Universidad Zaragoza
1542

Proyecto Fin de Carrera

Descriptores globales binarios para reconocimiento de imágenes

Autor:

Santiago Escorihuela Miravet

Director:

Javier Civera Sancho

Ingeniería Industrial
Escuela de Ingeniería y Arquitectura
2013

RESUMEN: Descriptores globales binarios para reconocimiento de imágenes

La visión por computador es la disciplina cuyo objetivo se suele plantear como "que un ordenador pueda ver". La definición de "ver" es bastante compleja, puesto que todos los mecanismos de la visión humana todavía no están bien entendidos. Pero sin duda alguna, uno de los aspectos que involucra la visión humana y que ha sido objeto de estudio por la visión por computador es el reconocimiento de escenas. En dicho problema, un computador recibe una imagen y debe clasificarla según la escena en la que ha sido tomada (parque, oficina, aeropuerto...).

Uno de los aspectos más importantes en el reconocimiento de imágenes es la forma de describir el contenido de la imagen. Algebraicamente, un descriptor suele ser un vector de números reales más o menos complejo de extraer a partir de la imagen. Idealmente, dicho descriptor debería contener la información necesaria para clasificar la escena de la imagen. En el estado actual de la técnica, las tasas de reconocimiento visual de escenas son bastante bajas y el problema dista mucho de estar resuelto, por lo que es objeto de investigación.

El problema de algunos descriptores es la cantidad de cómputo necesario para extraerlos y evaluarlos, así como la memoria requerida para almacenarlos. Este problema es muy relevante cuando las bases de datos de imágenes adquieren tamaños muy grandes, como Google Imágenes o las imágenes de Facebook. En estas bases de datos, que requieren técnicas avanzadas para la clasificación de las imágenes en función de su contenido, cualquier mejora en tiempo o almacenamiento conlleva un gran ahorro.

El objetivo del proyecto es la propuesta de un descriptor binario y global para la clasificación de imágenes. La ventaja de este descriptor respecto a otros es el ahorro en tiempo de cómputo y almacenamiento: Las operaciones binarias pueden realizarse muy rápidamente en los procesadores actuales. Y un número binario ocupa 1 bit, mientras que un real ocupa como mínimo 32 bits. Además de la propuesta, se evalúa el comportamiento del descriptor en una base de datos estándar de visión por computador (SUN database). En dicha evaluación se han explorado diferentes configuraciones del descriptor para encontrar la configuración óptima y poder compararla con un descriptor del estado del arte. En concreto, se ha comparado con el descriptor Tiny images, ya que es el que tiene una configuración más similar.

TABLA DE CONTENIDOS

1. INTRODUCCIÓN

1.1 VISION POR COMPUTADOR

La visión por computador es la rama de la inteligencia artificial cuyo objetivo es hacer "que el ordenador pueda ver". Pese a que pueda parecer simple, se trata de una tarea muy compleja, que todavía no está resuelta.

La visión por computador es una disciplina muy fragmentada que abarca múltiples aplicaciones, a continuación se listan las más importantes:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo: caras humanas, coches, personas, etc.). *(Véase Figura 1)*
- La evaluación de los resultados (por ejemplo, segmentación y registro).
- Registro de diferentes imágenes de una misma escena u objeto, es decir, hacer concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.
- Mapeo de una escena para generar un modelo tridimensional de la escena; este modelo podría ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.
- Reconocimiento de escenas: Búsqueda de imágenes digitales por su contenido.

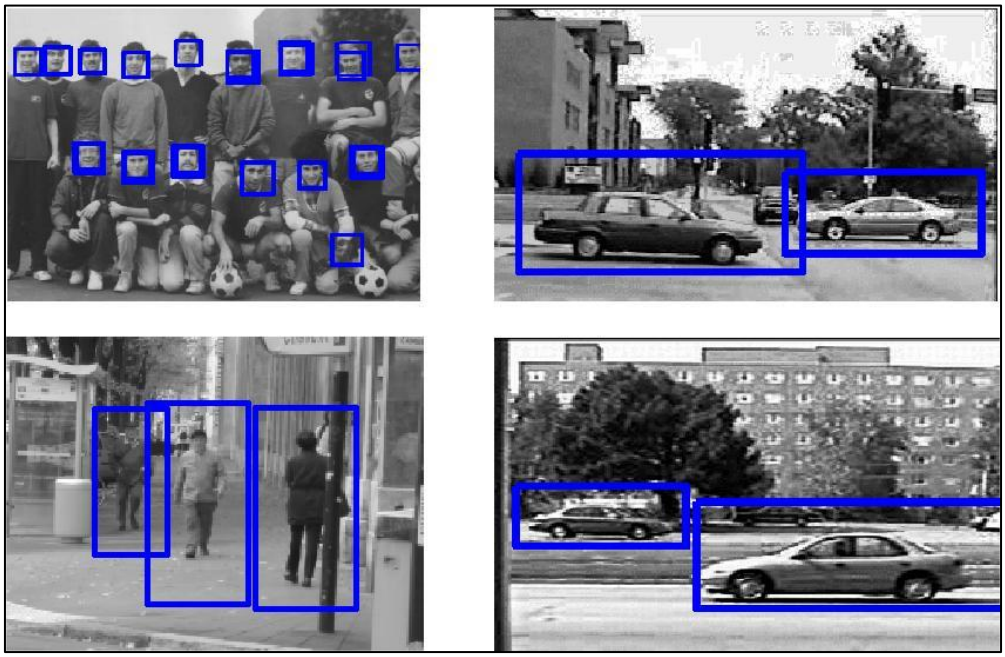

Figura 1: Ejemplos de detección de objetos: Imagen sup. Izq: Reconocimiento de caras. Imagen sup. dcha: Reconocimiento estático de vehículos. Imagen inf. izq: Reconocimiento de personas. Imagen inf. dcha. Reconocimiento de vehículos en movimiento [1].

1.2 RECONOCIMIENTO DE ESCENAS

De todos los objetivos y aplicaciones de la visión por computador indicados en el anterior apartado, el presente proyecto se centra en el reconocimiento de escenas.

El reconocimiento de escenas es el proceso de clasificación de imágenes en base a su contenido semántico. Es decir, asignar categorías a las imágenes en función de los objetos o escenas que se muestran en ellas.

El ejemplo de aplicación más claro lo encontramos en Google imágenes, en este conocido buscador, al introducir en texto una breve descripción de la imagen (preferiblemente una o dos palabras), nos muestra imágenes cuyo contenido está relacionado con la búsqueda.

En la *Figura 2* se muestran, a modo de ejemplo, los resultados obtenidos al introducir "playa" en el buscador.

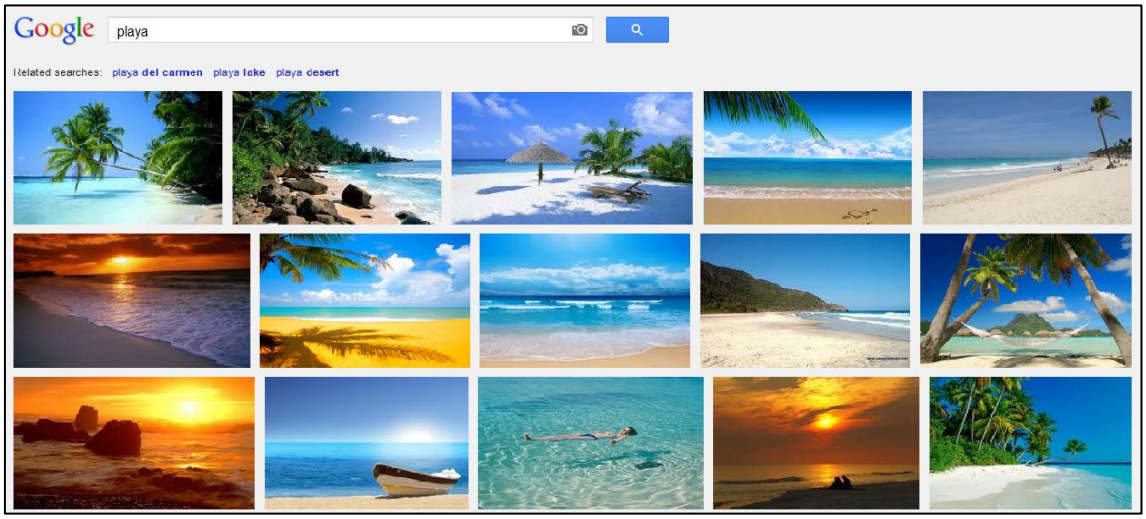

Figura 2: Ejemplo de aplicación práctica del reconocimiento de escenas, al introducir la palabra "playa" en el buscador de Google, este nos devuelve imágenes que concuerdan perfectamente con la búsqueda.

En el ejemplo de la *Figura 2* hemos realizado una búsqueda muy sencilla, ya que la palabra "playa" delimita perfectamente la búsqueda. Sin embargo, cuando realizamos la búsqueda peor delimitada, como la que se muestra en la *Figura 3*, los resultados obtenidos no son tan precisos. En esta figura se muestran los resultados de introducir "casa de campo" en el buscador, dónde podemos observar la presencia de imágenes cuyo contenido no se corresponde exactamente con lo buscado. En concreto, la tercera imagen de la primera fila (recuadrada en rojo) muestra una panorámica de un lugar costero. Mientras que la tercera imagen de la segunda fila (recuadrada en rojo) muestra una piscina.

Esta imperfección en los resultados es debida a que Google basa su clasificación en las palabras que acompañan a la imagen, fijándose mucho menos en el contenido semántico de las imágenes. Lo cual demuestra que el reconocimiento de escenas es una disciplina en desarrollo, en la que aún quedan muchos avances por realizar.

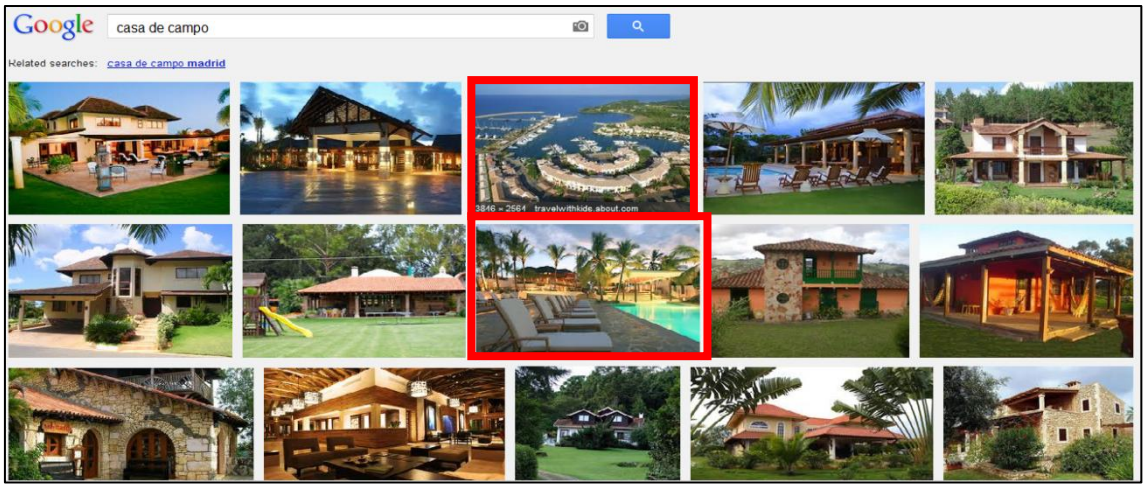

Figura 3: Ejemplo de imperfección en el reconocimiento de escenas. Al introducir la búsqueda "casa de campo" en el buscador de Google imágenes, obtenemos algunas imágenes cuyo contenido se corresponde con la búsqueda, y otras en las que no. Como son las imágenes recuadradas en rojo.

Finalmente, en la *Figura 4* se muestra un ejemplo ilustrativo de reconocimiento de escenas. En la parte izquierda de la tenemos las nuevas imágenes sin clasificar. El reconocimiento de escenas se encarga de organizarlas en función de su contenido semántico. Es decir, las imágenes 1 y 4 en las que se observan playas, son asignadas a la categoría "playa", mientras que las imágenes 2 y 3 son asignadas a las categorías "pueblo" y "montaña" respectivamente.

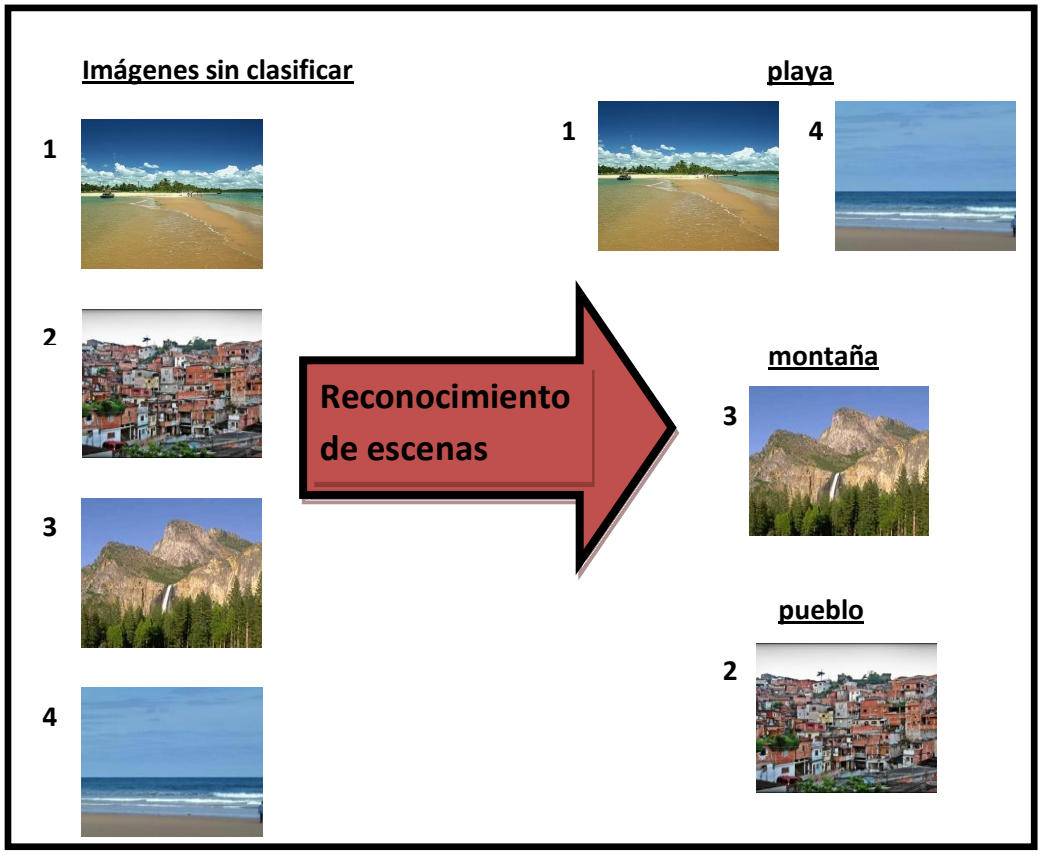

Figura 4: Ejemplo ilustrativo de reconocimiento de escenas. A la izquierda tenemos las imágenes de entrada, sin clasificar. El reconocimiento de escenas se encarga de clasificar las imágenes que muestran una playa (1 y 4) en la categoría "playa", la que muestra una montaña (3) en la categoría "montaña" y la que muestra un pueblo (2) en la categoría "pueblo".

1.3 OBJETIVOS Y ALCANCE DEL PROYECTO

El presente proyecto propone un nuevo descriptor de imagen para el reconocimiento de escenas, al que hemos llamado **G-BRIEF**.

Un descriptor de imagen es un conjunto de números ordenados, que contiene información acerca del contenido semántico de la imagen que describe. En la siguiente sección se profundiza en la definición de este concepto.

El G-BRIEF ha sido concebido como un descriptor sencillo, cuyo principal objetivo es obtener un descriptor que reduzca al máximo el consumo de recursos en los procesos de reconocimiento de escenas (tanto el tiempo de ejecución, como el espacio necesario para el almacenamiento de los descriptores), manteniendo un nivel de acierto similar, o incluso superior, al estado del arte.

Esta sencillez y eficiencia del G-BRIEF hacen que su principal aplicación sea el reconocimiento de escenas en grandes bases de datos, como Google imágenes. En estas bases de datos se realizan búsquedas entre millones de imágenes, por lo que se deben emplear descriptores muy sencillos y eficientes.

Una vez propuesto el descriptor, en el presente proyecto, se ha implementado y evaluado el G-BRIEF hasta llegar a su configuración óptima, de modo que su aplicación en el reconocimiento de escenas nos proporcione el mejor ratio de acierto posible.

Finalmente, con la intención de probar su valía, hemos comparado nuestro descriptor con el estado del arte.

En la sección 2 se profundiza en el concepto de descriptor de imagen, para posteriormente realizar una descripción detallada de los tipos de descriptores que se han utilizado hasta la fecha, con el objetivo de contextualizar la propuesta del G-Brief.

En la sección 3 se presenta la base de datos (o data set) que se ha utilizado para evaluar el descriptor, y compararlo con el estado del arte.

En la sección 4 se presenta el algoritmo de clasificación empleado, que será el encargado de clasificar los descriptores de las imágenes de la base de datos.

En la sección 5 se exponen las medidas de evaluación empleadas.

En la sección 6 se exponen los experimentos realizados junto con los resultados más relevantes.

Para finalizar, las principales conclusiones alcanzadas por el presente proyecto se exponen en la sección 7.

2. DESCRIPTORES

2.1 ¿QUÉ ES UN DESCRIPTOR?

Un descriptor es un conjunto de números ordenados, que contiene información acerca de la imagen que describe. Suelen presentarse en forma de vector, o matriz.

La imagen digital, en sí misma, puede considerarse un descriptor. Aunque con un grave problema, su tamaño: Una imagen de 1 Mega pixel (1.000.000 pixeles), teniendo en cuenta que cada pixel ocupa 3 bytes, tiene un tamaño de 3 Megabytes.

Por ello se han desarrollado multitud de descriptores, que tratan de condensar la información relevante de la imagen, para su posterior procesamiento.

A continuación se van a describir los diferentes tipos de descriptores existentes.

2.2 DESCRIPTORES GLOBALES

Los descriptores globales dan información acerca del conjunto de la imagen. Debido a la gran cantidad de pixeles que debe representar el descriptor, la información es muy genérica y nos informa sobre las texturas y formas generales de la imagen.

En la *Figura 5* observamos un ejemplo ilustrativo de la formación de un descriptor global D_G. A partir de una imagen digital (Izquierda) obtenemos el descriptor global (derecha).

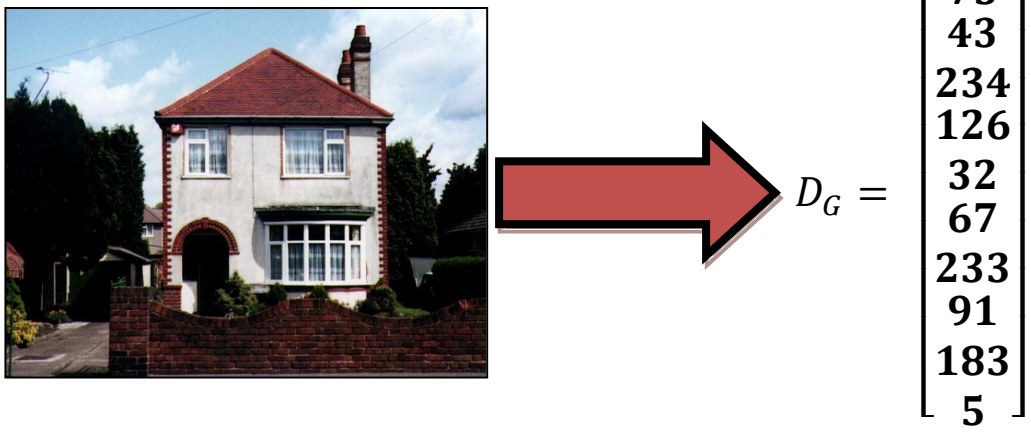

Figura 5: Ejemplo ilustrativo de la formación de un descriptor global D_G (derecha) a partir de una imagen digital (izquierda).

2.3 DESCRIPTORES LOCALES

Al contrario que los descriptores globales, los descriptores locales no describen toda la imagen en su conjunto, sino una pequeña parte de ella, seleccionada por ser más característica y diferenciable del resto de la imagen.

Lo más habitual es seleccionar zonas de alto gradiente, como pueden ser los bordes y esquinas de los objetos de la imagen.

En la *Figura 6* se observa, a modo ilustrativo, el proceso de formación de un descriptor local. A la izquierda de la figura se encuentra la imagen original, en la que se han recuadrado en rojo las regiones seleccionadas para la formación de un descriptor local. Arriba a la derecha, observamos una de las regiones recuadradas en rojo, esta región es la característica local. Finalmente, abajo tenemos el descriptor local D_L obtenido, en forma de vector.

De esta manera, la imagen se representa como un conjunto de descriptores locales. Nótese la diferencia con la sección anterior, donde la imagen se representa con un único descriptor global. A consecuencia de esto, el uso de descriptores locales requiere mayor tiempo de ejecución y espacio de almacenamiento que el uso de descriptores globales.

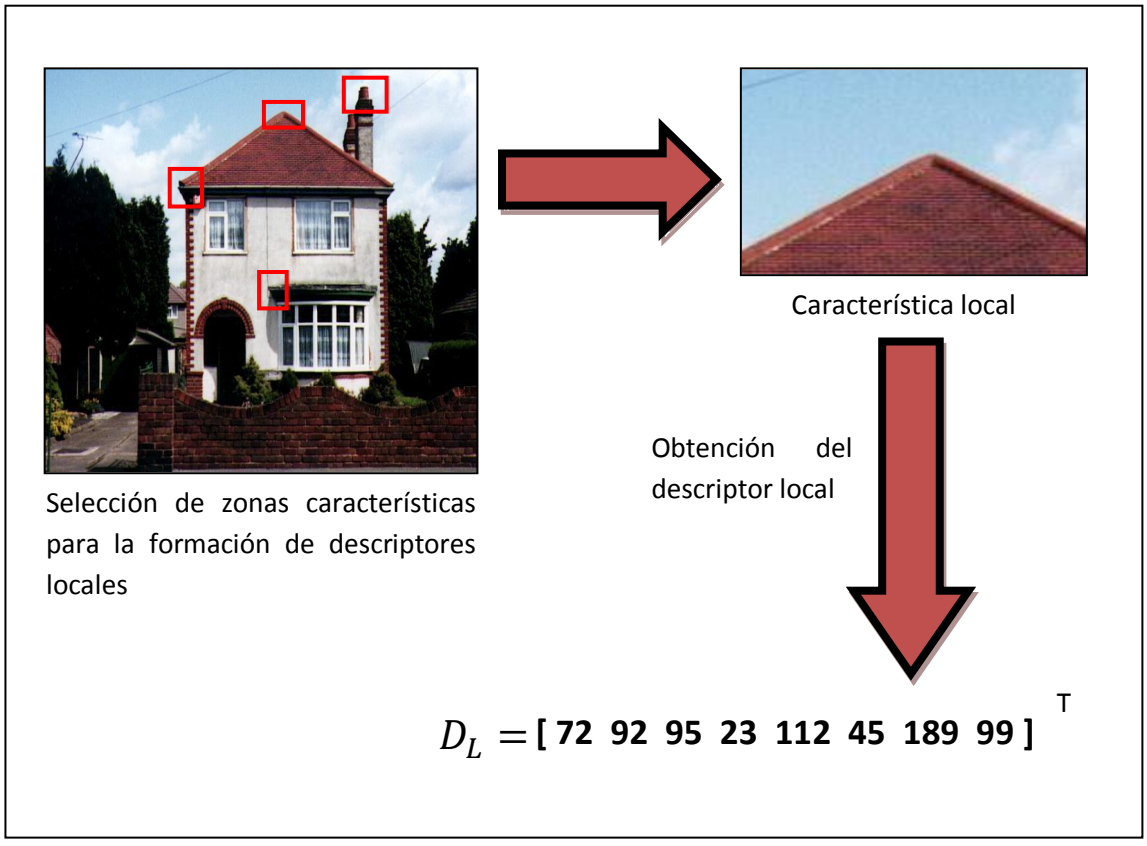

Figura 6: Ejemplo ilustrativo de la obtención de un descriptor local. Arriba a la izquierda: Imagen original con una selección de las regiones características (recuadradas en rojo) para la formación de descriptores locales. Arriba a la derecha: Imagen ampliada de una de estas regiones (Característica local). Abajo: Descriptor local obtenido en forma de vector.

2.4 DESCRIPTORES BINARIOS

Lo más común en el campo de los descriptores de imágenes es que cada elemento del descriptor tenga un tamaño de varios bytes (los vectores contienen números reales). Esto es debido a que el valor de sus elementos es el resultado de la combinación de varios pixeles de la imagen (un pixel es un número real).

Sin embargo, se han desarrollado otros tipos de descriptores en los que cada elemento es codificado en un bit (valor 0 ó 1), y por ello se conocen como descriptores binarios.

La principal ventaja de este tipo de descriptores radica en la rapidez con la que un ordenador actual puede realizar operaciones binarias. Al tiempo que estamos reduciendo drásticamente el espacio necesario para almacenar los descriptores.

Como contrapartida, la información contenida en un descriptor binario es mucho menor.

2.5 DESCRIPTORES EXISTENTES

2.5.1 Tiny images:

El Tiny images [2] es uno de los descriptores globales más populares y ha sido desarrollado por *Antonio Torralba, Rob Fergus y William T. Freeman*.

Con la creación del Tiny images se buscó un descriptor que fuese muy sencillo de obtener y que proporcionase unos ratios de acierto similares a los de otros descriptores mucho más complejos. Al tratarse de un descriptor global, cada imagen es representada por un único descriptor, con el consiguiente ahorro en costes de computación y almacenamiento. Los creadores del Tiny images llegaron a la conclusión de que la dimensión óptima de este descriptor son 1024 elementos.

A continuación se muestra la forma de obtener el descriptor Tiny images:

Partiendo de una imagen completa I de dimensiones *nxm* pixeles, aplicamos una reducción de tamaño hasta obtener una imagen reducida I_r , de dimensiones *32x32* pixeles (*véase Figura 7*):

Figura 7: Ejemplo ilustrativo de la formación del descriptor Tiny images. A la derecha tenemos la imagen de partida (I) que es reducida hasta un tamaño de 32x32 pixeles (I_r). Finalmente el Tiny images es obtenido posicionando en forma de vector los 1024 pixeles (32x32) que contiene la imagen reducida (I_r).

Una vez que tenemos la imagen reducida I_r , formaremos el descriptor ordenando sus 1024 elementos en forma de vector columna:

$$D_{Tiny}^{1024} = \begin{bmatrix} I_r(1,1) \\ I_r(1,2) \\ \vdots \\ I_r(1,32) \\ I_r(2,1) \\ \vdots \\ I_r(32,32) \end{bmatrix}$$

Donde:

- D_{Tiny}^{1024}: Es el descriptor Tiny images de longitud 1024.
- $I_r(n,m)$: Es el elemento situado en la fila n y columna m de la imagen reducida (I_r).

El proceso de reducción hace que la información contenida en aproximadamente 360.000 pixeles se comprima en tan solo 1024, con la consiguiente pérdida de información, ahorro de espacio de memoria y ahorro en tiempo de cómputo. Al mismo tiempo, la reducción de tamaño tiene un efecto de filtrado sobre la imagen, que facilitara el empleo del descriptor para el reconocimiento de escenas.

En la sección 6 (Resultados experimentales), se han comparado nuestros resultados con los obtenidos mediante el uso de este descriptor.

2.5.2 BRIEF: Binary Robust Independent Elementary Features

El BRIEF [3] es un exitoso descriptor local y binario desarrollado por *Michael Calonder, Vincent Lepetit, Christoph Strecha, y Pascal Fua.*

El gran éxito de este descriptor estriba en que sus elementos son binarios (valor 0 ó 1), lo que permite ahorrar gran cantidad de espacio de memoria y tiempo de computo, respecto a otros descriptores del estado del arte.

A modo de ejemplo, un descriptor binario de 1024 elementos ocupa un espacio de memoria de 128 bytes, mientras que un Tiny images de 1024 elementos ocupa 1024 bytes.

Para comprender como se implementan los descriptores BRIEF *véase la Figura 6*, en la que partimos de una imagen completa (I), en la que se seleccionan varias regiones características (recuadradas en rojo). A partir de cada una de estas regiones se forma un descriptor BRIEF, de modo que la imagen (I) está representada por el conjunto de los l descriptores BRIEF.

$$\mathrm{I} \Longrightarrow \{D_{BRIEF}^{q}(1), D_{BRIEF}^{q}(2), \ \dots \ , D_{BRIEF}^{q}(l)\}$$

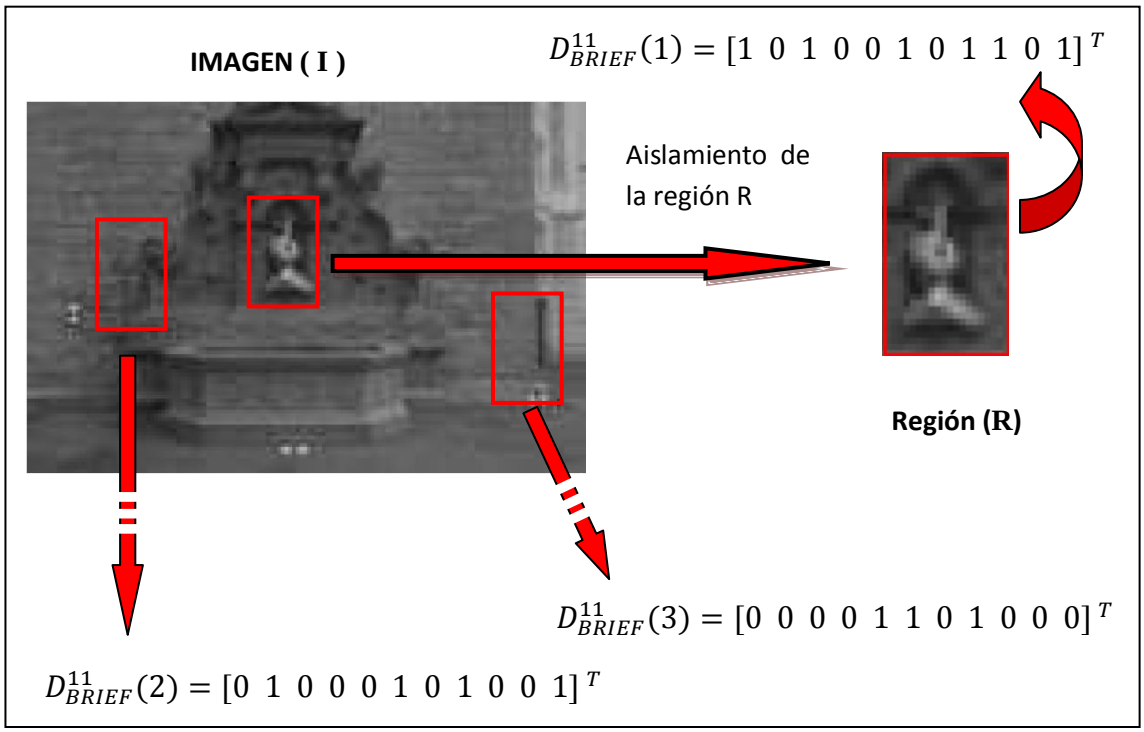

Figura 8: Ejemplo ilustrativo de formación de descriptores BRIEF. A la izquierda de la Figura tenemos la imagen (I) con tres regiones características recuadradas en rojo. A partir de cada una de estas regiones se obtiene un descriptor BRIEF, de modo que el conjunto de los tres descriptores BRIEF representan la imagen (I). El proceso ha sido detallado para una de estas regiones, a la que hemos llamado región (R). A la derecha de la figura podemos observar la región (R) aislada, a partir de esta imagen se forma el descriptor BRIEF.

A la derecha de la *Figura 8* observamos una región característica de la imagen (I) a la que hemos llamado región (R). El proceso para obtener el descriptor BRIEF a partir de esta región, se explica en los siguientes párrafos:

En primer lugar, se selecciona un número determinado de pixeles de R, agrupados por parejas (x^R, y^R).Para esta selección se pueden utilizar diferentes distribuciones estadísticas, o simplemente una selección aleatoria (*véase Figura 9*). Seleccionaremos tantas parejas de pixeles como elementos queramos que tenga nuestro descriptor:

$$x^R = \{x_1^R, x_2^R, x_3^R, \dots\dots\dots, x_{q-1}^R, x_q^R\}$$

$$y^R = \{y_1^R, y_2^R, y_3^R, \dots\dots\dots, y_{q-1}^R, y_q^R\}$$

Donde:

- q: Es el número de parejas de pixeles seleccionadas para la formación del descriptor. A su vez, define el tamaño del descriptor, para el que se eligen potencias binarias como 128, 256, 512, etc.
- x^R *e* y^R*:* son los vectores que contienen el primer y el segundo integrante de las parejas de pixeles procedentes de la región R.

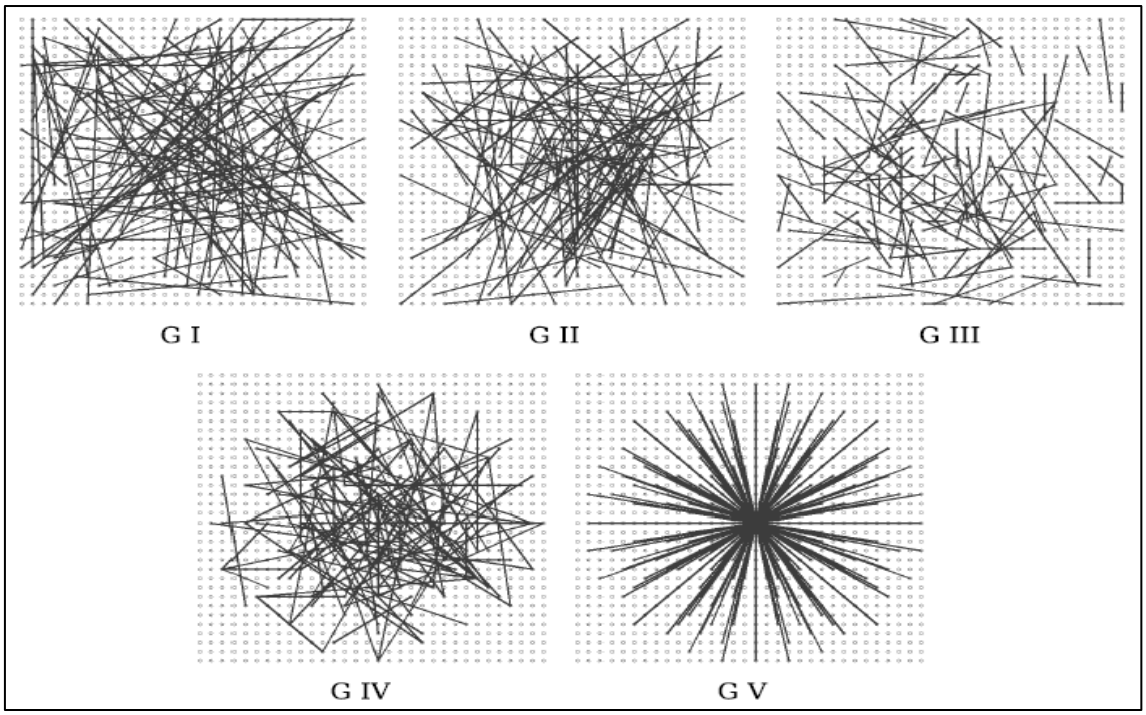

Figura 9: Ejemplos de de selección de pixeles mediante distribuciones aleatorias [3]:

- *G I: Distribución uniforme, todos los pixeles tienen las mismas probabilidades de ser seleccionados.*
- *G II: Distribución Gaussiana centrada en el origen.*
- *G III: Distribución Gaussiana de dos etapas. En primer lugar se seleccionan los pixeles x_i mediante una distribución Gaussiana centrada en el origen, posteriormente se obtiene cada uno de los pixeles y_i mediante una distribución Gaussiana centrada en el correspondiente x_i .*
- *G IV: Distribución aleatoria que selecciona pixeles discretos de una cuadricula polar previamente creada.*
- *G V: En esta distribución todos los pixeles x_i se encuentran en el origen (0, 0), y los pixeles y_i se seleccionan aleatoriamente de una cuadricula polar previamente creada.*

Una vez tenemos seleccionadas las q parejas de pixeles (x^R, y^R), se compara la intensidad del primer pixel de la pareja, $p(x^R)$, con la del segundo $p(y^R)$. Obteniéndose el descriptor de la aplicación de la siguiente regla:

Para $i = 1$ hasta $i = q$

$$D^q_{BRIEF}(i) = \begin{cases} 1 \; si \; p(x^R_i) > \; p(y^R_i) \\ 0 \qquad en \; otro \; caso \end{cases}$$

Donde:

- D^q_{BRIEF}: Es el descriptor BRIEF de longitud q
- i : Es el índice que recorre todas las posiciones del vector descriptor D^q_{BRIEF}
- $p(x_i)$: Es la intensidad del primer pixel de la pareja
- $p(y_i)$: Es la intensidad del segundo pixel de la pareja

2.6 G-BRIEF: Descriptor global y binario

Una vez descritos los tipos de descriptores de imágenes posibles, junto con algunos de los ejemplos más significativos. Vamos a presentar el descriptor que se desarrolla y evalúa en el presente proyecto: El *G-BRIEF (Global Binary Robust Independent Elementary Features).*

El descriptor G-BRIEF fusiona las características más ventajosas de los descriptores descritos en la sección anterior:

- Al igual que el BRIEF, el G-BRIEF es un descriptor binario.
- Al igual que el Tiny images, el G-BRIEF es un descriptor global.

La combinación de estas dos características nos proporciona un descriptor muy eficiente. Ya que al tratarse de un descriptor binario el espacio de memoria ocupado es muy reducido, a la vez que el tiempo de ejecución de tareas binarias es mucho menor. Por otra parte, cada imagen está representada por un único descriptor G-BRIEF, nótese la diferencia con el descriptor BRIEF, que al ser local, necesita varios descriptores para la definición de una imagen. Esta característica también nos proporciona un importante ahorro en espacio de almacenamiento, así como en tiempo de ejecución.

Estas características, hacen que el descriptor G-BRIEF sea especialmente interesante para aplicaciones que involucren grandes cantidades de imágenes. Como sucede en el reconocimiento de escenas realizado por Google imágenes.

El método de obtención del descriptor G-BRIEF es similar al explicado para el descriptor BRIEF, con la salvedad de que en este caso implementaremos un único descriptor para cada imagen. En la *Figura 10* puede observarse como a partir de la imagen (I) de la izquierda se forma un único descriptor G-BRIEF (en la parte derecha de la figura). Nótese la diferencia con la *Figura 8* en la que la imagen (I) queda representada por varios descriptores BRIEF.

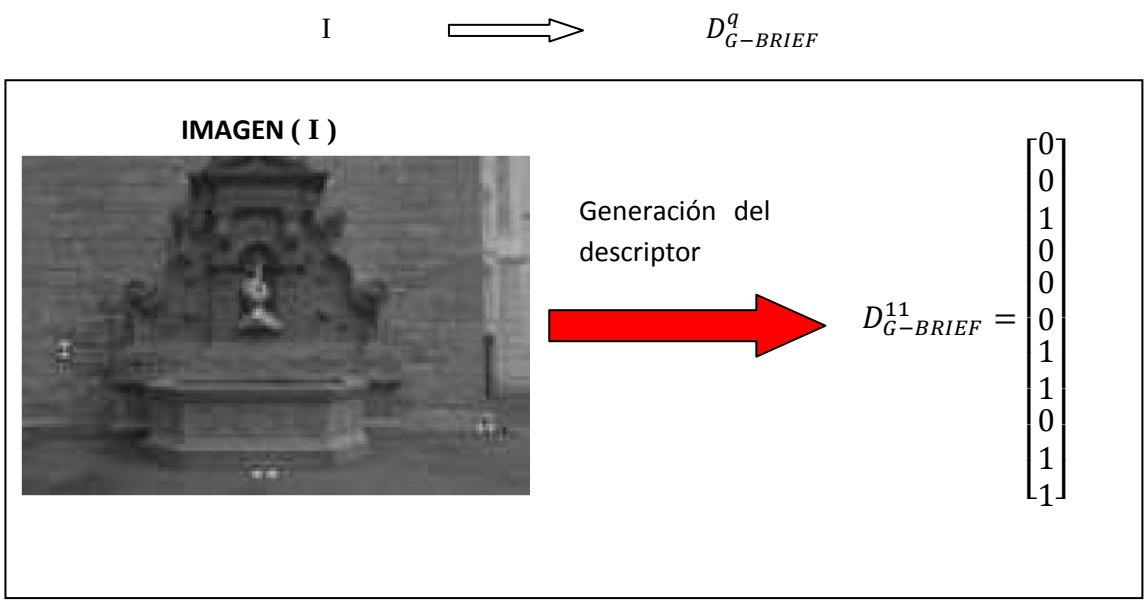

Figura 10: Ejemplo ilustrativo, en el que se observa cómo a partir de una imagen (I) se obtiene un único descriptor G-BRIEF.

Para la formación del descriptor G-BRIEF, en primer lugar, se seleccionan de forma aleatoria las parejas de pixeles (x^I, y^I) en la imagen completa I. Tal y como se ha explicado en la sección anterior, el número de parejas elegidas determinara el tamaño del descriptor obtenido:

$$x^I = \{x_1^I, x_2^I, x_3^I, \dots\dots\dots, x_{q-1}^I, x_q^I\}$$

$$y^I = \{y_1^I, y_2^I, y_3^I, \dots\dots\dots, y_{q-1}^I, y_q^I\}$$

Donde:

- q: Es el número de parejas de pixeles seleccionadas para la formación del descriptor. A su vez, define el tamaño del descriptor, para el que se eligen potencias binarias como 128, 256, 512, etc.
- x^R *e* y^R*:* son los vectores que contienen el primer y el segundo integrante de las parejas de pixeles procedentes de la región R.

Una vez tenemos seleccionadas las q parejas de pixeles (x^I, y^I), se compara la intensidad del primer pixel de la pareja, $p(x^I)$, con la del segundo $p(y^I)$. Obteniéndose el descriptor de la aplicación de la siguiente regla:

Para $i = 1$ hasta $i = q$

$$D_{G-BRIEF}^q = \begin{cases} 1 \; si \; p(x_i^I) > \; p(y_i^I) \\ 0 \qquad en \; otro \; caso \end{cases}$$

Donde:

- $D_{G-BRIEF}^q$: Es el descriptor G-BRIEF de longitud q
- i : Es el índice que recorre todas las posiciones del vector descriptor $D_{G-BRIEF}^q$
- p(x_i): Es la intensidad del primer pixel de la pareja
- p(y_i): Es la intensidad del segundo pixel de la pareja

Una vez finalizado este bucle, se obtiene el descriptor G-BRIEF que representa la imagen I. Nótese la diferencia con la sección anterior, en la que la imagen I es representada por varios descriptores BRIEF.

2.7 SUAVIZADO DE IMÁGENES: EL FILTRO GAUSSIANO

Uno de los principales inconvenientes de los descriptores binarios estriba en el modo en el que se seleccionan los pixeles tenidos en cuenta. Ya sean seleccionados de forma aleatoria, o siguiendo una distribución estadística, nos podemos encontrar con que hemos seleccionado un pixel con una intensidad muy diferente a la de sus vecinos. Ésto puede deberse al ruido presente en la imagen, o al excesivo nivel de detalle.

Estos ruidos pueden ser eliminados mediante el filtrado (suavizado) de las imágenes. La aplicación del filtro hace que se suavicen los contrastes entre pixeles contiguos, difuminando la imagen y haciéndola más homogénea. De este modo, evitamos que el pixel seleccionado no sea representativo de la zona de la imagen en la que está situado.

En la *Figura 11* se observan los resultados obtenidos al aplicar un filtro Gaussiano (tamaño de la máscara 8x8 pixeles, σ=0.5), sobre una imagen del Taj Mahal con fuerte presencia de ruido. En la parte superior izquierda de la figura observamos la imagen sin filtrar, y a su derecha la ampliación de un trozo de cielo de esta imagen, en la que se observan grandes diferencias de color entre pixeles contiguos. En la parte inferior izquierda de la figura se encuentra la imagen filtrada, y a su derecha, de nuevo tenemos la ampliación de un trozo de cielo. En este caso, se observa una total uniformidad de color entre los pixeles del cielo.

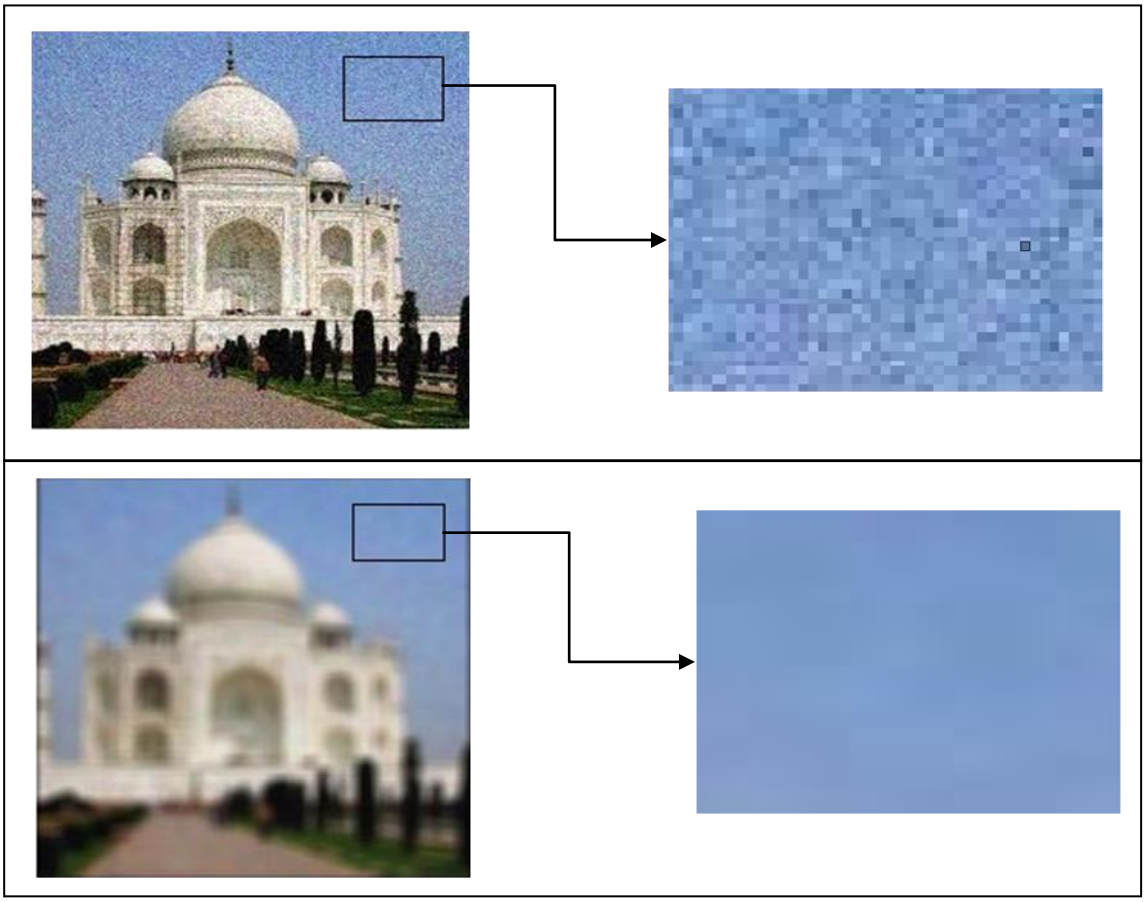

Figura 11: Arriba izquierda: Fotografía del Taj Mahal con importante presencia de ruido. Arriba derecha: Ampliación del cielo donde se observan las grandes diferencias entre pixeles contiguos debido al ruido. Abajo izquierda: Fotografía del Taj Mahal sometida a un filtro Gaussiano (tamaño de la máscara 8x8 pixeles, σ=0.5). Abajo derecha: Ampliación del cielo de la imagen filtrada, donde se observa una uniformidad absoluta en la intensidad de los pixeles.

Como ya sabemos, el descriptor G-BRIEF compara la intensidad de color entre distintos pixeles para obtener información semántica de la imagen. Nótese que las diferencias de color entre los pixeles del cielo de la imagen de arriba, no se corresponden con la información semántica, ya que todos ellos forman parte del cielo. Este ruido hace que obtengamos descriptores de peor calidad, y por lo tanto dificultan la tarea de reconocimiento de escenas.

Obsérvese como en la imagen filtrada todos los pixeles del cielo tienen la misma intensidad de color, de modo que podremos obtener un descriptor que represente mucho mejor el contenido semántico de la imagen.

Mediante un filtrado suficientemente agresivo, conseguimos obtener información acerca de las formas y texturas generales de la imagen, desechando los detalles y ruidos que dificultarían el proceso de reconocimiento.

De los múltiples filtros existentes, el filtro Gaussiano es el que mejor se adecua a nuestros requerimientos.

Matemáticamente el filtrado Gaussiano es la convolución de la imagen con una máscara función de Gauss.

Lo que se consigue aplicando un filtrado Gaussiano es reducir los componentes de alta frecuencia de la imagen (filtro de paso bajo).

La función Gaussiana que define el filtro bidimensional es la siguiente:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Donde:

- x: es la distancia al origen en el eje horizontal.
- y: es la distancia al origen en el eje vertical.
- σ: es la desviación estándar de la función Gaussiana.

La aplicación de esta ecuación produce una superficie cuyos contornos son círculos concéntricos al origen con una distribución Gaussiana. Los valores extraídos de esta superficie se utilizan para construir una matriz de convolución que se aplica a la imagen original. Obteniéndose la imagen filtrada como resultado de la convolución.

En teoría, se debe tener en cuenta la imagen completa para el cálculo de cada uno de los pixeles. En la práctica, cuando se calcula una aproximación discreta de la función de Gauss, se considera que los pixeles situados a una distancia de más de 3σ son despreciables.

A continuación vamos a explicar el funcionamiento del filtrado Gaussiano mediante un ejemplo:

Para comenzar, supondremos que tenemos una imagen de tamaño 5x5 pixeles, con los siguientes valores para cada pixel:

81	85	88	84	96
90	86	85	86	94
81	82	89	72	86
80	76	72	76	84
81	83	80	78	71

Tabla 1: Valores de los pixeles de una imagen 5x5

Para la aplicación del filtro Gaussiano debemos ajustar 3 parámetros:

- La anchura y la altura de la máscara Gaussiana, que determinaran el número de pixeles contiguos a tener en cuenta para calcular en nuevo valor del pixel sometido al filtro (situado en el centro de la máscara).
- σ, desviación estándar de la función Gaussiana, que determina el peso especifico que se le da al valor del pixel central, así como al resto de los pixeles tenidos en cuenta.

En las tablas 2 y 3 se observa la distribución simétrica respecto al centro de valores en la matriz de convolución del filtro Gaussiano.

C	B	C
B	A	B
C	B	C

Tabla 2: Esquema filtro Gaussiano 3x3

F	D	C	D	F
D	C	B	C	D
C	B	A	B	C
D	C	B	C	D
F	D	C	D	F

Tabla 3: Esquema filtro Gaussiano 5x5

El valor que tomaran las celdas A, B, C, D y F dependerá de la σ elegida (por simplicidad el ejemplo será explicado con el filtro 3x3):

$$\sigma = 0{,}5: \quad A = 0.619 \quad B = 0.084 \quad C = 0.011$$

$$\sigma = 1: \quad A = 0.204 \quad B = 0.123 \quad C = 0.075$$

Como podemos observar, cuanto mayor es el valor de la desviación estandar σ, menor es la influencia del propio pixel sometido al filtro y mayor es la de los pixeles contiguos, y por lo tanto más agresivo será el filtro. Del mismo modo que, cuanto mayor tamaño tenga la máscara utilizada, más pixeles influirán en el valor del pixel filtrado, haciendo el filtro más agresivo.

A modo de ejemplo, el valor del pixel sombreado de la *Tabla 1*(valor inicial 89), sometido a un filtro Gaussiano 3x3, σ=0,5 se obtendría de la siguiente forma:

$$P_{filtrado} = 89 \times A + (82 + 72 + 72 + 85) \times \mathrm{B} + (76 + 76 + 86 + 86) \times \mathrm{C} = 84.779$$

Para finalizar, en la *Tabla 4* vamos a mostrar el resultado de aplicar un filtro Gaussiano 3x3, σ=0,5 a la imagen representada en la *Tabla 1*:

65,811	76,004	77,744	76,687	75,353
78,423	85,776	85,562	85,416	82,452
73,127	82,324	84,779	76,640	76,056
71,369	77,396	75,072	76,417	73,257
64,692	72,995	70,802	69,106	58,415

Tabla 4: Resultado obtenido de la aplicación de un filtro Gaussiano 3x3, σ=0,5 sobre la imagen de la tabla1

Dado que el ejemplo ha sido resuelto para una supuesta imagen de dimensiones muy reducidas, en la *Figura 12* se muestran los resultados de la aplicación del filtro Gaussiano sobre una imagen procedente del SUN Data Base [4], para distintos valores de tamaño de la máscara Gaussiana y desviación estándar (σ). Puede observarse como un mayor valor de desviación estándar, así como una máscara más grande generan una imagen filtrada más difuminada.

Figura 12: Efectos del filtrado Gaussiano sobre una imagen procedente del SUN Data Base [4].Nótese que un mayor valor de sigma acompañado de una máscara más grande generan una imagen filtrada más difuminada.

3. DATA SET: SUN DATA BASE

Una vez definido el descriptor G-BRIEF, es necesario evaluar su funcionamiento, así como comparar los resultados obtenidos con el estado del arte.

Para ello necesitamos una base de datos de imágenes sobre la que probar tanto nuestro descriptor como otros descriptores del estado del arte. Y de este modo poder comparar nuestro descriptor con el estado del arte.

Existe la posibilidad de crear una base de imágenes estándar (data set) propia, sin embargo, en este proyecto se ha elegido una de las existentes. Ya que este data set nos proporciona muchos resultados del estado del arte con los que compararnos.

La base de imágenes estándar (o data set) elegida es la Sun data base, que posee dos secciones diferentes. Una para la detección de objetos y otra para el reconocimiento de escenas, en nuestro caso utilizaremos la segunda de ellas: SUN397 scene benchmark [4].

Esta base de datos contiene un total de 108.754 imágenes, organizadas en 397 categorías (Casa, oficina, playa, etc.). En la *Figura 13* se observan algunas de las categorías presentes en la SUN397 scene benchmark [4].

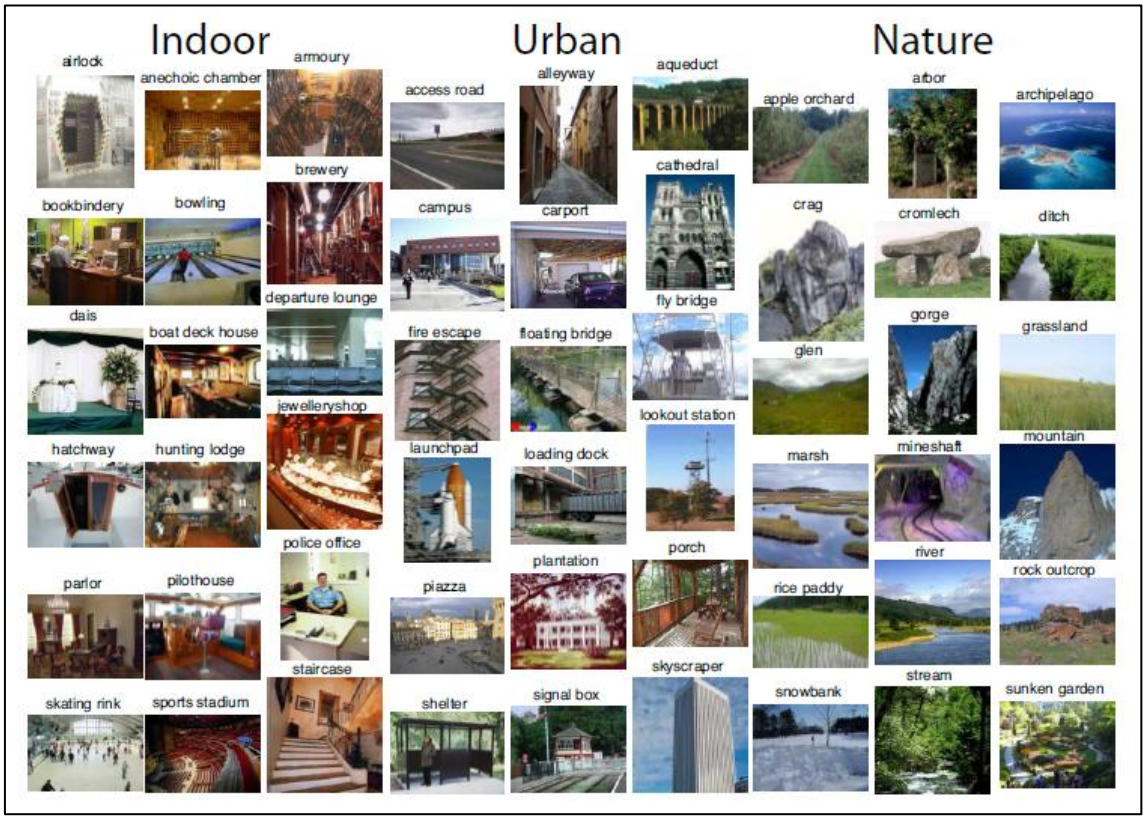

Figura 13: Algunas de las categorías presentes en la base de datos SUN397 scene benchmark [4].

Con la intención de que los experimentos sean lo más representativos e independientes posible, se nos propone utilizar una selección de imágenes organizadas por bloques, tal y como se explica a continuación.

En total se proponen diez bloques, de modo que repetiremos cada experimento 10 veces (una por bloque). Cada bloque está formado por 39.700 imágenes, divididas en dos grupos:

- 19.850 (50 por categoría) Imágenes de entrenamiento (Training), correctamente clasificadas por categoría. Estas imágenes nos sirven para generar el clasificador.

- 19.850 (50 por categoría) Imágenes de prueba (Test). La categoría de estas imágenes es desconocida, el objetivo es clasificarlas basándonos en el clasificador creado con las imágenes de entrenamiento. No obstante, con el objetivo de poder evaluar los resultados, y saber cuando el clasificador nos proporciona resultados correctos o incorrectos, en éstos data sets también conocemos la categoría de las imágenes de Test.

De los datos anteriores se extrae la gran cantidad de operaciones a realizar en cada ejecución, a modo ilustrativo, un solo experimento con el primer bloque de Test y Training conlleva 394.022.500 comparaciones de imágenes. Por lo que la eficiencia del método es crucial para que el tiempo y los recursos de computación sean asequibles.

En la *Figura 14* se observa un ejemplo ilustrativo del proceso de asignación de categoría a una imagen de Test, mediante la comparación con las imágenes de Training.

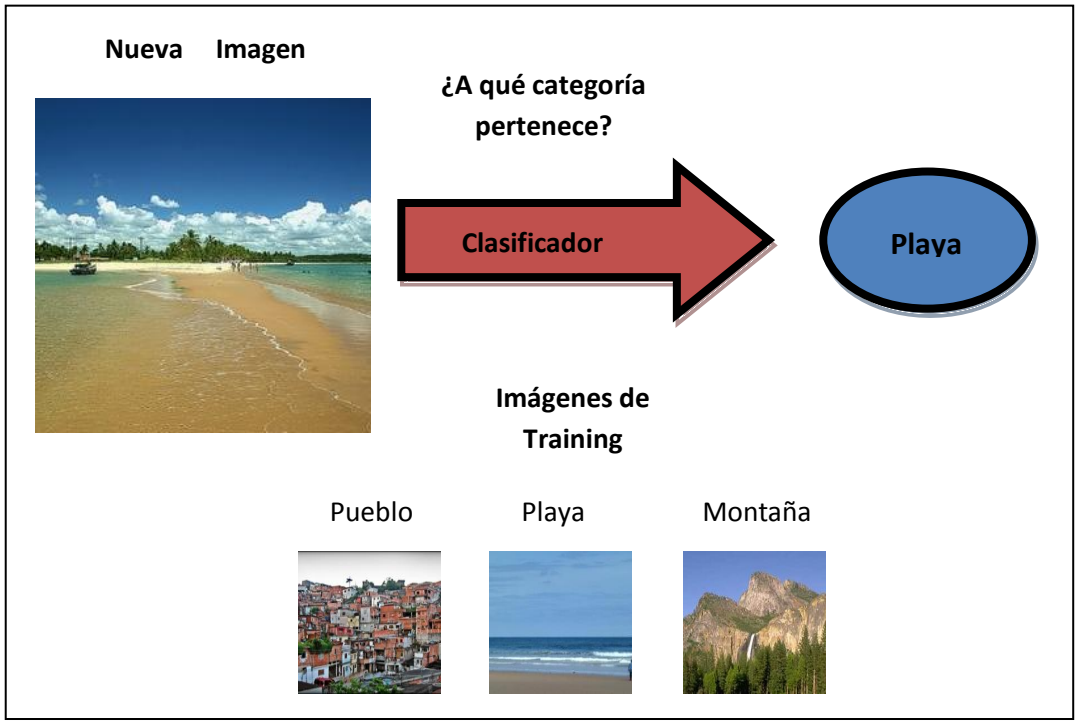

Figura 14: Ejemplo ilustrativo del proceso de asignación de categoría. La nueva imagen de la izquierda es comparada con las imágenes de Training de la base de datos, dependiendo con que categoría tenga más similitud será asignada a una u otra.

4. ALGORITMO DE CLASIFICACIÓN

Una vez definidos tanto los descriptores, como el data set, el siguiente paso es definir el algoritmo que va a servirnos para asignar categoría a las imágenes de Test.

El algoritmo empleado en nuestros experimentos es el conocido como k-vecinos más cercanos (K-nearest neighbors).

Este algoritmo utiliza los descriptores de las imágenes de entrenamiento (Training), para decidir a qué categoría pertenece la nueva imagen de Test.

Sean los descriptores vectores de q dimensiones:

$$D^q_{Training1} = \left(d^1_{Training1}, d^2_{Training1}, \dots, d^q_{Training1}\right) \in D$$

$$D^q_{Test1} = \left(d^1_{Test1}, d^2_{Test1}, \dots, d^q_{Test1}\right) \in D$$

Donde,

- D: es el espacio de q dimensiones, donde se sitúan los descriptores.
- $D^q_{Training1}$: Es el descriptor de longitud q, de la primera imagen de Training.
- D^q_{Test1}: Es el descriptor de longitud p, de la primera imagen de Test.
- q: Es la dimensión del vector descriptor, para poder realizar el emparejamiento de descriptores es necesario que los descriptores de imágenes de Test y Training tengan las mismas dimensiones.

Para el cálculo de la distancia entre descriptores, el hecho de que los G-BRIEF sean binarios nos proporciona una importante ventaja. Ya que nos permite calcular ésta mediante la distancia de Hamming, evitando calcular la distancia euclídea (mucho más costosa).

La distancia de Hamming es el número de bits que difieren entre un descriptor y otro:

$$d_{Test-Training} = W\left(D^q_{Test} \neq D^q_{Training}\right) = \sum_{i=1}^{q} \begin{bmatrix} d^1_{Test} \neq d^1_{Training} \\ d^2_{Test} \neq d^2_{Training} \\ \vdots \\ d^{i-1}_{Test} \neq d^{i-1}_{Training} \\ d^i_{Test} \neq d^i_{Training} \end{bmatrix}$$

Donde,

- $d_{Test-Training}$: Es la distancia de Hamming entre un descriptor de Test genérico y uno de Training, ambos de igual longitud p.

La distancia de Hamming tiene las siguientes propiedades:

- $d_{A-B} = d_{B-A}$
- $d_{A-B} = 0$ si y solo si $A = B$
- $d_{A-B} + d_{B-C} \geq d_{A-C}$

A modo de ejemplo, supongamos que queremos calcular la distancia de Hamming entre dos vectores(A y B) de 4 componentes:

Sean: A=(0 1 0 1) y B=(1 1 1 1)

$$d_{A-B} = \sum \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = 2$$

Por lo que el número de bits que difieren entre A y B son dos.

Una vez que conocemos las distancias entre el descriptor de la nueva imagen de Test, y todos los descriptores de las imágenes de Training, tenemos dos posibilidades:

- Asignar a la nueva imagen de Test la categoría resultante de tener en cuenta los k vecinos más cercanos.
- Asignar a la nueva imagen de Test, la categoría de la imagen de Training más cercana, es decir, mínima distancia de Hamming entre sus descriptores. Este es el caso particular cuando k es igual a 1.

Con el objetivo de conseguir los mejores resultados posibles, se tienen en cuenta las distancias ponderadas entre los k vecinos más cercanos, dando mayor peso a los descriptores más cercanos.

$$\hat{C}(Test) = arg\ max_{v \in V} \sum_{i=1}^{k} w_i \delta(v, C(Training_i))$$

Donde,

- $\hat{C}(Test)$: es la categoría asignada a la nueva imagen de Test.
- v es una determinada categoría del conjunto total de categorías V
- k: es el número de vecinos más cercanos tenidos en cuenta.
- $w_i = \frac{1}{d(Training_i, Test)}$: Es la inversa de la distancia de Hamming entre la imagen de Test y la imagen de Training que ocupa la posición i en el vector de vecinos más próximos.
- δ (Ci,Cj) es una función de selección, que es igual a 1 si las categorías Ci y Cj son la misma, y es cero en cualquier otro caso.

- $C(Training_i)$: es la categoría de la imagen de Training que ocupa la posición i , en el vector de vecinos más próximos.

Esta ecuación recorre todas las categorías de $v \in V$. Para cada categoría, suma los pesos de los vecinos más próximos ponderados con la inversa de su distancia de Hamming si el vecino más próximo ha elegido dicha categoría (función de selección delta). La categoría cuya suma de pesos sea máxima es la categoría asignada a la imagen de test.

5. MEDIDAS PARA LA EVALUACIÓN DEL RECONOCIMIENTO DE ESCENAS

Uno de los métodos más utilizados para valorar la calidad del funcionamiento de un método en el reconocimiento de escenas es la curva de Precision-Recall.

Para explicar este método es necesario introducir los términos: Verdadero positivo, Falso positivo, Verdadero negativo y Falso negativo. Los términos positivo y negativo se refieren a la predicción del clasificador, mientras que los términos verdadero y falso se refieren a si esa predicción se corresponde con la realidad. (*Véase Tabla 5)*

	Categoría real	
Categoría predicha por el clasificador	**Verdadero positivo (tp)** Predicción: Positivo Realidad: Positivo	**Falso positivo(fp)** Predicción: Positivo Realidad: Negativo
	Falso negativo (fn) Predicción: Negativo Realidad: Positivo	**Verdadero negativo (tn)** Predicción: Negativo Realidad: Negativo

Tabla 5: Definición de: tp, fp, tn, fn.

En este contexto se define la Precision como el porcentaje de verdaderos positivos entre el total de los positivos predichos.

$$Precision = \frac{tp}{tp + fp}$$

Recall se define como el porcentaje de verdaderos positivos, entre el total de los positivos reales.

$$Recall = \frac{tp}{tp + fn}$$

La curva precision recall se construye con distintos puntos de precision-recall variando un parámetro de threshold (las imágenes cuya distancia de Hamming sea inferior al threshold representan positivos, mientras que el resto de imágenes representan negativos). Si el threshold es muy bajo el número de positivos es muy bajo (de los que la gran mayoría son verdaderos positivos, siendo escasos los falsos positivos), mientras que el número de falsos negativos es muy alto. Precision=1 y recall=0

En el caso contrario, cuando el threshold es muy alto el número de negativos es muy bajo (de los cuales la gran mayoría son verdaderos negativos, siendo pocos los falsos negativos), mientras que el número de falsos positivos es muy alto. Precision=0 y recall=1

Cuanto mayor sea el área bajo la curva Precision-Recall, mejor será el método. En el caso ideal (fp=0, fn=0), el área será unitaria.

A continuación en la *Figura 15* se muestra un ejemplo de curva de Precision-Recall:

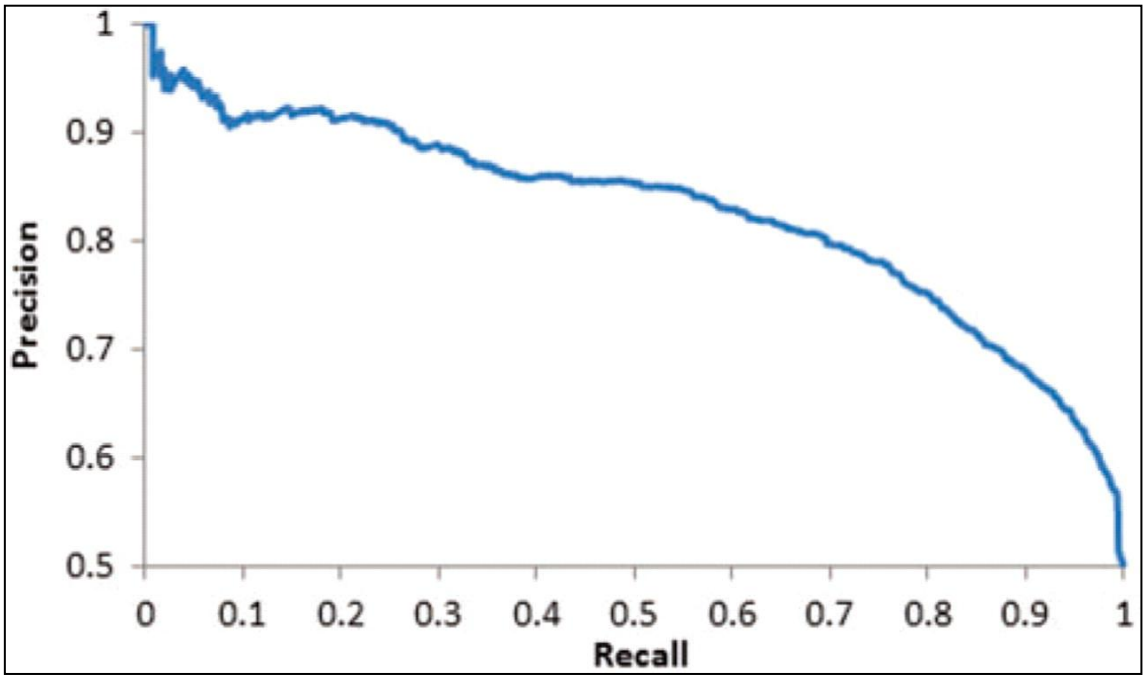

Figura 15: Ejemplo de curva Precision-Recall

6. RESULTADOS EXPERIMENTALES

A continuación se exponen los resultados obtenidos en los experimentos realizados con el descriptor G-BRIEF, haciendo uso de SUN DATABASE y empleando el método descrito en la sección 4.

6.1 EXPERIMENTO 1: IMAGEN EN BLANCO Y NEGRO O ELECCIÓN DE UN CANAL.

Como ya sabemos, las imágenes digitales en color (formato RGB) están formadas por tres canales (R,G y B), de modo que cada pixel tiene un valor para cada uno de los canales.

En este experimento se comparan los resultados obtenidos utilizando los datos del primer canal, frente a los obtenidos utilizando la imagen en escala de grises (formada por un único canal).

El experimento se ha realizado para los 10 bloques de imágenes propuestos en el SUN DATABASE. En la Tabla 6 se recogen los resultados obtenidos:

1 CANAL	**Resultados descriptor G-BRIEF de 64 bits**											
		Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Test 6**	**Test 7**	**Test 8**	**Test 9**	**Test 10**	**Promedio**
	Nº aciertos	132	117	124	123	153	134	130	115	139	133	130
	% de éxito	0,665%	0,589%	0,625%	0,620%	0,771%	0,675%	0,655%	0,579%	0,700%	0,670%	0,655%
	Resultados descriptor G-BRIEF de 128 bits											
	Nº aciertos	179	156	149	160	174	185	169	177	173	174	169,6
	% de éxito	0,902%	0,786%	0,751%	0,806%	0,877%	0,932%	0,851%	0,892%	0,872%	0,877%	0,854%
BLANCO Y NEGRO	**Resultados descriptor G-BRIEF de 64 bits**											
	Nº aciertos	149	143	137	123	159	137	156	147	140	135	142,6
	% de éxito	0,751%	0,720%	0,690%	0,620%	0,801%	0,690%	0,786%	0,741%	0,705%	0,680%	0,718%
	Resultados descriptor G-BRIEF de 128 bits											
	Nº aciertos	194	173	173	172	194	171	169	175	182	189	179,2
	% de éxito	0,977%	0,872%	0,872%	0,866%	0,977%	0,861%	0,851%	0,882%	0,917%	0,952%	0,903%

Tabla 6: Resultados obtenidos al comparar descriptores G-BRIEF de 64 y 128 bits obtenidos a partir del primer canal de las imágenes RGB con los resultados obtenidos de descriptores G-BRIEF procedentes de las imágenes en escala de grises.

En la *Figura 16* pueden observarse los resultados obtenidos para un descriptor G-BRIEF de 64 bits (arriba) y de 128 bits (abajo). En color azul se muestran los resultados obtenidos al generar el descriptor a partir de la información de los pixeles del primer canal de las imágenes RGB, en rojo los resultados utilizando la información de los pixeles de la imagen en escala de grises.

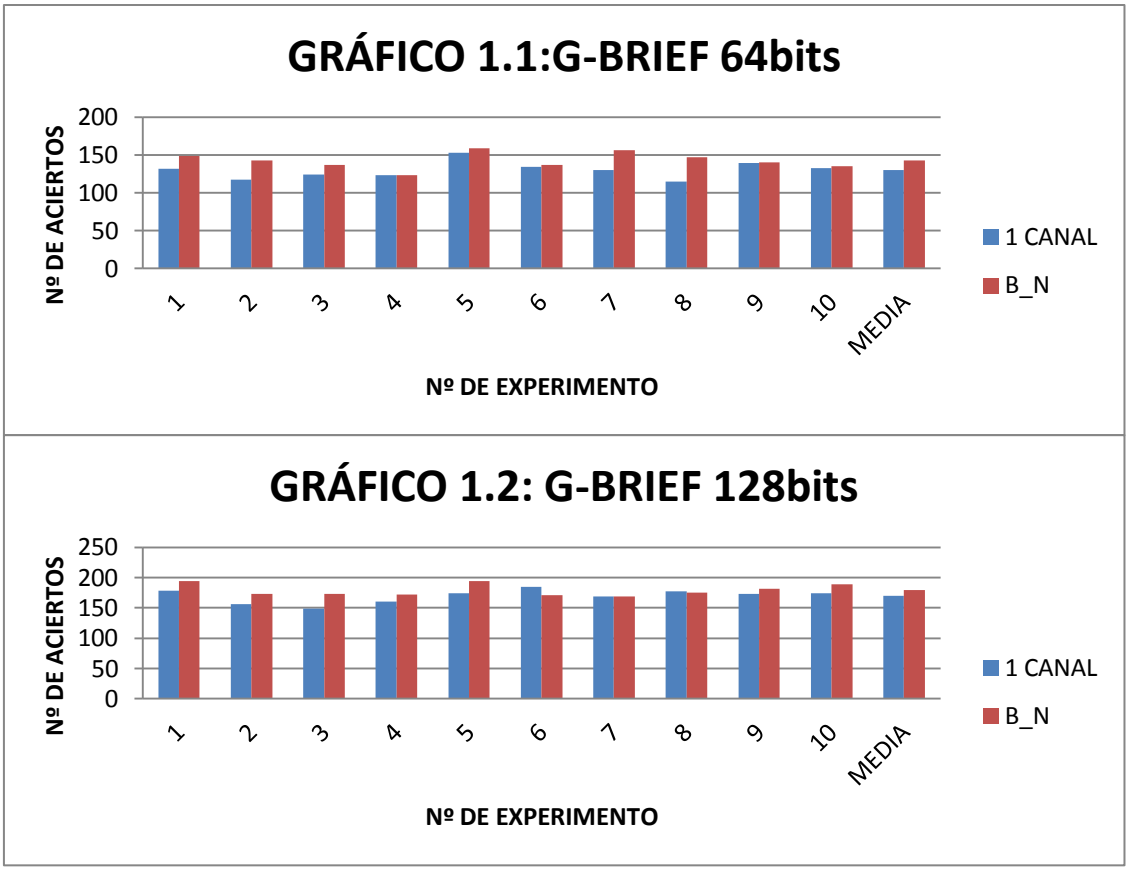

Figura 16: Comparación de resultados obtenidos para un descriptor G-BRIEF de 64 bits (arriba) y de 128 bits (abajo). En color azul se muestran los resultados obtenidos al generar el descriptor a partir de la información de los pixeles del primer canal de las imágenes RGB, en rojo los resultados utilizando la información de los pixeles de la imagen en escala de grises.

Como puede observarse, se obtienen mejores resultados utilizando las imágenes en escala de grises (B_N), y por ello, son las empleadas en los experimentos que se presentan a continuación.

6.2 EXPERIMENTO 2: TAMAÑO ÓPTIMO DEL DESCRIPTOR

Del experimento anterior se intuye que el número de bits del descriptor G-BRIEF influye en los resultados obtenidos.

Con el experimento 2 buscamos obtener el tamaño de descriptor que mejores resultados nos va a proporcionar, teniendo en cuenta criterios de eficiencia y efectividad.

En la *Tabla 7* se presentan los resultados obtenidos para Este experimento, detallando los obtenidos para cada uno de los 10 bloques de imágenes, así como el promedio de ellos.

Resultados descriptor G-BRIEF de 64 bits											
	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Test 6**	**Test 7**	**Test 8**	**Test 9**	**Test 10**	**Promedio**
Nº aciertos	149	143	137	123	159	137	156	147	140	135	142,6
% de éxito	0,751%	0,720%	0,690%	0,620%	0,801%	0,690%	0,786%	0,741%	0,705%	0,680%	0,718%
Resultados descriptor G-BRIEF de 128 bits											
Nº aciertos	194	173	173	172	194	171	169	175	182	189	179,2
% de éxito	0,977%	0,872%	0,872%	0,866%	0,977%	0,861%	0,851%	0,882%	0,917%	0,952%	0,903%
Resultados descriptor G-BRIEF de 256 bits											
Nº aciertos	251	244	206	219	226	230	249	204	230	248	230,7
% de éxito	1,264%	1,229%	1,038%	1,103%	1,139%	1,159%	1,254%	1,028%	1,159%	1,249%	1,162%
Resultados descriptor G-BRIEF de 512 bits											
Nº aciertos	263	282	257	229	291	305	276	258	255	272	268,8
% de éxito	1,325%	1,421%	1,295%	1,154%	1,466%	1,537%	1,390%	1,300%	1,285%	1,370%	1,354%
Resultados descriptor G-BRIEF de 1024 bits											
Nº aciertos	284	281	283	266	284	295	303	262	313	300	287,1
% de éxito	1,431%	1,416%	1,426%	1,340%	1,431%	1,486%	1,526%	1,320%	1,577%	1,511%	1,446%
Resultados descriptor G-BRIEF de 2048 bits											
Nº aciertos	283	282	309	288	299	320	300	291	286	335	299,3
% de éxito	1,426%	1,421%	1,557%	1,451%	1,506%	1,612%	1,511%	1,466%	1,441%	1,688%	1,508%

Tabla 7: Resultados obtenidos para descriptores G-BRIEF de diferentes tamaños.

A continuación, en la *Figura 17* pueden observarse gráficamente, los resultados mostrados en la *Tabla 7*.

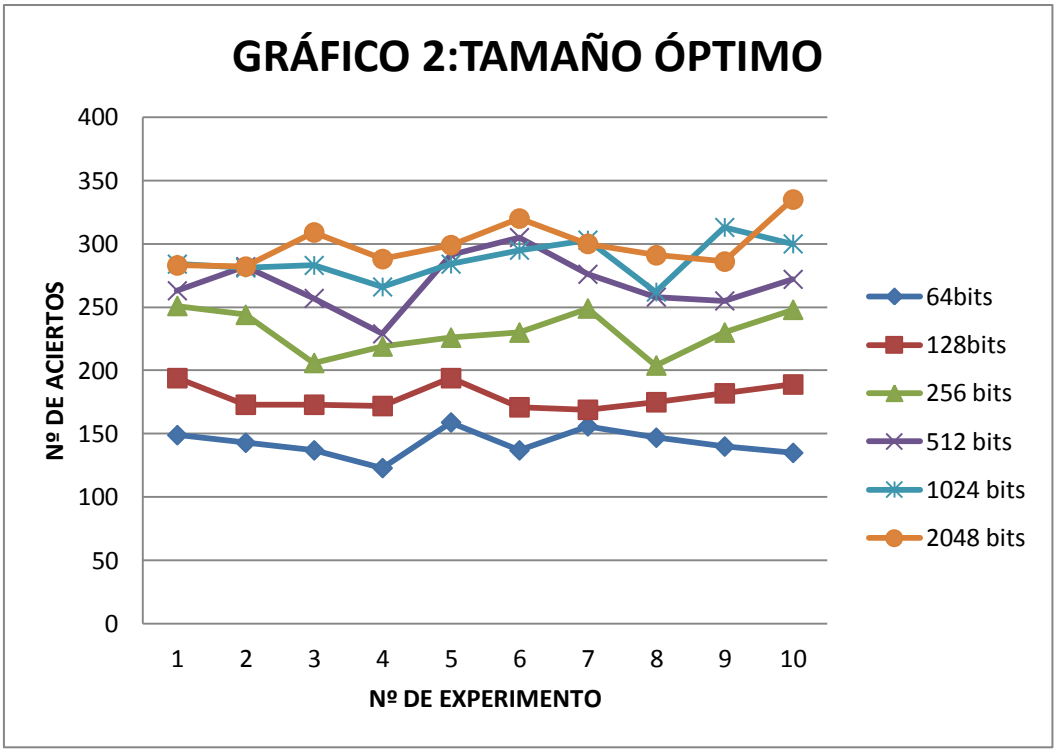

Figura 17: Resultados obtenidos para descriptores G-BRIEF de diferentes tamaños.

Como podemos observar en el gráfico de la *Figura 17*, los descriptores de menor tamaño proporcionan resultados peores. Sin embargo, las diferencias entre los resultados obtenidos para descriptores de 512, 1024 y 2048 bits son ya muy pequeñas.

Teniendo en cuenta ésto, y que el tamaño del descriptor es directamente proporcional al tiempo de ejecución del algoritmo de reconocimiento, hemos elegido como descriptor de tamaño óptimo, el de 1024 elementos. Al considerar que la relación coste/beneficio de aumentar el tamaño del descriptor a 2048 elementos, no es rentable.

6.3 EXPERIMENTO 3: FILTRADO DE IMAGEN

Tal y como se ha explicado en la sección 2.7, el filtrado de imágenes debería mejorar las prestaciones de los descriptores, proporcionando un mayor número de aciertos. Algo que se demuestra con este experimento.

En concreto, con este experimento se quiere conocer el nivel de filtrado de las imágenes que proporciona mejores resultados. Para ello se han calculado los descriptores G-BRIEF de las imágenes sometidas a distintos filtros Gaussianos. Cuyos parámetros modificables, como se explica en la sección 2.7, son el tamaño de la máscara Gaussiana y la desviación estándar.

Dado que el nivel de filtrado y el tamaño del descriptor se consideran parámetros independientes en el reconocimiento de escenas, este experimento se ha realizado únicamente con descriptores G-BRIEF de 256 bits. Sin embargo, los resultados obtenidos son válidos para cualquier tamaño de descriptor.

En la *Tabla 8* se muestran los resultados obtenidos para los diferentes filtros Gaussianos probados. Al igual que en las secciones anteriores, el experimento ha sido repetido para los 10 bloques de imágenes propuestos en SUN DATABASE [4].

IMAGEN NO FILTRADA											
	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Promedio
Nº aciertos	251	244	206	219	226	230	249	204	230	248	230,7
% de éxito	1,264%	1,229%	1,038%	1,103%	1,139%	1,159%	1,254%	1,028%	1,159%	1,249%	1,162%
IMAGEN FILTRADA : MÁSCARA 3X3; σ=1											
Nº aciertos	270	224	216	206	250	247	237	241	246	257	239,4
% de éxito	1,360%	1,128%	1,088%	1,038%	1,259%	1,244%	1,194%	1,214%	1,239%	1,295%	1,206%
IMAGEN FILTRADA : MÁSCARA 8X8; σ =5											
Nº aciertos	272	258	241	231	259	260	240	260	271	263	255,5
% de éxito	1,370%	1,300%	1,214%	1,164%	1,305%	1,310%	1,209%	1,310%	1,365%	1,325%	1,287%
IMAGEN FILTRADA : MÁSCARA 15X15; σ =15											
Nº aciertos	287	279	242	240	277	269	270	251	269	301	268,5
% de éxito	1,446%	1,406%	1,219%	1,209%	1,395%	1,355%	1,360%	1,264%	1,355%	1,516%	1,353%
IMAGEN FILTRADA : MÁSCARA 20X20; σ =15											
Nº aciertos	259	272	256	255	286	276	269	249	251	283	265,6
% de éxito	1,305%	1,370%	1,290%	1,285%	1,441%	1,390%	1,355%	1,254%	1,264%	1,426%	1,338%
IMAGEN FILTRADA : MÁSCARA 30X30; σ =20											
Nº aciertos	280	286	276	247	267	288	278	258	273	284	273,7
% de éxito	1,411%	1,441%	1,390%	1,244%	1,345%	1,451%	1,401%	1,300%	1,375%	1,431%	1,379%
IMAGEN FILTRADA : MÁSCARA 50X50; σ =50											
Nº aciertos	314	275	271	264	294	301	295	267	288	299	286,8
% de éxito	1,582%	1,385%	1,365%	1,330%	1,481%	1,516%	1,486%	1,345%	1,451%	1,506%	1,445%
IMAGEN FILTRADA : MÁSCARA 70X70; σ =50											
Nº aciertos	316	285	274	272	292	310	301	284	291	292	291,7
% de éxito	1,592%	1,436%	1,380%	1,370%	1,471%	1,562%	1,516%	1,431%	1,466%	1,471%	1,470%
IMAGEN FILTRADA : MÁSCARA 90X90; σ =90											
Nº aciertos	295	282	288	255	256	309	286	287	281	308	284,7
% de éxito	1,486%	1,421%	1,451%	1,285%	1,290%	1,557%	1,441%	1,446%	1,416%	1,552%	1,434%
IMAGEN FILTRADA : MÁSCARA 110X110; σ =110											
Nº aciertos	303	271	271	269	268	270	270	300	317	289	282,8
% de éxito	1,526%	1,365%	1,365%	1,355%	1,350%	1,360%	1,360%	1,511%	1,597%	1,456%	1,425%
IMAGEN FILTRADA : MÁSCARA 130X130; σ =130											
Nº aciertos	303	272	272	247	257	269	239	259	277	297	269,2
% de éxito	1,526%	1,370%	1,370%	1,244%	1,295%	1,355%	1,204%	1,305%	1,395%	1,496%	1,356%
IMAGEN FILTRADA : MÁSCARA 150X150; σ =150											
Nº aciertos	279	275	253	256	260	261	253	254	288	286	266,5
% de éxito	1,406%	1,385%	1,275%	1,290%	1,310%	1,315%	1,275%	1,280%	1,451%	1,441%	1,343%

Tabla 8: Resultados obtenidos para el reconocimiento de escenas, empleando descriptores G-BRIEF de 256 bits obtenidos de imágenes sometidas a diferentes filtros Gaussianos.

Los datos de la *Tabla 8* se representan en el gráfico de la *Figura 18*, debido a la gran cantidad de datos, sólo se han representado los valores promedio de los resultados obtenidos para los diferentes valores de filtrado Gaussiano.

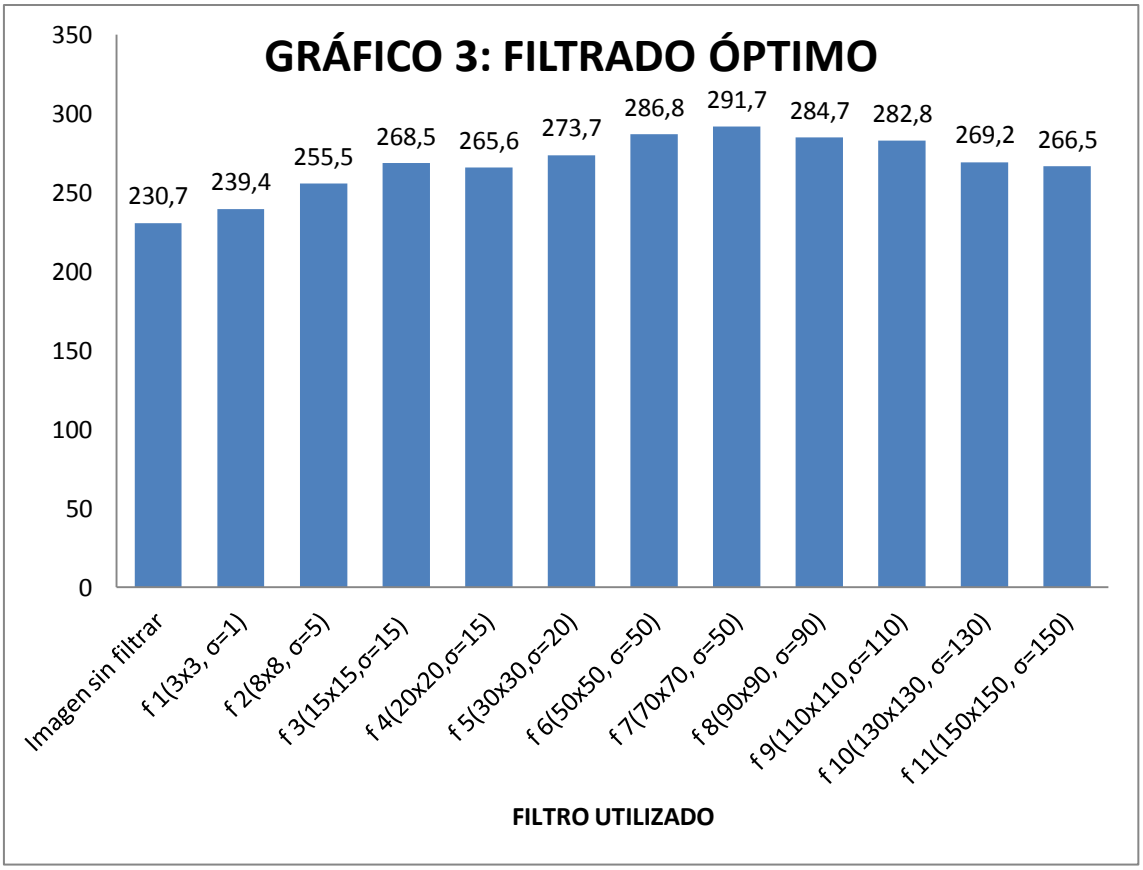

Figura 18: Resultados del experimento de filtrado óptimo. En el gráfico se representa el número de aciertos (promedio de los 10 bloques), para los diferentes niveles de filtrado Gaussiano.

Los resultados del experimento demuestran lo que ya predecíamos en la sección 2.7. Como puede observarse en la *Figura 18*, cuando la imagen no es filtrada o se realiza un pequeño filtrado, los resultados no son óptimos, esto es debido a la presencia de ruido y excesivos detalles en la imagen.

Sin embargo, cuando se realiza un filtro excesivamente agresivo (Máscara 150x150, σ=150) los resultados tampoco son buenos, debido a que el filtrado ha difuminado demasiado la imagen, perdiendo información importante.

Por lo tanto, y a tenor de los resultados, el filtrado óptimo ha de tener un valor intermedio entre estos valores extremos. En este caso, como se puede apreciar en el gráfico de la *Figura 18*, el filtro Gaussiano que proporciona mejores resultados es el que tiene dimensiones de máscara 70x70 pixeles, y desviación estandar σ=50. En concreto, los resultados obtenidos para este filtrado son un 26,4% mejores que los obtenidos a partir de las imágenes sin filtrar.

6.4 EXPERIMENTO 4: Nº ÓPTIMO DE VECINOS MÁS CERCANOS

En la sección 4 se ha presentado el algoritmo de clasificación empleado, para el que debemos elegir el número de vecinos más cercanos que vamos a tener en cuenta.

Con la intención de tomar esta decisión, se han realizado múltiples experimentos para diferentes valores de (k: nº de vecinos más cercanos tenidos en cuenta).

Dado que el número de vecinos más cercanos elegido y el tamaño del descriptor se consideran parámetros independientes en el reconocimiento de escenas, este experimento se ha realizado únicamente con descriptores G-BRIEF de 128 bits. Sin embargo, las conclusiones obtenidas son válidas para cualquier tamaño de descriptor.

En la *Tabla 9* se presentan los resultados obtenidos para los diferentes valores de k (nº de vecinos más cercanos tenidos en cuenta).

k=1											
	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**	**Test 6**	**Test 7**	**Test 8**	**Test 9**	**Test 10**	**Promedio**
Nº aciertos	194	173	173	172	194	171	169	175	182	189	179,2
% de éxito	0,977%	0,872%	0,872%	0,866%	0,977%	0,861%	0,851%	0,882%	0,917%	0,952%	0,903%
k=3											
Nº aciertos	197	177	176	169	193	176	175	174	182	186	180,5
% de éxito	0,992%	0,892%	0,887%	0,851%	0,972%	0,887%	0,882%	0,877%	0,917%	0,937%	0,909%
k=5											
Nº aciertos	196	179	179	172	196	179	184	179	186	191	184,1
% de éxito	0,987%	0,902%	0,902%	0,866%	0,987%	0,902%	0,927%	0,902%	0,937%	0,962%	0,927%
k=10											
Nº aciertos	217	185	177	171	200	180	182	182	204	192	189
% de éxito	1,093%	0,932%	0,892%	0,861%	1,008%	0,907%	0,917%	0,917%	1,028%	0,967%	0,952%
k=15											
Nº aciertos	211	188	168	196	193	194	198	195	222	189	195,4
% de éxito	1,063%	0,947%	0,846%	0,987%	0,972%	0,977%	0,997%	0,982%	1,118%	0,952%	0,984%
k=20											
Nº aciertos	205	186	180	198	209	183	192	205	224	200	198,2
% de éxito	1,033%	0,937%	0,907%	0,997%	1,053%	0,922%	0,967%	1,033%	1,128%	1,008%	0,998%
k=25											
Nº aciertos	213	202	182	203	212	196	192	202	228	206	203,6
% de éxito	1,073%	1,018%	0,917%	1,023%	1,068%	0,987%	0,967%	1,018%	1,149%	1,038%	1,026%
k=30											
Nº aciertos	202	199	182	209	218	194	208	202	234	205	205,3
% de éxito	1,018%	1,003%	0,917%	1,053%	1,098%	0,977%	1,048%	1,018%	1,179%	1,033%	1,034%
k=40											
Nº aciertos	207	204	195	215	240	218	216	206	233	225	215,9
% de éxito	1,043%	1,028%	0,982%	1,083%	1,209%	1,098%	1,088%	1,038%	1,174%	1,134%	1,088%

k=50											
Nº aciertos	222	209	196	201	244	219	239	227	239	230	222,6
% de éxito	1,118%	1,053%	0,987%	1,013%	1,229%	1,103%	1,204%	1,144%	1,204%	1,159%	1,121%
k=60											
Nº aciertos	204	208	191	200	241	218	223	226	234	224	216,9
% de éxito	1,028%	1,048%	0,962%	1,008%	1,214%	1,098%	1,123%	1,139%	1,179%	1,128%	1,093%
k=70											
Nº aciertos	217	208	206	210	247	224	224	235	236	230	223,7
% de éxito	1,093%	1,048%	1,038%	1,058%	1,244%	1,128%	1,128%	1,184%	1,189%	1,159%	1,127%
k=80											
Nº aciertos	225	218	206	229	242	231	207	239	226	247	227
% de éxito	1,134%	1,098%	1,038%	1,154%	1,219%	1,164%	1,043%	1,204%	1,139%	1,244%	1,144%
k=90											
Nº aciertos	225	220	207	215	228	225	216	234	221	229	222
% de éxito	1,134%	1,108%	1,043%	1,083%	1,149%	1,134%	1,088%	1,179%	1,113%	1,154%	1,118%
k=100											
Nº aciertos	239	232	211	221	233	223	221	248	229	233	229
% de éxito	1,204%	1,169%	1,063%	1,113%	1,174%	1,123%	1,113%	1,249%	1,154%	1,174%	1,154%
k=125											
Nº aciertos	228	235	220	220	247	218	228	231	218	256	230,1
% de éxito	1,149%	1,184%	1,108%	1,108%	1,244%	1,098%	1,149%	1,164%	1,098%	1,290%	1,159%
k=150											
Nº aciertos	240	237	203	212	243	238	227	221	211	244	227,6
% de éxito	1,209%	1,194%	1,023%	1,068%	1,224%	1,199%	1,144%	1,113%	1,063%	1,229%	1,147%

Tabla 9: Resultados obtenidos para diferentes valores de K en el método K vecinos más cercanos

Por motivos de claridad, en la *Figura 19* se muestra una selección de los resultados presentes en la *Tabla 9*. En dicha figura se observa cómo a partir de k=50 la mejora en los resultados obtenidos no está clara.

En la *Tabla 10* se ha incluido un resumen de los resultados de este experimento, en dicha tabla se observa como los aciertos promedio de los 10 bloques tiene una clara tendencia ascendente conforme va aumentando k, hasta que llegamos a k=50, dónde se considera que la relación coste/ beneficio de aumentar el valor de k, no es rentable.

Estos resultados pueden visualizarse de forma gráfica en la *Figura 20*.

Todo lo anterior demuestra que el valor óptimo de k es k_{opt}=50. Este valor es dependiente del número de categorías en las que se clasifiquen las imágenes (en nuestro caso 397 categorías), por lo que para un diferente número de categorías, k_{opt} será diferente.

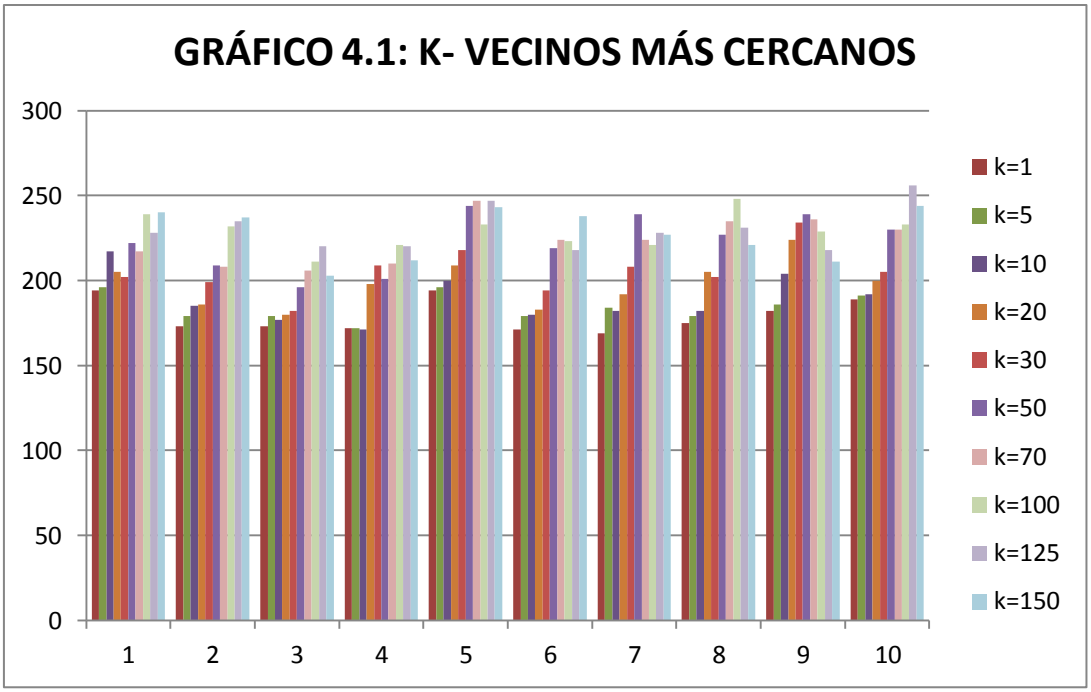

Figura 19: Gráfico de barras que representa los resultados para diferentes valores de k(k-vecinos más cercanos)

k	Aciertos promedio (10 bloques)
1	179,2
3	180,5
5	184,1
10	189
15	195,4
20	198,2
25	203,6
30	205,3
40	215,9
50	222,6
60	216,9
70	223,7
80	227
90	222
100	229
125	230,1
150	227,6

Tabla 10: Tabla resumen de aciertos promedio frente a los diferentes valores de k.

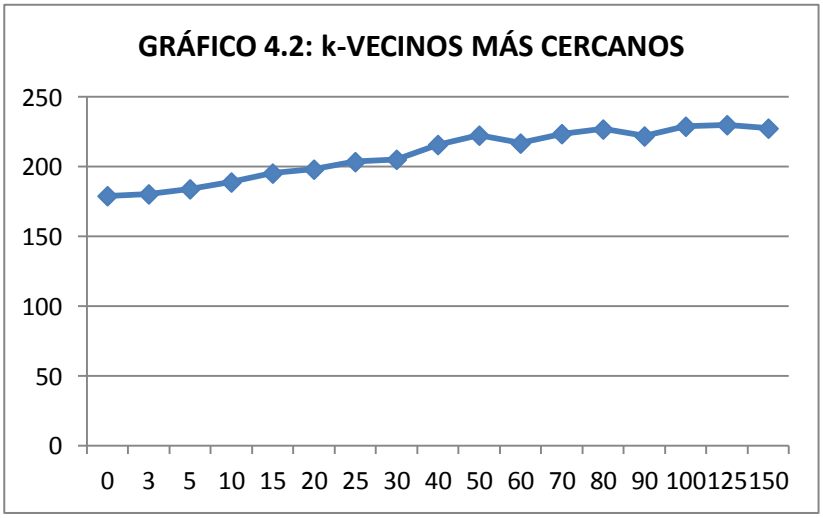

Figura 20: Tendencia de los resultados promedio para diferentes valores de k (k-vecinos más cercanos)

6.5 COMPARACIÓN CON EL ESTADO DEL ARTE (Tiny Images)

Una vez obtenidos los resultados para nuestro descriptor G-BRIEF, nos hemos comparado con el estado del arte. Para ello se ha elegido el Tiny images, ya que es un descriptor orientado a bases de datos de gran tamaño y por tanto necesidades computacionales y de almacenamiento muy bajas.

Uno de los parámetros utilizados para comprar los diferentes descriptores es la curva de Precision-Recall, explicada en la sección 5. Los resultados de dicha curva para ambos descriptores se muestran en la *Figura 21*, en la que se observa un "empate técnico" entre ambos descriptores.

Cabe destacar, que aunque la curva Precision-Recall sea prácticamente igual para ambos descriptores, en cuanto a coste y espacio de almacenamiento el G-BRIEF es más eficiente, y por lo tanto se considera mejor alternativa.

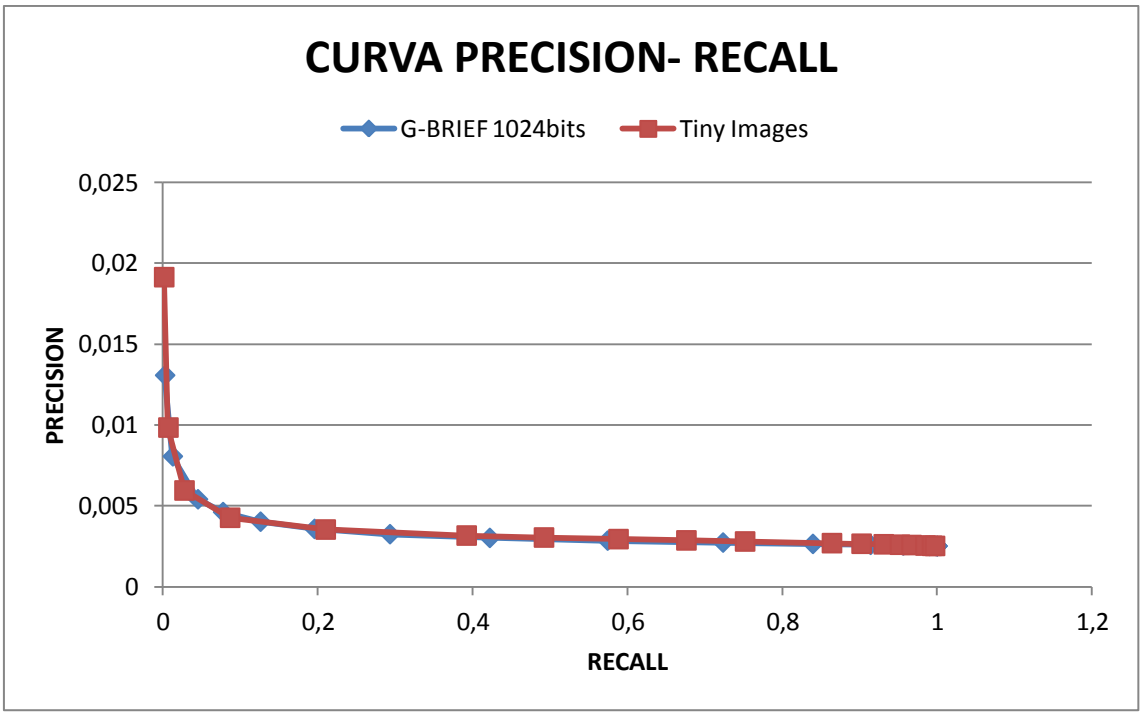

Figura 21: Comparativa de la curva precision/recall del descriptor G-BRIEF y el Tiny images.

La curva de Precision-Recall es un buen método para la comparación de descriptores, pero no el único.

A continuación, en la *Tabla 11* se muestra un resumen de los resultados obtenidos para las tres mejores configuraciones del G-BRIEF, así como los resultados obtenidos para el Tiny images. En dicha tabla se comparan los aciertos obtenidos con cada uno de estos descriptores, obsérvese que la configuración del descriptor G-BRIEF que mejores resultados proporciona es la que se deduce de los experimentos presentados anteriormente.

Los datos de la *Tabla 11* pueden visualizarse de forma gráfica en la *Figura 22*, en la que se observa claramente que los resultados obtenidos por el descriptor G-BRIEF son mejores que

los obtenidos mediante el descriptor Tiny images. En concreto el descriptor G-BRIEF obtenido a partir de la imagen filtrada proporciona resultados un 17,7% mejores que el Tiny images.

		Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	PROMEDIO DE ACIERTOS
G-BRIEF 1024 bits	Nº aciertos	284	281	283	266	284	295	303	262	313	300	**287,1**
	% de éxito	1,43%	1,42%	1,43%	1,34%	1,43%	1,49%	1,53%	1,32%	1,58%	1,51%	1,446%
G-BRIEF 1024: FILTRO 70X70, s=50, k=1	Nº aciertos	355	306	340	312	355	353	347	355	347	385	**345,5**
	% de éxito	1,788%	1,542%	1,713%	1,572%	1,788%	1,778%	1,748%	1,788%	1,748%	1,940%	1,741%
G-BRIEF 1024: FILTRO 70X70, s=50, k=50	Nº aciertos	355	366	352	329	363	380	351	339	354	355	**354,4**
	% de éxito	1,788%	1,844%	1,773%	1,657%	1,829%	1,914%	1,768%	1,708%	1,783%	1,788%	1,785%
TINY IMAGES	Nº aciertos	325	276	270	269	306	277	282	257	304	276	**284,2**
	% de éxito	1,637%	1,390%	1,360%	1,355%	1,542%	1,395%	1,421%	1,295%	1,531%	1,390%	1,432%

Tabla 11: Resumen de los resultados obtenidos para las tres mejores configuraciones del G-BRIEF y el Tiny images.

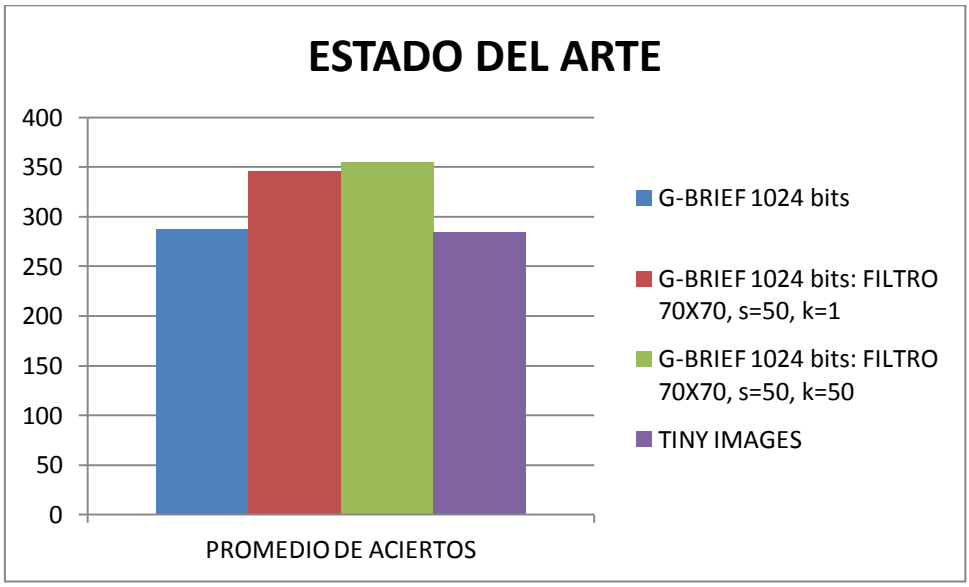

Figura 22: Resumen de los resultados obtenidos para las tres mejores configuraciones del G-BRIEF y el Tiny images

6.6 ESCALABILIDAD DEL DESCRIPTOR G-BRIEF

En este apartado se prueban los resultados de escalabilidad del descriptor G-BRIEF. Como se puede observar en la *Tabla 12*, el descriptor G-BRIEF mejora sus prestaciones cuanto mayor número de categorías de clasificación tenemos.

En esta tabla se comparan los porcentajes de acierto obtenidos mediante el uso del descriptor G-BRIEF con los que se obtendrían de un modo aleatorio.

	descriptor G-BRIEF 1024 bits Filtro 70x70, σ=50													
nº categorias	397		150		50		10		5		3		2	
nº imag total	19850		7500		2500		500		250		150		100	
	aciertos	%	aciertos	%	aciertos	%	aciertos	%	aciertos	%	aciertos	%	aciertos	%
Test 1	355	1,79%	252	3,36%	187	7,5%	128	25,6%	107	42,8%	70	46,7%	61	61%
Test 2	306	1,54%	233	3,11%	179	7,2%	118	23,6%	100	40,0%	76	50,7%	66	66%
Test 3	340	1,71%	234	3,12%	173	6,9%	112	22,4%	105	42,0%	70	46,7%	59	59%
Test 4	312	1,57%	242	3,23%	150	6,0%	120	24,0%	88	35,2%	66	44,0%	52	52%
Test 5	355	1,79%	229	3,05%	171	6,8%	117	23,4%	100	40,0%	76	50,7%	59	59%
Test 6	353	1,78%	254	3,39%	170	6,8%	123	24,6%	107	42,8%	79	52,7%	72	72%
Test 7	347	1,75%	253	3,37%	185	7,4%	132	26,4%	112	44,8%	81	54,0%	66	66%
Test 8	355	1,79%	223	2,97%	158	6,3%	104	20,8%	105	42,0%	75	50,0%	60	60%
Test 9	347	1,75%	244	3,25%	155	6,2%	123	24,6%	107	42,8%	88	58,7%	69	69%
Test 10	385	1,94%	229	3,05%	191	7,6%	116	23,2%	99	39,6%	66	44,0%	61	61%
Promedio	345,5	1,74%	239,3	3,19%	171,9	6,9%	119,3	23,9%	103	41,2%	74,7	49,8%	62,5	63%
Aleatorio	50	0,252%	50	0,667%	50	2,0%	50	10%	50	20%	50	33%	50	50%
G-BRIEF/ Aleatorio		6,91		4,79		3,44		2,39		2,06		1,49		1,25

Tabla 12: Resultados de escalabilidad del descriptor G-BRIEF

Los resultados presentados en la *Tabla 12* son representados en la *Figura 23*.

Si nos fijamos en los resultados obtenidos cuando se deben clasificar las imágenes en 397 categorías (1,74%) pueden parecer muy malos. Sin embargo, si lo comparamos con los resultados que se producirían de un modo aleatorio, los resultados del G-BRIEF son 7 veces mejores, con un consumo de recursos muy reducido.

Como podemos intuir, el descriptor G-BRIEF va a proporcionarnos resultados muy satisfactorios cuando el número de categorías sea muy elevado. Algo que ningún descriptor del estado del arte puede igualar con un consumo de recursos tan escaso.

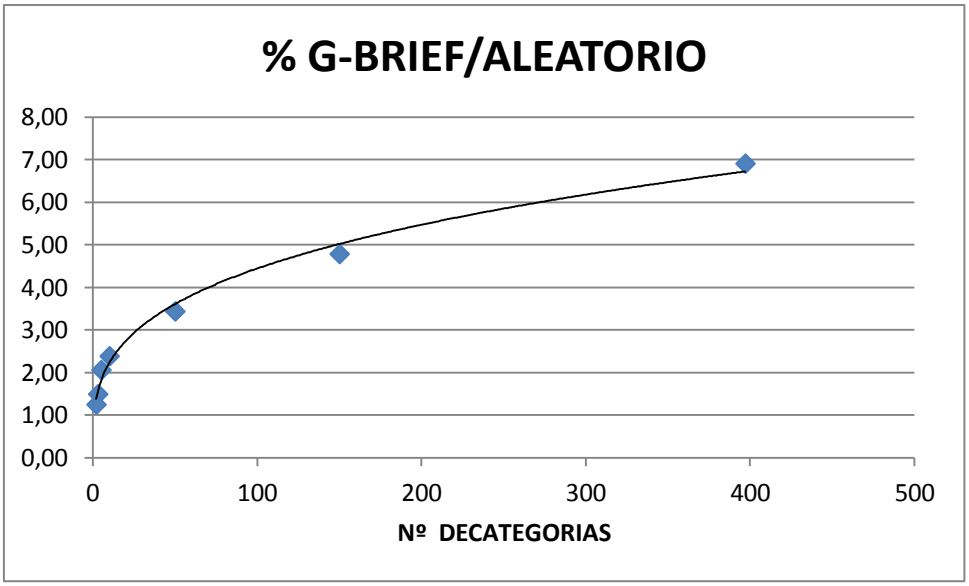

Figura 23: Resultados de escalabilidad del descriptor G-BRIEF

7. CONCLUSIONES

En este proyecto se ha propuesto un nuevo descriptor, el descriptor G-BRIEF, que combina lo mejor de los descriptores globales y los binarios.

Este descriptor ha sido evaluado hasta hallar su configuración óptima. Esta configuración óptima es un descriptor de 1024 bits, con un filtrado Gaussiano previo de las imágenes. Se ha demostrado que el filtrado que proporciona mejores resultados es el que emplea una máscara Gaussiana de 70x70 pixeles y una desviación estándar σ=50. El algoritmo de emparejamiento que se ha empleado es el k- vecinos más cercanos, para el que se ha demostrado que el valor óptimo de vecinos (k) tenidos en cuenta es de cincuenta.

El descriptor G-BRIEF, proporciona un ratio de reconocimiento similar, incluso superior, al descriptor más similar del estado del arte: Tiny images. Con un consumo de recursos (almacenamiento y procesamiento) en el proceso de reconocimiento que puede llegar a ser 8 veces inferior [3].

Como trabajo futuro sería interesante probar el descriptor G-BRIEF en la base de datos Tiny images (de 80 millones de imágenes), y así, comprobar los resultados de escalabilidad de este proyecto.

REFERENCIAS:

[1] **High Detection-rate Cascades for Real-Time Object Detection.**
Hamed Masnadi-Shirazi and Nuno Vasconcelos
Proceedings of *IEEE International Conference on Computer Vision(ICCV),* Rio de Janeiro, Brazil, 2007.

[2] **80 million tiny images: a large dataset for non-parametric object and scene recognition.**
Antonio Torralba, Rob Fergus y William T. Freeman**.**
Journal: IEEE Transactions on Pattern Analysis and Machine Intelligence Volume 30 Issue 11, November 2008.

[3] **BRIEF: Computing a Local Binary Descriptor Very Fast**
M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua
IEEE Transactions on Pattern Analysis and Machine Intelligence 2012

[4] **SUN Database: Large-scale Scene Recognition from Abbey to Zoo**
Jianxiong Xiao , James Hays, Krista, A. Ehinger, Aude Oliva y Antonio Torralba
IEEE Conference on Computer Vision and Pattern Recognition (CVPR2010)