Capítulo 2. Marco teórico

2.1 ¿Qué es CBIR?

Para poder hablar de búsqueda de imágenes por contenido, primero hay que introducir un concepto más amplio: recuperación de información visual. Se trata de un tema de investigación reciente en el área de tecnologías de la información y es una extensión de la recuperación de información tradicional que incluye material multimedial. Su objetivo es recuperar imágenes o secuencias de imágenes (video) que son pertinentes a una consulta [Del Bimbo 1999]. La búsqueda de imágenes por contenido es la rama que se ocupa de imágenes estáticas es decir, que no forman parte de una secuencia y simplemente son una representación de objetos en una situación. El término "por contenido" se refiere a que los únicos datos que se utilizan para formular las consultas son aquellos que comprenden la imagen en sí, o sea su contenido. Existen métodos para extraer información por medio de parámetros que se basan en las características más notables de las imágenes como son: colores, texturas y formas. Mediante las características anteriores, existen algoritmos que pueden calcular la similitud entre dos imágenes [Chua et. al 1998]. De esta manera, un programa puede recibir una imagen como entrada, procesar la consulta, y devolver aquellas imágenes que sean las más parecidas.

2.1.1 ¿Cómo funciona?

Para que una búsqueda por contenido se pueda llevar a cabo, se requiere básicamente de dos procesos. El primero se realiza sobre las imágenes que se insertan en

la base de datos y sirve para extraer y representar sus características de bajo nivel (color, textura, figuras), por ejemplo, como un vector. Cada posición del vector representa una cierta característica de la imagen y contiene un valor asociado que le da significado dependiendo de la característica.

El segundo proceso se realiza sobre la imagen de entrada al momento de la consulta. Se toman sus características y se comparan con aquellas previamente almacenadas para determinar la "distancia" entre ellas. La distancia representa el grado de similitud que existe entre las imágenes. Tanto los desarrolladores como los usuarios pueden especificar un rango de similitud para darle un cierto grado de flexibilidad al proceso de comparación. De igual forma puede haber una retroalimentación por parte del usuario para poder ir refinando los resultados. La Figura 2.1 muestra el procedimiento que el usuario realiza al utilizar CBIR. En este caso se tiene la opción de hacer una consulta por ejemplo visual. También podemos observar aquellos procesos que se llevan a cabo fuera del contexto de la consulta, como la extracción de características del material visual por medio de técnicas de procesamiento de imágenes. Una vez que el usuario haya visualizado los resultados, puede retroalimentar al sistema para filtrar la información relevante. El proceso se repite hasta que el usuario lo desee.

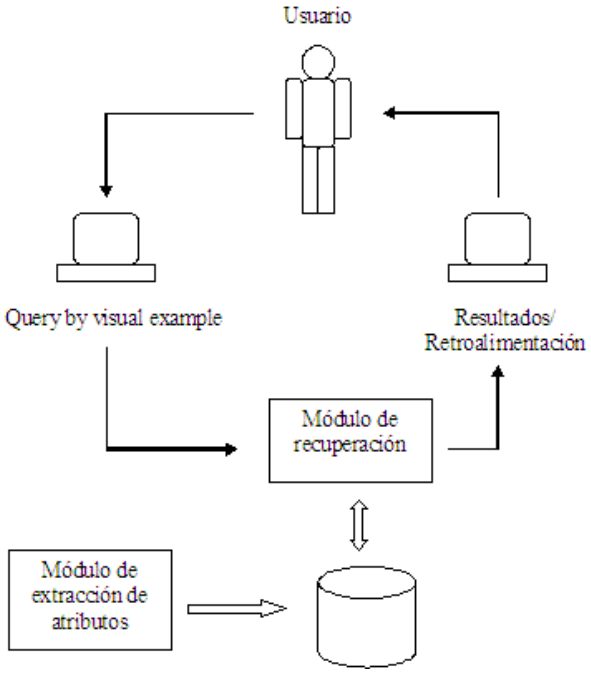

Figura 2.1 El proceso de CBIR (Adaptación de [Del Bimbo 1999]).

2.2 CBIR en la actualidad

En este capítulo mostramos algunas de las técnicas y aplicaciones más recientes que utilizan el paradigma de CBIR. En cuanto a las metodologías, estas se enfocan principalmente a la extracción de propiedades visuales de las imágenes, la representación y almacenamiento de dichas propiedades y su comparación para poder determinar la similitud entre imágenes. Ya que el área de CBIR se encuentra actualmente bajo una investigación extensiva, no fue posible incluir en este capítulo todas las técnicas, algoritmos e ideas existentes. Aquellas que sí forman parte de este trabajo fueron elegidas tomando en cuenta: *(1)* que su año de publicación sea relativamente reciente, *(2)* que sus resultados experimentales demuestren eficiencia y factibilidad comparados con otros enfoques, *(3)* que sea una técnica comúnmente utilizada en aplicaciones existentes. Al final del capítulo se comparan los distintos enfoques con base en sus ventajas,

desventajas y resultados experimentales. También se hará un análisis más detallado de las técnicas que se implementarán para el proyecto.

2.3 Aplicaciones existentes

Actualmente ya existen aplicaciones orientadas a la recuperación de información visual y/o audiovisual, ya sean aplicaciones con fines comerciales o aplicaciones desarrolladas para la investigación dentro del campo de CBIR. Entre las aplicaciones comerciales podemos encontrar a QBIC [Ashley et. al 1995], Virage[1], IMatch[2] de MWLabs e ImageFinder[3] de Attrasoft. Algunas de estas aplicaciones están diseñadas para trabajar sobre bases de datos que el usuario ya haya creado y tienen integrados componentes que permiten una fácil interacción con manejadores de bases de datos comerciales, tales como Informix y Oracle.

Asimismo, existen herramientas de desarrollo (SDK) que permiten crear aplicaciones para la recuperación de información visual, como Excalibur Visual RetrievalWare Software Developers Kit (Excalibur Corporation[4]).

Debido a que son aplicaciones comerciales, su uso está sujeto a la compra de licencias por lo que se descartan para ser utilizadas en este proyecto. Otras desventajas que presentan algunos sistemas como ImageFinder e IMatch es que tienen requerimientos específicos de plataforma (están diseñados para correr bajo Windows).

Ahora veamos un poco de las aplicaciones y prototipos que se han desarrollado con fines de investigación. Tenemos sistemas como VisualSEEk [Smith y Chang 1996]

[1] www.virage.com
[2] www.mwlabs.de
[3] www.attrasoft.com
[4] www.excalib.com

cuyo enfoque permite al usuario crear consultas seleccionando y acomodando regiones de colores. Además cuenta con una extracción automática de atributos de las imágenes, representados por sus regiones de colores más sobresalientes. WebSEEk [Chang et. al 1997] de la Universidad de Columbia, cuya característica principal es la recuperación de imágenes en Web, utilizando agentes autónomos que la "exploran" y recopilan contenido multimedial (a estos agentes se les denomina *web crawlers* en inglés). NeTra [Ma y Manjunath 1997] de la Universidad de California Santa Barbara permite la recuperación de imágenes formulando consultas basadas en color, textura y formas gracias a un algoritmo robusto de segmentación. En la misma institución también se ha desarrollado el sistema CIS [Newsam et. al 2001] que combina la búsqueda por contenido con la búsqueda textual. Blobworld [Carson et. al 1999] de la Universidad de Berkeley cuyo enfoque realiza una segmentación de regiones similares en cuanto a color y textura, además de combinar la búsqueda por contenido con la búsqueda textual. Dichas regiones corresponden de manera aproximada a los objetos contenidos en las imágenes. Finalmente, está Viper[5] de la Universidad de Génova (Suiza), que emplea extensivamente la retroalimentación utilizando un protocolo de comunicación basado en XML llamado MRML (Multimedia Retrieval Markup Language). Estos sistemas trabajan con bases de datos que contienen imágenes tanto de dominio específico como de dominio general, mientras que otras trabajan con imágenes obtenidas de fuentes comerciales como Corel Gallery Magic (imágenes disponibles en formato CD-ROM). Incluso hay aplicaciones que trabajan con contenido que reside en el Web, como es el caso de WebSEEk [Chang et. al 1997] y CIS [Newsam et. al 2001].

[5] http://viper.unige.ch

El objetivo principal de estas aplicaciones ha sido experimentar y comparar las numerosas y distintas técnicas, algoritmos y heurísticas que se han desarrollado por investigadores en el campo de CBIR. Gracias a sus resultados, se han podido perfeccionar ciertas metodologías, mientras que otras se vuelven populares y otras tantas se vuelven obsoletas. Sin embargo, debido a sus propósitos de investigación, muchas de estas aplicaciones no están hechas para el usuario común, por lo que sus interfaces de usuario e instrucciones de uso requieren de conocimiento específico. Todo lo anterior constituye una desventaja desde el punto de vista práctico, sin desmeritar su importante aportación al avance de la tecnología. Debemos recordar que uno de los objetivos de este proyecto es crear una aplicación flexible y sencilla de manejar para los usuarios de bibliotecas digitales.

2.4 CBIR en bibliotecas digitales

Actualmente, existe un considerable número de aplicaciones para la búsqueda de imágenes por contenido que operan en el contexto o forman parte de un proyecto de bibliotecas digitales. Algunas se encuentran a nivel de prototipo o *demo* accesible vía Web, como es el caso de CIS[6] [Newsam et. al 2001] y Blobworld[7] [Carson et. al 1999]. Otras aplicaciones sirven propósitos específicos de investigación y actualmente su acceso no está disponible al público, como es el caso de NeTra [Ma y Manjunath 1997].

La UCSB (University of California Santa Barbara) tiene un proyecto de biblioteca digital llamado *Alexandria Digital Library Project* [8] (ADL, por sus siglas en inglés) cuyo

[6] http://nayana.ece.ucsb.edu/imsearch/imsearch.html
[7] http://elib.cs.berkeley.edu/photos/blobworld
[8] http://www.alexandria.ucsb.edu/

objetivo es crear una biblioteca digital distribuida de datos geográficos como mapas, imágenes de satélite y fotografías aéreas. Dentro de esta gran iniciativa existe el interés de poder aplicar enfoques de CBIR para poder extraer información importante del material geográfico mencionado. Algunos ejemplos pueden ser el poder buscar regiones con vegetación similar, estacionamientos disponibles en áreas con grandes concentraciones de actividad humana o regiones de la tierra con determinadas características climatológicas. Este proyecto, denominado *Content-based image browsing and retrieval using texture features*, aún está en su etapa inicial y sus avances pueden encontrarse en [Bhagavathy et. al 2002]. En la investigación anterior se propone un modelo canónico para objetos existentes en imágenes aéreas, basándose en la observación de que existen regiones geográficas que pueden ser caracterizadas por texturas.

Por su parte, la Universidad de California en Berkeley tiene dentro de su proyecto de biblioteca digital un área de tecnología relacionada con CBIR. Una de las aplicaciones resultantes de esta rama es Blobworld [Carson et. al 1999] y combina la búsqueda por contenido con la búsqueda textual. Esta combinación es posible ya que las imágenes provienen de una fuente comercial (Corel Gallery Magic) y vienen con anotaciones textuales que describen su contenido (ejemplos disponibles en: http://elib.cs.berkeley.edu/photos/corel/).

Existe además otro proyecto que se lleva a cabo en los Archivos de Películas y Documentos Fotográficos del Estado Ruso (*Russian State Archives of Film and Photo Documents*[9]), y que cuenta con un sistema para realizar búsquedas textuales sobre el material. Los autores de este sistema reconocen la necesidad de crear nuevas

[9] http://www.russianarchives.com/rao/archives/rgakfd/

herramientas que permitan un acceso al contenido multimedial distinto al que ya se tiene, y al igual que el proyecto anterior, exploran la posibilidad de poder combinar las búsquedas textuales y las búsquedas por contenido. También hacen un análisis de algunos de los enfoques existentes de CBIR que se consideran para el desarrollo del proyecto. Dicho análisis se puede encontrar en [Baigarova et. al 2001].

En el contexto de U-DL-A, existen ya dos aplicaciones que permiten consultar y recuperar información visual y audiovisual. La primera es CIText [García 2002] que permite hacer consultas textuales en libros digitalizados de la Biblioteca Franciscana. Esto se lleva a cabo primero mediante el reconocimiento óptico de caracteres (OCR, por sus siglas en inglés) que extraen el texto de las imágenes y posteriormente utilizando los algoritmos de búsqueda Soundex y Similarex para realizar búsquedas textuales, los cuales tienen una cierta tolerancia a los errores que se pudieron haber generado por el OCR. El otro sistema, llamado Video U-DL-A [Arias 2002], permite crear índices en videos para poder realizar búsquedas por contenido y anotaciones. Este sistema primero extrae las características de los videos separando la parte audible de la visible. La parte audible se procesa mediante un reconocedor de voz y el texto generado sirve para generar los metadatos. La parte visible se segmenta en ventanas que, en conjunto, generan una secuencia de video y sus características se extraen utilizando extensiones de Informix (módulos DataBlade). Dichos módulos también se encargan de realizar la recuperación de imágenes por contenido. El usuario puede realizar anotaciones correspondientes a segmentos de video específicos y también puede visualizarlos mediante funciones de video en demanda.

2.5 Técnicas y enfoques para CBIR

En esta sección se presentan algunas técnicas para CBIR que han surgido de investigaciones previas. Antes de comenzar, es importante dar primero una clasificación de estos enfoques para tener más claro su contexto de aplicación y poder compararlos de manera justa. Autores como [Stehling et. al 2002] y [Li et. al 2000] definen una clasificación general en base a tres tipos de enfoques utilizados: *(1)* enfoques globales, *(2)* enfoques basados en particiones y *(3)* enfoques basados en regiones. A continuación veremos con más detalle cada uno de estos enfoques, así como aplicaciones y prototipos que se han desarrollado en base a ellos.

2.5.1 Enfoques globales

Los enfoques globales se refieren a aquellas metodologías que usan histogramas para representar imágenes. Un histograma es, de acuerdo al diccionario de la lengua española, una distribución gráfica de frecuencias por medio de rectángulos, cuyas anchuras representan intervalos de la clasificación y cuyas alturas representan las correspondientes frecuencias. En el contexto de CBIR, los intervalos de la clasificación corresponden a los colores que se están cuantificando y las frecuencias corresponden al número de píxeles, de un determinado color, que hay en una imagen dada. Más adelante veremos una definición formal de este término. Al utilizar esta técnica, se intenta capturar las propiedades globales de una imagen sin tomar en cuenta información espacial. Por lo general, la propiedad con la cual se trabaja es el color. Debido a que la visión humana percibe una gran diversidad de colores, esta propiedad es una de las más populares en CBIR. A continuación se presentan técnicas comunes y novedosas que

funcionan bajo este enfoque, así como una breve descripción de algunas representaciones de colores.

La investigación hecha por [Smuelders et. al 2000] proporciona información acerca de distintas representaciones de color y sugiere situaciones bajo las cuales es más conveniente usar cada una de ellas. Lo siguiente se toma de la referencia anterior:

- **RGB (*Red Green Blue*)**: Describe los colores en base a tres componentes: rojo, verde y azul. Cada componente puede tener un valor entre 0 y 255, lo cual nos da casi 16 millones de diferentes colores. Es conveniente usar esta representación cuando no existe variación en la grabación de las imágenes (por ejemplo, cuando un conjunto de imágenes se graban desde la misma posición utilizando una iluminación uniforme). Esto es debido a que esta representación de colores fue diseñada para igualar el canal de entrada del ojo humano.

- **Munsell**: Describe los colores en base a tres atributos: matiz, valor y cromo. El matiz es lo que nos hace distinguir entre dos colores diferentes. El valor especifica la claridad de un color y su valor va desde 0 a 10 (0 para negro puro y 10 para blanco puro). Cromo es el grado de lejanía que un color tiene con respecto al color neutral de su mismo valor. La ventaja de este sistema es que tiene una percepción uniforme.

- **HSV (Hue Saturation Value)**: Define un color en base al matiz, la saturación y el valor. El concepto de matiz es el mismo que en el sistema de Munsell. La saturación se refiere al grado de pureza del color y su porcentaje va desde 0%

(sombra gris) hasta 100% (color puro). El valor se refiere a la claridad del color. Su ventaja es que el matiz no varía con el cambio de iluminación y la posición de la cámara.

2.5.1.1 Histogramas de color global

Un histograma de color global (GCH, por sus siglas en inglés) define el número de píxeles de cada color que existen en una imagen determinada. De manera más formal, un histograma H de una imagen I es un vector ($h_1, h_2, \ldots, h_n$) en el que cada posición (también llamada "cubeta") h_j contiene el número de píxeles del color j [Pass et. al 1996] que hay en toda la imagen I. Nos proporciona información acerca de la cantidad de cada color que existe en la imagen. Sus ventajas son su rápida implementación, eficiencia (desde el punto de vista computacional) e invariabilidad ante ciertos cambios en la imagen como son rotación y ciertas variaciones en el ángulo con que fueron tomadas.

En cambio, sus desventajas son la sensibilidad a la compresión de la imagen, los cambios de brillo y sobre todo el hecho de que no proporciona ningún tipo de información espacial. Esto último quiere decir que no indica la localización de los diferentes colores, únicamente nos dice su cantidad dentro de la imagen. Debido a esto, puede haber dos imágenes completamente diferentes que tengan histogramas muy parecidos (como se muestra en la Figura 2.2), lo cual nos resulta en imágenes irrelevantes como resultado de una búsqueda por contenido.

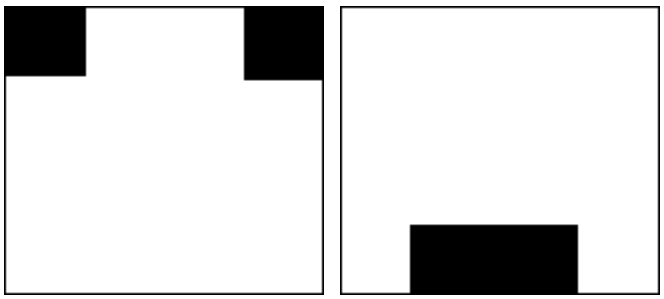

Figura 2.2 Dos imágenes distintas con histogramas similares.

2.5.1.1.1 Implementación de histogramas

Su implementación es sencilla y relativamente rápida debido a que únicamente se requiere hacer un barrido de la imagen, colocando cada píxel en la cubeta del color que le corresponda. Por esto, es claro que un algoritmo para calcular un histograma corre en $O(n)$, donde n es el número de píxeles que componen la imagen.

2.5.1.1.2 Comparación de histogramas

Para medir la distancia entre dos histogramas, y por consiguiente la distancia entre dos imágenes (representadas por los histogramas), se utilizan comúnmente dos tipos de métricas: la función de distancia L1 y la función L2 [Pass et. al 1996].

Función L1:

$$d(h,q) = \sum_{j=1}^{n} \left| h(j) - q(j) \right|$$

Función L2:

$$d(h,q) = \sum_{j=1}^{n} \left[h(j) - q(j) \right]^2$$

Donde:

- *h* y *q* son dos histogramas con *n* cubetas cada uno.
- *h(j)* es el número de píxeles del color *j* en el histograma *h* y *q(j)* es el número de píxeles del color *j* en el histograma *q*.

Otra limitante conocida es el hecho de que un valor muy alto en alguna de las cubetas de un histograma tiene una gran influencia sobre su distancia con respecto a otro histograma.

2.5.1.2 Vectores de coherencia de colores (Color coherence vectors)

Una manera eficiente de combatir la deficiencia más sobresaliente de los histogramas, la cual es la falta de información espacial, es el uso de vectores de coherencia de colores (CCV, por sus siglas en inglés) [Pass et. al 1996]. Los CCV utilizan histogramas de color junto con información acerca de la ubicación de los colores para poder determinar la similitud entre imágenes.

Los autores de éste método definen la coherencia de un color como el grado en que los píxeles de ese color pertenecen a regiones lo suficientemente grandes de colores similares. Un píxel coherente forma parte de una región grande de un color similar y un píxel incoherente no. Los CCV comparan las imágenes en base a su clasificación de píxeles coherentes e incoherentes.

2.5.1.2.1 Implementación de CCV

Podemos definir un algoritmo general para calcular el CCV de una imagen:

1. Distorsionar ligeramente la imagen sustituyendo el valor de los píxeles por el valor promedio en una "vecindad" (*neighborhood*), la cual se conforma del píxel en turno y todos sus píxeles adyacentes. Los autores justifican este proceso para eliminar pequeñas diferencias entre píxeles adyacentes.
2. Reducir el espacio de colores existentes en la imagen, tal que haya solamente n colores distintos. Esto se logra haciendo que cada cubeta del vector, que representa el histograma, acepte un cierto rango de tonalidades del mismo color (por ejemplo, todas las tonalidades del color azul caen dentro de la misma cubeta).
3. Calcular las regiones conectadas en la imagen. Una región conectada se define como un conjunto de píxeles, tal que para cada par de píxeles del mismo color pertenecientes al conjunto, existe una ruta entre ellos formada por píxeles adyacentes de igual color.
4. Para cada píxel de la imagen, determinar si es coherente o incoherente. Como se mencionó anteriormente, la coherencia existe si el píxel es miembro de una región conectada lo suficientemente grande y es incoherente en caso contrario. Para determinar si el tamaño (número de píxeles) de la región a la cual pertenece el píxel es aceptable se utiliza una variable τ cuyo valor es fijo. Los resultados experimentales sugieren que el valor de τ sea de aproximadamente 1% del área total de la imagen. Por ejemplo, si tenemos una región conectada R y su tamaño es de 500 y definimos $\tau = 400$, entonces cualquier píxel que pertenezca a R es coherente.

5. Para cada color *j* en el histograma se calcula el par (α_j,β_j), donde α representa el número total de píxeles coherentes del color *j* y β el número total de píxeles incoherentes. De esta manera, el histograma queda de la siguiente manera:

$$H = \left((\alpha_1,\ \beta_1),\ (\alpha_2,\ \beta_2),\ ...,\ (\alpha_n,\ \beta_n) \right)$$

Como podemos ver, se requieren dos procesos para calcular un CCV, por lo tanto tarda un poco más. Un proceso para determinar las regiones conectadas, el cual corre en tiempo lineal con respecto al número de píxeles. El segundo para calcular el histograma, el cual corre también en tiempo lineal (como se explicó en la sección 2.5.1.2). Sin embargo, debemos tomar en cuenta que los CCV mejoran el desempeño de los histogramas comunes, lo cual compensa por la complejidad de su algoritmo.

2.5.1.2.2 Comparación de CCV

Los autores suponen dos imágenes, *I* e *I'*, cada una con su CCV respectivo ya calculado *G* y *G'*. Como se explicó en la sección anterior, los CCV contienen un par de valores para cada color. Dichos valores representan el número de píxeles coherentes e incoherentes para cada cubeta de color. El método para comparar ambos CCV se basa en la comparación de histogramas convencionales. Podemos calcular la distancia entre *G* y *G'* como:

$$G = \left((\alpha_1,\ \beta_1),\ (\alpha_2,\ \beta_2),\ ...,\ (\alpha_n,\ \beta_n) \right)$$

$$G' = \left((\alpha'_1,\ \beta'_1),\ (\alpha'_2,\ \beta'_2),\ ...,\ (\alpha'_n,\ \beta'_n) \right)$$

$$d(G,G') = \sum_{j=1}^{n} \left|\alpha_j - \alpha'_j\right| + \left|\beta_j - \beta'_j\right|$$

2.5.1.3 Firmas binarias (Binary signatures)

Las firmas binarias fueron propuestas por [Nascimento y Chitkara 2002] y son una manera de representar imágenes mediante cadenas de caracteres binarios (0, 1). Dichas cadenas sirven para representar la distribución de los colores existentes en una imagen. Los autores ponen mayor énfasis en los colores que no predominan en la imagen, (a diferencia de la metodología de los histogramas globales de color que dan igual prioridad a todos los colores) sin olvidar aquellos que si lo hacen. Lo anterior se debe a que las investigaciones de [Falmagne, 1986] revelaron que la percepción de estímulos de colores, particularmente en imágenes, siguen una curva "sigmoidal" y no un patrón lineal (como se supone en los histogramas de color). Lo anterior se muestra en la Figura 2.3.

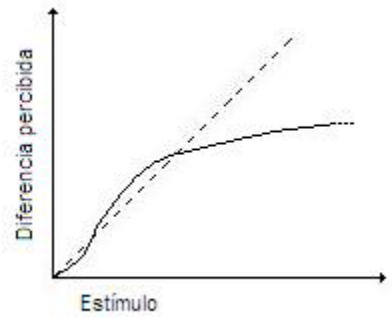

Figura 2.3 La percepción del estímulo de colores en el ser humano (Adaptación de [Falmagne, 1986]).

2.5.1.3.1 Implementación de firmas binarias

Un algoritmo general, propuesto por los autores, para calcular la firma binaria de una imagen se da a continuación:

1. Se cuantifica la imagen en *n* colores distintos $C = (c_1, c_2, \ldots, c_n)$. Esto se hace con la finalidad de eliminar pequeñas variaciones irrelevantes en las imágenes.

2. Cada color c_j se divide a su vez en t cubetas binarias $B_j = (b_j^1, b_j^2, \ldots, b_j^t)$. El número de cubetas puede ser constante o variable dependiendo del tipo de enfoque (se explicaran más adelante los tipos de enfoques: CBA y VBA). Cada cubeta acepta un cierto rango de densidad de su color respectivo. Supongamos una imagen constituida por *n* colores y *t* cubetas. Su firma binaria estaría compuesta por la siguiente cadena de bits:

$$S = b_1^1 b_2^1 \ldots b_t^1 \; b_1^2 b_2^2 \ldots b_t^2 \ldots b_1^n b_2^n \ldots b_t^n$$

   Donde el bit b_j^i representa la cubeta *j* del color *i*.

3. Se calcula la densidad de cada color en la imagen como:

$$d(j) = \frac{p_j}{T}$$

Donde p_j representa el número de píxeles del color *j* en la imagen y *T* es el número total de píxeles que la componen. Estos valores se usan para asignarle un valor (binario) a cada bit de la cuerda *S*.

4. Se le asigna el valor de 1 al bit b_j^i si la densidad del color *i* está dentro del rango que acepta el bit *j*. Se le asigna el valor de 0 en caso contrario.

Tomemos como ejemplo la Figura 2.4 que tiene 3 colores (negro, gris, blanco), $C = (c_1, c_2, c_3)$ y sea $t = 10$. Ya que la imagen tiene 16 píxeles en total, el histograma de densidades quedaría como $H = (0.18, 0.06, 0.75)$. Supongamos que cada cubeta acepta un rango de una décima parte del total de la densidad de su color. De esta manera tendríamos que la cubeta b_1 acepta de 1% a 10%, la cubeta b_2 acepta de 11% a 20% y así sucesivamente para cada color. Para el blanco, su densidad cae dentro de la cubeta b_8. La densidad del gris cae dentro de la cubeta b_1. Aquella del color negro cae dentro de la cubeta b_2. Finalmente, tenemos que la firma binaria S_A = 0100000000 1000000000 0000000100.

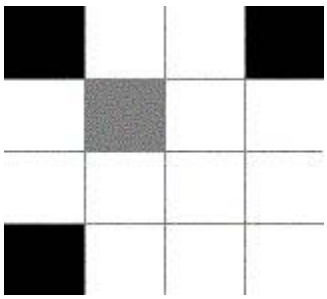

Figura 2.4 Una imagen de 16 bits con 3 colores: negro, gris y blanco (Adaptación de [Nascimento y Chitkara 2002]).

2.5.1.3.1.1 Asignación constante de cubetas (CBA)

Es uno de los enfoques que tienen las firmas binarias en donde cada cubeta tiene la misma capacidad (como en el ejemplo anterior). Es equivalente a dar la misma prioridad a todos los colores.

2.5.1.3.1.2 Asignación variable de cubetas (VBA)

Es el segundo enfoque que tienen las firmas binarias en el que cada cubeta tiene diferente capacidad para que las distancias que surjan de colores menos significativos tengan una mayor importancia. Los experimentos hechos por los autores definen $t = 10$ cubetas por color, donde las cubetas b_1, b_2, b_3 contienen cada una el 3% de la densidad total, las cubetas b_4, b_5 contienen cada una el 5%, las cubetas b_6, b_7, b_8, b_9 contienen cada una 10% y finalmente la cubeta b_{10} contiene del 60% hasta el 100%. Las capacidades anteriores fueron determinadas a través de la experimentación y se apegan al patrón de la curva sigmoidal mencionada anteriormente.

2.5.1.3.2 Ventajas y desventajas de usar firmas binarias

Ventajas:

- Fácil de implementar.
- Se tiene una representación compacta y discreta de la distribución de color en una imagen.
- La representación de la imagen ocupa menos espacio comparado con histogramas globales y CCV. Lo anterior se justifica si tomamos en cuenta que se requieren f bytes para almacenar un valor real. Para almacenar un histograma de n cubetas de color se requieren ($n \times f$) bytes, para almacenar un CCV se requieren el doble de bytes (debido a que para cada cubeta se almacena un valor para píxeles coherentes y otro para los incoherentes), sin embargo para este método se requieren solamente ($n \times t$) bits.
- La secuencia de valores binarios se pude comprimir para ahorrar todavía más espacio.

Desventajas:

- No toma en cuenta la ubicación espacial de los colores (a diferencia de CCV).

2.5.1.3.3 Comparación de firmas binarias

Para comparar dos imágenes y obtener su grado de similitud, los autores han desarrollado la siguiente ecuación:

$$d(Q,I) = \sum_{j=1}^{n} \left[pos(B_Q^j) - pos(B_I^j) \right]^2$$

Donde:

- d es la distancia entre la imagen Q (imagen de consulta) y la imagen I.
- n es el número de colores que componen las imágenes.
- pos (B_R^k) es la posición del bit cuyo valor es 1 y que corresponde al conjunto de cubetas del color k en la imagen R.

Una vez calculada la distancia entre la imagen de consulta y todas las demás imágenes de la base de datos, se ordenan de manera ascendente y se toman como resultados aquellas imágenes cuya distancia a la imagen de consulta es menor.

2.5.2 Enfoques basados en particiones

El objetivo de estos enfoques trata de superar la principal desventaja de los enfoques globales, la cual es la falta de información espacial acerca de las propiedades

extraídas de la imagen. Primero divide la imagen en un número fijo de particiones y para cada una, se obtienen las propiedades de bajo nivel deseadas. De esta manera, se tiene una relación entre dichas propiedades y su ubicación en la imagen. El problema con este esquema de segmentación fija es que no se identifican los objetos en la imagen, como un ser humano lo haría naturalmente, lo cual resulta en pérdida de semántica en la representación de la imagen (lo más probable es que los objetos queden partidos en varias regiones). Se complica aún más la determinación de similitud entre imágenes que contienen el mismo objeto en diferentes lugares dentro de la imagen ya que las particiones son estáticas. La siguiente sección presenta una técnica sencilla que explicará la metodología de este enfoque con mayor claridad.

2.5.2.1 Enfoque Grid9

Es uno de los enfoques más sencillos dentro de esta categoría. Consiste en dividir la imagen en 9 celdas de tamaño uniforme (3 filas x 3 columnas), como se muestra en la Figura 2.4. Para cada celda se calcula un histograma de color local, lo cual pretende representar el color y su ubicación dentro de la imagen.

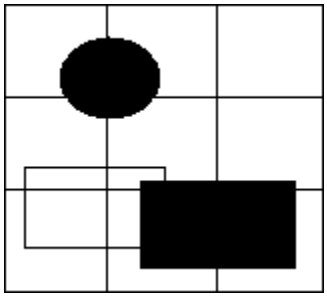

Figura 2.5 Una imagen segmentada en base al enfoque de Grid9.

2.5.2.1.1 Métrica de comparación

Para calcular la distancia entre dos representaciones de Grid9, se calcula el promedio de la distancia L1 (vista en la sección 2.4.1) entre los histogramas de cada celda, respectivamente.

2.5.2.1.2 Ventajas y desventajas

Ventajas

- Implementación sencilla y rápida.

Desventajas

- Se requieren 9 histogramas para representar una sola imagen.
- Tiene menor precisión que otros enfoques.

2.5.3 Enfoques basados en regiones

Con este enfoque se pretende suplir los enfoques anteriores en aspectos importantes. El primero de ellos es representar la imagen en base a los objetos contenidos en ella, lo cual asemeja la forma en que los humanos observamos y descomponemos una imagen en objetos (personas, árboles, cielo, mar, etc.) y no en regiones de color o particiones fijas. El segundo aspecto, el cual tiene que ver con la semántica, se encuentra relacionado con el primero en el sentido que los objetos dan un significado a la imagen. Si los identificamos, podemos deducir con mayor facilidad el contexto y la situación que la imagen describe. Representar una imagen a este nivel de objetos requiere de una segmentación mucho más compleja. Se deben identificar las regiones que correspondan a objetos, suponiendo que podemos lograr una segmentación

perfecta. La dificultad de este proceso depende de las condiciones bajo las cuales la imagen fue creada (iluminación, resolución al ser digitalizada, etc.).

2.5.3.1 Clasificación de píxeles borde/interior

Este enfoque para CBIR fue propuesto por [Stehling et. al 2002] y promete ser simple, eficiente y robusto. Este nuevo enfoque, el cual llamaremos de ahora en adelante BIC (Border/Interior píxel Classification) por sus siglas en inglés, esta formado por tres elementos: (*1*) un algoritmo que clasifica los píxeles de la imagen como *borde* o *interior*, creando así, dos histogramas, (*2*) una función logarítmica que compara las distancias entre dos histogramas y (*3*) una manera compacta de representar las propiedades que se extraen de la imágenes. Más adelante se explicará con mayor detalle cada uno de los componentes mencionados.

2.5.3.1.1 Algoritmo de clasificación

El algoritmo que utiliza el enfoque BIC para procesar la imagen trabaja sobre el espacio RGB con 64 colores, esto debido a que los resultados experimentales de muchos autores que trabajan con el procesamiento de color (comúnmente en histogramas) demuestran su efectividad. Sin embargo, no se descarta el uso de otro tipo de representación de colores para este método, ya que todo depende del contexto de la aplicación y sus requerimientos. También supone que un píxel tiene solamente 4 píxeles adyacentes (arriba, abajo, izquierda, derecha) en lugar de 8 debido a que reduce las operaciones necesarias para el proceso sin que los atributos extraídos pierdan influencia sobre la búsqueda.

Los siguientes pasos ilustran de manera más clara la función del algoritmo:

1. Reducir el espacio de colores existentes, tal que solo haya 64 colores distintos (por ejemplo, píxeles de diferentes tonalidades rojas se toman simplemente como de color rojo).
2. Clasificar cada píxel en la imagen como borde o interior según el siguiente criterio:
    - Un píxel es *borde* si se encuentra en el borde de la imagen o si por lo menos uno de los cuatro píxeles adyacentes es de un color distinto a él. Lo anterior significa que el píxel en cuestión se encuentra en una región donde la división de los colores puede indicar el borde de un objeto.

    - Un píxel es *interior* si todos sus píxeles adyacentes tienen el mismo color que él. Esto quiere decir que el píxel en cuestión es parte de una región homogénea (en cuanto al color), lo cual puede indicar una parte principal al interior de un objeto.
3. Una vez terminada la clasificación de todos los píxeles de la imagen, se tienen dos histogramas (uno para píxeles borde y otro para píxeles interior) de color, cada uno con 64 cubetas. Los autores también dan la opción de hacer un solo histograma con 128 cubetas. Los valores de cada cubeta se normalizan para que queden en un rango de 0 a 255. Los autores hacen esto para poder representar el valor de una sola cubeta con tan solo 1 byte de memoria ($2^8 - 1 = 255$). Otra de las razones que justifican tal normalización es que los resultados prácticos no han demostrado mejoras usando más de 255 valores para las cubetas.

2.5.3.1.2 Función de distancia logarítmica

La función que los autores proponen para medir la distancia entre los histogramas se denomina *dLog* debido a que compara las distancias en una escala logarítmica. Esta función reduce en la mayoría de los casos las limitantes que tienen las métricas comunes que se usan para medir la distancia entre histogramas (L1, L2). Recordemos que este tipo de métricas tienen la desventaja de que un valor muy elevado en una de las cubetas de un histograma influye de gran manera en la distancia con otro histograma (como se mencionó en la sección anterior). En cambio, usando una escala logarítmica, se reduce la diferencia entre las distancia más pequeña y la más grande, lo cual disminuye los efectos que estas distancias extremas pudieran tener sobre la distancia total entre histogramas.

De esta manera tenemos que la función *dLog* se define como:

$$dLog(q,d) = \sum_{i=0}^{i<M} \left| f(q[i]) - f(d[i]) \right|$$

Donde:

- q y d son histogramas de M cubetas cada uno.
- $q[i]$ es el valor de la cubeta i del histograma q.
- $d[i]$ es el valor de la cubeta i del histograma d.

2.5.3.1.3 Representación y almacenamiento de los histogramas

Una vez que se han calculado los histogramas de la imagen es necesario guardar los valores de cada cubeta. Los autores han desarrollado una función que permite reducir el espacio necesario para guardar tales valores, sustituyendo el valor normalizado de cada cubeta por su equivalente en una escala logarítmica.

La función es:

$$f(x) = 0, \text{ si } x = 0$$
$$f(x) = 1, \text{ si } 0 < x \leq 1$$
$$f(x) = \lfloor \log_2 x \rfloor + 1, \text{ de lo contrario}$$

Donde:

- x es el número de píxeles en la cubeta i del histograma h.
- $f(x)$ es el valor de la cubeta en la escala logarítmica.

Nótese que los valores en esta escala van desde 0 hasta 9 ($\log_2 255 + 1 = 9$), lo cual nos permite guardarlos con tan solo 4 bits. Esto representa una mejora del 50% en cuanto a espacio de almacenamiento (recordemos que se necesitan 8 bits para almacenar valores de 0 a 255). Todo esto es útil si consideramos que el proceso de búsqueda se hace en memoria RAM, así podemos tener un mayor número de histogramas BIC en memoria haciendo más veloz el proceso de comparación.

2.6 Comparación de enfoques

Ahora que existe un panorama general de las categorías en que se dividen actualmente las aplicaciones de CBIR y algunos de los enfoques que las componen, hacemos un análisis de la comparación experimental entre ellos para poder determinar aquel que se implementará en este proyecto y justificar su uso. Afortunadamente, esta comparación ya ha sido realizada por [Stehling et. al 2002] y la podemos tomar como una referencia confiable dada su vigencia. La siguiente sección explica el esquema de pruebas que fue utilizado y los resultados que se obtuvieron.

2.6.1 Pruebas y resultados

Como se mencionó en la sección anterior, los autores de BIC realizaron la comparación entre su enfoque y todos aquellos mencionados en las secciones anteriores (con excepción de firmas binarias) con el propósito de determinar la eficiencia de cada uno con respecto a la cantidad de imágenes relevantes obtenidas dada una imagen de consulta (*query-by-example*). Para ver con mayor detalles el esquema de pruebas referirse a la sección 4 de [Stehling et. al 2002].

Los resultados de todas las pruebas realizadas por [Stehling et. al 2002] fueron favorables para el enfoque BIC, el cual demostró ser el más eficiente. Las comparaciones entre BIC y los demás enfoques existentes comprobaron la suposición general de que aquellos que trabajan en base a regiones son mejores (por ejemplo, BIC es mejor que Grid9 y este a su vez supera a GCH).

Con respecto a la comparación de la función *dLog* y la métrica L1 utilizada en los demás enfoques, esta también resultó ser más eficiente. Aún cuando los otros enfoques fueron modificados para usar *dLog*, el enfoque BIC fue el más sobresaliente. Para ver con mayor detalle los resultados mencionados, referirse a la sección 5 de [Stehling et. al 2002].

2.6.2 Análisis de enfoques

Con base en la información que se ha presentado con respecto a los enfoques para CBIR, es necesario hacer un análisis comparativo de dichos enfoques para poder seleccionar aquel que será implementado y poder justificar su uso de manera válida. Los criterios de selección que han sido analizados son:

- Eficiencia del algoritmo de acuerdo con los resultados de las pruebas realizadas por [Stehling et. al 2002]
- Vigencia del algoritmo (año de publicación)
- Que el costo de implementación del algoritmo esté al alcance de los recursos disponibles para este proyecto
- Eficiencia de la función de comparación utilizado por el algoritmo de acuerdo con los resultados de las pruebas realizadas por [Stehling et. al 2002]

Los criterios mencionados se miden en una escala de 1 a 5, donde 5 es la mejor calificación y 1 es la peor. En caso de que algún criterio no se pueda aplicar o no que no haya información disponible al respecto se marcará con valor de 0. La Figura 2.4 muestra los resultados del análisis.

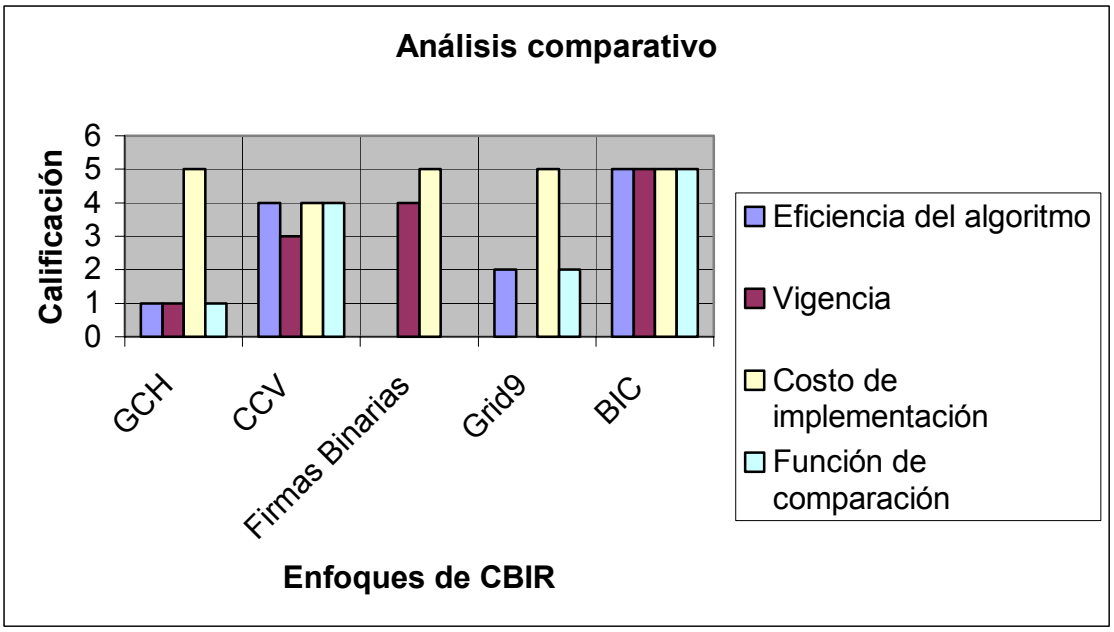

Figura 2.6 Gráfica del análisis comparativo entre los distintos enfoques de CBIR.

De la gráfica anterior podemos ver que el uso de BIC en nuestro sistema es una buena opción, y por lo tanto es el que será implementado.

2.7 Enfoque de retroalimentación

El objetivo de este proceso en nuestro sistema es que este aprenda en base a la información que los usuarios le proporcionen. Debido a que el algoritmo está limitado a establecer la similitud entre imágenes usando solamente características de bajo nivel (en este caso, el color), no puede hacerlo en base al contenido semántico de las imágenes. Aquí es donde el papel del usuario toma una gran importancia, ya que los seres humanos tenemos gran capacidad de percepción y distinción. Estas capacidades humanas servirán como complemento a las capacidades del algoritmo. Con esto, se pretende crear un ambiente de aprendizaje colectivo donde la retroalimentación de un usuario afecte la consulta del siguiente y así sucesivamente. Cuando un usuario da su retroalimentación seleccionando las imágenes más relevantes a su consulta, las posiciones de esas imágenes en el vector de resultados se actualizan. Las imágenes relevantes irán avanzando cada vez más hacia las primeras posiciones con cada retroalimentación. El sistema recordará las nuevas posiciones y estas se respetarán en la siguiente consulta.

El enfoque de retroalimentación propuesto se describe a continuación. Después de que un usuario seleccione las imágenes relevantes a una consulta, esta información se procesa de la siguiente manera:

1. Para cada imagen relevante se revisa, en la base de datos, si ya existe una relación entre ella y la imagen consulta. En otras palabras, se revisa si la imagen ya ha sido seleccionada por otro usuario como relevante, dada la imagen

consulta correspondiente. Si la relación existe, su probabilidad de semejanza (peso) aumenta en una unidad. De esta manera, el peso nos indica el número de usuarios que han afirmado que ambas imágenes son parecidas. Si la relación no existe, se crea una nueva como una tupla con los identificadores de ambas imágenes y un peso con valor de 1 unidad.

2. Para cada imagen relevante, se actualiza su posición dentro del vector de resultados de acuerdo a la siguiente función de relevancia:

$$p_i = a_i - w_{iq}^{\,2}(a_i\beta)$$

Donde:

- p_i es la nueva posición dentro del vector de resultados de la imagen relevante *i*.
- a_i es la posición actual dentro del vector de resultados de la imagen relevante *i*.
- w_{iq} es la probabilidad de semejanza entre la imagen *i* y la imagen consulta.
- β es una constante > 0.
- Si p_i resulta < 0, su valor será entonces 0, ya que el vector de resultados no tiene posiciones negativas.

3. Para cada imagen relevante, se guarda, en la base de datos, la nueva posición que debe tener dentro del vector de resultados para la consulta correspondiente.

El origen de la función anterior es heurístico y se basa en el objetivo de tener un aprendizaje colectivo. Lo que se busca es que una imagen que ha sido señalada como relevante por cinco usuarios, dada la imagen consulta *q*, quede en la primera posición del vector de resultados. Se ha determinado que sean cinco los usuarios los que deban corroborar dicha semejanza, porque es un número razonable de iteraciones para que una imagen que debe ser la más relevante a una consulta, aparezca como tal. Se considera que esta cantidad también evita que el criterio de una sola persona determine la relevancia de una imagen, y es obvio que distintos usuarios seguramente tendrán distintos opiniones al respecto. Para obtener la función de relevancia, se tomaron en cuenta los factores que, en este sistema, intervienen en la retroalimentación, que son:

- La posición natural (dada por el algoritmo) de una imagen con respecto a una consulta.
- El número de usuarios que han ratificado la semejanza entre una imagen resultante y su consulta (peso).

Se realizaron experimentos con estos factores, resultando en diferentes funciones de relevancia. Finalmente, se llegó a la función actual debido a que cumplía bastante bien con el objetivo mencionado anteriormente. También se observó que la función obtiene resultados aceptables cuando el valor de la constante Beta es 0.03. La idea de llevar un registro del número de usuarios que han marcado una imagen como relevante se tomó del trabajo realizado por [Lu et. al 2000]. Estos autores relacionan una imagen con palabras clave hechas al momento de la consulta. A esta relación se le coloca un peso, similar a lo que se hace en esta implementación, el cual indica el número de usuarios que han

señalado dicha relación. También consideran que una palabra clave que esté ligada a muchas imágenes debe ser penalizada y por ello calculan su relevancia como:

$$r_k = w_k(\log_2 \frac{M}{d_k} + 1)$$

Donde:

- r_k es la relevancia de la palabra clave k
- w_k es el peso de la relación
- M es el número total de imágenes en la base de datos
- d_k es el número de imágenes con las cuales la palabra clave k tiene relación

Los autores también consideran que a la hora de realizar la retroalimentación, el usuario debe seleccionar tanto imágenes relevantes como imágenes no relevantes. Para las imágenes relevantes, el peso de su relación con una palabra clave se aumenta en una unidad. Para las palabras no relevantes, el peso se disminuye dividiéndolo entre 4 y eliminándolo si es menor a una unidad. Para calcular la nueva posición de una imagen resultante, utilizan una complicada modificación de la fórmula de Rocchio, que a su vez utiliza parámetros generados por otras funciones que por simplicidad, no se detallan en este documento (para más detalles, consultar la sección 3.2 de [Lu et. al 2000]). Debido a que los autores usan factores que no están disponibles en esta aplicación, no fue posible implementar su función de relevancia.

2.8 Formato de las imágenes

Debido a que el proyecto se desarrolla en un contexto digital, es importante definir el formato de las imágenes sobre las cuales vamos a trabajar. Estos formatos serán: JPEG (*Joint Photographic Expert Group*), GIF (*Graphics Interchange Format*) y PNG (*Portable Network Graphics*). La razón para usar estos formatos se debe a que el dominio de aplicación del proyecto, el cual se detallará más adelante, lo requiere.

El primer formato es un método de compresión en el cual se pierden ciertos detalles de la imagen original. A este tipo de compresión se llama compresión con pérdida (*lossy compression*). El segundo formato, a diferencia del anterior, utiliza un método de compresión que genera pocas pérdidas de información con respecto a la imagen original y es un formato popular entre la mayoría de las aplicaciones. El último formato fue creado por el consorcio W3C con el objetivo de tener un formato similar al GIF pero de uso abierto. Una de sus ventajas interesantes es que puede alcanzar una compresión mayor al formato GIF hasta en un 30 por ciento. La información anterior fue tomada de la investigación hecha por [Rodríguez 2001]. Debido a que el propósito de esta investigación no está enfocado a formatos digitales de imágenes, no se dan todos los detalles técnicos. Para obtener mayor información al respecto se puede consultar la referencia descrita.

2.9 CBIR en U-DL-A

Como se mencionó en el capítulo anterior, el objetivo de este proyecto consiste en integrar una herramienta de búsqueda de imágenes por contenido al proyecto de

Bibliotecas Digitales de nuestra universidad. Por ello, se realizó un análisis de las colecciones digitales que podrían ser utilizadas por nuestra aplicación.

En un principio, se pensó hacer uso de imágenes que pertenecieran a la colección de libros en la Biblioteca Franciscana. Sin embargo, se llegó a la conclusión de no era factible por dos razones principales: La primera fue que el número de imágenes era insuficiente debido a que los libros de esa época las contenían en pequeñas cantidades o no las tenían en lo absoluto. La segunda razón fue la dificultad del proceso de recopilación de dichas imágenes, ya que no existe un catálogo que liste aquellos libros que contienen imágenes, ni la ubicación de estas dentro de su respectivo documento. Por todo esto, el proceso de recopilación de imágenes en esta colección resultaría muy costoso y está fuera del alcance de los recursos con los que cuenta este proyecto.

Más adelante, surgió la idea de utilizar las imágenes de la colección de tesis digitales. Esta opción fue más viable debido a que existe un repositorio bien estructurado de tesis en formato digital y que forma parte de los servicios que proporciona U-DL-A. Nos referimos al sistema Tales [Fernández y Sánchez 2003], el cual permite el almacenamiento de las tesis sometidas por los estudiantes de la UDLA, así como búsquedas sobre estos documentos vía una interfaz Web. Usar el sistema Tales nos proporciona ciertas ventajas en el sentido de que las imágenes de las tesis son aisladas y almacenadas en una base de datos, listas para ser utilizadas en búsquedas por contenido. Estas imágenes son almacenadas en sus formatos originales, los cuales son determinados por los procesadores de texto empleados para redactar dichos documentos. Estos formatos son: JPG, GIF y PNG. Recordemos que en la sección 2.8 se definieron estos mismos formatos de imágenes para su uso en este sistema. Con estos mecanismos

automáticos, se ahorra tiempo en la recopilación de las imágenes, a diferencia de la colección de la Biblioteca Franciscana. Otra de las ventajas importantes que nos brinda Tales es de tipo práctico, debido a que es una fuente de información importante para la investigación y por ello le daría un sentido de utilidad a nuestro proyecto.

Con Tales como una base importante para este proyecto, es tiempo de definir claramente las funciones que desempeña nuestra aplicación. Básicamente, sus usuarios pueden hacer búsquedas por contenido sobre la colección de imágenes de tesis digitales y visualizar los documentos fuente de aquellas imágenes que resulten de la búsqueda. Para efectuar las consultas, se usa una combinación del enfoque *query-by-visual-example* (visto en el capítulo anterior) y búsqueda textual. Esta última se efectuará sobre la descripción de las imágenes, la cual se almacena automáticamente por Tales para cada imagen en una tesis. El objetivo de la búsqueda textual es obtener un subconjunto de imágenes cuyas descripciones contengan la(s) palabra(s) clave dadas por el usuario al momento de realizar la consulta. Posteriormente, se realiza la búsqueda por contenido sobre las imágenes resultantes de la búsqueda textual, similar al proceso de búsqueda llevado a cabo en el sistema Blobworld (sección 2.4). La búsqueda por contenido se realiza mediante el enfoque BIC visto en la sección 2.5.3 debido a que resultó ser un algoritmo eficiente (ver la sección 2.6) y fácil de implementar. Una vez realizada la búsqueda, las imágenes resultantes se presentan en orden ascendente con respecto a su similitud (distancia) con la consulta. El usuario puede dar retroalimentación al sistema señalando las imágenes resultantes que, según su criterio, tienen mayor parecido con la imagen consulta. Como resultado de la retroalimentación, los resultados serán reordenados, con las imágenes elegidas por el usuario apareciendo en posiciones más

cercanas a la consulta. Finalmente, el usuario puede visualizar el documento fuente, en este caso la tesis, de una imagen seleccionada del conjunto de resultados. Con esto se pretende dar un resumen de las funcionalidades de la aplicación que se desarrolló. Todos los detalles de los procesos mencionados se darán en el Capítulo 4.