Apéndice A

Experimentos detallados con Representación Completa

En este anexo se detallan los diferentes experimentos realizados para evaluar la similitud entre imágenes utilizando métodos basados en la representación completa. Los experimentos que se detallan a continuación, 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.

Para la realización de los primeros experimentos, se comenzó mediante los métodos de búsqueda de emparejamiento haciendo uso de la búsqueda del vecino más cercano (ANN). Como era de esperar los resultados obtenidos utilizando ANN son muchos más adecuados ya que dan resultados similares con tiempos mucho más eficientes

A.1. Experimento 1: Selección de la medida de similitud

Para hacer el cálculo de la similitud de una imagen de prueba con las imágenes de referencia se plantearon varias medidas de similitud. Dadas dos Im_i y Im_j, las medidas de similitud estudiadas han sido las siguientes:

$$Sim_0(Im_i, Im_j) = |m_{ij}| \tag{A.1}$$

$$Sim_1(Im_i, Im_j) = \frac{|m_{ij}|}{Min(|s_i|, |s_j|)} \tag{A.2}$$

$$Sim_2(Im_i, Im_j) = \frac{|m_{ij}|}{|s_{referencia}|} \tag{A.3}$$

$$Sim_3(Im_i, Im_j) = \frac{|m_{ij}|}{Max(|s_i|, |s_j|)} \tag{A.4}$$

$$Sim_4(Im_i, Im_j) = (0{,}7 * \frac{|m_{ij}|}{|s_i|} + 0{,}3 * \frac{|m_{ij}|}{|s_j|}) * \frac{1}{0{,}2 * dE} \tag{A.5}$$

Siendo:

- $|m_{ij}|$ número de correspondencias encontradas entre la imagen i y j.
- $|s_i|$ número de características SURF encontradas en la imagen i.
- dE distancia euclídea para cada correspondencia encontrada entre i y j. Cuanto mayor sea la distancia, menor es la similitud entre Im_i e Im_j.

Sim_0 se desechó desde el principio, ya que dependiendo del tamaño de la imagen o la forma en que este tomada, en la extracción de características SURF se pueden obtener más o menos puntos, por lo que aquellas imágenes con más características tendrán una probabilidad mayor de ser seleccionadas.

Dadas estas 4 medidas de similitud (desechada Sim_0), lo primero que se realizó fueron pruebas para observar aquellas medidas que obtenían unos mejores resultados en la correcta identificación de edificios.

Configuración del experimento:

- Base de datos de referencia compuesta por 26 imágenes:
  - Aljafería: 7 imágenes
  - Puerta del Carmen: 12 imágenes
  - Puerta Cinegia: 7 imágenes
- Imágenes de test:
  - Aljaferia: 3 imágenes
  - Puerta del Carmen: 3 imágenes
  - Puerta Cinegia: 3 imágenes

Configuración del algoritmo de extracción de puntos SURF:

- Threshold: 500
- Error: 0.7
- Tamaño descriptor: 64 floats

Resultados obtenidos:

En la tabla A.1 aparece el porcentaje de acierto en la búsqueda de la imagen más similar. Pero para llegar a estos resultados se analizaron los resultados tal y como se muestra en la figura A.1 donde se muestra un ejemplo de comparación de una imagen de prueba con las imágenes de referencia.

	Sim_1 A.2	Sim_2A.3	Sim_3A.4	Sim_4 A.5
% Acierto	33.33	44.44	55.55	55.55

Tabla A.1: Resultados Experimento 1

ImagenBD	Pares Encontrados	PSurfTest	PSurfBD	Ponderacion
Cinegia2.jpg---->				
Cinegia 008.jpg	433	795	2433	0,17797
Aljaferia_018.jpg	44	795	322	0,055346
Aljaferia_010.jpg	132	795	663	0,166038
Cinegia 011.jpg	225	795	1679	0,134008
Cinegia 003.jpg	374	795	2345	0,159488
Cinegia 009.jpg	408	795	2299	0,177468
Cinegia 001.jpg	340	795	2091	0,162602
Carmen 015.jpg	159	795	1616	0,098391
Carmen 009.jpg	164	795	2077	0,07896
Aljaferia_022.jpg	50	795	228	0,062893
Carmen 004.jpg	215	795	2461	0,087363
Carmen 010.jpg	166	795	1910	0,086911
Cinegia 005.jpg	330	795	2256	0,146277
Carmen 021.jpg	159	795	1254	0,126794
Carmen 006.jpg	184	795	2167	0,08491
Aljaferia_021.jpg	46	795	234	0,057862
Cinegia 010.jpg	275	795	1947	0,141243
Carmen 016.jpg	128	795	1353	0,094605
Carmen 007.jpg	172	795	2075	0,082892
Carmen 020.jpg	160	795	1275	0,12549
Carmen 022.jpg	147	795	1530	0,096078
Carmen 017.jpg	122	795	1496	0,081551
Aljaferia_009.jpg	41	795	396	0,051572
Aljaferia_016.jpg	68	795	365	0,085535
Carmen 003.jpg	229	795	2291	0,099956
Aljaferia_020.jpg	50	795	306	0,062893
La eleccion para Cinegia2.jpg es: Cinegia 008.jpg				

Figura A.1: Ejemplo de resultados obtenidos en cada comparación

Discusión de los resultados:

A la vista de los resultados obtenidos tras este experimento, se decidió que en los siguientes experimentos se utilizaría como medidas de similitud las que mayor porcentaje de acierto obtuvieron. Medida de *similitud1*:

$$Sim_1 = \frac{Matches_{ij}}{Max(NumSURF_i, NumSURF_j)} \tag{A.6}$$

Y como medida de *similitud2*:

$$Sim_2 = (0{,}7 * \frac{Matches_{ij}}{NumSURF_i} + 0{,}3 * \frac{Matches_{ij}}{NumSURF_j}) * \frac{1}{0{,}2 * distanciaMedia} \tag{A.7}$$

A.2. Experimento 2: Comprobación de la implementación del algoritmo

Conocidas las medidas de similitud que se iban a emplear se pasó a la realización de pruebas con bases de datos de mayor tamaño y un número mayor de imágenes de consulta.

Para la realización de estos experimentos, lo primero que se realizó fue la comprobación de la correcta implementación de los métodos. Para ello se seleccionaron un conjunto de imágenes de la base de datos de imágenes un Proyecto fin de carrera anterior relacionado [13].

Configuración del experimento:

- Base de datos de referencia compuesta por 67 imágenes:
  - Aljafería
  - Puerta del Carmen
  - Puerta Cinegia
  - Edificio de Correos
  - Edificio de la Diputación Provincial de Zaragoza
  - Edificios de Zurich
- Imágenes de test:
  - 35 imágenes de los diferentes edificios

Configuración del algoritmo:

- Threshold: 500
- Error: 0.7
- Tamaño descriptor: 64 bits

Resultados obtenidos:

	Sim_1 A.6	Sim_2 A.7
Mayor Similitud	94.285 %	94.285 %
3 Primeras	94.285 %	97.142 %
5 primeras	94.285 %	100.0 %

Tabla A.2: Resultados Experimento 2

Discusión de los resultados:

A la vista de los resultados obtenidos, se pudo concluir que los métodos implementados estaban funcionando correctamente, ya que para la realización de esta prueba se seleccionaron imágenes de consulta similares a las que había de referencia, por eso se produjo este elevado porcentaje de acierto en los emparejamientos.

A.3. Experimento 3: Pruebas exhaustivas

Para la realización de este último experimento se utilizó como base de datos un conjunto de 46 imágenes y como consulta 22 imágenes.

El principal objetivo de este experimento fue determinar aquellos parámetros que en la búsqueda obtenían unos buenos resultados y se ejecutaban en un tiempo razonable.

Los parámetros que se decidieron modificar y evaluar su rendimiento fueron:

- Tipo de Búsqueda: ANN vs NN

  Con esta prueba se llegó a la conclusión de que la utilización de la fuerza bruta no proporcionaba unos resultados mejores. Sin embargo, incrementaba el tiempo de ejecución de el algoritmo.

- Nº hojas consultadas (ANN)

  Otro parámetro que se puede modificar es el número de hojas del árbol Kd que se consulta en la búsqueda del vecino más cercano. En la siguiente tabla se encuentran los resultados obtenidos variando este parámetro.

Configuración del experimento:

- Base de datos de referencia compuesta por 46 imágenes, entre ellas se encontraban imágenes descargadas de Internet:
  - Aljafería
  - Puerta del Carmen
  - Puerta Cinegia
  - Edificio de Correos
  - Edificio de la Diputación Provincial de Zaragoza
  - Edificios de Zurich
- Imágenes de test:
  - 22 imágenes de los diferentes edificios

Como se puede observar en la figura A.2 los resultados de imágenes correctamente emparejadas son los mismos utilizando NN (fuerza bruta) o ANN, pero el tiempo de ejecución de el experimento es distinto, se pasa de un tiempo de ejecución de 910 segundos con la utilización la búsqueda ANN a un tiempo de 1129 segundos con la búsqueda NN. Por lo que a la vista de estos resultados se concluyó que no sería interesante seguir realizando pruebas mediante el uso de fuerza bruta.

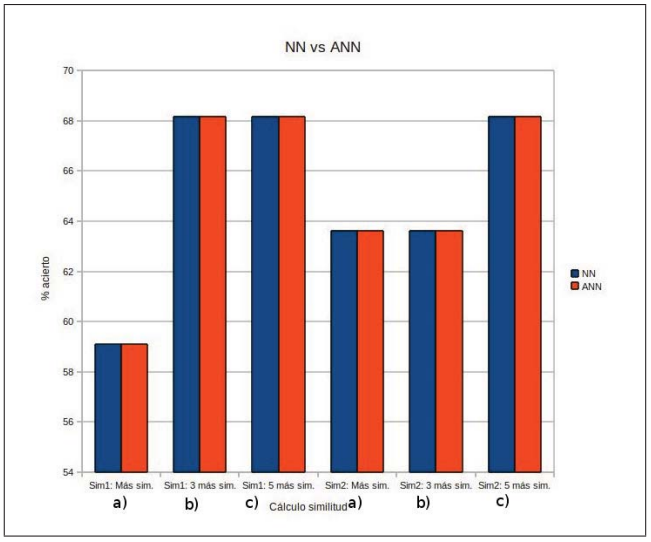


Figura A.2: Resultados obtenidos con búsqueda NN vs ANN

A.4. Conclusiones de los métodos de correspondencia punto a punto

Aunque los resultados son aceptables, este método presenta el inconveniente para la implementación en un dispositivo móvil, de un elevado consumo de memoria y un tiempo de ejecución todavía bastante elevado. Además, como 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 de media unas mil características SURF, y que cada descriptor está compuesto por un vector de 64 "float", la representación ocupará:

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

Siendo d el tamaño de descriptor y n el número de descriptores medio.

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 viable 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.

Apéndice B

Bolsa de Palabras o Clusterización

A continuación se realiza un análisis más detallado del algoritmo implementado usando técnicas de comparación de imágenes basadas en agrupaciones de descriptores. En primer lugar se analiza la función *K-means*, la cual genera el vocabulario de k palabras. En la Sección B.2 se analiza como tratar la información que nos proporciona el algoritmo *K-means* para realizar la evaluación de similitud entre imágenes.

B.1. K-Means

Como implementación del algoritmo *K-means* se utilizó la implementación proporcionada por las librerías OpenCV [1]. A continuación se detallan los parámetros de esta función:

```
1  double kmeans( const Mat & samples, int clusterCount, Mat & labels, TermCriteria termcrit,int attempts, int flags, Mat*
       centers );
```

- Samples: matriz que contiene todos los elementos para realizar el agrupamiento. En este caso, esta matriz contiene todos los descriptores SURF de las imágenes de referencia.
- ClusterCount: entero que indica el número de palabras del vocabulario.
- Labels: matriz que asocia para cada descriptor la palabra con la que se ha asociado.
- TermCrit: especifica el número máximo de iteraciones que se ejecutará el algoritmo, así como la precisión que se desea.
- Attempts: número de veces que el algoritmo se ejecutará con inicializaciones distintas.
- Flags: indica como se realiza la primera inicialización de palabras, es decir, si de manera aleatoria o utilizando alguna otra estrategia de inicialización.
- Centers: matriz donde se encuentra el centro de cada palabra.

Del algoritmo *K-means* se obtiene por lo tanto una matriz que contiene la relación descriptor - etiqueta palabra (*labels*) y otra con los centroides de cada palabra (*centers*). A partir de la

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Palabra_1$	1	4	...	3
$Palabra_2$	2	3	...	9
...	...	...	...	...
$Palabra_n$	6	3	...	5

Tabla B.1: Inverted File Index. Por ejemplo la $Imagen_1$ tiene 1 descriptor asociado a la $Palabra_1$, 2 a la $Palabra_2$ y 6 a la $Palabra_n$

matriz *labels* conocemos a que palabra se asocia cada característica SURF, por lo que podemos construir una matriz (*inverted file index*) que contiene cuantos descriptores asociados con cada una de las palabras del vocabulario hay en cada imagen de referencia. Obteniendo una matriz de la forma (Tabla B.1):

A partir de esta matriz, podemos ya evaluar la similitud de una imagen de prueba con las de referencia. Como se verá en la siguiente sección se ha implementado dos métodos de evaluación.

B.2. Algoritmo de Evaluación de similitud mediante Clusterización

Dado el modelo que hemos construido en el Capítulo 2, podemos determinar que imagen de la base de datos es más similar a la tomada por el usuario. Como ya se resumía en la Sección 2.3.2, los pasos del algoritmo implementado son:

1. Extracción de características SURF de imagen test.

2. Búsqueda de la palabra a la cual se asocia cada característica. Obteniendo así un vector de longitud nº de descriptores 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 (de longitud el número de descriptores) se crea un vector de longitud k (número de palabras del vocabulario) donde se almacena cuantas caráteristicas de cada palabra han ocurrido en esta imagen. Este vector es el histograma de palabras que aparecen en esta imagen.

4. Haciendo uso de este histograma se pueden estimar dos medidas de similitud:

   - Sim1: Comparación basada en los histogramas de palabras de cada imagen: La idea principal de esta medida se basa en el concepto de comparación de histogramas de palabras. Un histograma de palabras asociado a una imagen representa la frase que se asocia a dicha imagen. Se seleccionará como imagen más similar aquella cuya distancia entre histogramas (norma-L1) sea menor.

     Veámoslo con un ejemplo:

     Supongamos que tenemos el *inverted file index* siguiente:

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Palabra_1$	5	6	...	1
$Palabra_2$	7	2	...	4
...	...	...	...	...
$Palabra_n$	2	4	...	8

Y el histograma asociado a la imagen de consulta:

	$Palabra_1$	$Palabra_2$	...	$Palabra_n$
$Imagen_{consulta}$	3	6	...	5

Se compara el vector $(3, 6, \ldots, 5)$ de la imagen test con el vector $(5, 7, \ldots, 2)$ de la imagen1 de referencia. Utilizando la distancia norma1-L1 se obtiene como resultado:

$$\begin{array}{rcccc} & 3 & 6 & \ldots & 5 \\ - & 5 & 7 & \ldots & 2 \\ \hline & 2 & 1 & \ldots & 3 \end{array}$$

$\Longrightarrow Diferencia = 6$

$$\begin{array}{rcccc} & 3 & 6 & \ldots & 5 \\ - & 6 & 2 & \ldots & 4 \\ \hline & 3 & 4 & \ldots & 1 \end{array}$$

$\Longrightarrow Diferencia = 8$

$$\begin{array}{rcccc} & 3 & 6 & \ldots & 5 \\ - & 1 & 4 & \ldots & 8 \\ \hline & 2 & 2 & \ldots & 3 \end{array}$$

$\Longrightarrow Diferencia = 7$

Se seleccionará como imagen más similar a la de consulta la imagen de referencia n, que es aquella cuya distancia L1 (norma 1) es menor.

Más formalmente esta medida de similitud se calcula como se indica en la ecuación (B.1):

$$Dist_1(Imagen_1, Imagen_{consulta}) = |H_1 - H_{consulta}| \tag{B.1}$$

Donde H_1 representa el histograma de palabras asociado a la imagen de referencia1, y $H_{consulta}$ el histograma de palabras asociado a la imagen test.

- 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.

  Veámoslo con un ejemplo:

  Supongamos que tenemos el *inverted file index* siguiente:

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Palabra_1$	5	6	...	1
$Palabra_2$	7	2	...	4
...	...	...	...	...
$Palabra_n$	2	4	...	8

  Y el histograma asociado a la imagen de consulta:

	$Palabra_1$	$Palabra_2$	...	$Palabra_n$
$Imagen_{consulta}$	3	6	...	5

  Se compara palabra a palabra la frecuencia con la que aparece dicha palabra en la imagen consulta y en las de referencia. Aquella imágenes que tengan una frecuencia más parecida recibirán tantos votos como votos a esa palabra existen. Al inicio se inician la votación de todas imágenes a 0.

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Votos$	0	0	...	0

  - $Palabra_1$:

    Imagen1: $|3-5| = 2$

    Imagen2: $|3-6| = 3$

    ...

    ImagenN: $|3-1| = 2$

  $\Longrightarrow$ $Imagen_1$ e $Imagen_n$ reciben 3 votos.

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Votos$	3	0	...	3

  - $Palabra_2$:

Imagen1: $|6-6|=0$

Imagen2: $|6-2|=4$

...

ImagenN: $|6-4|=2$

$\Longrightarrow Imagen_1$ recibe 6 votos.

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Votos$	9	0	...	3

- $Palabra_n$:

  Imagen1: $|5-1|=4$

  Imagen2: $|5-4|=1$

  ...

  ImagenN: $|5-8|=3$

$\Longrightarrow Imagen_1$ recibe 5 votos.

	$Imagen_1$	$Imagen_2$	...	$Imagen_n$
$Votos$	9	5	...	3

$\Rightarrow$ Resultado: se selecciona como imagen más similar $Imagen_1$.

Más formalmente esta similitud se calcula como se indica en la ecuación (B.3):

$$Votos(I_i, w_j) = \begin{cases} H_c(j) \text{ si } [H_c(j) - H_i(j)] == min([H_c(j) - H_1(j)]), ..., [H_c(j) - H_n(j)] \\ 0 \text{ en cualquier otro caso} \end{cases} \tag{B.2}$$

$$Votos_i = \sum_{j=1}^{n} Votos(I_i, w_j) \tag{B.3}$$

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.

Apéndice C

Experimentos detallados con Bolsa de Palabras

A continuación se van a detallar los diferentes experimentos realizados para evaluar los métodos de evaluación de similitud basados en bolsa de palabras.

Las primeras pruebas se realizaron utilizando como imágenes de referencia subconjuntos de la base de datos de imágenes de referencia final. En los últimos experimentos se hace uso de la base de datos con todas imágenes de referencia.

C.1. Experimentos de Clusterización sin utilización GPS

C.1.1. Experimento 1

Como imágenes de referencia se tomaron 30 imágenes de Expo Zaragoza 2008, y como consulta 21 imágenes. En la tabla C.1 se encuentran detalladas las imágenes utilizadas para el experimento.

	Acuario	PAragón	Hiberus	MDigital	PCongresos
$\text{Imágenes}_{referencia}$	2	3	1	3	3
Imágenes_{test}	1	2	2	2	1
	PEspana	Estatua	PPuente	TercerMil	TorreAgua
$\text{Imágenes}_{referencia}$	4	3	3	4	4
Imágenes_{test}	2	1	1	3	6

Tabla C.1: Imágenes utilizadas en Experimento1

- Prueba1:

  Configuración:

N° Palabras	**200**
N° Iteraciones	10
N° Inicializaciones	200
Error	0.05

Tabla C.2: Configuración Prueba1 - Experimento1

Resultados Prueba1:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	15/21 = 71,42 %	17/21 = 80,95 %	17/21 = 80,95 %
Med. Sim2 (B.3)	15/21 = 71,42 %	18/21 = 85,71 %	18/21 = 85,71 %

Tabla C.3: Resultados Prueba1- Experimento1

Tiempo ejecución: 132.5 s.

- Prueba2:

Configuración:

$N°$ Palabras	**500**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.4: Configuración Prueba2 - Experimento1

Resultados Prueba2:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	13/21 = 61,90 %	16/21 = 76,19 %	17/21 = 80,95 %
Med. Sim2 (B.3)	15/21 = 71,42 %	18/21 = 85,71 %	18/21 = 85,71 %

Tabla C.5: Resultados Prueba2 - Experimento1

Tiempo ejecución: 231.35 s.

- Prueba3:

Configuración:

$N°$ Palabras	**3000**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.6: Configuración Prueba3 - Experimento1

Resultados Prueba3:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	9/21 = 42,85 %	11/21 = 52,38 %	12/21 = 57,14 %
Med. Sim2 (B.3)	18/21 = 85,71 %	19/21 = 90,47 %	19/21 = 90,47 %

Tabla C.7: Resultados Prueba3 - Experimento1

Tiempo ejecución: 1049.48 s.

C.1.2. Experimento 2

Para la realización de esta prueba se tomó como base de datos de imágenes de referencia 45 imágenes de Expo Zaragoza 2008 junto con algunas de edificios del centro de Zaragoza. En la tabla C.8 se describe más detalladamente las imágenes utilizadas en este experimento. Como imágenes de test se utilizaron 37 imágenes.

	Acuario	PAragón	Hiberus	MDigital	PCongresos
$\text{Imágenes}_{referencia}$	2	3	1	3	3
Imágenes_{test}	1	5	1	2	2
	PEspana	Estatua	PPuente	TercerMil	TorreAgua
$\text{Imágenes}_{referencia}$	4	3	4	4	6
Imágenes_{test}	4	2	2	3	6
	PAljafería	PCarmen	PCinegia	DipProv	TorreAgua
$\text{Imágenes}_{referencia}$	3	3	3	3	
Imágenes_{test}	3	2	2	1	7

Tabla C.8: Imágenes utilizadas en Experimento2

- Prueba1:

  Configuración:

$N°$ Palabras	**200**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.9: Configuración Prueba1 - Experimento2

Resultados Prueba1:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	$21/37 = 56{,}75\,\%$	$25/37 = 67{,}56\,\%$	$25/37 = 67{,}56\,\%$
Med. Sim2 (B.3)	$21/37 = 56{,}75\,\%$	$26/37 = 70{,}27\,\%$	$30/37 = 80{,}08\,\%$

Tabla C.10: Resultados Prueba1 - Experimento2

Tiempo ejecución: 179.83 s.

- Prueba2:

  Configuración:

$N°$ Palabras	**500**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.11: Configuración Prueba2 - Experimento2

Resultados Prueba2:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	$18/37 = 48{,}64\,\%$	$25/37 = 67{,}56\,\%$	$25/37 = 67{,}56\,\%$
Med. Sim2 (B.3)	$25/37 = 67{,}56\,\%$	$30/37 = 81{,}08\,\%$	$31/37 = 83{,}78\,\%$

Tabla C.12: Resultados Prueba2 - Experimento2

Tiempo ejecución: 302.92 s.

- Prueba3:

  Configuración:

$N°$ Palabras	**1000**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.13: Configuración Prueba3 - Experimento2

Resultados Prueba3:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	$17/37 = 45{,}94\,\%$	$19/37 = 51{,}35\,\%$	$22/37 = 59{,}45\,\%$
Med. Sim2 (B.3)	$25/37 = 67{,}56\,\%$	$29/37 = 78{,}37\,\%$	$29/37 = 78{,}37\,\%$

Tabla C.14: Resultados Prueba3 - Experimento2

Tiempo ejecución: 513.95 s.

C.1.3. Experimento 3

Para la realización de esta prueba se tomó como base de datos de imágenes de referencia 146 imágenes de Expo Zaragoza 2008, centro y casco histórico de Zaragoza, así como imágenes de edificios del municipio de Andorra. En la tabla C.15 se describe más detalladamente. Como imágenes de test se utilizaron 117 imágenes diferentes de estos edificios, entre las cuales se encuentran algunas obtenidas de Internet.

	Acuario	Aljafería	Pab. Aragón	Ayunt.Zgz	Ayunt.And
Im_{ref}	2	8	3	5	4
Im_{test}	1	7	8	3	1
	Capitanía	P.Carmen	Casa And	P.Cinegia	Ed. Comarca And
Im_{ref}	2	6	2	6	2
Im_{test}	1	8	1	5	2
	Correos Zgz	Diputación	Er.Pilar	SanMacario	Pab.España
Im_{ref}	1	5	2	2	4
Im_{test}	3	5	4	3	3
	P.Europa	Hiberus	Ed.Ibercaja	Pob.Ibero	Est.Iranzo
Im_{ref}	3	1	4	1	3
Im_{test}	3	1	2	1	2
	La Seo	Pl.Amadeo	M.SanMac	Merc.Central	MillaDig
Im_{ref}	4	4	3	3	3
Im_{test}	3	2	1	5	2
	M.Tambor	MuseoZgz	P.Congresos	Paraninfo	Peña
Im_{ref}	3	4	3	3	3
Im_{test}	1	5	2	2	1
	Pilar	Pisos Alj.	Pozo And.	Puente3	P.Regallo
Im_{ref}	8	3	6	4	4
Im_{test}	4	2	5	3	2
	Mon.S.Santa	M.Tambor	PuenteMil	TorreAgua	
Im_{ref}	3	4	4	6	
Im_{test}	1	3	2	4	

Tabla C.15: Imágenes utilizadas en Experimento3

- Prueba1:

  Configuración:

$N°$ Palabras	**600**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.16: Configuración Prueba1 - Experimento3

Resultados Prueba1:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	$89/117 = 76{,}06\,\%$	$95/117 = 81{,}19\,\%$	$96/117 = 82{,}05\,\%$
Med. Sim2 (B.3)	$73/117 = 62{,}39\,\%$	$89/117 = 76{,}06\,\%$	$93/117 = 79{,}48\,\%$

Tabla C.17: Resultados Prueba1 - Experimento3

Tiempo ejecución: 1373.84 s.

- Prueba2:

$N°$ Palabras	**1000**
$N°$ Iteraciones	10
$N°$ Inicializaciones	200
Error	0.05

Tabla C.18: Configuración Prueba2 - Experimento3

Configuración:

Resultados Prueba2:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 (B.1)	$87/117 = 74{,}35\,\%$	$90/117 = 76{,}92\,\%$	$96/117 = 82{,}05\,\%$
Med. Sim2 (B.3)	$80/117 = 68{,}37\,\%$	$96/117 = 82{,}05\,\%$	$99/117 = 84{,}61\,\%$

Tabla C.19: Resultados Prueba2 - Experimento3

Tiempo ejecución: 2060.76 s.

C.2. Experimentos de Clusterización con Posicionamiento Global GPS

Tras los resultados obtenidos en los experimentos anteriores, se decidió que el algoritmo a implementar en el dispositivo móvil iba a ser el algoritmo de bolsa de palabras. Tras tomar esta decisión se pasó a experimentar que resultados nos ofrece la inclusión de la información proporcionada por el GPS en la evaluación de similitud de imágenes.

Dado que hoy en día la mayoría de móviles del mercado, *smartphones*, disponen de Sistema de Posicionamiento Global (GPS), se pensó que esta herramienta nos podría ayudar a restringir las búsquedas a la hora de evaluar la similitud de cada edificio con una imagen tomada por el usuario, así como dar unos resultados de búsqueda más coherentes.

Para poder incluir la ayuda que nos proporciona el GPS, lo primero que se tuvo que realizar fue la geolocalización de cada uno de los edificios con los que se estaba trabajando. Para llevar a cabo esto, se hizo uso de Google Maps, ya que este nos proporciona las coordenadas de todo punto en un mapa, pudiendo así extraer con precisión las coordenadas de cada edificio.

Una vez se tenían las coordenadas tanto de la imagen de referencia como de test, se calculaba la distancia entre estas, y si no se encontraba dentro de un rango que varía desde 0.5 Km hasta 1 Km, esa imagen no se seleccionaba entre las más votadas, obteniendo así unos resultados mejores y más coherentes.

C.2.1. Experimento 1

Con este primer experimento se trató de demostrar que la implementación de la parte del GPS era correcta. Como se observa en los resultados obtenidos, el porcentaje de acierto en la búsqueda de imagen más similar se ha incrementado, así como los resultados en la búsqueda son más coherentes.

Configuración:

$N°$ Palabras	**1000**
$N°$ Imágenes Consulta	18
Radio entre coordenadas	1 Km

Tabla C.20: Configuración Experimento 1 GPS

Resultados:

	Imagen Más votada	3 Más votadas	5 Más votadas
Med. Sim1 sin GPS(B.1)	16/18 = 88,88 %	16/18 = 88,88 %	16/18 = 88,88 %
Med. Sim1 con GPS(B.1)	18/18 = 100 %	18/18 = 100 %	18/18 = 100 %
Med. Sim2 sin GPS(B.3)	12/18 = 66,66 %	15/18 = 83,33 %	16/18 = 88,88 %
Med. Sim2 con GPS(B.3)	17/18 = 94,44 %	18/18 = 100 %	19/18 = 100 %

Tabla C.21: Resultados Experimento1 GPS

A la vista de los resultados obtenidos, se puede observar que el porcentaje de acierto a la hora de encontrar la imagen más similar se ha incrementado, en la mayoría de los casos al 100 %. Otro aspecto muy importante que se puede observar es que los resultados de imágenes más similares también son más coherentes. Solo se muestran resultados que se encuentran dentro de un perímetro razonable, no se muestran resultados de otras ciudades o edificios muy lejanos.

Ejemplo: Imagen de prueba Palacio de la Aljafería

	Sin GPS	Con GPS
Resultado0	Aljafería	Aljafería
Resultado1	Mercado Central	Aljafería
Resultado2	Pilar	Aljafería
Resultado3	Pozo SJuan(And)	Aljafería
Resultado4	Pilar	Plaza Europa

Tabla C.22: Tabla comparativa resultados sin GPS y con GPS. Sin GPS aparecen resultados de edificios lejanos incluso de otras ciudades. Mientras que al incluir el GPS los resultados son más coherentes y sólo se muestran resultados de edificios cercanos.

C.2.2. Experimento 2

En este experimento se utilizó una colección de imágenes como referencia y consulta mucho más amplia (Tabla C.23). Lo que también se trató de determinar con esta prueba es como afecta el radio de distancia entre las coordenadas de la imagen de referencia y consulta en los resultados mostrados al usuario.

C.2.3. Experimento Final

En esta prueba se detalla las ventajas de la utilización de GPS frente a su no utilización, así como la mejora en el tiempo de ejecución mediante la utilización de *k-d trees* frente a la

	Más votada	3 Más votadas	5 Más votadas
Med.Sim1(B.1)	88/116 = 75,86 %	94/116 = 81,03 %	95/116 = 81,89 %
GPS (1 Km)Sim1(B.1)	88/116 = 75,86 %	92/116 = 79,31 %	95/116 = 81,89 %
GPS (0.5 Km)Sim1(B.1)	90/116 = 77,58 %	98/116 = 84,48 %	100/116 = 86,2 %
Med.Sim2(B.3)	72/116 = 62,06 %	88/116 = 75,86 %	92/116 = 79,31 %
GPS (1 Km)Sim2(B.3)	88/116 = 75,86 %	94/116 = 81,03 %	97/116 = 83,62 %
GPS (0.5 Km)Sim2(B.3)	91/116 = 78,44 %	97/116 = 83,62 %	101/116 = 87,06 %

Tabla C.23: Tabla comparativa resultados sin GPS vs GPS y distancia

búsqueda de palabra exhaustiva de cada punto SURF de la imagen de consulta. La configuración de la prueba ha sido:

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

En las Figuras C.1 y C.2 se observan los resultados de esta pruebas. La primera (C.1) muestra los resultados con la medida de similitud Sim1(ec. 2.1) (Tabla C.24) y Sim2(ec. 2.3) (Tabla C.25). Se utilizan estructuras *k-d tree* para la búsqueda de la palabra asignada a cada punto SURF extraido de la imagen consulta. La Figura C.2, sin embargo muestra los resultados de la prueba haciendo una búsqueda exhaustiva de el cluster asociado a cada punto SURF de la imagen de test.

Asignación de palabras con *k-d tree*

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.

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 de la prueba con *k-d trees* de 401 s a 3700 s algo que hay que tener en cuenta ya que en dispositivo móvil aún se

	Top 1	Top 3	Top 5
Sin GPS	65.24 %	76.59 %	80.85 %
GPS 1 Km	78.01 %	88.65 %	91.48 %
GPS 0.5 Km	84.39 %	91.48 %	93.61 %

Tiempo de cada test 0.2 s

Tabla C.24: Resultados 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 %
GPS 1 Km	63.82 %	78.01 %	83.68 %
GPS 0.5 Km	74.46 %	86.52 %	92.19 %

Tiempo de cada test 0.2 s

Tabla C.25: Porcentaje de imágenes reconocidas correctamente con k-d tree y Sim2 (ec.2.3)

	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 C.26: Porcentaje de imágenes reconocidas correctamente con búsqueda exhaustiva y Sim1 (ec.2.1)

	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 C.27: Porcentaje de imágenes reconocidas correctamente con búsqueda exhaustiva y Sim2 (ec.2.3)

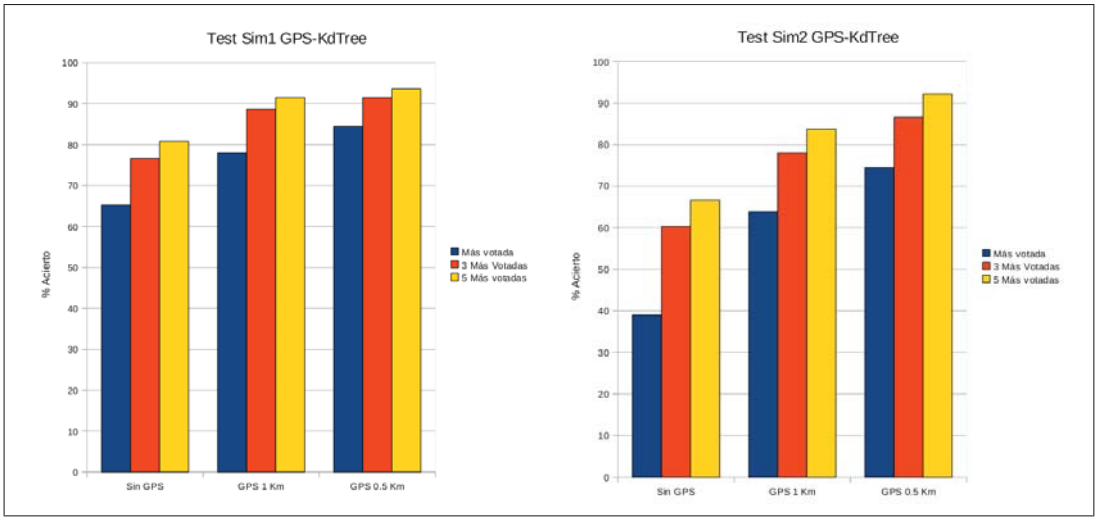


Figura C.1: Aciertos/Rendimiento utilizando o no filtro GPS y k-d tree

verá más acentuado. El tiempo medio de búsqueda de clúster con búsqueda exhaustiva es 15.42 s frente a 0.2 s mediante la utilización de *k-d trees*.

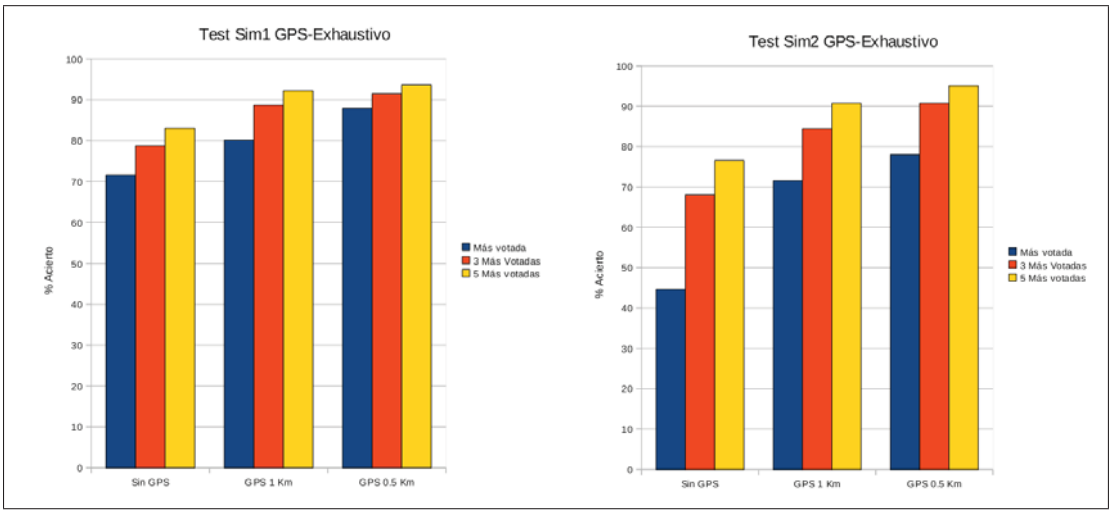


Figura C.2: Aciertos/Rendimiento utilizando o no filtro GPS y búsqueda exhaustiva

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*

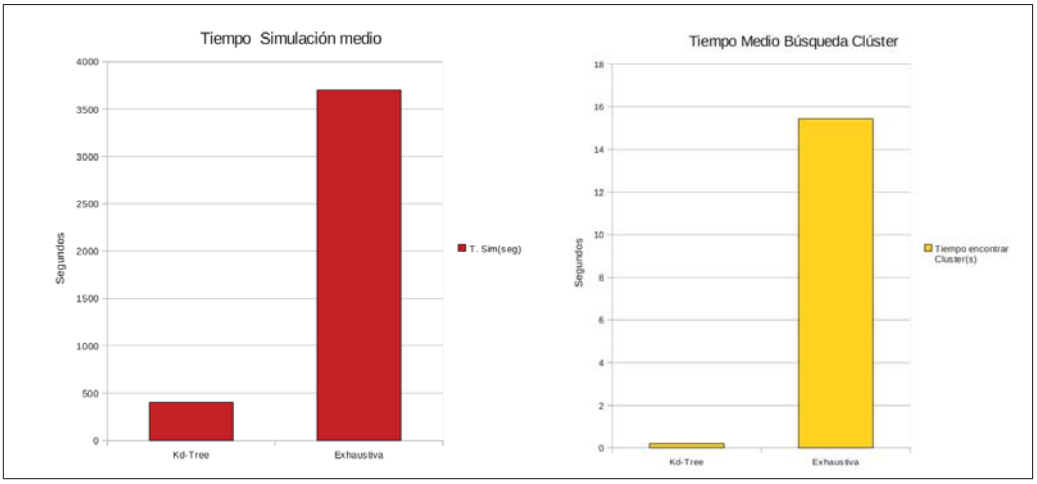


Figura C.3: Tiempo de ejecución de evaluación de similitud asignando las palabras utilizando *k-d tree* ó búsqueda exhaustiva

- 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. B.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.

Apéndice D

Android

D.1. Introducción

En este capítulo se van a detallar algunas características básicas del SO Android, en el cuál se ejecutará nuestra aplicación. También se detalla como se ha conseguido enlazar las librerías de OpenCV para su utilización en un dispositivo con SO Android.

D.2. Android OS

Android es un sistema operativo que inicialmente se desarrolló para ser implantado en teléfonos móviles inteligentes al igual que iOS, Symbian y Blackberry OS. Posteriormente, se extendió su desarrollo a otros tipos de dispositivos como PCs, tablets, netbooks, etc. Lo que lo hace diferente a sus competidores, es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

La estructura del sistema operativo Android se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual Dalvik con compilación en tiempo de ejecución. Dalvik es una máquina virtual de proceso dónde se ejecutan las aplicaciones Android. A diferencia de la máquina virtual de Java, cuya arquitectura está basada en una máquina de pila, Dalvik esta basada en un arquitectura de máquina de registros. Los programas se escriben en lenguaje Java y se genera su bytecode, con el compilador Java. Una vez compilados, se convierten al formato que ejecuta Dalvik, .dex(Dalvik Executable) antes de la instalación de la aplicación en el dispositivo. El formato de estos ficheros ejecutables está pensado para ser ejecutados en dispositivos con limitaciones de memoria y velocidad de proceso.

El sistema operativo proporciona todas las interfaces (ver Figura D.1) necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.) de una forma muy sencilla en un lenguaje de programación muy conocido como es Java.

Las aplicaciones se desarrollan habitualmente en el lenguaje Java con el kit de desarrollo de software (Android SDK), Sección D.3, pero están disponibles otras herramientas de desarrollo,

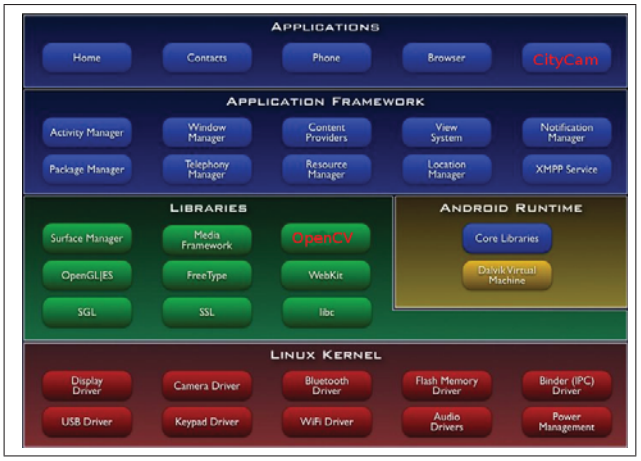


Figura D.1: **Arquitectura del sistema Android.**

incluyendo un kit de desarrollo nativo para aplicaciones o extensiones en C o C++, Sección D.4. Para la programación, lo recomendable es usar el entorno de programación integrado Eclipse[1], el cual permite el acceso a todas las herramientas del SDK de Android de forma sencilla.

D.3. Kit de Dasarrollo Software de Android (SDK)

El kit de desarrollo de software(SDK) de Android[2] es un kit que nos proporciona un conjunto amplio de herramientas para la programación de aplicaciones para la plataforma Android. Estas herramientas son necesarias para cualquier desarrollador que desee programar para la plataforma Android. Se dividen en dos grupos, las herramientas SDK y las herramientas de plataforma. Las primeras son independientes de la versión del SO Android sobre la que se este desarrollando la aplicación, mientras que las herramientas de plataforma se asocian con la versión Android sobre la cual se este desarrollando. Las principales herramientas incluidas son:

- android: permite gestionar proyectos, los componentes instalados del SDK y los dispositivos virtuales Android (AVD).
- Dalvik Debug Monitor Server (ddms): permite depurar aplicaciones Android.
- Android Emulator: herramienta de emulación basada en QEMU que permite diseñar, depurar y probar aplicaciones en un entorno Android.
- sqlite3: permite acceder a bases de datos creadas y accedidas desde un dispositivo Android.
- traceview: proporciona un visor para ver los logs durante la ejecución de la aplicación.

[1] http://www.eclipse.org/
[2] http://developer.android.com/sdk/installing.html

D.4. Kit de Dasarrollo Nativo de Android (NDK)

Este kit[3] nos proporciona un conjunto de herramientas para usar librerías escritas en C/C++ en nuestro proyecto Android. Dado que en este proyecto se hace uso de la librería OpenCV, la cual está escrita en C y C++, tuvimos que hacer uso del NDK. Las clases nativas son llamadas desde el código Java ejecutado en la máquina virtual Dalvik a través de la llamada *System.loadLibrary*, la cual es parte del estándar de las clases Java de Android.

A continuación se detalla como hacer uso de esta potente herramienta que nos proporciona Android, más detalles se pueden encontrar en la web oficial de desarrolladores de Android[4].

1. **Creación de el proyecto Android:** la primera tarea que se debe hacer es crear un proyecto Android en Eclipse. Suponemos que se tiene instalado y configurado el SDK y NDK de Android.

2. **Creación del directorio jni:** Creado el proyecto, se deberá crear un directorio nuevo llamado jni (Java Native Interface) donde se almacenarán todos los ficheros nativos relacionados con el kit NDK. Este directorio actúa como interfaz entre el código Java y el código nativo, permitiendo ejecutar desde la parte Java código C/C++.

3. **Añadimos al directorio jni los ficheros .c/.cpp:** Dentro de el directorio jni estarán almacenados todos los ficheros .c/.cpp. Cuando la máquina virtual invoca a la función nativa, le pasa como argumento un puntero a una variable de entorno (JNIEnv), la cual permite acceder a funciones Java dentro del código C permitiendo gestionar la comunicación entre funciones Java y C. En nuestro fichero .c/.cpp podremos incluir las librerías de OpenCV como son *cv.h* o *cxcore.h*, pudiendo acceder a todas las funciones que estas librerías nos proporcionan. Para que estas librerías sean accesibles habrá que compilar la aplicación indicando su ubicación en el *makefile* llamado *android.mk*. Una función nativa tendrá que tener el siguiente aspecto:

```
1  #include <jni.h>
2  #include <cv.h>  \\librer a de OpenCV
3  JNIEXPORT void JNICALL Java_cps_unizar_LibreriaOpenCV_EJEMPLO(JNIEnv* env, jobject thiz, jdouble real1)
       {
4      // Implementaci n de el m todo nativo.
5  }
```

El puntero JNIEnv *env es una estructura que contiene la interfaz hacia la máquina virtual. Incluye todas las funciones necesarias para interactuar con la DalvikVM y para trabajar con los objetos Java.

[3] http://developer.android.com/sdk/ndk/index.html
[4] http://developer.android.com

Además el nombre de la función nativa debe seguir este patrón: "Java", seguido por el nombre del package, de la clase y de el método donde se definió en la parte Java.

4. **Llamada a código Nativo desde Java:** Una vez escrito el código nativo, este deberá ser llamado desde Java. Dentro de una de las clases, en nuestro caso por claridad se creo una clase LibreríaOpenCV donde se definen los métodos nativos de la siguiente forma. También en este paso se carga la librería nativa para hacer uso de las funciones nativas ("System.loadLibrary").

```
1          package cps.unizar;
2          import java.io.FileDescriptor;
3
4          public class LibreriaOpenCV {
5              static{
6                  System.loadLibrary("milibreria");
7              }
8              //Funciones nativas
9              public native void EJEMPLO(double real1);
10         }
```

5. **Creación del Makefile:** Dentro de el directorio jni es necesario añadir un makefile con el nombre "android.mk". En él indicaremos donde encontrar la librería precompilada de OpenCV para Android, y compile nuestro fichero .cpp, que contiene llamadas a funciones de esta librería.

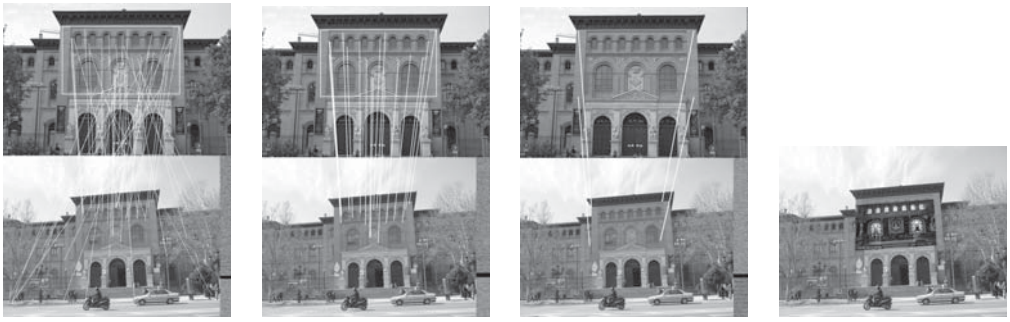
6. **Compilación de el código nativo:** Para la compilación de el código nativo se hace uso de la herramienta ndk-build contenida dentro del kit de desarrollado nativo (NDK) para Android.

Apéndice E

Ejemplos Realidad Aumentada

En este Anexo detallamos más ejemplos que se obtuvieron en la realización de las pruebas para la implementación del algoritmo de Realidad Aumentada.

En todos ellos primero se muestran las correspondencias de la región donde se proyectará la imagen del interior del edificio, la proyección del área de esta región en la imagen del usuario aplicando la matriz de transformación H, así como el resultado final tras la proyección de la imagen interior del edificio.



1) 2) 3) 4)

Figura E.1: Ejemplo. 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.

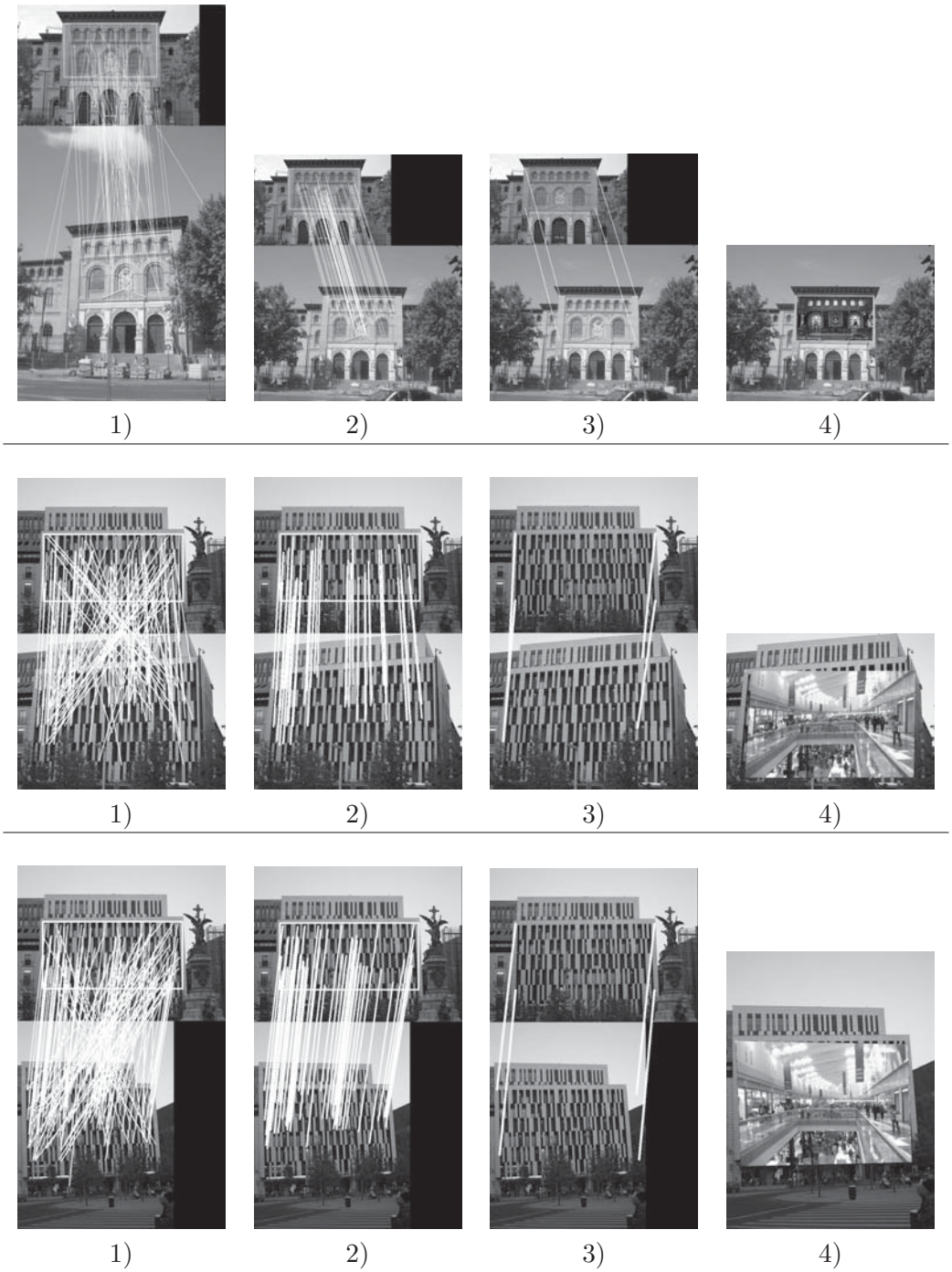


Figura E.2: Ejemplo. En la imagen 1) se observan todas las correspondencias dentro de la región donde se proyectará la imagen interior. En la imagen 2) tras calcular la matriz de homografía H y eliminar aquellas correspondencias no consistentes, sólo quedan las correctas. En la imagen 3) proyectamos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

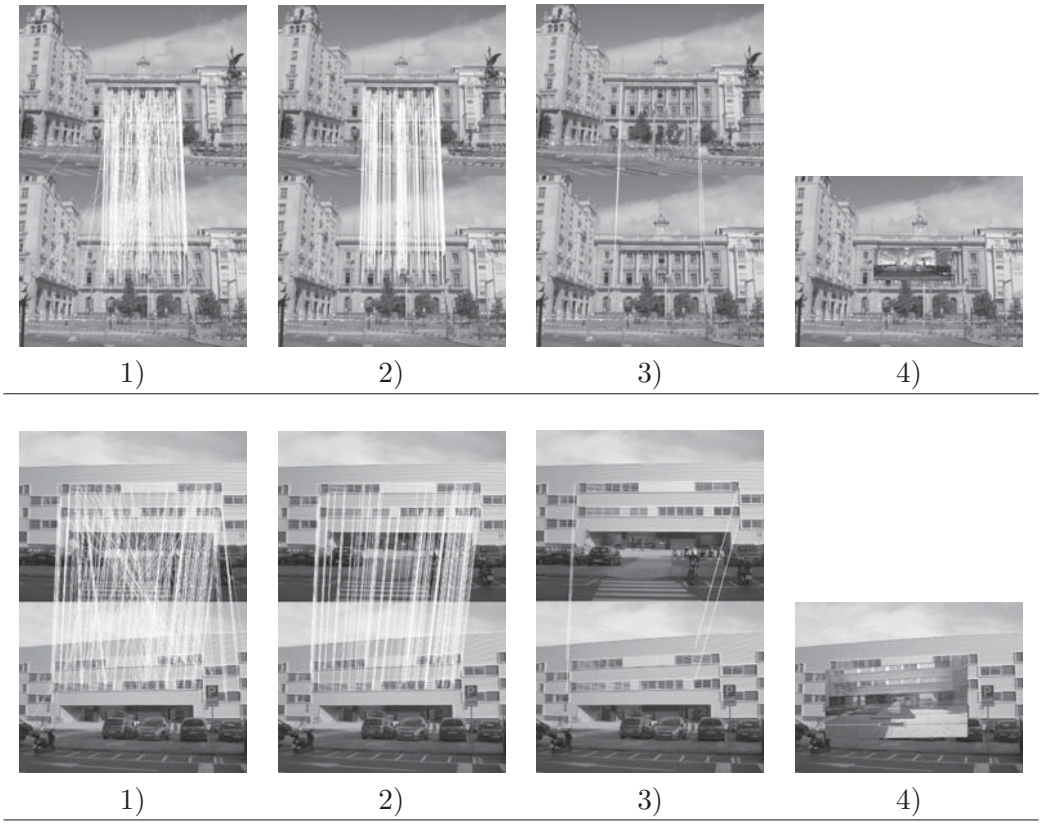


Figura E.3: Ejemplo. En la imagen 1) se observan todas las correspondencias dentro de la región donde se proyectará la imagen interior. En la imagen 2) tras calcular la matriz de homografía H y eliminar aquellas correspondencias no consistentes, sólo quedan las correctas. En la imagen 3) proyectamos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

Apéndice F

Diagrama de Flujo de Datos del Sistema de Reconocimiento

En este Anexo se detalla el Diagrama de Flujo de Datos de nuestro sistema de reconocimiento de imágenes.

F.1. Nivel 0

En la Figura F.1 se puede observar el nivel 0 de nuestro sistema de reconocimiento de imágenes. Expandiendo este nivel aparece el DFD de nivel 1 (Sección F.2) donde surgen los

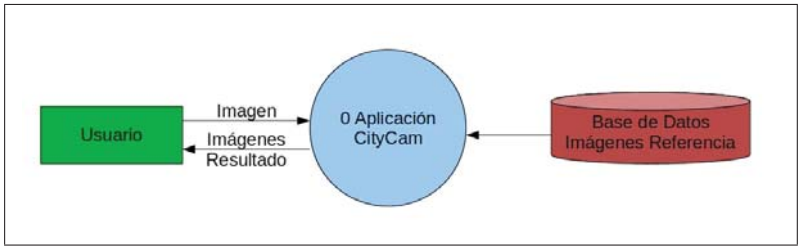


Figura F.1: Nivel 0 DFD del sistema de reconocimiento de imágenes.

diferentes subsistemas

F.2. Nivel 1

En la Figura F.2 se puede observar los diferentes subsistemas en los que se descompone nuestra aplicación.

Si expandimos cada uno de estos subsistemas, obtenemos los DFDs que se muestran en las secciones F.3, F.4 y F.5

F.3. Nivel 2, Subsistema 1

En la figura F.3 se observa el subsistema 1 expandido.

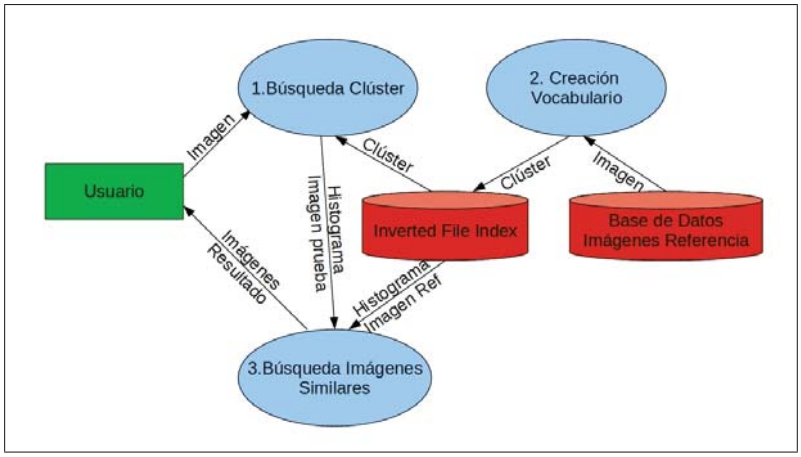


Figura F.2: Nivel 1 DFD.

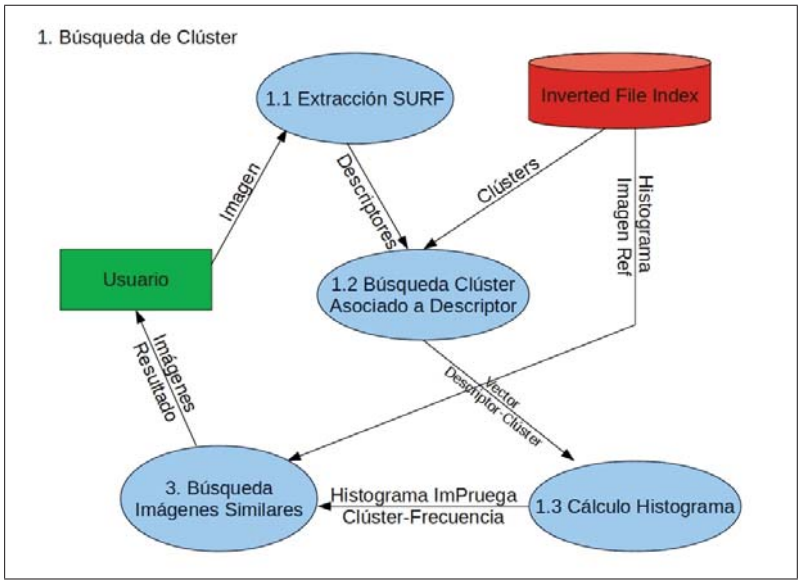


Figura F.3: Nivel 2 , subsistema 1 del DFD.

F.4. Nivel 2, Subsistema 2

En la figura F.4 se observa el subsistema 2 expandido.

F.5. Nivel 2, Subsistema 3

En la figura F.5 se observa el subsistema 3 expandido.

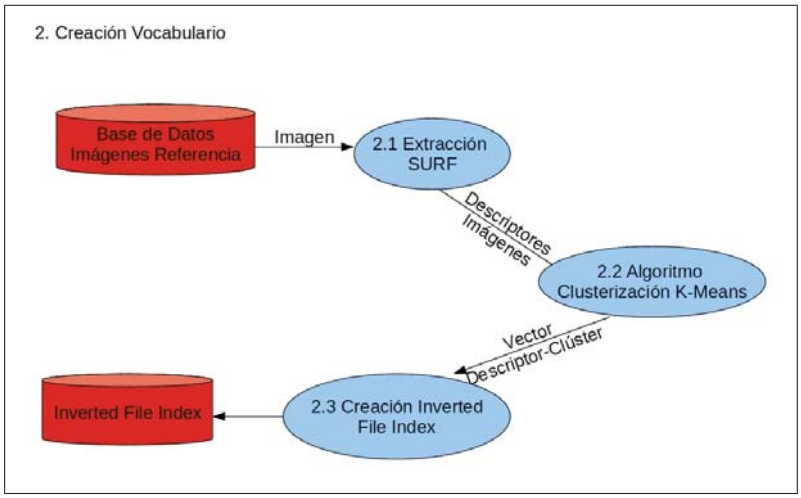


Figura F.4: Nivel 2 , subsistema 2 del DFD.

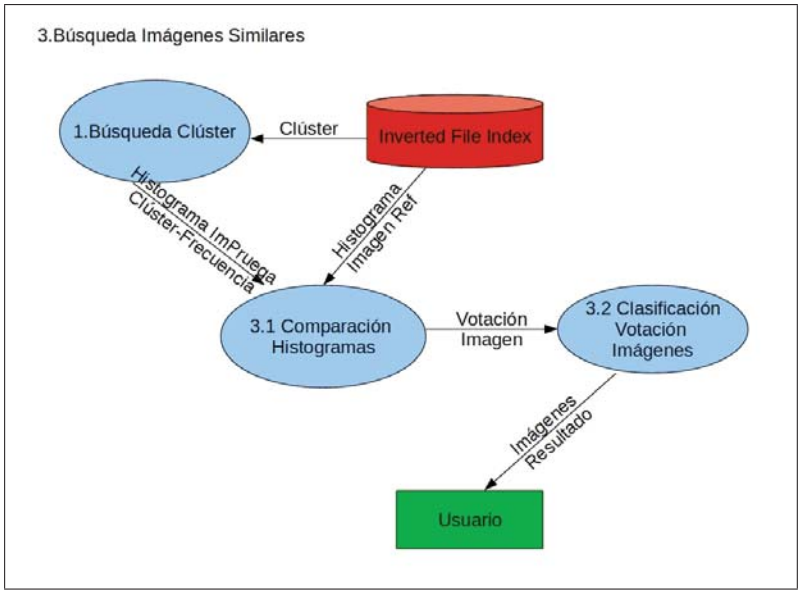


Figura F.5: Nivel 2 , subsistema 3 del DFD.

Apéndice G

Sistema de Posicionamiento Global (GPS)

El GPS (Global Positioning System) es un sistema global de navegación por satélite que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión hasta de centímetros, aunque lo habitual son unos pocos metros de precisión.

El GPS funciona mediante una red de 24 satélites en órbita sobre el globo, a 20.200 km, con trayectorias sincronizadas para cubrir toda la superficie de la Tierra. Cuando se desea determinar la posición, el receptor que se utiliza para ello localiza automáticamente como mínimo tres satélites de la red, de los que recibe unas señales indicando la identificación y la hora del reloj de cada uno de ellos. Con base en estas señales, el aparato sincroniza el reloj del GPS y calcula el tiempo que tardan en llegar las señales al equipo, y de tal modo mide la distancia al satélite mediante triangulación, la cual se basa en determinar la distancia de cada satélite respecto al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los tres satélites.

G.1. Utilidad de el GPS en este proyecto

En este proyecto el conocimiento de la posición de el usuario nos puede ser de gran utilidad. Conocida la posición de cada edificio de la base de datos y la de el usuario en el momento de la toma de la imagen, los resultados que se mostrarán a este serán más razonables, así como el tiempo de computo disminuirá, ya que habrá edificios que no será necesario su comparación.

Para que en la búsqueda solo se muestren resultados próximos al lugar donde se captura la fotografía se decidió que la distancia fuera de 1 Km a la redonda G.1. Consiguiendo que al usuario no se le muestren resultados no coherentes con su posición.

G.2. Cálculo de distancias entre coordenadas

Para el cálculo de la distancia entre coordenadas, se podría haber optado por el cálculo de la distancia euclídea como una transformación aproximada.

Pero el uso de está distancia tiene un problema, y es que como la superficie de la Tierra no es plana, por lo que se está cometiendo un error considerable. Aunque en nuestro caso no

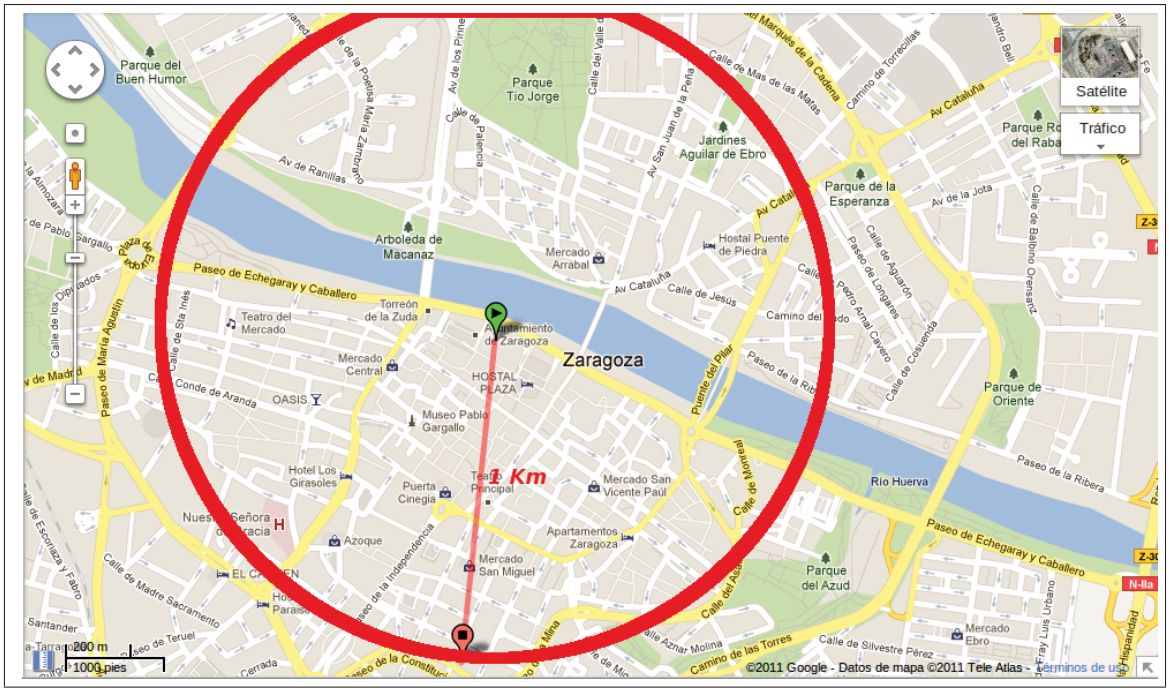


Figura G.1: Radio de resultados. Tomando como centro nuestra posición, sólo se mostrarán resultados de edificios en un radio de 1 km.

es necesaria una buena precisión, se decidió implementar un algoritmo que tuviera en cuenta la curvatura de la Tierra, cometiendo un menor error. Para ello se decidió utilizar la fórmula del semiverseno, que implementada en C es la que se muestra a continuación.

```
1  float calcularDistanciaCoordenadas(float longitud, float latitud, float longitudTest, float latitudTest){
2          /*------------------------------------------------------------------------*/
3          // Funcion que calcula la distancia en km entre 2 pares de coordenadas
4          /*------------------------------------------------------------------------*/
5          float pi = 3.14159265358979;
6          float dlon, dlat, a, distancia,longitudePto,latitudePto,latitude,longitude,distanciaReal;
7
8          longitudeA =longitud / 180 *pi;
9          latitudeA = latitud / 180 *pi;
10         longitudB = longitudTest / 180 *pi;
11         latitudB = latitudTest / 180 *pi;
12
13         dlon = longitudeA - longitudB;
14         dlat = latitudeA - latitudB;
15
16         a = pow(sin(dlat/2),2) + cos(latitudB) * cos(latitudeA) * pow(sin(dlon/2),2);
17         distancia = 2 * atan2(sqrt(a), sqrt(1-a));
18
19         distanciaReal = 6370.97327862 * distancia; /* 6378140 is the radius of the Earth in meters*/
20
21         return distanciaReal;
22
23 }
```

Apéndice H

Gestión del Proyecto

En este Anexo se detalla como se ha gestionado el tiempo para la realización de este proyecto. El periodo durante el cual se ha desarrollado este proyecto fin de carrera abarca desde principios del mes de Marzo de 2011 hasta Febrero de 2012. Durante los meses comprendidos entre Marzo y Junio la dedicación al proyecto fue a tiempo parcial, dado que tenía que compaginar el desarrollo del proyecto con las últimas asignaturas de la carrera. Como se puede observar en la Figura H.1, este tiempo se ha dividido principalmente en tres grandes bloques, los cuales representan el resultado final del proyecto. Se comenzó investigando proyectos anteriores relacionados con este, estudiando y desarrollando los métodos de representación completa. Una vez realizados experimentos y comprobado que su implantación en dispositivos móviles no era adecuada, se pasó a estudiar, implementar y probar los métodos de bolsa de palabras, los cuales si que se puede concluir que pueden ser implantados en un dispositivo móvil. Una vez desarrollada e implementada la parte de evaluación de similitud de imágenes en el dispositivo móvil, se pasó al estudio, desarrollo y pruebas de los métodos de realidad aumentada, con la que se concluyó este proyecto.

En cuanto a reuniones con los directores, se estableció un día a la semana en la cual yo comentaba cual había sido el trabajo realizado esa semana, de manera que yo pudiera recibir ayuda en las cuestiones que me iba ocasionando el desarrollo del proyecto y los tutores llevar un control del trabajo que estaba realizando. Durante los meses comprendidos entre Julio y Diciembre dado que Ana Cristina se encontraba en San Diego(Estados Unidos), se realizaron reuniones vía Internet.

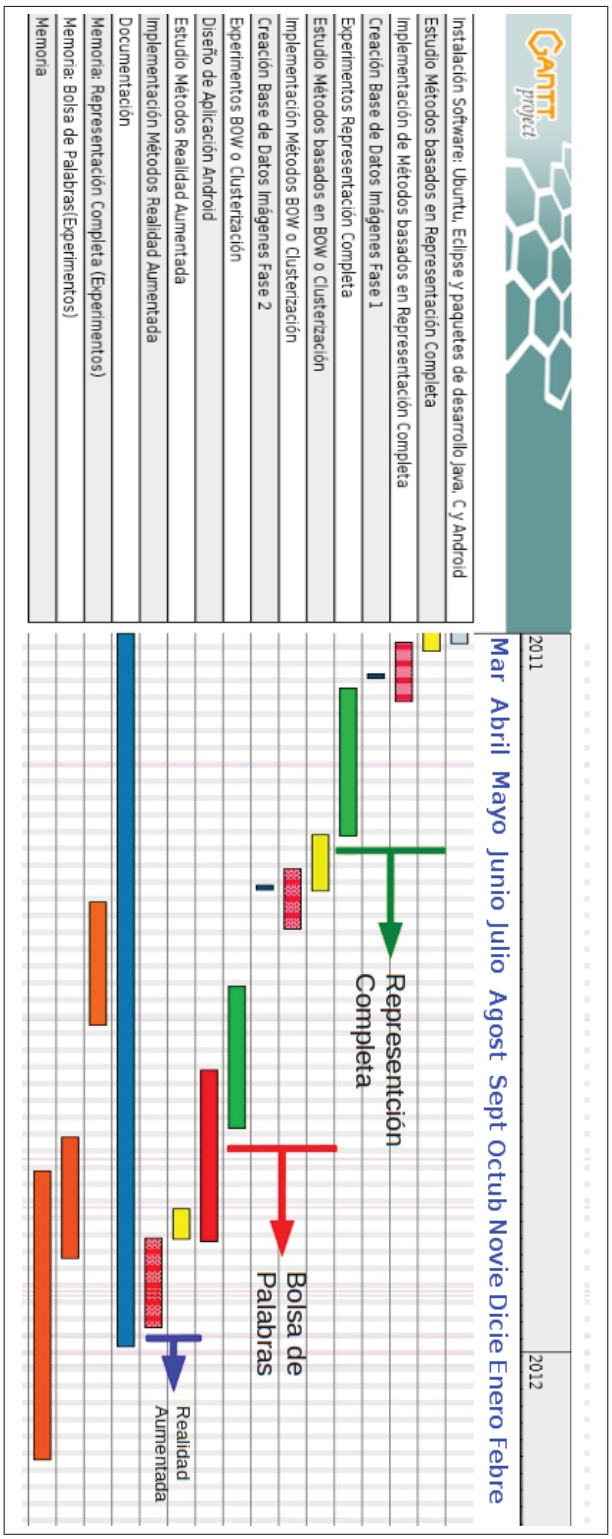


Figura H.1: Diagrama de Gantt de desarrollo del proyecto. Como se observa las tareas aparecen divididas en tres grandes grupos, correspondientes con los diferentes temas abordados en este proyecto.

Bibliografía

[1] *OpenCV Reference Manual. v2.1.* 2010.

[2] H. Bay, B. Fasel, and L. Van Gool. Interactive museum guide: Fast and robust recognition of museum objects. In *First Int. workshop on mobile vision*, 2006.

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conf. on Computer Vision*, 2006. http://www.vision.ee.ethz.ch/ surf/.

[4] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 1975.

[5] J.B. Burns, A.R. Hanson, and E.M. Riseman. Extracting straight lines. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(4):425–455, 1986.

[6] Mark Cummins and Paul Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research.*

[7] Sergei Vassilvitskii David Arthur. K-means++: The advantages of careful seeding. 2007.

[8] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. 1981.

[9] C. Harris and M. Stephens. A Combined Corner and Edge Detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[10] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[11] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE Int. Conf. on Computer Vision*, pages 1150–1157, 1999.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004. http://www.cs.ubc.ca/ lowe/keypoints/.

[13] Óscar Javier Calderón Nevot. Estudio, evaluación y desarrollo de métodos de visión por computador para reconocimiento de edificios, 2007. Proyecto Fin de Carrera.

[14] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.

[15] T. Quack, H. Bay, and L. Van Gool. Object recognition for the internet of things. In *Internet of Things 2008*, 2008.

[16] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, October 2003.