Capítulo 5. Evaluación y Conclusiones

En esta sección se llevaran a cabo pruebas de las dos opciones de consulta disponibles en el sistema que son consultas vía texto e imagen, con el propósito de observar de manera cuantitativa la eficiencia del sistema en cuanto a la precisión de los algoritmo, de procesamiento del texto, el algoritmo de detección de contorno y colores, y como se mostraron los resultados obtenidos.

Así mismo, se observara el tiempo de respuesta del sistema en las dos opciones de consulta con las que cuenta, con el motivo de identificar si el sistema tiene un tiempo de respuesta aceptable o si requiere de demasiado tiempo para regresar una respuesta al usuario. El propósito de estas pruebas es observar si el funcionamiento del sistema es eficiente, así como verificar si es viable la utilización del sistema en cuanto al tiempo de respuesta por parte de él.

5.1. Evaluación del Desempeño del Sistema

5.1.1 Pruebas de Algoritmo de Consulta Textual

A continuación se presentaran pruebas del modelo vectorial aplicado a las consultas vía texto, en las cuales se inserta(n) una(s) palabra(s) en una tabla temporal, mediante la cual se toma(n) el(los) termino(s) que compone(n) la consulta y se hace una comparación con las palabras que se encuentran almacenadas en la base de datos. Para el cálculo de la similitud se utiliza el producto.

El número de resultados de este algoritmo puede variar, desde no tener resultados hasta tener un número específico de resultados, esto debido a las palabras que componente la consulta y a las que se encuentran almacenados en la base de datos, ya que para este algoritmo puede haber tres casos importantes, que son los siguientes:

1. La estructura que se está consultando no se encuentra almacenada en la base de datos, por lo que no habría resultado de la consulta

2. La estructura que se consulta se encuentra almacenada en la base de datos y los términos que componen la consulta son muy específicos de la estructura (ej. Parte del nombre), por lo que el resultado sería solamente una estructura.
3. La estructura que se consulta se encuentra almacenada en la base de datos y los términos que componen la consulta no son muy específicos de la estructura (lugar en el que se encuentra), por lo que podría haber más de un resultado.

Para medir la eficiencia del algoritmo utilizado para las consultas vía textual, se utilizó la medida de precisión y cobertura propuesta por Stefan Büttcher, Charles L. A. Clarke y Gordon V. Cormack, las cuales están dadas por las siguientes formulas [20]:

$$precisión = \frac{|Res \cap Rel|}{|Res|} \tag{5.1}$$

$$cobertura = \frac{|Res \cap Rel|}{|Rel|} \tag{5.2}$$

Dónde:

- Res = Conjunto de Documentos Recuperados.
- Rel = Conjunto de Documentos Relevantes Recuperados

En la Tabla 5.1 se presenta una tabla con las pruebas realizadas al modelo vectorial, en la cual se podrán observar los tres casos que se presentaron anteriormente:

Prueba	Términos	Resultado 1	Resultado 2	Resultado 3	Precisión	Cobertura
1	"Eiffel"	36.5707	----------	----------	100.00%	100.00%
2	"Paris"	28.6677	28.6677	----------	100.00%	100.00%
3	"Estados Unidos"	48.9804	48.9804	48.9804	100.00%	100.00%
4	"Independencia"	36.5707	----------	----------	100.00%	100.00%
5	"Torre"	28.6677	28.6677	----------	100.00%	100.00%
6	"Canada"	----------	----------	----------	0.00%	0.00%
7	"Eiffel Paris"	65.2384	28.6677	----------	50.00%	100.00%
8	"D.C"	36.5707	----------	----------	100.00%	100.00%
9	"Estatua Nueva York"	93.9061	57.3354	----------	50.00%	100.00%
10	"Casa Estados"	61.0609	24.4902	24.4902	33.00%	100.00%
Total					73.30%	90.00%

Tabla 5.1. Pruebas de Modelo Vectorial

Donde Resultado 1, Resultado 2 y Resultado 3, son aquellas imágenes en las cuales tienen el termino de busqueda entre su información. Es por este motivo, por el que hay consultas en las cuales hay mas de un resultado.

Como se puede observar en la tabla anterior, la precisión del modelo vectorial se encuentra en un 73%, esto debido a que el algoritmo busca las palabras que componente la consulta y las compara con las que se encuentran en la base de datos; al hacer esto, muchas estructuras comparten características como el lugar en el que se encuentran (ej. Paris) o el tipo de estructura que es (ej. Torre), es por esto que al momento de hacer una consulta, se recuperan estructuras que no eran las deseadas, sin embargo, en las pruebas que se realizaron, el primer elemento que compone el resultado ha sido relevante de acuerdo a la consulta que se introdujo en el sistema.

La cobertura con la que cuenta el sistema es del 90%, esto debido a que, entre el conjunto de elementos que componen el resultado, la mayoría de los casos se recuperó la estructura más relevante de acuerdo a la consulta, excepto en los casos en que la estructura no se encuentra almacenada en la base de datos.

Con el propósito de observar si el tiempo de respuesta del sistema es el más óptimo, se realizaron pruebas midiendo el tiempo que tardaba el sistema en responder a una consulta dada por

el usuario. En estas pruebas se medirá el tiempo total que requiere hacer la consulta a la base de datos y regresar el resultado al usuario, con el propósito de verificar que no requiera una gran cantidad de tiempo para esperar una respuesta por parte del sistema, así como verificar que el sistema sea viable.

En la Tabla 5.2 se presentan los resultados de los tiempos de respuesta del modelo vectorial, estos resultados se midieron desde el momento en que el usuario oprime el botón de "Buscar" hasta el momento en que se regresa el resultado de la consulta.

Prueba	Términos	Envío de Consulta (ms)	Consulta a BD (ms)	Envío de Respuesta (ms)	Total (ms)
1	"Eiffel"	700	0.4	1399.6	2100
2	"Paris"	400	0.5	1799.5	2200
3	"Estados Unidos"	400	0.4	1999.6	2400
4	"Independencia"	700	0.4	1599.6	2300
5	"Torre"	400	0.4	1599.6	2000
6	"Canada"	500	0.4	1299.6	1800
7	"Eiffel Paris"	600	0.4	1499.6	2100
8	"D.C"	400	0.4	1699.6	2100
9	"Estatua Nueva York"	1100	0.5	2199.5	3300
10	"Casa Estados"	600	0.5	1899.5	2500
Promedio		580	0.43	1699.57	2280

Tabla 5.2. Resultados de los Tiempos de Respuesta del Modelo Vectorial

Como se puede observar en la tabla anterior, el tiempo de envío de los resultados por parte del sistema es de 1699.57 ms (1.7 seg), el cual es un buen tiempo de respuesta por parte del servidor, debido a que se debe de enviar la información relevante acerca de la consulta, además de enviar la imagen de la estructura que se está consultando.

El tiempo promedio de respuesta del modelo vectorial es de 2280 ms (2.28 seg), lo cual es un buen tiempo de respuesta por parte del sistema, ya que la gran cantidad de información que se debe de enviar tanto al servidor (consulta) como al dispositivo (información e imagen) puede provocar que la transmisión de la información se realice de manera lenta. Estas pruebas se

realizaron en condiciones ideales, en las cuales el tráfico de la red no se encuentra congestionada por otros usuarios.

5.1.2. Pruebas del Algoritmo de Detección de Contorno

A continuación se presentaran las pruebas que se le realizaron al algoritmo de detección del contorno de las imágenes. Estas pruebas serán realizadas con la funcionalidad que cuenta el sistema para enviar una imagen tomada desde la librería de imágenes del dispositivo para enviarla al servidor. El motivo por el cual se decidió ocupar esta funcionalidad es debido a que la mayoría de las imágenes que se encuentran almacenadas en la base de datos son de estructuras arquitectónicas que se encuentran fuera del país, por lo que sería imposible ocupar la funcionalidad de enviar una imagen tomada desde la cámara del dispositivo.

El procedimiento que se realizara para conocer la eficiencia del algoritmo se basa en la detección del número de puntos con los que cuenta la imagen enviada por el usuario y compararlas con el número de puntos similares entre esta imagen y las imágenes que se encuentran en la base de datos. Una vez realizada la comparación, se eligen las cuatro imágenes que tengan un porcentaje alto de similitud con las imágenes que se encuentran almacenadas en la base de datos.

Las pruebas se realizaron con imágenes que se encuentran almacenadas en la base de datos. Los resultados se muestran en la Tabla 5.3 y Tabla 5.4:

Prueba	Total de Pixeles	Pixeles Similares Resultado 1	Pixeles No Similares Resultado 1	% Similares Resultado 1	% No Similares Resultado 1	Pixeles Similares Resultado 2	Pixeles No Similares Resultado 2	% Similares Resultado 2	% No Similares Resultado 2
1	328125	325692	2433	99.26%	0.74%	310485	17640	94.62%	5.38%
2	328125	323128	4997	98.48%	1.52%	301288	26837	91.82%	8.18%
3	328125	322352	5773	98.24%	1.76%	288112	40013	87.81%	12.19%
4	328125	323081	5044	98.46%	1.54%	301058	27067	91.75%	8.25%
5	328125	325003	3122	99.05%	0.95%	304426	23699	92.78%	7.22%
6	328125	311512	16613	94.94%	5.06%	286683	41442	87.37%	12.63%
7	328125	321515	6610	97.99%	2.01%	293109	35016	89.33%	10.67%
8	328125	324342	3783	98.85%	1.15%	309819	18306	94.42%	5.58%
9	328125	325779	2346	99.29%	0.71%	311222	16903	94.85%	5.15%
10	328125	323872	4253	98.70%	1.30%	301883	26242	92.00%	8.00%
Promedio	328125	322627.6	5497.4	98.32%	1.68%	300808.5	27316.5	91.67%	8.33%

Tabla 5.3. Pruebas de Algoritmo de Detección de Contorno (1)

Prueba	Pixeles Similares Resultado 3	Pixeles No Similares Resultado 3	% Similares Resultado 3	% No Similares Resultado 3	Pixeles Similares Resultado 4	Pixeles No Similares Resultado 4	% Similares Resultado 4	% No Similares Resultado 4
1	309057	19068	94.19%	5.81%	308332	19793	93.97%	6.03%
2	298343	29782	90.92%	9.08%	296678	31447	90.42%	9.58%
3	285038	43087	86.87%	13.13%	283938	44187	86.53%	13.47%
4	298612	29513	91.01%	8.99%	296271	31854	90.29%	9.71%
5	301326	26799	91.83%	8.17%	299591	28534	91.30%	8.70%
6	283929	44196	86.53%	13.47%	282812	45313	86.19%	13.81%
7	289966	38159	88.37%	11.63%	288842	39283	88.03%	11.97%
8	306595	21530	93.44%	6.56%	304493	23632	92.80%	7.20%
9	306542	21583	93.42%	6.58%	305153	22972	93.00%	7.00%
10	298631	29494	91.01%	8.99%	297821	30304	90.76%	9.24%
Promedio	297803.9	30321.1	90.76%	9.24%	296393.1	31731.9	90.33%	9.67%

Tabla 5.4. Pruebas de Algoritmo de Detección de Contorno (2)

Como se puede observar en la Tabla 5.3 y Tabla 5.4, el porcentaje de efectividad del algoritmo de detección de contorno se encuentra entre un 98% y 90%. Los resultados presentados en las tablas anteriores son en base a 10 pruebas que se realizaron con imágenes que se encuentran almacenadas en la base de datos.

Se decidió presentar el porcentaje de los primeros cuatro resultados debido a la cantidad de imágenes que se encuentran almacenadas en la base de datos. Otro motivo por el cual se presentaron los resultados de los primeros cuatro resultados es debido a que son aquellos que se toman para ser utilizados en el último paso del algoritmo de procesamiento de imágenes, en donde se realiza una comparación entre estos resultados con los resultados tomados del algoritmo de detección de colores.

Una prueba adicional que se realizo fue la medición de los tiempos que requiere el sistema para realizar el algoritmo de detección de los contornos. Para esto se realizaron las pruebas con imágenes que se encuentran almacenadas en la base de datos y se le hicieron pocas modificaciones al sistema para que pudiese mostrarnos los tiempos requeridos de cada paso del algoritmo. Estas pruebas fueron realizadas con diez imágenes que se encuentran almacenadas en la base de datos y, con esto, se obtuvo el promedio del tiempo que requiere el sistema para realizar el algoritmo.

Los pasos que se midieron en las pruebas fueron la recuperación de la consulta del usuario, así como de las imágenes que se encuentran almacenadas en la base de datos, el tiempo necesario para detectar el contorno de la imagen enviada por el usuario, el tiempo que requiere el algoritmo para hacer la comparación entre los contornos de las imágenes y el tiempo que le tomo encontrar las cuatro imágenes con mayor similitud.

Los resultados de los tiempos que requiere el algoritmo de detección del contorno y comparación de la imagen se presentan en la Tabla 5.5:

Prueba	Cargar Consulta (ms)	Cargar Imágenes en BD (ms)	Detectar Contorno Consulta (ms)	Comparación (ms)	Elección de Resultados (ms)	Total (ms)
1	9.44	20.12	23.24	47.85	1.07	101.72
2	11.68	23.51	21.38	59.36	0.81	116.74
3	26.62	22.29	90.59	51.89	0.63	192.02
4	63.21	43.05	16.89	48.16	0.61	171.92
5	32.18	22.22	25.76	51.49	0.68	132.33
6	9.34	20.85	24.78	57.56	0.76	113.29
7	34.85	994.7	19.29	45.2	0.69	1094.73
8	12.35	21.04	20.77	44.27	0.65	99.08
9	28.92	23.6	17.43	44.96	0.67	115.58
10	11.85	27.56	18.77	59.47	0.71	118.36
Promedio	24.044	121.894	27.89	51.021	0.728	225.577

Tabla 5.5. Pruebas de Tiempo de Algoritmo de Detección de Contorno

Como se puede observar en la Tabla 5.5, el promedio de tiempo que requiere el algoritmo de detección de contorno es de 225.577 ms (0.23 seg), lo cual es un muy buen tiempo considerando que se tiene que hacer la detección del contorno de la imagen enviada por el usuario como consulta, además de hacer la comparación pixel-a-pixel de las imágenes almacenadas en la base de datos con la imagen enviada por el usuario y elegir las cuatro imágenes que tengan un alto valor de similitud con la imagen de consulta.

Un punto que se pudo observar durante las pruebas fue que, los valores que componen los resultados de cada paso se hallan dentro del mismo intervalo, lo que quiere decir que el algoritmo, sea la primera consulta o no, esta se comportara de la misma manera y requerirá aproximadamente el mismo tiempo que una consulta realizada después de un periodo de prueba.

5.1.3. Pruebas del Algoritmo de Detección de Colores

A continuación se presentan las pruebas del algoritmo de detección de colores de las imágenes. Estas pruebas, al igual que las realizadas para el algoritmo de detección del contorno, se realizaron

utilizando la funcionalidad de enviar una imagen tomada desde la galería de fotos que se encuentran almacenadas en el dispositivo por el motivo previamente mencionado.

El procedimiento que se realizara para conocer la eficiencia del algoritmo de detección de colores se basa en el cálculo del histograma de colores de la imagen enviada por el usuario y compararla con los histogramas de las imágenes almacenadas en la base de datos. Las facilidades que nos presenta OpenCV es que la comparación entre los histogramas se realiza de manera rápida y sencilla, y los resultados que regresa el sistema es el porcentaje de similitud de colores que cuentan las imágenes entre sí.

Las pruebas se realizaron con imágenes que se encuentran almacenadas en la base de datos. Los resultados se presentan en la Tabla 5.6:

Prueba	% Similitud Resultado 1	% Similitud Resultado 2	% Similitud Resultado 3	% Similitud Resultado 4
1	100.00%	30.73%	24.31%	18.73%
2	100.00%	39.05%	27.77%	10.34%
3	100.00%	30.54%	29.82%	13.94%
4	100.00%	27.66%	24.31%	18.65%
5	100.00%	25.40%	19.68%	18.73%
6	100.00%	53.91%	32.57%	20.50%
7	100.00%	38.60%	32.20%	30.54%
8	100.00%	25.40%	13.41%	12.29%
9	100.00%	53.91%	51.36%	33.46%
10	100.00%	51.36%	32.57%	16.25%
Promedio	100.00%	37.66%	28.80%	19.34%

Tabla 5.6. Pruebas de Algoritmo de Detección de Colores

Como se puede observar en la Tabla 5.6, el porcentaje de efectividad se encuentra entre el 100% y el 19%. Estos resultados son debido a que cada una de las imágenes cuenta con un vector particular de colores, por lo que las similitudes entre imágenes varían significativamente.

Las ventajas de este algoritmo de detección de colores es que la primera imagen resultante tiene un alto valor de similitud, por lo que se considera que es una imagen considerablemente

similar con la imagen enviada por el usuario, a diferencia que el algoritmo de detección de contorno, en el cual el valor de similitud de las imágenes se encuentran cercanos ya que es sencillo que haya puntos parecidos entre las formas de las imágenes, esto debido a que las estructuras comúnmente cuentan con formas muy similares como pilares (rectángulos), techos (varía entre cuadrados o triángulos), etc.

Como prueba adicional, se midieron los tiempos que requiere el sistema para realizar el algoritmo de detección de colores de las imágenes. Para estas pruebas se utilizaron imágenes que se encuentran almacenadas en la base de datos. Estas pruebas se realizaron con diez imágenes tomadas de la base de datos y, con esto, se obtuvo el promedio del tiempo que requiere el sistema para llevar a cabo el algoritmo.

Los pasos que se midieron en las pruebas fue el tiempo que requiere el sistema para recuperar la imagen enviada por el usuario, el tiempo en recuperar las imágenes que se encuentran en la base de datos, la duración del sistema para calcular el histograma de colores de la imagen enviada como consulta, el tiempo que requiere el sistema para comparar los histogramas de las imágenes de la base de datos con la imagen de consulta y el periodo de tiempo que se llevar para obtener las cuatro imágenes con mayor similitud.

Los resultados de los tiempos que requiere el sistema para el cálculo de los histogramas y su comparación se presentan en la Tabla 5.7:

Prueba	Cargar Consulta (ms)	Cargar Imágenes en BD (ms)	Calcular Histograma Consulta (ms)	Comparar Histogramas (ms)	Elegir Resultados (ms)	Total (ms)
1	43.22	208.12	17.87	384.81	0.93	654.95
2	61.75	41	19.69	274.65	0.89	397.98
3	11.16	34.03	17.22	272.34	0.86	335.61
4	92.32	28.74	16.1	270.14	0.76	408.06
5	13.42	217.16	21.06	258.98	0.77	511.39
6	15.27	27.75	24.75	249.55	0.8	318.12
7	13.55	27.54	18.95	262.24	0.72	323
8	13.35	51.56	19.71	265.62	0.81	351.05
9	9.91	90.39	12.33	256.28	0.81	369.72
10	72.58	27.85	14	263.44	0.79	378.66
Promedio	34.653	75.414	18.168	275.805	0.814	404.854

Tabla 5.7. Pruebas de Tiempo de Algoritmo de Detección de Colores

Como se puede observar en la Tabla 5.7, el tiempo promedio que requiere el sistema para llevar a cabo el algoritmo de detección de colores de 404.854 ms (0.4 seg), lo cual es un buen tiempo debido a que, para este algoritmo no se logró tener imágenes preprocesadas, por lo que el algoritmo debe obtener los histogramas de todas las imágenes almacenadas en la base de datos y hacer la comparación con el histograma de la imagen enviada por el usuario; esto se puede observar en los tiempos que requiere el sistema para comparar los histogramas, cuyo promedio es de 275.805 ms (0.28 seg), los cuales, en comparación con los tiempos de los pasos anteriores, requiere mucho más tiempo de procesamiento, por lo cual aumenta el tiempo requerido por el sistema para finalizar el algoritmo.

5.1.4. Comparación con Otros Sistemas

Con el propósito de comprobar la eficiencia del sistema en comparación con otros sistemas que se analizaron en el capítulo 1 se realizó una comparación entre los porcentajes presentados por otros sistemas con el propuesto en este trabajo.

La primera comparación se realizó con un sistema que se desarrolló para dispositivos celulares de la familia Samsung Galaxy, cuyo propósito era la detección de kanjis japoneses; los datos que se muestran en la Tabla 5.8 fueron obtenidos del trabajo analizado en el capítulo 1 [2].

	Hoja	Total caracteres	Caracteres Identificados	Caracteres NO Identificados	% Identificado	% no identificado
W595	1	34	32	2	94.12%	5.88%
	2	13	12	1	92%	7.69%
	3	19	17	2	89%	10.53%
	4	31	28	3	90%	9.68%
	5	32	27	5	84%	15.63%
	6	23	19	4	83%	17.39%
	7	56	52	4	93%	7.14%
	8	36	31	5	86%	13.89%
	9	34	31	3	91%	8.82%
	10	24	21	3	88%	12.50%
Suma		302	270	32	89%	10.60%

Tabla 5.8. Datos de Sistema de Detección de Kanjis Japoneses

Haciendo una comparación entre la Tabla 5.3, Tabla 5.4 y Tabla 5.8, se puede observar que el algoritmo propuesto en este trabajo presente un mayor porcentaje de similitud a pesar de solo considerar los primeros cuatro resultados. Cabe resaltar que los resultados encontrados en la Tabla 5.8 están orientados a la detección correcta de kanjis japoneses, a diferencia de los datos de la Tabla 5.3 y 5.4 la cual está orientada a la similitud entre un conjunto de imágenes con una imagen especifica.

A continuación, se realizó una prueba de los tiempos que se tomaron dos algoritmos enfocados al cálculo de histogramas de colores y realizar una búsqueda en una base de datos. Esta comparación se realizó con un sistema para la búsqueda de imágenes en bibliotecas digitales. Los datos que se muestran en la Tabla 5.9 fueron obtenidos del trabajo realizado en el capítulo 1 [13].

Consulta	Tiempo de Busqueda (seg)	Tiempo de Desplegado (seg)	Tiempo Total (seg)
1	43	18.6	61.6
2	10.1	70.3	80.4
3	22	86.3	108.3
4	24.6	112	136.6
5	4.8	0.9	5.7
6	13	179	192
7	14.7	196	210.7
8	15.2	221	236.2
9	16.8	256.8	16.8
10	20.4	19.3	39.7
11	11.6	51.9	63.5
12	9.2	79	88.2
13	9.3	8.3	17.6
14	7.2	5.6	12.8
15	25.7	21.4	21.4
16	10.8	50.9	61.7
17	6.5	72.2	78.7
18	11	13.3	24.3
19	14.9	46.2	61.1
20	11.8	19.7	31.5

Tabla 5.9. Datos de Sistema de Recuperación de Imágenes en Bibliotecas Digitales

Para esta comparación solamente se tomaron en cuenta los datos de Tiempo de Búsqueda, debido a que es el mismo parámetro que se está comparando. Como se puede observar en la Tabla 5.5 y la Tabla 5.9, el tiempo de búsqueda que presenta el algoritmo implementado en este trabajo es menor al que se presenta en el sistema de búsqueda de imágenes en bibliotecas digitales.

5.2. Conclusiones y Trabajo a Futuro

5.2.1. Resultados

A lo largo del trabajo, se explicó el desarrollo y la implementación de un sistema móvil de recuperación de información visual en base a características de bajo nivel, como el contorno y los colores, con el propósito de recuperar información relevante acerca de estructuras arquitectónicas importantes a lo largo del mundo. Este sistema fue desarrollado en un dispositivo móvil,

específicamente hablando del iPad, debido a que es uno de los dispositivos más utilizados en la actualidad.

El sistema se dividió en cuatro módulos, los cuales interactúan para regresar al usuario los resultados que el espera, en base a una imagen o texto enviado por el usuario. Los módulos que interactúan son los siguientes:

1. Cliente: Este se desarrolló en el dispositivo móvil, el cual es el encargado de mostrar una interfaz lo suficientemente intuitiva, con el propósito de que la interacción entre el usuario con el dispositivo se lleve a cabo de una manera fácil y sencilla. También es el encargado de mostrar los resultados ante el usuario
2. Servidor: Este se desarrolló en Apache, el cual es el encargado de escuchar las consultas por parte del cliente, específicamente es el encargado de almacenar la imagen enviada por el cliente para ser procesada y, también, se encarga de realizar la consulta textual y enviar los resultados al cliente
3. Procesador: Este fue desarrollado en XCode, el cual es el encargado de realizar todo el algoritmo de procesamiento de imágenes, haciendo las comparaciones pertinentes en la base de datos.
4. Base de Datos: Esta fue desarrollada en MySQL, la cual tiene almacenada toda la información relevante a las imágenes, además, de contener los path correspondientes a cada imagen almacenada.

Específicamente, este trabajo tiene como resultado el desarrollo de las siguientes aplicaciones:

1. Una aplicación móvil con interfaz gráfica en iPad
2. Base de Datos con imágenes almacenadas, así como información relevante acerca de ellas
3. Procesador de imágenes, el cual fue implementado en XCode

4. Implementación de algoritmos de procesamiento de imágenes tomando en cuenta características de bajo nivel en un dispositivo móvil

5.2.2. Aportaciones

Entre las aportaciones que se pueden puntualizar en este trabajo, es la implementación de algoritmos de procesamiento de imágenes que toman en cuenta características de bajo nivel en un dispositivo móvil que actualmente se está utilizando con mucha mayor frecuencia que en tiempos pasados. Este es un logro significativo porque no se han realizado muchas aplicaciones dedicadas al procesamiento de imágenes y a la recuperación de información relevante en base a sus características, comúnmente estas aplicaciones se han implementado en web.

Debido a la diferencia entre la sintaxis y el lenguaje de programación que presenta Objective-C, este trabajo presenta parte del código que se desarrolló en el transcurso de la aplicación, por lo que puede tomarse como guía para futuros trabajos que estén dedicados al procesamiento y recuperación de imágenes en base a sus características de alto o bajo nivel. Al hacer uso de este trabajo como referencia teórica para futuros trabajos relacionados al área, se podrá tener un marco de referencia de las consideraciones que se deben realizar al momento de hacer el diseño de la aplicación, así como los componentes que podrían ser necesarios al momento de desarrollar la aplicación, así como la viabilidad que tiene el desarrollar una aplicación de este tipo.

Entre los beneficios que trae consigo la implementación de este sistema seria el ahorro económico que tendrían las empresas en el ámbito turístico, ya que esta aplicación podría ocuparse como base para desarrollar una aplicación empresarial en la cual se pueda recuperar información relacionada con destinos turísticos de la zona, lo que traería consigo un ahorro en el folletaje de las zonas que se visitan en un determinado tour por la zona. Además de que gracias a este ahorro se estaría ayudando al cuidado del medio ambiente al dejar de utilizar papel para la impresión del folletaje de los destinos turísticos que se visitaran, que comúnmente al terminar el tour por la zona

estos se convierten en basura o pierde por completo el propósito con el que se realizó la impresión de dichos folletos.

Al incursionar en el área de procesamiento de imágenes en dispositivos móviles, trae consigo un impacto científico ya que es un área en la que muy pocas personas han incursionado y se tienen muy pocas investigaciones en las que se realicen aplicaciones móviles para el reconocimiento de características de alto o bajo nivel, tampoco se conocen muchas aplicaciones que hayan realizado en esta área. Además, este desarrollo puede motivar a otras personas a realizar desarrollo e investigación en el área de procesamiento de imágenes, específicamente enfocada al desarrollo de aplicaciones móviles que lleven a cabo algoritmo de detección de las características de las imágenes o aplicaciones de recuperación de imágenes basadas en ontologías.

Otra aplicación que se le puede dar a este desarrollo es para la localización de una estructura arquitectónica, en la cual el usuario desconoce el nombre de dicha estructura y mediante una imagen, se realiza el procesamiento de la imagen con el propósito de encontrar la imagen con una mayor similitud con la del usuario y en base a esto enviar la localización de dicha estructura en base a la localización actual del usuario. Esta aplicación sería similar a google map, solo que esta contaría con la posibilidad de introducir el nombre de la estructura o una imagen para realizar el procesamiento y marcar la ruta de llegada a dicha estructura en base a la ubicación del usuario.

5.2.3. Trabajo a Futuro

Al realizar las pruebas con imágenes ajenas a la base de datos, se pudo observar que en el algoritmo de detección del contorno de las imágenes, los resultados eran los mismo independientemente de la imagen de entrada, esto puede deberse a la calidad de la imagen enviada por el usuario o también se debe al valor de los umbrales que se le asignó al sistema que tomara en cuenta al momento de hacer la detección del contorno. Para esto se podría implementar un algoritmo mediante el cual se pueda detectar los umbrales a utilizar de manera dinámica. Esto mejoraría el desempeño del sistema ya que permitiría discriminar de mejor manera objetos ajenos a la estructura de la imagen, como por

ejemplo los automóviles que se pudiesen encontrar en la imagen, la vegetación que pudiese contar la estructura.

Un aspecto que se pudo observar al momento de realizar las pruebas de envío, procesamiento y obtención de los resultados fue que al momento de hacer las pruebas, a pesar de ser imágenes que se encontraban almacenadas en la base de datos (las cuales, teóricamente, los resultados tendrían que ser los ideales) los resultados eran erróneos, pero al momento de llevar a cabo el procesamiento de imágenes en el servidor sin el envío de los resultados, estos eran correctos. Al continuar con las pruebas, se le introdujo al sistema que realizara dos rondas de procesamiento, dejando de lado los primeros resultados y así se lograba obtener los resultados óptimos. Para conocer el motivo por el cual ocurre esto, se planea hacer investigación acerca de este tema, con el propósito de encontrar el motivo por el cual se esté produciendo estos detalles en el sistema.

Para lograr que el algoritmo de detección de colores obtenga los mejores resultados, se planteara la posibilidad de implementar un algoritmo de segmentación de las imágenes con el propósito de detectar el área más relevante de las imágenes, en este caso sería la estructura, esto con el propósito de que el sistema no deba detectar formas y colores de objetos ajenos a la estructura como los son árboles, autos, etc. También, se planteara implementar un algoritmo de discriminación de colores, en el cual el sistema no tome en cuenta colores de objetos ajenos a la estructuras como los antes mencionados. Estas mejoras permitirán que el sistema no deba comparar objetos que no agregan una valor a la imagen y permitirá que los resultados sean mucho más precisos.

Debido a la forma en la que se desarrolló el procesador, se requiere que este siempre este corriendo en el servidor, además de estar llevando a cabo ciclos constantemente en espera de que haya una imagen en el servidor para procesar. Para resolver este problema, se realizara una investigación en busca de encontrar la manera en que el procesador se pueda desarrollar como un ejecutable, con el propósito que, mediante un script, se pueda abrir el procesador al momento en

que el servidor recibe una imagen, se lleve a cabo todo el algoritmo de procesamiento de imágenes y, al finalizar el algoritmo y el envío de los resultados, el procesador se cierre de manera automática. Esto permitiría el ahorro de recursos dentro del servidor y evitaría que el servidor se pueda caer debido al alto consumo de recursos que generaría el procesador al momento de estar realizando ciclos repetitivos.

Debido a que las pruebas se realizaron con un número pequeño de imágenes en el servidor, los resultados de los tiempos pudiesen variar al momento de agregar más imágenes el servidor, lo que podría provocar que el sistema ya no sea viable y eficiente. Para observar esto, se continuara realizando pruebas en las cuales se irá aumentando el número de imágenes almacenadas en el servidor, con el propósito de observar si el sistema sigue teniendo tiempos de respuesta considerables.

Debido a que el desarrollo de esta aplicación se realizó con propósitos del presente trabajo, el servidor fue implementado en una computadora portátil y al momento de realizar la comunicación del dispositivo con el servidor, se requería conocer el IP de la computadora portátil, lo cual generaba que, al momento de que el IP cambiara, se tuviera que cambiar la IP del código de la aplicación. Para esto se implementó una solución temporal, en la cual se recurrió al software gratuito de www.noip.com, la cual nos proporcionaba un URL ficticio mediante el cual se podía hacer la comunicación con el servidor sin importar el IP de la computadora portátil; la desventaja de la utilización de este software fue que en todo momento debe estar corriendo el software para detectar los posibles cambios de IP y para que el URL ficticio funcionara de la manera en que se utilizó en este trabajo; otra desventaja del uso de este software fue que era necesario tener abierto el puerto 80 del lado del servidor, ya que si no se contaba con este puerto abierto, no se podría tener acceso a los scripts que se desarrollaron del lado del servidor, los cuales son indispensables para recibir las consultas por parte del usuario y comunicarse con la base de datos. Para solucionar esto,

se buscaran alternativas viables para realizar la comunicación sin necesidad de tener un software corriendo del lado del servidor, lo cual podría consumir recursos del mismo.