Universitat de les Illes Balears

TRABAJO DE FIN DE MÁSTER

EVALUACIÓN EXPERIMENTAL DE DIVERSOS ALGORITMOS DE SUPERPÍXELES PARA EL RECONOCIMIENTO VISUAL DE ESCENAS

Juan José Novo Arbona

Máster Universitario en Ingeniería Industrial

Centro de Estudios de Postgrado

Año Académico 2020-21

EVALUACIÓN EXPERIMENTAL DE DIVERSOS ALGORITMOS DE SUPERPÍXELES PARA EL RECONOCIMIENTO VISUAL DE ESCENAS

Juan José Novo Arbona

Trabajo de Fin de Máster

Centro de Estudios de Postgrado

Universidad de las Illes Balears

Año Académico 2020-21

Palabras clave del trabajo:

segmentación, superpíxeles

Tutor 1: Emilio García Fidalgo

Tutor 2: Alberto Ortiz Rodríguez

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ACRÓNIMOS

SLAM Simultaneous Localization and Mapping

SIFT Scale Invariant Feature Transform

SURF Speeded Up Robust Features

ORB Oriented FAST and Rotated BRIEF

BRIEF Binary Robust Independent Elementary Features

EAMS Edge-Aumented Mean Shift

POISE Proposals for Objects from Improved Seeds and Energies

CIS Constant Intensity Superpixels

SLIC Simple Linear Clustering

LSC Linear Spectral Clustering

SEEDS Superpixel Extracted via Energy-Driven Sampling

PF Path Finder

TPS Topological Preserving Superpixels

TP Turbopixels

MSLIC Manifold SLIC

SLICO Simple Linear Clustering 0 parameter

CNN Convolutional Neural Networks

GLCM Gray Level Co-Ocurrence Matrix

IoU Intersection over Union

DoG Difference of Gaussians

BRISK Binary Robust Invariant Scalable Keypoints

FAST Features from Accelerated Segment Test

FREAK Fast Retina Keypoint

ORB Oriented FAST and Rotated BRIEF

LBP Local Binary Pattern

ResNet Residual Networks

AuC Area under the Curve

RESUMEN

El reconocimiento de escenas previamente visitadas es una tarea compleja y ampliamente abordada, con múltiples aplicaciones en labores diversas, muy especialmente en robótica. Aunque resulta algo cotidiano y sencillo en la mayoría de ocasiones para un ser humano, supone un reto computacional que requiere de la captura de un conjunto de datos del entorno, la extracción de las características más relevantes de esa información y su descripción, de manera que podamos comparar esa definición de la escena con la información previamente capturada y así poder detectar si hemos vuelto al mismo emplazamiento. Por si fuera poco, debemos considerar que puede haber modificaciones en la percepción del entorno, ya sea por cambios en la perspectiva del observador o en los propios elementos que componen la escena. Cuando estos datos del entorno se toman en forma de imágenes, hablamos de reconocimiento visual de escenas, tema sobre el que trata este proyecto. Tal y como se ha comentado, existen muchas aproximaciones a esta problemática en lo referente a como abordar la descripción de la imagen y la extracción de sus características. Algunas de ellas se basan en descriptores holísticos que aglutinan la información de la imagen en un solo vector, mientras que otras identifican los puntos más relevantes de la escena (bordes, esquinas, etc.), los definen y componen una descripción de la escena basada en esos puntos. Esta última estrategia tiene una alta precisión, a costa de una mayor exigencia computacional. Este trabajo afronta la problemática mediante una solución intermedia, conocida como superpíxeles, que dividen la escena en regiones mediante algoritmos de segmentación, tratando de mejorar así el rendimiento de los enfoques tradicionales. Para ello, se evalúan diferentes algoritmos de superpíxeles públicos según unas métricas específicas con vistas a seleccionar aquellos que se ajusten mejor a la aplicación que nos ocupa. Posteriormente, con las técnicas escogidas, se consideran diversas herramientas de descripción basadas en características locales, así como recursos más avanzados como puedan ser descriptores producidos por redes neuronales convolucionales. Por último, los elementos escogidos conforman una serie de estrategias que procederán a evaluarse frente a secuencias de imágenes que comprenden cierres de bucle, para determinar la capacidad de reconocimiento de escenas previamente visitadas. De cada uno de los procesos se exponen un conjunto de conclusiones basadas en los resultados obtenidos, así como del recorrido experimental completo que recoge este trabajo.

CAPÍTULO 1

INTRODUCCIÓN

En este capítulo se presenta la temática del proyecto y la materia que se va a abarcar en el presente documento, así como los objetivos y estructura del mismo.

1.1 Robótica

Las primeras manifestaciones de elementos relativos a la robótica nacen en la antigua Grecia como fruto de la fascinación del hombre por el desarrollo de mecanismos capaces de imitar la figura y movimientos de aquello que observaban en la naturaleza [6]. Los conocimientos sobre estos artilugios (denominados *automatos*) se extendieron alrededor del mundo dando lugar a dispositivos con aplicaciones prácticas más allá del entretenimiento.

En la actualidad, el concepto asociado a la robótica es el de un campo que engloba un amplio abanico de disciplinas propias de múltiples ramas de la ingeniería (mecánica, informática, electrónica, etc.) enfocadas al desempeño de tareas realizadas por robots. La robótica se ha introducido en nuestras vidas, revolucionando todos los sectores económicos y cambiando nuestra forma de trabajar y de llevar a cabo muchas de nuestras tareas diarias. Dentro del desempeño de estas tareas, la robótica incide en cada una de las fases de la ejecución, ya sea a alto nivel en el caso del diseño, construcción y aplicación, como en capas menos visibles, como sería el control, en el cuál se incluye la percepción sensorial y el procesamiento de la información.

En el caso de los robots móviles, pueden clasificarse en cuatro grandes categorías en función del campo de aplicación: terrestres, acuáticos, aéreos o espaciales [7]. En cada una de estas categorías podemos encontrar robots operados por humanos, pero existen circunstancias en los que se precisa que éstos sean autónomos. Se entiende por robot autónomo un agente capaz de adoptar comportamientos o realizar tareas con un alto grado de independencia, pudiendo desempeñar su cometido sin intervención humana. Esta funcionalidad es especialmente interesante en campos de investigación espacial o exploración en entornos peligrosos, como rescates que podrían poner en riesgo la vida

de las personas. No obstante, también tiene cabida en tareas más cotidianas, como la gestión de tareas de limpieza en entornos urbanos, como sería el caso del robot Trombia, presentado en la Fig. 1.1.

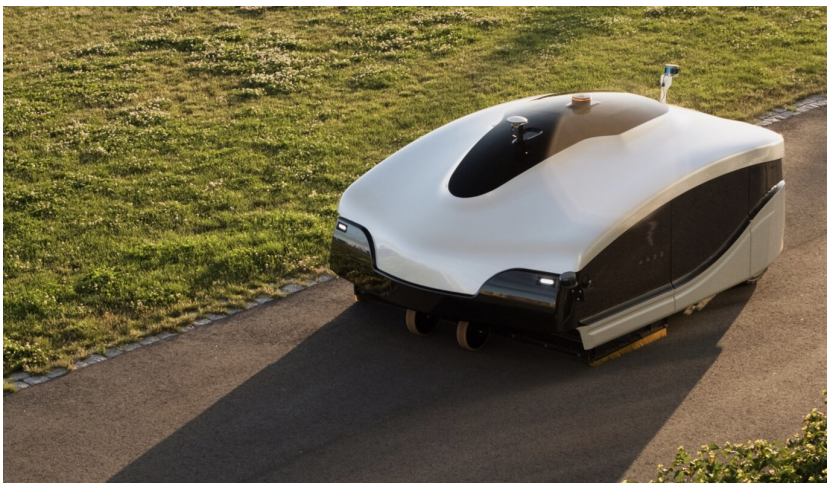


Figura 1.1: Imagen del vehículo autónomo de limpieza urbana Trombia [1].

1.2 Navegación

Uno de los principales retos que se deben afrontar para que una plataforma móvil pueda desempeñar labores de forma autónoma es dotarla de la capacidad para percibir información del entorno y procesarla de la forma adecuada. De este modo, se pretende que disponga de la capacidad de alcanzar un objetivo determinado evitando los posibles obstáculos que puedan presentarse en la trayectoria definida. La navegación es la tarea que se encarga de este cometido, abarcando la misión de establecer una trayectoria óptima entre una posición inicial y un objetivo, ciñéndose a las limitaciones de la plataforma. Para ello combina diversos procesos, como podrían ser: la autolocalización (*localization*), consistente en identificar la ubicación y posición del robot respecto al entorno; la planificación de la trayectoria (*path planning*), que identifica el mejor camino para alcanzar el objetivo; y la construcción de un mapa del entorno en base a la información captada por el equipamiento sensorial del que pueda disponer la plataforma. En cada uno de estos procesos, los mapas adquieren gran relevancia, dado que suponen el punto de partida para el desarrollo de otras tareas.

1.3 Construcción de mapas, Localización y SLAM

En función de la naturaleza de la misión para la que dispongamos el robot, el mapa del entorno puede ser cargado previamente. No obstante, esa opción no siempre es posible y es el propio robot el que debe construir la representación de su entorno. Este proceso recibe el nombre de construcción de mapas o, en inglés, *mapping*. Estas representaciones suelen tomar la forma de mapas métricos o topológicos. La diferencia entre ambos radica en que, mientras que los mapas métricos ubican la posición del

robot respecto a su entorno tomando como referencia un sistema de coordenadas métricas global, los topológicos representan el entorno de manera abstracta en forma de grafo [8]. Por otro lado, existen también soluciones híbridas que tratan de potenciar las ventajas de ambas estrategias.

Las tareas de *mapping* y localización presentan una estrecha relación, dándonos, por un lado, una representación del entorno y, por el otro, la posición del robot dentro de esa representación en base a la información sensorial obtenida. Ambas tareas son fundamentales en vehículos autónomos, pensados para desplazarse sin asistencia humana. La combinación de estas tareas ejecutadas en un entorno desconocido presentan una problemática dado que, necesitamos conocer la posición de los obstáculos para construir el mapa, y a su vez, durante la localización, se obtiene la ubicación y orientación de la plataforma en base a un mapa conocido. Para resolver este conflicto entre ambas tareas, existen técnicas que generan un mapa incremental del entorno a la vez que ubican la plataforma dentro de dicho mapa. Estas técnicas se conocen en inglés como Simultaneous Localization and Mapping (SLAM).

1.4 Detección Visual de Bucles

Este proceso de generar un mapa del entorno y ubicar en él la propia plataforma es susceptible de introducir imprecisiones propias de los sensores mediante los cuales el robot percibe su entorno. Estas imprecisiones y ruido, se traducen a la larga en una deriva que resulta en una representación deformada del escenario y, por ello, de la ubicación del robot [9]. Esta problemática se puede abordar mediante procesos de optimización. En dichos procesos suele ser fundamental detectar cuándo se ha vuelto a una zona en la que ya se ha estado previamente. Esto es lo que se conoce como detección de bucles (*Loop Closure Detection*, en inglés) o *Appearance-based Loop Closure* en los casos en los que se utiliza una cámara como sensor para identificar la escena. En general, cuando se desvincula esta tarea de la navegación de una plataforma móvil, como es el caso de este proyecto, se habla de *Visual Place Recognition*. Este tipo de tareas albergan una notable dificultad, que radica en problemáticas como la *escalabilidad*, generada por el incremento de la colección de imágenes del recorrido que debemos comparar con la escena actual así como avanzamos en el flujo de imágenes de la secuencia. Otro de los problemas más relevantes se conoce como *perceptual aliasing*, que sucede al interpretar escenas diferentes como la misma escena. La naturaleza de este problema está estrechamente ligada a la calidad del descriptor visual utilizado. Finalmente, cabe destacar también el reto que suponen los entornos dinámicos, en los cuales podemos malinterpretar dos capturas de la misma escena debido a objetos en movimiento, cambios de puntos de vista o variaciones en la iluminación.

1.5 Superpíxeles

El rendimiento de un algoritmo de detección de bucles basado en apariencia se fundamenta en dos puntos: por un lado, en la forma como se describen las imágenes, y, por otro lado, en la técnica utilizada para obtener de una forma eficiente imágenes similares previas. Respecto al primer punto, existen tradicionalmente dos formas de describir imágenes: extrayendo una serie de puntos de interés (*local feature descrip-*

tors) o extrayendo un descriptor global de la imagen (*global descriptors*). Una solución intermedia que consiste en segmentar la imagen usando un algoritmo de superpíxeles y utilizarlos para indexar la imagen. Esta es la estrategia que adoptamos y evaluamos en este Trabajo Final de Máster.

Según Barrientos et al. [6], la segmentación es la técnica mediante la cual se subdivide una imagen en las regiones u objetos que la constituyen. En función de la información que se quiera obtener, la segmentación será más o menos profunda. En el caso de los algoritmos de superpíxeles, la imagen se segmenta agrupando aquellos píxeles que presentan características visuales similares, adaptándose estos conjuntos, en la medida de lo posible, a los contornos de la escena, en función del criterio de agrupación empleado. Esta agrupación mejora sustancialmente el proceso posterior al reducir una imagen a un pequeño conjunto de segmentos, en contraste con los millones de píxeles originales de la imagen o los miles de puntos que resultan de las estrategias basadas en descriptores locales.

1.6 Objetivos del proyecto

Dentro de este contexto, el objetivo del trabajo es evaluar la utilidad de diversos algoritmos de superpíxeles en tareas de reconocimiento visual de escenas. De forma más precisa, los objetivos del trabajo son:

- Estudiar y comprender el funcionamiento de algunos algoritmos públicos de segmentación de imágenes basados en superpíxeles.
- Evaluar estos algoritmos de una forma clara y objetiva, de cara a poder decidir cuál de ellos es el más útil en tareas de reconocimiento visual de escenas.
- Evaluar algunos mecanismos de descripción de superpíxeles para reconocimiento visual de escenas.
- Plantear una estrategia de reconocimiento visual de escenas basada en superpíxeles de acuerdo con los resultados previos.

1.7 Organización del documento

El presente documento se organiza de la siguiente forma:

- En el capítulo 2 se desarrollan los fundamentos teóricos sobre los que se sustenta el trabajo planteado.
- En los capítulos 3 y 4 se exponen y evalúan los algoritmos y métodos considerados en este trabajo.
- Una vez definidas las opciones que mejor se ajustan a los requisitos del proyecto, en el capítulo 5 se describen las combinaciones valoradas para desarrollar un sistema de reconocimiento visual de escenas.
- Finalmente, en el capítulo 6 se detallan las conclusiones derivadas del trabajo realizado en este proyecto y se presentan posibles vías de trabajo futuro.

CAPÍTULO 2

FUNDAMENTOS DE DESCRIPCIÓN Y RECONOCIMIENTO DE ESCENAS

En este capítulo se introducen los conceptos teóricos propios de la estrategia de reconocimiento de escenas desarrollada en este proyecto.

2.1 Reconocimiento de Escenas

El reconocimiento de escenas en el campo de la robótica supone una tarea extremadamente compleja, a pesar de haber sido extensamente abordada desde una gran variedad de enfoques [8]. Más allá de la propia escena que se visualiza, se debe disponer de una precisa representación del entorno con el que comparar la imagen y, tras ello, derivar una certeza acerca de si la escena que se presencia corresponde a una vista anteriormente. La ejecución de esta tarea debe ser susceptible a cambios en la apariencia del entorno que puedan darse en una misma escena. Estos cambios de apariencia pueden surgir debido a cambios de enfoque, rotación de la escena, alteraciones en la iluminación o en los propios contornos de los objetos que componen la escena debidos, por ejemplo, a fenómenos meteorológicos. Para tratar con estos cambios de apariencia, se requiere un mecanismo de descripción de imágenes que los tolere. Debido a su importancia, los sistemas de reconocimiento visual de escenas se pueden clasificar de acuerdo al método usado para describir las imágenes.

2.1.1 Métodos basados en descriptores globales

Estas técnicas, computan un único vector a partir de la información contenida en la imagen completa [10]. En sus primeras implementaciones consistían en histogramas de color y descriptores basados en análisis de componentes principales. Las principales ventajas que presentan este tipo de algoritmos es su rapidez de computación, facilitando el proceso de emparejamiento de imágenes y reduciendo la exigencia computacional a los equipos dispuestos para ejecutar la tarea [9]. No obstante, presentan notables

flaquezas en presencia de cambios de enfoque en las escenas, en la iluminación o frente oclusiones parciales.

Entre los métodos más usados en reconocimiento visual de escenas, de esta categoría destaca GIST [11], el cual se genera aplicando filtros de Gabor en diferentes orientaciones y frecuencias para extraer la información de la imagen. La media del resultado de estas operaciones nos aporta un vector que define la imagen.

2.1.2 Métodos basados en descriptores locales

Frente a las debilidades comentadas acerca de los descriptores globales, encontramos los mecanismos de descripción basados en características locales. Estos llevan a cabo un procedimiento constituido principalmente por dos pasos. En primer lugar, se produce una fase de extracción, en la cual se identifican puntos significativos de la imagen (*keypoints*), los cuales presentan unas características particulares dentro de una región determinada. Posteriormente, se realiza la descripción, durante la cual se toman ciertas mediciones dentro de la vecindad de cada punto para generar un descriptor local. Dichos descriptores habitualmente son vectores de componentes reales, aunque existen algoritmos que generan vectores binarios, con las consecuentes ventajas computacionales que ello supone.

Dado que el objetivo en el uso de estas técnicas es identificar una misma escena en diferentes instantes de una secuencia, una característica muy importante que deben presentar los descriptores es la repetibilidad [12], es decir, debe aportarnos ciertas garantías de que dadas dos escenas iguales, se identificarán los mismos puntos significativos, independientemente de los cambios que pueda sufrir la escena. Del mismo modo, otros parámetros relevantes que debemos considerar del algoritmo es que nos proporcione descriptores claramente diferenciados y localizados, la cantidad de puntos significativos detectados, así como su precisión y eficiencia computacional.

Algunos ejemplos son SIFT [13] o Speeded Up Robust Features (SURF) [14], ambos detectores y descriptores. Por otro lado, tendríamos otras alternativas como BRIEF [15] u ORB [16] , que, a diferencia de los mencionados anteriormente, no generan descriptores de punto flotante, si no que generan vectores binarios, de modo que mejoran considerablemente las exigencias computacionales de la plataforma y reducen los tiempos de ejecución.

2.2 Superpíxeles

Además de los métodos expuestos hasta el momento, existe una alternativa que plantea un enfoque intermedio, basada en técnicas que dividen la imagen en diferentes segmentos, denominados superpíxeles.

Los superpíxeles agrupan píxeles en base a ciertos criterios de similitud para crear entidades visualmente significativas y, a su vez, reducir el número de datos de partida de cara a pasos posteriores en el procesamiento de la imagen. En la Fig. 2.1 se muestran unos ejemplos de segmentación obtenidos mediante el algoritmo SLIC aplicando diferentes parámetros. Con ello, se mejora el proceso en términos de eficiencia computacional, ya que aglutinan información de un conjunto de píxeles simplificando el proceso. Además, define aquellas características que generan que un píxel se asemeje a los que se ubican en una determinada vecindad. Dichos algoritmos típicamente

funcionan asociando un grafo a la imagen, cuyas conexiones se establecen basándose en unos criterios que dependen de ese algoritmo en particular o de los parámetros establecidos.

Figura 2.1: Ejemplo de segmentación empleando algoritmo de superpixeles [2]

Según expone Stutz et al. [17], la principal diferencia entre este tipo de algoritmos y los algoritmos de sobresegmentación radica en que, en el caso de los superpíxeles, podemos controlar el número de segmentos que se van a generar. Además, dichos algoritmos identifican cada segmento de forma única mediante una etiqueta, la cual se asigna a cada uno de los píxeles que los conforman. Además de la conectividad de estos píxeles comentada anteriormente, los conjuntos de píxeles deben respetar los bordes de los elementos que componen la imagen. En los casos en los que no haya elementos singulares, por ejemplo, un fragmento de cielo despejado, los segmentos tienden a ser compactos, distribuidos de forma regular y presentar bordes suaves.

Si se analiza la estrategia a alto nivel, podemos clasificar los algoritmos de superpíxeles en diferentes grupos. Entre los más destacados, tendríamos aquellos algoritmos basados en la transformación Watershed y sus variantes, típicamente basados en la gradiente de la imagen. Por otro lado, los algoritmos basados en densidad, como Edge-Aumented Mean Shift (EAMS) [18] o Quick Shift [19], ejecutan una búsqueda de tendencias en los parámetros de píxeles colindantes para formar los conjuntos. Algoritmos como Proposals for Objects from Improved Seeds and Energies (POISE) [20] y Constant Intensity Superpixels (CIS) [21] conforman el grupo de algoritmos basados en grafos, los cuales mediante comparaciones de color generan una serie de bordes que dividen la imagen. SLIC (Simple Linear Clustering) [22] o LSC (Linear Spectral Clustering) [23] forman parte de un grupo de algoritmos basados en conjuntos, partiendo de unos píxeles de partida y relacionándolos con su entorno en base a parámetros como podrían ser el color, distancia y profundidad. La estrategia adoptada por los algoritmos de optimización de energía, como SEEDS (Superpixel Extracted via Energy-Driven Sampling) [24], consiste en la división de la imagen en una cuadrícula regular, a

partir de la cual se va optimizando la afinidad entre los píxeles que conforman cada conjunto de forma iterativa. Existen también algoritmos en los cuales la segmentación se lleva a cabo a partir de la conexión de unos píxeles iniciales en función de unos criterios predefinidos, como serían Path Finder (PF) [25] o Topological Preserving Superpixels (TPS) [26]. Por último, existen algoritmos basados en evolución de contornos como Turbopixels (TP) [27], que definen unos píxeles iniciales y realizan una evolución del contorno, englobando píxeles colindantes y formando así cada segmento. Cabe destacar el trabajo de El Euch et al. [28], en el cual se combinan técnicas de *bag-of-words* y superpíxeles para reconocimiento visual de escenas.

En este trabajo, se ha experimentado con SLIC y dos de sus variantes (SLICO y MSLIC), LSC y SEEDS. Stutz et al [17] exponen una exhaustiva comparativa para un total de 28 algoritmos, obteniendo resultados muy favorables para los que se han considerado en la tarea que ocupa este proyecto, destacando el rendimiento de SEEDS, que ocupa la segunda posición de la clasificación.

2.3 Redes neuronales convolucionales

Actualmente, es ya extendido el uso de redes neuronales convolucionales (o en inglés Convolutional Neural Networks (CNN)) en el reconocimiento de escenas. Su funcionamiento consiste en un entramado de capas que extraen características específicas de las imágenes de entrada. En cada una de estas capas se suceden una serie de operaciones que sintetizan elementos concretos de la imagen a través de convoluciones, funciones de activación y, finalmente, *pooling* [29]. Las convoluciones consisten en aplicar una serie de filtros con pesos entrenados de forma automática para una aplicación concreta. Las funciones de activación permiten aproximar tendencias aleatorias complejas, identificando patrones no lineales. En el entramado de capas que conforman una red, podemos detectar rasgos más generales en las primeras capas y características mucho más complejas en sus últimas capas.

Existen múltiples arquitecturas de CNN, las cuales difieren en el número de capas y filtros por capa. En este trabajo se han empleado ResNet50 [30] y *AlexNet* [31]. Concretamente, esta última está compuesta por ocho capas, de las cuales las cinco primeras son convolucionales, útiles para extraer características y las tres últimas son de clasificación (denominadas *fully connected layers*).

Inicialmente, el uso de las CNN en el reconocimiento de escenas se centró en la extracción de un descriptor global de la imagen completa, obtenido de una capa particular de la red. Los resultados demostraron que, empleando las capas inferiores de la red, conseguían una respuesta robusta frente cambios provocados por fenómenos ambientales aunque con una importante sensibilidad de cara a rotaciones o cambios en la perspectiva [8]. Sünderhauf et al. [32] plantean una evaluación de redes convolucionales para reconocimiento de escenas. Del mismo modo, otros trabajos han planteado propuestas que combinan algunos de estos sistemas de detección con descriptores basados en CNN [33]. Posteriormente, otros autores han propuesto la confluencia de información de diversas CNN para llevar a cabo localización topológica [34].

CAPÍTULO 3

EVALUACIÓN DE ALGORITMOS DE SUPERPÍXELES

En este capítulo, se describen los algoritmos de segmentación seleccionados para este trabajo y el proceso de evaluación llevado a cabo para elegir aquellos con un mejor rendimiento para el reconocimiento de escenas.

3.1 Descripción de los algoritmos

3.1.1 SLIC y variante SLICO

La implementación original de SLIC [2], realiza la segmentación basándose en la proximidad de los píxeles en el plano imagen y en su color en el espacio CIELAB. Se define un número deseado de conjuntos, o superpíxeles, K. Inicialmente, éstos se distribuyen de forma regular y tienen el mismo tamaño. Suponiendo una imagen con N píxeles, el tamaño de cada uno de estos conjuntos deberá ser N/K y, por ello, la distancia al centro de cada conjunto será $S = \sqrt{N/K}$. De esta forma, todos los píxeles que formen parte de dicho conjunto deben encontrarse en una área de $2S \times 2S$ píxeles alrededor de su centro. El cálculo de distancia espacial y de color se lleva a cabo según el desarrollo que se muestra a continuación:

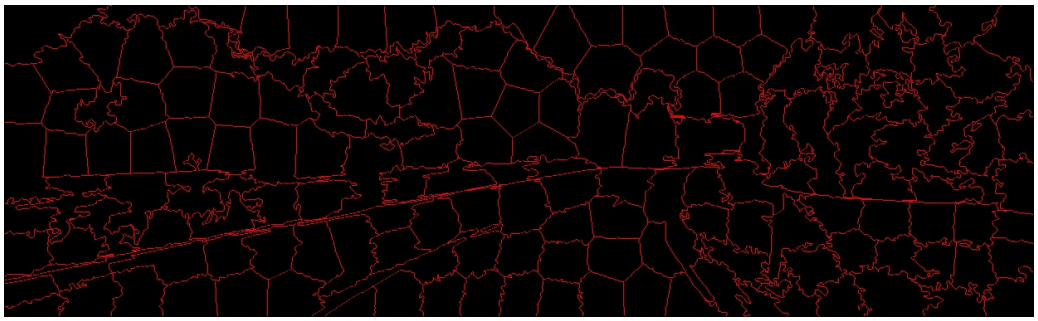
$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}$$

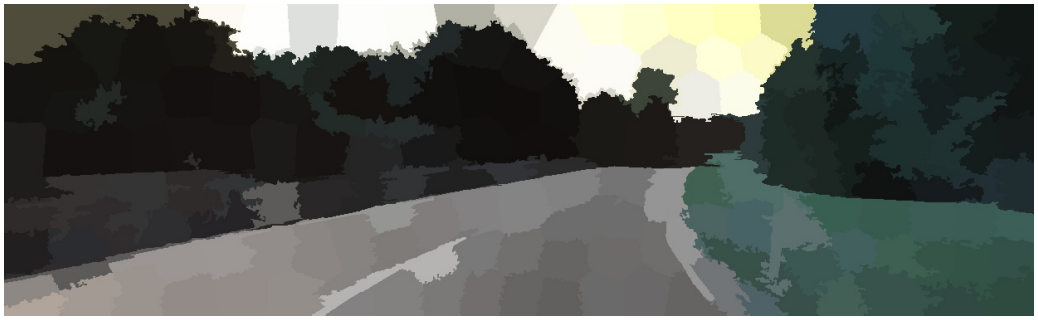
$$D_s = \sqrt{d_{lab}^2 + \frac{m^2}{S^2} d_{xy}^2}$$

dónde D_s es la distancia normalizada entre cada píxel y el centro del conjunto, que tiene en mente la distancia en el espacio de color CIELAB (d_{lab}) y en el plano imagen (d_{xy}). El parámetro m tiene la finalidad de evitar que la componente de distancia espacial

(a) Imagen perteneciente a la secuencia KITTI 01



(b) Bordes de los superpíxeles generados por SLIC



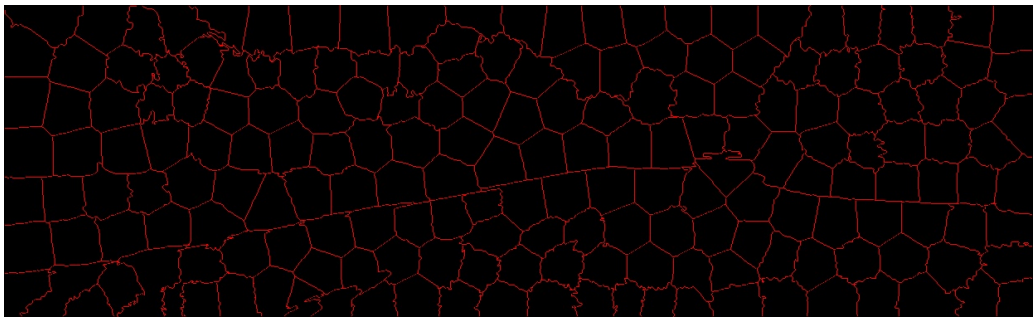
(c) Superpíxeles generados con el algoritmo SLIC

Figura 3.1: Imagen de ejemplo y resultados de la segmentación con SLIC

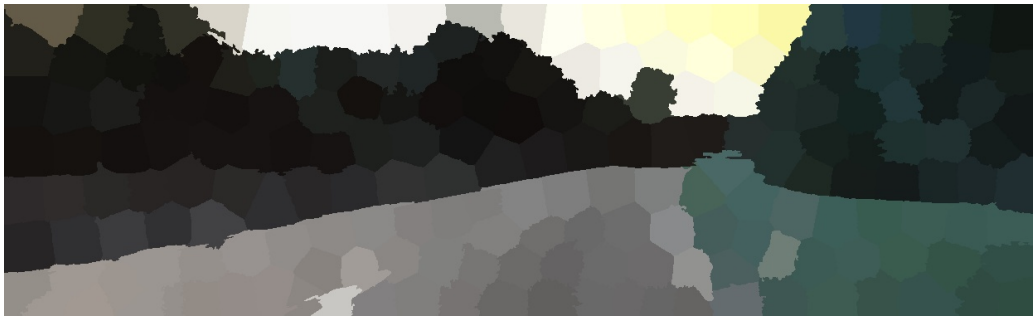
devalúe similitudes de color, pudiendo provocar que los límites de los superpíxeles no se ajusten correctamente a los bordes de los objetos representados en la imagen. Se denomina *factor de compacidad* y es constante para todo el proceso de segmentación. El algoritmo 1 ilustra el proceso de segmentacióñn con SLIC.

A lo largo de este proceso se emplea un único factor de compacidad. Esto genera que para imágenes con ligeros contrastes y tonos homogéneos se generen superpíxeles regulares y de tamaños muy similares. En cambio, en imágenes con mucha textura, se obtienen superpíxeles muy irregulares con formas y tamaños diversos, como puede verse en la Fig. 3.1. Por ello, se ha usado también la variante SLICO (también conocido como *SLIC zero parameter*), la cual tiene un factor de compacidad dinámico que varía según las características propias de la región de la imagen que se está analizando, de manera que divide la imagen en segmentos más regulares, como se aprecia en la Fig. 3.2.

(a) Imagen perteneciente a la secuencia KITTI 01



(b) Bordes de los superpíxeles generados por SLICO



(c) Superpíxeles generados con el algoritmo SLICO

Figura 3.2: Resultados de la segmentación con SLICO

3.1.2 MSLIC

El objetivo de esta versión modificada de SLIC es añadir información de textura a la hora de realizar la segmentación de la imagen [22]. Para ello, el vector C_k de cinco dimensiones usado anteriormente, en el cual se definen las características de color del píxel y su ubicación 2D en la imagen, se transforma en un vector de cuatro componentes de longitud variable $[C, t, x, y]$, donde C es a su vez un vector de longitud variable, entre 1 y 3, en función de los rangos de color que componen la imagen, y t contiene las características de textura. Por ello, esta modificación implica un ajuste en el cálculo de la distancia entre los centros de los conjuntos y los píxeles colindantes, que se obtiene según la siguiente expresión:

$$D_s = \sqrt{d_{CT}^2 + \frac{m^2}{S^2} d_{xy}^2}.$$

Algorithm 1: Segmentación SLIC

1 Inicializar los centros de los conjuntos $C_k = [l_k, a_k, b_k, x_k, y_k]^T$ muestreando los píxeles en una cuadrícula regular de S casillas.
2 Desplazar los centros de los conjuntos en una vecindad de $n \times n$ píxeles, hacia la posición de menor gradiente.
3 **repeat**
4 **for** *cada centro de conjunto* C_k **do**
5 Asignar los píxeles que más se ajustan dentro de una vecindad cuadrada de $2S \times 2S$ alrededor del centro del conjunto en función de la distancia
6 Obtener los nuevos centros de conjuntos y el error residual E
7 **until** E ≤ valor umbral
8 Forzar conectividad de conjuntos

La componente d_{CT} es la combinación de la distancia de color y textura, obtenida según la expresión siguiente:

$$d_{CT} = \sqrt{D_C^2 + D_T^2},$$

donde T corresponde a la inversa de la matriz de textura, obtenida mediante la matriz de co-ocurrencia de niveles de gris (conocida más comúnmente por sus siglas en inglés, Gray Level Co-Ocurrence Matrix (GLCM)). El motivo por el cual se utiliza la inversa se debe a que la distancia debe aumentar de manera inversamente proporcional a la similitud, y la matriz resultante de GLCM dará valores más altos cuando más homogénea sea la imagen.

3.1.3 LSC

Al igual que SLIC, se trata de un algoritmo de segmentación basado en conjuntos. No obstante, en este caso, se tienen en cuenta también parámetros globales de la imagen y los relaciona con la información local [23]. De esta forma, la estructura global de la imagen se utiliza de forma implícita para obtener una mejor segmentación. La Fig. 3.4 ilustra una segmentación de este algoritmo.

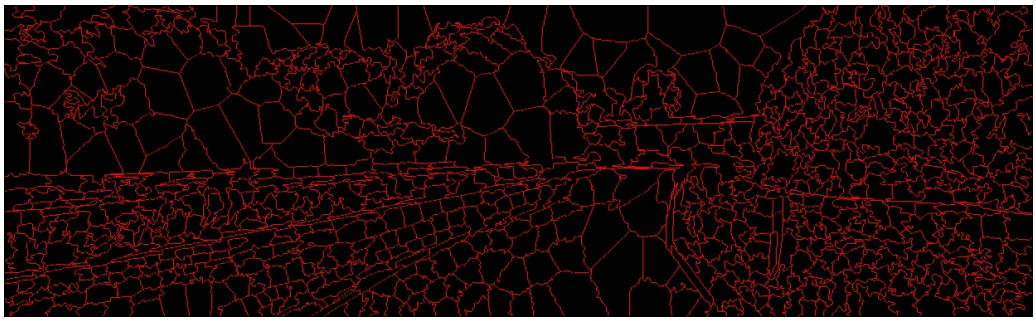
Para obtener este comportamiento, cada píxel p de la imagen se mapea en un espacio de características mediante una función $\phi(p)$. En dicho espacio se aplica un proceso de k-medias ponderadas para obtener la segmentación, mediante la minimización iterativa de una función objetivo F_{km}:

$$F_{km} = \sum_{k=1}^{K} \sum_{p \in \pi_k} \omega(p) \|\phi(p) - m_k\|^2$$

$$m_k = \frac{\sum_{p \in \pi_k} \omega(p)\phi(p)}{\sum_{p \in \pi_k} \omega(p)},$$

donde p representa un píxel, $w(p)$ es el peso correspondiente al píxel p, K es el número de conjuntos, π_k es el conjunto número k y m_k corresponde al centro del conjunto π_k. Este procedimiento es análogo a aplicar cortes normalizados en el espacio de imagen, con la ventaja de que la transformación del espacio de trabajo permite establecer relaciones no lineales que no podrían identificarse en el espacio original.

(a) Imagen perteneciente a la secuencia KITTI 01



(b) Bordes de los superpíxeles generados por MSLIC

(c) Superpíxeles generados con el algoritmo MSLIC

Figura 3.3: Resultados de la segmentación con MSLIC

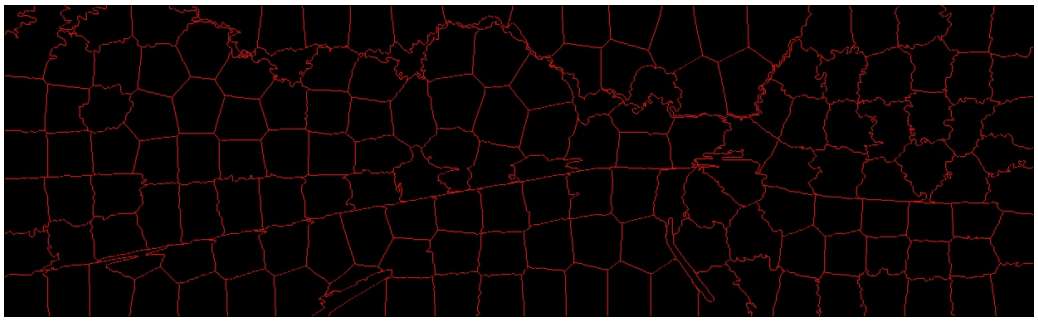
3.1.4 SEEDS

Al contrario que el resto de soluciones, este algoritmo opta por un punto de partida en el que la imagen se divide en un número determinado de bloques formando una cuadrícula regular [24]. A partir de ahí, dichos bloques se van refinando intercambiando píxeles con los segmentos vecinos siguiendo una jerarquía de niveles. El criterio para este intercambio se basa en comparativas de color y un parámetro de suavizado de los bordes de cada segmentos mediante la maximización de la función de energía:

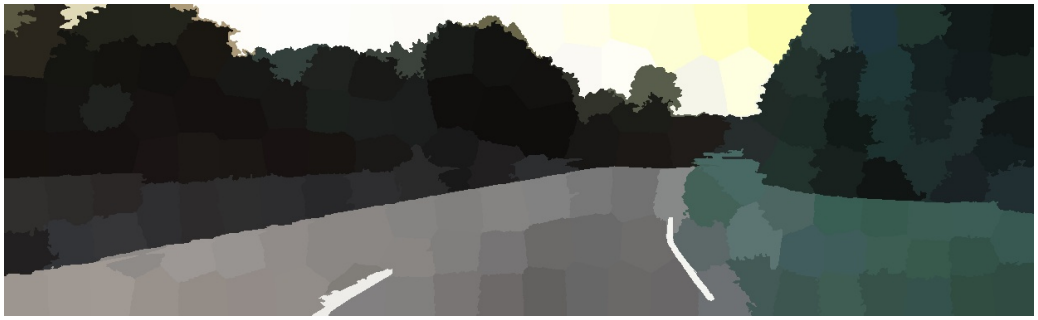
$$E(s) = H(s) + \gamma G(s),$$

donde s hace referencia a una partición concreta. El término $H(s)$ se basa en la similitud del color de los superpíxeles. Esto se consigue evaluando la distribución de densidad de cada segmento independientemente del resto de la imagen. Para ello, se genera un histograma de color, forzando que tras cada iteración el conjunto de

(a) Imagen perteneciente a la secuencia KITTI 01



(b) Bordes de los superpíxeles generados por LSC



(c) Superpíxeles generados con el algoritmo LSC

Figura 3.4: Imagen de ejemplo y resultados de la segmentación con LSC

valores se centre en un solo rango. Por otro lado, la componente $G(s)$ se encarga de la forma del superpíxel, estableciendo un cuadrado que engloba el superpíxel y midiendo la desviación entre sus lados y los bordes del segmento. Esta medida la lleva a cabo haciendo un recuento del número de píxeles pertenecientes a otros segmentos que se encuentran presentes dentro del cuadrado establecido, obteniendo un valor máximo cuando todos los píxeles pertenecen a un único superpíxel. Finalmente, γ define la influencia de cada término dentro de la función de energía.

El proceso se ilustra en el Alg. 2. Sea I una imagen a color, L el número de niveles, K el número deseado de superpíxeles y S el número final de superpíxeles. Inicialmente la imagen se divide agrupando píxeles en bloques de $w^{(1)} \times h^{(1)}$ píxeles. Posteriormente, estos bloques se van reagrupando de forma recurrente L veces hasta que en el último nivel $w^{(l)} = w^{(1)} \cdot 2^{l-1}$ y $h^{(l)} = h^{(1)} \cdot 2^{l-1}$. En el transcurso de los niveles, se comparan los histogramas de cada bloque ($h_{B_i^{(i)}}$) con el del superpíxel (h_{S_j}) al que pertenecen y

Algorithm 2: Segmentación LSC

```
Mapear cada punto p = [l_p, a_p, b_p, x_p, y_p] según la función φ(p).
Muestrear K semillas a lo largo de la imagen de forma uniforme en intervalos
  horizontales y verticales fijos υ_x y υ_y.
Mover cada semilla al punto de menor gradiente dentro de una vecindad de
  3 × 3 píxeles.
Inicializar cada media ponderada m_k y obtener el centro c_k de cada conjunto
  en función de la semilla correspondiente.
Inicializar la etiqueta L(p) = 0 para cada punto p.
Inicializar la distancia D(p) = ∞ para cada punto p.
repeat
    for cada media ponderada m_k y cada centro c_k do
        for cada punto p en vecindad τυ_x × τυ_y de c_k en el plano de imagen do
            D = distancia euclidea entre φ(p) y m_k en el espacio de
              características
            if D < d(p) then
                d(p) = D
                L(p) = k
    Actualizar medias ponderadas y buscar centros para todos los conjuntos
until todos los conjuntos converjan
Unir superpíxeles pequeños a sus vecinos
```

el de sus vecinos, de manera que el bloque se asigna al superpíxel con el que guarda mayor similitud. Para el refinamiento de los bordes, se sigue un procedimiento análogo al nivel de píxeles individuales $h(x_n)$.

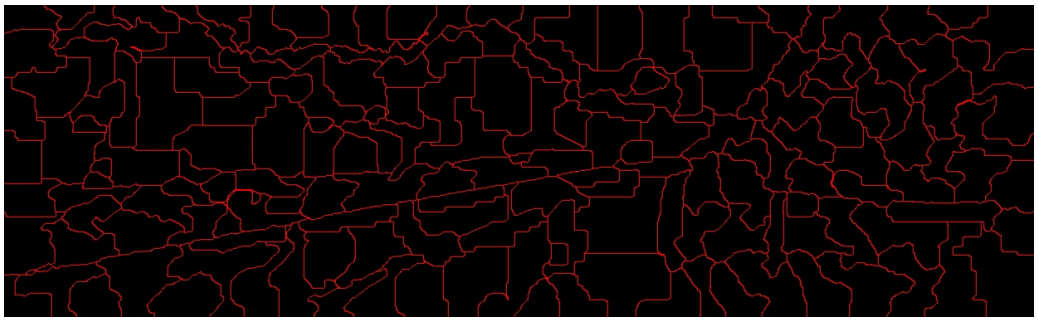
Este procedimiento requiere un consumo de memoria muy bajo por parte del computador, lo cual se traduce en un ahorro computacional importante.

3.2 Metodología

En primer lugar, para poder evaluar las diferencias en el rendimiento de cada uno de los algoritmos es crucial identificar qué parámetros son los más relevantes en la tarea de reconocimiento de escenas que aborda este trabajo.

De cara a comparar los algoritmos expuestos en la sección anterior, consideraremos una simplificación de las características expuestas en la sección dedicada a los superpíxeles del capítulo 2. Siendo así, el rendimiento de los algoritmos se comparará en los siguientes términos (de mayor a menor relevancia): estabilidad, compacidad y calidad. La estabilidad se medirá en base a la similitud entre las segmentaciones generadas por el algoritmo tomando como entrada, por un lado una escena, y por otro, esa misma escena vista desde otra perspectiva similar (bucle). La compacidad la entendemos como la medida de que los segmentos generados por el algoritmo tengan una forma y tamaño regulares, y se distribuyan uniformemente por la imagen. Por último, la calidad hace referencia a la capacidad del algoritmo para generar una segmentación que se adhiera a los bordes de los objetos representados en la escena. La estabilidad se ha evaluado

(a) Imagen perteneciente a la secuencia KITTI 01



(b) Bordes de los superpíxeles generados por SEEDS

(c) Superpíxeles generados con el algoritmo SEEDS

Figura 3.5: Resultados de la segmentación con SEEDS

de forma cuantitativa mediante dos métricas que se exponen en el apartado 3.2.2. En cambio, la compacidad y la calidad se evalúan de forma cualitativa comparando las segmentaciones resultantes de cada algoritmo.

Todos los experimentos han sido desarrollados usando Python, junto con las librerías *scikit-image* y OpenCV para tratamiento de imágenes y visión por computador.

Para cada algoritmo se han probado una serie de configuraciones modificando el valor de los parámetros de cara a alcanzar unos resultados óptimos sobre las secuencias escogidas.

3.2.1 Secuencias

Para medir estos parámetros de ejecución de cada uno de los algoritmos, se han seleccionado dos secuencias tomadas por una plataforma móvil que nos aportan una colección de fotogramas con escenarios urbanos. Las diferencias entre un fotograma

Algorithm 3: Segmentación SEEDS

Inicialización de jerarquía de bloques y segmentación incial de superpíxeles.
for *cada nivel l de 1 a L* **do**
    for *cada bloque* $B_i^{(l)}$ **do**
        Inicializar histograma $h_{B_i^{(l)}}$
for *cada nivel l de* $L-1$ *a* 1 **do**
    for *cada bloque* $B_i^{(l)}$ **do**
        if *bloque de píxels vecino pertenece a un superpixel diferente* **then**
            if $\cap(h_{B_i^{(l)}}, h_{S_k}) > \cap(h_{B_i^{(l)}}, h_{S_j})$ **then**
                $S_k := S_k \cup B_i^{(l)}$, $S_j := S_j - B_i^{(l)}$
for *cada píxel n desde 1 hasta* $W \times H$ **do**
    if *píxel vecino pertenece a superpíxel diferente* **then**
        if $h_{S_k}(h(x_n)) > h_{S_j}(h(x_n))$ **then**
            $S_k := S_k \cup x_n$, $S_j := S_j - x_n$
return S

y el siguiente radican fundamentalmente en el desplazamiento de la propia cámara en el sentido de la marcha del vehículo. Dichas secuencias son públicas y han sido capturadas por el Karlsruhe Institute of Technology and Toyota Technological Institute, dando lugar a las secuencias KITTI. Concretamente, para este experimento se ha hecho uso de las secuencias KITTI00 y KITTI01 en color. De cada una de ellas se ha escogido un fragmento de 90 imágenes consecutivas.

3.2.2 Métricas

De las segmentaciones obtenidas de cada uno de ellos, se ha valorado la estabilidad de las regiones reconocidas de un fotograma al siguiente en base a dos métricas, ambas consistentes en calcular en qué medida cada superpíxel reconocido en un fotograma en un instante determinado ocupa la misma posición en el fotograma del instante inmediatamente posterior.

La primera de ellas, bautizada como factor de solapamiento S, consiste en buscar en primer lugar los índices de los dos segmentos con una mayor coincidencia de píxeles entre un fotograma y el siguiente. Una vez identificados, ese número de píxeles coincidentes se divide entre el número total de píxeles de ese segmento en ese instante determinado. Así pues, definimos ese solapamiento como:

$$S(SP_A) = \frac{|SP_A \cap SP_B|}{|SP_A|}$$

donde $A \cap B$ es la intersección entre los superpixeles A y B y $|A|$ es el cadrinal del conjunto A.

Posteriormente, se define un factor de solapamiento global SG de cada fotograma, como el solapamiento medio del conjunto de superpíxeles:

$$S_G(t) = \frac{\sum_{i=1}^{N} S(SP_i)}{N}$$

siendo N el número de superpíxeles de la imagen t.

La segunda medida consiste en aplicar el índice de Jaccard, también conocido como Intersection over Union (IoU) [35], para medir el grado de similitud entre dos conjuntos. De esta forma, simplemente se divide el número de píxeles coincidentes entre los dos superpíxeles con una mayor coincidencia entre dos imágenes consecutivas y se divide entre la suma de píxeles no coincidentes de ambos segmentos.

$$J(SP_A) = \frac{|SP_A \cap SP_B|}{|SP_A| + |SP_B| - |SP_A \cap SP_B|}$$

donde $A \cap B$ es la intersección entre los superpixeles A y B, y $|A|$ es el cardinal del conjunto A y $|B|$ es el cardinal del conjunto B.

De la misma forma, se calcula también un índice global para cada frame:

$$J_G(t) = \frac{\sum_{i=1}^{N} J(SP_i)}{N}$$

siendo N el número de superpíxeles de la imagen t.

3.3 Resultados

En este apartado se presentan los resultados obtenidos al someter cada algoritmo a las métricas expuestas. Para cada uno de ellos se ha probado una variedad de parámetros de configuración para observar los cambios en su rendimiento y escoger la configuración más favorable en el caso de las secuencias seleccionadas. Los resultados obtenidos se recogen en las Tablas 3.1 a 3.10. En las gráficas de las Fig. 3.6 y 3.7 se exponen los resultados obtenidos para cada algoritmo en cada una de sus configuraciones para cada secuencia: el eje de las ordenadas corresponde a los valores del índice J y en el de las abscisas se enumeran todas las combinaciones probadas de cada algoritmo en el mismo orden en el que se exponen en las tablas.

En primer lugar en la Fig. 3.1, podemos ver como SLIC genera unos superpíxeles muy irregulares en general, salvo en las regiones más homogéneas. Esa distribución errática de los segmentos da lugar a que los índices de solapamiento S y Jaccard J sean muy bajos al no existir apenas coincidencia entre los segmentos de una imagen y la siguiente. En las Tablas 3.1 y 3.2 se muestran los resultados obtenidos con el algoritmo SLIC en términos de estabilidad para diferentes configuraciones de parámetros para las secuencias KITTI00 y KITTI01, respectivamente. Los parámetros modificados son: el tamaño deseado para cada superpíxel; un factor de suavidad de bordes o compacidad de los superpíxeles generados, denominado *ruler*; y el tamaño mínimo para cada superpíxel. En la última fila se expone el número medio de superpíxeles obtenido para la secuencia de imágenes.

Por otro lado, tal y como cabía esperar, SLICO es mucho más regular en la segmentación de la imagen, generando unos índices de solapamiento más altos. No obstante,

Tam. spx	**40**	**50**	**60**	**50**	**50**	**50**	**60**	**60**
Ruler	**15**	**15**	**15**	**25**	**35**	**25**	**25**	**35**
Tam. min spx	**30**	**30**	**30**	**30**	**40**	**50**	**40**	**40**
S mínimo	0.018	0.024	0.015	0.015	0.014	0.013	0.023	0.013
S máximo	0.110	0.120	0.180	0.135	0.165	0.155	0.189	0.184
S medio	0.055	0.065	0.093	0.067	0.073	0.074	0.099	0.104
S desv. est.	0.023	0.021	0.036	0.029	0.027	0.032	0.041	0.043
J mínimo	0.012	0.016	0.009	0.009	0.008	0.008	0.016	0.008
J máximo	0.083	0.088	0.131	0.103	0.124	0.120	0.141	0.142
J medio	0.037	0.044	0.063	0.047	0.053	0.053	0.071	0.077
J desv. est.	0.018	0.016	0.026	0.022	0.021	0.026	0.032	0.034
Núm. medio spx	237.26	167.71	105.88	178.28	184.75	156.24	111.84	116.13

Tabla 3.1: Resultados obtenidos para el algoritmo SLIC sobre la secuencia KITTI00

Tam. spx	**40**	**50**	**60**	**60**	**50**	**50**	**60**	**60**
Ruler	**15**	**15**	**15**	**25**	**35**	**25**	**25**	**35**
Tam. min spx	**30**	**30**	**30**	**30**	**40**	**50**	**40**	**40**
S mínimo	0.019	0.015	0.025	0.020	0.024	0.027	0.020	0.027
S máximo	0.125	0.165	0.200	0.227	0.188	0.165	0.227	0.237
S medio	0.069	0.086	0.111	0.115	0.100	0.091	0.115	0.132
S desv. est.	0.026	0.034	0.043	0.047	0.037	0.039	0.047	0.055
J mínimo	0.010	0.010	0.016	0.013	0.017	0.018	0.013	0.019
J máximo	0.102	0.129	0.163	0.188	0.151	0.136	0.188	0.211
J medio	0.051	0.064	0.081	0.088	0.079	0.069	0.088	0.104
J desv. est.	0.023	0.029	0.036	0.040	0.033	0.034	0.040	0.048
Núm. medio spx	230.89	159.49	100.15	110.13	186.17	156.49	110.13	116.36

Tabla 3.2: Resultados obtenidos para el algoritmo SLIC sobre la secuencia KITTI01

la regularidad en la creación de los segmentos hace que quede de lado la adherencia a los bordes, perdiendo así una cantidad muy relevante de información de la imagen original, tal y como ya se ha ejemplificado a través de la Fig. 3.2. Al igual que con SLIC, las tablas 3.3 y 3.4 muestran los resultados obtenidos para diferentes conjuntos de parámetros. En este caso, aparecen menos combinaciones ya que, como se ha comentado, el parámetro de compacidad es dinámico y su valor depende de las características de cada región concreta de la imagen.

Con MSLIC se obtiene una gran cantidad de superpíxeles, tal como ilustra la Fig. 3.3. Dado el amplio volumen de segmentos y la irregularidad con la que se distribuyen a lo largo de la imagen, los valores de lo índices empleados para la evaluación resultan extremadamente bajos. Al igual que con SLIC, las Tablas 3.5 y 3.6 muestran los valores obtenidos para cada configuración de parámetros del algoritmo.

Los resultados obtenidos para LSC muestran un comportamiento muy similar al de SLICO, aunque con una ligera mejora en los resultados de solapamiento. En las imágenes de segmentación obtenidas puede observarse que los segmentos generados se adhieren con cierta fidelidad a los objetos capturados en cada imagen, tal y como se puede apreciar en la Fig. 3.4. En las Tablas 3.7 y 3.8 se muestran los resultados obtenidos sobre las respectivas secuencias. En este caso, se muestran el tamaño medio

Tam. spx	**40**	**60**	**50**	**50**
Tam. min spx	**30**	**30**	**30**	**50**
S mínimo	0.023	0.028	0.038	0.031
S máximo	0.150	0.218	0.187	0.178
S medio	0.078	0.118	0.093	0.092
S desv. est.	0.033	0.050	0.038	0.037
J mínimo	0.015	0.020	0.026	0.021
J máximo	0.120	0.178	0.152	0.145
J medio	0.061	0.092	0.072	0.071
J desv. est.	0.028	0.043	0.033	0.032
Núm. medio spx	277.78	122.46	191.02	185.91

Tabla 3.3: Resultados obtenidos para el algoritmo SLICO sobre la secuencia KITTI00

Tam. spx	**40**	**50**	**60**	**50**
Tam. min spx	**30**	**30**	**30**	**50**
S mínimo	0.035	0.030	0.051	0.028
S máximo	0.162	0.197	0.271	0.192
S medio	0.097	0.109	0.137	0.105
S desv. est.	0.034	0.045	0.062	0.047
J mínimo	0.024	0.023	0.033	0.018
J máximo	0.138	0.164	0.234	0.154
J medio	0.079	0.088	0.112	0.084
J desv. est.	0.030	0.040	0.055	0.041
Núm. medio spx	277.65	191.39	122.55	186.92

Tabla 3.4: Resultados obtenidos para el algoritmo SLICO sobre la secuencia KITTI01

de los superpíxeles, el tamaño mínimo y el factor de compacidad, que en este caso se denomina *ratio*.

Por último, SEEDS es el que ha aportado mejores resultados en ambos índices de manera más que notable. En las imágenes puede observarse como los superpíxeles se distribuyen de forma muy regular con superficies muy similares de una imagen a la siguiente, permitiendo así un mayor solapamiento, tal y como puede verse en la Fig. 3.5. Por otro lado, se observa que la adherencia a los bordes es muy inferior al resto de algoritmos. En las tablas 3.9 y 3.10 se exponen los resultados obtenidos. En el caso de este algoritmo, los parámetros corresponden al número de superpíxeles deseado, un factor de suavizado denominado *prior*, el número de niveles o pasos en los que se va a ejecutar el proceso de segmentación, y el número de clases que compondrán los histogramas de cada bloque.

3.4 Discusión

En las tablas 3.11 y 3.12 se recogen los mejores resultados obtenidos para cada algoritmo en aquellas configuraciones para las que se obtenían un número de superpíxeles dentro de un rango similar. Se ha destacado el mejor valor medio y el segundo mejor, en rojo

Tam. spx	**40**	**50**	**60**	**50**	**50**	**50**	**60**	**60**
Ruler	**15**	**15**	**15**	**35**	**35**	**25**	**25**	**35**
Tam. min spx	**30**	**30**	**30**	**30**	**40**	**50**	**40**	**40**
S mínimo	0.001	0.001	0.001	0.005	0.001	0.002	0.002	0.002
S máximo	0.023	0.018	0.027	0.042	0.034	0.039	0.031	0.040
S medio	0.007	0.008	0.008	0.017	0.012	0.014	0.011	0.017
S desv. est.	0.004	0.004	0.005	0.008	0.007	0.008	0.006	0.009
J mínimo	0.000	0.001	0.001	0.002	0.001	0.001	0.001	0.001
J máximo	0.012	0.010	0.014	0.025	0.019	0.021	0.017	0.023
J medio	0.003	0.004	0.004	0.010	0.006	0.008	0.006	0.009
J desv. est.	0.002	0.002	0.003	0.005	0.004	0.004	0.003	0.005
Núm. medio spx	2023.63	1772.77	1483.52	674.75	1056.78	784.65	994.57	659.52

Tabla 3.5: Resultados obtenidos para el algoritmo MSLIC sobre la secuencia KITTI00

Tam. spx	**40**	**50**	**60**	**60**	**50**	**50**	**60**
Ruler	**15**	**15**	**15**	**25**	**35**	**25**	**35**
Tam. min spx	**30**	**30**	**30**	**30**	**40**	**50**	**40**
S mínimo	0.001	0.001	0.000	0.002	0.003	0.003	0.003
S máximo	0.015	0.018	0.031	0.033	0.042	0.033	0.034
S medio	0.005	0.006	0.007	0.011	0.016	0.013	0.016
S desv. est.	0.003	0.003	0.004	0.006	0.008	0.006	0.007
J mínimo	0.000	0.001	0.000	0.001	0.001	0.002	0.002
J máximo	0.009	0.009	0.016	0.017	0.025	0.019	0.018
J medio	0.003	0.003	0.004	0.006	0.010	0.007	0.009
J desv. est.	0.002	0.002	0.002	0.004	0.005	0.004	0.004
Núm. medio spx	2353.03	1878.57	1431.28	992.16	715.91	825.67	687.32

Tabla 3.6: Resultados obtenidos para el algoritmo MSLIC sobre la secuencia KITTI01

y en naranja respectivamente, para los parámetros S y J. MSLIC no se ha incluido ya que en ningún caso dio lugar a un número de segmentos comparable al resto. Basándonos en estos valores, el algoritmo que ha mostrado un mejor rendimiento de acuerdo con los parámetros de evaluación propuestos ha sido SEEDS, con una diferencia más que notable respecto al resto de métodos de segmentación. La literatura consultada ya destaca el buen rendimiento de este algoritmo en términos de precisión en la segmentación de objetos y adherencia a los bordes existentes en la imagen [17] [23] [24]. En el caso de este proyecto, ha demostrado también realizar un buen papel en términos de estabilidad según las métricas propuestas. Para ambas secuencias ha sido el que ha mostrado un mejor desempeño. Trasladando los datos referentes al índice de Jaccard J en relación al número de superpíxeles, ilustrados en las gráficas 3.8 y 3.9, dicha diferencia es aún más llamativa. También es cierto que esa estabilidad de la segmentación generada se produce a expensas de una menor adherencia a los bordes de los objetos en la imagen, tal y como podemos ver en la imagen 3.5, donde se aprecia la compacidad de los segmentos obtenidos.

En cuanto a SLIC y sus variantes, parece que podemos determinar que tanto la implementación original como MSLIC son las que han obtenido peor resultado. En el

Tam. spx	**50**	**50**	**50**	**40**	**40**	**40**
Tam. min spx	**40**	**40**	**40**	**30**	**30**	**30**
Ratio	**0.05**	**0.075**	**0.09**	**0.05**	**0.075**	**0.09**
S mínimo	0.0166	0.0128	0.0187	0.0091	0.0152	0.0237
S máximo	0.2177	0.2021	0.2527	0.1376	0.1641	0.1814
S medio	0.0904	0.1061	0.1097	0.0663	0.0817	0.0879
S desv. est.	0.0396	0.0476	0.0471	0.0299	0.0297	0.0357
J mínimo	0.0093	0.0072	0.0117	0.0053	0.0082	0.0153
J máximo	0.1765	0.1685	0.2053	0.1077	0.1303	0.1505
J medio	0.0636	0.0779	0.0819	0.0459	0.0596	0.0660
J desv. est.	0.0324	0.0399	0.0397	0.0240	0.0246	0.0301
Núm. medio spx	173.81	172.4	172.72	278.01	276.23	276.04

Tabla 3.7: Resultados obtenidos para el algoritmo LSC sobre la secuencia KITTI00

Tam. spx	**50**	**50**	**50**	**40**	**40**	**40**
Tam. min spx	**40**	**40**	**40**	**30**	**30**	**30**
Ratio	**0.05**	**0.075**	**0.09**	**0.05**	**0.075**	**0.09**
S mínimo	0.0172	0.0459	0.0486	0.0318	0.0419	0.0463
S máximo	0.2174	0.2779	0.3405	0.1944	0.2338	0.2835
S medio	0.1195	0.1577	0.1690	0.0979	0.1233	0.1371
S desv. est.	0.0443	0.0536	0.0584	0.0346	0.0411	0.0491
J mínimo	0.0096	0.0283	0.0383	0.0184	0.0273	0.0324
J máximo	0.1722	0.2409	0.2986	0.1665	0.1996	0.2370
J medio	0.0907	0.1285	0.1400	0.0745	0.0996	0.1135
J desv. est.	0.0381	0.0494	0.0541	0.0304	0.0374	0.0449
Núm. medio spx	168.52	168.94	168.78	271.72	271.40	271.16

Tabla 3.8: Resultados obtenidos para el algoritmo LSC sobre la secuencia KITTI01

caso de este último, dada la gran cantidad de segmentos que genera, es comprensible que las medidas de estabilidad sean mucho más bajas, ya que cualquier variación en la región de la imagen ocupada por un superpíxel entre un fotograma y el siguiente tiene mucho más impacto en los índices de estabilidad analizados. Por otro lado, si observamos las segmentaciones expuestas en las Fig. 3.1 y 3.2, podemos ver la efectividad del algoritmo SLICO a la hora de ajustar su compacidad en función de la textura de la región concreta de cada segmento, generando una segmentación regular que favorece un mejor resultado que SLIC en los términos de estabilidad planteados. Es destacable su rendimiento en la secuencia KITTI00, como puede verse en la Tabla 3.11, obteniendo el segundo mejor resultado. No obstante, LSC demuestra un rendimiento muy similar en esta secuencia y lo supera con algo más de margen en la secuencia KITTI01, como puede verse en la Tabla 3.12.

En la Tabla 3.13 se recoge la valoración asignada a cada algoritmo en términos de compacidad y calidad según su desempeño en los experimentos realizados sobre la secuencia KITTI01. Dicha valoración puede ser "Muy baja", "Baja", "Media", "Alta" o "Muy alta". Del mismo modo, a cada valoración se le ha asignado una nota, del 1 al 5

Núm. de spp.	**200**	**300**	**400**	**300**	**300**	**300**	**300**	**300**	**300**	**300**	**400**
Prior	**1**	**1**	**1**	**3**	**4**	**5**	**4**	**4**	**4**	**4**	**4**
Núm. niveles	**3**	**3**	**3**	**3**	**3**	**3**	**4**	**5**	**4**	**4**	**4**
Núm. clases hist.	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**5**	**6**	**5**
S mínimo	0.488	0.420	0.401	0.468	0.481	0.473	0.476	0.457	0.471	0.455	0.463
S máximo	0.609	0.520	0.473	0.565	0.576	0.575	0.590	0.570	0.602	0.593	0.575
S medio	0.532	0.468	0.439	0.525	0.533	0.530	0.530	0.515	0.536	0.528	0.511
S desv. est.	0.023	0.020	0.015	0.020	0.019	0.019	0.026	0.026	0.026	0.025	0.025
J mínimo	0.309	0.255	0.237	0.319	0.325	0.311	0.320	0.306	0.314	0.303	0.310
J máximo	0.448	0.353	0.314	0.411	0.413	0.404	0.434	0.417	0.449	0.425	0.422
J medio	0.360	0.297	0.268	0.371	0.374	0.367	0.373	0.359	0.378	0.371	0.356
J desv. est.	0.024	0.019	0.015	0.018	0.018	0.017	0.024	0.024	0.027	0.023	0.025
Núm. medio spx	132	224	340	224	224	224	175	175	175	175	279

Tabla 3.9: Resultados obtenidos para el algoritmo SEEDS sobre la secuencia KITTI00

Núm. de spp.	**200**	**300**	**400**	**300**	**300**	**300**	**300**	**300**	**300**	**300**	**400**
Prior	**1**	**1**	**1**	**3**	**4**	**5**	**4**	**4**	**4**	**4**	**4**
Núm. niveles	**3**	**3**	**3**	**3**	**3**	**3**	**4**	**5**	**4**	**4**	**4**
Núm. clases hist.	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**5**	**6**	**5**
S mínimo	0.452	0.399	0.388	0.457	0.459	0.461	0.429	0.415	0.440	0.433	0.451
S máximo	0.608	0.538	0.491	0.616	0.613	0.609	0.616	0.604	0.613	0.595	0.576
S medio	0.532	0.469	0.444	0.530	0.532	0.529	0.530	0.514	0.538	0.528	0.515
S desv. est.	0.038	0.030	0.023	0.032	0.030	0.029	0.039	0.039	0.038	0.035	0.029
J mínimo	0.266	0.226	0.223	0.307	0.306	0.307	0.277	0.267	0.289	0.290	0.301
J máximo	0.442	0.356	0.313	0.457	0.452	0.447	0.462	0.450	0.462	0.451	0.417
J medio	0.359	0.296	0.273	0.378	0.377	0.370	0.376	0.361	0.381	0.374	0.360
J desv. est.	0.041	0.029	0.023	0.031	0.030	0.029	0.038	0.037	0.038	0.035	0.028
Núm. medio spx	132	224	340	224	224	224	175	175	175	175	279

Tabla 3.10: Resultados obtenidos para el algoritmo SEEDS sobre la secuencia KITTI01

respectivamente, y se ha calculado una valoración total como la suma de las dos notas. Nuevamente, los mejores resultados y los segundos mejores se han destacado en rojo y naranja, respectivamente.

Si observamos la Fig. 3.5, vemos que los superpíxeles generados por el algoritmo SEEDS presentan una alta compacidad: todos los píxeles de cada segmento se encuentran muy próximos entre sí y los superpíxeles tienen formas muy similares. En este aspecto, destaca sobre LSC o SLICO (figuras 3.2 y 3.4, respectivamente), aunque su distribución en la imagen no alcanza la uniformidad que presenta tanto LSC como SLICO. Por otro lado, la adherencia a los bordes de los superpíxeles generados no es tan alta como la que se obtiene mediante LSC y SLIC (figura 3.1), en los cuales se intuye fácilmente el contorno de los elementos presentes en la imagen tras la segmentación.

En el caso de LSC, podemos apreciar una mejor adherencia a los bordes que SEEDS y SLICO [23], con una penalización menor que SLIC en las medidas de solapamiento y Jaccard expuestas. Esto se debe a que en las regiones más homogéneas de la imagen, como en el caso del cielo o el asfalto, es capaz de generar segmentos muy regulares.

Los resultados expuestos demuestran que en general SEEDS sobrepasa con amplio margen al resto de algoritmos en términos de estabilidad, obteniendo también una valoración considerablemente alta en cuanto a compacidad y calidad. No obstante, es igualmente destacable el desempeño de LSC, generando unos superpíxeles con una alta compacidad y calidad, y obteniendo un resultado notable en los experimentos de

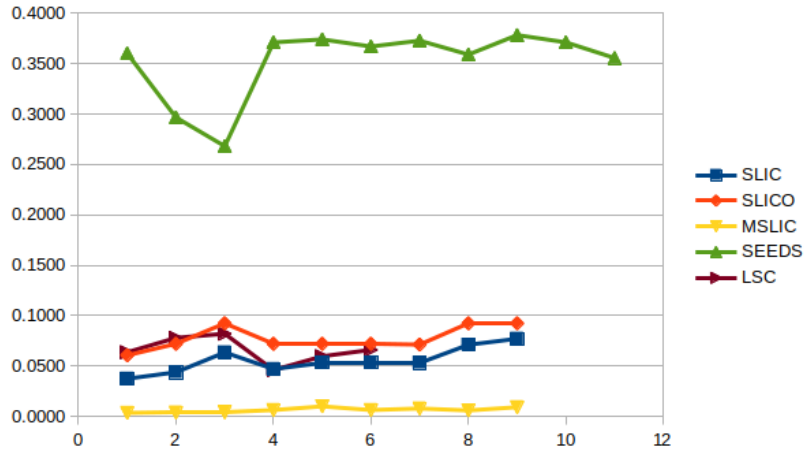


Figura 3.6: Índices de Jaccard obtenidos para las secuencias KITTI00 para cada configuración de parámetros de los algoritmos probados.

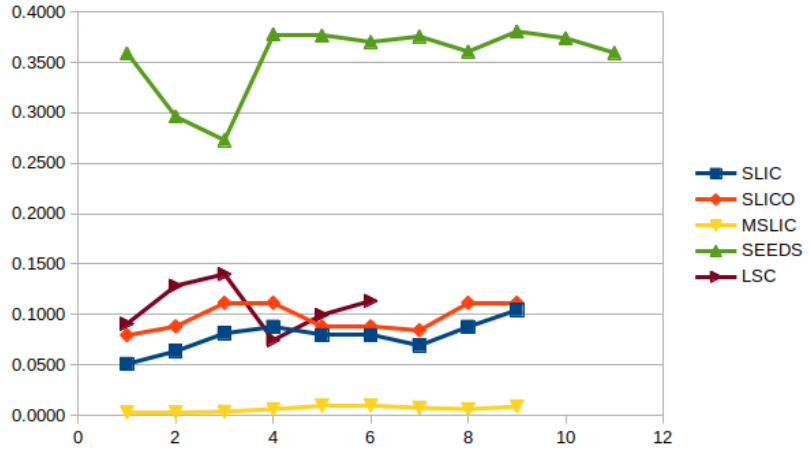


Figura 3.7: Índices de Jaccard obtenidos para las secuencias KITTI01 para cada configuración de parámetros de los algoritmos probados.

estabilidad. Estos razonamientos han servido como motivación para seguir considerando los algoritmos SEEDS y LSC en las siguientes etapas del proyecto y descartar el resto de opciones.

KITTI00	**SLIC**	**SLICO**	**SEEDS**	**LSC**
S mínimo	0.0142	0.0314	0.4706	0.0166
S máximo	0.1647	0.1778	0.6024	0.2177
S medio	0.0729	**0.0917**	**0.5363**	0.0904
S desv. est.	0.0267	0.0374	0.0260	0.0396
J mínimo	0.0078	0.0206	0.3142	0.0093
J máximo	0.1240	0.1449	0.4487	0.1765
J medio	0.0535	**0.0712**	**0.3783**	0.0636
J desv. est.	0.0214	0.0322	0.0265	0.0324
Núm. medio spx	184.75	185.91	175	173.81

Tabla 3.11: Resumen de los mejores resultados de cada algoritmo sobre la secuencia KITTI00

KITTI01	**SLIC**	**SLICO**	**SEEDS**	**LSC**
S mínimo	0.0235	0.0278	0.4402	0.0459
S máximo	0.1879	0.1916	0.6131	0.2779
S medio	0.1003	0.1049	**0.5376**	**0.1577**
S desv. est.	0.0367	0.0469	0.0376	0.0536
J mínimo	0.0166	0.0181	0.2892	0.0283
J máximo	0.1508	0.1544	0.4616	0.2409
J medio	0.0794	0.0843	**0.3809**	**0.1285**
J desv. est.	0.0329	0.0411	0.0384	0.0494
Núm. medio spx	186.17	186.92	175.00	168.94

Tabla 3.12: Resumen de los mejores resultados de cada algoritmo sobre la secuencia KITTI01

	Compacidad	**Calidad**	**Valor. TOTAL**
SLIC	Baja (2)	Alta (4)	6
SLICO	Alta (4)	Baja (2)	6
MSLIC	Muy baja (1)	Muy Alta (5)	6
LSC	Alta (4)	Alta (4)	**8**
SEEDS	Alta (4)	Media (3)	**7**

Tabla 3.13: Resumen de valoración cualitativa de los algoritmos

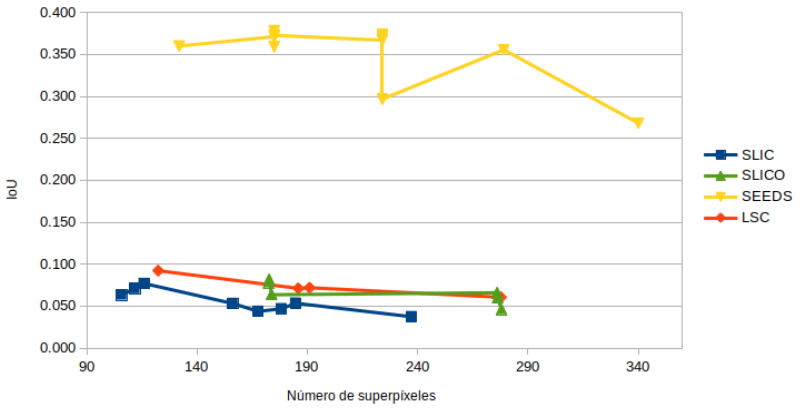


Figura 3.8: Índices de Jaccard frente número de superpíxeles para KITTI00.

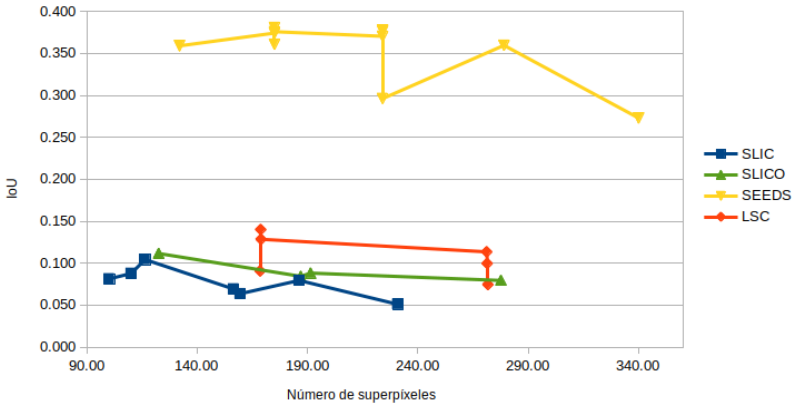


Figura 3.9: Índices de Jaccard frente número de superpíxeles para KITTI01.

CAPÍTULO 4

DESCRIPTORES

Una vez seleccionados los algoritmos de segmentación, es preciso seleccionar una técnica de descripción de superpíxeles que sea adecuada para efectuar tareas de reconocimiento de escenas. En el siguiente capítulo se exponen los mecanismos considerados en este trabajo y la metodología seguida para seleccionar dicha técnica.

Dentro de los diferentes tipos de mecanismos de descripción expuestos en el capítulo 2, se han valorado los métodos basados en características locales tradicionales y dos alternativas basadas en redes neuronales convolucionales.

4.1 Descriptor basado en histogramas de color y textura

En este trabajo, se ha implementado un descriptor de superpíxeles basado en la combinación de los histogramas de color y textura de la región rectangular de tamaño mínimo (*bounding box*) que contiene cada segmento. Para ello, por un lado se elaboran los histogramas de color en los canales L, A y B, generando un vector de 64 posiciones para cada uno de ellos.

Por otro lado, se extrae el histograma de Local Binary Pattern (LBP), para describir en términos de textura cada superpíxel [36]. El primer paso para obtener este descriptor es disponer de la región a procesar en escala de grises. Para cada píxel g de la imagen en escala de grises se selecciona una vecindad de un radio r determinado a su alrededor. De esta forma, se obtiene una matriz cuadrada de tamaño $M \times M$ donde el píxel g ocupa el centro. A continuación, se llevan a cabo una serie de comparaciones de intensidad entre el píxel central y cada uno de los píxeles vecinos. El resultado de estas comparaciones es igual a 1 sí la intensidad del píxel central es superior a la del píxel vecino, o 0 en el caso contrario. Mediante este proceso obtenemos una matriz de tamaño $M \times M$ donde la posición del píxel central g queda vacía y en el resto de posiciones se almacenan los resultados de las respectivas comparaciones, como se ilustra en la Fig. 4.1. El siguiente paso es calcular el valor LBP del píxel central. Para ello, se compone un código binario con los valores obtenidos de la comparación,

seleccionando un píxel vecino como bit menos significativo y añadiendo el resto en sentido horario o anti-horario. Este valor binario se convierte a decimal sumando los pesos correspondientes de cada bit y el resultado se almacena en la matriz LBP. En la Fig. 4.2 se muestra un ejemplo de este proceso. Finalmente se genera un histograma con todos los valores LBP extraídos de la zona de la imagen a describir.

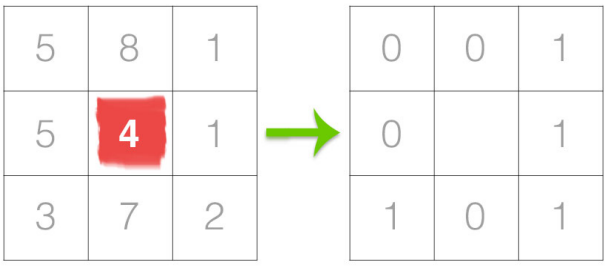


Figura 4.1: A la derecha, la matriz formada por los valores de intensidad del píxel central y su vecindad. A la izquierda, la matriz resultante tras realizar las comparaciones. [3]

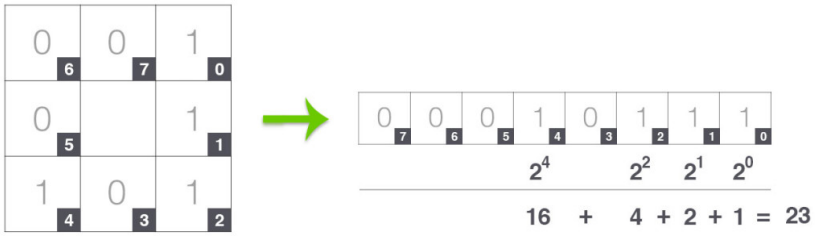


Figura 4.2: Conversión de la matriz resultante de las comparaciones con el píxel central. [3]

En el caso de este trabajo, se ha seleccionado un radio de vecindad igual a 1, por tanto, se valoran 8 píxeles vecinos. De esta forma, se genera un vector binario de 8 posiciones, dando lugar a un rango de valores LBP de 0 a $2^8 - 1 = 255$. Es importante destacar también que se ha empleado la variante de patrones uniformes [37]. Se consideran patrones uniformes aquellos que presenten como máximo dos transiciones 1-0 o 0-1. De este modo, el vector 00100000 sería uniforme, mientras que 01001100 no lo sería. Considerando solo los patrones uniformes, se consigue reducir el rango de valores de 256 a 59, además de obtener un descriptor invariante a la rotación. A la hora de generar el histograma LBP, todos los patrones uniformes con el mismo número de ceros/unos consecutivos se cuentan dentro de una misma clase, por ejemplo, 01110000 y 00111000 se contabilizarían como el mismo patrón. De esta forma, el histograma LBP se reduce a 9 clases más una décima para los patrones no uniformes.

Finalmente, para construir el descriptor de un superpíxel determinado, se normalizan y concatenan los histogramas LAB y el LBP dando lugar a un vector de 202 posiciones para cada segmento.

4.2 Mecanismos de descripción basados en características locales

A continuación, se exponen los descriptores locales considerados en esta fase del proyecto.

4.2.1 SIFT

Este algoritmo de detección y descripción utiliza una estrategia de filtro en cascada, en el cual las operaciones más complejas se aplican en aquellas regiones de la imagen que cumplen una serie de requisitos previos [13]. Las etapas principales del proceso se dividen en: detección de picos en el espacio de escala, localización de puntos característicos (*keypoints*), asignación de la orientación de cada *keypoint* y descripción del píxel en función de los gradientes locales de la imagen en el área circundante.

En su función como detector, SIFT usa una diferencia de gaussianas (Difference of Gaussians (DoG)) de la imagen como aproximación del laplaciano de la gaussiana. La DoG detecta los puntos más relevantes (esquinas, bordes, puntos, etc.) buscando puntos de máximos y mínimos locales en la pirámide 3D de la DoG. Posteriormente, en función de la escala del punto detectado, se define el tamaño del filtro gaussiano correspondiente para calcular gradientes de magnitud y orientación entre diferentes píxeles de vecindad del punto significativo. Con los vectores de orientación se genera un histograma, del cual la componente predominante se considerará la orientación del punto. Una vez asignadas la posición, escala y orientación de cada punto, se calcula un descriptor para cada región. Para ello, se define una ventana de 16x16 píxeles que a su vez se divide en 16 bloques de 4x4. Para cada bloque, se crea un histograma de gradientes de 8 conjuntos. De esta forma, se obtiene un vector de características de 128 dimensiones ($4 \times 4 \times 8$).

4.2.2 BRIEF

Este algoritmo de descripción realiza un suavizado de la imagen mediante un filtro gaussiano. Posteriormente, genera un vector binario en función de un test τ definido de la siguiente manera:

$$\tau(p;x,y) = \begin{cases} 1 & si \quad p(x) < p(y) \\ 0 & si \quad p(x) \geq p(y) \end{cases} \tag{4.1}$$

donde $p(x)$ denote el valor de intensidad del píxel x. El proceso se inicia considerando una región de la imagen y seleccionando un conjunto $n(x,y)$ de parejas de píxeles (hasta 512) [15]. Posteriormente, realiza una serie de comparaciones de intensidad entre estas parejas de puntos basadas en distribuciones uniformes, gaussianas y sistemas de coordenadas polares. Con ello compone un vector binario de n dimensiones.

4.2.3 BRISK

Del mismo modo que SIFT, BRISK también efectúa una búsqueda de puntos significativos en el espacio de escala, aunque en este caso se desarrolla mediante una descomposición piramidal en octavas e intra-octavas, realizando muestreos parciales

de la imagen de forma sucesiva [38]. A continuación, se aplica el detector Features from Accelerated Segment Test (FAST) a cada una de estas octavas para identificar las regiones de interés. Posteriormente, se efectúa la descripción de cada uno de estos puntos, la cual proviene de los valores resultantes de una serie de comparaciones de intensidad efectuadas en la vecindad del punto, de una forma similar a BRIEF pero asumiendo un patrón circular. Además, se determina la dirección característica de cada punto de manera que se obtiene invarianza a la rotación, dotándolo así de una mayor robustez. En última instancia, se obtiene un vector de 512 bits que describe cada punto clave detectado.

4.2.4 FREAK

Del mismo modo que BRISK, la motivación en el desarrollo de este algoritmo de detección y descripción es el ahorro en coste computacional. La aproximación de esta estrategia se basa en el procesamiento de la luz que lleva a cabo la retina humana, aplicando una analogía entre los fotoreceptores y los píxeles [39]. Para ello, utiliza un patrón circular de puntos equidistantes en círculos concéntricos para evaluar la región de interés en torno al punto característico. La densidad de los puntos disminuye exponencialmente así como aumenta la distancia al centro. El descriptor binario de 512 bits se construye mediante la comparación de los receptores utilizando para ello una diferencia de gaussianas. El resultado de estas comparaciones se filtra mediante un proceso de afinado, mediante el cual se va aumentando la resolución de los campos receptores. En su etapa de detección, FREAK utiliza FAST en el espacio de escala del mismo modo que BRISK.

4.2.5 ORB

Esta alternativa combina el algoritmo de detección FAST con el descriptor BRIEF añadiendo ciertas características a cada herramienta para dotarlo de una mayor robustez [16]. En primer lugar, implementa una componente de orientación a FAST. Esta cualidad se obtiene mediante la representación de la imagen en un espacio piramidal multiescala, compuesta por diferentes versiones de la imagen. Mediante la ejecución de FAST en cada uno de los niveles de la pirámide, se obtiene una descripción robusta frente a cambios de escala. A continuación, se determina la orientación del punto en función de cómo cambian los niveles de intensidad a su alrededor. Para ello emplea el centroide de la intensidad, definiendo un vector entre una esquina de la región y el centroide. Dicho vector se emplea para determinar la gradiente dominante. De esta forma se obtiene un parámetro de orientación robusto frente a rotaciones. Al final del proceso se obtiene un vector de descripción binario de 256 bits, rotado de acuerdo a la orientación dominante.

4.3 Mecanismos de descripción basados en redes convolucionales

Dentro de las estrategias de descripción planteadas, se consideró la opción de combinar las técnicas de segmentación con descriptores basados en redes neuronales convolucionales. Tal y como expone [10], dichas estrategias presentan una robustez

muy alta en tareas de emparejamiento de imágenes, aunque éstas capturen escenas sometidas a cambios de apariencia severos. Con estas características, se plantea el uso de esta herramienta en el presente proyecto en comparación con los descriptores locales anteriormente expuestos.

El procedimiento consiste en extraer la región rectangular circundante mínima (*bounding box*, en inglés) correspondiente a cada superpíxel, utilizar esta región como entrada a la red convolucional para obtener el descriptor. Posteriormente, el descriptor de cada segmento de la escena se compara con los descriptores del resto de escenas de la secuencia mediante la función de emparejamiento y se obtiene una medida de coincidencia que determinará si nos encontramos frente a una escena previamente capturada o no.

Para este proyecto se ha hecho uso de dos arquitecturas: AlexNet [4] y ResNet [30], ambas implementadas en el framework *pytorch*. En el caso de la primera, tal y como se ha comentado en el capítulo 2, se dispone de 60 millones de parámetros y 650.000 neuronas. Se compone de cinco capas convolucionales y tres capas de clasificación o *fully-connected layers*, tal y como puede verse en la Fig. 4.3. Mediante una función de *hook*, se extraen las características resultantes de la quinta capa convolucional. La salida de esta capa se compone de 256 matrices de tamaño 13×13. Para generar un vector de descripción unidimensional de 256 posiciones, se han valorado dos alternativas: la extracción de los valores máximos de cada matriz 13×13 y la media de sus valores. Los resultados obtenidos de ambas soluciones se exponen al final de este capítulo.

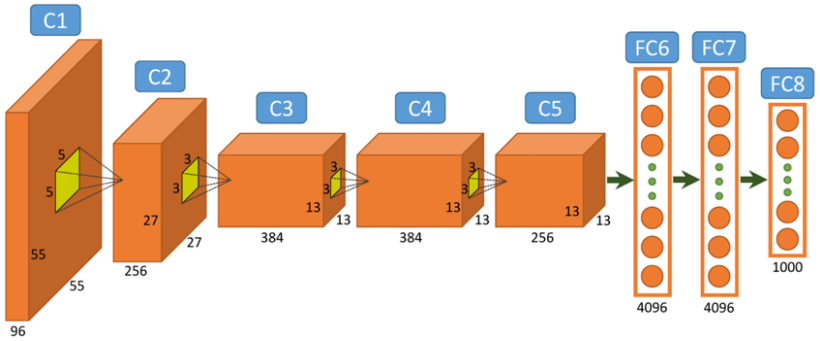


Figura 4.3: Diagrama de la arquitectura AlexNet [4]

En el caso de la red Residual Networks (ResNet), su implementación se caracteriza por la inclusión de conexiones directas con capas posteriores [30]. De esta forma, una capa determinada podría recibir como entrada, la entrada y la salida de la capa anterior, tal como se muestra en la Fig. 4.4. Esta modificación responde a dos motivaciones: reducir el problema de desvanecimiento de gradientes y otorgar al modelo la capacidad de que las capas superiores tengan un rendimiento igual o mejor que las capas inferiores. De esta forma, se consigue corregir en gran medida la problemática existente con las redes más profundas. Para este proyecto, concretamente, se ha empleado ResNet50, la cual se compone de un total de 50 capas. Al igual que con AlexNet, se han extraído las características resultantes de la capa *average pool*. En este caso, dicha capa ya nos proporciona un vector unidimensional de 2048 posiciones.

Estas dos arquitecturas se han configurado cargando modelos preentrenados. Este

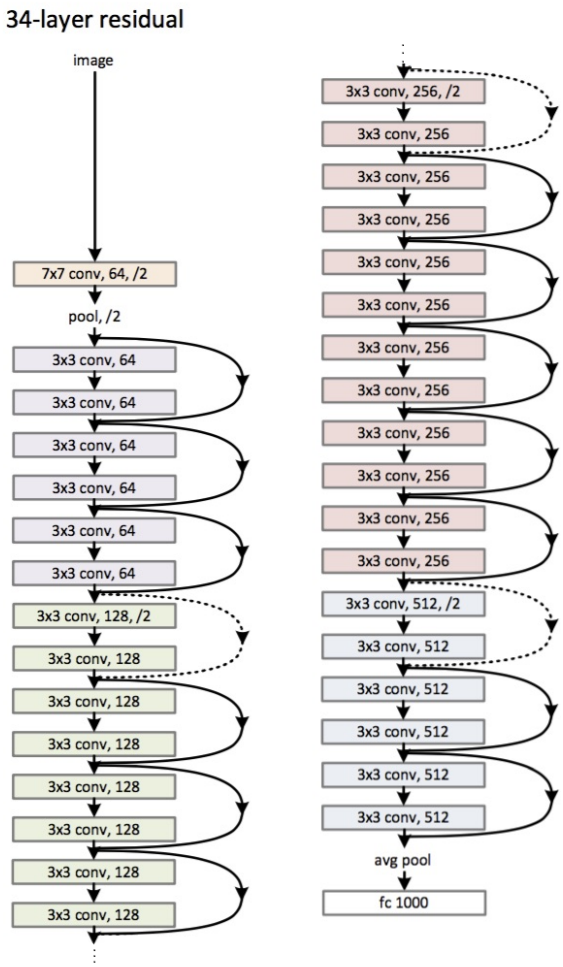


Figura 4.4: Diagrama de 34 capas de ejemplo de la arquitectura ResNet50 [5]

procedimiento consiste en optimizar los pesos de la red en base a los resultados obtenidos tras procesar una larga colección de imágenes, de manera que los parámetros convergen para generar una identificación correcta de la escena representada en la imagen. Cada una de las arquitecturas se ha validado mediante dos modelos: ImageNet, resultante de procesar una colección de imágenes genéricas (paisajes, objetos, vehículos, etc.) y Places365, el cual se obtiene tras entrenar la red mediante una larga colección de únicamente escenas, ya sean de interior, urbanas o de naturaleza. El objetivo es evaluar si el hecho de usar un modelo preentrenado específicamente con escenas mejora en comparación a uno genérico.

4.4 Metodología

Los mecanismos de descripción descritos se validan mediante un procedimiento consistente en detectar el número de emparejamientos (*matches*) entre tres imágenes. Este

trío de imágenes está formado por tres fotogramas concretos de una secuencia en la que hay uno o más cierres de bucle. De esta forma, se contrastan los descriptores obtenidos tras procesar una imagen de la secuencia, la imagen correspondiente al fotograma anterior y la escena candidata a cierre de bucle con la imagen consultada, tal y como se muestra en la Fig. 4.5. Con esta configuración, se pretende comprobar la capacidad de la estrategia propuesta para la tarea de reconocimiento de escenas y su robustez frente a cambios de escala y orientación, provocados por el propio desplazamiento de la cámara en el entorno.

Las imágenes forman parte de las secuencias KITTI00 y KITTI05. De cada una de ellas se han seleccionado dos recorridos, y de cada recorrido un trío de imágenes en los términos previamente expuestos.

(a)

(b)

(c)

Figura 4.5: Secuencia de tres imágenes seleccionadas del dataset KITTI00 compuesta por: (a) imagen consultada, (b) fotograma anterior y (c) imagen candidata.

En el caso de los descriptores basados en características locales, precisan de un

listado de puntos característicos (*keypoints*) de los cuales se extrae la información. Estos puntos incorporan una serie de datos, entre los cuales, aparte de su ubicación, se encuentra una medida del tamaño que ocupan. Tal y como se ha comentado anteriormente, estos puntos característicos se extraen en el proceso de segmentación. De cada segmento, se calcula su caja circundante o *bounding box*. Con esta información, se asignan las coordenadas de imagen del centro de la caja circundante como ubicación del punto y el mayor de los dos lados como tamaño.

Figura 4.6: Ejemplo de keypoints generados a partir de la segmentación de la imagen.

Por otro lado, para operar con los descriptores basados en CNN, la operación consiste en realizar la descripción del segmento de imagen correspondiente a cada *bounding box*, redimensionando previamente el cuadrante correspondiente al superpíxel.

Una vez generados los descriptores, se calcula la distancia entre ellos, ya sea euclídea o de *Hamming*, en función de si los descriptores se componen de vectores de punto flotante o binarios, respectivamente. Dicha distancia se obtiene comparando los descriptores mediante un árbol KD, en el caso de los vectores de punto flotante. Para los binarios, se realiza mediante una búsqueda exhaustiva o de fuerza bruta. Con este procedimiento se obtienen dos candidatos de emparejamiento para cada punto significativo. Estos dos candidatos se comparan por sus distancias en términos relativos, de forma que si el mejor de ellos representa una distancia menor al segundo candidato en un porcentaje suficiente, se considera ese emparejamiento como bueno. Para este proyecto se han probado todas las estrategias aplicando un ratio de 0.7 y 0.8. Tras este primer cribado, se aplica la verificación mediante un conjunto de valores de referencia específicos para cada pareja de imágenes. El proceso se ilustra en el Alg. 4.

Algorithm 4: Selección de emparejamientos veraces

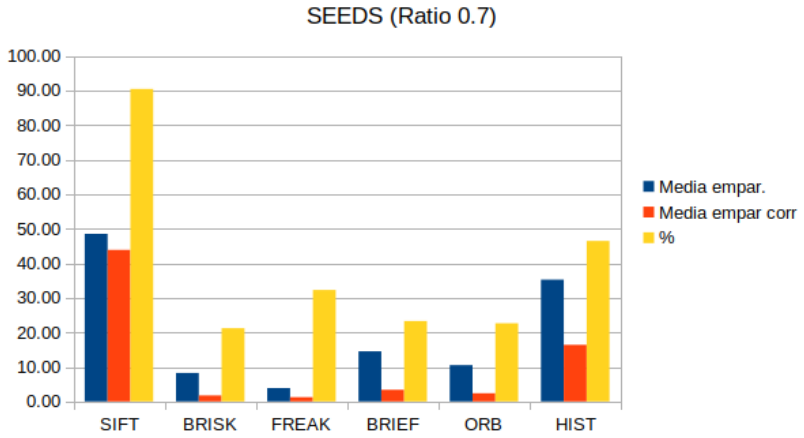
```
1 m, n = matcher(descriptores imagen 1, descriptores imagen 2)
  /*                                                          */
2 for cada pareja de candidatos m, n do
3     if m.distancia <n.distancia * ratio then
4         if m pertence a lista de valores de referencia then
5             añadir m a lista de emparejamientos
```

Para disponer de unos valores de referencia (*ground truth*) del emparejamiento, se ha definido de forma manual la relación entre los segmentos generados en cada

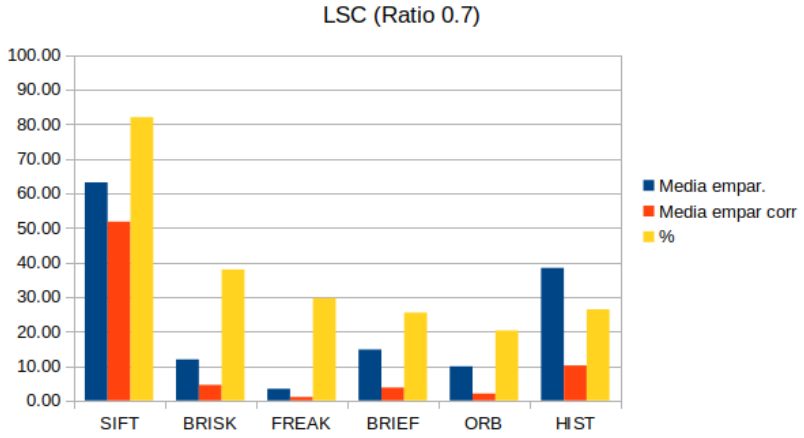
escena para cada una de las estrategias de segmentación elegidas en el capítulo 3. Con ello, se ha verificado si los segmentos emparejados representan realmente la misma fracción de la escena. Para dotar al proceso de evaluación de cierto grado de tolerancia, se han considerado como correctos aquellos emparejamientos entre un segmento de la imagen consultada y el segmento correspondiente de la imagen candidata o cualquier segmento inmediatamente próximo en el plano imagen.

4.5 Resultados

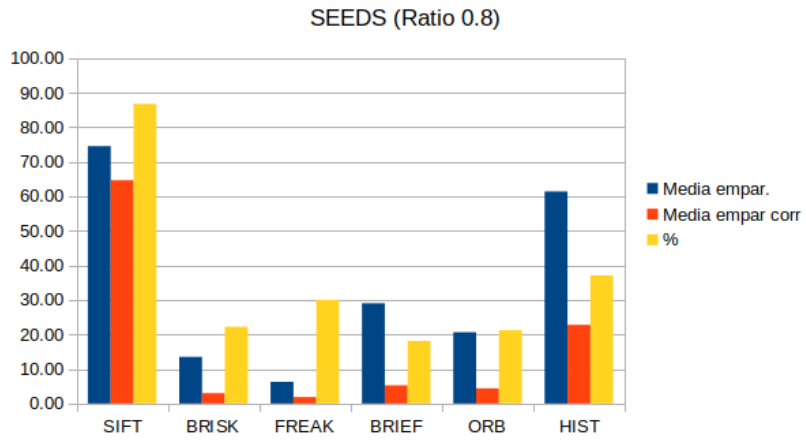
Las gráficas siguientes muestran en términos porcentuales la media de emparejamientos detectados por cada algoritmo, la media de emparejamientos correctos y el porcentaje de emparejamientos correctos sobre el total. A su vez, se exponen los resultados por separado tanto de los ratios empleados (de la figura 4.7a a 4.10b) como de las diferentes secuencias (de 4.11a a 4.14b). Los resultados obtenidos con el descriptor basado en histogramas se han identificado en las gráficas como 'HIST'.



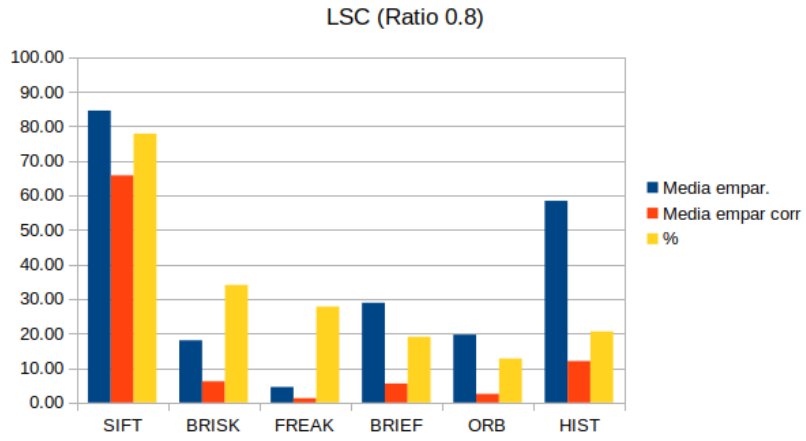
(a) Resultados obtenidos con SEEDS y descriptores locales aplicando ratio de 0.7



(b) Resultados obtenidos con LSC y descriptores locales aplicando ratio de 0.7



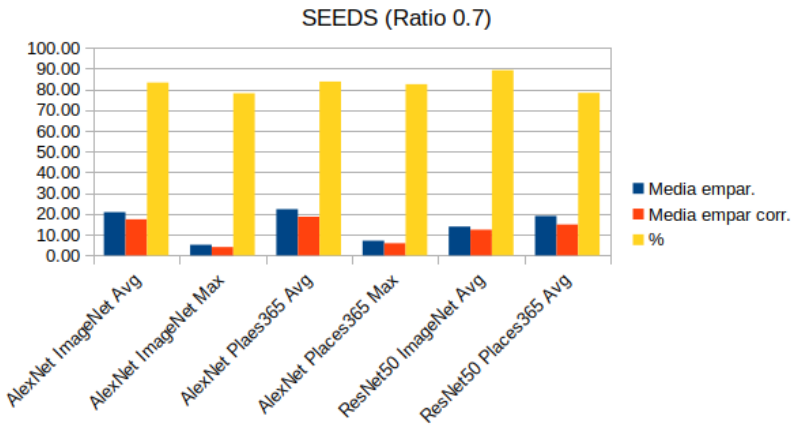
(a) Resultados obtenidos con SEEDS y descriptores locales aplicando ratio de 0.8



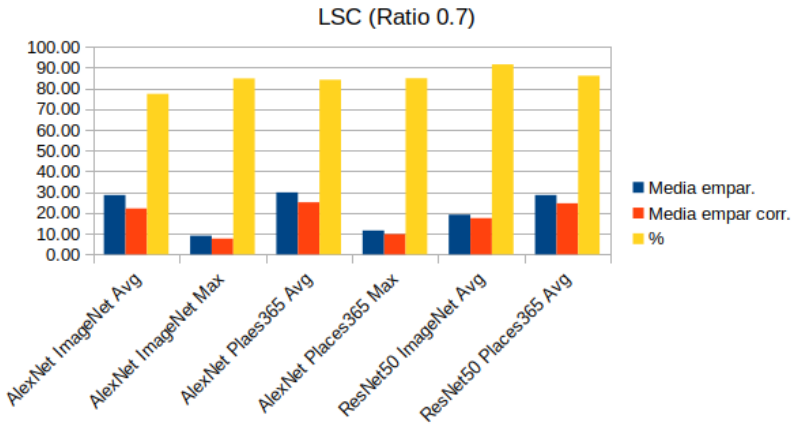
(b) Resultados obtenidos con LSC y descriptores locales aplicando ratio de 0.8

El primer detalle evidente observando las gráficas es que el descriptor local que ha dado mejores resultados para los dos algoritmos de segmentación ha sido SIFT. La diferencia tanto en términos de precisión como de número de emparejamientos es muy acusada con el resto de descriptores. Los resultados muestran también que el hecho de aumentar el ratio, siendo por tanto menos restrictivos a la hora de identificar un emparejamiento, no penaliza en exceso la precisión a pesar del aumento notable en el número de emparejamientos. No obstante, dicha penalización sí es algo más acusada empleando LSC como algoritmo de segmentación. Ocurre igual si observamos los resultados para cada secuencia, donde nuevamente SEEDS+SIFT desempeña un mejor papel en términos de precisión.

De los demás descriptores, es de destacar que BRISK parece arrojar los resultados más consistentes en términos de precisión. El descriptor basado en histogramas detecta un gran número medio de emparejamientos, especialmente en la secuencia KITTI05 y en combinación con SEEDS. Pero podemos observar que de esos emparejamientos solo un pequeño porcentaje son veraces, demostrando así una precisión muy baja. Más



(a) Resultados obtenidos con SEEDS y descriptores CNN aplicando ratio de 0.7

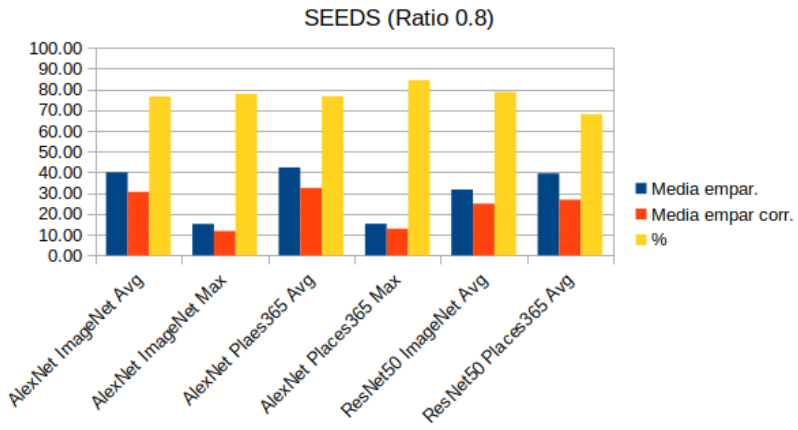


(b) Resultados obtenidos con LSC y descriptores CNN aplicando ratio de 0.7

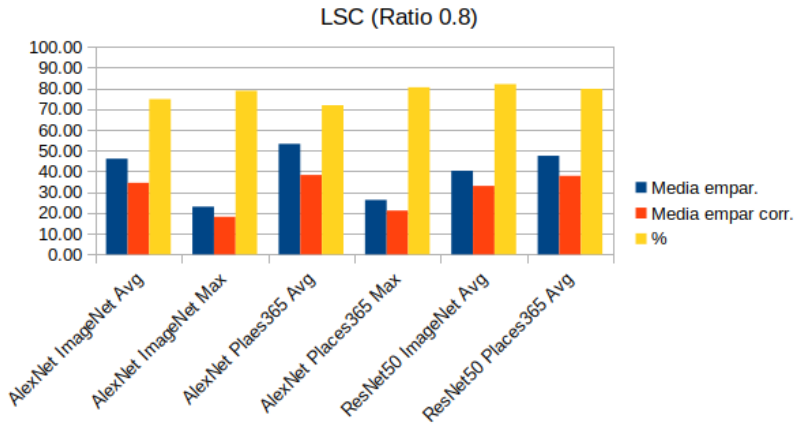
allá de eso, analizando los resultados en detalle para este descriptor, en muchos casos ni el 10 % de emparejamientos supera la criba de los valores de referencia.

Por otro lado, se ha realizado la misma comparativa con los descriptores basados en CNN. En general, es fácilmente perceptible a primera vista que el número de emparejamientos es mucho menor respecto a SIFT. En los casos más favorables en referencia a este aspecto para cada caso, apenas se supera el 50 % de emparejamientos para la estrategia CNN respecto a la tradicional. Otro hecho que puede apreciarse sin entrar en detalle en los datos, es que el aumento en el ratio tiene un impacto más significativo sobre el número de emparejamientos, así como en el porcentaje de emparejamientos veraces. Valorando los resultados catalogados por dataset, no se observan grandes diferencias entre los algoritmos ejecutados sobre KITTI00 y KITTI05.

Si comparamos el rendimiento basándonos en el algoritmo de segmentación empleado, se observa como de nuevo utilizando LSC se obtiene un mayor número de emparejamientos veraces, aunque su proporción respecto a los emparejamientos totales es menos favorable que utilizando SEEDS, teniendo éste mejores porcentajes de acierto.



(a) Resultados obtenidos con SEEDS y descriptores CNN aplicando ratio de 0.8



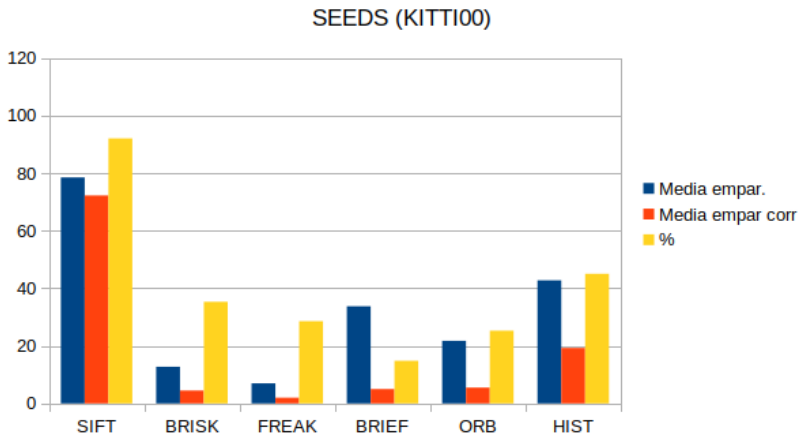
(b) Resultados obtenidos con LSC y descriptores CNN aplicando ratio de 0.8

Otro detalle interesante que puede derivarse de los resultados obtenidos es que en general da mejor rendimiento aplicar la media sobre los componentes de la capa seleccionada de la red neuronal, en lugar de seleccionar su máximo. Esto puede apreciarse comparando las diferentes variantes de la arquitectura AlexNet consideradas.

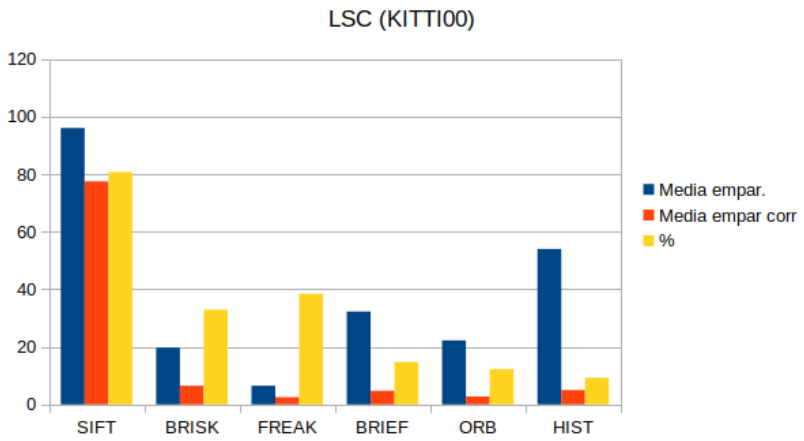
Del mismo modo, los experimentos demuestran también una mejor respuesta utilizando el modelo Places365, entrenado específicamente para el reconocimiento de escenas, sobre ImageNet.

4.6 Discusión

Valorando los resultados expuestos, se dispone de recursos suficientes para definir un conjunto de estrategias que podrían ser útiles en experimentos más complejos de reconocimiento de escenas. Los datos que justifican los argumentos expuestos en este apartado se exponen en las Tablas 4.1 y 4.2. Dichas tablas recogen, respectivamente, los resultados de cada algoritmo de segmentación con la mejor alternativa de descriptores locales y con la mejor combinación de las estrategias basadas en redes convolucionales.



(a) Resultados obtenidos con SEEDS y descriptores locales sobre KITTI00

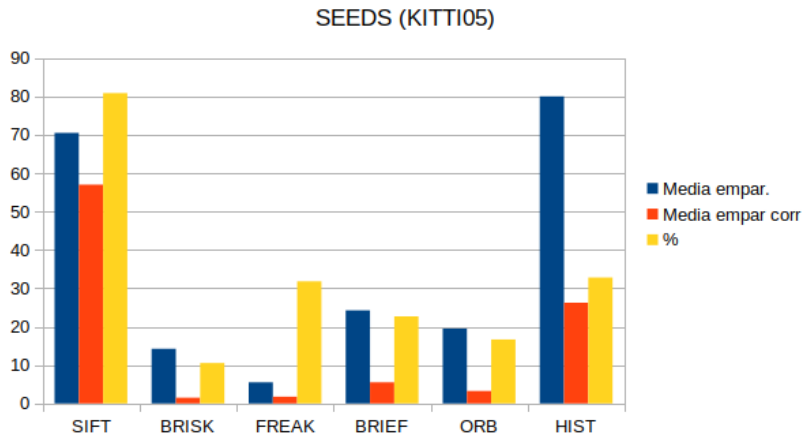


(b) Resultados obtenidos con LSC y descriptores locales sobre KITTI00

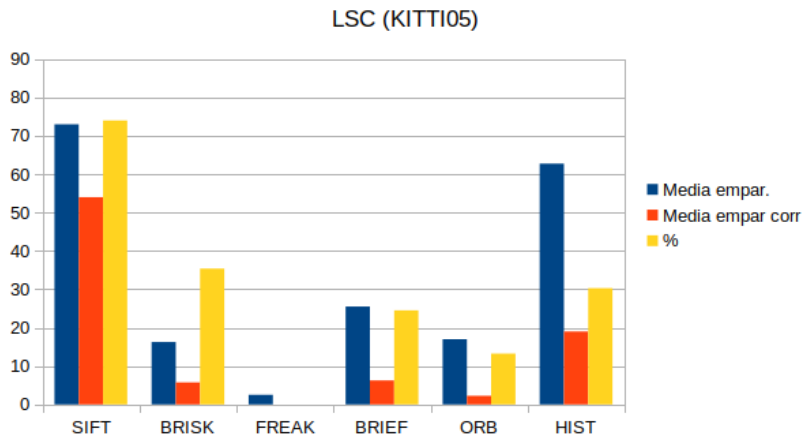
En cada caso, se muestra el número total de emparejamientos, el número de emparejamientos correctos y el porcentaje de emparejamientos correctos sobre los totales, tanto entre la imagen consultada y el fotograma anterior, como con la imagen candidata de cierre de bucle. La última fila corresponde a la media de los valores de cada columna.

En relación a los descriptores locales, todos los resultados apuntan a que SIFT es la mejor alternativa para considerarla en tareas de reconocimiento visual de escenas. El rendimiento observado por este descriptor concuerda con las conclusiones extraídas por Ikenaga et al. [40], en las cuales se expone una gran precisión en el número de emparejamientos correctos en comparación con las alternativas más compactas, como podrían ser los descriptores binarios. En cuanto a qué algoritmo de segmentación responde mejor en combinación con este descriptor, se ha observado que LSC da un mayor número de emparejamientos a costa de una reducción en la precisión respecto a SEEDS. Los resultados obtenidos de cada algoritmo de segmentación con el descriptor SIFT se recogen en la Tabla 4.1.

En cuanto a los descriptores CNN, en general se ha obtenido un mejor resultado con la combinación de la arquitectura AlexNet, el modelo Places365 y aplicando la media sobre la capa seleccionada. No obstante, es cierto que ResNet+Places365 en

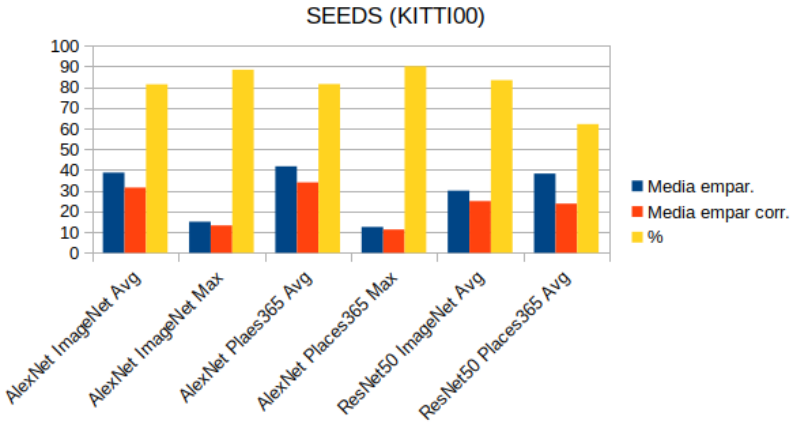


(a) Resultados obtenidos con SEEDS y descriptores locales sobre KITTI05

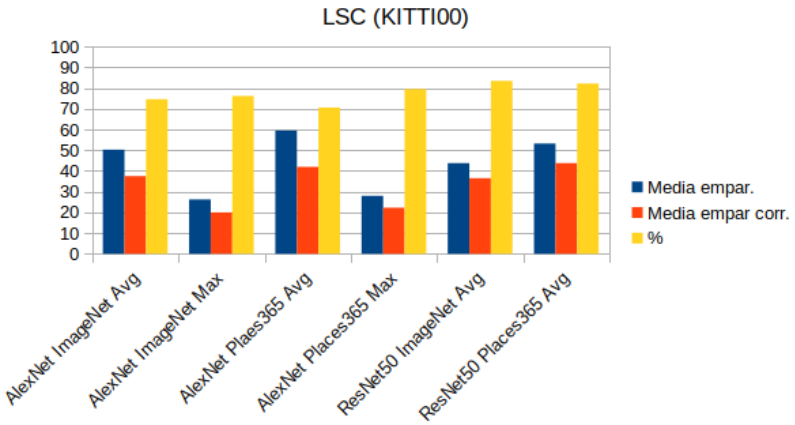


(b) Resultados obtenidos con LSC y descriptores locales sobre KITTI05

combinación con LSC presenta también un buen desempeño. Para seleccionar una de las dos alternativas de cara a validarla frente a experimentos de reconocimiento visual de escenas, se han comparado porcentajes totales de precisión para cada estrategia. En el cómputo global, se ha obtenido un mejor rendimiento para la opción basada en AlexNet. Se pueden encontrar los resultados obtenidos en la Tabla 4.2.



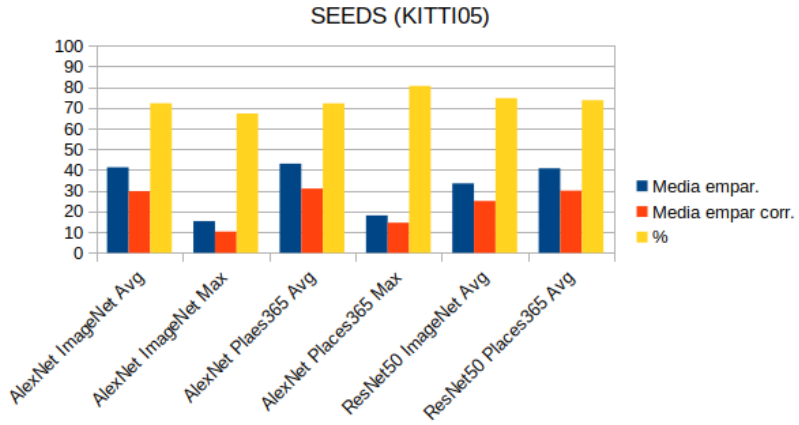
(a) Resultados obtenidos con SEEDS y descriptores CNN sobre KITTI00



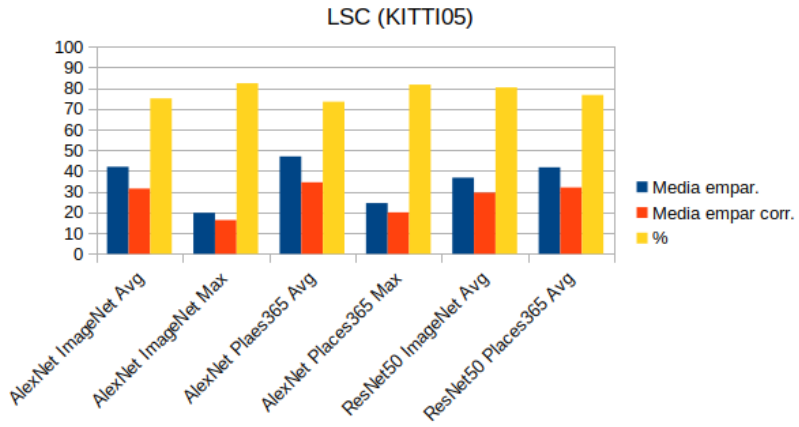
(b) Resultados obtenidos con LSC y descriptores CNN sobre KITTI00

			Segmentación	**SEEDS**			**LSC**		
			Descriptor	**SIFT**			**SIFT**		
Ratio	**Dataset**	**Sec.**	**Imagen**	**Empar.**	**Corr.**	**%**	**Empar.**	**Corr.**	**%**
0.8	00	1	Imagen ant	107	105	98.13	116	90	77.59
			Candidata	37	26	70.27	35	21	60.00
		2	Imagen ant	88	85	96.59	128	115	89.84
			Candidata	82	73	89.02	105	84	80.00
	05	1	Imagen ant	110	104	94.55	121	92	76.03
			Candidata	30	10	33.33	22	12	54.55
		2	Imagen ant	103	99	96.12	120	94	78.33
			Candidata	39	15	38.46	29	18	62.07
			Media	79.57	71.71	**82.57**	92.43	72.57	73.76

Tabla 4.1: Resultados más favorables empleando descriptores locales



(a) Resultados obtenidos con SEEDS y descriptores CNN sobre KITTI05



(b) Resultados obtenidos con LSC y descriptores CNN sobre KITTI05

			Segmentación	**SEEDS**			**LSC**		
			Arquitectura	**AlexNet**			**AlexNet**		
			Modelo	**Places365**			**Places365**		
			Operación	**Avg**			**Avg**		
Ratio	**Dataset**	**Sec.**	**Imagen**	**Empar.**	**Corr.**	**%**	**Empar.**	**Corr.**	**%**
0.8	00	1	Imagen ant	73	62	84.93	93	74	79.57
			Candidata	18	10	55.56	25	11	44.00
		2	Imagen ant	47	42	89.36	77	61	79.22
			Candidata	29	22	75.86	43	22	51.16
	05	1	Imagen ant	52	47	90.38	66	54	81.82
			Candidata	26	13	50.00	25	10	40.00
		2	Imagen ant	60	50	83.33	77	63	81.82
			Candidata	34	14	41.18	20	11	55.00
			Media	43.57	35.14	**75.63**	58.00	42.14	65.37

Tabla 4.2: Resultados más favorables empleando descriptores basados en CNN

CAPÍTULO 5

SISTEMA DE RECONOCIMIENTO DE ESCENAS BASADO EN SUPERPÍXELES

En el siguiente capítulo, se procede a evaluar las estrategias seleccionadas en capítulos anteriores frente a secuencias más extensas en las que se repiten escenas y las imágenes correspondientes presentan cambios de enfoque, rotación, iluminación y/o efectos de fenómenos meteorológicos. Se exponen las diferentes herramientas y técnicas empleadas para llevar a cabo este proceso, así como los resultados obtenidos.

5.1 Metodología

En procedimientos de reconocimiento de escenas, es habitual el uso de métricas basadas en los parámetros de precisión y sensibilidad (habitualmente referidos por sus nombres en inglés, *precision* y *recall*). Estos parámetros se obtienen comparando los resultados obtenidos frente a unos valores de referencia (*ground truth*). La precisión (P) se define como el número de emparejamientos detectados correctos (denominados positivo real o TP, por su nombre en inglés, *true positive*) respecto del total de emparejamientos identificados por el proceso (es decir, el total de predicciones positivas). Es decir, de las veces que el sistema detecta que dos imágenes corresponden a la misma escena, cuántas corresponden realmente a una imagen previamente procesada. Por tanto, una vez realizada la detección, la precisión se calcula como el cociente entre los positivos reales y el total de emparejamientos identificados, los cuales se compone de la suma de los positivos reales y los falsos positivos (*FP*).

$$P = \frac{TP}{TP + FP}$$

La sensibilidad (R) se define como la proporción de emparejamientos detectados correctos dentro del total de emparejamientos existentes. Dicho de otro modo, mide la capacidad de la estrategia para detectar las veces que se presencia una escena en más de una ocasión. El total de emparejamientos correctos existentes se obtiene a partir de

la suma de positivos reales y falsos negativos (*FN*), y, en consecuencia la sensibilidad viene dada por:

$$R = \frac{TP}{TP + FN}$$

El objetivo es maximizar la sensibilidad garantizando el 100 % de precisión, debido a que un falso positivo es crucial en un sistema de localización. La relación entre estas dos variables se representa mediante las denominadas curvas de precision-recall (PR). Este tipo de curvas representan de forma gráfica la evolución de la precisión frente a la sensibilidad en función de una variable común que determine el desempeño de la estrategia. En el caso de este proyecto, esta variable común viene definida por el valor de umbralización empleado para determinar a partir de qué valor de similitud entre escenas se considera ambas como la misma escena.

Para determinar el valor de umbralización que aporta el rendimiento óptimo del proceso, se utiliza la media harmónica que combina la precisión y la sensibilidad [41], denominado valor-F, el cual nos aporta un valor ponderado entre estos dos valores:

$$F - Score = 2\frac{PR}{P + R}$$

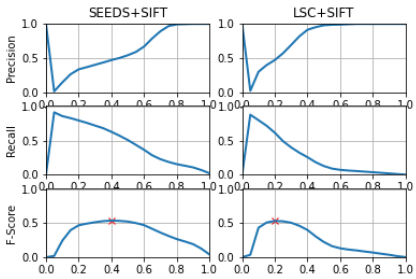
Otra medida útil para comparar el rendimiento de las diferentes estrategias que se plantean en este proyecto consiste en calcular el área bajo la curva (Area under the Curve (AuC)) de la gráfica PR, obteniendo el valor de precisión media del proceso [42].

Para ejecutar los experimentos se han escogido una serie de *datasets* públicos: KITTI05 [43], Lip6 Indoor [44], Nordland [45], además de UIB Small Loop y UIB Large Loop [9]. Cada uno de estos *datasets* presenta unas características concretas que servirán para medir la capacidad de las estrategias de reconocimiento de escenas. En el caso de KITTI05, se compone de imágenes tomadas en un recorrido exterior urbano bien iluminado. Como su nombre indica, Lip6 Indoor muestra un recorrido por el interior de un edificio. Nordland, a diferencia del resto, no incluye cierres de bucle sino que se compone de dos largas secuencias de un recorrido exterior y ambas muestran las mismas escenas: una en verano y la otra en invierno. UIB Small Loop ha sido tomada a pie en exterior y, además de diversos cierres de bucle, presenta cambios de orientación y rotación en la captura de las imágenes a lo largo del recorrido. Finalmente, UIB Large Loop es muy similar a UIB Small Loop pero se compone de más imágenes.

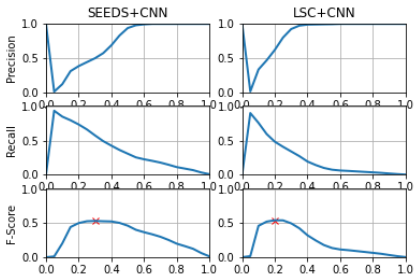
Estas secuencias disponen de unos valores de referencia específicos, los cuales se constituyen mediante una matriz cuadrada con tantas filas y columnas como fotogramas tiene la secuencia. Típicamente, si en la posición (i, j) de esta matriz se encuentra el valor 1 es porque se considera que la imagen i cierra bucle con la imagen j.

5.2 Resultados

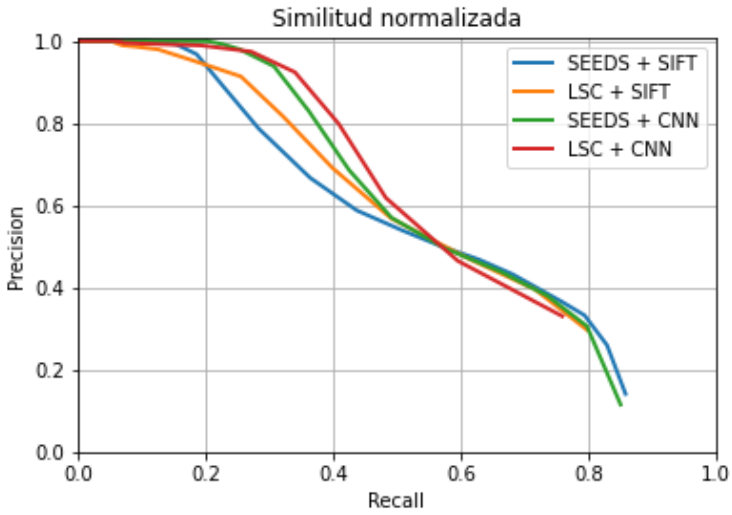
Con el conjunto de métricas expuesto, se van a analizar los resultados obtenidos para cada secuencia aplicando las estrategias descritas en los capítulos 3 y 4. En concreto, se van a probar los algoritmos de segmentación SEEDS y LSC combinados con los descriptores SIFT y el descriptor CNN compuesto por la arquitectura AlexNet y el modelo Places365, dando lugar a un total de cuatro estrategias.



(a) Gráficas P, R y Valor-F para SEEDS+SIFT y LSC+SIFT



(b) Gráficas P, R y Valor-F para SEEDS+CNN y LSC+CNN



(c) Curvas PR para cada combinación

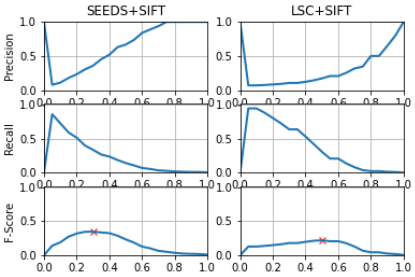
Figura 5.1: Gráficas correspondientes a los experimentos sobre el dataset KITTI05

5.2.1 KITTI05

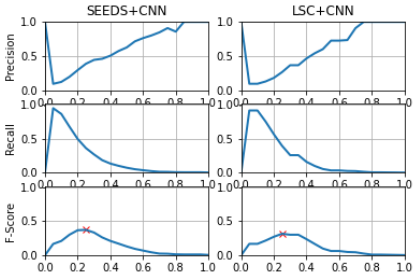
Si observamos la Fig. 5.1c, en las curvas PR de cada estrategia podemos apreciar que siguen una tendencia similar y en general exhiben un comportamiento muy parecido. No obstante, los descriptores basados en CNN muestran un mejor rendimiento, el cual es más evidente observando los valores de AuC recogidos en la Tabla 5.3, dónde se aprecia claramente que la combinación SEEDS+CNN es la más favorable para esta secuencia en términos generales. Las gráficas 5.1a y 5.1b nos sirven para determinar que esa mejora en el rendimiento viene dada por un notable incremento en la precisión de los emparejamientos con una ligera disminución en la sensibilidad por parte de las estrategias CNN. Es muy destacable el rendimiento obtenido con la combinación LSC+CNN, alcanzando un Valor-F de 0.5425 con una umbralización a 0.2.

5.2.2 Lip6 Indoor

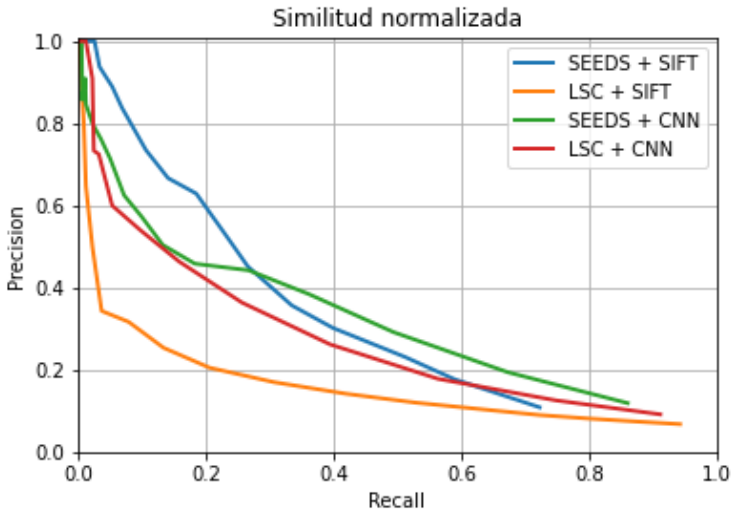
Por otro lado, para la secuencia de interiores Lip6, las tendencias cambian notablemente, ya que en la Fig. 5.2c podemos ver como disminuye rápidamente la precisión así



(a) Gráficas P, R y Valor-F para SEEDS+SIFT y LSC+SIFT



(b) Gráficas P, R y Valor-F para SEEDS+CNN y LSC+CNN



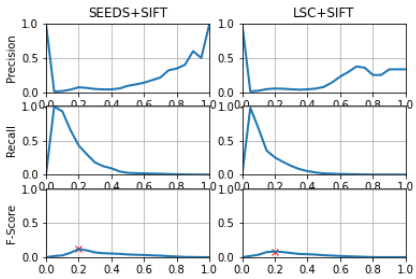
(c) Curvas PR para cada combinación

Figura 5.2: Gráficas correspondientes a los experimentos sobre el dataset Lip6 Indoor

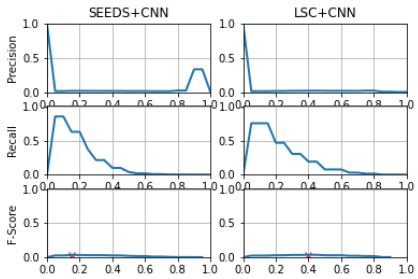
como aumenta la sensibilidad. Esto provoca que los valores de AuC sean mucho menores, siendo nuevamente la estrategia SEEDS+CNN la que aporta un mejor resultado (0.31), muy lejano de lo obtenido en la anterior secuencia. Observando las gráficas 5.2a y 5.2b podemos apreciar una importante mejora en la precisión al aplicar los descriptores CNN sobre la segmentación realizada con LSC, frente a los resultados obtenidos con SIFT. Es de destacar también, que en este caso, el segundo mejor resultado no proviene de la otra estrategia con descriptores CNN, si no de la combinación SEEDS+SIFT. El valor de umbralización óptimo se encuentra en 0.25 para SEEDS+CNN, dando lugar a un Valor-F de 0.37.

5.2.3 Nordland

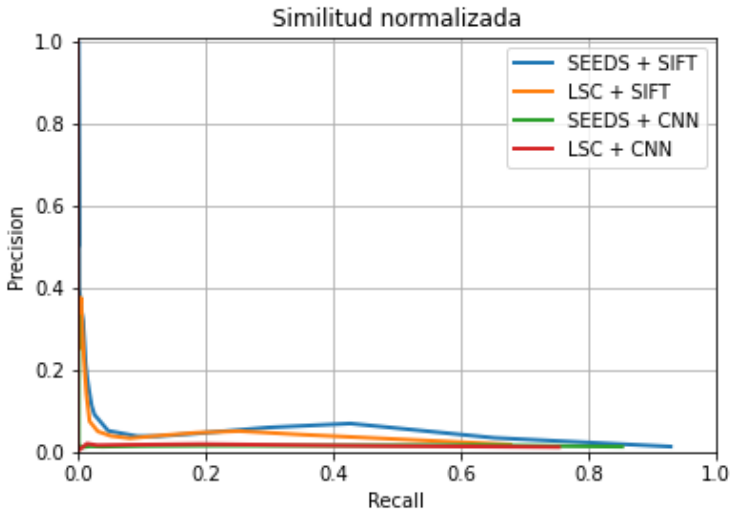
En el caso de Nordland, solo observando la gráfica 5.3c, podemos apreciar que se trata del experimento más ambicioso para las estrategias de reconocimiento de escenas planteadas, quedando fuera de su capacidad de detección, al tratarse de entornos con cambios de apariencia muy drásticos entre las capturas de la secuencia de verano e



(a) Gráficas P, R y Valor-F para SEEDS+SIFT y LSC+SIFT



(b) Gráficas P, R y Valor-F para SEEDS+CNN y LSC+CNN



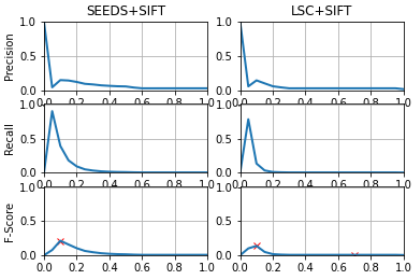
(c) Curvas PR para cada combinación

Figura 5.3: Gráficas correspondientes a los experimentos sobre el dataset Nordland

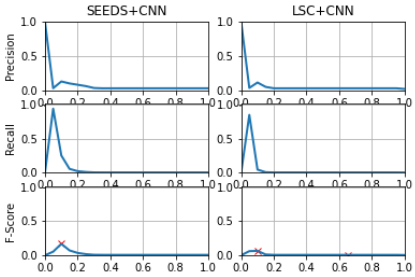
invierno. La gráfica muestra un rendimiento pobre, con un valor de precisión muy próximo a 0 para todo el rango de valores de umbrales. En las gráficas 5.3a y 5.3b se ve claramente que los resultados en términos de precisión son muy bajos, especialmente aplicando descriptores CNN. En este caso, cabe mencionar que la estrategia con mejor rendimiento corresponde a la compuesta por la segmentación SEEDS con el descriptor SIFT, pero en cualquier caso no alcanza siquiera 0.05. El valor óptimo de umbralización se encuentra en 0.2.

5.2.4 UIB large

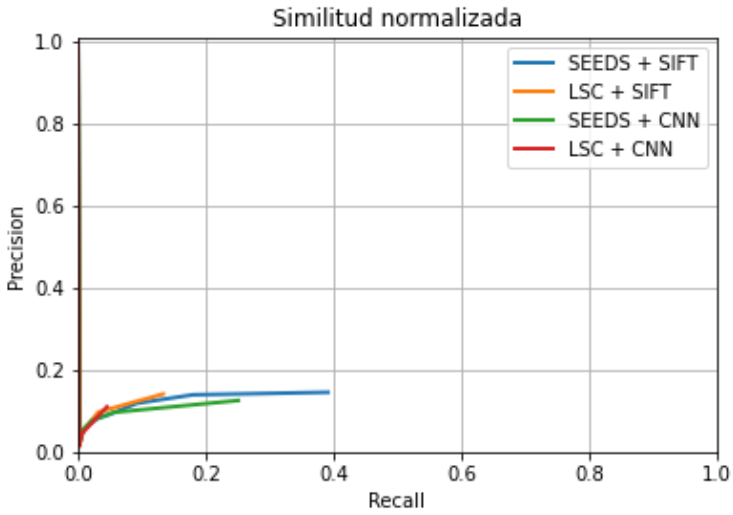
En las gráficas 5.4 correspondientes a la secuencia UIB Large Loop se aprecia un rendimiento muy bajo nuevamente, dado que los valores de precisión obtenidos son notablemente bajos. En este caso, es especialmente llamativo que además de una baja precisión, la cantidad de emparejamientos detectados es muy baja, como puede apreciarse en las gráficas 5.4a y 5.4b, donde vemos que la curva de sensibilidad se mantiene próxima a 0 para casi todo el rango de valores umbrales muestreados. Del



(a) Gráficas P, R y Valor-F para SEEDS+SIFT y LSC+SIFT



(b) Gráficas P, R y Valor-F para SEEDS+CNN y LSC+CNN



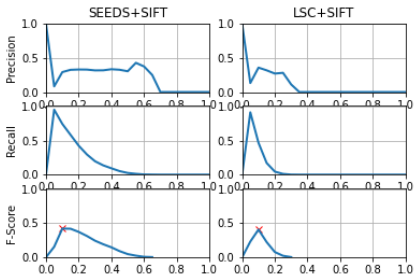
(c) Curvas PR para cada combinación

Figura 5.4: Gráficas correspondientes a los experimentos sobre el dataset UIB Large

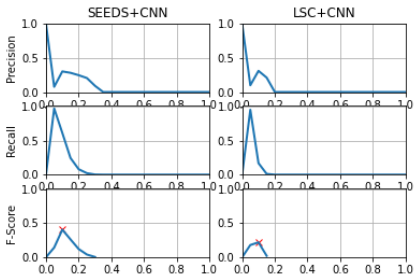
mismo modo, la baja capacidad de detección se refleja en que los valores de sensibilidad obtenidos no superan el 0.4, como podemos ver en la curva PR de la figura 5.4c. La estrategia con un mejor desempeño general (nuevamente SEEDS+SIFT), presenta un valor de AuC de 0.05. Por otro lado, el rendimiento óptimo se ha obtenido mediante la estrategia SEEDS+CNN con un valor de umbralización de 0.2, aportando un Valor-F de 0.1676.

5.2.5 UIB small

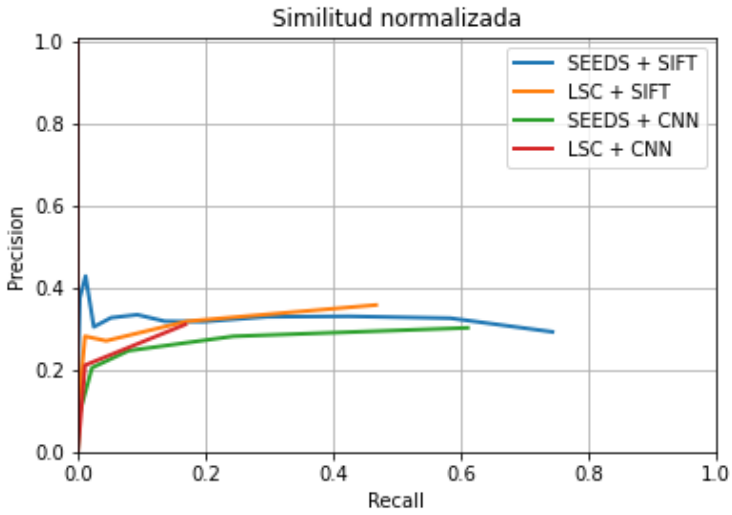
Finalmente, en las gráficas 5.5 podemos observar que los valores para UIB Small Loop mejoran ligeramente respecto a la versión extensa de la secuencia, pero siguen muy lejos de un resultado deseable para una estrategia de reconocimiento de escenas. Nuevamente, las curvas expuestas en la gráfica 5.5c transcurren por debajo de un nivel de precisión de 0.5 para todo el rango de valores, obteniendo unos valores de AuC notablemente bajos. En este caso, vuelve a destacar la estrategia SEEDS+SIFT, con un valor de 0.24. En las gráficas 5.5a y 5.5b podemos ver que esta combinación obtiene



(a) Gráficas P, R y Valor-F para SEEDS+SIFT y LSC+SIFT



(b) Gráficas P, R y Valor-F para SEEDS+CNN y LSC+CNN



(c) Curvas PR para cada combinación

Figura 5.5: Gráficas correspondientes a los experimentos sobre el dataset UIB Small

una mayor precisión y un mayor número de emparejamientos que la variante con LSC y que las estrategias basadas en descriptores CNN, las cuales demuestran una precisión menor que la obtenida con el descriptor SIFT. Las gráficas PR presentan un recorrido más largo, indicando la mejora en sensibilidad respecto a la secuencia UIB Large Loop. En este caso, el valor óptimo de umbralización se sitúa en 0.1, generando un Valor-F de 0.42, el segundo más alto después del obtenido con KITTI05.

5.3 Discusión

Tras lo observado en los resultados de las diferentes secuencias, se procede a realizar una evaluación general, analizando las posibles causas de la variabilidad en el desempeño de cada estrategia en función de las características particulares de cada secuencia. En primer lugar, como ya se ha expuesto anteriormente, los resultados más favorables en general se han obtenido para la secuencia KITTI, con amplio margen respecto a las otras secuencias. En el caso de KITTI05, nos encontramos frente una secuencia larga que captura un recorrido de un entorno exterior. Ha sido tomada por una cámara de

alta calidad fija sobre un vehículo que se desplaza a una velocidad constante en un barrio residencial, sin una presencia notable de tráfico ni elementos que alteren en gran medida la apariencia para dos capturas de la misma escena en momentos diferentes. El hecho de ir instalada de forma fija sobre un vehículo aporta también una alta estabilidad a las imágenes a lo largo de la secuencia, siendo mínima la presencia de rotaciones u otras desviaciones. Todos estos factores favorecen que el desempeño de todas las estrategias validadas sobre esta secuencia aporten unos resultados muy superiores al resto. Al tratarse de imágenes grandes con una textura en general muy regular (grandes regiones ocupadas por el cielo, tramos de asfalto y vegetación), favorecen al algoritmo de segmentación LSC.

	Tipo	Características					Cambios			
Secuencia	**Int/Ext**	**Tamaño**	**Calidad**	**Estabilidad**	**Velocidad**	**Longitud**	**Dinam.**	**Rotación**	**Iluminación**	**Contornos**
KITTI05	Exterior	Grande	Alta	Alta	Media	Larga	Medio	Bajo	Constante	Constante
UIB Small	Exterior	Pequeño	Baja	Baja	Baja	Corta	Medio	Alto	Constante	Constante
Lip6	Interior	Pequeño	Baja	Baja	Baja	Corta	Bajo	Medio	Constante	Constante
UIB Large	Exterior	Pequeño	Baja	Baja	Baja	Media	Medio	Alto	Constante	Constante
Nordland	Exterior	Grande	Alta	Muy Alta	Alta	Muy Larga	Alto	Bajo	Variable	Variable

Tabla 5.1: Resumen de características de las secuencias empleadas.

	Estrategia con mejor F-Score		Resultados				
Secuencia	**Segmentación**	**Descriptor**	**P**	**R**	**Umbral**	**F-Score**	**AuC**
KITTI05	LSC	CNN	0.619	0.483	0.20	0.54	0.57
UIB Small	SEEDS	SIFT	0.293	0.744	0.10	0.42	0.05
Lip6	SEEDS	CNN	0.385	0.361	0.25	0.37	0.31
UIB Large	SEEDS	CNN	0.126	0.251	0.10	0.17	0.03
Nordland	SEEDS	SIFT	0.070	0.427	0.20	0.12	0.05

Tabla 5.2: Resultados obtenidos para la estrategia más favorable en cada secuencia según el Valor-F.

	Estrategia con mejor AuC		Resultados				
Secuencia	**Segmentación**	**Descriptor**	**P**	**R**	**Threshold**	**F-Score**	**AuC**
KITTI05	SEEDS	CNN	0.496	0.576	0.30	0.53	0.59
Lip6	SEEDS	CNN	0.385	0.361	0.25	0.37	0.31
UIB Small	SEEDS	SIFT	0.293	0.744	0.10	0.42	0.24
UIB Large	SEEDS	SIFT	0.070	0.427	0.20	0.12	0.05
Nordland	SEEDS	SIFT	0.070	0.427	0.20	0.12	0.05

Tabla 5.3: Resultados obtenidos para la estrategia más favorable en cada secuencia según la métrica AuC.

Este amplio conjunto de características no se dan en ningún caso en las otras secuencias. Tal y como se ha comentado, en las siguientes secuencias se obtienen unos resultados muy inferiores. En el caso de UIB Small Loop, UIB Large Loop y Lip6, nos encontramos frente a secuencias compuestas por imágenes de un tamaño muy inferior. Además, estas secuencias han sido tomadas con una cámara de mano realizando el desplazamiento a pie, disminuyendo así la calidad y la estabilidad de las imágenes. Esto provoca rotaciones y desplazamientos en el encuadre de las escenas. El hecho de

que las imágenes sean más pequeñas genera también que la aparición de elementos dinámicos tenga más influencia en la apariencia general de la escena.

Finalmente, Nordland se sitúa claramente fuera de los límites de rendimiento de las estrategias planteadas. A pesar de tratarse de una secuencia con una alta estabilidad, capturada mediante una cámara de alta calidad dispuesta sobre un tren a una velocidad en general constante, mucho más alta que cualquiera de las anteriores, los cambios dramáticos entre las escenas de cada estación en cuanto a iluminación y contornos debidos a los cambios día/noche y a los fenómenos atmosféricos resultan determinantes.

CAPÍTULO 6

CONCLUSIONES Y TRABAJO FUTURO

En este capítulo se exponen las conclusiones derivadas de los experimentos llevados a cabo en el desarrollo de este proyecto y se plantean posibles líneas de trabajo futuro de cara a mejorar el rendimiento de las estrategias evaluadas.

6.1 Conclusiones

El presente documento recoge un conjunto de estrategias de reconocimiento visual de escenas basadas en técnicas de segmentación en superpíxeles y descriptores locales. Además, se ha estudiado una alternativa consistente en describir cada segmento mediante descriptores basados en redes neuronales convolucionales, dado el vasto crecimiento de estas técnicas en multitud de campos y aplicaciones. Para ello, se han analizado una serie de algoritmos de segmentación, midiendo su capacidad para generar resultados estables a lo largo de una secuencia de imágenes. De esta forma, se han seleccionado aquellos que han demostrado solidez identificando las mismas regiones de la escena como segmentos únicos entre fotogramas consecutivos de una secuencia. Posteriormente, se han valorado una serie de descriptores locales, además de la implementación de un descriptor de superpíxeles basado en histogramas de color y textura, y descriptores basados en CNN. Éstos se han probado frente a diversos conjuntos de tres fotogramas de la misma escena sometida a ligeros cambios de perspectiva. En el desarrollo de esta tarea, ha sido preciso elaborar un conjunto de valores de referencia para cada pareja de imágenes consistente en un listado compuesto por la relación de los índices de los segmentos que comprendían la misma región de la escena. Finalmente, aquellas estrategias que han aportado unos resultados más favorables se han validado frente a diferentes secuencias de características diversas para comprobar su capacidad en la tarea de reconocimiento visual de escenas.

Los resultados obtenidos en los experimentos finales indican que el rendimiento de las estrategias propuestas tienen una alta sensibilidad frente a cambios en la orientación de las imágenes, así como a la aparición de elementos dinámicos. No obstante, resultan

especialmente determinantes las variaciones en la iluminación y los contornos de las escenas, siendo factores clave en la capacidad de reconocimiento de las propuestas planteadas. En lo referente a la segmentación, se ha comprobado un mejor rendimiento en términos de estabilidad para aquellos algoritmos basados en conjuntos que trabajan con parámetros de textura globales (LSC) o los que varían según la región (SLICO), frente aquellos más rígidos (SLIC). No obstante, todos ellos se han visto ampliamente superados en esta faceta por SEEDS. No obstante, éste se ve penalizado por una menor adherencia a los bordes de los objetos. En cuanto a la descripción, SIFT ha demostrado ser un descriptor de superpíxeles muy eficaz, superando notablemente en precisión al resto de descriptores locales probados y con un número de emparejamientos mucho mayor a los descriptores CNN. Dentro de esta categoría de descriptores, se ha podido comprobar que su eficacia aumenta de manera considerable al utilizar modelos pre-entrenados mediante secuencias específicas de escenas.

6.2 Trabajo futuro

Con todo lo expuesto en este documento, se pueden valorar diversas líneas de trabajo en las que indagar para obtener una estrategia de reconocimiento de escenas más completa aumentando así su eficacia:

- Plantear una estrategia de indexación de imágenes, basada en superpíxeles, para mejorar los tiempos de acceso.
- Añadir una componente al descriptor basado en histogramas que aporte información acerca de la forma del superpíxel.
- Integrar dicha solución como detector de bucles en un sistema completo de SLAM.
- Probar estrategias más completas de descripción de superpixeles basadas en CNN, tales como: RMAC [46] [47], SPoC [48], CROW [49], etc.

BIBLIOGRAFÍA

[1] C. Rus. (2021) La roomba para ciudades se llama trombia: tiene el tamaño de un coche y limpia las calles de forma autónoma. [Online]. Available: https://www.xataka.com/robotica-e-ia/roomba-para-ciudades-se-llama-trombia-tiene-tamano-coche-limpia-calles-forma-autonoma (document), 1.1

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels," *Technical report, EPFL*, 06 2010. (document), 2.1, 3.1.1

[3] A. Rosebrock. (2015) Local binary patterns with python and opencv. [Online]. Available: https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/ (document), 4.1, 4.2

[4] R. Mash, N. Becherer, B. Woolley, and J. Pecarina, "Toward aircraft recognition with convolutional neural networks," 07 2016. (document), 4.3, 4.3

[5] M. Tiwari, T. Tiwari, M. Kassab, A. Roy, D. Chaudhary, and E. Onyema, "Detection of coronavirus disease in human body using convolutional neural network," vol. 29, pp. 2861–2866, 04 2020. (document), 4.4

[6] A. Barrientos, A. Cruz, L. Peñín, and C. Balaguer, *Fundamentos de robótica*. McGraw-Hill, 2007. 1.1, 1.5

[7] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*, 2nd ed. USA: Cambridge University Press, 2010. 1.1

[8] S. Lowry, N. Suenderhauf, P. Newman, J. Leonard, D. Cox, P. Corke, and M. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016. 1.3, 2.1, 2.3

[9] E. G. Fidalgo, "Appearance-based loop closure detection and its application to topologial mapping and image mosaicing," Ph.D. dissertation, Universitat de les Illes Balears, Palma de Mallorca, 6 2016. 1.4, 2.1.1, 5.1

[10] P. Neubert and P. Protzel, "Beyond holistic descriptors, keypoints and fixed patches: Multi-scale superpixel grids for place recognition in changing environments," *IEEE Robotics and Automation Letters*, 12 2015. 2.1.1, 4.3

[11] A. Oliva and A. Torralba, "Torralba, a.: Building the GIST of a scene: The role of global image features in recognition. progress in brain research 155, 23-36," *Progress in brain research*, vol. 155, pp. 23–36, 02 2006. 2.1.1

[12] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: A survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, pp. 177–280, 11 2007. 2.1.2

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," vol. 60, no. 2, p. 91–110, 2004. 2.1.2, 4.2.1

[14] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," vol. 3951, 07 2006, pp. 404–417. 2.1.2

[15] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 778–792. 2.1.2, 4.2.2

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to sift or surf," 11 2011, pp. 2564–2571. 2.1.2, 4.2.5

[17] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *CoRR*, vol. abs/1612.01601, 2016. 2.2, 3.4

[18] D. Comaniciu and P. Meer, "Meer, p.: Mean shift: A robust approach toward feature space analysis. IEEE transactions on pattern analysis and machine intelligence 24(5), 603-619," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 603 – 619, 06 2002. 2.2

[19] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," vol. 5305, 10 2008, pp. 705–718. 2.2

[20] A. Humayun, F. Li, and J. M. Rehg, "The middle child problem: Revisiting parametric min-cut and seeds for object proposals," in *Computer Vision (ICCV), IEEE International Conference on*. IEEE, Dec 2015, pp. 1600–1608. [Online]. Available: http://rehg.org/POISE/ 2.2

[21] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," 09 2010, pp. 211–224. 2.2

[22] L. Lu, C. Wang, and X. Yin, "Incorporating texture into SLIC super-pixels method for high spatial resolution remote sensing image segmentation," in *2019 8th International Conference on Agro-Geoinformatics (Agro-Geoinformatics)*, 2019, pp. 1–5. 2.2, 3.1.2

[23] J. Chen, Z. Li, and B. Huang, "Linear spectral clustering superpixel," *IEEE Transactions on Image Processing*, vol. PP, pp. 1–1, 01 2017. 2.2, 3.1.3, 3.4, 3.4

[24] M. V. den Bergh, X. Boix, G. Roig, B. de Capitani, and L. V. Gool, "SEEDS: Superpixels extracted via energy-driven sampling," 2013. [Online]. Available: https://arxiv.org/abs/1309.3848v1 2.2, 3.1.4, 3.4

[25] F. Drucker and J. MacCormick, "Fast superpixels for video analysis," *2009 Workshop on Motion and Video Computing (WMVC)*, pp. 1–8, 2009. 2.2

[26] D. Tang, H. Fu, and X. Cao, "Topology preserved regular superpixel," in *2012 IEEE International Conference on Multimedia and Expo*, 2012, pp. 765–768. 2.2

[27] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, pp. 2290–7, 12 2009. 2.2

[28] R. El Euch, E. Garcia-Fidalgo, A. Ortiz, F. Chaabane, and A. Ghazel, "Superpixel description and indexing for visual loop closure detection," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 1591–1594. 2.2

[29] E. G. Fidalgo, "Lecture 6: Deep learning in CBIR," March 2020. 2.3

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. 2.3, 4.3, 4.3

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," vol. 60, no. 6, p. 84–90, May 2017. 2.3

[32] N. Sünderhauf, S. Shirazi, A. Jacobson, E. Pepperell, F. Dayoub, B. Upcroft, and M. Milford, "Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free," 07 2015. 2.3

[33] P. Neubert, "Superpixels and their application for visual place recognition in changing environments," Ph.D. dissertation, 12 2015. 2.3

[34] R. Arroyo, P. Fernández Alcantarilla, L. Bergasa, and E. Romera, "Fusion and binarization of CNN features for robust topological localization across seasons," 10 2016. 2.3

[35] P. Jaccard, "The distribution of the flora in the alpine zone.1," *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912. [Online]. Available: https://nph.onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8137.1912.tb05611.x 3.2.2

[36] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002. 4.1

[37] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 1960–1967. 4.1

[38] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555. 4.2.3

[39] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," 06 2012, pp. 510–517. 4.2.4

[40] T. I. Songlin Du, "Low-dimensional superpixel descriptor and its application in visual correspondence estimation," 2019. 4.6

[41] J. Hui, "Map mean average precision for object detection," 2018. 5.1

[42] W. Su, Y. Yuan, and M. Zhu, "A relationship between the average precision and the area under the ROC curve," in *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, ser. ICTIR '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 349–352. [Online]. Available: https://doi.org/10.1145/2808194.2809481 5.1

[43] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361. 5.1

[44] A. Angeli, D. Filliat, S. Doncieux, and J. Meyer, "A fast and incremental method for loop-closure detection using bags of visual words," *Robotics, IEEE Transactions on*, vol. 24, pp. 1027 – 1037, 11 2008. 5.1

[45] P. Neubert, N. Sünderhauf, and P. Protzel, "Superpixel-based appearance change prediction for long-term navigation across seasons," *Robotics and Autonomous Systems*, vol. 69, pp. 15–27, 2015. 5.1

[46] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of CNN activations," *CoRR*, vol. abs/1511.05879, 2016. 6.2

[47] O. Seddati, S. Dupont, M. Saïd, and M. Parian-Scherb, "Towards good practices for image retrieval based on CNN features," 10 2017, pp. 1246–1255. 6.2

[48] B. Rothrock, R. Kennedy, C. Cunningham, J. Papon, M. Heverly, and M. Ono, "SPOC: Deep learning-based terrain classification for mars rover missions," 09 2016. 6.2

[49] Y. Kalantidis, C. Mellina, and S. Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," vol. 9913, 10 2016, pp. 685–701. 6.2