint, it can be computed depending on the boundary extension:

$$\begin{array}{cc} \text{Half-sample symmetric} & y^+(0) = x(0) + r_i \sum_{n=0}^{\infty} r_i^n x(n) \\ \text{Whole-sample symmetric} & y^+(0) = \sum_{n=0}^{\infty} r_i^n x(n) \end{array} \tag{A.43}$$

Since $|r_i| < 1$, the terms of the sum are decaying, so in practice we only evaluate as many terms as are needed for the desired accuracy. For the right endpoint, we have:

$$\begin{array}{cc} \text{Half-sample symmetric} & y(K-1) = \frac{r_i}{r_i - 1} y^+(K-1) \\ \text{Whole-sample symmetric} & y(K-1) = \frac{r_i}{r_i^2 - 1}\{y^+(K-1) + r_i y^+(K-2)\} \end{array} \tag{A.44}$$

Finally, the constant scale factor is applied $y(k) = c_0 y(k)$. The cost of prefiltering is linear in the number of pixels. Its computational load, for example, for the popular cubic B-spline is two additions and three multiplications per produced coefficient. Prefiltering can be computed in-place so that the prefiltered values overwrite the memory used by the input. For prefiltering in two-dimensions, the image is first prefiltered along each

n	B-Spline transfer function	c_0	Poles (r_i)
2	$\frac{8}{z+6+z^{-1}}$	8	$r_1 = -3 + 2\sqrt{2}$
3	$\frac{6}{z+4+z^{-1}}$	6	$r_1 = -2 + \sqrt{3}$
5	$\frac{120}{z^2+26z+66+26z^{-1}+z^{-2}}$	120	$r_1 = -0.4300575$, $r_2 = -0.0430963$
7	$\frac{5040}{z^3+120z^2+1191z^1+2416+1191z^{-1}+120z^{-2}+z^{-3}}$	5040	$r_1 = -0.53528$, $r_2 = -0.122555$ $r_3 = -0.00914869$

Table A.2 – Transfer functions, poles and characteristic values for several B-splines

column, then the column-prefiltered image is prefiltered along each row. It is worth mentioning that input images with size shorter than the support of the interpolation kernel are not considered in this report since the type of images obtained by the satellite are large enough. However, if this would not be the case, then another boundary extension method should be applied, like assuming periodicity on the input images.

So finally, the only thing missing is how to calculate the roots and the value c_0, usually referred as constant scale factor or characteristic value, for each of the B-splines and o-Moms. In order to do so, an alternative formulation for the B-splines based on iterative equations, as given in equation (A.45), is used to determine the transfer functions in the Z-domain. Then, the roots of these polynomials are searched so that these functions are properly factorized obtaining both the c_0 and all the r_i values.

$$b^n(k) = \frac{1}{n}\left[\left(k + \frac{n+1}{2}\right)c^{n-1}(k) + \left(\frac{n+1}{2} - k\right)c^{n-1}(k-1)\right] \tag{A.45}$$

$$c^n(k) = \frac{1}{n}\left[\left(k + \frac{n+2}{2}\right)b^{n-1}(k+1) + \left(\frac{n}{2} - k\right)b^{n-1}(k)\right] \tag{A.46}$$

(A.47)

In table A.2, the transfer functions, poles and characteristic values are given for several B-splines.

Once the image has been prefiltered $c = (p)^{-1} * f$, interpolation is computed like any other interpolation kernel.

This prefiltering is developed in detail in algorithm 17. For the $PrefilterScan$ method of algorithm 18, it takes $4(N+1)$ additions and $2(N+1)$ multiplications where N is the 1D sample size. Then, to prefilter the whole image, it is required $8 \times NumFilterPairs \times (Width-1) \times (Height-1)$ additions and $4 \times NumFilterPairs \times (Width-1) \times (Height-1) + Width \times Height$ multiplications. For example, for Cubic B-Spline interpolation, it is required $8 \times (Width-1) \times (Height-1)$ additions and $4 \times (Width-1) \times (Height-1) + Width \times Height \approx 5 \times Width \times Height$ multiplications, which is linear on the size on the image.

A.3.4 Fourier Interpolation

To perform non-integer interpolation without loosing information, one has just to modify the phase of the signal in the frequency domain. This procedure is described in detail in algorithm 19. This procedure costs $\mathcal{O}(n)$ where n is the size of the image, however both the FFT and its inverse has to be computed for the input image, an operation that takes $\mathcal{O}(n \log n)$.

Algorithm 17 In-place prefiltering

```
Input: $r_i$: roots (poles) of the kernel ($0 \leq i \leq NumFilterPairs - 1$), $NumFilterPairs$: amount of roots, $c_0$ characteristic value of the kernel, $Data$: source image, $Height, Width$: source image size, $boundaryHandling$: kind of boundary handling to use.
Output: $Data$: prefiltered image

$c_0 \leftarrow (c_0)^2$ // Square the ConstantFactor for two spatial dimensions
// Apply the filter for each column first
for $x = 0$ to $Width - 1$ do
    for $k = 1$ to $NumFilterPairs$ do
        $PrefilterScan(Data, x, Width, Height, r_k, boundaryHandling)$
    end for
end for
// Now filter the rows
for $y = 0$ to $Height - 1$ do
    for $k = 1$ to $NumFilterPairs$ do
        $PrefilterScan(Data, Width * y, 1, Width, r_k, boundaryHandling)$
    end for
end for
// Apply constant scale factor
for $k = 0$ to $Width * Height - 1$ do
    $Data[k] \leftarrow Data[k] * c_0$
end for
```

A.4 Results

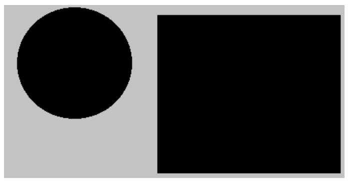
Every interpolation algorithm was evaluated both quantitatively and qualitatively. To do this, we evaluated each method using three different images, shown in figure A.5. To complete this report, the execution time of each algorithm is presented.

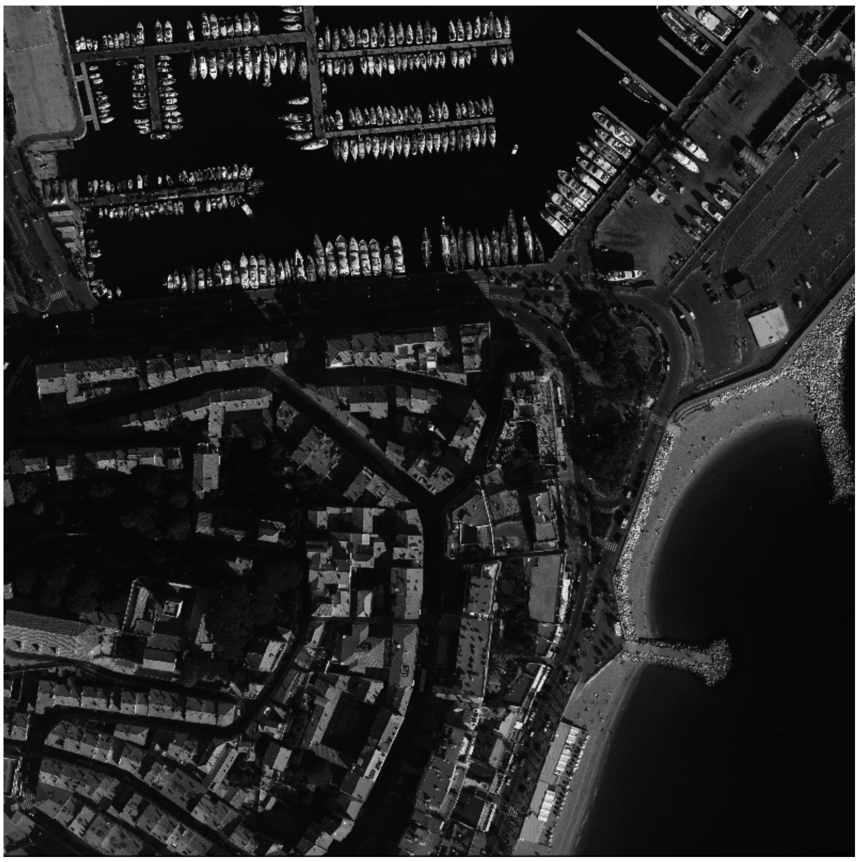
Algorithm 18 PrefilterScan (1D in-place filtering with a first-order recursive filter pair)

```
**Input**: $Data$: source image, $pos$ position to start filtering, $stride$: stride (distance) between successive elements, $N$: number of samples, $r$: root (pole) of the kernel, $boundaryHandling$: kind of boundary handling to use.
**Output**: $Data$: prefiltered image

$Eps \leftarrow e^{-4}$ // Desired precision
$n0 \leftarrow \left\lceil \frac{\log(Eps)}{\log(|r|)} \right\rceil$ // Amount of elements to process to achieve desired precision
**if** $n0 > N$ **then**
    $n0 \leftarrow N$
**end if**
// First apply the causal filter
// To do so, we need to compute the first value using the desired boundaryHandling
**if** $boundaryHandling = WHOLE - SYMMETRIC$ **then**
    $Sum \leftarrow \sum_{i=0}^{n0} Data[pos + i * stride] * r^i$
**else if** $boundaryHandling = HALF - SYMMETRIC$ **then**
    $Sum \leftarrow Data[pos] + r \sum_{i=0}^{n0} Data[pos + i * stride] * r^i$
**end if**
$Data[pos] \leftarrow Sum$
$Last \leftarrow Data[pos]$
$i \leftarrow pos + stride$
$iEnd \leftarrow pos + (N - 1) * stride$
**while** $i \leq iEnd$ **do**
    $Data[i] \leftarrow Data[i] + r * Last$
    $Last \leftarrow Data[i]$
    $i \leftarrow i + stride$
**end while**
// Then apply the anti-causal filter
// Again, we need to compute the last value using the desired boundaryHandling
**if** $boundaryHandling = WHOLE - SYMMETRIC$ **then**
    $Data[End] \leftarrow \frac{r}{r^2-1} * (Data[iEnd] + r * Data[iEnd - stride])$
**else if** $boundaryHandling = HALF - SYMMETRIC$ **then**
    $Data[iEnd] \leftarrow Data[iEnd] * \frac{r}{r-1}$
**end if**
$Last \leftarrow Data[iEnd]$
$i \leftarrow iEnd - stride$
**while** $i \geq 0$ **do**
    $Data[i] \leftarrow r * (Last - Data[i])$
    $Last \leftarrow Data[i]$
    $i \leftarrow i - stride$
**end while**
```

Algorithm 19 FourierShift

```
Input: $\delta_x, \delta_y$: translation parameters, $img$: input image, $w, h$: width and height of input image.
Output: $res$: shifted image

$res \leftarrow 2D - DFT(img)$ // Calculate 2D Discrete Fourier Transform of the input image
if $w \mod 2 = 0$ then
    $limHoriz \leftarrow \frac{w}{2}$ // If the width is even, we iterate horizontally from $[-\frac{w}{2}, \frac{w}{2}]$
else
    $limHoriz \leftarrow \frac{w-1}{2}$ // otherwise the range is $[-\frac{w-1}{2}, \frac{w-1}{2}]$.
end if
if $h \mod 2 = 0$ then
    $limVert \leftarrow \frac{h}{2}$ // If the height is even, we iterate vertically from $[-\frac{h}{2}, \frac{h}{2}]$
else
    $limVert \leftarrow \frac{h-1}{2}$ // otherwise the range is $[-\frac{h-1}{2}, \frac{h-1}{2}]$.
end if
for $x = -limHoriz$ to $limHoriz$ do
    for $y = -limVert$ to $limVert$ do
        if $x < 1$ then
            $posX \leftarrow w + x$ // Horizontal periodicity of the image is assumed
        else
            $posX \leftarrow x$
        end if
        if $y < 1$ then
            $posY \leftarrow h + y$ // Vertical periodicity of the image is assumed
        else
            $posY \leftarrow y$
        end if
        $res(posY, posX) \leftarrow res(posY, posX)e^{-2\pi i(\frac{x\delta_x}{w} + \frac{y\delta_y}{h})}$ // Change the phase
    end for
end for
$res \leftarrow 2D - IDFT(img)$ // Transform the resulting image back to the spatial domain
```


(a) Image 1


(b) Image 2

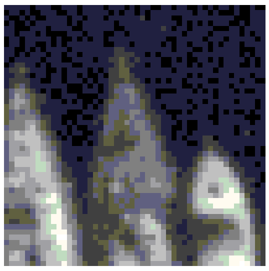

(c) Image 3

Figure A.5 – Images used to measure interpolation performance.

A.4.1 Quantitative results

In this section, we evaluate the performance of each interpolation algorithm based on measuring the difference they produce when resampling. Several metrics were used to analyze the performance of each method, which will be introduced in section A.4.1. This will be followed by the results of every interpolation algorithm for each of the three images shown in figure A.5.

Performance metrics

To evaluate the performance of each method, some well-known metrics were employed. Given the original color $m \times n$ image I defined in the RGB color space and the resulting twice-interpolated image $\hat{I}$, the Mean Square Error (MSE) is defined by

$$MSE = \frac{1}{3}\frac{1}{m\,n} \sum_{c=\{R,G,B\}} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I_c(i,j) - \hat{I}_c(i,j)]^2 \tag{A.48}$$

The MSE has the same units of measurement as the square of the quantity being estimated. In an analogy to standard deviation, taking the square root of MSE yields the root-mean-square error (RMSE), which has the same units as the quantity being estimated.

The peak signal-to-noise ratio, usually referred to as PSNR, is given by

$$PSNR = 10 \cdot \log_{10}\left(\frac{255^2}{MSE}\right) \tag{A.49}$$

$$= 20 \cdot \log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \tag{A.50}$$

$$= 20 \cdot \log_{10}(255) - 10 \cdot \log_{10}(MSE) \tag{A.51}$$

and is an engineering term used for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. In our case, the noise is introduced by the interpolation algorithms and possibly, some minor numerical issues. The PSNR is usually expressed in terms of the logarithmic decibel scale.

Another interesting measure is the maximum absolute difference between the pixels in the original image and the resulting pixels in the interpolated image, defined by

$$MaxAbsDiff = \max_{x,y,c} |I_c(x,y) - \hat{I}_c(x,y)| \tag{A.52}$$

Finally, the SSIM metric considers image degradation as perceived change in structural information. Structural information is the idea that the pixels have strong interdependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene, and are completely ignored by the MSE or PSNR metrics. The SSIM metric is calculated on various windows of an image. The measure between two windows $\mathbf{x}$ and $\mathbf{y}$ of common size $N \times N$ is:

$$\text{SSIM}(\mathbf{x},\mathbf{y}) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{A.53}$$

with μ_x, σ_x^2 the average and the variance of $\mathbf{x}$, μ_y, σ_y^2 the average and the variance of $\mathbf{y}$, σ_{xy} the covariance of $\mathbf{x}$ and $\mathbf{y}$, $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ two variables to stabilize the division with weak denominator, L the dynamic range of the pixel-values (in our case $L = 1$ since floating point images are in the $[0, 1]$ range, $k_1 = 0.01$ and $k_2 = 0.03$ the constant default values suggested in the original article [184]). In this report, a rounded 11-tap Gaussian window is used with $\sigma = 1.5$.

Then to compare both images I^1 and I^2, in our case the final SSIM is calculated by averaging the SSIM of all possible Gaussian windows located in the same position on both images on each RGB channel, as it is shown in equation A.54

$$SSIM(I^1, I^2) := \frac{\sum_{x=1}^{w-11}\sum_{y=1}^{h-11}\sum_{c=1}^{3} SSIM(I^1_{x,y,c}, I^2_{x,y,c})}{w * h * 3} \tag{A.54}$$

where w and h stands for the width and the height of both images and $I^1_{x,y,c}$ is a circular 11-tap Gaussian window of the first image, on color channel c, with top-left corner (x, y) and $I^2_{x,y,c}$ is the same window but on the second image.

Results shifting half a pixel

To evaluate the presented interpolation methods, a simple test was performed. An input image was shifted $v_x = 0.5, v_y = 0.5$ pixels and then shifted back the same amount, thus allowing to compare the result with the original image. Half a pixel of shift is chosen because the contribution of each neighbor for each interpolated pixel is the same, and their distance is the maximum possible, which apparently makes this the most 'troublesome' case. In tables A.3, A.4 and A.5 all interpolation algorithms of section A.2 are evaluated using the metrics mentioned above.

For the three evaluated images, Fourier interpolation achieves no approximation error, as expected, since it is the only one among all methods that is invertible. Results for the first image on table A.3, where steep edges are present, the SSIM index, that focuses its measurement on visual quality rather than approximation error (is intended to measure structure similarity), b-spline2 outperforms obtains the best results. This is explained by the resulting ripple artifacts found after interpolation by high order methods such as bsplines, or oMoms. For this particular case, best results are obtained using bspline2, schaum2, bicubic and lanczos2 methods.

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
fourier	0	∞	1	0	0
nearest	195	21.1146	0.9245	503.0599	22.4290
bilinear	85	29.6779	0.9824	70.0313	8.3685
quadInterp	85	29.6779	0.9824	70.0313	8.3685
bicubic	64	32.2473	0.9895	38.7574	6.2255
lanczos2	64	32.2473	0.9895	38.7574	6.2255
lanczos3	44	35.1170	0.9878	20.0161	4.4739
lanczos4	37	36.1669	0.9833	15.7177	3.9646
schaum2	64	32.2473	0.9895	38.7574	6.2255
schaum3	64	32.2473	0.9895	38.7574	6.2255
bspline2	53	33.5434	**0.9899**	28.7569	5.3625
bspline3	47	34.7281	0.9884	21.8911	4.6788
bspline5	34	36.8641	0.9809	13.3865	3.6588
bspline7	26	38.3039	0.9750	9.6092	3.0999
bspline9	21	39.1890	0.9685	7.8375	2.7996
bspline11	**18**	**40.0858**	0.9657	**6.3753**	**2.5249**
omoms3	40	35.9879	0.9854	16.3792	4.0471
omoms5	31	37.4795	0.9786	11.6181	3.4085
omoms7	24	38.7618	0.9727	8.6477	2.9407

Table A.3 – Results for the first image

For the map image, image 2 in our test set, results in table A.4 show that again Fourier interpolation has no error. Besides this, high order oMoms and bSplines, as better approximators of the Shannon-Whitaker kernel, also manage to get good results, however their computational time, as it will be seen later, is prohibitive. As for faster methods, both oMoms3 and Lanczos3 obtain the best results without a heavy computational cost.

Finally, for the third image, whose results can be appreciated in table A.5, again Fourier interpolation achieves zero error. The third input image is badly sampled, noisy and aliased, which does not favor any interpolation method. In fact, because of its noisy conditions, every other interpolation method presents low PSNR and SSIM scores than for the other two test images. Again as with the previous image, higher order interpolation methods usually get better results, and among the faster methods, third order B-Splines seems to achieve better results.

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
fourier	0	∞	1	0	0
nearest	240	18.7423	0.5993	868.6580	29.4730
bilinear	160	25.4543	0.8673	185.2054	13.6090
quadInterp	160	25.4543	0.8673	185.2054	13.6090
bicubic	141	27.4889	0.9217	115.9279	10.7670
lanczos2	141	27.4889	0.9217	115.9278	10.7670
lanczos3	110	29.7068	0.9539	69.5657	8.3406
lanczos4	103	30.8751	0.9628	53.1580	7.2910
schaum2	141	27.4889	0.9217	115.9279	10.7670
schaum3	141	27.4889	0.9217	115.9279	10.7670
bspline2	125	28.6285	0.9405	89.1732	9.4432
bspline3	110	29.5757	0.9521	71.6993	8.4675
bspline5	104	31.4393	0.9673	46.6820	6.8324
bspline7	101	32.6689	0.9732	35.1718	5.9306
bspline9	99	33.6026	0.9763	28.3677	5.3261
bspline11	**97**	**34.3562**	**0.9782**	**23.8486**	**4.8835**
omoms3	106	30.5785	0.9613	56.9155	7.5442
omoms5	102	31.9686	0.9701	41.3254	6.4285
omoms7	100	33.0891	0.9748	31.9282	5.6505

Table A.4 – Results for the second image

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
fourier	0	∞	1	0	0
nearest	100	18.7247	0.4300	872.1973	29.5330
bilinear	48	25.1255	0.7513	199.7704	14.1340
quadInterp	48	25.1255	0.7513	199.7704	14.1340
bicubic	51	26.8631	0.8352	133.8970	11.5714
lanczos2	51	26.8630	0.8352	133.9008	11.5716
lanczos3	51	28.7841	0.9014	86.0332	9.2754
lanczos4	42	29.8844	0.9249	66.7784	8.1718
schaum2	51	26.8631	0.8352	133.8970	11.5714
schaum3	51	26.8631	0.8352	133.8970	11.5714
bspline2	50	27.8419	0.8717	106.8795	10.3383
bspline3	46	28.6835	0.8967	88.0501	9.3835
bspline5	39	30.4155	0.9340	59.0916	7.6871
bspline7	36	31.6181	0.9517	44.7995	6.6932
bspline9	34	32.5335	0.9620	36.2856	6.0238
bspline11	**32**	**33.2479**	**0.9686**	**30.7815**	**5.5481**
omoms3	41	29.5965	0.9186	71.3565	8.4473
omoms5	38	30.9313	0.9422	52.4751	7.2440
omoms7	35	32.0323	0.9565	40.7235	6.3815

Table A.5 – Results for the third image

Results shifting 0.7 pixel

In the previous section, some methods present the same results. This is due to the fact that interpolation was performed using half a pixel in both directions, and both interpolation kernels coincide on those values. To further evaluate the methods, the images were shifted $v_x = 0.7, v_y = 0.7$ pixels and then again, shifted back the same amount. Results

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
Fourier	0	∞	1	0	0
bilinear	73	31.0627	0.9871	50.9110	7.1352
quadInterp	48	34.7631	0.9945	21.7156	4.6600
bicubic	45	35.3128	0.9950	19.1339	4.3742
lanczos2	44	35.5982	0.9953	17.9167	4.2328
lanczos3	28	38.5934	0.9943	8.9896	2.9983
lanczos4	23	39.7711	0.9913	6.8545	2.6181
schaum2	23	41.1192	**0.9987**	5.0253	2.2417
schaum3	52	34.0538	0.9930	25.5679	5.0565
bspline2	32	37.9017	0.9963	10.5417	3.2468
bspline3	32	37.9901	0.9937	10.3294	3.2139
bspline5	22	40.5412	0.9897	5.7407	2.3960
bspline7	17	41.7125	0.9869	4.3836	2.0937
bspline9	15	42.9595	0.9838	3.2895	1.8137
bspline11	**14**	**43.7995**	0.9828	**2.7110**	**1.6465**
omoms3	23	40.3572	0.9938	5.9891	2.4473
omoms5	19	41.0794	0.9893	5.0715	2.2520
omoms7	15	42.4439	0.9864	3.7041	1.9246

Table A.6 – Results for the first image.

can be seen in tables A.6, A.7 and A.8. It can be perceived now that every method gives different results. The nearest neighbor interpolation is ignored in this test because the nature of the performed test implies always zero approximation error.

As expected the results of table A.6 are better than in table A.3. Again Schaum2 gets the best SSIM due to the same reasons, however now it does also obtain excellent results on the PSNR metric, only comparable to higher order (and more costly) methods.

In table A.7, similar to the previous case, results are better than in table A.4 implying our assumption that shifting by half a pixel achieves the biggest approximation error possible. Interesting about these results is that again Schaum2 obtains not only good results in terms of MSE and SSIM, comparable to higher order methods such as bspline7 and omoms7, but also that it has the lowest $MaxAbsDiff$ score.

Finally, results of performing this test on image 3 are found in table A.8. Again the same pattern of Schaum2 performing as good as the higher order methods is present for this case.

It has to be noted that the supremacy of the Fourier interpolation method over the others should not be considered. This can be understood because when shifting the image in one direction and then shifting it back, the Fourier method is exact. The other methods, since they work in the spatial domain, not only they are not exact because they try to approximate a function, but also this approximation error is performed twice, first when shifting into the chosen direction, and second when shifting back. This implies that the performed test is not fair for the spatial domain methods, however since no prior knowledge of the problem in which interpolation should be used, and thus, no ground truth information is available, this test seemed a good approach for estimating the accuracy of the methods. If a way to measure intermediate results other than by visual perception would be available, then artifacts produced by Fourier interpolation would discourage its use, and the obtained error would be considerably higher, while the other methods would produce similar results. Unluckily, these problems could not be captured by the proposed test method.

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
Fourier	0	∞	1	0	0
bilinear	142	26.6153	0.8998	141.7585	11.9062
quadInterp	100	29.8499	0.9555	67.3113	8.2043
bicubic	101	30.2592	0.9601	61.2576	7.8267
lanczos2	99	30.4228	0.9618	58.9928	7.6807
lanczos3	93	32.9321	0.9788	33.1034	5.7536
lanczos4	89	34.0843	0.9829	25.3895	5.0388
schaum2	**69**	35.4712	0.9892	18.4487	4.2952
schaum3	116	29.1232	0.9472	79.5711	8.9203
bspline2	90	32.5147	0.9770	36.4430	6.0368
bspline3	93	32.4658	0.9763	36.8552	6.0709
bspline5	89	34.6317	0.9849	22.3827	4.7310
bspline7	87	35.9127	0.9878	16.6654	4.0823
bspline9	86	36.8491	0.9892	13.4328	3.6651
bspline11	85	**37.5954**	**0.9901**	**11.3120**	**3.3633**
omoms3	88	34.6185	0.9855	22.4508	4.7382
omoms5	88	35.2247	0.9865	19.5257	4.4188
omoms7	86	36.3364	0.9885	15.1161	3.8879

Table A.7 – Results for the second image.

Method	MaxAbsDiff	PSNR	SSIM	MSE	RMSE
Fourier	0	∞	1	0	0
bilinear	42	26.1686	0.8077	157.1142	12.5345
quadInterp	32	29.1778	0.9100	78.5777	8.8644
bicubic	37	29.4530	0.9149	73.7535	8.5880
lanczos2	37	29.6075	0.9183	71.1752	8.4365
lanczos3	34	31.8706	0.9547	42.2690	6.5015
lanczos4	30	32.9629	0.9655	32.8695	5.7332
schaum2	**22**	34.5612	0.9774	22.7488	4.7696
schaum3	43	28.3708	0.8874	94.6249	9.7275
bspline2	31	31.5245	0.9499	45.7756	6.7658
bspline3	32	31.4294	0.9487	46.7882	6.8402
bspline5	28	33.4657	0.9697	29.2762	5.4107
bspline7	26	34.6997	0.9781	22.0347	4.6941
bspline9	26	35.5805	0.9827	17.9900	4.2415
bspline11	27	**36.2902**	**0.9859**	**15.2778**	**3.9087**
omoms3	27	33.4635	0.9698	29.2908	5.4121
omoms5	27	34.0415	0.9738	25.6408	5.0637
omoms7	26	35.1030	0.9804	20.0806	4.4811

Table A.8 – Results for the third image.

A.4.2 Qualitative results

In this section, every interpolation method is evaluated first by performing half a pixel shift in both directions, and second, by shifting them back and observing the visual results.

Intermediate Interpolation Results

To visually evaluate the algorithms, the input images were first shifted $v_x = 0.5, v_y = 0.5$ pixels. Results are shown in figures A.6, A.7 and A.8 respectively. For the first image, because it is not bandlimited, ripple artifacts can be perceived next to the steep edges. Since high order methods better approximate Fourier interpolation, they also present some artifacts, however less extended over the image. As for lower interpolation methods, results seem better and no artifacts are produced, however they tend to blur the edges.

For the second image of figure A.7, except for bilinear interpolation which heavily blurs the image, every other method has acceptable results. Higher order methods present some minor artifacts that can be perceived by zooming in the image. As for the other methods, visual difference seems not to be significant and thus, quadratic interpolation, which offers the lowest computational cost, seems to be the best option.

Finally, for the third image, shown in fig. A.8, lower order interpolation methods blur the image removing both noise and detail. As for higher order methods, their results seem fine, however they also add some minor blur. It can be noted as well, specially on the noisy background and on the central ship, that Fourier interpolation seems to add fine grained noise to the results.

An important remark is that Fourier interpolation assumes periodical images and thus, when performing the image shifting it can be seen, in particular on smaller images like image 3, how the other edges of the image are transfered to the edges where new values should be put. This does not occur with the other methods since mirror symmetry is assumed.

Final Results

In this section we analyze the resulting quality, using the same images, of applying the interpolation method to shift the images $v_x = 0.5, v_y = 0.5$ pixels into one direction and then shifting them back $v_x = -0.5, v_y = -0.5$ pixels into its original position. Ideally, the resulting image should be exactly the same as the original, however some interpolation error introduced by each algorithm can be perceived. In figure A.9, edges are heavily blurred on interpolation methods such as Bilinear and Biquadratic, while higher order methods such as BSpline9, BSpline11 and oMoms7 introduce visible ripple artifacts close to the edges. Fourier interpolation produces zero error due, again, to the nature of the performed test.

On figure A.10, again lower order methods usually blur the image, while higher order produce ripple artifacts. However, this time, since the input image is slightly better sampled, these artifacts are less perceivable. Fourier interpolation, because of its invertible nature, again obtains the same original image.

On figure A.11, the results of applying the evaluation method to a small noisy image is shown. In this case, as in both previous examples, lower order methods blur the image.

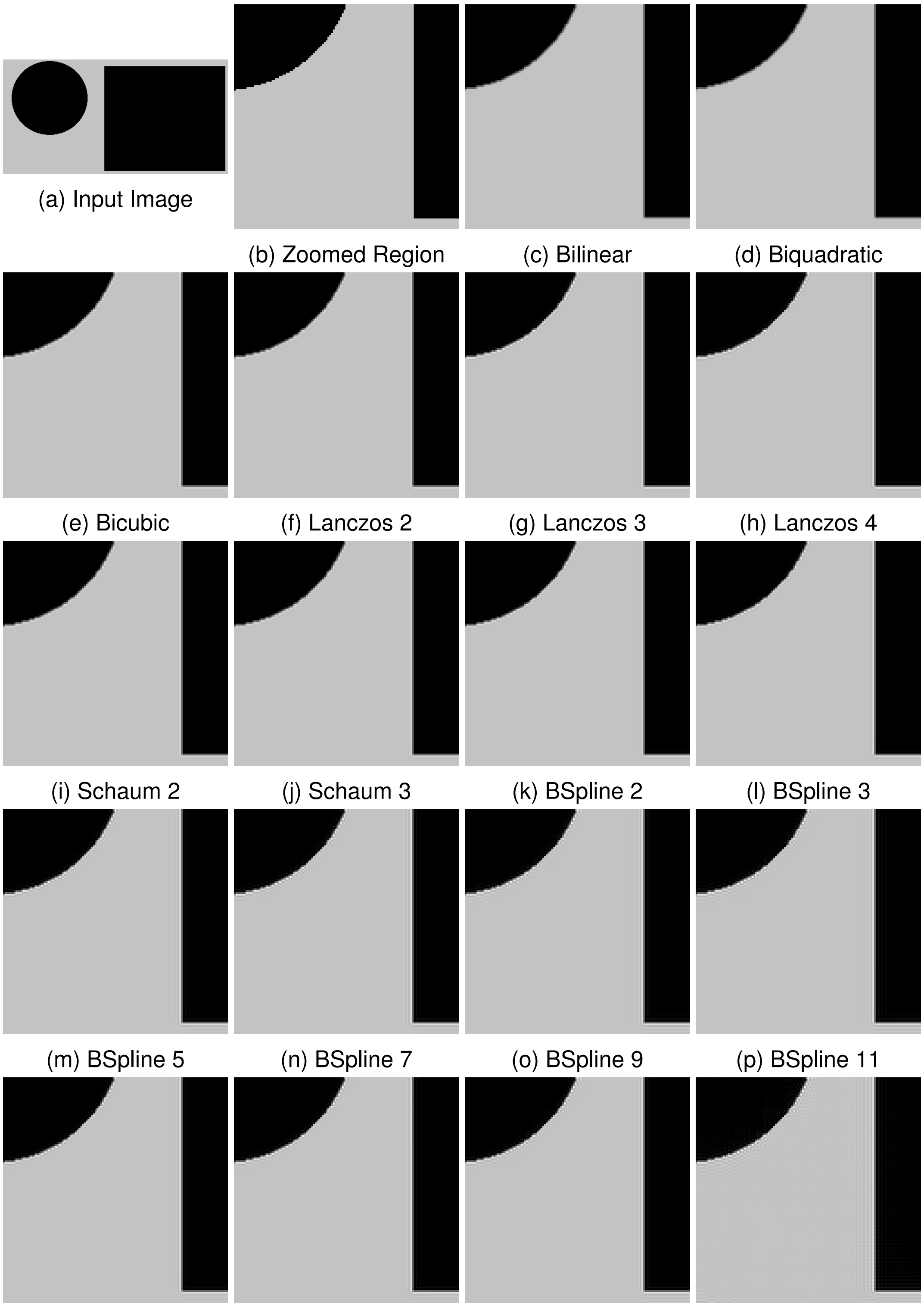
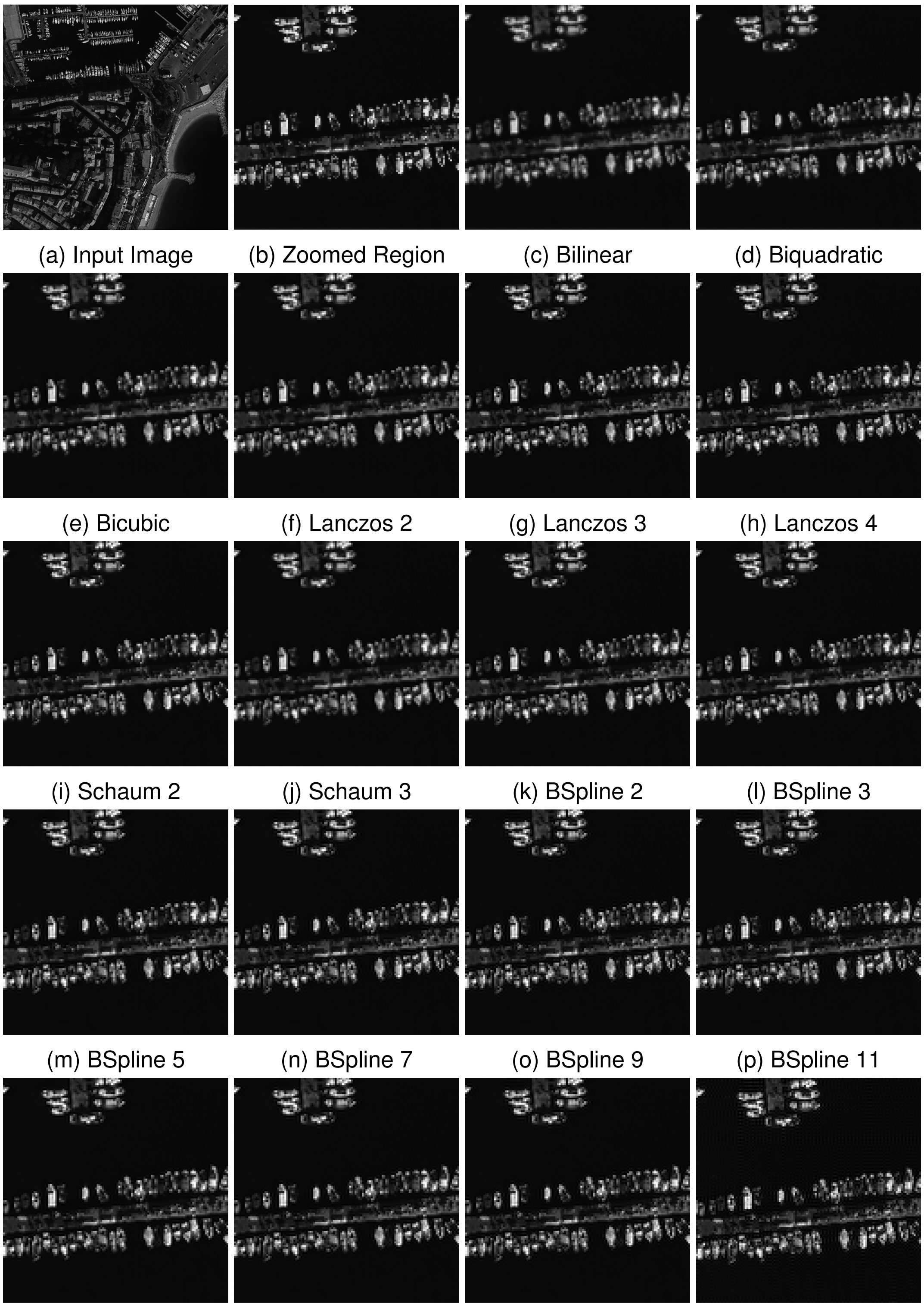

(a) Input Image
(b) Zoomed Region
(c) Bilinear
(d) Biquadratic
(e) Bicubic
(f) Lanczos 2
(g) Lanczos 3
(h) Lanczos 4
(i) Schaum 2
(j) Schaum 3
(k) BSpline 2
(l) BSpline 3
(m) BSpline 5
(n) BSpline 7
(o) BSpline 9
(p) BSpline 11
(q) Moms 3
(r) Moms 5
(s) Moms 7
(t) Fourier

Figure A.6 – Image shifting results on image with steep edges. Important ripple artifacts could be observed in several high order methods.



(a) Input Image (b) Zoomed Region (c) Bilinear (d) Biquadratic

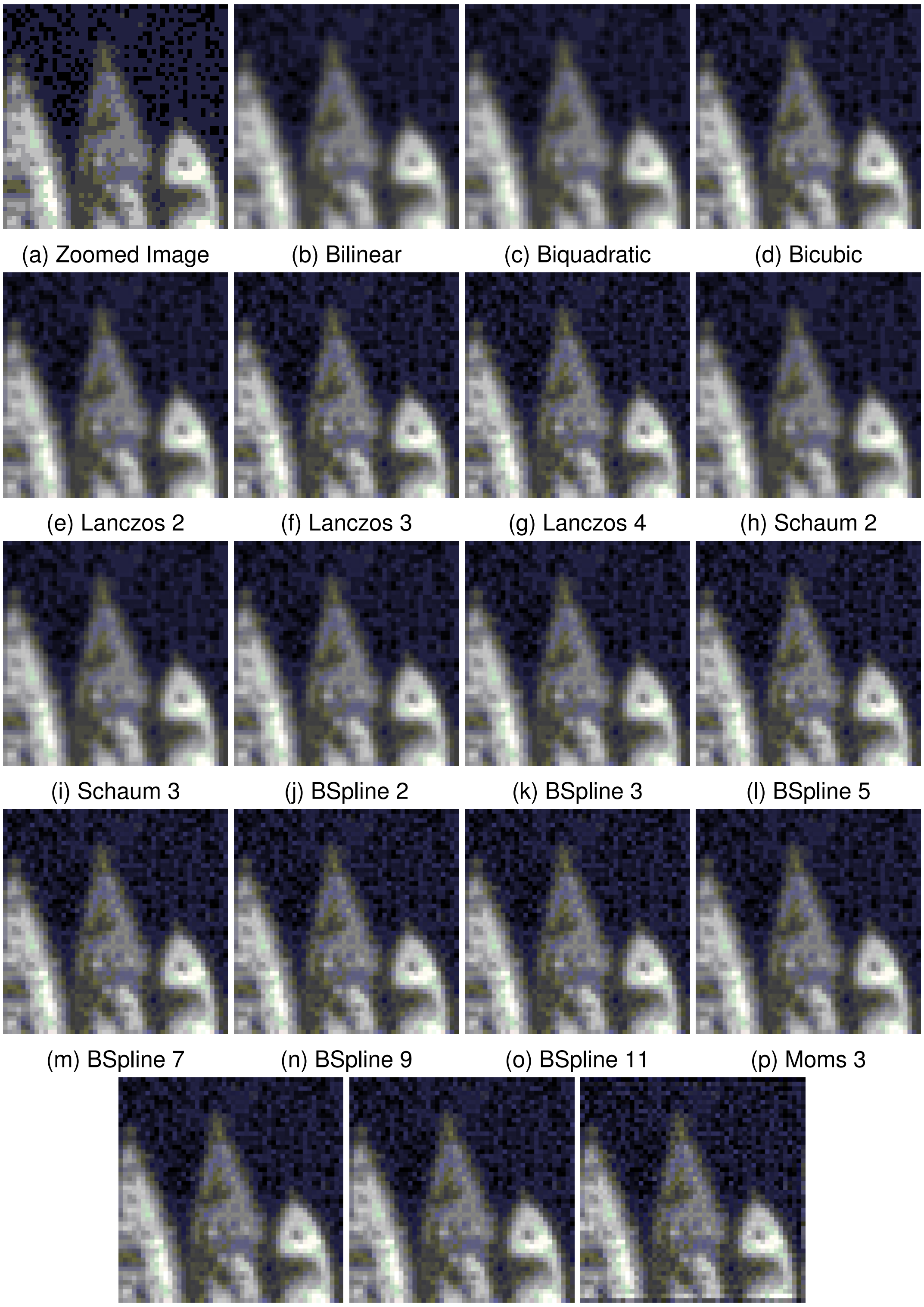
(e) Bicubic (f) Lanczos 2 (g) Lanczos 3 (h) Lanczos 4

(i) Schaum 2 (j) Schaum 3 (k) BSpline 2 (l) BSpline 3

(m) BSpline 5 (n) BSpline 7 (o) BSpline 9 (p) BSpline 11

(q) Moms 3 (r) Moms 5 (s) Moms 7 (t) Fourier

Figure A.7 – Image shifting results on map image. Ripple artifacts exist in methods with higher complexity, specially under Fourier interpolation, however not easily perceivable.



(a) Zoomed Image (b) Bilinear (c) Biquadratic (d) Bicubic

(e) Lanczos 2 (f) Lanczos 3 (g) Lanczos 4 (h) Schaum 2

(i) Schaum 3 (j) BSpline 2 (k) BSpline 3 (l) BSpline 5

(m) BSpline 7 (n) BSpline 9 (o) BSpline 11 (p) Moms 3

(q) Moms 5 (r) Moms 7 (s) Fourier

Figure A.8 – Image shifting results on noisy image. Interpolation methods tend to blur the image removing both noise and detail. Fourier interpolation seems to add fine grained noise.

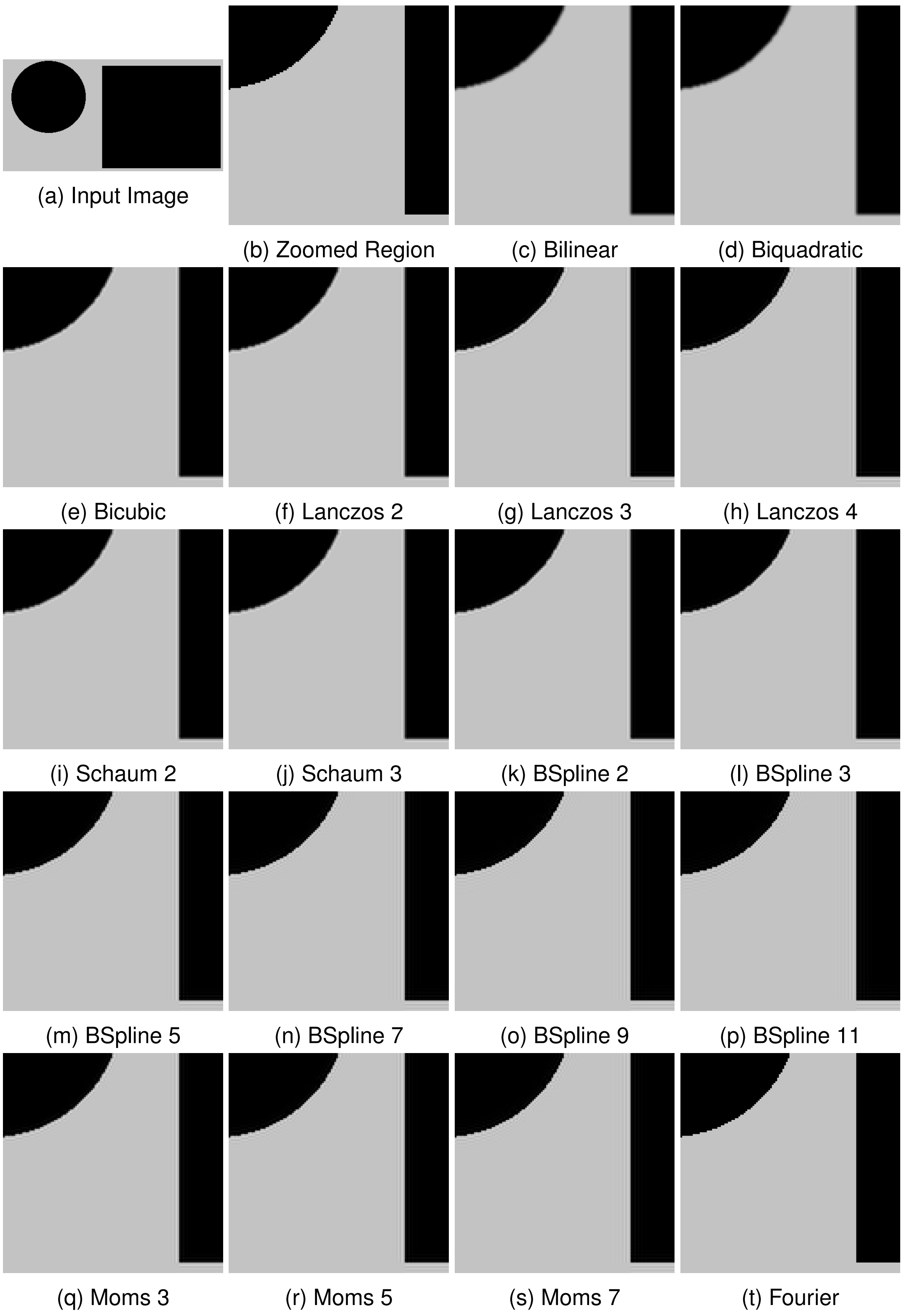
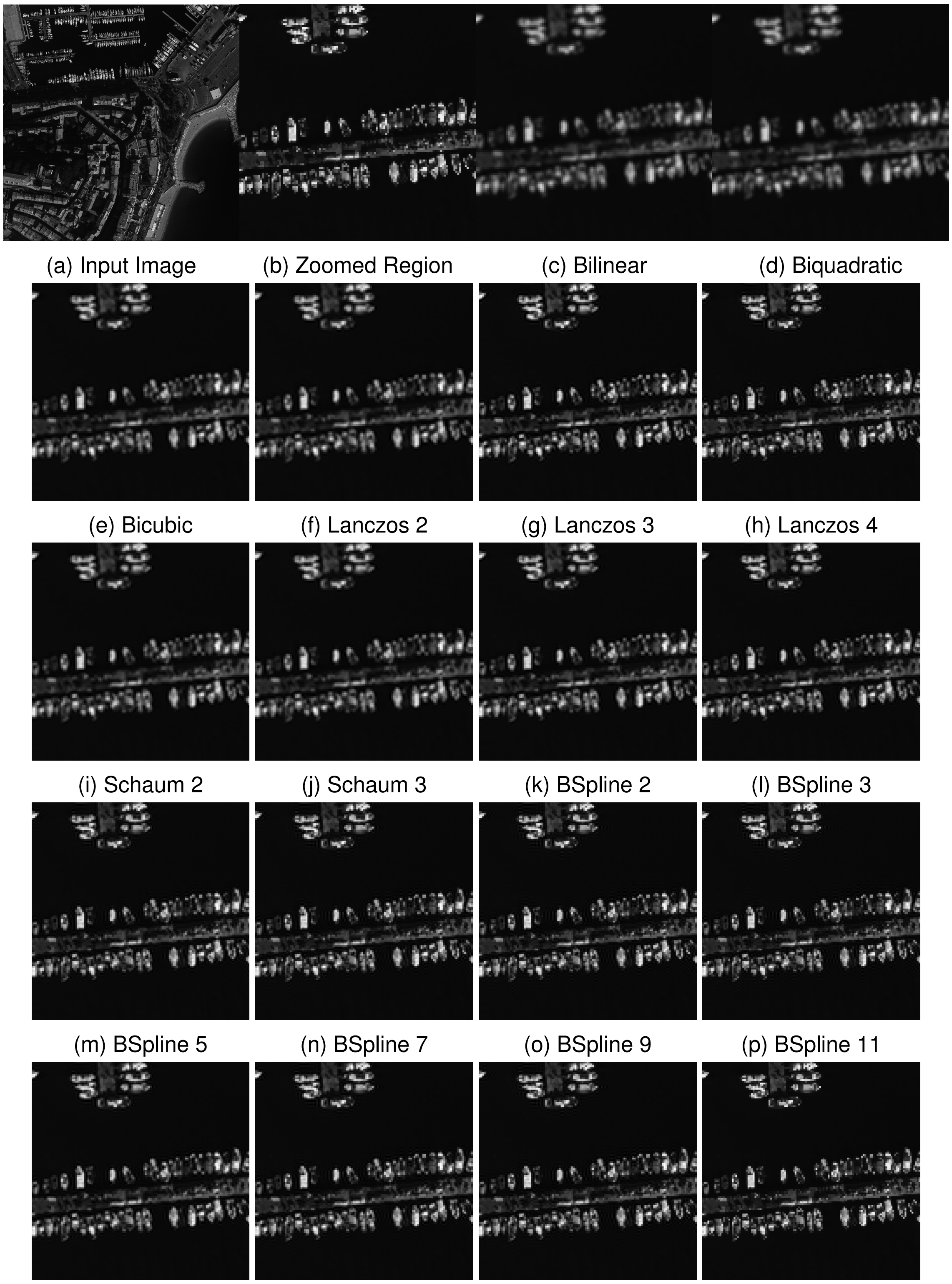


Figure A.9 – Image shifting results on image 1. Blur can be perceived on edges on interpolation algorithms with small support kernels.



(a) Input Image (b) Zoomed Region (c) Bilinear (d) Biquadratic

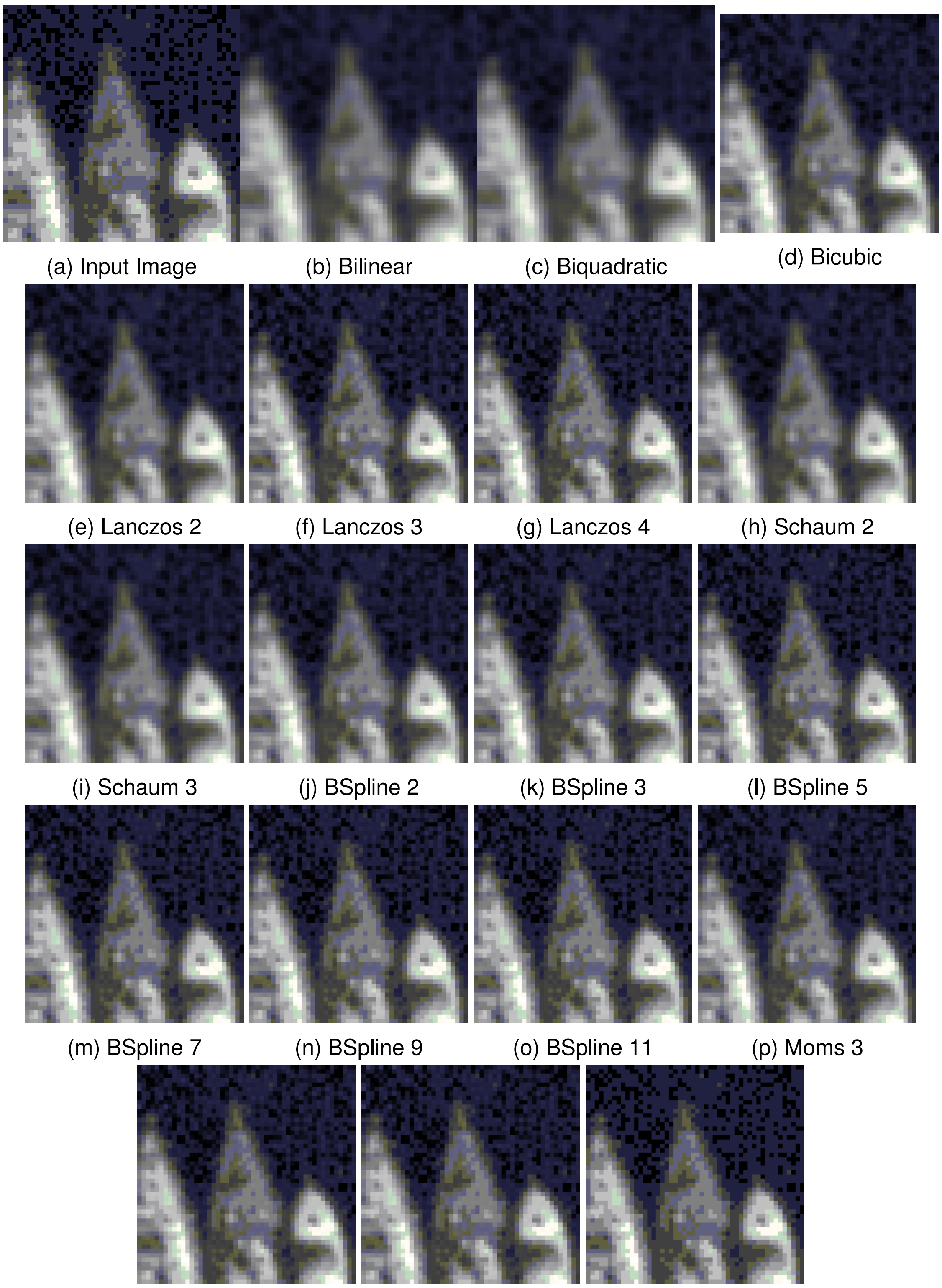
(e) Bicubic (f) Lanczos 2 (g) Lanczos 3 (h) Lanczos 4

(i) Schaum 2 (j) Schaum 3 (k) BSpline 2 (l) BSpline 3

(m) BSpline 5 (n) BSpline 7 (o) BSpline 9 (p) BSpline 11

(q) Moms 3 (r) Moms 5 (s) Moms 7 (t) Fourier

Figure A.10 – Image shifting results on image 2.



(a) Input Image (b) Bilinear (c) Biquadratic (d) Bicubic

(e) Lanczos 2 (f) Lanczos 3 (g) Lanczos 4 (h) Schaum 2

(i) Schaum 3 (j) BSpline 2 (k) BSpline 3 (l) BSpline 5

(m) BSpline 7 (n) BSpline 9 (o) BSpline 11 (p) Moms 3

(q) Moms 5 (r) Moms 7 (s) Fourier

Figure A.11 – Image shifting results on image 3.

A.4.3 Execution Time

To evaluate the execution time, each method was used to shift the map image (image 2) again $v_x = 0.5, v_y = 0.5$ pixels. This procedure was repeated 500 times and the average running time for each algorithm was measured. The results are shown in table A.9. Interestingly, the bilinear and the nearest neighbor interpolation took the same time, however it has to be noted that since the nearest neighbor method has practically no computational burden, the overhead of the implementation probably dominates the execution and thus no speedup could be perceived. On the other side, the quadratic interpolation is, as claimed by its authors [44], considerably faster than the bicubic, and its results are visually and numerically better than bilinear interpolation. Also, Fourier interpolation executed 20% faster than bicubic interpolation, while its results are among the best. In this case, the FFTW library was used, which is a highly optimized FFT library. For the rest, it has to be noted that both Schaum methods took the same time as bicubic while its results are generally better. Finally, high order methods such as bsplines, oMoms and Lanczos take excesively long time and its use is discouraged. At the same time, lower order bsplines and oMoms, such as bspline3 and oMom3, offer fine results without taking much execution time.

Method	Time (seconds)
Fourier	0.450
nearest	0.235
bilinear	0.235
quadInterp	0.371
bicubic	0.554
lanczos2	0.556
lanczos3	1.101
lanczos4	1.817
schaum2	0.554
schaum3	0.554
bspline2	0.498
bspline3	0.682
bspline5	1.347
bspline7	2.186
bspline9	3.229
bspline11	4.472
omoms3	0.683
omoms5	1.347
omoms7	2.187

Table A.9 – Average processing time for each method for shifting image 2.

A.4.4 Combining time and approximation error

In order to give an estimate about the best methods in terms of both speed and precision, a performance index is introduced. If MSE_{method} is the resulting MSE value for a particular method, $\max(MSE)$ represent the maximum MSE value over all evaluated methods and T_{method} stands for the total execution time of $method$, then the performance index is calculated in the following way:

$$P_{method} = T_{method} \frac{MSE_{method}}{\max(MSE)} \tag{A.55}$$

It is worth remarking that Fourier interpolation method is not included in this list since the performed evaluation is suited for Fourier interpolation and thus, would always be the best interpolation algorithm possible. It is also necessary to clarify that for the MSE measurements, an average of both performed test scores was used, always for the same image (image 2). Results are shown in figure A.12. oMoms3 shows up as the most recommended method, followed by BSpline2. Then Schaum2 and BSpline3 appear to be a good compromise between precision and processing cost, while bilinear interpolation comes just after those, specially because of its low execution time.

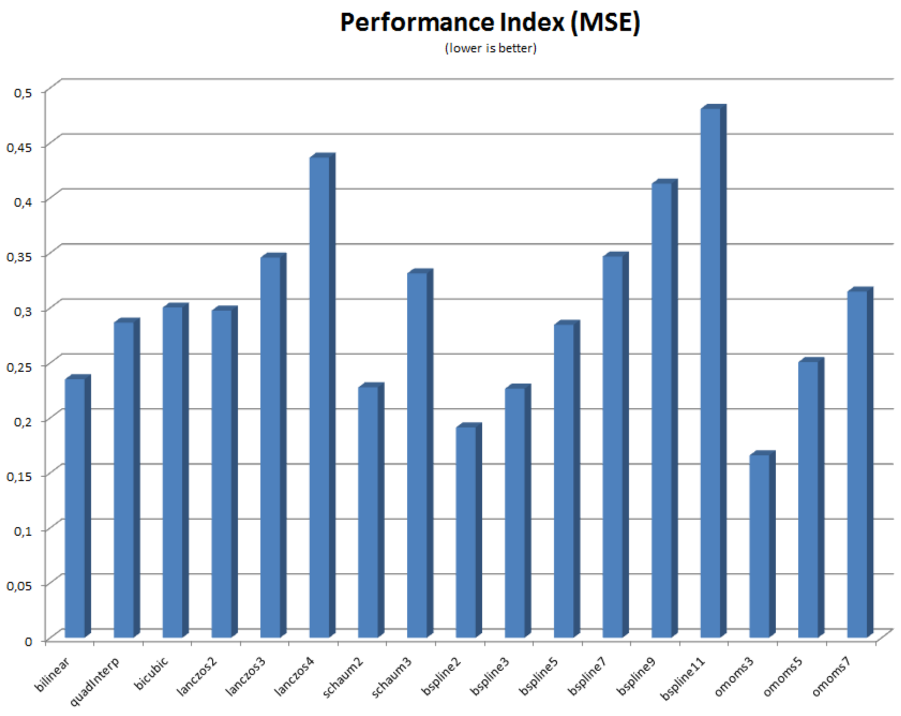


Figure A.12 – Performance Index for all methods based on the MSE.

The same evaluation was performed for the RMSE score in figure A.13, and in this case, another method that appears interesting is the quadratic interpolation. On the other side, only if high precision is required, methods with long support such as bspline11 or lanczos4 should be avoided.

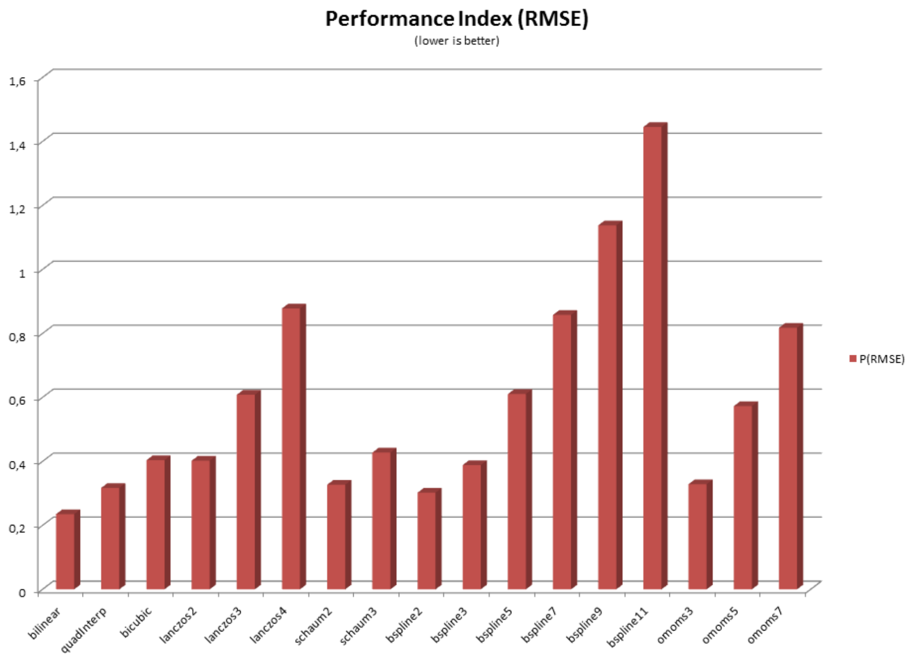


Figure A.13 – Performance Index for all methods based on the RMSE.

Appendix B

Fast Anscombe VST implementation for fast on board noise stabilization

As explained in section 4.3, we apply systematically an Anscombe transform to all images, $\tilde{I} \to 2\sqrt{\frac{3}{8} + I}$. If the noise is pure Poisson, this transform is known to make it into an almost white additive Gaussian noise. By applying it to our data, we get closer to this situation, and this tightens slightly the noise estimates.

However, since this transform involves applying a square root, its computation on the satellite hardware may be restrictive, as indicated by the hardware development team of CNES. Therefore, an alternative to the square root has to be analyzed. Based on the analysis of El-Magdoub [49], the inverse square root approximation algorithm used by Quake 3 seemed to suit our needs for both precision and speed. This algorithm takes advantage of the 32-bit representation of floating point values, thus, allowing it only to be run on a 32-bit architecture. Its code is the following:

```
#define SQRT_MAGIC_F 0x5f3759df
float sqrt(const float x)
{
  const float xhalf = 0.5f*x;
  union // get bits for floating value
  {
    float x;
    int i;
  } u;
  u.x = x;
  u.i = SQRT_MAGIC_F - (u.i >> 1);  // gives initial guess y0
  invsqrt = x*u.x*(1.5f - xhalf*u.x*u.x);// Newton step, repeating increases accuracy
  return 1/invsqrt;
}
```

Experiments showed similar performance when applying this approximation, which implies that it can be implemented in the satellite.

Appendix C

Algorithms for gradient-based shift estimation used in the Stab-Active application

For the sake of completeness, we describe here the rest of the algorithms used in the Stab-Active project. Algorithm 20 estimates the noise for the whole line, using Eq. (4.2). The *hypomode* gradient estimation method is described in algorithm 21. Finally, algorithm 23 describes the gradient-based shift estimation procedure. This algorithm indeed resembles Alg. 1 from the first chapter, although it is explained in detail and optimized to avoid unnecessary computations.

Algorithm 20 Noise estimation, two consecutive noisy frames

```
**Input**: $I_i(x, y), i \in [1, \dots, N]$ N noisy images defined on the image domain $D$ with size $N = 50 \times 50$ pixels

**Output**: average noise variance $\hat{\sigma}^2$

$\hat{\sigma}^2 := \frac{1}{N-1} \sum_{i=1}^{N-1} \left[ \frac{1}{2|S|} \sum_{x,y} (I^*_{i+1}(x, y) - I^*_i(x, y))^2 \right]$
```

Algorithm 21 Compute Gradient

```
**Input**: $I(x, y)$ input image with size $N = 50 \times 50$ pixels
**Output**: $I_x, I_y$ gradient of I

**for** $x, y$ between $(1, 1)$ and $(49, 49)$ **do**
    $I_x(x, y) \leftarrow \frac{I(x+1,y)-I(x,y)+I(x+1,y+1)-I(x,y+1)}{2}$
    $I_y(x, y) \leftarrow \frac{I(x,y+1)-I(x,y)+I(x+1,y+1)-I(x+1,y)}{2}$
**end for**
```

Algorithm 22 Temporal convolution

```
**Input**: $I(t), 1 \leq t \leq 64$ images, $p \in \mathbb{N}$.
**Output**: $I_D(t)$ denoised version of $I(t)$ by temporal sliding average.

**for** $i = p + 1$ to $64 - p$ **do**
    $I_D(i - p) \leftarrow \sum_{j=i-p}^{i+p} I(j)$
**end for**
```

Algorithm 23 EstimateShift: gradient-based shift estimation

Input: $I_1(x,y)$, $I_2(x,y)$ two images (generally obtained after temporal convolution) defined on the image domain D with size $N = 50 \times 50$ pixels, $I_x(x,y), I_y(x,y)$ gradient (*hypomode*) for both dimensions of I_1, $itQty$ amount of iterations for the algorithm, $interpolationMethod$ interpolation method used,
Output: $v = (v_x, v_y)^t$ motion vector.

```
// Since the gradient is computed using forward differences, precision is improved by using $I(x+0.5, y+0.5)$
// Do $I_1(x,y) \leftarrow I_1(x+0.5, y+0.5)$ using bilinear interpolation
for $x, y$ between $(1,1)$ and $(49,49)$ do
    $I_1(x,y) \leftarrow \frac{I_1(x,y)+I_1(x,y+1)+I_1(x+1,y)+I_1(x+1,y+1)}{4}$
end for
$sI_x^2 \leftarrow \sum_{x,y=1}^{49} I_x(x,y)^2$, $sI_y^2 \leftarrow \sum_{x,y=1}^{49} I_y(x,y)^2$
$sI_xI_y \leftarrow \sum_{x,y=1}^{49} I_x(x,y)I_y(x,y)$
$Det \leftarrow sI_x^2 sI_y^2 - (sI_xI_y)^2$
$I_A \leftarrow I_2$
for $i = 1$ to $itQty$ do
    for $x, y$ between $(1,1)$ and $(49,49)$ do
        // Do $I_A(x,y) \leftarrow I_A(x+0.5, y+0.5)$ using bilinear interpolation (for the same reason as before)
        $I_A(x,y) \leftarrow \frac{I_A(x,y)+I_A(x,y+1)+I_A(x+1,y)+I_A(x+1,y+1)}{4}$
        $I_t(x,y) \leftarrow I_A(x,y) - I_1(x,y)$, a column vector with $49^2$ components
    end for
    $sI_xI_t \leftarrow \sum_{x,y=1}^{49} I_t(x,y)I_x(x,y)$
    $sI_yI_t \leftarrow \sum_{x,y=1}^{49} I_t(x,y)I_y(x,y)$
    $v_x \leftarrow \frac{1}{Det}(sI_xI_t\ sI_y^2 - sI_yI_t\ sI_xI_y)$, $v_y \leftarrow \frac{1}{Det}(sI_yI_t\ sI_x^2 - sI_xI_t\ sI_xI_y)$
    if $i < itQty$ then
        $I_A \leftarrow Interpolate\,(I_2, -v_x, -v_y, interpolationMethod)$
    end if
end for
```

Bibliography

[1] Ikram E. Abdou. Practical approach to the registration of multiple frames of video images, 1998. URL: http://dx.doi.org/10.1117/12.334685, doi:10.1117/12.334685[1].

[2] Andrew Adams, Natasha Gelfand, and Kari Pulli. Viewfinder alignment. *Computer Graphics Forum*, 27(2):597–606, 2008.

[3] Cecilia Aguerrebere, Mauricio Delbracio, Alberto Bartesaghi, and Guillermo Sapiro. Fundamental limits in multi-image alignment. *CoRR*, abs/1602.01541, 2016. URL: http://arxiv.org/abs/1602.01541.

[4] Alfonso Alba, J. Flavio Vigueras-Gomez, Edgar R. Arce-Santana, and Ruth M. Aguilar-Ponce. Phase correlation with sub-pixel accuracy: A comparative study in 1d and 2d. *Computer Vision and Image Understanding*, 137:76 – 87, 2015. URL: http://www.sciencedirect.com/science/article/pii/S1077314215000685, doi:10.1016/j.cviu.2015.03.011[2].

[5] S. Alliney and C. Morandi. Digital image registration using projections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(2):222–233, March 1986. doi:10.1109/TPAMI.1986.4767775[3].

[6] F. J. Anscombe. The transformation of poisson, binomial and negative-binomial data. *Biometrika*, 35(3/4):246–254, 1948. URL: http://www.jstor.org/stable/2332343.

[7] V. Argyriou and T. Vlachos. Using gradient correlation for sub-pixel motion estimation of video sequences. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, volume 3, pages iii–329–32 vol.3, May 2004. doi:10.1109/ICASSP.2004.1326548[4].

[8] V. Argyriou and T. Vlachos. A study of sub-pixel motion estimation using phase correlation. In *Proc. BMVC*, pages 40.1–40.10, 2006. doi:10.5244/C.20.40.

[9] V. Argyriou and T. Vlachos. Quad-tree motion estimation in the frequency domain using gradient correlation. *IEEE Transactions on Multimedia*, 9(6):1147–1154, Oct 2007. doi:10.1109/TMM.2007.898926[5].

[10] Alireza Bab-Hadiashar and David Suter. Robust optic flow computation. *International Journal of Computer Vision*, 29(1):59–77, 1998.

[1] http://dx.doi.org/10.1117/12.334685
[2] http://dx.doi.org/10.1016/j.cviu.2015.03.011
[3] http://dx.doi.org/10.1109/TPAMI.1986.4767775
[4] http://dx.doi.org/10.1109/ICASSP.2004.1326548
[5] http://dx.doi.org/10.1109/TMM.2007.898926

[11] Alireza Bab-Hadiashar and David Suter. Robust total least squares based optic flow computation. In *Proc. 3rd ACCV, Hong Kong*, pages 566–573, 1998.

[12] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56(3):221–255, 2004.

[13] M. Balci and H. Foroosh. Estimating sub-pixel shifts directly from the phase difference. In *IEEE International Conference on Image Processing 2005*, volume 1, pages I–1057–60, Sept 2005. doi:10.1109/ICIP.2005.1529936[6].

[14] Shaul K. Bar-Lev and Peter Enis. On the classical choice of variance stabilizing transformations and an application for a poisson variate. *Biometrika*, 75(4):803–804, 1988. URL: http://biomet.oxfordjournals.org/content/75/4/803.abstract, doi:10.1093/biomet/75.4.803[7].

[15] M. S. Bartlett. The square root transformation in analysis of variance. *Supplement to the Journal of the Royal Statistical Society*, 3(1):68–78, 1936. URL: http://www.jstor.org/stable/2983678.

[16] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[17] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995.

[18] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE transactions on pattern analysis and machine intelligence*, 24(4):509–522, 2002.

[19] J. W. Bergstrom, R. W. Dissly, W. A. Delamere, A. McEwen, and L. Keszthelyi. High-Resolution Multi-Color Pushbroom Imagers for Mars Surface Characterization and Landing Safety. *LPI Contributions*, 1679:4332, June 2012.

[20] G. M. Bernstein and D. Gruen. Resampling Images in Fourier Domain. *Publications of the Astronomical Society of the Pacific*, 126:287–295, March 2014. arXiv:1401.2636[8], doi:10.1086/675812[9].

[21] A. Bonnefois et al. Comparative theoretical and experimental study of a Shack-Hartmann and a Phase Diversity Sensor, for high-precision wavefront sensing dedicated to Space Active Optics. In *ICSO 2014*, October 2014.

[22] Martin J Booth. Adaptive optics in microscopy. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 365(1861):2829–2843, 2007. doi:10.1098/rsta.2007.0013[10].

[23] Tom Botterill, Steven Mills, and Richard Green. Refining essential matrix estimates from ransac. In *Image and Vision Computing New Zealand*, pages 500–505, 2011.

[24] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM computing surveys (CSUR)*, 24(4):325–376, 1992.

[6] http://dx.doi.org/10.1109/ICIP.2005.1529936
[7] http://dx.doi.org/10.1093/biomet/75.4.803
[8] http://arxiv.org/abs/1401.2636
[9] http://dx.doi.org/10.1086/675812
[10] http://dx.doi.org/10.1098/rsta.2007.0013

[25] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, 2006. URL: http://dx.doi.org/10.1007/s11263-006-0002-3, doi:10.1007/s11263-006-0002-3[11].

[26] R. Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley, 2009. URL: https://books.google.fr/books?id=TlnWjwEACAAJ.

[27] Antoni Buades, Gloria Haro, and Enric Meinhardt-Llopis. Obtaining High Quality Photographs of Paintings by Image Fusion. *Image Processing On Line*, 5:159–175, 2015. doi:10.5201/ipol.2015.49[12].

[28] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. *Communications, IEEE Transactions on*, 31(4):532–540, 1983.

[29] Haiyang Chao, Yu Gu, and Marcello Napolitano. A survey of optical flow techniques for robotics navigation applications. *Journal of Intelligent & Robotic Systems*, 73(1):361–372, 2014. URL: http://dx.doi.org/10.1007/s10846-013-9923-6, doi:10.1007/s10846-013-9923-6[13].

[30] Peter C Chen, Charles W Bowers, Marzouk Marzouk, and Robert C Romeo. Advances in very lightweight composite mirror technology. *Optical Engineering*, 39(9):2320–2329, 2000.

[31] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, Aug 1995. doi:10.1109/34.400568[14].

[32] Tat-Jun Chin, Jin Yu, and David Suter. Accelerated hypothesis generation for multistructure data via preference analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):625–638, April 2012. URL: http://dx.doi.org/10.1109/TPAMI.2011.169, doi:10.1109/TPAMI.2011.169[15].

[33] Jongmoo Choi and G. Medioni. Starsac: Stable random sample consensus for parameter estimation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 675–682, June 2009. doi:10.1109/CVPR.2009.5206678[16].

[34] Sunglok Choi, Taemin Kim, and Wonpil Yu. Performance evaluation of ransac family. In *BMVC*. British Machine Vision Association, 2009.

[35] William J Christmas. Spatial filtering requirements for gradient-based optical flow measurement. In *BMVC*, pages 1–10, 1998.

[36] O. Chum and J. Matas. Matching with PROSAC-progressive sample consensus. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 220–226. IEEE, 2005.

[37] O. Chum and J. Matas. Optimal Randomized RANSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1472–1482, August 2008. URL: http://

[11]http://dx.doi.org/10.1007/s11263-006-0002-3
[12]http://dx.doi.org/10.5201/ipol.2015.49
[13]http://dx.doi.org/10.1007/s10846-013-9923-6
[14]http://dx.doi.org/10.1109/34.400568
[15]http://dx.doi.org/10.1109/TPAMI.2011.169
[16]http://dx.doi.org/10.1109/CVPR.2009.5206678

ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4359381, doi:10.1109/TPAMI.2007.70787[17].

[38] O. Chum, T. Werner, and J. Matas. Two-view geometry estimation unaffected by a dominant plane. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 772–779 vol. 1, June 2005. doi:10.1109/CVPR.2005.354[18].

[39] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized RANSAC. In *Pattern Recognition*, pages 236–243. Springer, 2003.

[40] J. H. Curtiss. On transformations used in the analysis of variance. *The Annals of Mathematical Statistics*, 14(2):107–122, 1943. URL: http://www.jstor.org/stable/2235813.

[41] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C Sinan Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

[42] Carl De Boor. *A practical guide to splines; rev. ed.* Applied mathematical sciences. Springer, Berlin, 2001. URL: https://cds.cern.ch/record/1428148.

[43] Julie Delon. Midway image equalization. *Journal of Mathematical Imaging and Vision*, 21(2):119–134, 2004. URL: http://dx.doi.org/10.1023/B:JMIV.0000035178.72139.2d, doi:10.1023/B:JMIV.0000035178.72139.2d[19].

[44] Neil A Dodgson. Quadratic interpolation for image resampling. *Image Processing, IEEE Transactions on*, 6(9):1322–1326, 1997.

[45] Z. Drezner and H.W. Hamacher. *Facility Location: Applications and Theory*. Springer Berlin Heidelberg, 2001. URL: https://books.google.es/books?id=qUT-f95eJ0kC.

[46] V. N. Dvornychenko. Bounds on (deterministic) correlation functions with application to registration. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-5(2):206–213, 1983. doi:10.1109/TPAMI.1983.4767373[20].

[47] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936. URL: http://dx.doi.org/10.1007/BF02288367, doi:10.1007/BF02288367[21].

[48] Churchill Eisenhart. The assumptions underlying the analysis of variance. *Biometrics*, 3(1):1–21, 1947. URL: http://www.jstor.org/stable/3001534.

[49] Mahmoud Hesham El-Magdoub. Best square root method - algorithm - function (precision vs speed). http://www.codeproject.com/Articles/69941/Best-Square-Root-Method-Algorithm-Function-Precisi, September 2010.

[50] M. Elad, P. Teo, and Y. Hel-Or. Optimal filters for gradient-based motion estimation. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 559–565 vol.1, 1999. doi:10.1109/ICCV.1999.791272[22].

[17] http://dx.doi.org/10.1109/TPAMI.2007.70787
[18] http://dx.doi.org/10.1109/CVPR.2005.354
[19] http://dx.doi.org/10.1023/B:JMIV.0000035178.72139.2d
[20] http://dx.doi.org/10.1109/TPAMI.1983.4767373
[21] http://dx.doi.org/10.1007/BF02288367
[22] http://dx.doi.org/10.1109/ICCV.1999.791272

[51] S Erturk. Digital image stabilization with sub-image phase correlation based global motion estimation. *IEEE Transactions on Consumer Electronics*, 49(4):1320–1325, 2003.

[52] C. Escolle, E. Hugot, T. Fusco, M. Ferrari, V. Michau, and T. Bret-Dibat. Adapting large lightweight primary mirror to space active optics capabilities, 2014. URL: http://dx.doi.org/10.1117/12.2055782, doi:10.1117/12.2055782[23].

[53] Hany Farid and Eero P Simoncelli. Differentiation of discrete multidimensional signals. *IEEE Transactions on Image Processing*, 13(4):496–508, 2004.

[54] Homa Fashandi, Reza Fazel-Rezai, and Stephen Pistorius. Optical flow and total least squares solution for multi-scale data in an over-determined system. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Nikos Paragios, Syeda-Mahmood Tanveer, Tao Ju, Zicheng Liu, Sabine Coquillart, Carolina Cruz-Neira, Torsten Müller, and Tom Malzbender, editors, *Advances in Visual Computing*, volume 4842 of *Lecture Notes in Computer Science*, pages 33–42. Springer Berlin Heidelberg, 2007. URL: http://dx.doi.org/10.1007/978-3-540-76856-2_4, doi:10.1007/978-3-540-76856-2_4[24].

[55] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. URL: http://doi.acm.org/10.1145/358669.358692, doi:10.1145/358669.358692[25].

[56] David J. Fleet and Yair Weiss. Optical flow estimation, 2005.

[57] H. Foroosh, J.B. Zerubia, and M. Berthod. Extension of phase correlation to subpixel registration. *Image Processing, IEEE Transactions on*, 11(3):188 –200, mar 2002. doi:10.1109/83.988953[26].

[58] Per-Erik Forssén and David G Lowe. Shape descriptors for maximally stable extremal regions. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[59] S. Fourest, X. Briottet, Ph. Lier, and Ch. Valorge. *Satellite Imagery from Acquisition Principles to Processing of Optical Images for Observing the Earth*. Cépaduès Éditions, 10 2012.

[60] J. M. Frahm and M. Pollefeys. Ransac for (quasi-)degenerate data (qdegsac). In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 453–460, June 2006. doi:10.1109/CVPR.2006.235[27].

[61] Murray F. Freeman and John W. Tukey. Transformations related to the angular and the square root. *Ann. Math. Statist.*, 21(4):607–611, 12 1950. URL: http://dx.doi.org/10.1214/aoms/1177729756, doi:10.1214/aoms/1177729756[28].

[62] Pascal Getreuer. Linear Methods for Image Interpolation. *Image Processing On Line*, 2011, 2011. http://dx.doi.org/10.5201/ipol.2011.g_lmii. doi:10.5201/ipol.2011.g_lmii[29].

[23]http://dx.doi.org/10.1117/12.2055782
[24]http://dx.doi.org/10.1007/978-3-540-76856-2_4
[25]http://dx.doi.org/10.1145/358669.358692
[26]http://dx.doi.org/10.1109/83.988953
[27]http://dx.doi.org/10.1109/CVPR.2006.235
[28]http://dx.doi.org/10.1214/aoms/1177729756
[29]http://dx.doi.org/10.5201/ipol.2011.g_lmii

[63] Dennis C Ghiglia and Mark D Pritt. *Two-dimensional phase unwrapping: theory, algorithms, and software*, volume 4. Wiley New York, 1998.

[64] Andrew Gilman and Arno Leist. Global Illumination-Invariant Fast Sub-Pixel Image Registration. pages 95–100, July 2013. URL: http://www.thinkmind.org/index.php?view=article&articleid=iccgi_2013_5_20_10195.

[65] G.H. Golub, Alan Hoffman, and G.W. Stewart. A generalization of the eckart-young-mirsky matrix approximation theorem. *Linear Algebra and its Applications*, 88:317 – 327, 1987. URL: http://www.sciencedirect.com/science/article/pii/0024379587901145, doi:10.1016/0024-3795(87)90114-5[30].

[66] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson/Prentice Hall, 2008. URL: https://books.google.fr/books?id=8uGOnjRGEzoC.

[67] A. Goshtasby, G. C. Stockman, and C. V. Page. A region-based approach to digital image registration with subpixel accuracy. *IEEE Transactions on Geoscience and Remote Sensing*, GE-24(3):390–399, May 1986. doi:10.1109/TGRS.1986.289597[31].

[68] D Gratadour, LM Mugnier, and D Rouan. Sub-pixel image registration with a maximum likelihood estimator. *Astronomy & Astrophysics*, 443(1):357–365, 2005.

[69] Manuel Guizar-Sicairos, Samuel T. Thurman, and James R. Fienup. Efficient subpixel image registration algorithms. *Opt. Lett.*, 33(2):156–158, Jan 2008. doi:10.1364/OL.33.000156[32].

[70] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[71] A. Hast, J. Nysjo, and A. Marchetti. Optimal ransac - towards a repeatable algorithm for finding the optimal set. *Journal of WSCG*, no.1:21–30, 2013.

[72] Fumio Hayashi. *Econometrics*. Princeton University Press, Princeton, 2000.

[73] Jan Herrmann. Least-squares wave front errors of minimum norm. *JOSA*, 70(1):28–35, 1980.

[74] Huy Tho Ho and Roland Goecke. Optical flow estimation using fourier mellin transform. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[75] W. S. Hoge. A subspace identification extension to the phase correlation method [mri application]. *IEEE Transactions on Medical Imaging*, 22(2):277–280, Feb 2003. doi:10.1109/TMI.2002.808359[33].

[76] William Scott Hoge. A subspace identification extension to the phase correlation method [mri application]. *Medical Imaging, IEEE Transactions on*, 22(2):277–280, 2003.

[77] H.S. Hou and H. Andrews. Cubic splines for image interpolation and digital filtering. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(6):508–517, 1978. doi:10.1109/TASSP.1978.1163154[34].

[30] http://dx.doi.org/10.1016/0024-3795(87)90114-5
[31] http://dx.doi.org/10.1109/TGRS.1986.289597
[32] http://dx.doi.org/10.1364/OL.33.000156
[33] http://dx.doi.org/10.1109/TMI.2002.808359
[34] http://dx.doi.org/10.1109/TASSP.1978.1163154

[78] Paul VC Hough. Method and means for recognizing complex patterns. Technical report, 1962.

[79] Peter J. Huber. *Robust Statistics*. Wiley, 1981.

[80] Sabine Van Huffel, Joos Vandewalle, and Ann Haegemans. An efficient and reliable algorithm for computing the singular subspace of a matrix, associated with its smallest singular values. *Journal of Computational and Applied Mathematics*, 19(3):313 – 330, 1987. URL: http://www.sciencedirect.com/science/article/pii/0377042787902019, doi:10.1016/0377-0427(87)90201-9[35].

[81] Hui Ji and Cornelia Fermüller. Noise causes slant underestimation in stereo and motion. *Vision Research*, 46(19):3105 – 3120, 2006. URL: http://www.sciencedirect.com/science/article/pii/S0042698906002124, doi:10.1016/j.visres.2006.04.010[36].

[82] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160, Dec 1981. doi:10.1109/TASSP.1981.1163711[37].

[83] S.J. KIRKPATRICK and D.D. DUNCAN. Performance of a gradient-based shift estimator in a spatially sparse data environment: tracking the sub-pixel motion of fluorescent particles. *Journal of Microscopy*, 232(1):64–72, 2008. URL: http://dx.doi.org/10.1111/j.1365-2818.2008.02087.x, doi:10.1111/j.1365-2818.2008.02087.x[38].

[84] Claude Knaus and Matthias Zwicker. Dual-domain image denoising. In *ICIP*, pages 440–444, 2013.

[85] Per Knutsson, Mette Owner-Petersen, and Chris Dainty. Extended object wavefront sensing based on the correlation spectrum phase. *Optics express*, 13(23):9527–9536, 2005. doi:10.1364/OPEX.13.009527[39].

[86] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. *IEEE Conference on Cybernetics and Society*, pages 163–165, 1975.

[87] Z. Kukelova, J. Heller, M. Bujnak, and T. Pajdla. Radial distortion homography. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 639–647, June 2015. doi:10.1109/CVPR.2015.7298663[40].

[88] Karel Lebeda, Jiri Matas, and Ondrej Chum. Fixing the locally optimized ransac–full experimental evaluation. In *British Machine Vision Conference*, pages 1–11. Citeseer, 2012.

[89] Thomas Martin Lehmann, Claudia Gönner, and Klaus Spitzer. Survey: Interpolation methods in medical image processing. *IEEE Trans. Med. Imaging*, 18(11):1049–1075, 1999. URL: http://dblp.uni-trier.de/db/journals/tmi/tmi18.html#LehmannGS99.

[35]http://dx.doi.org/10.1016/0377-0427(87)90201-9
[36]http://dx.doi.org/10.1016/j.visres.2006.04.010
[37]http://dx.doi.org/10.1109/TASSP.1981.1163711
[38]http://dx.doi.org/10.1111/j.1365-2818.2008.02087.x
[39]http://dx.doi.org/10.1364/OPEX.13.009527
[40]http://dx.doi.org/10.1109/CVPR.2015.7298663

[90] S. Leprince, S. Barbot, F. Ayoub, and J. P. Avouac. Automatic and precise orthorectification, coregistration, and subpixel correlation of satellite images, application to ground deformation measurements. *IEEE Transactions on Geoscience and Remote Sensing*, 45(6):1529–1558, June 2007. doi:10.1109/TGRS.2006.888937[41].

[91] Marloes Maria Johanna Letteboer, Peter WA Willems, Max A Viergever, and Wiro J Niessen. Brain shift estimation in image-guided neurosurgery using 3-d ultrasound. *IEEE Transactions on Biomedical Engineering*, 52(2):268–276, 2005.

[92] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944. URL: http://www.jstor.org/stable/43633451.

[93] A. Levin and B. Nadler. Natural image denoising: Optimality and inherent bounds. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2833–2840, June 2011. doi:10.1109/CVPR.2011.5995309[42].

[94] Guangwei Li, Yin Jian, Yunpeng Liu, and Zelin Shi. *Projective Registration with Manifold Optimization*. INTECH Open Access Publisher, 2009.

[95] Mats G Löfdahl. Evaluation of image-shift measurement algorithms for solar Shack-Hartmann wavefront sensors. *Astronomy & Astrophysics*, 524:A90, 2010.

[96] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110. URL: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94, doi:10.1023/B:VISI.0000029664.99615.94[43].

[97] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. URL: http://dl.acm.org/citation.cfm?id=1623264.1623280.

[98] L. Magri and A. Fusiello. T-linkage: A continuous relaxation of j-linkage for multimodel fitting. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3954–3961, June 2014. doi:10.1109/CVPR.2014.505[44].

[99] Luca Magri and Andrea Fusiello. Robust multiple model fitting with preference analysis and low-rank approximation. In Mark W. Jones Xianghua Xie and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 20.1–20.12. BMVA Press, September 2015. URL: https://dx.doi.org/10.5244/C.29.20, doi:10.5244/C.29.20[45].

[100] Raman Maini and Himanshu Aggarwal. A comprehensive review of image enhancement techniques. *CoRR*, abs/1003.4053, 2010. URL: http://arxiv.org/abs/1003.4053.

[101] JB Antoine Maintz and Max A Viergever. A survey of medical image registration. *Medical image analysis*, 2(1):1–36, 1998.

[41] http://dx.doi.org/10.1109/TGRS.2006.888937
[42] http://dx.doi.org/10.1109/CVPR.2011.5995309
[43] http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94
[44] http://dx.doi.org/10.1109/CVPR.2014.505
[45] http://dx.doi.org/10.5244/C.29.20

[102] Kirill Marinichev, Victor Gaganov, Anton Konushin, and Vezhnevets Vladimir. Gml ransac matlab toolbox. http://graphics.cs.msu.ru/en/science/research/machinelearning/ransactoolbox.

[103] Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. *Signal processing*, 87(10):2283–2302, 2007.

[104] J Matas and O Chum. Randomized RANSAC with Td,d test. *Image and Vision Computing*, 22(10):837–842, September 2004. URL: http://linkinghub.elsevier.com/retrieve/pii/S0262885604000514, doi:10.1016/j.imavis.2004.02.009[46].

[105] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 36.1–36.10. BMVA Press, 2002. doi:10.5244/C.16.36.

[106] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[107] A. Materne, O. Puig, and Ph. Kubik. Capteur d'images avec correction de bougé par recalage numérique de prises de vue fractionnées. Patent, 2012. FR 2976754-A1.

[108] V. Michau, J.-M. Conan, T. Fusco, M. Nicolle, C. Robert, M.-T. Velluet, and E. Piganeau. Shack-Hartmann wavefront sensing with extended sources. In *SPIE Optics+Photonics*, pages 63030B–63030B. SPIE, 2006.

[109] V. Michau, G. Rousset, and J. Fontanella. Wavefront sensing from extended sources. In *Real Time and Post Facto Solar Image Correction*, volume 1, page 124, 1993.

[110] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct 2005. doi:10.1109/TPAMI.2005.188[47].

[111] Lionel Moisan, Pierre Moulon, and Pascal Monasse. Automatic Homographic Registration of a Pair of Images, with A Contrario Elimination of Outliers. *Image Processing On Line*, 2:56–73, 2012. doi:10.5201/ipol.2012.mmm-oh[48].

[112] Jean Michel Morel and Guoshen Yu. Is sift scale invariant? *Inverse Problems and Imaging*, 5(1):115–136, 2011. doi:10.3934/ipi.2011.5.115[49].

[113] Rhonda M Morgan, Kirill V Shcheglov, Joseph J Green, Catherine M Ohara, Jennifer Roberts, and Erkin Sidick. Testbed for extended-scene Shack-Hartmann and phase retrieval wavefront sensing. In *Optics & Photonics 2005*, pages 59030I–59030I. International Society for Optics and Photonics, 2005.

[114] Mohammad Abdul Muquit, Takuma Shibahara, and Takafumi Aoki. A high-accuracy passive 3d measurement system using phase-based image matching. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E89-A(3):686–697, March 2006. URL: http://dx.doi.org/10.1093/ietfec/e89-a.3.686, doi:10.1093/ietfec/e89-a.3.686[50].

[46] http://dx.doi.org/10.1016/j.imavis.2004.02.009
[47] http://dx.doi.org/10.1109/TPAMI.2005.188
[48] http://dx.doi.org/10.5201/ipol.2012.mmm-oh
[49] http://dx.doi.org/10.3934/ipi.2011.5.115
[50] http://dx.doi.org/10.1093/ietfec/e89-a.3.686

[115] Darren R. Myatt, Philip H. S. Torr, Slawomir J. Nasuto, J. Mark Bishop, and R. Craddock. Napsac: High noise, high dimensional robust estimation - it's in the bag. In *BMVC*, 2002.

[116] L. Ng and V. Solo. Errors-in-variables modeling in optical flow estimation. *Image Processing, IEEE Transactions on*, 10(10):1528–1540, Oct 2001. doi:10.1109/83.951538[51].

[117] Kai Ni, Hailin Jin, and F. Dellaert. Groupsac: Efficient consensus in the presence of groupings. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2193–2200, Sept 2009. doi:10.1109/ICCV.2009.5459241[52].

[118] David Nistér. Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, December 2005. URL: http://link.springer.com/10.1007/s00138-005-0006-y, doi:10.1007/s00138-005-0006-y[53].

[119] T. Q. Pham, M. Bezuijen, L. J. Van Vliet, K. Schutte, and C. L. L. Hendriks. Performance of optimal registration estimators. In *Proc. SPIE*, volume 5817, pages 133–144, 2005. URL: http://dx.doi.org/10.1117/12.603304, doi:10.1117/12.603304[54].

[120] T. Q. Pham and P. Cox. Multi-hypothesis projection-based shift estimation for sweeping panorama reconstruction. In *2012 IEEE International Conference on Multimedia and Expo*, pages 97–102, July 2012. doi:10.1109/ICME.2012.38[55].

[121] T. T. Pham, T. J. Chin, J. Yu, and D. Suter. The random cluster model for robust geometric fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1658–1671, Aug 2014. doi:10.1109/TPAMI.2013.2296310[56].

[122] T.Q. Pham and M. Duggan. Bidirectional bias correction for gradient-based shift estimation. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 829–832, Oct 2008. doi:10.1109/ICIP.2008.4711883[57].

[123] Ben C Platt et al. History and principles of Shack-Hartmann wavefront sensing. *Journal of Refractive Surgery*, 17(5):S573–S577, 2001.

[124] Lisa A Poyneer. Scene-based Shack-Hartmann wave-front sensing: analysis and simulation. *Applied Optics*, 42(29):5807–5815, 2003.

[125] Lisa A. Poyneer, Donald T. Gavel, and James M. Brase. Fast wave-front reconstruction in large adaptive optics systems with use of the Fourier transform. *J. Opt. Soc. Am. A*, 19(10):2100–2111, Oct 2002. doi:10.1364/JOSAA.19.002100[58].

[126] Lisa A. Poyneer, David W. Palmer, Kai N. LaFortune, and Brian Bauman. Experimental results for correlation-based wavefront sensing. In Mark T. Gruneisen, John D. Gonglewski, and Michael K. Giles, editors, *Optics & Photonics 2005*, pages

[51] http://dx.doi.org/10.1109/83.951538
[52] http://dx.doi.org/10.1109/ICCV.2009.5459241
[53] http://dx.doi.org/10.1007/s00138-005-0006-y
[54] http://dx.doi.org/10.1117/12.603304
[55] http://dx.doi.org/10.1109/ICME.2012.38
[56] http://dx.doi.org/10.1109/TPAMI.2013.2296310
[57] http://dx.doi.org/10.1109/ICIP.2008.4711883
[58] http://dx.doi.org/10.1364/JOSAA.19.002100

58940N–58940N–142. International Society for Optics and Photonics, August 2005. doi:10.1117/12615231[59].

[127] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A Universal Framework for Random Sample Consensus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):2022–2038, August 2013. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6365642, doi:10.1109/TPAMI.2012.257[60].

[128] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. *Computer Vision–ECCV 2008*, pages 500–513, 2008. URL: http://www.springerlink.com/index/84T23624123Q1377.pdf.

[129] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. Exploiting uncertainty in random sample consensus. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2074–2081. IEEE, 2009. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5459456.

[130] M. Rais, C. Thiebaut, J. M. Delvit, and J. M. Morel. A tight multiframe registration problem with application to earth observation satellite design. In *2014 IEEE International Conference on Imaging Systems and Techniques (IST) Proceedings*, pages 6–10, Oct 2014. doi:10.1109/IST.2014.6958436[61].

[131] Martin Rais, J.-M. Morel, and Gabriele Facciolo. Iterative gradient-based shift estimation: To multiscale or not to multiscale? In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications - 20th Iberoamerican Congress, CIARP 2015, Montevideo, Uruguay, November 9-12, 2015, Proceedings*, pages 416–423, 2015. URL: http://dx.doi.org/10.1007/978-3-319-25751-8_50, doi:10.1007/978-3-319-25751-8_50[62].

[132] B.S. Reddy and B.N. Chatterji. An fft-based technique for translation, rotation, and scale-invariant image registration. *Image Processing, IEEE Transactions on*, 5(8):1266 –1271, aug 1996. doi:10.1109/83.506761[63].

[133] RA Reed. Comparison of subpixel phase correlation methods for image registration. Technical report, DTIC Document, 2010.

[134] J. Ren, J. Jiang, and T. Vlachos. High-accuracy sub-pixel motion estimation from noisy images in Fourier domain. *IEEE Transactions on Image Processing*, 19(5):1379–1384, May 2010. doi:10.1109/TIP.2009.2039056[64].

[135] Jinchang Ren, Theodore Vlachos, Yi Zhang, Jiangbin Zheng, and Jianmin Jiang. Gradient-based subspace phase correlation for fast and effective image alignment. *Journal of Visual Communication and Image Representation*, 25(7):1558 – 1565, 2014. URL: http://www.sciencedirect.com/science/article/pii/S1047320314001163, doi:10.1016/j.jvcir.2014.07.001[65].

[59]http://dx.doi.org/10.1117/12615231
[60]http://dx.doi.org/10.1109/TPAMI.2012.257
[61]http://dx.doi.org/10.1109/IST.2014.6958436
[62]http://dx.doi.org/10.1007/978-3-319-25751-8_50
[63]http://dx.doi.org/10.1109/83.506761
[64]http://dx.doi.org/10.1109/TIP.2009.2039056
[65]http://dx.doi.org/10.1016/j.jvcir.2014.07.001

[136] Ives Rey Otero and Mauricio Delbracio. Anatomy of the SIFT Method. *Image Processing On Line*, 4:370–396, 2014. doi:10.5201/ipol.2014.82[66].

[137] Clélia Robert, Jean-Marc Conan, Vincent Michau, Thierry Fusco, and Nicolas Vedrenne. Scintillation and phase anisoplanatism in Shack-Hartmann wavefront sensing. *JOSA A*, 23(3):613–624, 2006.

[138] D. Robinson and P. Milanfar. Efficiency and accuracy tradeoffs in using projections for motion estimation. In *Signals, Systems and Computers, 2001. Conference Record of the Thirty-Fifth Asilomar Conference on*, volume 1, pages 545–550 vol.1, Nov 2001. doi:10.1109/ACSSC.2001.986983[67].

[139] D. Robinson and P. Milanfar. Fundamental performance limits in image registration. *Image Processing, IEEE Transactions on*, 13(9):1185–1199, Sept 2004. doi:10.1109/TIP.2004.832923[68].

[140] Dirk Robinson and Peyman Milanfar. Bias minimizing filter design for gradient-based image registration. *Signal Processing: Image Communication*, 20(6):554 – 568, 2005. Special Issue on Advanced Aspects of Motion Estimation Special Issue on Advanced Aspects of Motion Estimation. URL: http://www.sciencedirect.com/science/article/pii/S0923596505000263, doi:10.1016/j.image.2005.03.010[69].

[141] M. Roopashree, A. Vyas, and B Raghavendra Prasad. Performance analysis of Fourier and Vector Matrix Multiply methods for phase reconstruction from slope measurements. *ArXiv e-prints*, November 2009. arXiv:0911.0813[70].

[142] B. Rouge, H. Vadon, and A. Giros. Fine stereoscopic image matching and dedicated instrument having a low stereoscopic coefficient, November 22 2011. US Patent 8,064,687. URL: http://www.google.ch/patents/US8064687.

[143] Peter J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984. URL: http://www.tandfonline.com/doi/abs/10.1080/01621459.1984.10477105, arXiv:http://www.tandfonline.com/doi/pdf/10.1080/01621459.1984.10477105[71], doi:10.1080/01621459.1984.10477105[72].

[144] Gerard Rousset. Wave-front sensors. *Adaptive optics in astronomy*, 1:91, 1999.

[145] N. Sabater, S. Leprince, and J. P. Avouac. Contrast invariant and affine sub-pixel optical flow. In *2012 19th IEEE International Conference on Image Processing*, pages 53–56, Sept 2012. doi:10.1109/ICIP.2012.6466793[73].

[146] N. Sabater, J.-M. Morel, and A. Almansa. How accurate can block matches be in stereo vision? *SIAM J. Img. Sci.*, 4(1):472–500, March 2011. URL: http://dx.doi.org/10.1137/100797849, doi:10.1137/100797849[74].

[66]http://dx.doi.org/10.5201/ipol.2014.82
[67]http://dx.doi.org/10.1109/ACSSC.2001.986983
[68]http://dx.doi.org/10.1109/TIP.2004.832923
[69]http://dx.doi.org/10.1016/j.image.2005.03.010
[70]http://arxiv.org/abs/0911.0813
[71]http://arxiv.org/abs/http://www.tandfonline.com/doi/pdf/10.1080/01621459.1984.10477105
[72]http://dx.doi.org/10.1080/01621459.1984.10477105
[73]http://dx.doi.org/10.1109/ICIP.2012.6466793
[74]http://dx.doi.org/10.1137/100797849

[147] Alan P. Schaum. Theory and design of local interpolators. *CVGIP: Graphical Model and Image Processing*, 55(6):464–481, 1993. URL: http://dblp.uni-trier.de/db/journals/cvgip/cvgip55.html#Schaum93.

[148] C. E. Shannon. Communication in the Presence of Noise. *Proceedings of the IRE*, 37(1):10–21, January 1949. URL: http://dx.doi.org/10.1109/jrproc.1949.232969, doi:10.1109/jrproc.1949.232969[75].

[149] H. Shekarforoush, M. Berthod, and J. Zerubia. Subpixel image registration by estimating the polyphase decomposition of cross power spectrum. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pages 532–537, Jun 1996. doi:10.1109/CVPR.1996.517123[76].

[150] Shinsuke Shimojo, Gerald H. Silverman, and Ken Nakayama. Occlusion and the solution to the aperture problem for motion. *Vision Research*, 29(5):619 – 626, 1989. URL: http://www.sciencedirect.com/science/article/pii/0042698989900473, doi:10.1016/0042-6989(89)90047-3[77].

[151] E. Sidick et al. An Adaptive cross-correlation algorithm for extended scene Shack-Hartmann wavefront sensing. In *Adaptive Optics: Methods, Analysis and Applications*. OSA, 2007.

[152] E. Sidick, R. Morgan, J. Green, C. Ohara, and D. Redding. Adaptive Cross-Correlation Algorithm and Experiment of Extended Scene Shack-Hartmann Wavefront Sensing. In Howard A. MacEwen and James B. Breckinridge, editors, *UV/Optical/IR Space Telescopes: Innovative Technologies and Concepts III*, volume 6687, pages 668710–668711, September 2007. doi:10.1117/12.735030[78].

[153] Erkin Sidick. Adaptive periodic-correlation algorithm for extended scene Shack-Hartmann wavefront sensing. In *Imaging and Applied Optics*, page CPDP1. Optical Society of America, 2011. doi:10.1364/COSI.2011.CPDP1[79].

[154] E.P. Simoncelli. Design of multi-dimensional derivative filters. In *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, volume 1, pages 790–794 vol.1, Nov 1994. doi:10.1109/ICIP.1994.413423[80].

[155] Julius O. Smith. *Spectral Audio Signal Processing*. http:[81]//ccrma.stanford.edu/-~jos/sasp/. online book, 2011 edition.

[156] R. Soummer, L. Pueyo, A. Sivaramakrishnan, and R. J. Vanderbei. Fast computation of lyot-style coronagraph propagation. *Opt. Express*, 15(24):15935–15951, Nov 2007. URL: http://www.opticsexpress.org/abstract.cfm?URI=oe-15-24-15935, doi:10.1364/OE.15.015935[82].

[157] William H Southwell. Wave-front estimation from wave-front slope measurements. *JOSA*, 70(8):998–1006, 1980.

[158] J.-L. Starck, F. Murtagh, and A. Bijaoui. *Image Processing and Data Analysis: The Multiscale Approach*. Cambridge University Press, New York, NY, USA, 1998.

[75] http://dx.doi.org/10.1109/jrproc.1949.232969
[76] http://dx.doi.org/10.1109/CVPR.1996.517123
[77] http://dx.doi.org/10.1016/0042-6989(89)90047-3
[78] http://dx.doi.org/10.1117/12.735030
[79] http://dx.doi.org/10.1364/COSI.2011.CPDP1
[80] http://dx.doi.org/10.1109/ICIP.1994.413423
[81] http://ccrma.stanford.edu/~jos/sasp/
[82] http://dx.doi.org/10.1364/OE.15.015935

[159] H.S. Stone, M.T. Orchard, E.-C. Chang, and S.A. Martucci. A fast direct fourier-based algorithm for subpixel registration of images. *Geoscience and Remote Sensing, IEEE Transactions on*, 39(10):2235–2243, Oct 2001. doi:10.1109/36.957286[83].

[160] I. E. Sutherland. Three-dimensional data input by tablet. *Proceedings of the IEEE*, 62(4):453–461, April 1974. doi:10.1109/PROC.1974.9449[84].

[161] Richard Szeliski and Daniel Scharstein. Sampling the disparity space image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):419–425, 2004.

[162] Kenji Takita, Takafumi Aoki, Yoshifumi Sasaki, Tatsuo Higuchi, and Koji Kobayashi. High-Accuracy Subpixel Image Registration Based on Phase-Only Correlation. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 86(8):1925–1934, 2003.

[163] T. Tarbell and R. Smithson. A Simple Image Motion Compensation System for Solar Observations. *Solar instrumentation: What's next? Proc. of a Conf.*, 1981.

[164] R. B. Tennakoon, A. Bab-Hadiashar, Z. Cao, R. Hoseinnezhad, and D. Suter. Robust model fitting using higher than minimal subset sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):350–362, Feb 2016. doi:10.1109/TPAMI.2015.2448103[85].

[165] Mariano Tepper and Guillermo Sapiro. A biclustering framework for consensus problems. *SIAM Journal on Imaging Sciences*, 7(4):2488–2525, 2014. URL: http://dx.doi.org/10.1137/140967325, arXiv:http://dx.doi.org/10.1137/140967325[86], doi:10.1137/140967325[87].

[166] P. Thévenaz, T. Blu, and M. Unser. Image interpolation and resampling. In I.N. Bankman, editor, *Handbook of Medical Imaging, Processing and Analysis*, chapter 25, pages 393–420. Academic Press, San Diego CA, USA, 2000.

[167] P. Thevenaz, U. E. Ruttimann, and M. Unser. A pyramid approach to subpixel registration based on intensity. *IEEE Transactions on Image Processing*, 7(1):27–41, Jan 1998. doi:10.1109/83.650848[88].

[168] Qi Tian and Michael N Huhns. Algorithms for subpixel registration. *Comput. Vision Graph. Image Process.*, 35(2):220–233, August 1986. URL: http://dx.doi.org/10.1016/0734-189X(86)90028-9, doi:10.1016/0734-189X(86)90028-9[89].

[169] Roberto Toldo and Andrea Fusiello. *Robust Multiple Structures Estimation with J-Linkage*, pages 537–547. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. URL: http://dx.doi.org/10.1007/978-3-540-88682-2_41, doi:10.1007/978-3-540-88682-2_41[90].

[170] Xiaohua Tong, Zhen Ye, Yusheng Xu, Shijie Liu, Lingyun Li, Huan Xie, and Tianpeng Li. A novel subpixel phase correlation method using singular value decomposition and unified random sample consensus. *IEEE Transactions on Geoscience and Remote Sensing*, 53(8):4143–4156, 2015.

[83] http://dx.doi.org/10.1109/36.957286
[84] http://dx.doi.org/10.1109/PROC.1974.9449
[85] http://dx.doi.org/10.1109/TPAMI.2015.2448103
[86] http://arxiv.org/abs/http://dx.doi.org/10.1137/140967325
[87] http://dx.doi.org/10.1137/140967325
[88] http://dx.doi.org/10.1109/83.650848
[89] http://dx.doi.org/10.1016/0734-189X(86)90028-9
[90] http://dx.doi.org/10.1007/978-3-540-88682-2_41

[171] Ben Tordoff and David W. Murray. Guided sampling and consensus for motion estimation. In *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, pages 82–98, London, UK, UK, 2002. Springer-Verlag. URL: http://dl.acm.org/citation.cfm?id=645315.649190.

[172] P. Torr and A. Zisserman. Robust computation and parametrization of multiple view relations. In *Computer Vision, 1998. Sixth International Conference on*, pages 727–732, Jan 1998. doi:10.1109/ICCV.1998.710798[91].

[173] P.H.S. Torr and A. Zisserman. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, 78(1):138–156, April 2000. URL: http://linkinghub.elsevier.com/retrieve/pii/S1077314299908329, doi:10.1006/cviu.1999.0832[92].

[174] R. Tron and K. Daniilidis. On the quotient representation for the essential manifold. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1574–1581, June 2014. doi:10.1109/CVPR.2014.204[93].

[175] Chun-Jen Tsai, N.P. Galatsanos, and AK. Katsaggelos. Total least squares estimation of stereo optical flow. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, volume 2, pages 622–626 vol.2, Oct 1998. doi:10.1109/ICIP.1998.723549[94].

[176] Robert Tyson. *Principles of adaptive optics*. CRC press, 2010.

[177] Robert K Tyson. Bit-error rate for free-space adaptive optics laser communications. *JOSA A*, 19(4):753–758, 2002.

[178] G. Tzimiropoulos and V. Argyriou. Gradient schemes for robust fft-based motion estimation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1781–1785, March 2016. doi:10.1109/ICASSP.2016.7471983[95].

[179] G. Tzimiropoulos, V. Argyriou, and T. Stathaki. Subpixel registration with gradient correlation. *IEEE Transactions on Image Processing*, 20(6):1761–1767, June 2011. doi:10.1109/TIP.2010.2095867[96].

[180] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. SIAM, Philadelphia, 1991.

[181] Sabine Van Huffel and Hongyuan Zha. An efficient total least squares algorithm based on a rank-revealing two-sided orthogonal decomposition. *Numerical Algorithms*, 4(1):101–133, 1993. URL: http://dx.doi.org/10.1007/BF02142742, doi:10.1007/BF02142742[97].

[182] Nicolas Védrenne, Vincent Michau, Clélia Robert, and Jean-Marc Conan. Shack-Hartmann wavefront estimation with extended sources: anisoplanatism influence. *Journal of the Optical Society of America A*, 24(9):2980, September 2007. doi:10.1364/JOSAA.24.002980[98].

[91] http://dx.doi.org/10.1109/ICCV.1998.710798
[92] http://dx.doi.org/10.1006/cviu.1999.0832
[93] http://dx.doi.org/10.1109/CVPR.2014.204
[94] http://dx.doi.org/10.1109/ICIP.1998.723549
[95] http://dx.doi.org/10.1109/ICASSP.2016.7471983
[96] http://dx.doi.org/10.1109/TIP.2010.2095867
[97] http://dx.doi.org/10.1007/BF02142742
[98] http://dx.doi.org/10.1364/JOSAA.24.002980

[183] A. Wald. *Sequential Analysis*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. J. Wiley & Sons, Incorporated, 1947. URL: https://books.google.es/books?id=0nREAAAAIAAJ.

[184] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 13(4):600–612, 2004.

[185] Joseph Weber and Jitendra Malik. Robust computation of optic flow in a multiscale differential framework. *International Journal of Computer Vision*, pages 67–81, 1995.

[186] Joachim Weickert, Andrés Bruhn, Thomas Brox, and Nils Papenberg. *A Survey on Variational Optic Flow Methods for Small Displacements*, pages 103–136. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. URL: http://dx.doi.org/10.1007/978-3-540-34767-5_5, doi:10.1007/978-3-540-34767-5_5[99].

[187] E. Weiszfeld and Frank Plastria. On the point for which the sum of the distances to n given points is minimum. *Annals of Operations Research*, 167(1):7–41, 2008. URL: http://dx.doi.org/10.1007/s10479-008-0352-z, doi:10.1007/s10479-008-0352-z[100].

[188] A. Wirth. Ophthalmic instrument having Hartmann wavefront sensor with extended source, July 22 2003. US Patent 6,595,642.

[189] P Wizinowich et al. First light adaptive optics images from the Keck II telescope: a new era of high angular resolution imagery. *Publications of the Astronomical Society of the Pacific*, 112(769), 2000.

[190] Zhengyou Zhang, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(12):87 – 119, 1995. Special Volume on Computer Vision. URL: http://www.sciencedirect.com/science/article/pii/0004370295000224, doi:10.1016/0004-3702(95)00022-4[101].

[191] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21:977–1000, 2003.

[192] M. Zuliani. Ransac toolbox for matlab, Nov. 2008. URL: https://github.com/RANSAC/RANSAC-Toolbox.

[99] http://dx.doi.org/10.1007/978-3-540-34767-5_5
[100] http://dx.doi.org/10.1007/s10479-008-0352-z
[101] http://dx.doi.org/10.1016/0004-3702(95)00022-4