

REPORTE TÉCNICO
**Reconocimiento
de Patrones**

Estado actual de la indexación en espacios métricos aplicado al reconocimiento de rostros

Ing. Ricardo Enrique Serrano Callaba,
Dr. C. José Hernández Palancar

RT_030

abril 2010





CENATAV

Centro de Aplicaciones de
Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

RNPS No. 2142
ISSN 2072-6287
Versión Digital

SERIE AZUL

REPORTE TÉCNICO
**Reconocimiento
de Patrones**

**Estado actual de la indexación en es-
pacios métricos aplicado al recono-
cimiento de rostros**

Ing. Ricardo Enrique Serrano Callaba,
Dr. C. José Hernández Palancar

RT_030

abril 2010



Índice

1	Introducción.....	1
2	Aspectos introductorios a las búsquedas por proximidad en espacios métricos	3
2.1	Características fundamentales de los espacios métricos	6
2.2	Funciones de distancias en los espacios métricos	7
2.3	Procesamiento de consultas similares en espacios métricos	8
3	Resumen de las Técnicas de Indexación en espacios métricos.....	9
3.1	Clasificación general de las estructuras de indexación	9
3.1.1	Estructuras de Indexación utilizando algoritmos de pivotes	10
3.1.2	Estructuras de indexación utilizando algoritmos de agrupamiento	11
3.2	M-Tree	12
3.2.1	Procesamiento de consultas similares en el M-Tree.....	13
3.2.2	Aspectos de la construcción y actualización de M-Tree	19
3.3	Paralelismo aplicado a las estructuras de indexación.....	23
4	Estructuras de indexación aplicadas al reconocimiento de rostros	29
4.1	Aspectos generales del reconocimiento de rostros.....	29
4.2	Ubicación de las imágenes de rostros en estructuras de indexación métricas.....	31
4.2.1	Descriptor de imágenes de rostros.....	32
4.2.2	Algoritmo SIFT como descriptor local.....	33
4.2.3	Algoritmo SIFT en el reconocimiento de rostros	34
4.2.4	Algoritmo SURF en el reconocimiento de rostros	37
4.3	Principales técnicas de indexación aplicadas a imágenes de rostros.....	38
4.3.1	Resumen de los esquemas de indexación de rostros	38
4.4	Aplicación de los métodos de accesos métricos (MAM) en el reconocimiento de rostros	39
4.5	Resumen de las principales métricas utilizadas en el reconocimiento de objetos biométricos .	41
4.6	Estructuras de grafos aplicadas al reconocimiento de rostros	42
5	Conclusiones.....	43
	Referencias bibliográficas	44

Estado actual de la indexación en espacios métricos aplicado al reconocimiento de rostros

Ing. Ricardo Enrique Serrano Callaba, Dr. C. José Hernández Palancar

Centro de Aplicaciones de Tecnología Avanzada
7ª #21812 e/ 218 y 222, Siboney, Playa, C.P 12200, Ciudad de La Habana, Cuba
rserrano@cenatav.co.cu

RT_030 CENATAV

Fecha del camera ready: 26 de marzo de 2010

Resumen: Las estructuras de indexación aplicadas a espacios métricos, constituyen uno de los aspectos más investigados en el campo de la aceleración de consultas en sistemas de bases de datos multimediales, debido a su capacidad de organizar grandes volúmenes de objetos de una gran complejidad estructural. A pesar de tener gran aplicación en sistemas tales como recuperación de imágenes por contenido, recuperación de documentos, textos, biología computacional entre otras, su utilización en el proceso de reconocimiento de rostros no es muy abordada en la literatura actual. Debido a esto el objetivo fundamental de este trabajo es mostrar las principales ventajas y desventajas de las estructuras de indexación en espacios métricos para organizar grandes bases de datos de imágenes de rostros, con el propósito de acelerar la ejecución de algoritmos de reconocimiento por verificación, tratando de reducir la probabilidad de aparición de falsos positivos.

Palabras clave: indexación en espacios métricos, consultas de similitud, reconocimiento de rostros, SIFT, SURF

Abstract: Indexing structures applied to metric spaces, is one of the most researched aspects in the field of query acceleration in multimedia databases systems, due to its ability to organize large volumes of objects having high structural complexity. Despite having wide applications in systems such as content-based image retrieval, document and text retrieval, including computational biology, its use in the process of face recognition is not addressed in current literature. Because of this, the fundamental objective of this paper is to show the main advantages and disadvantages of indexing structures in metric spaces to organize large face images databases, with the aim of accelerating the execution of verification algorithms, avoiding the occurrence of false positives.

Keywords: Indexing in Metric Space, Similarity Queries, Face Recognition, SIFT, SURF

1 Introducción

El aumento acelerado de las tecnologías de la información ha traído aparejado un crecimiento de la capacidad de almacenamiento de las bases de datos de objetos multimediales. Este tipo de bases de datos contienen objetos del mundo real tales como (voz, audio, imágenes, ADN, etc.).

En las bases de datos multimediales la ejecución de consultas no se realiza de la forma tradicional por la complejidad estructural de estos objetos, debido a esto se definen los términos de búsquedas por proximidad. Las búsquedas por proximidad tienen como objetivo recuperar en tiempo de ejecución de una consulta los elementos más cercanos a un elemento de consulta dado. Esta proximidad se expresa mediante la evaluación de una distancia que permita definir este concepto. En una base de datos de alrededor de millones de objetos, pensar en la evaluación secuencial de esta distancia sería ineficiente por lo que una de las variantes más utilizadas son los índices. Un índice no es más que una abstracción de la base de datos que permite acelerar la ejecución de consultas de similitud [6].

Existen dos formas fundamentales de aplicar índices, en la primera de ellas los objetos son modelados como vectores en múltiples dimensiones y después tratados como índices para

espacios vectoriales. En grandes bases de datos estos índices tienden a perder su efectividad, debido al aumento de las dimensiones de los datos, esto es conocido como “curse of dimensionality” o simplemente maldición de las dimensiones [1]. Debido a esta dificultad se proponen los índices para espacios métricos, pues los mismos no tienen en cuenta las coordenadas de cada vector, solamente distancia relativa entre los objetos.

Por otra parte el reconocimiento de personas por su rostro se efectúa en grandes bases de datos de alrededor de millones de imágenes de rostros, por tanto obtener los resultados con gran eficiencia y precisión es el principal objetivo de las aplicaciones dedicadas a este tipo de reconocimiento para poder ser empleadas en tiempo real.

De forma general este trabajo tiene como objetivo realizar un resumen de las principales estructuras de indexación ubicadas en espacios métricos, desde el punto de vista de sus capacidades para organizar grandes cantidades de objetos multimediales. Por otra parte se propone la vinculación de estas estructuras de indexación en la aceleración del proceso de reconocimiento de rostros. La figura 1 brinda una perspectiva general de este trabajo.

El trabajo tiene la siguiente estructura: en el primer capítulo se realiza una introducción a los aspectos más generales de los espacios métricos tales como búsquedas por proximidad, procesamiento de consultas similares y las principales métricas que se utilizan en este espacio. El segundo capítulo está dedicado a las principales estructuras de indexación aplicadas a espacios métricos, donde se realizará una clasificación general de estas estructuras, mostrando sus principales características para organizar grandes cantidades de objetos multimediales. El tercer capítulo tiene como objetivo mostrar las principales características del proceso de reconocimiento de rostros que permitan definir la aplicación de las estructuras de indexación en espacios métricos dentro del campo de la biometría, en específico en el reconocimiento automático de personas por su rostro. Para lograr este objetivo nos apoyaremos en las principales formas que existen para representar el rostro humano para el proceso de reconocimiento haciendo énfasis en los populares descriptores locales SIFT [37], SURF [38] y en las principales medidas de similitud definidas en estos espacios.

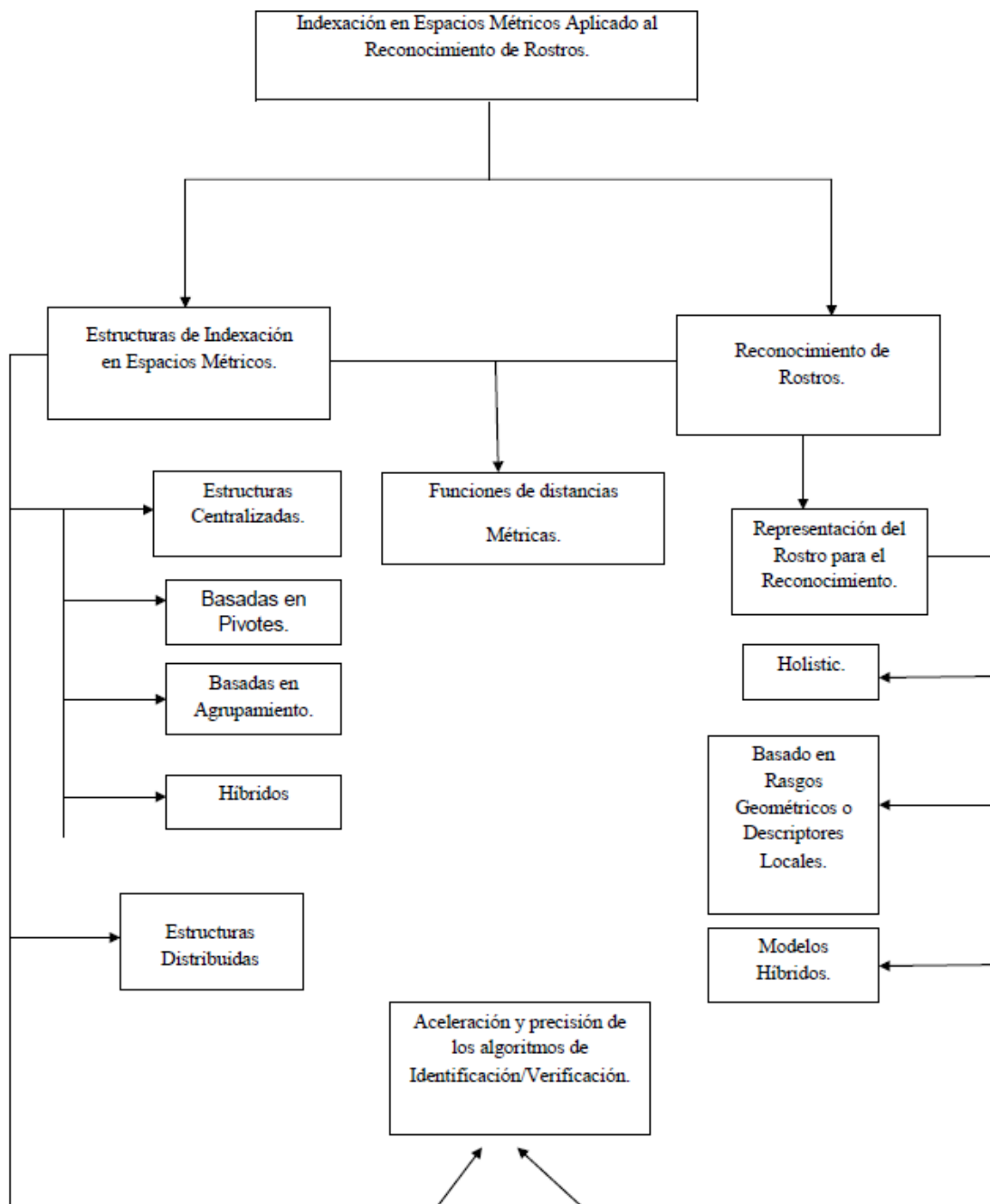


Fig. 1. Taxonomía general del trabajo

2 Aspectos introductorios a las búsquedas por proximidad en espacios métricos

El desarrollo vertiginoso de las tecnologías de la información, ha traído consigo un aumento de los sistemas que utilicen, almacenen y devuelvan objetos de una gran complejidad computacional [1,2,4,5]. Esta complejidad está referida a las características estructurales de estos objetos debido a ello los sistemas tradicionales de bases de datos no tienen forma de definir relaciones entre los mismos. Las bases de datos multimediales constituyen el principal medio de

almacenamiento de estos objetos. Una base de datos multimedial es aquella capaz de almacenar y devolver objetos del mundo real tales como (imágenes, voz, video entre otros) [1,5].

En la actualidad las búsquedas por proximidad son aplicadas en varias aplicaciones entre ellas se destacan:

- Recuperación de la información [1].
- Visión por computadora [47].
- Biología Computacional [45].

Recuperación de la Información: Este tipo de aplicaciones se puede dividir en varios tipos de campos como: Recuperación de Documentos, Textos e Imágenes por Contenidos, donde una función de distancia es útil para medir similitud entre un objeto de consulta y el resto de los objetos almacenados en la base de datos. Este tipo de búsqueda es útil en aplicaciones Web como los buscadores, donde se obtienen documentos e imágenes que sean similares a las consultas realizadas por los usuarios.

Visión por computadoras: En este tipo de aplicaciones las funciones de distancias son utilizadas para brindarles a los sistemas automatizados la posibilidad de reconocer personas mediante huellas digitales, iris, rostros entre otros. Las aplicaciones que utilizan este criterio son empleadas en sistemas de seguridad donde la identidad de las personas es vital para efectuar control de accesos a determinados lugares tales como: espacios virtuales, terminales de aeropuertos, zonas militares entre otros.

Biología Computacional: En este campo se utiliza el criterio de la distancia para medir similitud entre cadenas biológicas tales como: secuencias de proteínas, ADN entre otras. Su principal aplicación se encuentra en el marco de las bases de datos biomédicas y genéticas, con el objetivo de identificar o clasificar determinadas secuencias de consultas con respecto a secuencias conocidas almacenadas en una base de datos [45].

Existen dos formas de desarrollar estas búsquedas por proximidad atendiendo a la precisión con que se obtienen los resultados.

En [5] los autores plantean que las búsquedas similares se pueden clasificar en dos grupos fundamentales:

- Búsquedas similares exactas.
- Búsquedas similares aproximadas.

Las búsquedas similares exactas suponen encontrar en una colección de datos todos los elementos relevantes a uno de consulta dado, o sea, no se permiten falsos positivos debido a que elementos irrelevantes no deben aparecer en el resultado de una consulta.

Las búsquedas similares aproximadas no son tan estrictas en cuanto a la precisión de los resultados, pequeñas cantidades de falsos positivos se permiten en el resultado de una consulta. El objetivo de estas búsquedas es disminuir el tiempo de respuesta de las búsquedas exactas. Por tanto, mediante este tipo de búsquedas se gana en velocidad en tiempo de ejecución de una consulta pero se paga un precio en cuanto a la precisión de los resultados. Generalmente el objetivo de estas búsquedas es brindarles a los usuarios la posibilidad de limitar los resultados de la ejecución de una consulta [5]. Un ejemplo de esto lo encontramos en los sistemas de recuperación por contenido de imágenes, cadenas de textos, entre otros, donde el usuario puede limitar la búsqueda cuando ya obtuvo el resultado deseado.

En el caso de las bases de datos con altas dimensiones, con el objetivo de efectuar procesos como el reconocimiento e identificación de personas por sus rasgos más significativos: tales como: voz, huellas digitales, rostros entre otros, es impensable utilizar el concepto de búsquedas similares aproximadas, pues la aparición de pequeñas cantidades de falsos positivos implicaría autenticar erróneamente a una persona.

Como se explicó anteriormente la similitud entre dos elementos se define mediante la evaluación de una función distancia que permita definir cuan similares son dos objetos [1,2]. Mientras menor sea esta distancia mayor será la similitud entre estos objetos. Por lo tanto la ejecución de un algoritmo de búsquedas por proximidad, consistiría en evaluar secuencialmente las distancias relativas entre un elemento de consulta y una base de datos que almacene objetos multimediales. El concepto de similitud constituye la principal diferencia entre los sistemas de

base de datos tradicionales y los multimediales, donde la relación que se establece en los primeros entre las llaves y los atributos alfa-numéricos es útil para la ejecución de las consultas, en las bases de datos multimediales las relaciones que se establecen son relativas a las distancias entre objetos [1,5].

En grandes volúmenes de datos pensar una evaluación secuencial de esta distancia sería ineficiente. Por tanto un mecanismo eficiente sería devolver los elementos más relevantes a una consulta disminuyendo el número de accesos a la información, para evitar de esta forma el cálculo innecesario de distancias y disminuir los costos computacionales C/C que se derivan del cálculo de distancias y de acceso a la información I/O. La principal solución para optimizar las estructuras de búsquedas por proximidad sería, la utilización de índices que permitan devolver los objetos más relevantes a uno de consulta dado, tratando de calcular el menor número de distancias posibles. En la bibliografía el concepto de índice es muy variado y utilizado pero de forma general se puede resumir de la siguiente forma. Un índice en este entorno sería una abstracción de la base de datos que permita acelerar la ejecución de los algoritmos de búsquedas por proximidad [1]. De igual forma en [45] los autores definen los índices como una estructura que permita minimizar los requerimientos de almacenamiento de los datos y los tiempos necesarios para la recuperación de la información relevante a un elemento de consulta.

Existen dos formas fundamentales de aplicar índices para aumentar la velocidad de las búsquedas en bases de datos de objetos multimediales:

- Índices aplicados a espacios vectoriales.
- Índices aplicados a espacios métricos.

Muchos objetos que se almacenan en bases de datos multimediales son representados como objetos en múltiples dimensiones del espacio y luego representados como índices para espacios vectoriales. Estos índices utilizan las coordenadas de cada vector pero se degradan rápidamente con el aumento de las dimensiones del espacio, esto es conocido como “maldición de las dimensiones” o curse of dimensionality[1]. La maldición de las dimensiones está estrictamente relacionado a las bajas disimilitudes entre un elemento de consulta dado y los objetos almacenados en una base de datos. Esto implica que cuando aumenten las dimensiones del espacio a valores próximos a 20 dimensiones, un algoritmo de ejecución de una consulta tendría que recorrer la base de datos prácticamente completa para devolver el resultado deseado [2]. Una variante para evitar la maldición de las dimensiones lo constituyen los índices que se aplican a espacios métricos.

En [1] los autores plantean que los índices aplicados a espacios vectoriales presentan una mejor ejecución que los índices aplicados a espacios métricos, cuando el espacio presenta pocas coordenadas, sin embargo, cuando crecen las dimensiones del espacio esto se complica un poco más debido a que cualquier algoritmo que se aplique a un espacio vectorial tiene una exponencial dependencia de las dimensiones del espacio. Por lo tanto la aplicación de una consulta en un espacio vectorial pudiera verse afectado por el recorrido completo de la base de datos.

En [1] los autores afirma que una ventaja de ver un espacio vectorial como un espacio métrico es que la ejecución de los índices tan solo depende de lo intrínseco de sus dimensiones, en vez de la representación de sus coordenadas, sin necesidad de ningún mapeo entre los elementos representados.

La definición de intrínseco en un espacio métrico expresa que la relación que se establece entre los objetos sólo está basada en sus distancias relativas en vez de la representación de sus coordenadas en el espacio.

Los índices aplicados a espacios métricos a pesar de no verse afectados directamente por la maldición de las dimensiones, si presentan una desventaja fundamental cuando aumentan las dimensiones del espacio y es que los histogramas de distancias tiende a estar muy concentrados, aumentando la media de los mismos μ y disminuyendo su varianza σ^2 e impiden de esta forma el descarte de objetos, debido a que las distancias de los objetos con respecto a un objeto de consulta son muy similares[5].

Existen muchas medidas para medir el grado de dimensionalidad intrínseca en estos histogramas de distancias en espacios métricos. Una de las más precisas fue desarrollada en [1] como $\rho = \frac{\mu^2}{2\pi^2}$ con un factor constante entre 1.0 y 1.5. Esta medida depende en gran medida del tipo de métrica utilizada.

2.1 Características fundamentales de los espacios métricos

Los espacios métricos de forma general se caracterizan por la existencia de una colección de objetos los cuales están relacionados por la existencia de una función de distancia para evaluar la similitud o disimilitud que exista entre ellos [1,2,3,4,5].

Las características fundamentales de un espacio métrico están dadas por las principales propiedades que debe cumplir la función de distancia definida entre objetos. Un espacio métrico $M(D,d)$ está compuesto por una colección de objetos D donde por cada par de ellos existe una función de distancia d asociada [1,2,3,4,5]. La función distancia debe cumplir lo siguiente:

$\forall x,y \in D, d(x,y) \geq 0$	No negatividad.
$\forall x,y \in D, d(x,y) = d(y,x)$	Simetría.
$\forall x,y,z \in D, d(x,z) \leq d(x,y) + d(y,z)$	Desigualdad Triangular.

A diferencia de los espacios vectoriales donde la relación de los objetos está definida por la representación de estos en el espacio mediante sus coordenadas, en los espacios métricos la única relación que existe entre ellos está delimitada por una función distancia que permita precisar cuan similares son dos objetos determinados. Esta similitud se expresa como la menor distancia que exista entre estos elementos.

Las búsquedas similares en espacios métricos tienen como objetivo buscar los elementos más próximos a un elemento de consulta, a través de las distancias medidas entre todos los elementos de la base de datos y el elemento de consulta dado, esta distancia medida tiene como objetivo cuantificar cuan próximos están los objetos de una base de datos con respecto al elemento de consulta [5].

Los algoritmos de búsquedas similares para hacer frente al procesamiento de consultas deben garantizar devolver la información con un alto grado de eficiencia y precisión. La eficiencia de estos algoritmos se traduce como el tiempo que se emplea en devolver la información relevante, en [1] se plantea que debe estar en el orden de los milisegundos. La precisión no es más que % de elementos relevantes que deben aparecer en el resultado de una consulta y que debe ser tan alto como sea posible con el objetivo de eliminar el número de falsos rechazos o la cantidad de elementos relevantes descartados en el procesamiento de una consulta.

Lo anteriormente dicho no presenta grandes problemas en bases de datos de un tamaño aceptable, donde con una búsqueda secuencial es suficiente para obtener los resultados deseados en poco tiempo y con un % prácticamente nulo de elementos relevantes a una consulta descartados.

Sin embargo, esta situación se torna un poco más compleja en bases de datos de altas dimensiones, donde el número de elementos puede estar en orden de millones, por lo que aplicar una búsqueda secuencial sería impensable [1]. Por lo tanto, una solución factible es utilizar los índices aplicados a espacios métricos con el objetivo aumentar la velocidad del procesamiento de consultas, evitando el cálculo excesivo de distancias y el número innecesario de accesos a la información disminuyendo el costo computacional y los costos por acceso a la información respectivamente.

2.2 Funciones de distancias en los espacios métricos

Las funciones de distancias de los espacios métricos constituyen una forma de cuantificar la proximidad de los objetos en un dominio dado, por lo general son específicas de las aplicaciones o de algunas clases de posibles aplicaciones [1]. Dependiendo del carácter de los valores retornados, las funciones de distancias se pueden clasificar de dos formas:

- Discreta.
- Continúa.

Las funciones discretas son aquellas que devuelven únicamente una pequeña colección de valores. Mientras que las funciones continuas son aquellas que devuelven una colección de valores muy largos o infinitos. A continuación mencionamos varios tipos de distancias aplicadas en los espacios métricos de forma general:

Función de distancia Minkowski

Las funciones de distancias Minkowski forman una familia de funciones métricas nombradas como métricas L_p , debido a que los casos individuales dependen del parámetro numérico p . Estas funciones son definidas en vectores de n -dimensiones de números reales.

En esta familia de funciones métricas L_1 denota la función de distancia Manhattan también conocida como City-Block, L_2 denota la conocida función Euclidiana y L_∞ es conocida como distancia máxima o distancia infinita. Estas funciones son utilizadas en casos donde los vectores numéricos tienen coordenadas independientes [2].

Función de distancia basada en la forma cuadrática o (Quadratic Form)

Este tipo de función métrica se utiliza para medir la relación que existe entre los componentes individuales de los vectores. Ejemplo de esto lo constituyen los histogramas de colores de las imágenes, donde cada dimensión representa un color específico. En estos casos la función de distancia L_2 no refleja ninguna correlación en los rasgos de histogramas de colores. Esta función tiene gran utilidad en base de datos de imágenes, debido a la capacidad de reflejar las dependencias entre diferentes componentes de rasgos en histogramas diferentes. Esta función puede ser computacionalmente costosa, dependiendo del tamaño de los vectores. Los histogramas de colores de las imágenes contienen muchas dimensiones, pues sus vectores pueden alcanzar de 64 a 256 dimensiones [2].

Distancia de edición

Este tipo de función de distancia se utiliza para medir la similitud de secuencias de cadenas, también es conocida como distancia *Levenshtein*. La distancia entre dos cadenas se define como el número mínimo de operaciones atómicas *edit* tales como: inserciones, eliminaciones y actualizaciones de una cadena con respecto a otra. La función de distancia *edit* asigna medidas o sea números reales positivos para estas operaciones, debido a esto esta función de distancia entre una cadena X y otra cadena Y es el valor mínimo de las sumas de las medidas de las operaciones atómicas necesitadas para transformar la cadena X en la cadena Y [2].

Distancia de edición arbórea

Esta función define una distancia para medir la similitud de dos estructuras de árboles, como el costo mínimo necesario para convertir un árbol de entrada en otro ya seleccionado, utilizando operaciones *edit* para árboles, tales como inserciones o eliminaciones de nodos. Esta distancia tiene gran utilidad para medir la similitud estructural de objetos que se modelen como árboles un ejemplo de esto lo constituyen los documentos XML [2].

Distancia basada en el coeficiente de Jaccard

Esta función se aplica para medir la similitud de dos colecciones de datos (A y B). Dado dos colecciones de datos A y B esta función mediante la intersección y unión de las mismas. Un ejemplo de esto lo constituyen el comportamiento de las búsquedas de los usuarios en Internet, que puede ser expresada como la colección de sitios visitados y el coeficiente de Jaccard es utilizado para evaluar la similitud o disimilitud de los intereses de los usuarios por los sitios visitados [2].

En [2] los autores plantean que el cálculo de distancias constituye un proceso de una gran complejidad computacional en bases de datos de objetos multimediales, a diferencia de las

bases de datos tradicionales donde la comparación entre los objetos se realiza por medio de las llaves definidas en este entorno.

Las funciones de distancias constituyen el principal elemento para medir proximidad entre los objetos multimediales almacenados en una base de datos, por lo que su estudio en el campo de las búsquedas por proximidad constituye un aspecto de gran importancia.

2.3 Procesamiento de consultas similares en espacios métricos

Uno de los aspectos más importantes en un espacio métrico es la ejecución de consultas de similitud. Una consulta de similitud está definida explícita e implícitamente por un objeto de consulta q y una restricción en cuanto a la forma de extender la proximidad requerida, típicamente expresada como una distancia [2]. La respuesta de una consulta devuelve los elementos que satisfagan la condición de selección, presumiblemente los objetos que estén más próximos al objeto de consulta dado. La ejecución de consultas similares puede clasificarse fundamentalmente en dos formas [1,2,3, 4].

- Consulta por rango de similitud (Query Range).
- K-vecinos más cercanos (K-NN).

Consulta por rango de similitud (Query Range)

Probablemente sea el tipo de consulta de similitud más utilizada, la solicitud basada en el rango de consulta se denota como $R(q,r)$. La consulta está especificada por un objeto q con r como radio de restricción de la consulta. Por lo tanto la ejecución de esta consulta devuelve todos los objetos que estén comprendido dentro de su radio r , o sea, que la distancia del objeto al objeto de consulta sea menor que el radio de esta consulta. $R(q,r)=\{o \in X, d(o,q) \leq r\}$.

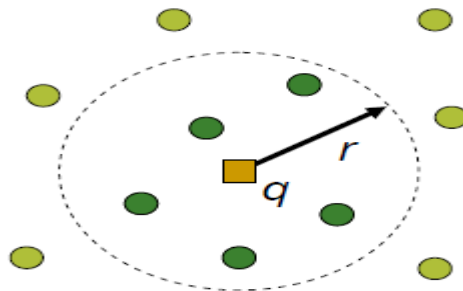


Fig. 2. Ejecución de la consulta de similitud Rango de consulta

La figura muestra como los objetos de color verde claro pueden ser descartados de la búsqueda por no estar comprendidos dentro del área de la consulta descrita por el radio de consulta en este caso r .

Consulta de tipo K-NN

Otro tipo de consulta de similitud que se utiliza es K-NN o consulta de los k vecinos más cercanos, con el objetivo de poder cuantificar los elementos más próximos a un elemento de consulta dado. O sea devolver los k elementos que estén más próximos a q . En este caso se devolverán los k objetos X cuya distancia al elemento de consulta sea menor que la distancia de otro objeto de referencia Y al elemento de consulta, siendo q el elemento de consulta.

$$kNN(q) = \{R \subseteq X, |R| = k \wedge \forall x \in R, y \in X - R : d\{q, x\} \leq d\{q, y\}\}.$$

Si en la ejecución de este algoritmo se presenta el caso que exista más de un elemento con la misma distancia a q , el algoritmo selecciona el elemento arbitrariamente.

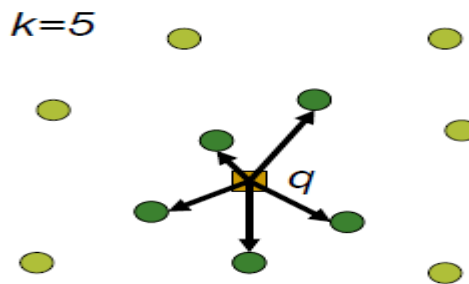


Fig. 3. Ejecución de una consulta similitud mediante K-NN

La figura muestra la ejecución del algoritmo de K-NN, siendo $k=5$. Lo que significara devolver los 5 elementos más cercanos a q en este caso serían los objetos de color verde oscuro.

En [5] los autores plantean que un algoritmo de búsqueda en una base de datos de N objetos debe realizar N evaluaciones de distancias, lo que resulta altamente costoso computacionalmente. Por lo que se hace necesario aplicar un mecanismo que permita obtener una abstracción de la base de datos para hacer frente a la ejecución de una consulta similar, con el objetivo de aceptar o descartar objetos sin necesidad de evaluar su distancia al elemento de consulta dado. Esto es lo que se conoce como un índice en una base de datos de objetos multimediales.

De igual forma en [2] los autores plantean que el alto costo computacional de las funciones de distancias ha determinado aplicar estructuras de indexación para minimizar el número de distancias evaluadas en una base de datos de grandes dimensiones.

En [1,4,5,6] de igual forma se afirma que los índices aplicados a espacios métricos son efectivos para acelerar la ejecución de los algoritmos de búsquedas similares en largas colecciones de datos. En la siguiente sección se realiza una descripción de las principales estructuras de indexación aplicadas a espacios métricos.

3 Resumen de las técnicas de indexación en espacios métricos

Con el aumento de las dimensiones en bases de datos que manejen objetos complejos, se hace muy costosa una evaluación secuencial de la distancia para clasificar objetos similares con respecto a un objeto de consulta dado, de ahí la importancia de aplicar estructuras de indexación en este tipo de búsquedas, debido que las mismas limitan el espacio de búsquedas, evitando de esta forma el cálculo de distancias innecesarias. La taxonomía mostrada en la figura 4 brinda un resumen de estas estructuras.

3.1 Clasificación general de las estructuras de indexación

En [2][5] los autores plantean que los métodos de indexación en espacios métricos se pueden clasificar en dos grupos fundamentales:

- Estructuras de Indexación utilizando algoritmos de pivotes.
- Estructuras de Indexación utilizando algoritmos de agrupamiento.

3.1.1 Estructuras de indexación utilizando algoritmos de pivotes

Estos algoritmos seleccionan un número de pivotes de una base de datos y clasifican todos los otros elementos acorde a sus respectivas distancias al pivote. Las distancias de los elementos al pivote y entre el objeto de consulta y los pivotes se utilizan para filtrar fuera de las búsquedas, elementos mediante la propiedad de desigualdad triangular de los espacios métricos. Estos algoritmos generalmente mejoran mientras aumente el número de pivotes, sin embargo, los requerimientos por la construcción del espacio aumentan. Por lo que un aspecto a tener en cuenta es la cantidad óptima de pivotes a utilizar.

Burkhard-Keller Trees(bk-trees)

Está diseñado para funciones discretas, ellos seleccionan un elemento de pivote p como la raíz del árbol y colocan los hijos como los elementos que estén más próximos al pivote. Cada subárbol es construido recursivamente utilizando la misma técnica hasta que exista un número de elementos determinado, en ese caso los elementos son almacenados en “bucket” en las hojas del árbol. Para descartar o reportar subárboles se utiliza la propiedad de desigualdad triangular de los espacios métricos [3].

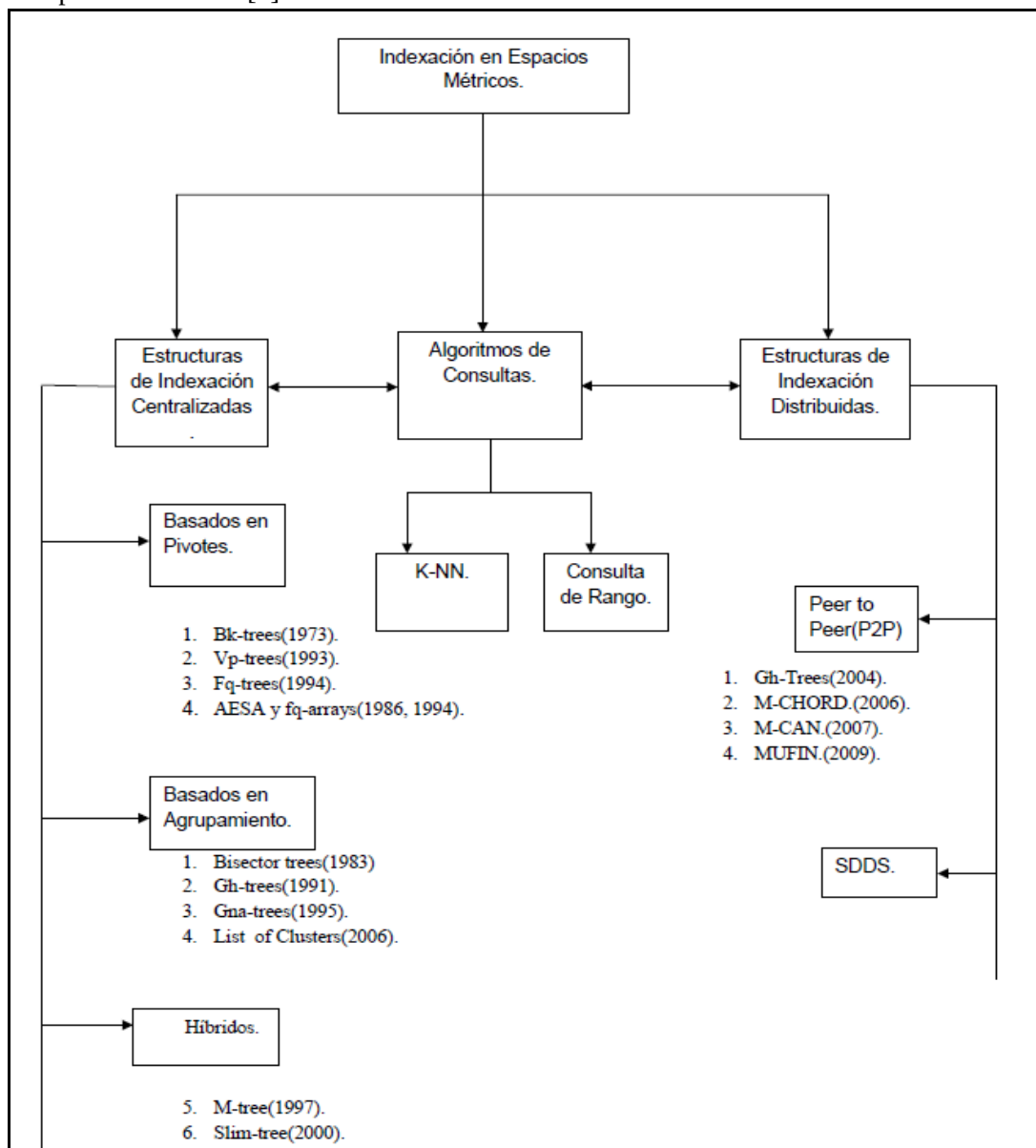


Fig. 4. Taxonomía de las principales estructuras de indexación

Fixed Queries Trees (fq-trees)

Constituye una evolución de la estructura anterior donde los mismos pivotes son usados para todos los nodos del mismo nivel del árbol. En este caso el pivote no necesita pertenecer al subárbol. Muchas comparaciones son salvadas en el proceso de backtracking debido a que solamente un pivote diferente existe por nivel. Existe una variante de este árbol llamada Fixed Height fq-tree(fhq-tree) donde las hojas están en la misma profundidad, independientemente del tamaño de los buckets [3].

Vantage Point Trees(vp-trees)

Está diseñado para funciones continuas de distancias. La raíz tiene dos subárboles de igual tamaño los cuales dividen la información en elementos próximos o más lejanos a la raíz del árbol [3].

Algoritmos basados en AESA y arreglos de consultas fijas (Fixed Queries Arrays-fq arrays)

Todas estas estructuras están basadas en una idea común, la utilización de k pivotes y cada objeto es asignado a una zona representada como un pivote, donde la distancia a este pivote es mínima con respecto al resto de los pivotes seleccionados. Después en la ejecución de una consulta de similitud, si el pivote mediante la propiedad de desigualdad triangular no está comprendido dentro del área de la consulta, todos los elementos pertenecientes al pivote serán filtrados fuera de la búsqueda.

Los algoritmos basados en pivotes pueden reducir el número de cálculos de distancias por el incremento del número de pivotes. Usando k pivotes es equivalente a descartar k elementos donde se cumpla $D(q,u) > r$, o sea que la distancia entre el objeto de consulta y el pivote sea mayor que el valor del radio de consulta[1].

3.1.2 Estructuras de indexación utilizando algoritmos de agrupamiento

Las estructuras de indexación comprendidas en esta clasificación tienen la principal característica de dividir la colección de datos en zonas espaciales, las cuales son tan compactas como sea posible y son capaces de descartar zonas completas por la evaluación de pocas distancias. La distancia se mide desde el objeto de consulta al centro de la zona, este centro está constituido por un elemento que es seleccionado aleatoriamente [3].

Generalized hyperplane Trees (gh-trees.)

Este algoritmo utiliza dos centros por cada nodo del árbol y divide el espacio acorde a cuál de los centros está más próximo a cada objeto. En tiempo de ejecución de la consulta, el algoritmo entra al subárbol que se interseque con el elemento de consulta [3].

Bisector Trees

Es similar al esquema anterior pero las zonas no están definidas acorde a cual tiene los centros más próximos sino que usan el concepto de radio de cobertura, que no es más que el mínimo radio necesario para contener todos los elementos de la zona, y los elementos son insertados en los subárboles tratando de minimizar el radio de cobertura de la zona. Este árbol es mejorado por el Voronoi Trees o v-trees [3].

Geometric Near-Neighbor (gna-trees)

Gna-trees está extendido a una partición m -ary en el gnat-trees el cual realiza una partición del espacio acorde a Voronoi[1] entre los m pivotes para cada nodo del árbol. Sin embargo, el criterio utilizado por el gna-trees para podar la búsqueda sólo está limitado al radio de cobertura [3].

En [2,3,4,5] los autores plantean que los métodos anteriormente planteados son intrínsecamente estáticos y no están preparados para cuestiones relacionadas con eliminaciones e inserciones producto de la actualización de la base de datos. Además son diseñados para memoria principal, por lo que no se pueden utilizar para memoria secundaria.

Por ese motivo surgió el M-Tree[6] que es un árbol dinámico, paginado y balanceado capaz de almacenar la información en memoria secundaria y que está diseñado para las continuas

actualizaciones realizadas en la base de datos [2,6]. El M-Tree constituye el primer método de acceso métrico (MAM) conocido [6]. Un método de acceso métrico tiene la capacidad de organizar una base de datos de objetos, en un ambiente dinámico teniendo en cuenta solamente la información de la distancia que se define entre los objetos, la cual satisface las tres reglas de los espacios métricos, Simetría, No negatividad y Desigualdad triangular[9]. La eficiencia de los MAM queda demostrada de la siguiente forma, debido al aumento de la cantidad de objetos que impide mantener en memoria principal todos los objetos existentes en la ejecución de una consulta de similitud, los MAM utilizan memoria secundaria para almacenar estas estructuras de indexación, en páginas de tamaño fijo de disco, y el acceso a esta información expresaría la cantidad de accesos necesarios en la ejecución de una consulta similar o la hora de insertar o devolver elementos [3,2].

3.2 M-Tree

El M-Tree es por naturaleza un árbol dinámico y balanceado, capaz de organizar la información en una estructura de indexación que se almacena en disco [2]. El M-Tree por su forma de construcción contiene en los nodos hojas de los objetos almacenados o referencias a ellos. Los nodos internos del M-Tree contienen punteros a subárboles, en los que se encuentran los objetos de la base de datos.

- $Rout(O_i) = [O_i, ptr(T(O_i)), rO_i, d(O_i, Par(O_i))]$.
- $grnd(O_i) = [O_i, oid(O_i), d(O_i, Par(O_i))]$.

La primera estructura muestra un nodo interno o de ruta, donde O_i contiene la información del objeto de ruta, $ptr(T(O_i))$ es un puntero al subárbol $T(O_i)$ llamado subárbol de cobertura de O_i , rO_i es el radio de cobertura del subárbol $T(O_i)$ y $d(O_i, Par(O_i))$ constituye la distancia del nodo O_i a su nodo padre, donde $d(O_i, Par(O_i))$ constituye un índice que se almacena, evitando de esta forma la evaluación innecesaria de la distancia en el procesamiento de una consulta de similitud.

La segunda estructura muestra la información de los objetos de la base de datos que se almacena en los nodos hojas, donde O_i es el objeto, $oid(O_i)$ constituye su identificador y $d(O_i, Par(O_i))$ es la distancia a su padre en este caso el nodo de ruta $Rout(O_i)$.

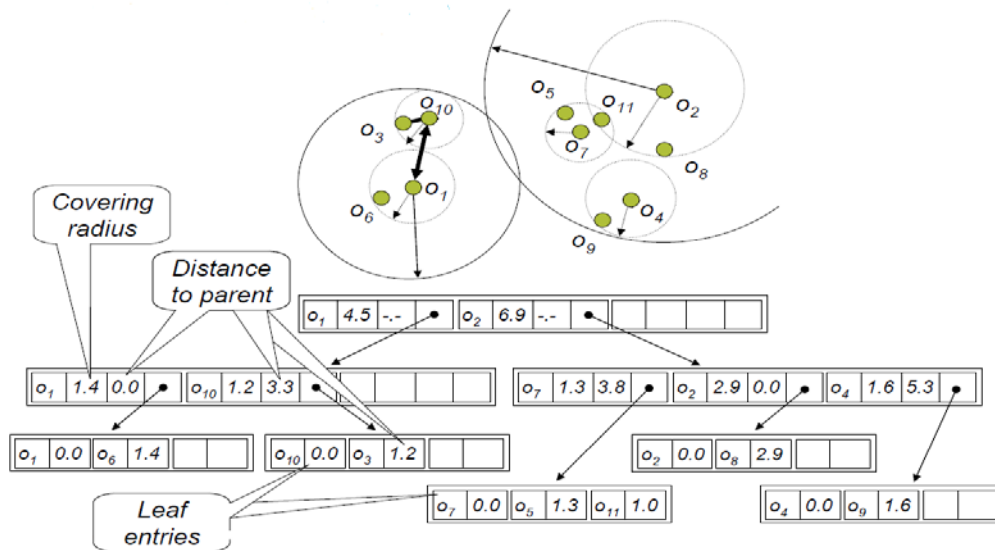


Fig. 5. Estructura jerárquica del M-Tree

El M-Tree divide los objetos en base a sus distancias relativas, esta distancia es calculada a partir de una métrica, y almacena estos objetos en nodos de tamaño fijo que corresponden a regiones cerradas de un espacio métrico. En la figura 5 se puede apreciar la forma en que es construido el M-Tree donde O_1 y O_2 constituyen raíces del árbol que a la vez describen dos regiones métricas, con varios objetos comprendidos. En la figura se observa como la región descrita por O_1 está compuesta por dos nodos de ruta O_1 y O_{10} que igualmente describen dos regiones con radio de cobertura 1.4 y 1.2 respectivamente. La región O_1 contiene en su interior los nodos hojas O_1 y O_6 , en este caso se puede apreciar como la distancia del nodo hoja O_1 a su padre es cero debido a que el nodo hoja constituye el mismo nodo, que por aspectos de construcción de la estructura presenta los roles de objeto de ruta y nodo hoja; mientras que la distancia de O_6 a O_1 que constituye su nodo padre es 1.4, siendo la entrada O_6 frontera de la región debido a que constituye el objeto más alejado del centro de la zona, donde la distancia del centro de la región al objeto frontera constituye el radio de cobertura. La entrada de ruta O_{10} comprendida en la región O_1 describe otra región métrica en el que se almacenan los objetos de la base de datos O_{10} y O_3 . Debido a la forma que describe esta estructura los algoritmos de consultas aplicados recorren el árbol recursivamente desde la raíz a las hojas en donde se encuentran los objetos de la base de datos almacenados. Las distancias de los nodos de ruta y hojas a sus nodos padres constituyen índices que impiden la evaluación de distancias innecesarias en el procesamiento de una consulta de similitud.

3.2.1 Procesamiento de consultas similares en el M-Tree

El principal objetivo de esta estructura es brindar un eficiente soporte a una base de datos de altas dimensiones, para hacer frente al procesamiento de consultas de similitud, con el objetivo de devolver la información en el menor tiempo posible evitando la evaluación de distancias innecesarias y tratando de minimizar los costos por acceso a disco [2]. En [6] se afirma que el M-Tree es la primera estructura de indexación jerárquica capaz de organizar largas colecciones de datos y a diferencia de las anteriores estructuras no sólo presta atención al C/C derivado del cómputo de distancias sino que también tiene muy presente el I/O, costo derivado por acceso a disco en la ejecución de una consulta de similitud. De forma general en el M-Tree se implementan dos tipos de consultas de similitud, Consulta de los K-Vecinos Más Cercanos (K-NN) y Consulta por Rango de Similitud (Range Query). Un aspecto en común que tienen estos dos tipos de consultas, es que utilizan la propiedad de desigualdad triangular para descartar en tiempos de búsquedas elementos irrelevantes. Esta propiedad de desigualdad triangular se expresa de la siguiente forma:

Dado una entrada de ruta $route(O_i)$, todos los objetos almacenados en el subárbol de cobertura $T(O_i)$ están localizados dentro de la región (O_i, r_{O_i}) donde r_{O_i} constituye el radio de la región, ejemplo: $\forall O_j \in T(O_i), d(O_i, O_j) \leq r_{O_i}$, siendo O_j un objeto almacenado en el subárbol $T(O_i)$. Esta propiedad expresa que todos los objetos cuya distancia al centro de la zona O_i sea menor que el radio de la región r_{O_i} están comprendidos en el subárbol de cobertura $T(O_i)$.

Como se explicó anteriormente, la propiedad de desigualdad triangular es el criterio empleado para descartar elementos irrelevantes tanto en la ejecución de consultas K-NN como Consulta por Rango de Similitud. La base teórica para efectuar estos dos algoritmos de consultas se puede apreciar en tres reglas propuestas en [3]. La propiedad de desigualdad triangular en los espacios métricos para descartar elementos en el procesamiento de consultas similares en el M-Tree se expresa de la siguiente forma:

Sea (Q, r_q) una región de consulta esférica y (O_i, r_{O_i}) una región métrica descrita por una entrada de ruta catalogada como $\text{rout}(O_i)$. Si $d(O_i, Q) > r_{O_i} + r_q$, entonces para cada objeto O_j en el subárbol $T(O_i)$, se cumple que $d(O_i, Q) > r_q$. Esta propiedad expresa que la región descrita por el objeto de ruta no se interseca con la región de consulta por tanto el subárbol $T(O_i)$ puede ser excluido de la búsqueda, con todos los objetos comprendidos dentro de su radio de cobertura. Esto se obtiene mediante la propiedad de desigualdad triangular:

1. $d(O_j, O_i) + d(O_j, Q) \geq d(O_i, Q)$.
2. $d(O_j, Q) \geq d(O_i, Q) - d(O_j, O_i)$.
3. $d(O_j, Q) \geq d(O_i, Q) - r_{O_i}$.
4. $d(O_j, Q) \geq r_q$.

En el primer paso se aprecia como se utiliza la propiedad de desigualdad triangular, siendo O_j un objeto comprendido en la región descrita por O_i que constituye el objeto de ruta, por tanto $d(O_j, O_i)$ constituye la distancia de un elemento de la zona de cobertura al objeto que actúa como pivote o referencia en esta región, de igual forma $d(O_j, Q)$ constituye la distancia de un objeto de la región a un elemento de consulta dado y $d(O_i, Q)$ constituye la distancia del elemento de consulta al centro de la región. En el paso 3 se sustituye $d(O_j, O_i)$ por el radio de cobertura de la zona debido a que O_j constituye el elemento más alejado al centro de la zona o sea, coincide con el radio de cobertura de la región.

Algoritmo de consulta basado en rango de similitud en M-Tree

El procesamiento de este algoritmo de consulta en el M-Tree, es de forma recursiva siguiendo todos los caminos desde la raíz a las hojas y reportando todos los objetos O_j cuya región $T(O_i)$ se interseque con la región de consulta (Q, r_q) o sea donde se cumpla que $d(Q, O_j) \leq r_{O_j}$. En [5] los autores muestran el siguiente pseudo-código que valida la ejecución de este algoritmo:

QueryResult RangeQuery(Node N, RQuery(Q, r_q))

{

Sea O_p el padre del objeto ruta N /* if N es la raíz entonces $d(O_i, O_p) = d(O_p, Q) = 0$ */

1. Si N no es un nodo hoja entonces {

2. Para cada $\text{rout}(O_i)$ en N, Hacer{

```

3. Si  $| (O_p, Q) - d(O_i, O_p) | \leq r_q + r_{O_i}$  { /* Aplicación de Desigualdad Triangular.*/

4. Calcula  $d(O_i, Q) \leq r_q$ 

5. Si  $d(O_i, Q) \leq r_q + r_{O_i}$  entonces /* Si se intersecan las dos regiones,*/

6. RangeQuery(ptr(T(O_i)), (Q,  $r_q$ )) /*Llamada recursiva a la función */

    }

}

} Sino { /*Si el nodo es hoja*/

7. Para cada  $grnd(O_i)$  in N Hacer{

8. If  $| d(O_p, Q) - d(O_i, O_p) | \leq r_q$  entonces { /*Si se intersecan las dos regiones*/

9. calcula  $d(O_i, Q)$  /*Distancia del objeto al elemento de Consulta*/

10. if  $d(O_i, Q) \leq r_q$ 

11. añadir  $O_i$  al resultado de la consulta.

    }

}

}

```

En el pseudo-código anterior se puede apreciar la forma recursiva en que es implementado este algoritmo (Ver paso 6), donde luego de aceptar el subárbol ($T(O_i)$) se hace necesario

recorrer el mismo hasta llegar a las hojas utilizando la propiedad de desigualdad triangular y descartar regiones donde se encuentran almacenados estos objetos. En [13] los autores afirman que este algoritmo utiliza el concepto de podas negativas debido que primero chequean que el subárbol de cobertura se interseque con la región de consulta, para luego, recorrerlo recursivamente con el objetivo de descartar elementos irrelevantes; lo que ciertamente constituye un proceso de una alta complejidad computacional, por lo que proponen una variante para optimizar este proceso conocido como podas positivas. La aplicación de este proceso supondría devolver todos los objetos cuyo subárbol de cobertura se encuentre completamente comprendido en la región de consulta, ahorrando de esta forma esfuerzo computacional derivado del cálculo de distancias innecesarias. Estas podas positivas se pueden expresar de la siguiente forma:

$$\text{Si } d(O_i, O_q) + r_{O_i} \leq r_{O_q} \text{ entonces } \forall O \in T(O_i): d(O, O_q) \leq r_{O_q}.$$

Lo que significaría evitar recorrer recursivamente el subárbol $T(O_i)$, debido a que el mismo se encuentra completamente comprendido en la región de consulta. Este paso es introducido en el pseudo-código anterior, sustituyendo la llamada recursiva a la función debido a que todos los elementos serán reportados sin necesidad evaluar sus respectivas distancias al elemento de consulta, ya que el subárbol que los contiene esta completamente contenido en la región de consulta. La modificación realizada en el pseudo-código anterior se puede apreciar de la siguiente forma:

4. Calcula $d(O_i, Q)$.

5. Si $d(O_i, Q) + r_{O_i} \leq r_{O_q}$ entonces .

6. Reportar todos los objetos en el subárbol $T(O_i)$.

Estas líneas muestran el concepto de podas positivas planteadas por los autores [13] lo que significaría reportar todos los objetos contenidos en el subárbol de cobertura, debido a que el mismo se encuentra contenido completamente en la región de consulta. De esta forma se evita la llamada recursiva a la función (ver paso 6) para recorrer la región descrita por la entrada de ruta O_i .

Consulta de similitud K-NN en el M-Tree

A diferencia del algoritmo de consulta anterior donde no se pudo limitar la cantidad de objetos devueltos en una consulta de similitud, en el procesamiento de una consulta por similitud por medio de los K-Vecinos Más Cercanos (K-NN) si es posible limitar el resultado de una consulta a los k elementos más cercanos al elemento de consulta dado, evitando de esta forma la aparición de falsos positivos en los resultados requeridos de la ejecución de una consulta. En [2] los autores proponen un pseudo-código para mostrar la ejecución de este algoritmo donde utilizan una cola de prioritarios para manipular los elementos candidatos y además un arreglo de K elementos en el que se almacenará el resultado final de la ejecución. La cola de elementos prioritarios en cada posición presenta la siguiente estructura:

$$[\text{ptr}(T(O_i)), \text{dmin}(T(O_i))]$$

Para mantener punteros a subárboles cuyas regiones métricas se solapan con la región de consulta, o sea, estos subárboles no han sido descartados de la búsqueda, mientras que el orden prioritario dentro de esta cola es dado por $\text{dmin}(T(O_i))$, lo cual constituye la distancia mínima posible entre un elemento de la región (O_i, r_{O_i}) almacenado en $T(O_i)$ y el elemento de consulta. Por tanto los elementos de menor distancia se localizan primero en esta estructura.

De igual forma el arreglo NN que contendrá los K elementos finales de la ejecución de este algoritmo que en cada posición presenta la siguiente estructura:

$$[\text{oid}(O_i), d(Q, O_i)] \text{ o } [-, \text{dmax}(T(O_i))]$$

La primera estructura contiene un identificador del objeto almacenado, descrito por $oid(O^i)$, mientras que $d(Q, O^i)$ constituye la distancia del elemento de consulta al objeto O^i almacenado en la región (O^i, r_{O^i}) . La segunda estructura expresa la distancia máxima que existe entre el elemento de consulta y un objeto almacenado en la región métrica (O^i, r_{O^i}) . El arreglo de NN como se explicó anteriormente contendrá los elementos ordenados ascendentemente atendiendo a su cercanía al elemento de consulta. En [5] se proponen tres pseudo-códigos para mostrar la ejecución de este algoritmo de consulta en el M-Tree. En el primero de ellos Node Search se busca la región que se interseque con la región de consulta siguiendo el criterio de la propiedad de desigualdad triangular al igual que en el algoritmo anterior, si cumple este criterio se inserta el subárbol en la cola de los prioritarios. El segundo pseudo-código Choose Node selecciona el subárbol de la cola de pendientes que más próximo esté del elemento de consulta, y elimina esta entrada de la cola, esto se garantiza en el procedimiento de actualización de la cola que se emplea en Node Search. De igual forma K-NNQuery constituye el pseudo-código principal donde se crea el arreglo NN que contendrá el resultado final de la ejecución de esta consulta. A continuación se muestran estos pseudo-códigos:

```

Node Search(Node N, KNNQuery(Q,K))
{
    /*Sea  $O_p$  padre del objeto de ruta  $N^*$ */

    1. Si N es un nodo interno {
    2. Por cada nodo del tipo rout( $O_i$ ) Hacer {
    3. If  $|d(O_p, Q) - d(O_i, O_p)| \leq r_q + r_{O_i}$  entonces {
    4. Calcula  $d(O_i, Q)$ 
    5. Si  $dmin(T(O_i)) \leq r_q$  entonces
        {
        6. Insertar  $[ptr(T(O_i)), dmin(T(O_i))]$  a Cola de Pendientes
        7. Si  $dmax(T(O_i)) \leq r_q$  entonces
            {
            8.  $r_q = NNUpdate([- , dmax(T(O_i))])$ 

            Se elimina de la cola de pendiente todas las entradas
            en las cuales  $dmin(T(O_i)) > r_q$ 
        }
    }
    }
}

```

```

    }

    }

    }

    Sino

    /* N es un nodo hoja */

    9. Para cada  $grnd(O_i)$  en N Hacer {

    10. If  $|d(O_p, Q) - d(O_i, O_p)| \leq r_q$  entonces {

    11. } Calcula  $d(O_i, Q)$ 

    12. Si  $d(O_i, Q) \leq r_q$  entonces {

    13  $r_q = NNUpdate([oid(O_i), d(O_i, Q)])$ 

        /* Elimino de la cola de pendientes las entradas donde

         $dmin(T(O_i)) > r_q$  */ }

    }

Node ChooseNode(PRQueue PR)
{
    1. dado  $dmin(T(O_i)) = \min\{dmin(T(O_i))\}$ 

    2. Eliminar entrada  $[ptr(T(O_i)), dmin(T(O_i))]$  de la cola de pendiente

    3. Devuelve un puntero a  $(T(O_i))$ 

}

QueryResult kNNQuery(KNNQuery)
{
    1.  $PR = \{[ptr(root, \infty)]\}$ 
    2. Desde  $i=1$  a  $k$  hacer{
    3.  $NN[i] = [-, \infty]$ 
    4.  $NextNode = ChooseNode(PR)$ 
    5.  $NodeSearch(NextNode, (Q, K))$ 
    6. devuelve  $NN$  /*Arreglo con los K elemento más cercanos al elemento de consulta*/.

}

```

De forma general el algoritmo planteado por los autores es bastante óptimo, debido a que presenta una estructura de cola para mantener los subárboles candidatos aceptados mediante la

propiedad de desigualdad triangular ver pasos(3,4,5y 6) y además los mismos son ordenados acorde a su distancia al elemento de consulta mediante el procedimiento *NNUpdate*. A pesar de que el algoritmo resulta bastante óptimo, todavía puede ser mejorado, por lo que se podría aplicar el concepto de podas positivas empleado en [13] para el algoritmo de Consulta basado en Rango de Similitud. La introducción de este concepto en los K Vecinos Más Cercanos significaría una disminución en la cantidad de distancias evaluadas, debido a que los subárboles que no estén comprendidos completamente en la región de consulta, serían descartados de la búsqueda sin necesidad de evaluar distancias a los elementos comprendidos. Por otra parte si la región se encuentra totalmente comprendida en la región de consulta, todos los objetos de esa región serían reportados como candidatos sin necesidad de evaluar sus distancias al elemento de consulta.

A pesar de que el concepto de podas positivas constituye una valiosa herramienta para acelerar la ejecución de consultas similares, pudiera traer efectos negativos en cuanto a la precisión de los resultados debido a que se pudieran rechazar algunos subárboles por no estar comprendidos totalmente en la región de consulta, aumentando la probabilidad de obtener falsos rechazos.

3.2.2 Aspectos de la construcción y actualización de M-Tree

Uno de los aspectos más importantes del M-Tree es lo referente a las políticas de inserción y actualización del árbol con el objetivo de que la ejecución de consultas similares no se vea afectada por el crecimiento vertiginoso de la colección de datos.

Los algoritmos de inserción de elementos en M-Tree tienen como objetivo encontrar la forma óptima de colocar nuevos elementos en la estructura de indexación, recorriendo el árbol recursivamente desde la raíz hasta las hojas donde se encuentran almacenados los objetos de la base de datos. Estos algoritmos se apoyan en el concepto del M-tree como árbol balanceado por lo que la distribución de nodos por niveles debe ser equitativa. En [6] los autores plantean una buena política para evitar la sobrecarga y la baja densidad de nodos por niveles, con el objetivo de mantener el balance de nodos después de insertar un nuevo elemento, e impedir que el radio de cobertura en cada subárbol aumente debido a la inserción de un nuevo elemento, por lo que proponen políticas de división de los nodos. Esto garantizaría obtener las regiones lo más compactas posibles, lo cual beneficia el procesamiento de la consultas similares.

No obstante, a todas las facilidades que brinda el M-Tree como estructura de indexación para organizar elementos en un espacio métrico de altas dimensiones, no está excepto de desventajas las cuales afectarían enormemente la ejecución de consultas similares. En [6] los autores plantean que con el desarrollo de las tecnologías ha variado el concepto de grandes bases de datos, lo que era grande hace diez años en la actualidad puede resultar pequeño. Debido a esto el principal desafío de las estructuras de indexación es la escalabilidad ante el aumento de los datos.

En [2] los investigadores han identificado algunas desventajas al M-tree, en cuanto a los mecanismos de construcción y de escalabilidad cuando los datos aumentan ostensiblemente y las regiones métricas donde se almacena la información tienden a solaparse con la región de consulta, debido a la gran similitud que existe entre estas regiones, teniendo el algoritmo de consulta que evaluar la distancia de casi todas las regiones aumentando enormemente los C/C y I/O.

Optimización de la estructura de indexación M-Tree.

Debido a la capacidad del M-Tree para indexar grandes volúmenes de datos por ser un árbol dinámico, paginado y balanceado, los investigadores se han dedicado a perfeccionar esta estructura de indexación.

En [7] los autores desarrollan un modelo de costo para el procesamiento de consultas de similitud en el M-Tree. El objetivo de este modelo es estimar el costo computacional derivado del cálculo de distancias y el costo derivado por la cantidad de accesos a discos y de esta forma poder optimizar esta estructura para hacer frente al procesamiento de consultas. A diferencia de

los espacios métricos, en los espacios vectoriales debido a la representación por sus coordenadas de los datos en el espacio, existen muchos modelos de costos para estimar la eficiencia de la ejecución de consultas. Según los autores, este es el primer modelo de costo aplicado de forma general a los espacios métricos y aunque es diseñado específicamente al M-Tree puede ser extensible a otras estructuras de indexación. A diferencia de los espacios vectoriales, este modelo en espacios métricos no tiene en cuenta la representación de los datos sino la distribución de la distancia entre objetos.

Los autores desarrollan dos modelos de costos para el M-Tree:

- Node based metric cost model (N-MCM).
- Level based metric cost model (L-MCM).

El primero de ellos está referido a la estimación del cálculo de distancias y la cantidad de nodos accedidos en tiempo de ejecución de consulta. En este modelo se tiene en cuenta la ejecución de consultas, mediante el Rango de Consulta y K-Vecinos más cercanos.

El segundo recopila las mismas estadísticas que el modelo anterior pero con la diferencia de que es relativo a los diferentes niveles del árbol, por lo que utiliza solamente la cantidad de nodos por cada nivel, tomando la raíz del árbol como el nivel 1 y los nodos hojas como el nivel L. Como no accede a todos los nodos utiliza un promedio de los radios de cobertura de los subárboles.

A pesar de que el segundo modelo no tiene la precisión del primero para estimar la cantidad de distancias calculadas y la cantidad de nodos accedidos en tiempo de ejecución de una consulta, por ser relativo a los diferentes niveles del árbol requiere menos complejidad computacional para ser desarrollado, sin embargo, de forma práctica se pudo observar que la diferencia en cuanto a precisión de la estimación en los experimentos entre los dos modelos es casi insignificante.

De forma general estos modelos son muy útiles para optimizar las estructuras de indexación con el objetivo de definir las formas óptimas de construcción de estas estructuras y disminuir tanto el costo derivado por el cálculo de distancia C/C como el derivado por la cantidad de acceso al disco I/O. En el caso específico del M-Tree permitió obtener el tamaño de los nodos óptimo para minimizar los efectos del C/C y I/O.

Slim-Tree

En [9] se analiza en detalles la ejecución del M-Tree para la ejecución de consultas similares en largas colecciones de datos. En este punto los autores plantean las principales desventajas del M-Tree cuando aumentan las dimensiones de los datos y las regiones donde se almacenan los mismos tienden a solaparse impidiendo el descarte de regiones y por consecuente aumenta el C/C y de I/O.

El Slim-Tree es una estructura de indexación que recoge las principales ventajas del M-Tree como árbol dinámico, paginado y donde la información se almacena en nodos de tamaño fijo. La principal ventaja del Slim-Tree es su capacidad de cuantificar el grado de solapamiento entre las regiones. A pesar de que existen varias vías de disminuir el grado de solapamiento entre los nodos que describen regiones métricas, el Slim-Tree es la primera estructura de indexación capaz de reducir este efecto indeseado.

Para disminuir el grado de solapamiento entre las regiones se propone el algoritmo Slim-Tree down, el cual utiliza los algoritmos fat-factor y bloat-factor que tienen como objetivo medir el grado de solapamiento entre estas regiones.

El fat-factor permite obtener una medida del grado de solapamiento de los nodos donde un valor próximo a cero denota poco nivel de solapamiento y próximo a uno muestra un alto nivel de solapamiento. Este parámetro se construye con valores relativos al árbol tales como: la altura del árbol, cantidad de nodos y la cantidad de nodos accedidos en tiempo de ejecución de consulta.

Sea T un árbol métrico con altura H y M nodos, siendo $M \geq 1$. Sea N el número de objetos. Entonces fat-factor del árbol métrico T se expresa como:

$$\text{fat}(T) = \frac{I_c - H \times N}{N} \times \frac{1}{(M - H)}$$

Donde I_c denota el número total de nodos accedidos para responder una consulta por cada uno de los N objetos almacenados en el árbol métrico. $\text{Fat}(T)$ devuelve un valor en el rango $[0,1]$. Siendo el valor cero un resultado ideal.

A diferencia del fat-factor, el bloat-factor se utiliza para comparar árboles que contengan el mismo tipo de objetos almacenados pero que hayan sido construidos siguiendo diferentes criterios. La idea de este valor es identificar árboles que utilicen más de la cantidad de nodos permitidos y por consiguiente mayor cantidad de accesos a discos, y fundamentalmente encontrar el árbol óptimo con la menor altura y cantidad de nodos posibles.

El bloat-factor $\text{bl}(T)$ de un árbol métrico T se expresa como:

$$\text{bl}(T) = \frac{I_c - H_{\min} \times N}{N} \times \frac{1}{(M - H)}$$

El bloat factor varía desde cero a un número positivo que puede ser mayor que uno. Este valor habilita la comparación directa de dos árboles con diferentes bloat-factor, siendo el árbol de menor bloat-factor el que presenta menor número de accesos a discos.

La altura mínima del árbol que organiza N objetos es $H_{\min} = \lceil \log_c N \rceil$ y el número mínimo de nodos para una colección de datos es calculado como:

$$M_{\min} = \sum_{i=1}^{H_{\min}} \frac{N}{C} \text{ donde } C \text{ es la capacidad de los nodos}$$

El algoritmo Slim-Dow tiene como objetivo disminuir el solapamiento entre las regiones y obtener un árbol más ajustado para optimizar la ejecución de consultas similares. Los algoritmos bloat y fat-factor indican si el árbol debe ser mejorado. Si el valor del bloat-factor se considera elevado, se hace necesario disminuir el número de objetos para restringir la intersección de las regiones del mismo nivel y de esta forma reducir el número de nodos en el árbol.

El algoritmo Slim-down es propuesto por los autores para procesar un árbol después de construido y disminuir los dos valores anteriormente descritos. De forma general el algoritmo Slim-down se puede resumir de la siguiente forma:

- 1 Por cada nodo i en un nivel del árbol dado, encontrar el objeto c más lejano al objeto b , siendo b el centro del nodo.
- 2 Encontrar un nodo j próximo a i , el cual también contenga al objeto c . Si el nodo j existe y no está lleno, eliminamos c del nodo i e insertamos c en j y se actualiza el radio del nodo i .
- 3 Los pasos 1 y 2 deben ser aplicados secuencialmente sobre todo los nodos de un mismo nivel del árbol. Si después de efectuar una ronda completa por medio de estos dos pasos un objeto se mueve de un nodo a otro, se debe repetir el proceso.

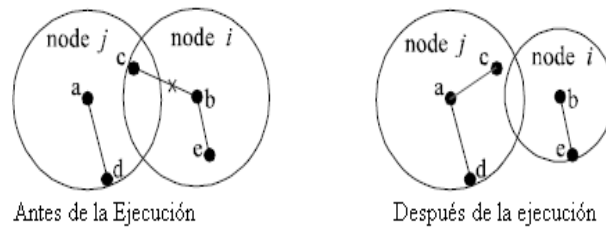


Fig. 6. Ejecución del algoritmo Slim-down

La figura 6 muestra la ejecución del algoritmo Slim-down, donde antes de la ejecución del algoritmo el objeto c más lejano del centro b del nodo i se encuentra también cubierto en el nodo j. Luego después de la ejecución del algoritmo el objeto c es eliminado del nodo i e insertado en el nodo j.

De forma general la estructura jerárquica Slim-Tree mejora la ejecución de consultas similares con respecto al M-Tree, debido a que es la primera estructura de indexación aplicada a espacios métricos que mide el grado de solapamiento entre las regiones y el algoritmo Slim-down se utiliza con el objetivo de disminuir este efecto, por lo que aplica un conjunto de criterios para que el diseño del árbol sea lo más eficiente posible. De igual forma en esta estructura se implementa un nuevo algoritmo de división de los nodos, el cual se ejecuta más rápido que los anteriores algoritmos aplicados al M-Tree, sin sacrificar la rapidez de la ejecución de las búsquedas. Además otro algoritmo se presenta para guiar la inserción de un objeto como un nodo interno en un subárbol apropiado.

En [3] los autores proponen una estructura de indexación nombrada Lista de Agrupamiento o List of Cluster (LC) para reducir la dimensionalidad intrínseca de las colecciones de datos. Está diseñada tanto para memoria principal como para memoria secundaria. La idea de esta estructura es dividir la colección de objetos en zonas compactas acorde a sus distancias al centro. Cada zona es conocida como una esfera o ball y se denota como (c, R_c) donde c es el centro de la esfera, y R_c denota el radio de la región. La idea de construcción de esta estructura es bastante sencilla. Primeramente se selecciona un centro como un elemento de la base de datos y luego se construye la zona compacta con los elementos que contengan una distancia al centro de la esfera menor que el radio de la zona. Para esto se define una estructura de almacenamiento nombrada como bucket que almacene los elementos que estén comprendidos dentro del radio de la zona.

Los elementos que cumplan la condición anteriormente planteada se denotan como:

$$I_{U,c,r_c} = \{u\} \in U - (c), d(c, u) \leq r_c$$

Mientras que los elementos que no cumplan la condición anteriormente descrita se consideran externos a la zona generada anteriormente y se denotan como:

$$E_{U,c,r_c} = \{u \in U, d(c, u) > r_c\}$$

Luego el proceso es construido recursivamente en E para obtener una estructura de tres elementos (C_i, r_i, I_i) .

La ejecución de consultas similares en esta estructura, primero mide la distancia al primer grupo generado y si cumple con la condición de que el grupo se interseque con la zona de consulta se revisa recursivamente el bucket I. Si la región de consulta no se cruza con el primer grupo se revisan los elementos externos denotados como E. Con el objetivo de acelerar las búsquedas, los autores introducen el concepto de podas positivas, esto implicaría descartar grupos que no se encuentren totalmente contenidos en la zona de consulta. Las pruebas realizadas por los autores a estas estructuras son con el objetivo de definir aspectos de la construcción de la estructura tales como:

- Centro de cada zona.
- Tamaño de los grupos.

La definición de estos dos elementos tienen gran importancia para definir aspectos tales como cantidad de distancias computadas en la ejecución de una consulta similar, y el % de elementos accedidos o el % de la base de datos evaluada. En las pruebas experimentales se muestra que el criterio fundamental para escoger los centros es la distancia que exista entre los centros de cada zona. Esto sería útil para evitar la intersección de estos grupos.

A pesar de que el propósito de esta estructura es disminuir el efecto de la dimensionalidad intrínseca de los datos, el diseño propuesto por los autores es para el trabajo en memoria principal, por lo que aspectos referentes a los costos derivados por acceso a memoria secundaria no son tomados en cuenta, impidiendo su comparación con estructuras clásicas como M-Tree[6] y Slim-Tree[9]. No obstante a eso, esta estructura tiene gran utilidad pues permitió a los autores reafirmar la superioridad de las estructuras de indexación basadas en agrupamiento a las basadas en pivotes, debido a los costos derivados por cálculos de distancias cuando aumenten las dimensiones de los datos.

3.3 Paralelismo aplicado a las estructuras de indexación

Las estructuras de indexación centralizadas¹ pueden alcanzar un aumento significativo de velocidad, en términos de cálculos de distancias y páginas de discos leídas en la ejecución de una consulta de similitud. Sin embargo, las experiencias adquiridas con estas estructuras centralizadas muestran una fuerte relación entre el tamaño de la colección de datos y los costos derivados de las búsquedas; esto se expresa como el aumento de los costos de búsquedas linealmente cuando aumenta el tamaño de la colección de datos. La habilidad de las estructuras de indexación centralizadas para mantener un razonable tiempo de respuesta de consultas, cuando aumenta el tamaño de la colección de objetos, está limitada. De ahí la importancia de emplear paralelismo [2].

Un sistema paralelo es un dispositivo compuesto de múltiples unidades independientes de procesamiento y unidades de almacenamiento igualmente independientes. Todas las unidades mantienen una comunicación entre sus medios y sus datos. El entorno de computación paralela, está referido a múltiples procesadores con varias unidades de disco. Los procesadores (CPUs) comparten memoria operativa RAM y utilizan bus internos para comunicarse con las unidades discos [15].

En orden de explotar completamente un entorno paralelo una estructura de indexación debe presentar las siguientes propiedades:

- Comunicación de datos: Cualquier objeto almacenado en una base de datos debe estar habilitado para cualquier procesador al mismo tiempo, aquí el sincronismo es fundamental, debido a que si un objeto está siendo analizado por un procesador no debe estar accesible para otro.
- Múltiples operaciones al mismo tiempo: el sistema puede evaluar varias operaciones independientes en diferentes procesadores, el número de tareas procesadas en paralelos está limitado por el número de unidades de procesamiento (CPU). Estas tareas definidas son inserciones, actualizaciones o eliminaciones de nodos.
- Almacenamiento Paralelo: Los datos pueden ser almacenados en múltiples discos, y cada disco está habilitado para cada procesador, donde existe la posibilidad de mover datos de un disco a otro.

Los pasos anteriormente descritos permiten a una estructura de indexación paralela procesar objetos almacenados en una base de datos, usando múltiples procesadores al mismo tiempo que almacenen los datos eficientemente distribuidos a través de las unidades de discos, brindando la posibilidad de accesos paralelos a los objetos almacenados mientras se ejecuta una consulta.

El paradigma del paralelismo ha mostrado un real aumento de velocidad en estructuras de indexación centralizadas. Sin embargo, esto presenta una escalabilidad limitada debido a la naturaleza de la computación paralela. Computadoras con largos números de procesadores, así como, arreglos de discos de gran almacenamiento, son mucho más costosas que una red con varias estaciones de trabajo. Esto es conocido como un entorno distribuido [2].

En un entorno distribuido, las computadoras son conectadas a una red de alta velocidad. Aquí se comparte el poder de procesamiento de sus CPU así como sus recursos de

¹ Diseñadas para máquinas personales.

almacenamiento. Con el objetivo de devolver consultas, almacenar nuevos datos o eliminar datos innecesarios, las computadoras encuentran otras computadoras por medio de una navegación específica o un mecanismo de enrutamiento. Existen dos paradigmas para las estructuras de indexación distribuida:

- Scalable and Distributed Data Structures (SDSS).
- Peer-to-Peer (P2P).

Scalable and Distributed Data Structures (SDSS)

El paradigma SDSS fue originalmente propuesto en [16] para búsquedas simples de llaves y cadenas. Los objetos son almacenados en archivos distribuidos o computadoras especializadas conocidas como servidores. A medida que aumenten la cantidad de archivos se necesitarán más servidores como capacidad adicional de almacenamiento. Los archivos son modificados y consultados por nodos de red llamados clientes, para inserciones, eliminaciones y operaciones de búsquedas. Para obtener altos niveles de efectividad se debe cumplir los siguientes requisitos:

- Escalabilidad: Los datos deben migrar a otros nodos de red sólo cuando los nodos de red existentes estén suficientemente cargados.
- No hotspot²: No existen sitios maestros a los cuales se acceda para resolver el direccionamiento de las búsquedas de objetos. Por lo tanto, no existe un directorio centralizado.
- Independencia: El acceso a los ficheros y a las primitivas de mantenimiento, tales como: búsquedas, inserciones o divisiones de nodos, nunca requieren actualizaciones atómicas en múltiples nodos.

La segunda propiedad se refiere a la desventaja que presentan los directorios centralizados como los hotspots para conducir al bottlenecks³ cuando aumentan las dimensiones de los datos. La tercera propiedad es vital en un ambiente distribuido, debido a que consultar otros nodos puede ser ineficiente o en ocasiones imposible en largas escalas de redes. Por lo que estas actualizaciones se efectuarán de forma independiente en cada nodo.

Peer to Peer

Otro paradigma distribuido es el P2P. En este ambiente, los nodos de la red son llamados peers. Los peers ofrecen algunos recursos computacionales que pueden ser usados por otros nodos. Las redes P2P presentan los mismos principios que SDSD pero añaden nuevos requerimientos para resolver problemas de inestabilidad en la red principal. La estructura P2P se puede resumir en las siguientes ideas:

- Peer: cada nodo que participa en la estructura presenta ambos roles, cliente o servidor, o sea, los nodos pueden ejecutar consultas y al mismo tiempo almacenar una parte de los datos procesados.
- Tolerancia de fallo: las fallas en un nodo de la red que pertenece a la estructura no es fatal. Todas las operaciones pueden ser ejecutadas, menos la parte afectada de la colección de datos pues será inaccesible.
- Redundancia: Los componentes de los datos son replicados en múltiples nodos para incrementar su disponibilidad, esto permite que los algoritmos de búsquedas tengan la posibilidad de acceder a las instancias por múltiples caminos.

No todas las estructuras P2P cumplen esas condiciones, pero son necesarias para obtener máxima escalabilidad y efectividad en los sistemas de indexación.

El principal objetivo de aplicar paralelismo a las estructuras de indexación es disminuir los C/C y los I/O costos derivados de la ejecución de consultas similares.

² Elemento de la red que permite la comunicación de múltiples nodos.

³ Problemas que ocurren en las redes cuando existe gran número de peticiones en una entrada de un elemento de ruta y el sistema no es capaz de responder a todas estas peticiones.

En [2] se aplicó teóricamente el paradigma del paralelismo al M-tree por ser una de las pocas estructuras de indexación jerárquica que trabaja tanto en memoria principal como en memoria secundaria. Para ejecutar paralelismo en esta estructura se tiene en cuenta los siguientes aspectos:

- Se debe respetar las dependencias jerárquicas entre un nodo padre y sus respectivos nodos hijos.
- Las búsquedas deben comenzar por la raíz del árbol y continuar recursivamente atravesando los nodos hijos relevantes hasta llegar a los nodos hojas donde se encuentra almacenada la información.
- Un nodo hijo en un nivel dado no puede ser accedido antes que todos sus antecesores hayan sido procesados. Debido a esto sólo los nodos del mismo nivel serán procesados en paralelo.

A pesar de plantear el paradigma del paralelismo a la estructura de indexación M-tree los autores no mostraron resultados prácticos de su aplicación, esto fue llevado a cabo en Gh-Trees.

Las pruebas realizadas por los autores tuvieron lugar usando dos bases de datos del mundo real. Ambas contaron con 100,000 objetos del mundo real, de la primera base de datos se extrajeron vectores de 45 dimensiones de rasgos de colores de imágenes, mientras que en la segunda se utilizaron 100 000 oraciones del lenguaje checo.

Las pruebas se realizaron en una red de 100 máquinas. Los autores desarrollaron varias pruebas utilizando el prototipo para la indexación distribuida P2P, aplicada a la estructura de indexación Gh-tree, que a pesar de no presentar los niveles de eficiencia del M-Tree y el Slim-Tree, puede ser útil para validar la importancia del paralelismo en las estructuras de indexación. De todas las pruebas realizadas tiene vital importancia las referentes a la ejecución de algoritmos de consultas similares tanto K-NN o Range Query, para lograr esto se evaluaron estos dos algoritmos de la forma tradicional y mediante la implementación en paralelo.

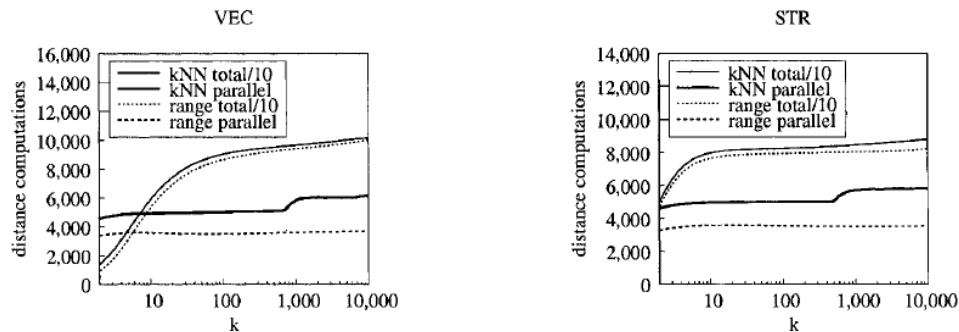


Fig. 7. Cantidad de distancias calculadas mediante la ejecución de consultas similares

Las figuras 7 y 8 muestran como la ejecución de consultas similares para las dos pruebas planteadas, mediante los algoritmos K-NN y Range Query evaluadas paralelamente mantienen números de distancias calculadas estables cuando aumentan la cantidad de elementos devueltos en la ejecución de las consultas.

De igual forma los autores plantean pruebas para medir los costos de la computación paralela cuando aumenta el volumen de los datos, lo que valida la escalabilidad de la computación paralela para hacer frente al crecimiento global de la información.



Fig. 8. Cantidad de distancias calculadas vs aumento de la colección de datos

Los resultados muestran como la cantidad de distancias calculadas se mantiene prácticamente constante a medida que el tamaño de la colección de datos aumenta.

A pesar de que el paradigma del paralelismo propuesto por los autores no tuvo en cuenta los esquemas de las computadoras con varios procesadores y varias unidades de almacenamiento, sino, que estuvo centrado a las estaciones de trabajo con varias máquinas conectadas en red, los resultados validaron su capacidad para ser aplicados a los espacios métricos para disminuir los costos computacionales derivados del cálculo de distancias, y de esta forma brindarles a estas estructuras la capacidad de la escalabilidad cuando aumenta ostensiblemente el volumen de los datos.

En [28] se propone otra estructura de indexación distribuida llamada M-Chord para la aceleración de búsquedas similares en un espacio métrico. La estructura de indexación M-Chord tiene como objetivo mapear el espacio métrico a un dominio de una dimensión. Para ello utilizan idistance[29] un método de indexación para espacios vectoriales el cual será generalizado para su utilización en espacios métricos. De forma general la estructura M-Chord divide la colección de datos utilizando el criterio de los pivotes basado en idistance, en la cual el único criterio a tener en cuenta será las distancias relativas entre los objetos. De forma general la estructura M-Chord para cada zona obtenida de la partición asigna una llave que será el identificador de cada zona. La estructura se observa de una forma circular donde cada nodo contiene las direcciones de su antecesor y su predecesor, así como una tabla de rutas llamada *fingers table* que posee las direcciones de todos los restantes nodos. De forma general la estructura de M-Chord se puede describir de la siguiente forma:

- Divide el espacio en intervalos y en cada nodo almacena datos con llaves del mismo intervalo.
- Utiliza una tabla de ruta para la navegación.
- En el se aplica los algoritmos de consulta Range Search y K-NN.
- Provee un mecanismo de filtrado adicional para reducir los costos computacionales del procesamiento de consultas.

El sistema M-Chord constituye una capa lógica sobre una red de nodos directamente accesibles, los nodos son independientes para almacenar datos, y pueden insertar objetos en el sistema y recuperarlos por la ejecución de consultas similares. Estos nodos se comunican vía mensajes mediante el protocolo TCP.

Las pruebas experimentales para validar la importancia de la aplicación de estructuras distribuidas sobre las centralizadas, demostró cuan importante es la aplicación del paradigma del paralelismo para disminuir los costos computacionales derivados del cálculo de distancias en un espacio métrico de altas dimensiones.

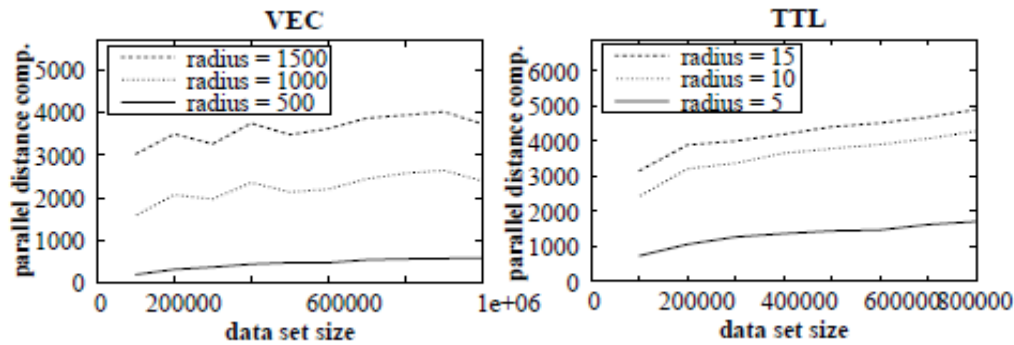


Fig. 9. Escalabilidad del M-Chord cuando aumenta ostensiblemente el tamaño de la colección

La figura 9 muestra como las dos pruebas realizadas demuestran la escalabilidad de la estructura de indexación distribuida M-Chord, para el aumento de la colección de datos, donde los costos computacionales derivados del cálculo de distancia se mantienen prácticamente constantes.

En [27] se propone una estructura de acceso escalable y distribuida para búsquedas de similitud en un espacio métrico nombrada M-CAN, la cual está basada en el paradigma CAN⁴, donde se utiliza una tabla de hash distribuida para la organización de la estructura. Esta estructura es capaz de soportar almacenamiento y recuperación de objetos de un espacio métrico.

En esta estructura se utilizan técnicas de indexación basadas en pivotes [2] para distribuir los objetos en los diferentes nodos de la red. La estructura de la red utilizada está basada en el protocolo P2P, el cual proporciona tiempos de ejecución de consultas estables a medida que aumenta el tamaño de la colección de datos. La estructura M-CAN diseñada para soportar algoritmos de consultas mediante el rango de consulta.

Un aspecto notable de esta estructura es lo referente a los aspectos de construcción, inserciones y procesamiento de consultas mediante el rango de consulta, donde la utilización de los mensajes entre los nodos, son útiles para la ejecución de estas operaciones. La construcción de la estructura tiene como objetivo adaptarse dinámicamente al aumento constante del tamaño de la colección de datos, esto refleja la escalabilidad de esta estructura, lo que garantiza óptimos tiempos de respuesta en la ejecuciones de consultas. Cuando un nodo tiene una cantidad de objetos que excede el límite definido, la estructura es capaz de efectuar la división en el nodo y asignar una cantidad de objetos determinada a un nodo que se encuentre libre. De igual forma el procesamiento de consultas en esta estructura se efectúa comenzando por cualquier nodo de la red y evaluar la distancia del objeto de consulta al nodo de la red mediante la estructura de los pivotes. El algoritmo de consulta utilizado Range Query, lo que significaría evaluar el nodo si el pivote representativo del nodo se encuentra comprendido dentro del radio de la consulta. Un aspecto notable de esta estructura es la utilización interna del esquema de pivotes dentro de cada nodo lo que garantizaría disminuir el número de distancias calculadas cuando el algoritmo de consulta tenga que evaluar un nodo internamente.

Los autores en las pruebas experimentales utilizaron dos colecciones de pruebas. La primera colección está compuesta por 100 000 vectores de 45 dimensiones de rasgos de colores extraídos de 100 000 imágenes y la segunda colección contiene 100 000 oraciones de tamaño entre 20 y 300 caracteres. Las métricas utilizadas fueron la distancia euclidiana entre vectores y la distancia de Edición para la comparación de cadenas respectivamente. Los resultados mostrados por los autores en cuanto a la escalabilidad de esta estructura, cuando aumentan las dimensiones de los datos para diferentes radios de consulta, demuestran la efectividad de estas

⁴ Content-addressable Network. Redes donde los accesos son dirigidos por contenidos.

estructuras de indexación distribuidas para paralelizar la ejecución de consultas en los espacios métricos.

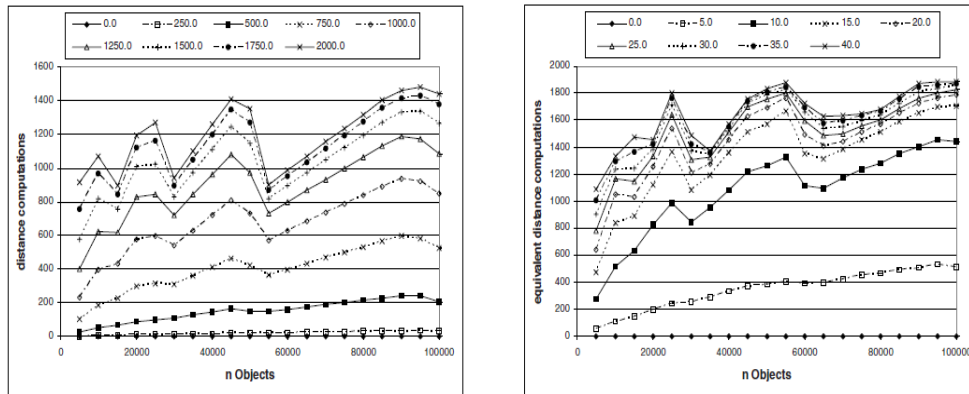


Fig. 10. Cantidad de distancias evaluadas en forma paralela cuando aumentan las dimensiones de los datos para las dos colecciones de pruebas

La figura 10 muestra como la cantidad de distancias paralelas calculadas se mantiene en un rango aceptable, cuando aumenta la cantidad de objetos en la base de datos para las dos pruebas realizadas con respecto a los diferentes radios de consultas seleccionados.

Del análisis de las principales características de las técnicas de indexación aplicadas a espacios métricos, podemos concluir que en la bibliografía en general se aborda esta temática desde un punto de vista genérico. El principal objetivo de los investigadores es desarrollar estructuras que permitan obtener eficientes ejecuciones de consultas similares, sin tener en cuenta los aspectos de la precisión en la recuperación de los resultados, debido a que este último aspecto es dependiente de las aplicaciones desarrolladas, y los artículos referidos abordan el problema desde un punto de vista genérico.

De forma general se ha identificado en la bibliografía que el problema fundamental en los espacios métricos está centrado, cuando aumentan las dimensiones del espacio donde todos los elementos están aproximadamente a la misma distancia uno de los otros [2,3,5]. Esto implica que en los algoritmos de consultas similares no se puedan descartar regiones sin computar sus distancias.

Por tanto, el objetivo fundamental de los investigadores en la actualidad está encaminado a desarrollar estructuras de indexación distribuidas que presenten gran escalabilidad, para hacer frente al marcado crecimiento de los datos. Por tanto implementar modelos paralelos en los espacios métricos es de gran importancia.

4 Estructuras de indexación aplicadas al reconocimiento de rostros

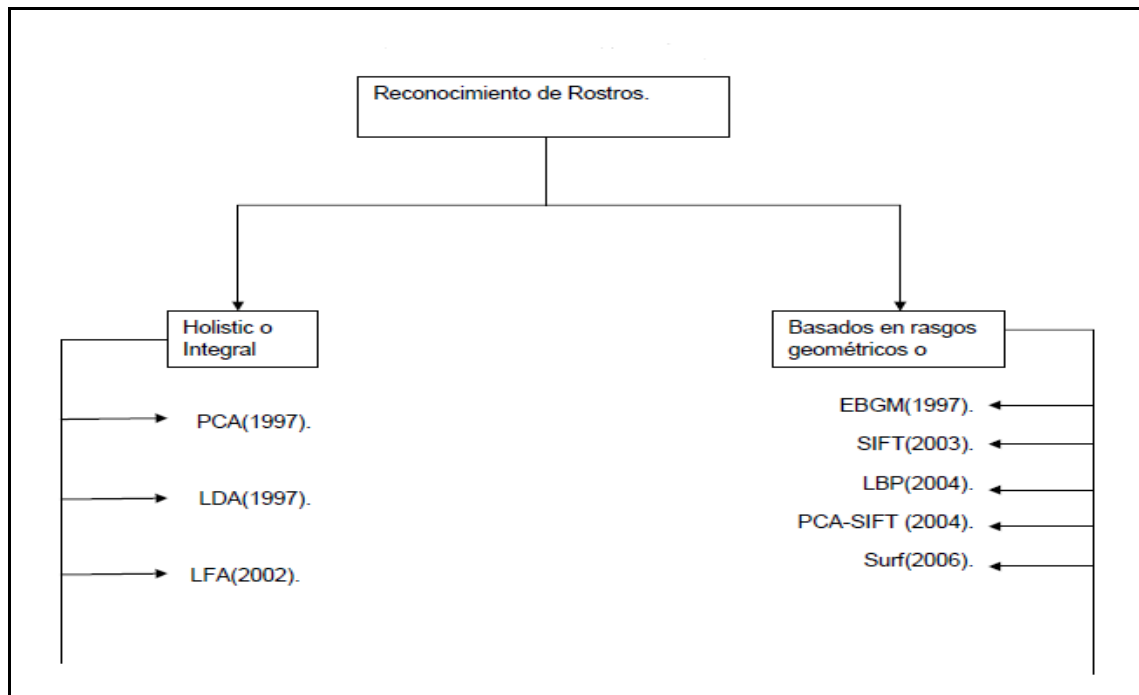


Fig. 11. Taxonomía de los principales algoritmos de extracción de rasgos

4.1 Aspectos generales del reconocimiento de rostros

El reconocimiento de personas por su rostro es una actividad cotidiana para las personas. Las personas prácticamente desde que nacen son capaces de reconocer por las características de su rostro a otros de su mismo entorno. El desarrollo de las tecnologías y el aumento de la interacción entre los hombres y los sistemas automatizados de la información, han permitido que los humanos le brinden a las computadoras la capacidad de reconocer personas por su voz, huellas digitales y rostros [24].

El reconocimiento de rostros es uno de los métodos biométricos para el reconocimiento de personas mediante sus rasgos faciales. Las investigaciones en esta área existen desde hace más de 30 años y actualmente tiene gran utilidad en áreas con sistemas de seguridad donde puede ser útil para control de acceso de personas a determinados locales y en sistemas de identificación de persona en áreas tales como terminales de aeropuerto donde tener el control de la verdadera identidad de las personas que arriban a un país es fundamental.

Desde que la secuencia de una imagen es capturada por una cámara el objetivo es obtener la mejor identificación con otra imagen de rostro dada. El proceso de reconocimiento de rostro se puede apreciar de forma general en el siguiente diagrama.

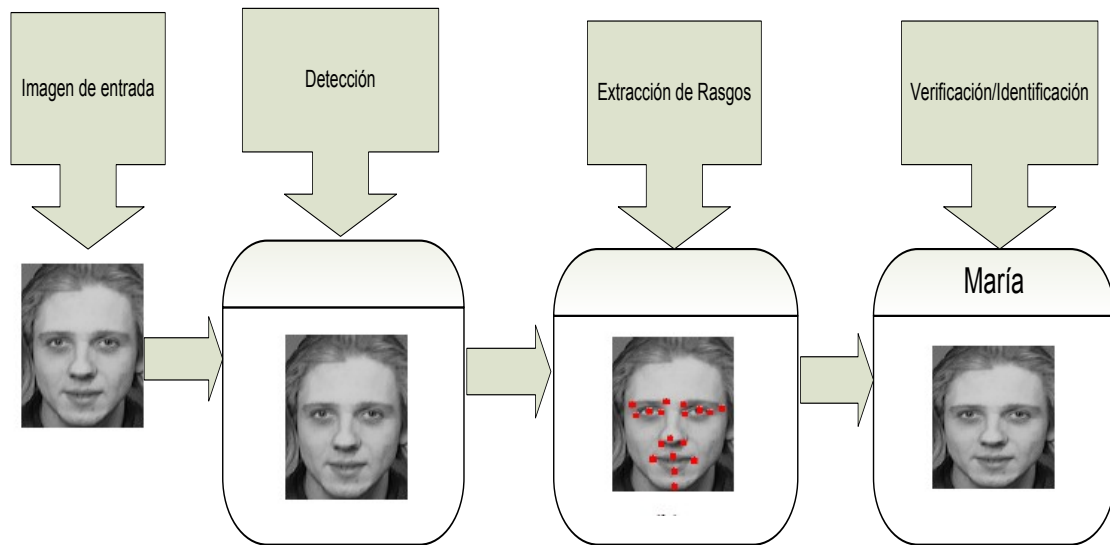


Fig. 12. Secuencia del Reconocimiento de Rostros

El reconocimiento automático de personas por su rostro contiene tres etapas fundamentales:

- Detección de rostros en una escena determinada.
- Extracción de rasgos.
- Identificación/Verificación.

En la primera etapa, el sistema dado una imagen de entrada tanto de video como estática debe capturar la región donde se encuentra enmarcado el rostro. En la segunda etapa luego de tener capturado el rostro, se hace necesario obtener los rasgos que contengan la información más discriminativa para luego en la última etapa lograr obtener precisos procesos de identificación o verificación en una base de datos donde las imágenes de rostros se encuentren previamente almacenadas. El proceso de verificación se refiere a la comparación 1: 1 de una imagen de rostro dada con una imagen previamente almacenada, mientras que el proceso de identificación se refiere a la comparación de una imagen de rostro dada con respecto a todas las imágenes de rostros previamente almacenadas.

Para efectuar el proceso de reconocimiento de rostro se hace necesario obtener una representación del rostro que permita obtener descriptores que modelen la apariencia de los mismos mediante sus rasgos. Estas representaciones se pueden clasificar en tres grupos fundamentales: Modelo basado en rasgos geométricos o descriptores locales.

- Modelo integral (Holistic).
- Modelo basado en rasgos geométricos o descriptores locales.
- Modelos híbridos.

En el primer grupo, la región completa del rostro es tomada en cuenta en los sistemas de reconocimiento, para luego aplicar reducción de la dimensionalidad y conducir el análisis hacia un sub-espacio de análisis, mediante análisis de componentes principales PCA, análisis de discriminantes lineales LDA o discriminantes lineales de Fisher LFA entre otros.

En el segundo grupo, rasgos locales en el rostro alrededor de áreas discriminativas como la nariz, ojos y boca son obtenidos y después usados como datos de entrada para los sistemas de reconocimiento de rostros.

En el tercer grupo los métodos de reconocimiento se refieren a la forma en que se perciben en los sistemas de visión humana los rasgos locales y el rostro entero. En este grupo se usan módulos eigenfaces, rasgos híbridos locales y normalización de formas.

Del análisis de estos tres grupos podemos decir que los descriptores locales han ido ganando popularidad en el campo del reconocimiento de rostro, debido a que son más tolerantes a los cambios de pose y problemas de iluminación que los modelos Holistic o Integrales, además de

regiones mediante los descriptores de rasgos locales [30,34,37,42]. Por tanto el objetivo de este trabajo está centrado en la representación de imágenes de rostros mediante los modelos basados en rasgos geométricos o descriptores locales obtenidos de las imágenes de rostros.

4.2 Ubicación de las imágenes de rostros en estructuras de indexación métricas

La característica fundamental de un espacio métrico se expresa como la relación que se establece entre los objetos mediante la utilización de una función de distancia conocida como métrica. El concepto de similitud entre objetos va más allá de la forma en que los humanos aprecian esta similitud, debido a la complejidad estructural que compone estos objetos, no permite definir en un sistema computacional la similitud necesaria. Una forma de medir la similitud entre dos objetos, sería obteniendo una abstracción que contenga la información estructural de los mismos [12].

La información estructural de estos objetos se obtiene mediante algoritmos de extracción de rasgos que permitan obtener una descripción de las principales características presentes en los objetos y que son almacenadas en vectores de rasgos o *features vector* [12]. En el caso específico de los objetos biométricos tales como: rostros, huellas, digitales, iris entre otros, estos vectores de rasgos contendrían la información más discriminativa. A diferencia de los SAM donde los objetos son representados en el espacio de coordenadas, los MAM sólo definen distancias entre estos elementos. Por tanto, en tiempo ejecución de una consulta de similitud sería evaluar la distancia entre los descriptores de rasgos de un objeto de consulta y todos los descriptores de rasgos de los objetos almacenados en la base de datos; definiendo la distancia más corta como los objetos más similares. Un problema adicional lo constituyen las funciones de distancias utilizadas, pues las mismas deben cumplir las tres propiedades de los espacios métricos, para poder medir similitud entre los descriptores de rostros extraídos. De forma general el problema anteriormente planteado se puede resumir en la siguiente figura:

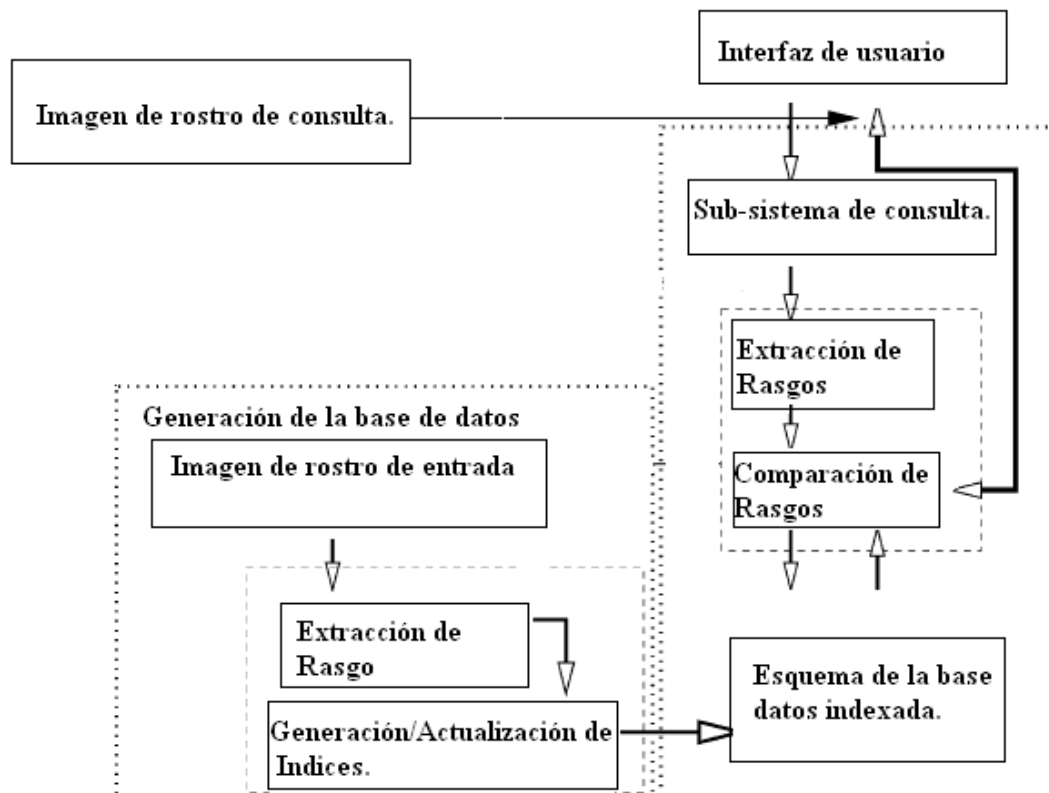


Fig. 13. Esquema general del proceso de reconocimiento de rostros en una base de datos

La figura 13 muestra una secuencia del proceso de indexación en espacios métricos, cuando una imagen de rostro de consulta se presenta en el sistema. Este proceso se puede dividir en tres subsistemas fundamentales. El primer subsistema es conocido como Interfaz de Usuario, donde el rostro de una o varias personas es detectado para luego efectuar procesos de reconocimiento con respecto a una base de datos de imágenes de rostros. El segundo bloque es conocido como subsistema de consulta y tiene como objetivo realizar la comparación de la imagen de consulta con respecto a una base de datos de imágenes de rostros, esta comparación se efectúa definiendo distancia entre los rasgos obtenidos de la imagen mediante algoritmos de extracción con respecto a los rasgos de cada imagen almacenada en la base de datos, definiendo la distancias más cortas como los rostros más similares. La principal dificultad con este proceso es relativa al esfuerzo computacional derivado del cálculo de distancias, donde mientras más imágenes se encuentren almacenadas mayor número de comparaciones será necesario realizar para efectuar el proceso de reconocimiento. Como se planteó anteriormente la principal vía identificada para atenuar este problema lo constituyen las estructuras de indexación aplicadas a espacios métricos, por lo que el tercer sub-sistema tendrá como objetivo la generación de una estructura de indexación para acelerar el proceso de reconocimiento de rostros, disminuyendo de esta forma los costos computacionales derivados del cálculo de distancias.

Debido a la importancia que toman los descriptores de imágenes de rostros en los procesos de verificación e identificación, se hace necesario realizar un estudio de las principales formas existentes para representar el rostro, que permitan definir la forma óptima de vincular estas representaciones en las estructuras de indexación identificadas.

4.2.1 Descriptor de imágenes de rostros

Los descriptores de imágenes de rostros están representados por vectores que contienen los rasgos más discriminativos de los rostros, tales como: información de los ojos, boca, nariz, lo que permitirá efectuar comparaciones entre varios descriptores de rostros con la misma información. De forma general un descriptor de una imagen de rostros contendrá la siguiente estructura:

Sea D un descriptor de imagen: D está definido como un par (ϵ_D, ∂_D) donde $\epsilon_D: I \rightarrow R^N$ es una función que extrae un vector de rasgos $\bar{v}$ de una imagen I , y $\partial_D: R^N \times R^N \rightarrow R^N$ denota la función de distancia usada para calcular la similitud entre dos imágenes de rostros mediante sus vectores de rasgos. Mientras menor sea esta distancia, más similares son estas imágenes. Donde la similitud entre dos descriptores de imágenes constituye un caso especial de espacios métricos [12,19].

Un vector de rasgos $\bar{v}$ de una imagen I es un punto en el espacio R^N tal que:

$\bar{v} = (v_1, v_2, \dots, v_n)$, donde n denota las dimensiones del vector.

Recientemente los algoritmos de extracción de rasgos basados en modelos geométricos o locales han cobrado una mayor popularidad en los sistemas de reconocimiento de rostros que los modelos integrales o *holistic*. Entre los principales artículos que abordan esta temática podemos encontrar LBP[25], Elastic Bunch Graph Matching[14] y descriptores locales tales como SIFT[38] y SURF[39] dedicados originalmente al reconocimiento de objetos en escenas determinadas, pero debido a sus cualidades de obtener descripciones locales de puntos invariantes a efectos tales como: rotación, iluminación y cambios de ángulos de vista en las imágenes han motivado a los principales investigadores en esta temática a incluirlos en el reconocimiento de rostros[37,40,41,42,43]. En este artículo se aborda el algoritmo SIFT como descriptor local de imágenes de rostros ya que constituye un poderoso algoritmo para la extracción de puntos invariantes que permitan comparar imágenes bajo diferentes condiciones de iluminación y rotación. De igual forma se propone un acercamiento al descriptor local SURF que al igual que el SIFT ha sido empleado recientemente en el reconocimiento de rostros.

4.2.2 Algoritmo SIFT como descriptor local

El algoritmo Scale Invariant Transforms Features SIFT [38] propuesto por Lowe ha sido ampliamente utilizado para la detección y reconocimiento de objetos en escenas determinadas, pero recientemente ha comenzado a ser utilizado en aplicaciones dedicadas al reconocimiento de rostros.

A pesar de que el algoritmo SIFT ha demostrado ser efectivo en el reconocimiento de objetos en escenas determinadas, su aplicación en el reconocimiento de rostros es diferente debido a que los imágenes de rostros no son rígidas como los objetos, pues las mismas pueden verse afectadas debido a cambios de pose y de expresión lo cual dificulta mucho el reconocimiento. Por tanto, la efectividad del SIFT en el reconocimiento de rostros constituye un problema abierto en la bibliografía actual.

El algoritmo SIFT contiene dos etapas fundamentales, la primera es conocida como detección de puntos claves que son estables a diferentes resoluciones e invariantes a la escala y a la rotación y la segunda descripción de las regiones donde se encuentran enmarcados esos puntos. La etapa de detección contiene dos fases fundamentales:

- Detección de extremos en el espacio escala.
- Ajuste de la localización de los puntos claves.

Detección de extremos en el espacio escala

En la primera fase se identifican potenciales puntos de interés por medio de una función de diferencias gaussianas, los cuales son invariantes a la escala y orientación. De la convulsión de la imagen con un kernel gaussiano se obtiene la imagen vista a varias escalas, de estas vistas se conforma una matriz de $3 \times 3 \times 3$ o sea 27 elementos, de los cuales se obtienen los puntos extremos, es decir, máximo o mínimos locales, derivados de la comparación de un píxel con sus 8 vecinos de la misma escala y los restantes 18 de escalas próximas. Si un píxel constituye un máximo o un mínimo en esa comparación es seleccionado como punto de potencial interés.

Ajuste de la localización de los puntos claves

En esta fase del algoritmo se utiliza como un filtro para las localizaciones candidatas seleccionadas en el paso anterior, por lo que los puntos candidatos que resulten sensibles al ruido o que se encuentren pobremente localizados en un arco serán eliminados. Esto se realiza mediante la medida de estabilidad de estos puntos por lo que un umbral se define para filtrar los puntos que su medida sea menor que este umbral.

Luego de obtener los puntos claves en la primera etapa de detección se hace necesario obtener las descripciones de las regiones donde se encuentran localizados estos puntos. Esta etapa es conocida como descripción y contiene dos fases fundamentales.

Asignación de orientación

En esta fase se le asignan una o más orientaciones a cada localización mediante la dirección del gradiente local calculado en la vecindad de cada punto utilizando también la imagen más próxima en escala. Para obtener la orientación del punto se calcula un vector con la orientación dominante del gradiente local, mediante la medida del gradiente de cada píxel en la vecindad del punto y la vecindad obtenida en la escala próxima a la imagen. Un punto clave es obtenido por la dirección del gradiente de mayor medida en la región. En esta fase todas las medidas son obtenidas relativas a la orientación del punto, lo que garantiza invariancia a la rotación.

Descripción de los puntos claves

En etapa se obtienen los vectores de rasgos para cada región resultante en los pasos anteriores. Estos descriptores se obtienen relativos a la orientación de cada punto clave en una vecindad de 4×4 alrededor de un punto con 8 orientaciones dominantes, por lo que el vector resultante contendrá 128 elementos, o sea los descriptores SIFT presentan dimensiones de 128 coordenadas. De forma general un vector obtenido mediante el algoritmo SIFT contiene la información relativa a un punto clave y el mismo tiene la siguiente estructura $(x, y, m, \theta, \partial)$:

- X: Coordenada horizontal del punto.
- Y: Coordenada Vertical del punto.
- M: Magnitud del Gradiente Local.

- θ : Orientación del Gradiente Local.
- ∂ : Escala donde se localizó el punto.
- Lista de 128 elementos que conforman la región.

Como se planteó anteriormente el algoritmo SIFT constituye uno de los principales descriptores locales para la detección de objetos en escenas determinadas, pero recientemente la comunidad del reconocimiento de rostro ha comenzado a utilizarlo debido su capacidad para obtener poderosas descripciones de puntos claves invariantes a la rotación y a la escala. Por tanto a continuación describiremos los principales artículos que abordan el SIFT en el reconocimiento de rostros.

4.2.3 Algoritmo SIFT en el reconocimiento de rostros

Recientemente algunos investigadores han tratado de vincular el algoritmo SIFT propuesto por Lowe en la detección y descripción de los rasgos locales presentes en las imágenes de rostros, a pesar de que el reconocimiento de rostros se ve afectado por efectos tales como cambios de pose, y expresión lo cual difiere con el reconocimiento de objetos donde fue empleado originalmente este algoritmo. Las características de este algoritmo han motivado a muchos investigadores a vincularlo en esta materia. A continuación presentamos las principales ventajas de los rasgos SIFT a criterios de varios autores [37, 40, 41, 42,43], para su utilización en el reconocimiento de rostros:

- Los rasgos obtenidos presentan un alto grado de invariancia a la escala.
- Son invariantes a transformaciones 2D como traslación y rotación.
- Son invariantes a cambios de iluminación.

Además de todos estos factores antes mencionados, en [42] los autores plantean la posibilidad de que los rasgos SIFT sean también robustos a la oclusión parcial y a cambios extremos de expresión facial, aunque esta es una variante que queda abierta en el artículo.

Del análisis de los factores explicados en el artículo anterior podemos afirmar que las principales ventajas del SIFT coinciden con los principales factores que afectan el proceso de reconocimiento de rostros tales como: cambios de iluminación, cambios en la pose, y la oclusión parcial del rostro. Por todo esto en este artículo se propone un acercamiento a las principales publicaciones que abordan este descriptor local en el proceso de extracción de rasgos locales en imágenes de rostro.

De forma general en la bibliografía se han identificado varios artículos [37, 40, 41, 42,43] que abordan el SIFT para la descripción de puntos locales de los rostros humanos y en los cuales se han propuesto mecanismos de reconocimiento los cuales difieren del original propuesto por Lowe[38], debido a que el reconocimiento de rostro presenta dificultades adicionales a las presentadas en los sistemas de detección y reconocimientos de objetos.

Del análisis de esto artículos y de las comparaciones propias de este algoritmo con otros algoritmos de extracción de rasgos podemos afirmar, que el algoritmo SIFT presenta buenos resultados en la comparación con otros métodos que fueron diseñados originalmente para el reconocimiento de rostros tales como: LBP, PCA y LDA, sobre todo en aspectos relativos a: cambios de iluminación, rotación y problemas de pose [37]. Sin embargo, la principal dificultad identificada en este algoritmo es la dimensionalidad de sus vectores y la cantidad de puntos que se extraen de cada rostro, lo cual influye negativamente en la complejidad computacional derivada de la correspondencia de rasgos en bases de datos de muchas imágenes de rostros. Por lo que se hace necesario pensar en soluciones que permitan disminuir las dimensiones de los descriptores SIFT sin afectar la precisión del reconocimiento de este algoritmo.

Algunos artículos propuestos en la bibliografía actual abordan la complejidad computacional derivada del proceso de correspondencia de los descriptores SIFT. Uno de los primeros artículos lo constituye PCA-SIFT [44], el cual combina la fase de detección de puntos invariantes del algoritmo SIFT con un algoritmo empleado para la reducción de dimensionalidad basado en el modelo de representación de rostros integral o *holistic*. En este artículo los autores proponen aplicar los tres primeros pasos del SIFT para detectar los puntos

invariantes presentes en las imágenes, además obtener los aspectos salientes de la imagen por la obtención del gradiente local calculado en la vecindad del punto. El principal aporte obtenido en esta técnica es referente a la sustitución del cuarto paso del algoritmo original donde se obtienen los descriptores de rasgos de 128 dimensiones y en el artículo mencionado los autores proponen normalizar el gradiente local para obtener descriptores de 20 dimensiones, lo cual implica una reducción significativa en la dimensionalidad de los vectores originales con el objetivo de acelerar la velocidad en el proceso de reconocimiento. La principal dificultad identificada en este artículo es que el problema es dirigido a la detección/ reconocimiento de objetos y no al reconocimiento de rostros, a pesar de que los autores mostraron en los pruebas experimentales que el PCA-SIFT además de obtener tiempos de correspondencia muy inferiores, también obtuvo por cientos de reconocimientos superiores al algoritmo SIFT original. Debido a esto se hace necesario ver que implicación tendría en el reconocimiento de rostros esta reducción de dimensionalidad por las diferencias que existen entre los dos procesos anteriormente mencionados.

En [40] los autores plantean que el algoritmo SIFT tiene como principal desventaja el proceso de correspondencia derivado de sus descriptores, ya que en cada imagen no se extraen la misma cantidad de puntos. Este aspecto tendría como consecuencia que el proceso de correspondencia de una imagen con respecto a varias imágenes previamente almacenadas sea muy costoso computacionalmente. En ese artículo que se propone el algoritmo SIFT en el reconocimiento de rostro, los autores plantean eliminar este efecto, dividiendo la imagen de rostro en varias regiones teniendo en cuenta las zonas más significativas del rostro tales como: ojo izquierdo, ojo derecho, nariz y ángulos laterales de la boca. Los puntos extraídos en el algoritmo SIFT son asignados a cada zona utilizando el algoritmo K-Means teniendo en cuenta su distancia al centro del grupo. De esta forma se garantiza que en cada zona exista aproximadamente la misma cantidad de puntos y que la correspondencia se efectúe por medio de las zonas fijas en cada imagen, acelerando así el proceso de reconocimiento. En las pruebas experimentales se valida la utilización del algoritmo SIFT en el reconocimiento de rostros, donde en efectos tales como cambios de pose y expresión facial supera a algoritmos clásicos de extracción de rasgos faciales como LBP [25].

En el artículo [41] se propone una nueva metodología para la reducción de dimensionalidades de los vectores SIFT, pero directamente aplicado al reconocimiento de rostros a diferencia del PCA-SIFT. La idea de este artículo es reducir el número de rasgos SIFT para disminuir el esfuerzo computacional derivado del reconocimiento. Los autores proponen para seleccionar los rasgos utilizar el poder discriminativo de cada rasgo. Plantean que un rasgo es discriminativo si se encuentra en más de una imagen de la misma persona, pero no aparece en imágenes de personas diferentes. Para reducir los rasgos SIFT los autores presentan Análisis Discriminantes de Fisher FDA, mediante el cual se escogen los rasgos que maximicen la siguiente tasa:

$$\text{Tasa de Fisher} = \frac{\text{variación entre clases}}{\text{variación dentro de las clases}}.$$

O sea se seleccionan los rasgos que tengan alta discriminación entre todas las imágenes de rostro y mínima variación dentro de la imagen de una misma persona. La idea general propuesta por los autores es utilizar el esquema de división de la imagen del rostro en celdas al igual que en el artículo [37], extraer rasgos y asignarlos a cada región de la imagen, para luego ordenar cada rasgo acorde a su nivel de discriminación y eliminar aquellos que presenten bajos valores.

Las pruebas experimentales mostradas por los autores demuestran que la velocidad de la correspondencia del algoritmo propuesto es superior a las propuestas en los artículos [37,40] al igual que la tasa de reconocimiento. Asimismo los autores plantean que la metodología propuesta, debido a los tipos de bases de datos analizadas, demostró ser robusta a cambios de iluminación, oclusión parcial del rostro y cambios de expresión facial.

A diferencia de los artículos anteriores donde se plantea que la principal dificultad del algoritmo SIFT es referente a la alta dimensionalidad de los vectores y a lo complejo que resulta el proceso de reconocimiento, en [42] los autores plantean que el algoritmo SIFT

propuesto en los artículos anteriores [37,40] no es óptimo para el reconocimiento de rostros debido a las principales diferencias que existen entre el reconocimiento de rostros y el reconocimiento de objetos. Donde fue planteado el SIFT originalmente estas diferencias se resumen de la siguiente forma:

Las imágenes de objetos de forma general son principalmente rígidos y presentan transacciones bien definidas entre sus diferentes lados, mientras que las imágenes de rostros no constituyen imágenes rígidas o sea se ven afectadas por los cambios de expresión, además son menos estructuradas.

De igual forma los objetos presentan estructuras más definidas que los rostros tales como líneas rectas o arcos, las cuales presentan puntos donde el nivel de contraste se define más que en los rostros. Estos puntos con bajos contrastes son más sensibles al ruido por lo que serán eliminados en el segundo paso del algoritmo propuesto. En los rostros como estas estructuras no están bien definidas se pondrían eliminar puntos con determinada discriminación. Debido a esto los autores proponen dos modificaciones a la implementación original del SIFT.

Una de las propuestas hechas por los autores se cataloga como *Keypoints Preserving* o KP-SIFT. Este algoritmo tiene como objetivo preservar todos los puntos detectados en el primer paso de la propuesta original, debido a esto los puntos que se eliminan en el segundo paso del algoritmo por estar débilmente localizados sobre los arcos permanecen en esta propuesta. La segunda propuesta se cataloga PD-SIFT o Descriptor Parcial SIFT. El algoritmo PD-SIFT tiene como objetivo evitar que el tamaño de los descriptores exceda el tamaño real de la imagen debido a los puntos que se preservan en la implementación anterior, por lo que sólo los descriptores contendrán los puntos detectados en largas escalas o los que estén más próximos a zonas más discriminativas del rostro. A pesar de que los resultados mostrados por los autores demuestran que tanto KP-SIFT como PD-SIFT mantienen tasas de error inferiores al SIFT original, presentan como principal dificultad el aumento significativo en la dimensionalidad de los vectores, aspecto negativo en los procesos de correspondencia entre muchas imágenes de rostro.

Del análisis de estos artículos que abordan el descriptor local SIFT en el reconocimiento de rostros podemos afirmar, que el mismo presenta buenos resultados con respecto a otros descriptores locales de gran precisión como LBP [25] y otros métodos de carácter integral como PCA y LDA. Sin embargo, las aplicaciones en tiempo real dedicadas al reconocimiento de rostros, requieren que los procesos de identificación o verificación se realicen con alta precisión y a la vez gran eficiencia.

Recientemente otro descriptor local utilizado para el reconocimiento de objetos al igual que el SIFT ha comenzado a ser utilizado en el reconocimiento de rostros [39] debido en gran medida a que los descriptores obtenidos mediante este algoritmo tienen la mitad de la dimensionalidad que los descriptores SIFT, o sea 64 dimensiones, el proceso de correspondencia es mucho más rápido. A continuación se muestra un resumen de las principales características de este algoritmo y las principales ventajas y desventajas identificadas con respecto al SIFT.

Comparación entre SIFT y SURF

El algoritmo *Speed Up Robust Features* SURF fue propuesto en [39] por Bay para la detección, y reconocimiento de objetos en determinadas escenas. El algoritmo SURF reutiliza algunos conceptos del SIFT, pero introduce nuevos conceptos para aumentar la velocidad de generación de los descriptores y correspondencia de los mismos. A diferencia del algoritmo SIFT, donde se obtienen descripciones de regiones basadas en la magnitud del gradiente local, en el SURF se utiliza la transformada de haar-wavelets en una vecindad 4x4 alrededor de un punto, con cuatro componentes por lo que el vector que se obtiene presenta 64 dimensiones. De forma general se puede apreciar que el algoritmo SURF presenta algunas ventajas con respecto al algoritmo SIFT en cuanto a la velocidad de generación de los puntos claves y en cuanto a la velocidad del proceso de correspondencia. En [48] los autores plantean que el proceso de generación de puntos claves en el algoritmo SURF es de 354 ms y para el SIFT el tiempo de generación de puntos claves es de alrededor de 1036 ms. A pesar de esas ventajas apreciadas del SURF con respecto al SIFT, es importante aclarar que eso no es condición suficiente para

afirmar que un algoritmo es superior a otro debido fundamentalmente a que estos dos algoritmos fueron diseñados con el objetivo de detectar y reconocer objetos en escenas determinadas, por lo que un aspecto a tener en cuenta es referente a los % de reconocimiento que se alcanzan en los mismos.

En cuanto a las tasas de reconocimientos que se alcanzan en estos algoritmos, en la bibliografía consultada existe diversidad de criterio en cuanto a cual de ellos es más óptimo en el proceso de reconocimiento en sentido general. En [39] los autores plantean que en los experimentos realizados el algoritmo SURF obtuvo una tasa de reconocimiento de alrededor de 82.6% mientras que el SIFT obtuvo una tasa de alrededor del 78.1%, sin embargo, en [48] los autores plantean que el SIFT archiva una tasa de reconocimiento de alrededor de 81.1% mientras que el SURF alcanza una tasa de 79.6%. Vale destacar que en los dos artículos anteriores estas comparaciones fueron realizadas con respecto a la detección y reconocimiento de objetos en determinadas escenas. Estas diferencias en gran medida se deben a los tipos de base de datos empleadas en los experimentos. Los distintos cambios a que están sometidas las imágenes no influyen de igual forma en los algoritmos SURF y SIFT.

Debido a que el algoritmo SURF archiva similares tasas de reconocimiento con respecto al algoritmo SIFT, y además supera a este último en cuanto a la velocidad de generación y correspondencia de puntos claves, recientemente también ha sido empleado en la representación de imágenes de rostros para reconocimiento mediante los rasgos locales extraídos.

4.2.4 Algoritmo SURF en el reconocimiento de rostros

Como se explicó anteriormente el algoritmo SURF debido a su capacidad de obtener poderosas descripciones locales y tiempos de ejecución superiores al SIFT, ha comenzado a ser utilizado en el reconocimiento de rostros. Los resultados mostrados por los autores demuestran que tanto el algoritmo SIFT como el SURF superan a métodos clásicos para el reconocimiento de rostros tales como: LBP, PCA, DCT, entre otros. De forma general mostramos un resumen de los artículos que abordan este descriptor en el reconocimiento de rostros.

En [49] se propone un sistema de detección y reconocimiento de rostros por medio de un robot móvil en tiempo real que es capaz a través de los descriptores SURF de procesar imágenes de rostros y archivar altos índices de reconocimiento. El reconocimiento de rostros se efectúa mediante los descriptores SURF extraídos de la región del rostro y los algoritmos de verificación son efectuados con respecto a una base de datos, donde las imágenes de rostros se encuentran previamente almacenadas. Los autores afirman que este sistema es capaz de efectuar efectivas detecciones y reconocimiento de rostros bajo diferentes condiciones tales como: cambios de iluminación y cambios de expresión, lo que garantiza su utilización aplicaciones en tiempo real.

En [51] se propone otro esquema para la utilización del algoritmo SURF en el reconocimiento de rostros. En este esquema los autores proponen extraer rasgos SURF de imágenes de rostros bajo diferentes criterios tales como: cambios de iluminación, oclusión parcial del rostro e imágenes de rostros no alineadas. De igual forma los autores utilizan dos tipos de descriptores SURF, uno de 64 dimensiones y otro de 128 al igual que el SIFT. En las pruebas experimentales los autores demuestran que el algoritmo SURF de 128 dimensiones es más robusto que la implementación de 64 dimensiones, de igual forma muestran que el algoritmo SURF presenta mejores resultados que el SIFT en cuanto a problemas de iluminación, mientras que el SIFT presenta mejor ejecución bajo cambios de expresión facial y cambios de puntos de vista.

En [50] los autores proponen el algoritmo SURF para el reconocimiento de rostros y plantean que el mismo alcanza tasas de reconocimiento comparables o superiores al SIFT, además el proceso de correspondencia entre imágenes es mucho rápido que en el algoritmo SIFT.

Como se explicó anteriormente los descriptores locales han ido cobrando mayor auge en los sistemas de reconocimiento de rostros por encima de los métodos holístico o integrales debido a

que son más robustos a cambios de pose, expresión e iluminación. Los descriptores locales SIFT y SURF originalmente propuestos para la detección/ reconocimiento de objetos de imágenes en determinadas escenas, han comenzado a ser utilizados en el reconocimiento de rostros obteniendo tasas de reconocimiento superior a métodos clásicos propuestos en esta temática [37].

En la bibliografía consultada existe diversidad de criterios en cuanto a qué método es superior a otro en % de reconocimiento. Algunos artículos afirman que el SURF alcanza % de reconocimiento superiores al SIFT [39], y otros afirman lo contrario [48] aunque las diferencias obtenidas son pequeñas. Por otra parte vale destacar que las comparaciones más sólidas de estos algoritmos son realizadas con respecto al reconocimiento de objetos [39,44,48]. Las comparaciones de estos algoritmos en el reconocimiento de rostros constituyen un problema abierto en la bibliografía actual debido a que los rostros no forman estructuras rígidas como los objetos, por lo que se ven afectados por cambios de pose y expresión facial además en los rostros las zonas más discriminativas se encuentran más definidas que en los objetos, por tanto, estos algoritmos deben ser modificados para obtener descripciones alrededor de estas regiones, de igual forma los mecanismos de reconocimiento son diferentes.

4.3 Principales técnicas de indexación aplicadas a imágenes de rostros

Debido a los grandes volúmenes que manejan las bases de datos rostros se hace necesario emplear eficientes mecanismos que permitan acelerar el proceso de reconocimiento. Estas estructuras son conocidas como esquemas de indexación. Las estructuras de indexación para el reconocimiento de rostros son abordadas desde diferentes puntos de vistas en la bibliografía.

4.3.1 Resumen de los esquemas de indexación de rostros

Los esquemas de indexación aplicados al reconocimiento de rostros no constituyen un aspecto muy abordado en la literatura actual, ya que los principales investigadores en esta materia están centrados en los aspectos relacionados con la precisión de este proceso, por lo que aspectos relacionados con la eficiencia de los algoritmos de verificación han quedado en un segundo plano. Debido a esto proponemos un breve resumen de los principales esquemas de indexación propuestos para acelerar el proceso de reconocimiento de rostros en grandes bases de datos de imágenes de rostros.

Unos de los primeros artículos que aborda la temática de la indexación en bases de datos de rostros lo constituye [30] donde los autores proponen una estructura de árbol para representar la descomposición de una imagen en símbolos, y mediante esta representación llevar a cabo el indexado y reconocimiento.

En [31] los autores presentan otro esquema de indexación basado en la extracción de palabras claves del rostro mediante técnicas semánticas y de esta forma se realiza el indexado. Otro esquema propuesto en la literatura es [32] donde los autores sugieren utilizar algoritmos genéticos para extraer rasgos de Gabor de personas específicas y mediante la representación de estos rasgos efectuar el indexado. En [33] los autores plantean un esquema de indexación y reconocimiento basado en la representación de 19 límites salientes del rostro en forma de grafo, por lo que una secuencia de nodos e índices es generada mediante esta técnica. En este esquema la principal dificultad identificada lo constituyen las técnicas de comparación de grafos.

En [34] se emplea una estructura de indexación aplicada a espacios vectoriales conocida como kd-tree donde los descriptores de rostros son almacenados en esta estructura para acelerar la ejecución de consultas. La principal dificultad identificada en este esquema se refiere a lo conocido como “Curse Of Dimensionality” o maldición de la dimensiones debido a la principal desventaja de los espacios vectoriales cuando las dimensiones de los vectores extraídos exceden el número de 20 coordenadas.

En [18] los autores expresan que todas las técnicas anteriormente mencionadas, presentan problemas de iluminación y no son robustas a cambios de pose y de expresión facial e identifican estos aspectos al igual que los cambios de perspectiva de las cámaras como uno de los principales problemas que afecta el reconocimiento de rostros.

En [18] los autores proponen de igual forma aliviar los problemas anteriormente mencionados utilizando un esquema de indexación basado en una técnica de similitud espacial, conocida como kd-tree. En las imágenes de rostros se anotan manualmente puntos dominantes para luego ser etiquetados de forma única acorde a su ubicación en el rostro. Técnicas de representación espacial se utilizan para la recuperación de las imágenes de rostros.

El proceso de indexación es uno de los aspectos más importante en el proceso de reconocimiento de rostros, sin embargo, el problema más abundado en la actualidad es referente a que los esquemas de reconocimiento de rostros tengan gran precisión en procesos tales como identificación y verificación. Pero en aplicaciones reales el proceso debe estar dirigido a lograr identificaciones positivas en grandes bases de datos, con tiempos de ejecución de consultas que satisfagan las necesidades de los usuarios. Debido a esto se hace necesario diseñar eficientes mecanismos que permitan acelerar los procesos de reconocimiento manteniendo las tasas de falsos positivos próximas a cero %.

El objetivo de este trabajo es buscar las soluciones más factibles para ubicar el proceso de reconocimiento de rostros en estructuras de indexación pertenecientes a espacios métricos. Partiendo de la hipótesis que son útiles para organizar grandes volúmenes de datos multimediales y de obtener tiempos óptimos en la ejecución de consultas similares [1,2,5].

4.4 Aplicación de los métodos de accesos métricos (MAM) en el reconocimiento de rostros

Los métodos de accesos métricos MAM constituyen uno de los aspectos más investigados en el campo de la optimización de consultas similares en bases de datos de objetos no tradicionales. Debido a la complejidad estructural de estos objetos y al aumento acelerado de la cantidad de estos en las bases de datos, los mecanismos de recuperación deben estar centrados en obtener la información con la mayor precisión posible, ya que el número de falsos positivos aumenta linealmente con el aumento de los objetos en una base de datos, además de esto acelerar los tiempos de búsquedas de los algoritmos de consultas similares existentes es de igual importancia para que el proceso sea óptimo.

Las estructuras de indexación en espacios métricos constituyen uno de los aspectos más investigados en la literatura en el campo de búsquedas por proximidad por una gran cantidad de investigadores alrededor del mundo. No obstante, los autores reflejan la utilización de estas estructuras desde un punto de vista genérico, debido a esto modelos de indexación que aborden la manera exacta en que se mida la similitud entre los objetos son escasos en la bibliografía consultada. El caso especial de los rostros toma una significación adicional debido a que los principales esquemas de indexación de estos objetos están dirigidos a otros campos tales como indexación basada en la segmentación de los rasgos más significativos y mediante estos segmentos se realiza el indexado, de igual forma se utiliza la indexación basada en contenido, donde se extraen características de los rostros y luego son etiquetados, esto permite la recuperación basado en el contenido de estos rostros. Otro esquema de indexación utilizado es la representación en el espacio de coordenadas de los rasgos extraídos de los rostros utilizando métodos de accesos espaciales conocido SAM⁵.

Las estructuras de indexación aplicadas a espacios métricos sólo definen una relación entre los objetos en base a sus respectivas distancias, por tanto relaciones geométricas espaciales entre los rasgos discriminativos de los rostros no son tomados en cuenta. Esquemas de indexación mediante MAM aplicados a los rostros no son muy abordados en las publicaciones

⁵ Estructuras de Indexación aplicadas a espacios vectoriales.

actuales. Por tanto valorar la aplicación de estas estructuras al proceso de rostros es un aspecto de vital importancia en esta investigación.

En [26] los autores proponen el esquema de las búsquedas similares en una base de datos de imágenes de rostros basados en técnicas de retroalimentación, donde mediante una imagen de consulta de rostro se obtiene una pequeña cantidad de imágenes similares mediante la ejecución del algoritmo K-NN. Luego este resultado es filtrado tanto por los usuarios o de forma automática. A pesar de que en este artículo el procedimiento se ejecuta de forma secuencial, o sea, no existe referencia a estructuras de indexación, sí es de gran importancia los aspectos desarrollados con respecto a los algoritmos de extracción de rasgos y la forma en que se mide la similitud entre los vectores de rasgos extraídos de cada imagen.

La extracción de rasgos en este artículo se lleva a cabo mediante rasgos de texturas. Los autores plantean que la información de los colores de los rostros no es importante en aplicaciones de recuperación de imágenes, por lo que todas las imágenes deben ser convertidas a escala de gray⁶. El algoritmo de extracción de rasgo de textura utilizado es Local Binary Pattern(LBP)[25], debido a esto la imagen es dividida en regiones y cada región se le asocia un histograma que representa los patrones obtenidos de cada región, luego el vector de rasgos global se obtiene por la concatenación de los histogramas locales para conformar un histograma global. La similitud es obtenida por medio de una métrica no definida por los autores que permita definir proximidad mediante los histogramas LBP globales, por tanto la ejecución de una consulta mediante el algoritmo K-NN encontrará las k imágenes de rostros más cercanos a una imagen de consulta dada.

En [19] a pesar de no estar aplicado a rostros, la estructura de indexación utilizada pertenece a los espacios métricos, los autores extraen vectores de rasgos de las huellas digitales, la estructura de indexación utilizada es el Slim-Tree[9], donde en cada nodo hoja se almacenan los descriptores de rasgos de cada huella. Los autores plantean que un aspecto clave en los sistemas de recuperación de la información en espacios métricos, son las medidas de similitud utilizadas, además de la calidad de los algoritmos de extracción de rasgos empleados. A pesar de que el objetivo del artículo sean las huellas digitales, puede ser utilizado en cualquier objeto biométrico, como rostros.

Recientemente importantes investigadores del campo de la aceleración de búsquedas similares en espacios métricos han propuesto un sistema de indexación distribuido con el objetivo de acelerar el proceso de reconocimiento de rostros nombrado MUFIN[47] o Red de Indexación Multi-Rasgos. Este sistema distribuido tiene como principal característica que es diseñado siguiendo el protocolo P2P, lo que garantiza un entorno paralelo para el cómputo de distancias entre los nodos presentes en esta infraestructura. Debido a que la implementación de este esquema de indexación distribuido está dirigido a los espacios métricos, el mismo tiene como principal ventaja la disminución de los C/C derivados del cálculo de distancias y la presencia de una infraestructura que permite una marcada escalabilidad para hacer frente al marcado crecimiento de los datos, lo cual posibilita su utilización en aplicaciones en tiempo real.

Este sistema se divide en tres capas fundamentales:

1. Datos y tipos de consultas.
2. Estructuras de indexación y algoritmos de búsquedas.
3. Infraestructura de ejecución del sistema.

En la primera capa están comprendidos todos los tipos de datos que manipula este sistema entre ellos objetos biométricos tales como: rostros, huellas digitales además de la recuperación de imágenes por contenidos. De igual forma se implementan los tipos de consultas K-Vecinos más Cercanos (K-NN) y las consultas por Rango de Distancia (Range Query), las cuales utilizan

⁶ Imágenes sin color o acromáticas, el rango de la escala de gray se expresa de cero(negro) a uno(blanca).

las métricas para su ejecución , debido a que el sistema está implementado sobre los espacios métricos. Por lo tanto la distancia en este espacio debe cumplir las tres reglas anteriormente señaladas: No Negatividad, Simetría y Desigualdad Triangular. La segunda capa tiene como característica fundamental la implementación de estructuras de indexación presentes en este sistema tanto centralizadas como distribuidas. Entre las centralizadas se destacan estructuras clásicas como M-Tree[6] y PM-Tree[8] mientras que en las distribuidas encontramos estructuras tales como GHT[53], M-Grid[52] y M-Chrod[28], todas estas estructuras utilizan mecanismos de indexación diferentes.

El sistema distribuido MUFIN fue aplicado directamente al reconocimiento de personas por sus rostros, por lo que los autores contaron con una base de datos de alrededor de 16 000 imágenes de rostros, las cuales fueron indexadas mediante descriptores de rasgos extraídos por el algoritmo PCA y la distancia utilizada para medir esta similitud fue la distancia Euclidiana. A pesar de que los autores no mostraron resultados que validen la utilización de este sistema en cuanto % de reconocimiento y velocidad con que se realizó este proceso, sí mostraron imágenes gráficas que permiten validar la efectividad de este proceso en la recuperación de imágenes de rostros similares a una imagen de rostros de consulta.

Este artículo a pesar de constituir uno de los pocos artículos identificados que aborden el reconocimiento de rostros desde el punto de vista de aceleración de este proceso mediante las estructuras indexación aplicadas en espacios métricos, carece de informaciones importantes referentes a tiempos, constitución de las estructuras de indexación, cantidad de distancias evaluadas, tiempo de ejecución de consultas de similitud, % de reconocimiento y costos por accesos a la información que se encuentra almacenada en discos. De igual forma en las pruebas experimentales se debió utilizar varios algoritmos de extracción de rasgos, al igual que los procesos de correspondencia se debieron efectuar utilizando varias métricas y no sólo la distancia euclidiana .De todas formas el artículo tiene como principal cualidad la inclusión del reconocimiento de rostros en los sistemas de indexación distribuidos, con el objetivo de disminuir los costos derivados de la evaluación de distancias en los algoritmos de consultas aplicados en las bases de datos de imágenes de rostros.

4.5 Resumen de las principales métricas utilizadas en el reconocimiento de objetos biométricos

Un elemento muy importante en los sistemas de indexación en los espacios métricos son las diferentes medidas de similitud que se emplean para medir distancia entre los vectores que se obtienen mediante los diferentes algoritmos de extracción de rasgos. La siguiente tabla muestra un resumen de algunas medidas de similitud que se emplean en estos sistemas incluyendo el campo de reconocimiento de rostros:

Medidas de Similitud	Ecuación
Bray-Curtis	$d_{BC} = \sum_{i=1}^N \frac{ X_i - Y_i }{X_i + Y_i}$
Canberra	$d_c = \sum_{i=1}^N \frac{ X_i - Y_i }{ X_i + Y_i }$
Euclidiana	$d_E = \sqrt{\sum_{i=1}^N (X_i - Y_i)^2}$

Manhatan	$d_M = \sqrt{\sum_{i=1}^N X_i - Y_i }$
Squared Chord	$d_{SC} = \sum_{i=1}^N (\sqrt{X_i} - \sqrt{Y_i})^2$
Squared Chi-Squared	$d_{SChi} = \sum_{i=1}^N \frac{(X_i - Y_i)^2}{X_i + Y_i}$

Las anteriores medidas de similitud fueron empleadas en el artículo [19] para la comparación e indexación de varios descriptores de rasgos pertenecientes a huellas digitales en un MAM como Slim-Tree[9], estos descriptores de rasgos fueron obtenidos mediante la Transformada Gabor-Wavelets. Por lo tanto podemos resumir que en las estructuras de indexación en espacios métricos no sólo se debe tener en cuenta como obtener los vectores de rasgos de los elementos indexados, sino también las medidas de similitud pertenecientes a espacios métricos que se emplean pues las mismas tienen influencia en la precisión con que se obtienen los resultados.

4.6 Estructuras de grafos aplicadas al reconocimiento de rostros

Los grafos tienen gran utilidad para el modelado de objetos complejos. Las estructuras de grafos han sido estudiadas en muchos campos tales como, reconocimiento de objetos [22] y reconocimiento de rostros [14,21] debido a su capacidad de modelar la relación estructural de objetos complejos.

Los grafos de forma general se pueden denotar como $G(V,E)$ donde V es una colección de vértices y E una colección de arcos y se denotan por $\text{attr}(v)$ y $\text{attr}(e)$ respectivamente. Las bases de datos de grafos se denotan $D = \{G_1, G_2, \dots, G_m\}$, donde m denota la cantidad de grafos existentes en la base de datos.

Existen varios artículos que abordan la utilidad de los grafos en el reconocimiento de rostros. Uno de los primeros fue propuesto en [14] donde cada imagen de rostro humano es representada por un grafo por medio de la técnica *Elastic Bunch Graph Matching* (EBGM), donde los nodos contendrán puntos claves de los rostros tales como: ojos, nariz y boca mientras que los arcos son empleados para modelar la relación estructural de estos nodos por medio de una distancia. La transformada de Gabor-Wavelets es empleada para extraer 40 coeficientes complejos de Gabor los cuales se localizarán en los vértices. El reconocimiento mediante esta técnica es realizado mediante una función de similitud de grafos, identificando los grafos más similares como los que maximicen esta función de similitud. De igual forma otros artículos han abordado los grafos en el reconocimiento de rostros. En [21] se propone otra forma de representación de imágenes de rostros como grafos, pero esta vez los vértices estarán representados por los rasgos extraídos por medio del algoritmo Análisis de Rasgos Locales (LFA). A diferencia del artículo anterior donde el reconocimiento es obtenido de forma secuencial en este artículo se utiliza un clasificador Bayesiano para podar el espacio de búsqueda y obtener una colección de grafos candidatos, luego se utiliza el algoritmo K-NN para obtener los grafos más próximos a un grafo de consulta dado. La medida de similitud empleada por los autores es la distancia Coseno, los resultados demuestran como esta técnica supera en cuanto a porcentaje de reconocimiento la propuesta anterior [14] y al algoritmo LBP [25].

Partiendo de los resultados apreciados en los artículos anteriores donde demuestran como los grafos pueden ser empleados en el reconocimiento de rostros debido a que son capaces de modelar la relación estructural de los mismos, proponemos el estudio de estas estructuras en los

espacios métricos, con el objetivo de acelerar el proceso de reconocimiento de rostros en bases de datos donde los objetos indexados sean imágenes de rostros modelados como grafos y un algoritmo de consulta recupere los grafos más similares a un grafo de consulta dado.

5 Conclusiones

Con la realización de este trabajo podemos concluir que a pesar de que la indexación en espacios métricos constituye uno de los aspectos más abordados en el campo de la aceleración de búsquedas por proximidad en bases de datos de objetos multimediales, su aplicación en el reconocimiento de rostros constituye un problema abierto en la literatura actual.

Aunque el reconocimiento de personas por su rostro es uno de los aspectos más investigados en el campo de la biometría, la mayoría de estos artículos abordan este problema desde el punto de vista de la precisión en los resultados por lo que aspectos relacionados con la eficiencia en este proceso quedan en un segundo plano. Se ha identificado en la bibliografía consultada que el reconocimiento de rostros ve limitada su implantación en sistemas de tiempo real por dos factores principales. El primero de ellos está referido a las dificultades intrínsecas a que se ven sometidos los rostros humanos tales como: cambios de pose, iluminación, cambios de expresión facial entre otros. Debido a esto los esquemas existentes para modelar la apariencia de los rostros tienen como objetivo fundamental obtener robustas representaciones mediante los rasgos faciales extraídos para efectuar precisos reconocimientos. En sistemas de tiempo real este proceso debe garantizar que los tiempos de ejecución de este reconocimiento estén acordes a los lugares donde estén siendo explotados. Las bases de datos de rostros por lo general están conformadas por millones de objetos y por eso los algoritmos de verificación existentes no garantizan obtener los resultados sin tener que recorrer la base de datos completa, aumentando de esta forma los costos computacionales y los tiempos de ejecución.

Como se explicó anteriormente el objetivo de este trabajo está centrado en las estructuras de indexación en espacios métricos con el propósito de acelerar la ejecución de los algoritmos de verificación en bases de datos de imágenes de rostros. Estas estructuras a lo largo de los años han demostrado ser eficientes para organizar objetos multimediales, definiendo relaciones de distancias entre los objetos almacenados, pero la evaluación de la distancia en una base de datos de millones de objetos es muy costosa computacionalmente, por lo que estas estructuras de indexación han ido evolucionando debido a que el concepto de grande hace diez años no es el mismo que en la actualidad. Por consiguiente las estructuras de indexación centralizadas desarrolladas para máquinas personales han ido evolucionando hasta alcanzar las estructuras de indexación distribuidas, donde un entorno paralelo garantiza que la ejecución de algoritmos consultas de similitud K-NN y Rango de Consulta por Similitud obtengan tiempos más acordes a los lugares donde estén siendo explotados, tales como: aeropuertos y puntos de control de pases, tanto reales como virtuales donde el dinamismo es vital para un correcto funcionamiento.

De forma general se ha identificado en la literatura que existen dos formas fundamentales de representar el rostro humano para efectuar reconocimiento: *Holistic* o Integral y Basado en los Rasgos Geométricos o Mediante Descriptores Locales *Features Descriptor*. Del análisis de la bibliografía consultada podemos concluir que los modelos basados en rasgos geométricos han obtenido mejores resultados [37,40,41] que los modelos Integrales o *Holistic* debido en gran medida a que las representaciones obtenidas relativas a regiones de la imagen presentan un mayor grado de discriminabilidad que la representación integral anteriormente mencionada. Esta diferencia provee al segundo modelo cierta robustez respecto a cambios de pose e iluminación con respecto al primer modelo. Entre los algoritmos de extracción de rasgos existentes se hizo énfasis en los descriptores locales SIFT [37] y SURF [38] propuestos originalmente para el reconocimiento de objetos en escenas determinadas, pero que recientemente han comenzado a ser utilizados en el reconocimiento de rostros pues proveen poderosas descripciones locales robustas a los cambios de iluminación, rotación y oclusión parcial presentes tanto en las imágenes de objetos como de rostros. Con respecto a estos

algoritmos se ha identificado que no existe un criterio definido en cuanto a cómo representar las imágenes de rostros mediante estos descriptores. De igual forma se han propuestos diferentes mecanismos de correspondencia [40,41,42] para efectuar reconocimiento existiendo gran diversidad de criterios en este sentido. También no existe diversidad de criterios en cuanto a las medidas de similitud empleadas siendo la distancia Euclidiana la más empleada en estos casos. En este sentido vale destacar que en nuestro caso existe la restricción de que estas medidas deben ser métricas o sea deben cumplir las tres principales propiedades de los espacios métricos [1].

De forma general con la realización de este trabajo y teniendo en cuenta los artículos consultados podemos definir las siguientes líneas de trabajos en el campo de la indexación en espacios métricos aplicadas al reconocimiento de rostros:

1. Ampliar la investigación de los principales mecanismos de consultas de los espacios métricos K-NN y Consulta por Rango de Similitud vía principal identificada para efectuar mecanismos de consultas en bases de datos de objetos multimediales, con el objetivo de proponer modificaciones para acelerar estos algoritmos atendiendo al tipo de estructura de indexación implementada.
2. Ampliar la investigación de las estructuras de indexación distribuidas con el objetivo de proponer algoritmos paralelos de consultas tanto K-NN como Consulta por Rango de Similitud que permitan obtener escalabilidad para hacer frente al marcado crecimiento de los datos.
3. Continuar investigando la aplicación de los descriptores locales SIFT y SURF en el reconocimiento de rostros, debido a que esta línea de trabajo a mi criterio y según la literatura consultada aún es incipiente sobre todo en las políticas de correspondencia desarrolladas.
4. Investigación de la dimensionalidad de los descriptores SIFT para definir en que sentido afecta esto al reconocimiento de rostros, debido a que en la literatura consultada existe diversidad de criterios en este sentido. De igual forma se propone un acercamiento al algoritmo PCA-SIFT [48] en el reconocimiento de rostros y obtener comparaciones con respecto a los algoritmos SURF [38] y SIFT [39].
5. Investigación de las estructuras de grafos en el reconocimiento de rostros debido a que son capaces de modelar las relaciones estructurales de los objetos entre ellos los rostros, vincular los algoritmos SIFT y SURF con estas estructuras.
6. Obtener implementaciones de esquemas de indexación que nos permitan realizar precisas comparaciones de los descriptores locales SIFT[38], SURF[39], LBP[25] y PCA-SIFT[48] mediante diferentes métricas, en el proceso de reconocimiento de rostros.
7. Diseñar un esquema de indexación distribuida en un entorno con características métricas mediante los rasgos de rostros extraídos mediante los algoritmos SURF y SIFT.

Referencias bibliográficas

1. Edgar Chávez, Gonzalo Navarro, Ricardo Baeza-Yates, and J. Marroquín. "Proximity searching in metric spaces". ACM Computing Surveys, 2001.
2. Marcos Zezula, Giuseppe Amato, Michal Batko and Vlastislav Dohnal. "Similarity Search The Metric Space Approach" .Springer Science+Business Media, Inc. 2006.
3. Edgar Chávez, Gonzalo Navarro. "An effective clustering algorithm to index high dimensional metric spaces". In Proceedings of the Seventh String Processing and Informational Retrieval c(SPIRE'00), IEEE CS Press 2000.
4. Marco Patella. "Similarity Search in Multimedia Databases". PhD thesis, Dipartimento di Elettronica Informatica e Sistemistica, Bologna, <http://www-db.deis.unibo.it/Mtree/index.html>, 1999.
5. Tomas Skopal. "Metric Indexing in Information Retrieval". VSB – Technical University of Ostrava, Faculty of Electrical Engineering and Computer Science. Department of Computer Science.2004.

6. Paolo Ciaccia, Marco Patella, and Pavel Zezula. "M-tree: an efficient access method for similarity search in metric spaces". In Proc. 23rd Conf. on Very Large Databases (VLDB'97), pages 426–435. Software available for download at <http://www.db.deis.unibo.it/Mtree/>. 1997.
7. Paolo Ciaccia, Marco Patella, and Pavel Zezula. "A cost model for similarity queries in metric spaces". In Proceedings of the Seventeenth ACM SIGACTSIGMOD-SIGART Symposium on Principles of Database Systems, June 1-3, pages 59–68. ACM Press, 1998.
8. Tomas Skopal. "Pivoting M-tree: A Metric Access Method for Efficient Similarity Search". In Proceedings of the 4th annual workshop DATESO, Desná, Czech Republic, ISBN 80-248-0457-3, also available at CEUR, Volume 98, ISSN 1613-0073, <http://www.ceur-ws.org/Vol-98>, pages 21–31, 2004.
9. Caetano Traina Jr, Agma Traina, Bernhard Seeger, and Christos Faloutsos. "Slim-Trees: High performance metric trees minimizing overlap between nodes". Lecture Notes in Computer Science, 1777, 2000.
10. Vlastislav Dohnal, Claudio Gennaro, Pasquale Savino, and Pavel Zezula. "D-index: Distance searching index for metric data sets". Multimedia ToolsApplications, 21(1):9–33, 2003.
11. Tomáš Skopal, Jaroslav Pokorný, and Václav Snasel. "Nearest Neighbours Search in Multimedia Databases using the PM-tree". In submitted, 12 pp., 2004.
12. Stephan Volmer, Fast Approximate Nears-Neighbor Queries in Metric Feature Space by Buoy Indexing. Fraunhofer Institute for Computer Graphics, Fraunhoferstr. 5, 64283 Darmstadt, Germany. 2002.
13. Karin Kailing, Stefan Schönaur, Martin Pfeifle, Hans-Peter Kriegel. "Extending metric index structures for efficient range query processing" Springer-Verlag London Ltd. 2006.
14. Laurenz Wiskott, Jean Marc Fellous, Norbert Krüger, Christoph von der Malsburg: "Face recognition by elastic bunch graph matching". IEEE PAMI pag (775–779).1997.
15. Claudia Leopold. "Parallel and Distributed Computing: A Survey of Models, Paradigms and Approaches". John Wiley & Sons, Inc. 2001.
16. Witold Litwin, Neimat Marie-Anne, Donovan A. Schneider. "LH* - a scalable, distributed data structure". ACM Transactions on Database Systems (TODS), 21(4):480- 525. ACM Press. 1996.
17. Sameer Antani, Rangachar Kasturi. "A survey on the use of pattern recognition method for abstraction, indexing and retrieval of images and video". Pattern Recognition, Volume 35, Issue4, April, Pages 945-965. 2002
18. Vikram T. N., Chidananda Gowda K., Guru D. S., Shalini R. Urs, "Face Indexing and Retrieval by Spatial Similarity," cisp, vol. 1, pp.543-547, 2008 Congress on Image and Signal Processing, Vol. 1, 2008
19. Javier A. Montoya, Neucimar J. Leite, Ricardo da S. Torres. "Wavelet-based Feature Extraction for Fingerprint Image Retrieval". Technical Report. Instituto de Computacao, Universidade Estadual de Campinas. Brasil. 2006.
20. Ehsan Flasz-Ersi, John S. Zelek and John K. Tsotsos. "Robust Face Recognition through Local Graph Matching". Journal Of Multimedia, Vol 2. 2007.
21. Huet, B., Cross, A., Hancock, E.: Shape retrieval by inexact graph matching. In: Proc. IEEE Int. Conf. on Multimedia Computing Systems. Volume 2., IEEE Computer Society Press.1999.
22. Kubicka, E., Kubicki, G., Vakalis, I.: Using graph distance in object recognition. In: Proc. ACM Computer Science Conference. 1990
23. Huahai He and Ambuj K. Singh : "An index structure for graph queries". University of California , Santa Bárbara. 2006.
24. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, "A.Face recognition: a literature Survey". ACM Computing Surveys 35(4), 399–458 .2003.
25. Ahonen, T., Hadid, A., Pietikäinen, M.: Face Recognition with Local Binary Patterns. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 469–481. Springer, Heidelberg 2004.
26. A, Franco. D, Maio.: Similarity Searches in Face Databases. DEIS – Università di Bologna, Viale Risorgimento, 2 – 40136 Bologna – Italy. 2009 .
27. F.Falchi. C,Gennaro . P. Zezula.: A Content- Addressble Network for Similarities Search in Metric Spaces. ISTI-CNR, via G. Moruzzi 1, 56124 Pisa, Italy.2007.
28. D, Novak and P,Zezula: M-Chord : A Scalable and Distributed Similarity Search Structure. Masaryk University, Brno, Czech Republic.2006.
29. H. V. Jagadish, B. C. Ooi, K.-L. Tan, C. Yu, and R. Zhang. iDistance: An adaptive B -tree based indexing method for nearest neighbor search. ACM Transactions on Database Systems (TODS 2005), 30(2):364 397, 2005.

30. Imran Ahmad and William I. Grosky. Indexing and retrieval of images by spatial constraints. *J. Vis. Commun. Image R.* 14. 2003.
31. Hideaki Ito and Hiroyasu Koshimizu. Keyword and Face Image Retrieval Based on Latent Semantic Indexing. *IEEE International Conference on Systems, Man and Cybernetics.* 2004.
32. Sreekar Krishna, John Black and Sethuraman Panchanathan. Using Genetic Algorithms to Find Person- Specific Gabor Feature Detectors for Face Indexing and Recognition. *LNCS 3832.*2005.
33. Mineo Yoshino, Mari Taniguchi, Kazuhiko Imaizumi, Sachio Miyasaka, Toyohisa Tanijiri, Hiroshi Yano, C. David, L. Thomas and John G. Clement. A new retrieval system for a database of 3D facial images. *Forensic Science International.*2005.
34. Yongsheng Gao and Yutao Qi. "Robust visual similarity retrieval in single model face databases". *Pattern Recognition.* 2005.
35. Yuchun Fang and Donald Geman. "Experiments in mental face retrieval". In proceedings of AVBPA , *LNCS 3546*, pp. 2005.
36. Shu-Sheng Liaw, Geeng Neng You, Fu-Song Syu and Hsiu-Mei Huang. A new knowledge based indexing system through the internet. *ICACT.* 2006.
37. Manuele Bicego, Andrea Lagorio, Enrico Grosso, and Massimo Tistarelli, "On the use of SIFT features for face authentication," *Proc. of IEEE Int Workshop on Biometrics*, in association with CVPR, NY, 2006.
38. David Lowe, "Distince image features from scale-invariant keypoints," *Int. Journal of Computer Vision*, ,2004.
39. Herbert Bay, Tinne Tuytelaars, Luc Van Gool. "SURF: Speeded Up Robust Features". 9th European Conference on Computer Vision.2006.
40. Jun Luo, Yong Ma, Erina Takikawa, Shihong Lao, Masato Kawade, and Bao L. Lu, "Person-Specific SIFT Features for Face Recognition", *International Conference on Acoustic, Speech, and Signal Processing (ICASSP)*, 2007.
41. Majumdar , R.K. Ward. "Discrimantive SIFT features for Face Recognition", *Department of Electrical and Computer Engineering, University of British Columbia.*2008 .
42. Cong Geng, Xudong Jiang. "SIFT Feaures for Face Recognition."2nd IEEE Internacional Conference on Computer Science and Information Technology.2009.
43. Han Yanbin, Yin Jianqin, Li Jinping, "Human Face Feature Extraction and Recognition Based on SIFT," *iscsct*, vol. 1, pp.719-722, 2008 *International Symposium on Computer Science and Computational Technology*, 2008.
44. Yan Ke and Rahul Sukthankar "PCA-SIFT" : A More Distinctive Representation for local Image Descriptors". *Computer Vision and Pattern Recognition*, 2004.
45. Chengjun Liu and H. Wechsler, "Gabor Feature Based Classification Using the Enhanced Fisher Linear Discriminant Model for Face Recognition," *IEEE Trans. Image Processing.*2002.
46. Xiangyang Jiang, Peiheng Zhang "Survey on index based homology search algorithms " *Kluwer Academic Publishers.* 2007.
47. Pavel Zezula. Michal Batko. Vlastislav Dohnal. David Novak. "Similarity Search in Large Collections of Biometric Data".*Faculty of Informatics, Masaryk Univeristy.* 2009.
48. Paul Fogg, Gilbert Peterson, and Michael Veth . "Image Background for Identitifyng Suspects" in *IFIP Internacional Federation for Inormation Processing, Volume 285 Advances in Digital Forensics IV.* 2008.
49. Shan An, Xin Ma, Rui Song. "Face Detection and recognition with Surf for human-robot interaction". *Sch. Of Control Sci. & Eng., Shandong Univ, Jian, China. Automation and Logistics,ICAL.*IEEE International Conference. 2009.
50. Du Geng, Su Fei, Cai Anni."Face Recognition using SURF features ". *MIPPR : Pattern Recognition and Computer Vision. Proceedings of SPIE.* 2009.
51. Philippe Dreuw, Pascal Steingrube, Harald Hanselmann."SURF –FACE: Face Recognition Under Viewpoint Consistency Constraints". *Human Language Technology and Pattern Recognition RWTH Aechen, Germany.* 2009.
52. Michal Batko, Vlastislav Dohnal and Pavel Zezula. M-Grid: Similarity Searching in Grids. In *Proceedings of International Workshop on Information Retrieval in Peer-to-Peer Networks, ACM CIKM 2006.* Arlington, ACM Press. 8 pages, 2006. ISBN 1-59593-531-2. 2004.
53. Michal Batko, Claudio Gennaro and Pavel Zezula. Scalable and Distributed Similarity Search in Metric Spaces. In *5th Workshop on Distributed Data & Structures. Revised Selected Papers.* Canada, Carleton Scientific. 10 pages, 2004. ISBN 1-894145-18-6. 2006.

RT_030, abril 2010

Aprobado por el Consejo Científico CENATAV

Derechos Reservados © CENATAV 2010

Editor: Lic. Lucía González Bayona

Diseño de Portada: DCG Matilde Galindo Sánchez

RNPS No. 2142

ISSN 2072-6287

Indicaciones para los Autores:

Seguir la plantilla que aparece en www.cenatav.co.cu

C E N A T A V

7ma. No. 21812 e/218 y 222, Rpto. Siboney, Playa;

Ciudad de La Habana. Cuba. C.P. 12200

Impreso en Cuba

