Proyecto Final de Carrera
Ingeniería en Informática
Curso 2011/2012

Guía turística de monumentos basada en reconocimiento visual con un móvil

Javier Arroyo Espallargas

Febrero 2012

Directora: Ana Cristina Murillo Arnal
Codirector: Danilo Tardioli

Departamento de Informática e Ingeniería de Sistemas
Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza

RESUMEN

La visión es una de las fuentes de información más potentes que tenemos las personas. Además, cada vez nos encontramos con más cámaras y grandes bases de datos de imágenes a nuestra disposición. Dada esta situación, el procesamiento automático e inteligente de imágenes esta adquiriendo gran relevancia en el desarrollo de nuevas tecnologías, que basan su desarrollo en técnicas de visión por computador. En particular, en este proyecto el trabajo se ha centrado en las tecnologías aplicables a dispositivos móviles para aprovechar sus cámaras integradas y su cada vez mayor capacidad de computo.

En el presente proyecto se ha desarrollado una aplicación que pueda servir como guía turística aprovechando la cámara de un teléfono móvil, siendo capaz de reconocer diferentes edificios o monumentos mediante el procesamiento de imágenes. En esta aplicación, a diferencia de la gran mayoría de aplicaciones similares, el objetivo es poder realizar todo el procesamiento de la imagen en el dispositivo, mientras que muchas otras aplicaciones hacen uso de la red para poder acceder a sistemas más potentes que realicen los cálculos. Con este objetivo, se han estudiado y desarrollado diferentes técnicas de visión por computador que permitan el reconocimiento de un edificio, haciendo especial hincapié, además de un buen rendimiento en el reconocimiento, en que sean capaces de ejecutarse en tales dispositivos con un tiempo de respuesta aceptable.

El proceso llevado a cabo tiene dos grandes bloques. Por un lado, se ha recopilado una base de datos de imágenes de edificios de interés. Esta información de referencia debe almacenarse de manera adecuada para su posterior uso, para lo cual se ha diseñado un proceso basado en técnicas de extracción de puntos de interés en imágenes.

Por otro lado se han estudiado diferentes métodos para el reconocimiento de la imagen del usuario. Este bloque tiene tres pasos básicos: el usuario captura una imagen de un edificio; la aplicación diseñada, comparará la información de esa imagen con las de referencia, haciendo uso de estructuras de búsqueda de árbol y vocabularios de descriptores de imágenes, muy utilizados en visión por computador. Una vez realizada la comparación, la aplicación devuelve la respuesta al usuario con información relativa al monumento. En este último paso, por un lado se devuelve información textual a cerca del monumento, y por otro lado se ha estudiado e implementado una técnica basada en la geometría entre varias imágenes de una misma escena, que permite añadir información virtual, es decir, aumentar la realidad. Todas las técnicas estudiadas en cada paso han sido evaluadas en el entorno descrito y las configuraciones con mejor rendimiento han sido integradas con una interfaz implementada para la plataforma Android, también desarrollada en este proyecto. El prototipo final tiene unos resultados de reconocimiento buenos y un tiempo de respuesta adecuado para una aplicación que necesita interactuar con un usuario.

Índice general

Capítulo 1

Introducción

Este capítulo describe la motivación que ha llevado al desarrollo de este proyecto, así como la descripción de los objetivos y el alcance del mismo. También se detallarán las herramientas y el entorno en el que se ha realizado.

1.1. Motivación

En la actualidad, el reconocimiento de objetos dentro de bases de datos de imágenes de gran tamaño es un problema de visión por computador con numerosas aplicaciones, en particular en el ámbito de los *smartphones* (véase el ejemplo de la Figura 1.1).

Figura 1.1: Ejemplo de aplicación de reconocimiento visual en *smartphones*: identificación de monumentos.

El término *smartphone* denota a un teléfono móvil que ofrece más funciones que un teléfono convencional, con una mayor capacidad de computo, un sistema operativo más potente que permite instalar aplicaciones y el uso de múltiples sensores integrados en el teléfono. Existen multitud de modelos de diferentes fabricantes, con diferentes Sistemas Operativos (SO). Dependiendo

del fabricante y modelo, los *smartphones* pueden incluir distintos sensores y funcionalidades, como por ejemplo: cámara de fotos y/o vídeo de alta definición integrada, acceso a Internet, *GPS*, acelerómetro, teclado físico y/o una pantalla táctil; así como múltiples aplicaciones ya instaladas, como por ejemplo navegador web, servicios de correo electrónico, editor/lector de textos, etc. En la actualidad, el mercado de los *smartphones* está repartido principalmente entre dos grandes compañías: *Google* con el sistema *Android* [1] y *Apple*[2] con el SO iOS.

En este proyecto se ha tratado de aplicar en estos dispositivos técnicas de visión por computador. La visión por computador es una rama de la inteligencia artificial que pretende dotar a un computador de la capacidad de analizar automáticamente la información contenida en imágenes. La tarea de desarrollar un sistema de visión artificial que imite o sustituya al humano es complicada, ya que la mayoría del procesamiento que realizamos las personas es inconsciente y desconocido, por lo que es difícil de replicar en un sistema informático. Aún así, el creciente número de cámaras presentes en nuestro entorno está posibilitando su uso en diversas aplicaciones, y convierte la visión por computador en una disciplina de creciente y continuo interés dentro del campo científico-técnico. Algunas de sus aplicaciones están destinadas a la robótica, a los procesos de inspección automática, a la navegación autónoma de vehículos, al reconocimiento de objetos/edificios, al filtrado del contenido de imágenes, etc.

El uso de la visión por computador en aplicaciones móviles se puede dividir en dos grandes grupos: aplicaciones que envían y analizan la información de la imagen de manera remota en la *nube*, como por ejemplo Google Goggles[3], o los servicios de Kooaba[4] [11]. Y aplicaciones que realizan todo el procesado de la imagen en el dispositivo móvil como Wordlens[5] o Nintendo DS dictionary y como será nuestro caso.

Como se detalla en el siguiente punto, el objetivo general de este proyecto es implementar un prototipo funcional, que realice todo su procesado en el teléfono móvil, que permita el reconocimiento de diferentes edificios ó monumentos de las ciudades de Zaragoza y Andorra.

1.2. Objetivos y Alcance del proyecto

El objetivo principal de este proyecto ha sido estudiar, desarrollar y evaluar diferentes métodos de visión por computador para el reconocimiento de edificios, creando un algoritmo que trabaje con precisión, de forma eficiente y que posibilite su escalabilidad para permitir la inclusión de más información.

Implementar este algoritmo en un dispositivo móvil *smartphone* no es tarea sencilla y diversos

[1] http://www.android.com/
[2] http://www.developer.apple.com/
[3] http://www.google.com/mobile/goggles/
[4] http://www.kooaba.com/
[5] http://www.questvisual.com/

factores deben ser tenidos en cuenta, algunos específicos a este tipo de aplicaciones. Por ejemplo, requerimientos de memoria, coste computacional o el rendimiento en el reconocimiento, así como una rápida interacción con el usuario. Los objetivos y tareas a realizar son los siguientes:

- Diseñar, implementar y evaluar diferentes métodos de búsqueda de correspondencias en imágenes y evaluación de similitud entre dos imágenes.
- Integrar dichos métodos de comparación de imágenes sobre la plataforma Android.
- Diseñar e implementar un sistema sencillo de realidad aumentada para presentar los resultados del reconocimiento.
- Creación de una base de datos con imágenes de los edificios más emblemáticos de Zaragoza y Andorra.
- Creación de una aplicación sobre Android con una interfaz clara y sencilla para el usuario. El proceso debe generar una respuesta rápida, debe ser eficiente, teniendo en cuenta las capacidades de computo de un *smartphone*.
- Evaluar y documentar los resultados incluyendo rendimiento, coste computacional y uso de memoria.

1.3. Herramientas y Entorno de Trabajo

El desarrollo de este proyecto se ha enmarcado dentro de las líneas de investigación del grupo de Robótica, Percepción y Tiempo Real.

Todo su desarrollo se ha realizado sobre el sistema operativo Ubuntu Linux v10.10[6]. Para la programación de los distintos métodos, sus pruebas y el prototipo de aplicación para Android se ha utilizado Eclipse Indigo[7], con los paquetes para la programación tanto en Java, como en C/C++, así como el SDK[8] y NDK[9] de Android. Para el procesamiento de imágenes se ha utilizado la librería OpenCV v2.3 [1].

1.4. Estructura de la Memoria

En el capítulo 2, se detallan los métodos implementados y los experimentos más importantes realizados relacionados con el reconocimiento visual en imágenes, cuyo objetivo es determinar la imagen más similar entre la imagen tomada por el usuario y las existentes en la base de datos. En el capítulo 3 se explica cómo se ha diseñado e implementado el sistema de realidad aumentada para presentar los resultados del reconocimiento. En el capítulo 4 se describe la

[6] http://www.ubuntu.com/
[7] http://www.eclipse.org/
[8] http://developer.android.com/sdk/
[9] http://developer.android.com/sdk/ndk/

aplicación Android diseñada y su interfaz de usuario. Finalmente se concluye la memoria con un capítulo, de conclusiones y futuros trabajos a realizar a partir de este proyecto.

Los Anexos detallan diversos puntos del proyecto. En el Anexo A están detallados todos los experimentos realizados mediante la representación completa. En el Anexo B se explica de manera más extensa las técnicas de clusterización o bolsa de palabras, cuyos experimentos están detallados en el Anexo C. En el Anexo D se describen los aspectos fundamentales del SO Android, así como detalles prácticos para enlazar las librerías de OpenCV. El Anexo E contiene diferentes ejemplos de pruebas del algoritmo de realidad aumentada implementado. El Anexo F incluye los diagramas de flujo de datos de nuestro sistema de reconocimiento de imágenes. En el Anexo G se explica para qué y cómo se ha utilizado el GPS en este proyecto. Por último, el Anexo H resume como se ha gestionado a lo largo de tiempo el desarrollo de este proyecto fin de carrera.

Capítulo 2

Evaluación Automática de Similitud entre Imágenes

2.1. Introducción

Este capítulo describe la tarea principal estudiada en este proyecto junto con las soluciones propuestas y su evaluación. Esta tarea consiste en obtener un método que permita reconocer un determinado edificio en una imagen nueva adquirida por el usuario, mediante la evaluación de su similitud con respecto a un conjunto de imágenes de referencia, es decir, la búsqueda de la imagen de la base de datos más similar a una tomada por el usuario. Nuestro sistema se ha implementado utilizando un conjunto de imágenes de referencia de 25 edificios de Zaragoza y 14 de Andorra, con múltiples fotografías de cada uno de ellos.

El proceso a realizar se resume en la Figura 2.1. El usuario toma una fotografía con su móvil, y la aplicación mediante un procesado automático, la compara con los diferentes edificios de la base de datos, para terminar mostrando al usuario la imagen más similar encontrada.

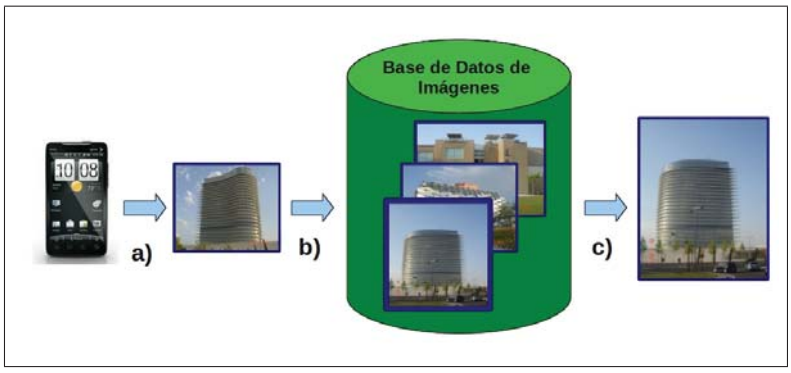


Figura 2.1: **Búsqueda en bases de datos de imágenes.** El usuario toma una foto de un edificio a), esta se procesa b) y se compara con las imágenes de referencia. Como resultado se muestran las imágenes más similares c).

En las siguientes secciones, se resume en primer lugar (Sección 2.2) como representar el contenido de las imágenes y luego como utilizar estas representaciones para evaluar la similitud

entre imágenes (Sección 2.3). Finalmente se muestra la evaluación de los métodos diseñados en este proyecto para estas dos tareas (Sección 2.4).

2.2. Representación de Imágenes

En cualquier tarea de procesamiento de imágenes, el primer paso es decidir como se va a representar la información contenida en la imagen. En la mayoría de los casos, no se puede trabajar con la información en crudo de cada píxel de la imagen, sino que se extraen una serie de descriptores más complejos. Los descriptores que se utilizan se pueden agrupar en dos grandes grupos, globales y locales, como se detalla a continuación y se resume en la Figura 2.2.

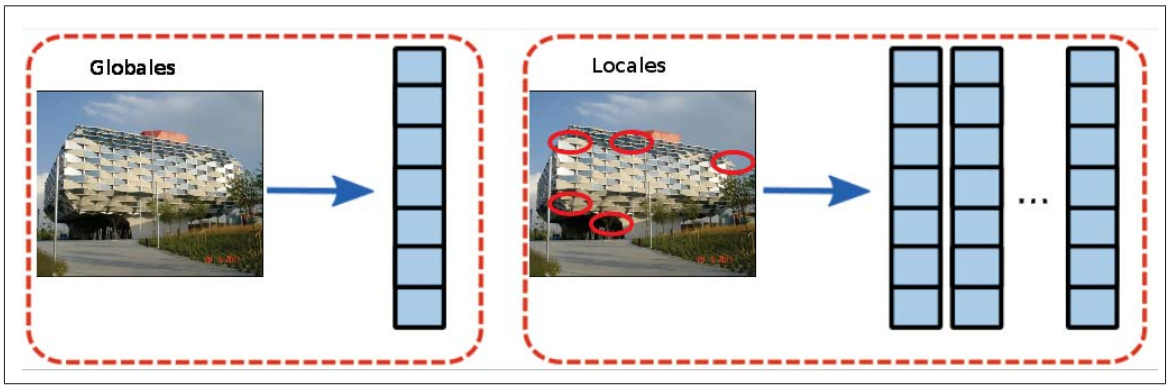


Figura 2.2: **Descriptores Globales - Locales.** A la izquierda vemos un ejemplo de una imagen representada por un único descriptor global (por ejemplo, un vector con los colores dominantes en la imagen). A la derecha, la misma imagen puede ser representada por un conjunto de descriptores locales, por ejemplo los mismos vectores de color pero correspondientes a diferentes regiones de la imagen.

Descriptores Globales

Cada imagen es representada con un sólo descriptor, que almacena información de toda la imagen. Este descriptor puede ser momentos de color, es decir, los valores de color de todos los píxeles (media, mediana...), histogramas de los niveles de gris en la imagen, histogramas de los valores del gradiente entre píxeles vecinos, etc. La principal ventaja de estos descriptores es que se calculan de forma rápida y normalmente requieren menos memoria para almacenarlos. Sin embargo, su capacidad descriptiva es menor que los descriptores locales, por lo que el rendimiento que nos pueden ofrecer a la hora de detectar detalles en la escena es menor. Además también suelen ser menos robustos a cambios en la escena.

Descriptores Locales

Cada imagen es representada por un conjunto de descriptores asociados a diferentes regiones de la imagen. Estos descriptores se calculan en partes o regiones de la imagen determinadas automáticamente mediante distintos procesos y heurísticas. Hay muchos tipos de características locales, basados en líneas [5], esquinas [9], puntos [12], etc. La características que se han utilizado

en este proyecto son SURF [3] (Speeded Up Robust Features). SURF es un algoritmo de detección y descripción de puntos de interés en imágenes, cuya principal ventaja es la velocidad e invarianza a cambios en las imágenes (luz, escala o perspectiva). Estos puntos SURF ya se han utilizado con anterioridad para reconocimiento de objetos [2].

En este proyecto nos hemos centrado en las características locales para resolver el problema de la búsqueda en colecciones de imágenes, ya que en la literatura relacionada presentan en general resultados mucho mejores y robustos para reconocimiento de objetos dentro de conjuntos de imágenes [11] [15] con gran variabilidad de parámetros.

2.3. Algoritmos de Evaluación de Similitud entre Imágenes

En general, todos los métodos para evaluar la similitud entre imágenes nuevas y un conjunto de referencia se dividen en dos fases, resumidos en la Figura 2.3:

1. **Fase de entrenamiento:** En esta fase se extraen las características de las imágenes de referencia y se almacenan y organizan en estructuras de datos que permitan rápido acceso y comparación.

2. **Fase de consulta:** En esta fase, dada una nueva imagen (en nuestro caso proveniente del móvil del usuario) se extraerán las características de la misma y se evaluará su similitud con la estructura almacenada en la fase anterior, para decidir que imagen de referencia parece tener un contenido más similar.

En este proyecto se han estudiado dos métodos para almacenar la información de los descriptores de las imágenes de referencia, detallados a continuación. Ambos métodos hacen uso, exacto o aproximado, de algoritmos de búsqueda del "vecino más cercano", (NN, de la definición en inglés nearest neighbour), es decir, buscar el descriptor más parecido a uno dado, de acuerdo a una medida o distancia de comparación.

2.3.1. Representación Completa

En este caso se almacenan y evalúan todos los descriptores de todas las imágenes. Para realizar esta evaluación, se compara cada descriptor de la imagen del usuario con cada uno de los descriptores de las imágenes de referencia, obteniendo con cada una de estas un número determinado de correspondencias. Aquella imagen de las de referencia con más correspondencias será considerada la más similar (ver Figura 2.4).

Para la búsqueda de correspondencias de características, se pueden utilizar diversos algoritmos variantes de la búsqueda del "vecino más cercano". Algunos de ellos se detallan a continuación:

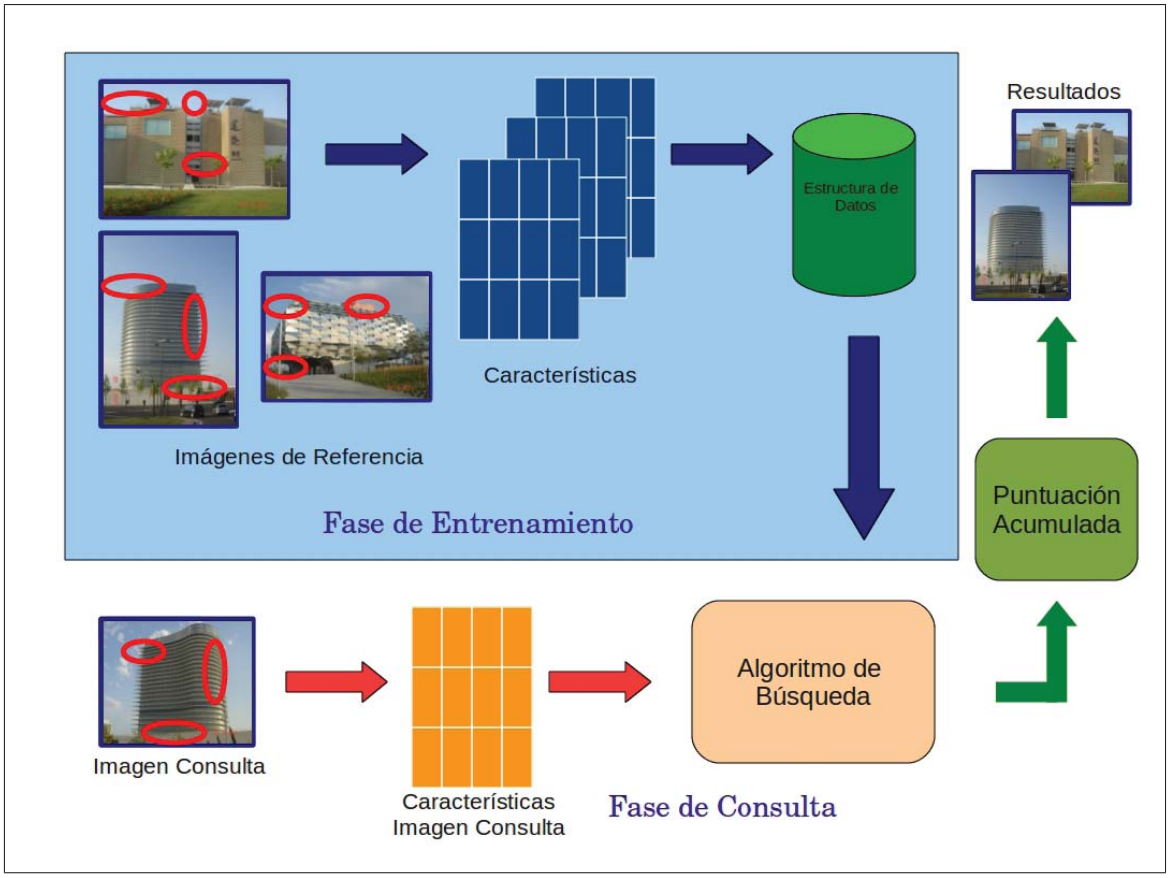


Figura 2.3: Algoritmo de búsqueda de imágenes similares en una base de datos. Se divide en dos fases, la fase de entrenamiento y la fase de consulta.

- Búsqueda del vecino más cercano (NN):
  consiste en una búsqueda exhaustiva. Dada una imagen de test y una imagen de referencia con la que queremos comparar, este algoritmo compara cada característica de la imagen de prueba con cada una de la imagen de referencia para buscar posibles correspondencias. El coste de este algoritmo de correspondencia entre imágenes es cuadrático dependiendo del número de características a emparejar (n): O(n^2).

- Búsqueda aproximada del vecino más cercano (ANN):
  realiza la misma tarea que la búsqueda anterior pero con un coste computacional menor, gracias a mantener la información de referencia más ordenada. Para ello, construye una estructura de árbol, en particular *k-d trees* [4], para ordenar los datos. Esto permite obtener unos tiempos de ejecución mucho más bajos, con resultados similares a los que se obtienen con búsqueda exhaustiva. Un inconveniente de esta búsqueda es que no siempre garantiza que se encuentre la solución óptima. El coste de la búsqueda es O($log(n)$). Para más detalle, en el anexo C se pueden ver experimentos que muestran las mejoras del uso de *k-d trees*.

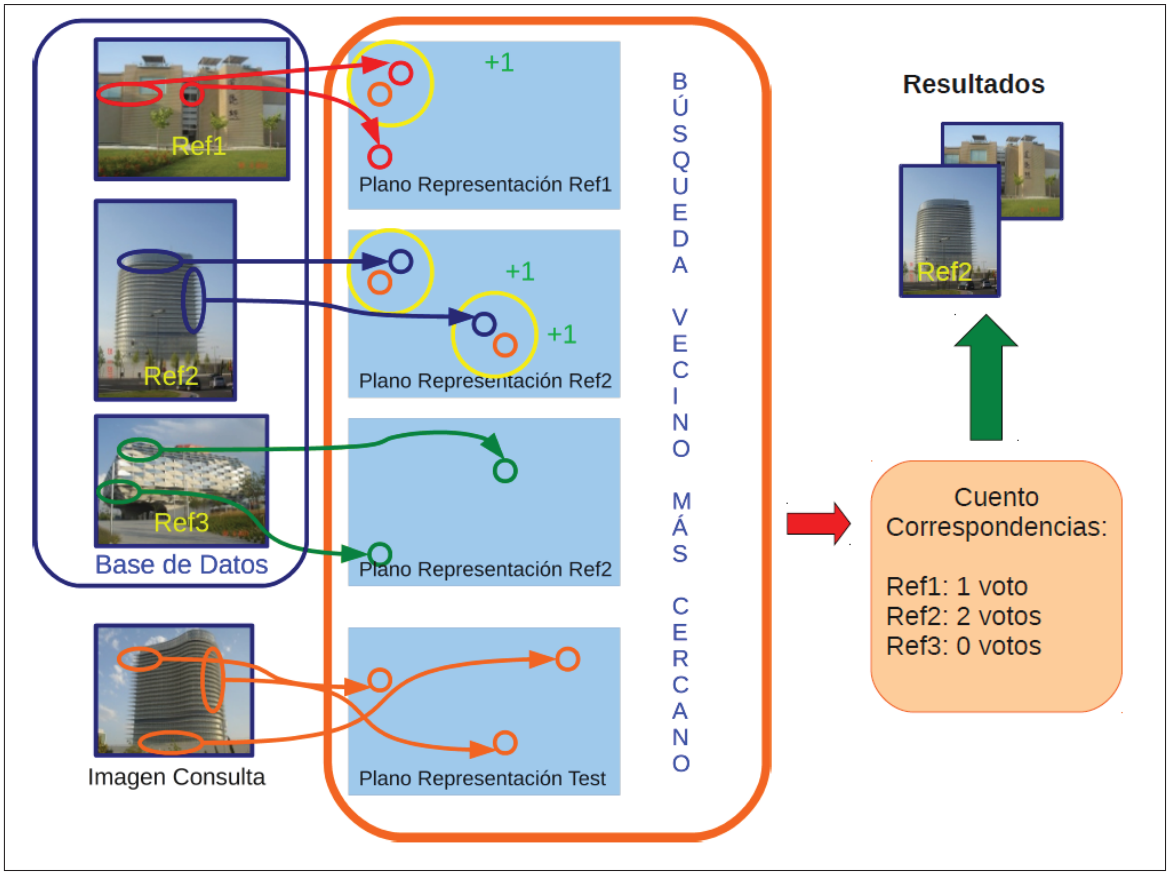


Figura 2.4: **Evaluación de similitud entre imágenes basada en representación completa.** Este algoritmo selecciona aquella imagen de referencia con más correspondencias encontradas. La imagen Ref1 tiene una correspondencia, Ref2 dos y Ref3 ninguna. Por lo que la elegida será Ref2.

- Búsqueda aproximada del vecino más cercano con métodos robustos:

  dado que la búsqueda mediante el algoritmo ANN o NN puede generar correspondencias incorrectas, se puede añadir a estos métodos un paso de estimación robusta, como es el método RANSAC [8]. Es común estimar una restricción geométrica entre varias imágenes de una misma escena (homografía o matriz fundamental) mediante RANSAC a la vez que se evalúa que correspondencias entre imágenes son consistentes con ella [10]. Esto se consigue rechazar muchas correspondencias incorrectas que no son consistentes con el modelo geométrico que se estima de la escena. Pese a que la robustez de las correspondencias crece notablemente, este método no resulta del todo práctico para el fin que se persigue en este trabajo, ya que el tiempo de ejecución de esta estimación robusta es muy elevado, y lo que se intenta es encontrar un método eficiente y de rápida ejecución.

2.3.2. Bolsa de Palabras o Clusterización

En este caso se intenta almacenar una representación más comprimida de los descriptores de las imágenes de referencia. Se agrupan las características de todas las imágenes de referencia en n conjuntos (o *clústers*) de descriptores parecidos, o lo que se suele denominar, un vocabulario de n palabras visuales [14] [16]. Este vocabulario almacena por un lado los *centroides* de cada palabra y por otro una tabla que indica en que imágenes de referencia y con qué frecuencia aparece cada palabra (esta estructura se suele denominar "inverted file index"[6], y tiene su origen en trabajos de análisis de documentos de texto, de ahí la nomenclatura). Utilizando diversos métodos de comparación, basados en analizar que palabras aparecen en una imagen de consulta y las de referencia, se puede calcular una similitud entre ellas (ver Figura 2.5).

Estos métodos de evaluación de similitud mediante bolsa de palabras se dividen en dos partes. La primera consiste en obtener el vocabulario a partir de las imágenes de referencia, y los histogramas de votos a cada palabra que tiene cada imagen de referencia. La segunda consiste en asociar cada descriptor de la imagen de consulta a una palabra obteniendo su histograma de votos a palabras del vocabulario previamente calculado. A continuación se detalla más detenidamente estos dos procesos.

Creación del Vocabulario

Para la creación del vocabulario se ha utilizado el algoritmo de *k-means* [1]. Este algoritmo busca los k grupos más representativos entre los datos a analizar. Cada uno de los k grupos o palabras se representa con el valor medio de los elementos que pertenecen a él, w_k.

Más formalmente, dado un conjunto de observaciones $(x_1, x_2, \ldots, x_n)$ donde cada observación es un vector de dimensión d, *k-means* trata de agrupar esos n datos en k grupos que formarán el vocabulario V, con $k <= n$, $V = \{w_1, w_2, \ldots, w_k\}$ donde cada w_k representa una palabra del vocabulario.

El funcionamiento básico del algoritmo es el siguiente:

1. Se seleccionan k elementos del conjunto de observaciones, estos representarán los centroides iniciales de las palabras del vocabulario.

2. Seleccionados los centros de esas k palabras, se asocia cada observación a la palabra más cercana.

3. El nuevo centroide de cada una de las k palabras será la media de todos los elementos que han sido asignados a esa palabra.

4. Se repiten los pasos 2 y 3 hasta que el algoritmo converge. La convergencia se produce cuando no hay cambio en los pasos 2 y 3, obteniendo así un conjunto estable de palabras.

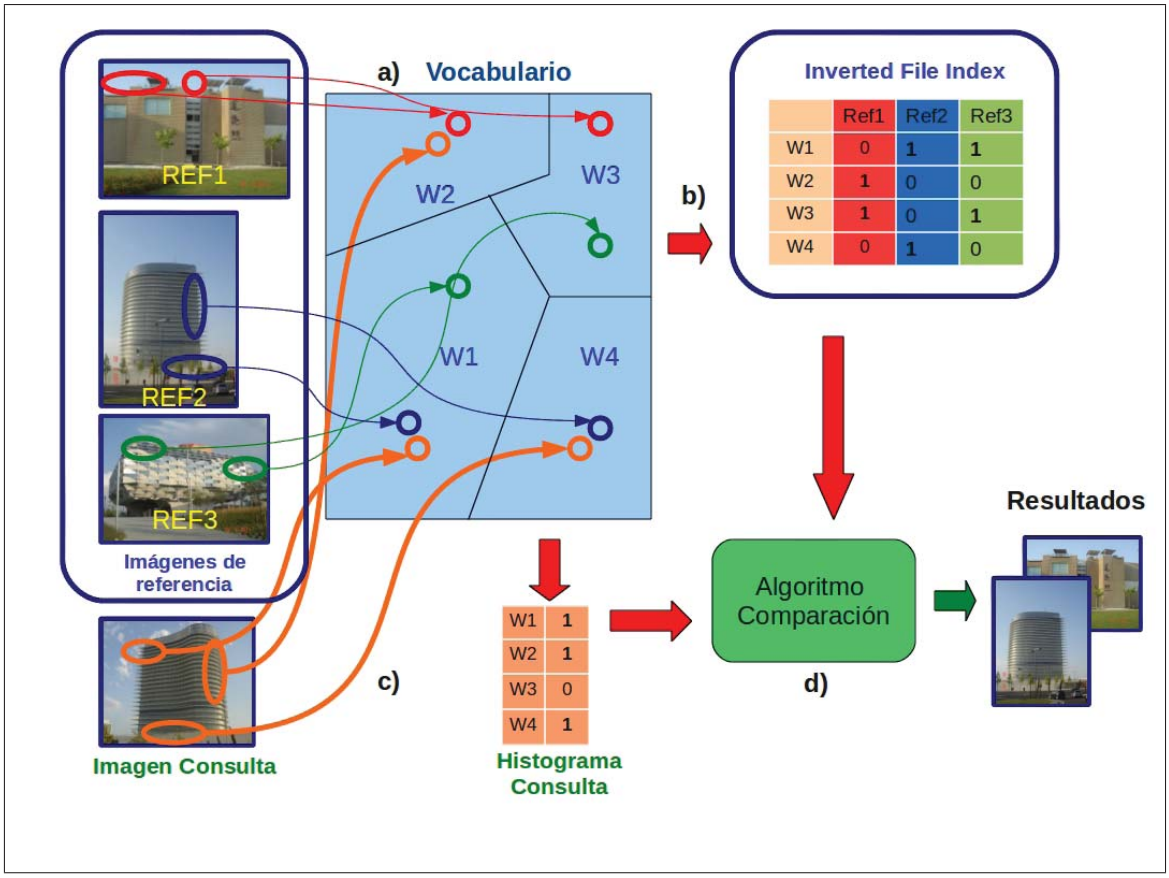


Figura 2.5: **Evaluación de similitud entre imágenes basada en representación de bolsa de palabras.** Primero, a partir de los descriptores de las imágenes de referencia obtenemos las k palabras que formarán el vocabulario a), en este caso {W1, ..., W4}. A partir de este vocabulario creamos el *inverted file index* b), almacenando la información de forma compacta. Con esta información, se asocia cada descriptor de la imagen de consulta a la palabra más representativa, obteniendo su histograma de votos a palabras c). Por lo que comparando dicho histograma con cada uno de los almacenados en el *inverted file index* podemos determinar la imagen más similar d).

La forma en que se realiza la primera asignación de los k clúster iniciales puede ser aleatoria, o basada en el uso de diversos algoritmos como puede ser el creado por Arthur and Vassilvitskii [7], utilizado en el desarrollo de este proyecto.

El algoritmo *k-means* devuelve a qué palabra se asocia cada unos de los descriptores de las imágenes de referencia, para disponer de esa información de forma más compacta se comprime, obteniendo la matriz del *inverted file index*. Esta matriz almacena un histograma de votos a cada palabra del vocabulario para cada una de las imágenes de referencia (Figura 2.6).

Evaluación de similitud mediante Bolsa de palabras

Una vez construido el vocabulario y la representación de las imágenes de referencia, se utilizan los siguientes pasos para evaluar una imagen nueva.

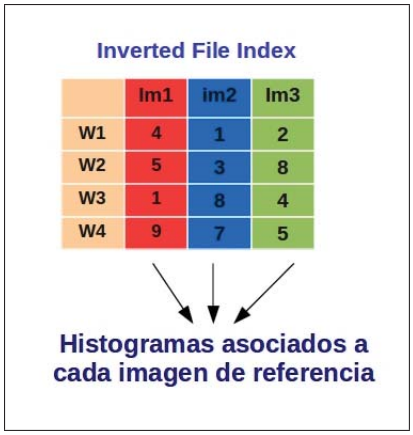


Figura 2.6: ***Inverted File Index.*** Representa para cada imagen de referencia {Im1, Im2, Im3} el número de votos a cada palabra que ha resultado en la asignación de sus descriptores a las palabras del vocabulario {W1, W2, W3, W4}, es decir, su histograma.

1. Extracción de características SURF de imagen test.

2. Búsqueda de la palabra que se asigna a cada característica. Obteniendo así un vector de longitud igual al número de características, donde cada posición indica a que palabra ha sido asignada dicha característica.

3. Histograma de palabras. A partir del vector obtenido en el paso anterior se crea un vector de longitud k (número de palabras del vocabulario) donde se almacena la frecuencia con que ha ocurrido cada palabra en esta imagen. Este vector es el histograma de palabras que aparecen en esta imagen, similar al que tenemos para cada imagen de referencia.

4. Haciendo uso de este histograma se calcula una medida de similitud. Hemos estudiado dos posibilidades, detalladas en el Anexo B.

   - Sim1: Comparación basada en los histogramas de palabras de cada imagen, es decir, se compara el histograma de la imagen consulta con cada uno de los histogramas de las imágenes de referencia, y aquellos cuya distancia L1 (norma 1) es menor serán seleccionados como más similares.

   $$Dist_1(I_i, I_{consulta}) = |H_i - H_{consulta}| \tag{2.1}$$

   Donde H_i representa el histograma de palabras asociado a la imagen de referencia i (I_i), y $H_{consulta}$ el histograma de palabras asociado a la imagen consulta ($I_{consulta}$).

   - Sim2: Medida basada en la frecuencia con la que aparece cada palabra w en cada imagen de referencia I_i. Es decir, cada palabra que aparece en la imagen de consulta

($I_{consulta}$) vota a aquellas de imágenes de referencia donde aparecía con una frecuencia más similar. Las imágenes seleccionadas cómo más similares serán aquellas con más votos.

$$Votos(I_i, w_j) = \begin{cases} H_c(j) \text{ si } [H_c(j) - H_i(j)] == min([H_c(j) - H_1(j)]), ..., [H_c(j) - H_n(j)] \\ 0 \text{ en cualquier otro caso} \end{cases} \tag{2.2}$$

$$Votos_i = \sum_{j=1}^{n} Votos(I_i, w_j) \tag{2.3}$$

Donde $H_c(j)$ es el número de votos a la palabra j existente en la imagen consulta. $H_i(j)$ es el número de votos a la palabra j existente en la imagen de referencia i.

2.4. Experimentos

A continuación se detallan los experimentos más representativos realizados para evaluar la similitud entre imágenes utilizando los métodos descritos anteriormente. En esta memoria principal detallamos solamente los experimentos finales más significativos, dejando en los Anexos A y C los detalles de los demás experimentos realizados, que justifican las diferentes decisiones a lo largo del proceso y diseño.

Los experimentos que se detallan a continuación, así como los detallados en los Anexos A y C, se han realizado en primer lugar sobre un computador (procesador Intel Core 2 Duo 2.1 GHz, memoria RAM 4 GB). Esto es debido a la necesidad de realizar pruebas a gran escala para evaluar aquellos métodos con mejores resultados y una rápida ejecución para su posterior migración al dispositivo móvil.

En todos los experimentos se utiliza una base de datos que contiene un número determinado de imágenes de referencia, de las cuales se extrae la información y se almacena de diferentes formas según el método ejecutado. Como imágenes de consulta se utiliza un conjunto variado de imágenes distintas a las almacenadas en la base de datos, entre las que se incluyen imágenes de edificios obtenidas de Internet.

El objetivo en este caso, o criterio de similitud que queremos cumplir, es que las imágenes contengan el mismo edificio. Es decir, que la búsqueda de las imágenes más similares nos indique que monumento aparece en la imagen de consulta.

Es importante estudiar qué métodos dan mejores resultados en la identificación de los monumentos/edificios, pero también se debe prestar atención a aquellos que son más eficientes (tanto en ocupación de memoria y tiempo de ejecución), ya que los métodos desarrollados finalmente se deben implementar en un teléfono móvil.

La medida utilizada para evaluar la calidad del reconocimiento de cada método es el porcentaje de imágenes cuyo contenido ha sido reconocido correctamente. Además como la aplicación final tiene interacción con el usuario, queremos que la aplicación resulte realista y robusta. Por eso, en casos en que el reconocimiento no este claro y haya varios candidatos con similitud parecida, evaluaremos la opción de dar como solución varias opciones ordenadas por similitud para que el usuario decida.

Se ha realizado el estudio tanto de los métodos de representación completa (subsección 2.4.1) como mediante el uso de bolsa de palabras (subsección 2.4.2), para evaluar la técnica más adecuada para implementar en el dispositivo móvil.

2.4.1. Representación completa

En primer lugar se realizaron una serie de experimentos para evaluar los métodos de representación completa.

Detalles de los experimentos para su evaluación y selección de que medida de similitud es más adecuada están en el Anexo A. Como resumen vemos en la Figura 2.7 que los resultados obtenidos mediante la utilización de una búsqueda exhaustiva (NN) frente a la búsqueda aproximada (ANN) (subsección 2.3.1) son similares, pero con la diferencia de que el tiempo de ejecución mediante la utilización de NN frente a ANN se incrementa considerablemente. En este caso el tiempo de simulación total pasa de 910 a 1129 segundos.

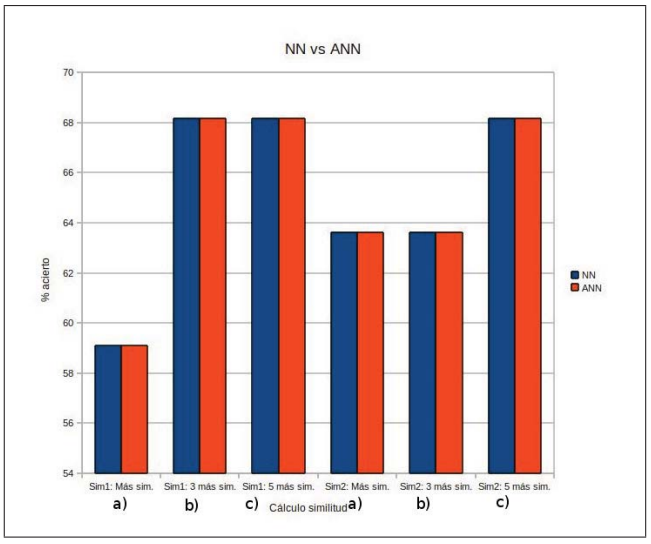


Figura 2.7: Acierto en el reconocimiento de edificios utilizando búsqueda NN vs ANN. Los tres primeros grupos de barras representan los resultados o porcentaje de acierto para la medida de similitud Sim1 (ec. A.6) teniendo en cuenta si se elige la más votada a) o entre las 3 ó 5 más votadas hay una correcta (b) y c) respectivamente). Idem para los otros tres grupos de barras pero con Sim2 (ec. A.7).

Aunque los resultados de reconocimiento son aceptables, estos métodos presentan el inconveniente, para la implementación en un dispositivo móvil, de un tiempo de ejecución todavía bastante elevado. Además un elevado consumo de memoria, porque se debería almacenar en una matriz todos los descriptores de las imágenes de referencia y realizar la comparación con cada una de ellas. Suponiendo que de cada imagen se obtienen n caraterísticas SURF (unas 1000 de media), y que cada descriptor está compuesto por un vector de longitud d (64) de datos "float", la representación ocupará:

$$d * n * 4B = 375KB \; por \; imagen$$

Suponiendo una base de datos de 100 imágenes, el tamaño que ocupa toda esa información aproximadamente es $375KB * 100 = 37500KB = 36{,}62MB$.

Viendo estos datos de ocupación de memoria, se ve claramente que no sería nada recomendable su implantación completa en un dispositivo móvil. Conforme aumente la base de datos, la memoria necesaria sería muy elevada para un un móvil y sería necesaria una conexión remota para acceder a estos datos de referencia.

2.4.2. Bolsa de Palabras o Clusterización

En segundo lugar, se han estudiado técnicas de comparación de imágenes basadas en en agrupaciones de descriptores, técnica más frecuentemente referida como clusterización o bolsa de palabras, que utilizaremos en el resto del documento. Estos algoritmos son muy utilizados por sus ventajas de eficiencia y requerimientos de memoria más bajos. Se usan en muchos campos para procesar datos, entre los cuales se encuentra el reconocimiento de imágenes.

Al igual que en la subsección anterior, aquí sólo se incluye el experimento más completo realizado con estos métodos. Experimentos anteriores están detallados en el Anexo C, donde se muestran pruebas iniciales, evaluaciones de las distintas medidas de similitud, y análisis de la utilización del GPS integrado en los móviles.

El experimento principal descrito a continuación confirmó los dos pasos siguiente como claves para obtener buenos resultados:

1. Por un lado, la incorporación del filtrado con GPS (en el Anexo G se describen más detalles acerca del GPS), que selecciona las imágenes candidatas que además de alta similitud, tengan una distancia métrica dentro de un rango. Este filtrado se podría incluir con cualquier método de similitud, pero se tomó la decisión de sólo evaluarlo con los métodos que mejores resultados proporcionaban.

2. Por otro lado, a la hora de realizar la asignación de palabras a cada característica de la imagen consulta, es importante el uso de *k-d trees* para realizar una búsqueda de la palabra más cercana, que ahorra mucho tiempo respecto a hacer una búsqueda exhaustiva.

Configuración del experimento

- Modo búsqueda: Bolsa de Palabras
- Nº palabras vocabulario: 600
- Nº imágenes prueba: 141

En las Figuras 2.8 y 2.9 se observan los resultados de esta pruebas. La primera (2.8) muestra los resultados con la medida de similitud Sim1(ec. 2.1) mostrados en la Tabla 2.1, y Sim2(ec. 2.3) mostrados en la Tabla 2.2. Se utilizan estructuras *k-d tree* para la búsqueda de la palabra asignada a cada punto SURF extraído de la imagen consulta. La Figura 2.9, sin embargo muestra los resultados de la prueba haciendo una búsqueda exhaustiva a la hora de asignar una palabra a cada característica.

Asignación de palabras con *k-d tree*

	Top 1	Top 3	Top 5	
Sin GPS	65.24 %	76.59 %	80.85 %	Tiempo de cada test 0.2 s
GPS 1 Km	78.01 %	88.65 %	91.48 %	
GPS 0.5 Km	84.39 %	91.48 %	93.61 %	

Tabla 2.1: Reconocimiento correcto de edificios/monumentos utilizando *k-d tree* y Sim1 (ec.2.1): Porcentaje de imágenes de consulta donde se encuentra un resultado correcto entre los "n" más similares ("top-n"). En cada una de las filas aparece los resultados teniendo en cuenta si se selecciona entre el top-1, top-3 o top-5 sin hacer uso del GPS, GPS con un radio de 1 Km o 0.5 Km. "Top-n" nos indica los resultados que obtenemos si se observan los "n" resultados más similares, es decir, si en esos "n" resultados hay alguna imagen asociada correctamente.

	Top 1	Top 3	Top 5	
Sin GPS	39.00 %	60.28 %	66.66 %	Tiempo de cada test 0.2 s
GPS 1 Km	63.82 %	78.01 %	83.68 %	
GPS 0.5 Km	74.46 %	86.52 %	92.19 %	

Tabla 2.2: Reconocimiento correcto de edificios/monumentos utilizando k-d tree y Sim2 (ec.2.3). Tabla con el mismo formato que Tabla 2.1

De estas gráficas se puede observar que con la medida de similitud 1 se obtienen mejores resultados, además del notable incremento en el porcentaje de acierto al incluir el GPS. Se pasa de un 65.24 % sin usar GPS a un 78.01 % usando un radio de 1 Km ó 84.39 % con radio 0.5 Km.

En el caso de la medida de similitud 2 se observa que los resultados son inferiores a los mostrados en otras pruebas. Esto es debido a que al asignar las palabras utilizando *k-d trees* la precisión disminuye, por lo que el método se ve perjudicado.

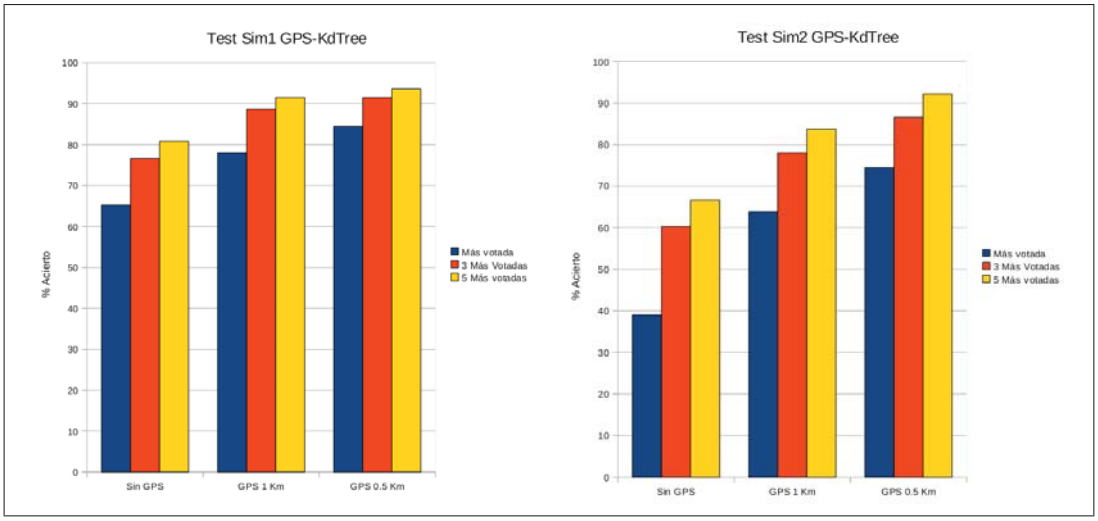


Figura 2.8: Reconocimiento correcto de edificios/monumentos utilizando o no filtro GPS y k-d tree

	Top 1	Top 3	Top 5
Sin GPS	71.63 %	78.72 %	82.97 %
GPS 1 Km	80.14 %	88.65 %	92.19 %
GPS 0.5 Km	87.94 %	91.48 %	93.61 %

Tiempo de cada test 15.42 s

Tabla 2.3: Reconocimiento correcto de edificios/monumentos utilizando búsqueda exhaustiva y Sim1 (ec.2.1). Tabla con el mismo formato que Tabla 2.1

Asignación de palabras con búsqueda exhaustiva

En este caso se observa que los resultados son ligeramente mejores que los obtenidos haciendo uso de *k-d trees*. Por ejemplo, sin uso de GPS, con búsqueda exhaustiva obtenemos un 71.63 % frente a un 65.24 % con *k-d tree*, para la medida Sim1 seleccionando sólo la imagen más similar.

Pero el principal inconveniente de la búsqueda exhaustiva es el tiempo de ejecución que lleva asociado. Como se observa en la gráfica pasamos de un tiempo de ejecución del test con *k-d trees* de 401 s a 3700 s algo que hay que tener en cuenta ya que en dispositivo móvil aún se verá más acentuado. El tiempo medio de asignación de palabra con búsqueda exhaustiva es 15.42 s frente a 0.2 s mediante la utilización de *k-d trees*, como se ve en la Figura 2.10.

	Top 1	Top 3	Top 5
Sin GPS	44.68 %	68.08 %	76.59 %
GPS 1 Km	71.63 %	84.39 %	90.78 %
GPS 0.5 Km	78.01 %	90.78 %	95.03 %

Tiempo de cada test 15.42 s

Tabla 2.4: Reconocimiento correcto de edificios/monumentos utilizando búsqueda exhaustiva y Sim2 (ec.2.3). Tabla con el mismo formato que Tabla 2.1

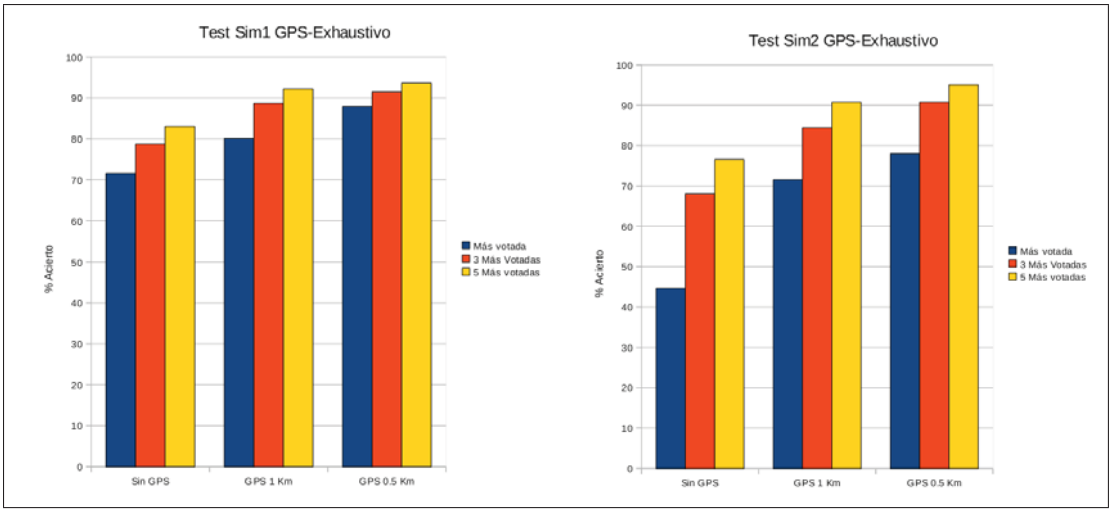


Figura 2.9: Reconocimiento correcto de edificios/monumentos utilizando o no filtro GPS y búsqueda exhaustiva

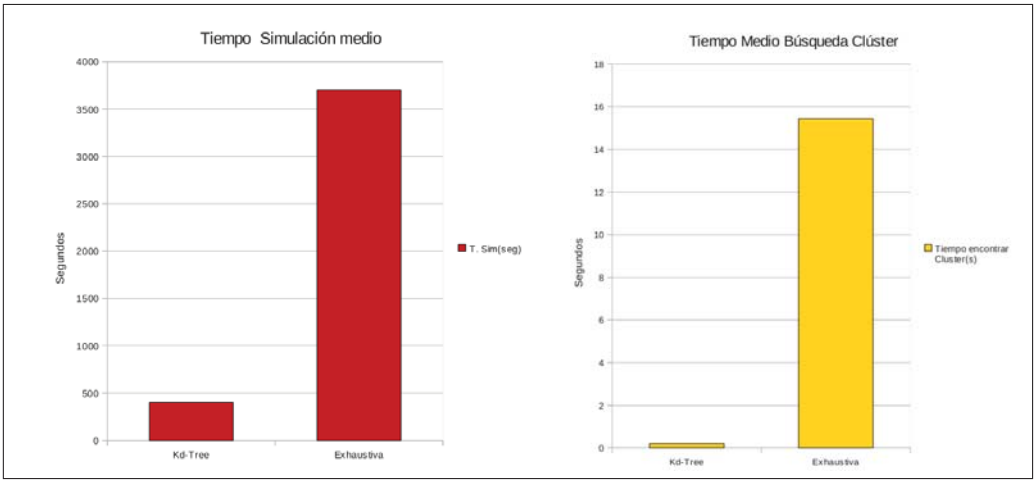


Figura 2.10: Tiempo de ejecución del paso de asignación de palabras a cada característica utilizando *k-d tree* ó búsqueda exhaustiva. La gráfica izquierda representa el tiempo total de simulación medio de todo el experimento con el conjunto de imágenes de consulta. La gráfica derecha muestra el tiempo medio de asignación de palabra para una característica.

Conclusión

A la vista de los resultados obtenidos, la configuración que se implantará en el dispositivo móvil será la siguiente:

- Modo búsqueda: Bolsa de Palabras
- Nº palabras del vocabulario: 600
- Método de asignación de palabra: *k-d tree*

- Radio GPS: 1 Km
- Medida similitud: Sim1 (ec. 2.1)

Se ha decidido utilizar para la asignación de palabras la búsqueda mediante ANN que hace uso de las estructuras *kd-trees*, ya que aunque el porcentaje de acierto disminuye ligeramente con respecto a la búsqueda exhaustiva, el tiempo de ejecución se ve reducido drásticamente. Como medida de similitud se ha seleccionado Sim1(ec. 2.1), ya que los resultados que obtiene son considerablemente mejores que Sim2(ec. 2.3). En cuanto al radio utilizado para el filtrado con GPS se ha seleccionado 1 Km, para asegurarnos que no descartamos soluciones correctas por errores en la localización del GPS.

Capítulo 3

Realidad Aumentada basada en Geometría Multi-Vista

3.1. Introducción

La realidad aumentada es un término que se usa para definir la visualización de una escena con información aumentada, que realmente no esta presente en la escena. Los elementos de la escena real se combinan con elementos virtuales para la creación de una *realidad mixta*. Existen una serie de dispositivos que se pueden utilizar para añadir información virtual a la escena real. En nuestro caso, el dispositivo encargado de añadir esa información será el *smartphone*.

Este capítulo describe un pequeño ejemplo de como mostrar los resultados de nuestro proyecto en una escena de realidad aumentada. Para la visualización de los elementos virtuales, se pueden usar diferentes técnicas. Hay técnicas que usan directamente medidas del hardware del dispositivo, por ejemplo el GPS, acelerómetro, etc. para conocer en todo momento la posición y orientación del dispositivo, permitiendo el posicionamiento de los elementos virtuales correctamente en la escena real. Otras técnicas se basan en técnicas de visión por computador, procesando imágenes tomadas con la cámara del dispositivo para estimar la geometría de la escena, permitiendo mostrar elementos virtuales integrados correctamente en la escena real. Para ello se utilizan técnicas muy extendidas de la visión por computador y geometría multi-vista, como son la estimación de la matriz fundamental y la homografía entre dos imágenes [10]. La estimación de este tipo de restricciones e información geométrica de la escena permite que los objetos virtuales que se añaden parezcan consistentes con la estructura de la escena.

En nuestro proyecto nos centraremos en una técnica basada en el uso de algoritmos basados en restricciones geométricas para superponer la información virtual. En particular, en nuestro proyecto de reconocimiento de monumentos, la idea es mostrar información sobre el monumento identificado proyectada sobre las paredes principales del edificio. Estas técnicas podrían utilizarse simplemente para mostrar texto o publicidad, pero por ejemplo podría ser interesante mostrar el interior del edificio, manteniendo la perspectiva desde donde el usuario está tomando la foto. Para

ello, anotamos cómo debería visualizarse en las imágenes de referencia y el algoritmo realizará las transformaciones correspondientes para cualquier punto de vista que tenga el usuario. En la Figura 3.1 se puede ver un ejemplo.

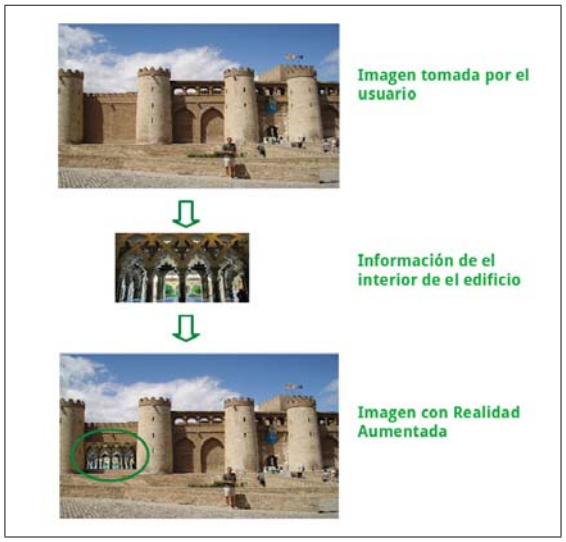


Figura 3.1: Ejemplo de visualización aumentada de los resultados de identificación de monumentos.

3.2. Restricciones Geométricas Multi-Vista

Como se ha comentado, para mostrar la información virtual de la escena, vamos a utilizar restricciones geométricas que ayuden a estimar la posición relativa de la imagen actual respecto de las imágenes de nuestra base de datos, para poder mostrar la información en una posición adecuada en la visualización que se da al usuario. Ya que el objetivo es proyectar información en las paredes de los edificios, vamos a utilizar la homografia, construcción que relaciona la proyección de un plano en la escena 3D en distintas imágenes.

La homografía es una transformación proyectiva que establece una correspondencia entre elementos de dos imágenes. Una explicación más detallada se puede ver en la Figura 3.2.

3.3. Implementación

Para implementar esta visualización aumentada de los resultados de nuestro proyecto hay dos puntos importantes a desarrollar:

- Anotar las imágenes de referencia con la información aumentada que se quiere incorporar, y detallar en esa imagen de referencia cual es el plano/pared sobre el cual se proyectará esta información.
- Implementar un método para estimar la proyección necesaria desde la imagen de referencia a una imagen que ha tomado el usuario, para visualizar la información aumentada correc-

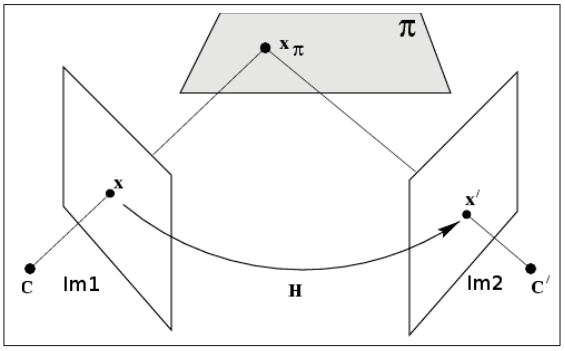


Figura 3.2: **Homografía.** Dos imágenes (Im_1,Im_2) de un mismo plano π en la escena 3D, están relacionadas por una transformación proyectiva H (homografía), de manera que dada la proyección de un punto en la imagen Im_1 (x), la H predice dónde se verá ese mismo punto en la segunda imagen Im_2 (x').

tamente integrada en la escena. El algoritmo desarrollado para este punto está detallado a continuación, además en la Figura 3.3 se muestra un esquema visual del proceso.

1. Extracción de puntos SURF en las imágenes de referencia y consulta.
2. Selección de puntos SURF según la región anotada como zona para visualización de información adicional.
3. Búsqueda de correspondencias en la imagen consulta, de las características que se encuentran dentro de la región marcada en la imagen de referencia.
4. A partir de las correspondencias encontradas, estimar de manera robusta, utilizando RANSAC [8] la homografía, H, entre ambas imágenes. La H relaciona los puntos de ambas imágenes que pertenecen al mismo plano en la escena real en 3D.
5. Utilizando la transformación proyectiva H, podemos proyectar todos los puntos de la información "aumentada" que tenemos visualizada sobre la imagen de referencia a la imagen de consulta, de manera que en la imagen del usuario se visualizará de forma correcta.
6. Superponemos la imagen del interior que hemos proyectado en la posición calculada de la imagen tomada por el usuario para mostrarle el resultado final.

3.4. Experimentos

Para llevar a cabo la implementación del proceso detallado en este capítulo se realizaron varias pruebas. Lo primero fue comprobar que las correspondencias de características entre imágenes se realizaban correctamente, como se puede ver en la Figura 3.4. Pero como se observa en la imagen izquierda, aparecen representadas todos las correspondencias y nosotros sólo necesitábamos los de una región concreta donde luego proyectar la imagen interior. Por lo que como

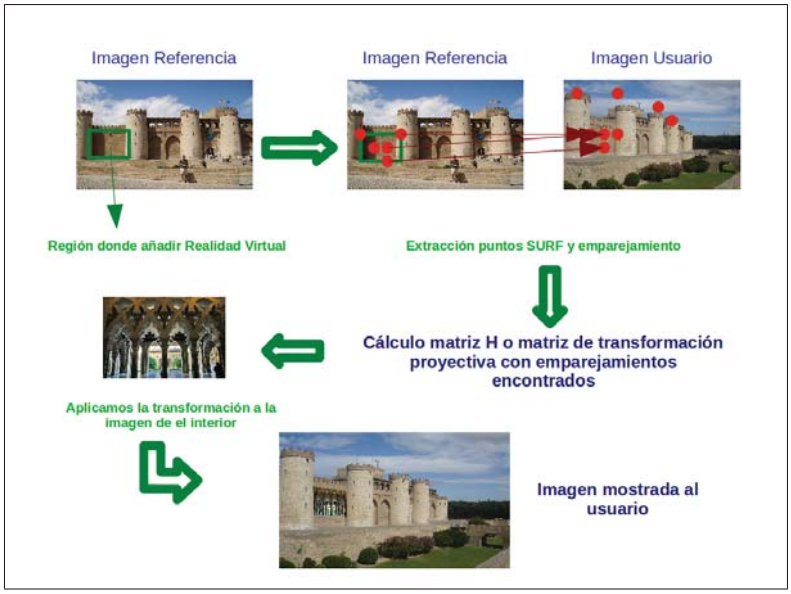


Figura 3.3: **Esquema algoritmo realidad aumentada.**

se ve en la imagen derecha de la Figura 3.4, limitamos la región dónde se buscarán correspondencias. En el Anexo E se pueden observar más ejemplos.

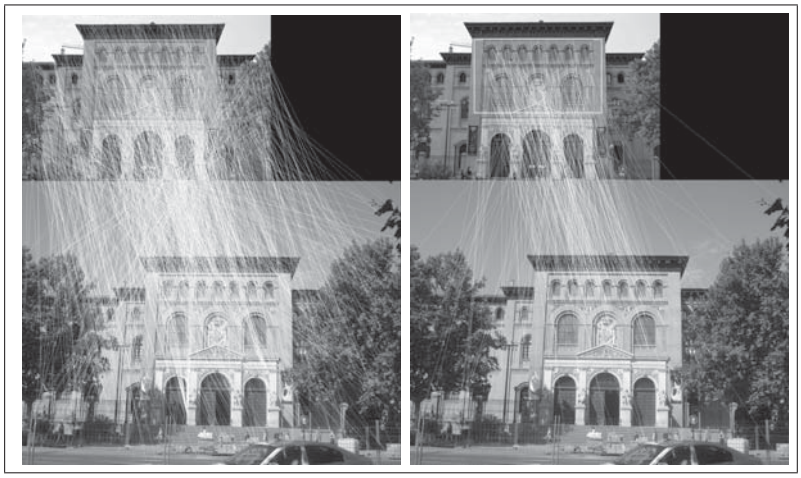


Figura 3.4: Correspondencias de puntos SURF entre imagen de referencia y la de consulta del usuario. En la imagen izquierda aparecen representados todos las correspondencias, mientras que en la imagen derecha sólo aparecen representados las correspondencias dentro de la región que nos interesa para añadir información aumentada.

Como se observa en ambas imágenes de la Figura 3.4, existen correspondencias erróneas. Como las correspondencias son la base para estimar la matriz H y tienen "ruido" (errores), necesitamos estimar dicha matriz a partir de las correspondencias pero utilizando un método robusto que sea capaz de procesar datos con ruido, como es el algoritmo RANSAC mencionado anteriormente. Este algoritmo estimará simultáneamente la homografía y el subconjunto de

correspondencias consistentes con la misma. Tras estimación de H y filtrado de las correspondencias consistentes con ella, mantenemos sólo el subconjunto de correspondencias entre imagen de referencia y consulta, que como se puede ver (Figura 3.5) son prácticamente 100 % correctas.

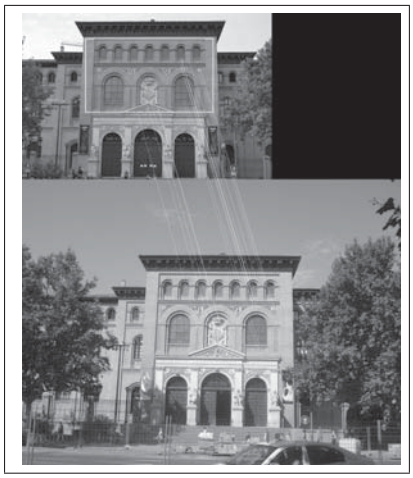


Figura 3.5: Filtrado de correspondencias tras el cálculo de H y exclusión de aquellas no consistentes mediante RANSAC.

En esta última Figura 3.6, en la imagen izquierda se observa como las cuatro coordenadas que limitan la región en la imagen de referencia se proyectan sobre la imagen de consulta utilizando la matriz H. Aplicando esta matriz a cada uno de los puntos de la imagen del interior del edificio, proyectamos esa imagen sobre la imagen del usuario, obteniendo como resultado final la imagen derecha.

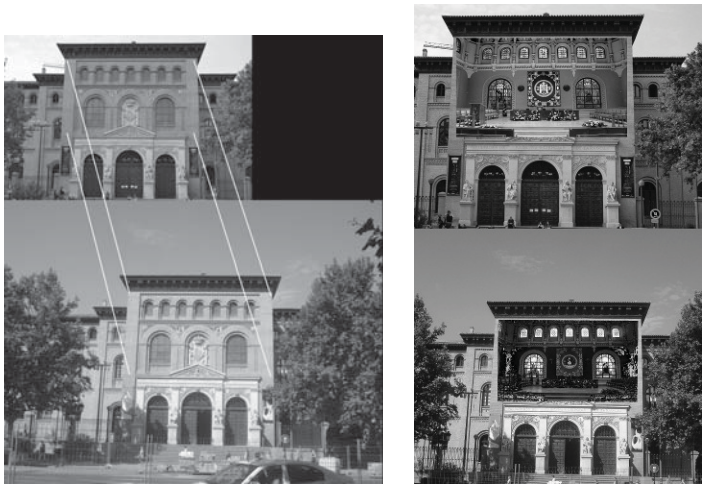
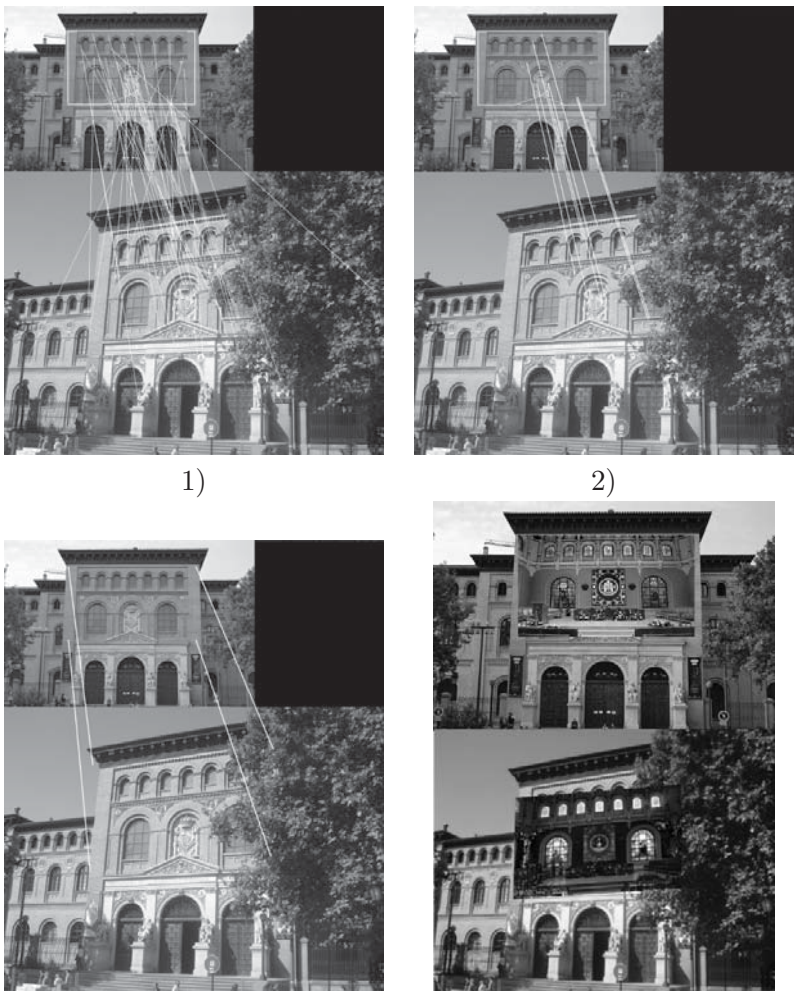


Figura 3.6: Proyección de la región e imagen interior sobre la imagen consulta. En la imagen izquierda aparecen proyectadas sobre la imagen consulta las cuatro coordenadas de la región marcada en la imagen de referencia para proyectar la imagen interior del edificio. En la derecha aparece la imagen interior proyectada sobre la imagen consulta.

Pero dado que este algoritmo se ha creado para que la adición de información extra a la

escena se haga automáticamente, no siempre funciona de forma perfecta. Hay ocasiones en las que la precisión con la que se calcula la matriz H no es del todo precisa y la proyección no es tan perfecta como nos gustaría (ver Figura 3.7). En el siguiente ejemplo se puede observar esto. Además en el Anexo E se encuentra algún otro ejemplo.



1) 2)

3) 4)

Figura 3.7: Ejemplo de proyección ruidosa. En la imagen 1) se observan todas las correspondencias dentro de la región donde se proyectará la imagen interior. En la imagen 2) tras calcular la matriz de homografía H y eliminar aquellas correspondencias no consistentes, sólo quedan las correctas. En la imagen 3) proyectamos dónde corresponden las coordenadas de la región en la imagen consulta. Finalmente, la imagen 4) muestra el resultado que se muestra al usuario.

Capítulo 4

Aplicación Desarrollada

En este capítulo se explica los diferentes módulos que componen aplicación diseñada, Sección 4.1, así como la navegación a través de las diferentes pantallas de la aplicación, Sección 4.2. Algunos otros detalles relacionados con la implementación son: el Anexo D con más detalles a cerca del sistema Android, el Anexo F donde se encuentran los diagramas de flujo de datos de la aplicación desarrollada y el Anexo H con la gestión del proyecto.

4.1. Módulos de la Aplicación

A continuación se va a explicar detalladamente los módulos básicos que forman nuestra aplicación. Estos módulos, básicamente se pueden dividir en tres grandes grupos (Figura 4.1), el módulo de interacción con el usuario, el cual está desarrollado en lenguaje Java; y el módulo de análisis y evaluación de similitud y el módulo de realidad aumentada, los cuales hacen uso de funciones de la librería OpenCV y están escritos en los lenguajes C y C++.

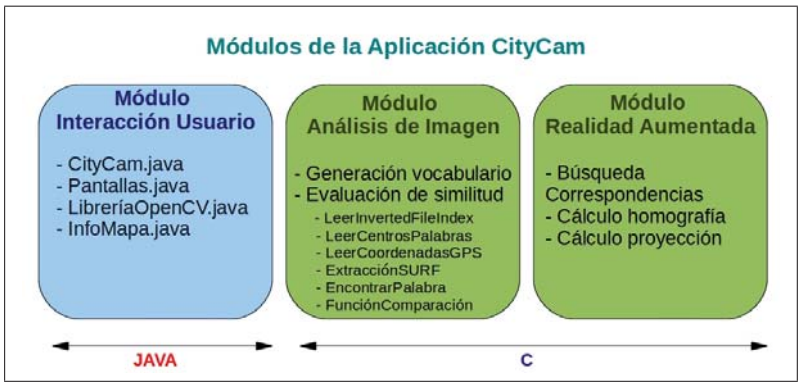


Figura 4.1: Módulos de la Aplicación. 1) Módulo de interacción con el usuario. Implementado en lenguaje Java. Contiene las clases Java para la gestión de los eventos de la aplicación al interactuar con el usuario, así como para invocar a la librería OpenCV o Google Maps. 2) Módulo para el análisis de la imagen, se divide en 2 submódulos, uno para la generación del vocabulario y otro para el evaluación de similitud. 3) Módulo para la implementación de realidad aumentada.

4.1.1. Módulo de Interacción con el Usuario

Este módulo es el encargado de gestionar la interacción con el usuario, es decir, es la implementación de la aplicación que interactúa con el usuario y cuyo uso está detallado en la Sección 4.2. Esta parte del proyecto se ha programado en lenguaje Java, ya que para la programación de Android se utiliza normalmente este lenguaje.

Las clases que componen este módulo son las siguientes:

- CityCam.java: clase principal de nuestro proyecto. Es la encargada de iniciar todos los componentes y en ella se carga la ventana principal de la aplicación y de gestionar los eventos en esta.
- PantallaMuestraImagenes.java: es una de las clases que se encargan de gestionar los eventos ocurridos en la pantalla donde se muestran los resultados tras el análisis de la imagen del usuario. Hay tantas clases como pantallas tiene nuestra aplicación.
- LibreriaOpenCV.java: en esta clase cargamos nuestro módulo de OpenCV donde se realiza el análisis de la imagen. También aparecen definidas las funciones de código nativo a las cuales podemos llamar desde código Java. En nuestro caso aparecen definidas las funciones *Análisis* y *RealidadAumentada*.
- InfoMapa.java: clase encargada de la carga de los mapas de Google Maps.

4.1.2. Módulo de Análisis de la Imagen

En este módulo aparecen definidas todas las funciones que realizan el análisis de la imagen, para evaluar la similitud con respecto a las imágenes de la base de datos. Todas ellas están escritas en C y C++. Estas funciones se pueden agrupar en dos grupos, las que se utilizan para generar el vocabulario y aquellas que se utilizan para evaluar la similitud de una imagen dada. En nuestro caso en la implantación en el dispositivo móvil sólo se han integrado las funciones de evaluación de similitud, ya que se supone que la generación de vocabulario es independiente y se ha generado previamente y el usuario nunca tendrá acceso a ello.

Generación de Vocabulario

Este submódulo es el encargado de generar la matriz de datos (*Inverted File Index)* que luego hace uso el módulo de evaluación de similitud. El algoritmo lo que realiza es, en primer lugar extraer las características SURF de cada imagen de la base de datos y almacenarlas en una matriz. Una vez se tienen todas características almacenadas, se aplica el algoritmo de clusterización, *KMeans*, el cual se encarga de generar a partir de todas esas características el vocabulario de n palabras. Como se indica en el Anexo B, este nos devuelve a la palabra a la

cual se asocia cada característica, por lo que a partir de esta información generamos el *Inverted File Index* que luego se almacena en el dispositivo móvil junto a la matriz de centros de cada una de las palabras del vocabulario, para que el algoritmo de evaluación de similitud haga uso de esta información.

Por tanto, de este módulo se implanta en el dispositivo móvil el fichero *Inverted File Index* y la matriz de centros de las palabras del vocabulario generado, ya que esta información es necesaria para la evaluación de similitud entre imágenes.

Evaluación de Similitud

En este submódulo, existen funciones para realizar la evaluación de similitud de una imagen tomada por el usuario con las de la imágenes de la base de datos. Las funciones más importantes que lo componen son:

- LeerInvertedFileIndex: a partir de un fichero de texto, esta función se encarga de cargar en la memoria del dispositivo la matriz conocida como *Inverted File Index*, la cual ha sido generada anteriormente.
- LeerCentrosPalabras: función similar a la anterior, se encarga de cargar en memoria la matriz de centros de cada una de las palabra que forman el vocabulario.
- LeerCoordenadasGPS: también se almacena en un fichero las coordenadas de los edificios de nuestra base de datos, por lo que esta función carga en memoria esta información.
- ExtracciónSURF: función encargada de obtener las características SURF de la imagen del usuario. Invoca a la función de la librería OpenCV, *cvExtractSURF*.
- EncontrarPalabra: esta función busca cual es la palabra más cercana a la cual se asocia cada una las características SURF. Requiere la matriz de centros para calcular la distancia a cada una de las palabras del vocabulario. Como resultado tendremos un vector que asocia a cada característica su palabra.
- FuncionComparación: a partir de la matriz *Inverted File Index* y los votos a cada una de las palabras del vocabulario de las características de la imagen del usuario, esta función nos devuelve las imágenes más similares de la base de datos.

4.1.3. Módulo de Realidad Aumentada

En este módulo aparecen las funciones encargadas de realizar el algoritmo que permite añadir información adicional a la imagen tomada por el usuario. Las funciones que se pueden encontrar en este módulo son las encargadas de realizar el algoritmo explicado en la Figura 3.3. Estas sin

mucho detalle son, una función para la extracción SURF en una determinada región, función para el cálculo de la matriz de homografía H, función para el cálculo de posicionamiento de la región en la imagen del usuario y función para el cálculo de proyección de la imagen interior del edificio.

4.2. Diseño de la Aplicación

A continuación se va a explicar detalladamente las pantallas a través de las cuales el usuario navegará cuando ejecute *CityCam*. En la Figura 4.2 podemos ver un diagrama de navegación general.

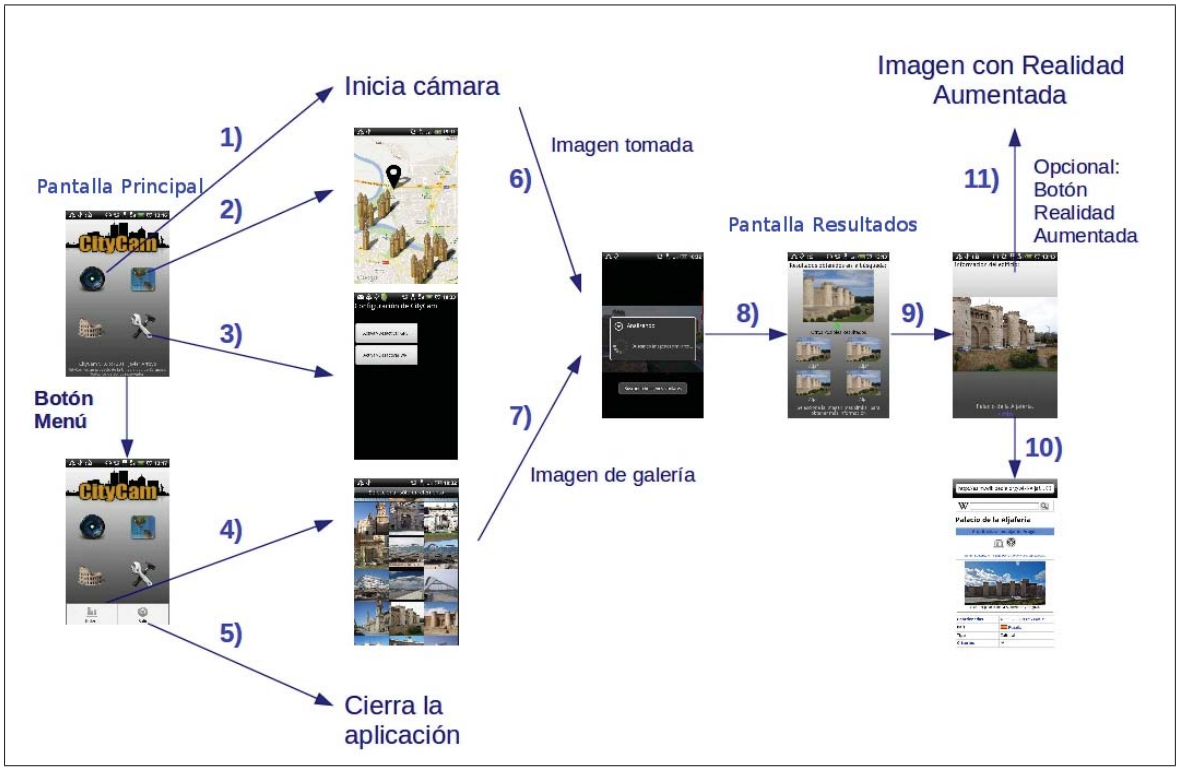


Figura 4.2: **Diagrama de navegación de la aplicación CityCam.**

4.2.1. Pantalla Principal

Al iniciar la aplicación el usuario desde su *smartphone*, la primera pantalla con la que se encontrará es la que se muestra en la Figura 4.3. En ella tendrá básicamente cuatro opciones.

1) Botón Cámara

La principal función de esta aplicación es la que se ejecuta si pulsa el botón superior izquierdo (Figura 4.2 transición *1)*), cuando es pulsado se inicia la cámara de nuestro *smartphone* permitiendo capturar una imagen.

Figura 4.3: **Pantalla Inicio CityCam.** En la imagen izquierda sin pulsar el botón *Menú* y en la imagen derecha pulsado este, pudiendo acceder a imágenes de galería o salir de la aplicación

2) Botón Mapa

La segunda opción es el botón superior derecho, al hacer click en éste (Figura 4.2 transición *2)*), se abrirá Google Maps mostrando todos los edificios registrados en nuestra base de datos (Figura 4.4).

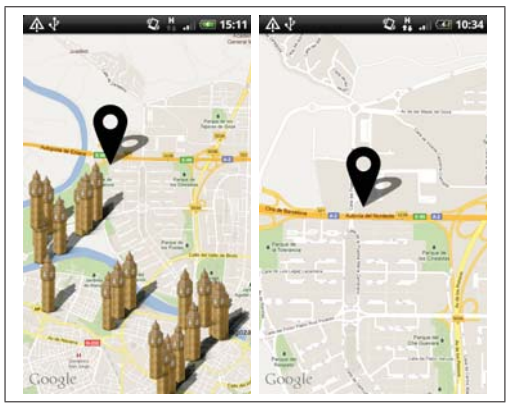


Figura 4.4: **Pantalla Mapa.** En esta pantalla se muestra el mapa de Google Maps con los diferentes edificios de nuestra base de datos situados sobre el mapa.

3) Botón Configuración

Con la pulsación del botón inferior derecho (Figura 4.2 transición *3)*), accederemos a una pantalla donde se podrá configurar que la aplicación acceda a redes Wi-Fi y al GPS, para que la localización sea más precisa y por tanto los resultados mostrados al usuario mejores.

4) Botón Galería (Menú)

Pulsando el botón *Menú* de nuestro dispositivo, accederemos a un submenú, donde aparecen las opciones *Galería* y *Salir*. Presionado *Galería*, accedemos a nuestra galería de imágenes

(Figura 4.5), permitiendo la realización de pruebas de una forma más sencilla y rápida.

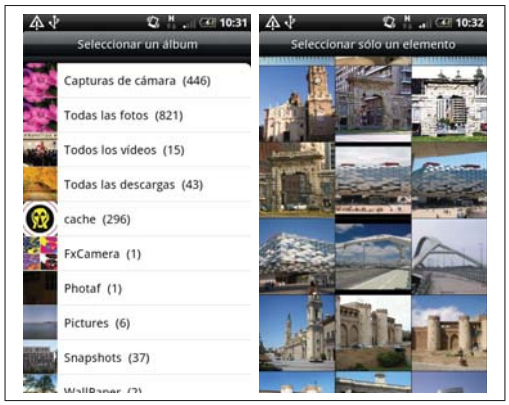


Figura 4.5: **Galería de imágenes.** En esta pantalla aparecen todos los directorios con imágenes de nuestro dispositivo para que se seleccione una para su análisis.

4.2.2. Pantalla Análisis de Imagen

Una vez tomada una imagen o seleccionada de galería(Figura 4.2 transición *6)* ó *7)*), se pasa al análisis de la información de esta para mostrar al usuario aquellas más similares. Este proceso lleva asociado un tiempo (4-8 seg.) donde se visualizará la pantalla mostrada en la Figura 4.6.

Figura 4.6: **Pantalla de Análisis de Imagen.** Pantalla en la cual el usuario deberá esperar a que se produzca el análisis de la imagen.

4.2.3. Pantalla Resultados

Cuando el análisis haya acabado(Figura 4.2 transición *8)*), ya se podrá mostrar al usuario las imágenes de la base de datos con mayor similitud, mostrándolas como aparece en la Figura 4.7.

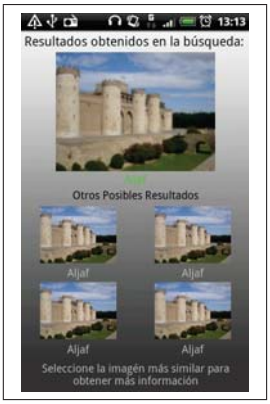


Figura 4.7: **Pantalla de resultados tras el análisis.** En esta pantalla se muestran las imágenes de la base de datos que tras el análisis han resultados más similares.

4.2.4. Pantalla información del Edificio

Una vez el usuario selecciona uno de los resultados mostrados (Figura 4.2 transición *9)*), se muestra la imagen tomada por este con información adicional, así como un *link* para que tenga más información de forma directa 4.8. Si este *link* es pulsado, se abrirá nuestro navegador web por defecto, redirigiéndonos a una web con más información sobre el edificio.

Figura 4.8: **Pantalla con información del edificio.** En la imagen izquierda aparece la imagen tomada por el usuario con información a cerca del edificio. Si es pulsado el *link +info* se abrirá el navegador apareciendo más información del edificio, imagen derecha.

4.2.5. Pantalla Realidad Aumentada

En algunos ejemplos, en la última pantalla (Figura 4.8) aparecerá el botón *Realidad Aumentada* que permite al usuario mostrar la imagen con información virtual sobre su imagen, en este caso será una imagen del interior del edificio.

Capítulo 5

Conclusiones

Este capítulo contiene las conclusiones extraídas tras la realización de este Proyecto Final de Carrera y una propuesta de posibles líneas de trabajo futuro. Al final se presenta una valoración de lo que su realización ha supuesto a nivel personal.

5.1. Conclusiones

El avance en la tecnología en dispositivos móviles en los últimos años ha ocasionado que el desarrollo de aplicaciones para estos dispositivos crezca de manera increíble. Además, la capacidad de computo de estos, así como la calidad de sus cámaras ha provocado que en la actualidad se puedan implementar aplicaciones que años antes serían impensables que pudieran ejecutarse en tales dispositivos. Gracias a ello y con el deseo de crear una aplicación para un teléfono móvil, combinada con técnicas del campo de la visión por computador, se ha desarrollado este proyecto.

Los objetivos principales de este proyecto han sido, en primer lugar construir un algoritmo que permita evaluar la similitud de las imágenes con respecto a unas imágenes de referencia para su posterior implementación en un dispositivo móvil, así como un sistema de realidad aumentada para presentar los resultados del reconocimiento. Objetivos cumplidos e implementados en un dispositivo móvil, obteniendo unos resultados satisfactorios.

Se han estudiado diferentes métodos para la evaluación de similitud entre imágenes. Se comenzó estudiando los métodos de representación completa (subsección 2.3.1), pero tras su estudio, implementación y pruebas se observó que estos métodos no eran los más apropiados para ser implementados en un dispositivo con tales limitaciones de memoria y capacidad de computo. Tras esta decisión, se pasó a estudiar los métodos de bolsa de palabras (subsección 2.3.2) que según la bibliografía ofrecían resultados satisfactorios con una eficiencia en ocupación de memoria y tiempo de ejecución mucho más óptimos que los métodos de representación completa. Tras la realización exhaustiva de pruebas, se concluyó que este método obtenían unos resultados muy buenos, por lo que se pasó a implementar en una aplicación Android en la cual se hiciera uso del algoritmo desarrollado. Tras el estudió del funcionamiento de Android, y la forma con la

que utilizar las librerías OpenCV en Android, se obtuvo como resultado la aplicación CityCam, detallada en el Capítulo 4.

Tras el desarrollo de los métodos de evaluación de similitud, como se había propuesto, se estudió y desarrollo métodos para utilizar técnicas de realidad aumentada. Como se ha podido observar los resultados son satisfactorios aunque hay ocasiones en los que el método no funciona del todo correctamente, por lo que en esta línea de investigación se deja abierto un amplio campo de desarrollo.

5.2. Trabajo futuro

Tras la realización del presente proyecto, se proponen a continuación algunas posibles líneas de trabajo futuro:

- Mejorar el modelo de bolsa de palabras desarrollado de manera que el sistema pudiera verse ampliado conforme los usuarios realizaran búsquedas de imágenes no presentes en la base de datos de referencia, incluyendo esa información.
- Mejorar el algoritmo de realidad aumentada implementando, utilizando otras técnicas que ofrezcan unos resultados mejores.
- Ofrecer al usuario una visión de realidad aumentada de forma directa mediante vídeo.

5.3. Valoración personal

La realización de este proyecto ha ampliado mis conocimientos, desde darme la oportunidad de aprender a manejar un nuevo sistema operativo, como es Android, el desarrollo en un entorno de trabajo no del todo conocido, hasta el aprendizaje de conceptos mucho más teóricos sobre la visión por computador. También, explorar el campo de la programación sobre dispositivos móviles me ha aportado una visión mucho más cuidadosa hacia la programación, ya que en estos dispositivos, los recursos son limitados.

A nivel personal, he aprendido a ser constante y a dar la importancia que requiere al diseño de las aplicaciones. He perdido el miedo a enfrentarme con lo desconocido y a ser más paciente a la hora de obtener unos buenos resultados. Además he aprendido que un proyecto se puede realizar manteniendo reuniones vía Internet, dada la situación de Ana Cristina en los meses comprendidos entre Julio y Diciembre en los cuales se encontraba en San Diego(Estados Unidos). Pero sobre todo, que el esfuerzo invertido ha merecido la pena.