Facultad 10, Junio del 2009

Implementación de algoritmos de reconocimiento de imágenes duplicadas.

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:

Javier Chirino Sánchez

Adrián Alberto Mesa Pujals

Tutor:

Ing. Ailyn Alfonso González

"...Si no eres parte de la solución, entonces eres parte del problema .Actúa!!"

Vladímir Ilich Lenin

Agradecimientos

De Javier:

A mis padres por darme la vida, el apoyo moral y por la confianza que depositaron en mí.

Al muñeco de mi hermano.

Al Charly mi otro hermano y a Lien que me sobornó.

A Leydis por tantas horas de espera, por acompañarme, pelearme y entenderme en todo este tiempo de estrés.

A todos mis innumerables amigos por soportarme todos estos años y hacer mi estancia en la UCI digna de repetirse en otra vida.

A Ailyn mi tutora, siempre tan buena.

A Manuel por sus consejos.

De Adrian:

A mis padres por su confianza y apoyo incondicional, constituyen todo lo que soy hoy

A mi hermano uña y carne así será siempre.

A mi familia, a mi novia la elegida “la mía”, a mi primo, mis amistades “los muchachones” que están en las buenas y en las malas.

A mi tutora y al “Manu” jefe de proyecto por ser los dos ejemplos profesionales.

Dedicatoria

De Javier:

Dedico este trabajo de diploma a mis padres que tanto se han esforzado por hacer este, mi sueño, realidad, a mi hermano que le sirva de guía y de apoyo para que logre los suyos, a mis abuelos que siempre me llenaron de cariño y fuerza.

A todos los involucrados con el Proyecto “Informatización de la Prensa”.

De Adrian:

Dedico este trabajo de diploma a mis padres, hermano, a mi familia toda que contribuyeron en hacer mi y nuestro sueño realidad.

A todos los involucrados en el Proyecto “Informatización de la Prensa”.

Resumen:

Las imágenes constituyen un componente esencial en un sistema de prensa, estas son mayormente usadas tanto para darle más veracidad a los temas como para hacer más atractivo el contenido a la vista de los usuarios. Sin embargo, también contribuyen a la existencia de contenido duplicado, lo cual constituye un problema a la hora de gestionar el trabajo con imágenes. Por esta razón en la actualidad se han desarrollado algoritmos que van dirigidos al reconocimiento y detección de imágenes duplicadas a las cuales les hayan sido alterados algunos parámetros. Este trabajo se centra en la implementación de algunos de estos algoritmos, adaptándolos a las necesidades especificas de los sistemas de prensa con el objetivo de integrarlos a Pyxel (Acosta, 2009), el cual es un archivo de imágenes fotográficas especialmente diseñado para el trabajo de medios de prensa.

Palabras Claves

Sistema, detección, duplicados, imágenes, *wavelets*, momentos, Hu, algoritmo, implementación, tesis, Pyxel

Índice

Introducción:

La función de un sistema de prensa no es únicamente informativa, en el se comparan y contraponen unos sucesos con otros, se argumenta, concluye y plantean soluciones. En todo este proceso juega un papel importante el uso de imágenes producto de la necesidad de darle más veracidad a los temas expuestos.

Históricamente en un sistema de prensa siempre prima la variable inmediatez, más aún en la era que estamos viviendo donde el flujo de información es tan grande y violento. Por lo tanto no es raro encontrar en varios productos noticiosos la misma imagen. Lo realmente preocupante de esto es que no existe una traza entre la original y su o sus copias.

Por otra parte los duplicados de las imágenes o copias de estas no son siempre fieles exactamente desde el punto de vista binario a la original, sino que mayormente han sido tratadas con alguna herramienta para adecuarlas al medio de publicación: papel periódico(blanco y negro), web, cambios resolución, formato, iluminación, contraste, colores, entre otros. Incluso en algunos contextos de prensa se prefiere escoger la zona caliente de la imagen original recortándola a gusto para darle uso; por estas razones una comparación a nivel binario en muchos casos no es suficiente para la detección de duplicados.

Todo esto trae consigo que no se usen eficientemente los recursos de almacenamiento ya que al no existir para los trabajadores de la prensa una forma factible de comprobar si una imagen ya existe en el repositorio o con cuantas copias de las mismas se cuenta, simplemente introducen las imágenes en muchos casos duplicando las ya existentes. El trabajo de detectar manualmente estas imágenes duplicadas es sumamente tedioso e ineficiente y no existe actualmente en los medios de prensa una forma de hacerlo automáticamente.

Dada la situación problémica planteada anteriormente, donde se evidencia la necesidad de implementar módulos para la detección de imágenes duplicadas en un sistema de prensa, surge el siguiente **problema científico**: ¿Cómo detectar y eliminar imágenes duplicadas en los medios de prensa cubanos, aún cuando algunos de sus parámetros hayan sido alterados?

Los algoritmos de detección de duplicados de imágenes son el **objeto de estudio,**

tomando como **campo de acción** la implementación de algoritmos de detección de duplicados de imágenes.

Para este trabajo se plantea, como **objetivo general**, diseñar e implementar una biblioteca de algoritmos para la detección de imágenes duplicadas, surgen entonces los siguientes objetivos específicos:

1. Realizar un estudio del estado del arte en algoritmos de detección de imágenes duplicadas (incluyendo algoritmos paralelos).
2. Seleccionar 3 algoritmos para implementarlos.
3. Implementar los algoritmos en el lenguaje Python.
4. Evaluar los algoritmos en cuanto a tiempo de procesamiento y recursos consumidos.

Para darle cumplimiento a estos objetivos se han definido las siguientes tareas investigativas:

1. Estudiar herramientas libres para acelerar el desarrollo del presente trabajo.
2. Estudiar trabajos similares para poner en práctica la reutilización de los componentes.
3. Estudiar las características de la imagen
4. Estudiar los algoritmos paralelos.
5. Estudiar las diferentes ventajas que nos ofrece el lenguaje *Python*, tales como librerías entre otros.
6. Desarrollar la implementación de los algoritmos en el lenguaje *Python.*
7. Seleccionar una colección de imágenes para realizar las pruebas.
8. Realizar las pruebas.

9. Documentar los resultados obtenidos en las pruebas.

Para la realización de la presente investigación se utilizaron los siguientes métodos científicos de investigación:

1. **Analítico-Sintético**: Se buscaron y analizaron diferentes documentos, algoritmos y teorías acerca de la detección de imágenes, extrayendo los elementos fundamentales relacionados con el objeto de investigación.
2. **Inductivo-Deductivo**: A partir de este método se obtuvieron conocimientos generalizados sobre la detección de imágenes a partir del análisis de lo particular a lo general.

El presente documento se estructura en dos capítulos, además de las secciones de Conclusiones, Recomendaciones, Bibliografía, Referencias Bibliográficas, Glosario de Términos y Anexos

Capitulo 1-Fundamentación teórica: Se da una breve descripción sobre las tendencias actuales existentes sobre la detección de duplicados, se describen algunos conceptos y características de los sistemas de detección de duplicados que se deben tener en cuenta para la implementación de los mismos. Se realiza un análisis de las tecnologías de software a emplearse para desarrollar el sistema.

Capitulo 2- Algoritmos propuestos para la detección de imágenes duplicadas: En este Capítulo se analizarán los algoritmos candidatos. Se hará hincapié en los aspectos relacionados con su funcionamiento, comportamiento y comprensión, además de documentarse las pruebas efectuadas a los mismos.

Capitulo 1 Fundamentación Teórica

El problema de la creación de duplicados de la imagen procede de diferentes campos informáticos. Como resultado o consecuencia han surgido diferentes definiciones de problemas y soluciones relacionados con este tema. En este capítulo se definirán brevemente los problemas de detección de duplicados. Sé explicará el origen de la detección de duplicados. Se hará mención de algunos sistemas de recuperación de imágenes existentes. Se presenta y se analizan métodos de detección de duplicados candidatos a implementar dadas sus características. Además se describen las herramientas tecnológicas utilizadas para desarrollar el presente trabajo.

1.1 ¿Qué es la detección de un duplicado?

Empezaremos por explicar que un duplicado no es más que una versión transformada de una obra original, la cual tiene un parecido apreciable de valor. Es una relación de equivalencia por pares que vincula el original a cualquiera de sus variaciones a través de una operación de transformación, por ejemplo compresión, cambios de brillo o cultivo.

Si una imagen A es duplicado de otra imagen B y aún otra imagen C es duplicado de la imagen B, entonces esta imagen C es duplicado de la imagen A.

El objetivo de la detección de duplicado de imágenes consiste en encontrar todos los duplicados de una imagen en particular entre una colección de imágenes, es detectar si dos imágenes son duplicadas una a la otra o relacionadas.

1.2 Orígenes de la detección de imágenes duplicadas basada en contenido

En los últimos años se ha visto un rápido incremento del tamaño de las colecciones de imágenes digitales (médicas, históricas, policiales, pictóricas, entre otras). Esta situación plantea una necesidad básica: la búsqueda y obtención de las imágenes deseadas entre una extensa colección.

Los métodos tradicionales para la clasificación de la información de una imagen (anotaciones textuales, establecimiento de taxonomías) se muestran insuficientes e inadecuados por diversos motivos. (M.C. Aranda, 2002)

Básicamente podemos destacar dos problemas: la cantidad de trabajo que requiere asociar un texto a cada imagen y expresar el rico contenido de una imagen en texto no es fácil y además es subjetivo. Por otra parte la subjetividad e imprecisión de las anotaciones pueden causar errores en la recuperación. Para solucionar esos problemas surgió la recuperación de imágenes basada en el contenido (*content-based image retrieval CBIR*). En ella las imágenes son indexadas y recuperadas por su propio contenido visual, como el color o la textura. La recuperación de imágenes basada en el contenido se fundamenta en 3 aspectos: la extracción de características visuales, la indexación multidimensional (para facilitar una búsqueda rápida) y el diseño del sistema de recuperación. (M.C. Aranda, 2002)

El termino CBIR parece tener su origen a principios de los 90, cuando fue utilizado para describir experimentos en la recuperación automática de imágenes de una base de datos, basado en colores y formas. Desde entonces, el término se ha utilizado para describir el proceso de recuperación de imágenes deseadas dentro de una gran colección sobre la base de las características sintácticas de la imagen. Las técnicas, herramientas y algoritmos que se utilizan proceden de ámbitos tales como estadísticas, reconocimiento de patrones, y procesamiento de señales.

El ideal de un sistema CBIR desde la perspectiva del usuario implicaría lo que se denomina semántica de recuperación, donde el usuario realiza una solicitud como "encontrar fotos de perros" o incluso "encontrar imágenes del Che". Este tipo de tarea resulta difícil para las computadoras llevarla a cabo, fotos de chihuahuas y de Gran daneses lucen muy diferentes, y el Che no siempre le da el frente a la cámara o aparece en poses diferentes. Por tanto los sistemas CBIR actuales se centran en características de bajo nivel como la textura, el color y la forma, aunque algunos sistemas toman ventaja de las características de muy alto nivel común como caras

Ejemplos de aplicaciones del CBIR son:

- **Prevención de crimen**: los sistemas de reconocimiento automático, utilizados por las fuerzas de la policía.
- **Comprobación de seguridad**: Huellas digitales o la digitalización de la retina para privilegios de acceso
- **Diagnóstico médico**: Uso del CBIR en una base de datos de imágenes

médicas de diagnóstico para ayudar a identificar los casos anteriores similares.

- **Propiedad intelectual**: registro de marcas de imagen, donde una marca de un nuevo candidato se compara con las marcas existentes para asegurar el riesgo de confundir la propiedad.

1.2.1 Formas de representación del contenido visual

Las características visuales de la imagen se clasifican en generales y específicas. Estas últimas dependen de la aplicación. Las características visuales generales que son usadas en la mayoría de las aplicaciones son:

Color: Es una de las características más usadas. Es relativamente robusta a las variaciones del fondo e independiente del tamaño y orientación de la imagen. Las técnicas más usadas como representación del color son el histograma de color, el histograma de color acumulado, los momentos de color y los conjuntos de color. (M.C. Aranda, 2002)

Textura: Se refiere a patrones visuales homogéneos formados por diversos colores o intensidades. Es una propiedad innata de prácticamente todas las superficies como nubes, árboles, pelo o ladrillos. Las características de textura se suelen representar usando una matriz de concurrencia, propiedades psicológicas (contraste, regularidad, tosquedad, aspereza, entre otros), transformadas wavelet entre otras. (M.C. Aranda, 2002)

Formas de objetos: Algunas aplicaciones requieren que la representación de la forma sea invariante a traslación, rotación y escalado mientras que otras no. En general, las representaciones de la forma se dividen en dos categorías: las basadas en contornos y las basadas en regiones. Las primeras usan sólo el contorno exterior de la forma mientras que las segundas usan la región de la forma completa. Se han desarrollado numerosos métodos para ambas categorías, pero los más representativos son los descriptores de Fourier (transformada de Fourier del contorno) para la primera, y los momentos invariantes (momentos basados en regiones que sean invariantes a transformaciones) para la segunda. (M.C. Aranda, 2002)

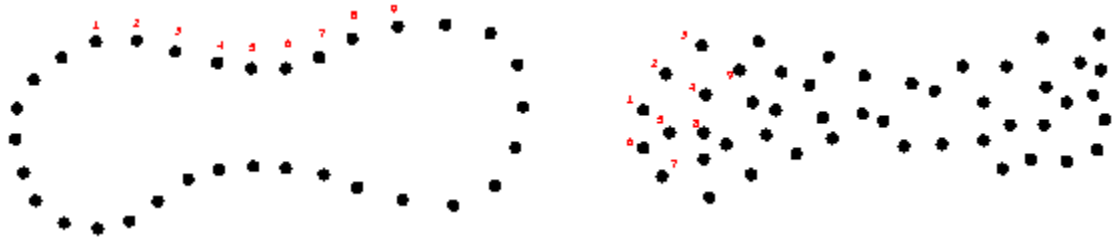


Fig. 1 Basado en contorno y en región

Diseño de color: Se trata de usar conjuntamente la característica de color y las relaciones espaciales. Una aproximación sencilla es dividir la imagen en bloques y extraer las características de color de cada bloque. Otra aproximación es segmentar la imagen en regiones con características de color destacadas y luego almacenar el conjunto de características de color y la posición de cada región. Su desventaja es la problemática que supone la segmentación de una imagen. Otras técnicas son usar momentos de color sobre regiones, usar una matriz de concurrencia de color. (M.C. Aranda, 2002)

Existen numerosos sistemas de recuperación de imágenes basados en el contenido, los cuales incluyen alguna o varias de las características anteriores para hacer la búsqueda.

1.3 Estado del arte de las técnicas de recuperación de imágenes por contenido

El objetivo de las técnicas de consulta basadas en contenido de las imágenes es encontrar, de forma eficiente, las imágenes en una base de datos de imágenes que son similares a una indicada. A diferencia de las típicas consultas en bases de datos, en este caso se utiliza un criterio de similitud que no tiene porque ofrecer una coincidencia exacta. Con el objetivo de ser capaces de describir los contenidos de una imagen, sobre la base de las características y/o representaciones de interés con la pretensión de abordar el diseño de un sistema de consultas por contenido en bases de datos de imágenes, se han venido utilizando diferentes aproximaciones a la temática de reconocimiento de objetos en imágenes, en las que es posible distinguir: búsqueda de primitivas visuales de bajo nivel (incluyendo características como color y texturas), pasando por estrategias de aprendizaje y agrupamiento jerárquico e incluyendo la

capacidad de manipular objetos genéricos en contextos y configuraciones no predeterminadas (Etemad Kamran, 1997). En la Figura 2 se relacionan las diferentes técnicas o aproximaciones que se emplean para esta temática y que se comentan a continuación.

Es posible distinguir tres tipos de descriptores que permiten un análisis de las imágenes a tres niveles con diferente capacidad de abstracción y que son:

- Nivel1: primitivas básicas. En este caso la posibilidad de establecer relaciones entre imágenes se realiza en base a valores de color de los puntos, a las texturas que se pueden observar y a los contornos que es posible detectar con diferentes operadores.
- Nivel 2: sintáctico. Denominado "reconocimiento de patrones" por aquí se unen a las anteriores informaciones para buscar regiones de mayor entidad y comprobar si se ajustan a reglas de descripción de elementos tipo.
- Nivel 3: semántico. Llamado "reconocimiento de objetos" en el que la complejidad es un poco mayor al permitir que existan imprecisiones entre la definición del objeto de la realidad a buscar y las posibles trasformaciones sufridas en el paso a 2D, las variaciones de las condiciones de adquisición de la imagen, entre otros (Manuel Agustí i Melchor).

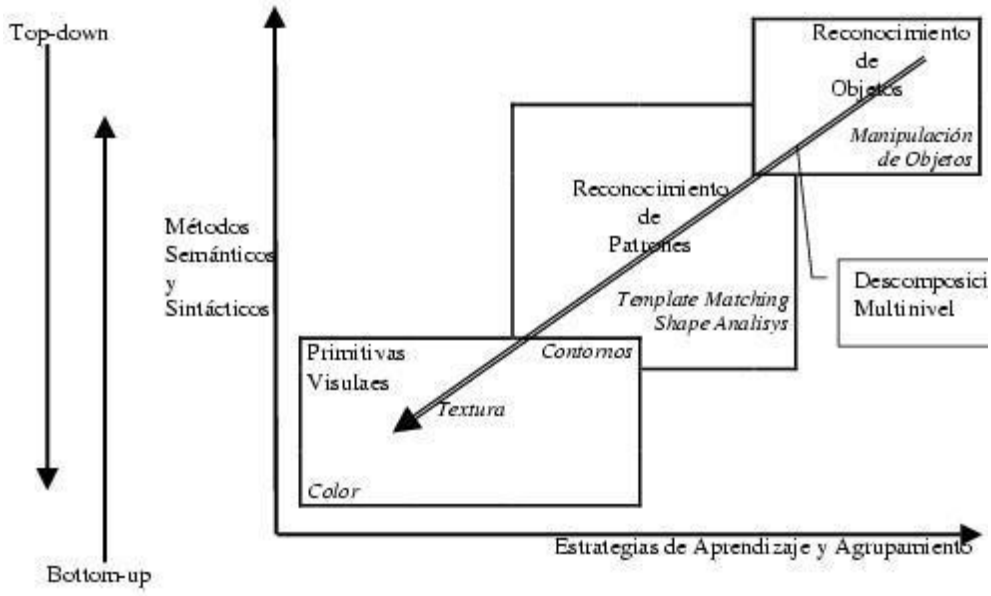


Fig. 2 Técnicas para la descripción del contenido de una imagen

Los sistemas de consultas en bases de datos de imágenes se basan, tradicionalmente, en que realizan un pre-proceso en la imagen encaminado a obtener un conjunto de descriptores y utilizar estos para la búsqueda en la base de datos de imágenes. De esta forma se elimina el coste de realizar el análisis de contenidos de los elementos que componen la base de datos en cada consulta, pero por otra parte limita las búsquedas a aquellas que se pueden realizar con la combinación de los descriptores utilizados. De manera que este tipo de análisis es efectivo para conjuntos de imágenes en los que los descriptores satisfacen cualquier consulta permitida como por ejemplo en bases de datos de caracteres, caras, representantes de categorías (solo gatos, peces, sillas, entre otros). Todas estas se caracterizan porque el motivo de las mismas es único, ocupa gran parte de la escena, es fácilmente distinguible del fondo y se puede normalizar los valores de color, tamaño de objeto o su posición. Pero no es suficiente en aquellas colecciones de imágenes con grandes riquezas de detalles que no es representable de forma compacta por los descriptores cuyas características son: la falta de control sobre las condiciones con que se obtienen las imágenes (posición de la cámara, iluminación, movimiento de la escena, etc.), la gran variabilidad de las características de los objetos de interés (color, tamaño, posición, entre otros) y la existencia de partes ocultas.

En los últimos años, varios sistemas prototipos se han propuesto abordando diferentes aspectos de la información contenida en las imágenes, como son las texturas, similitud de formas y relaciones semánticas entre objetos de la imagen. El objetivo de las técnicas de consulta basadas en contenido de las imágenes es encontrar de forma eficiente las imágenes en una base de datos de imágenes que son similares a una indicada. A diferencia de las típicas consultas en bases de datos, en este caso se utiliza un criterio de similitud que no tiene porque ofrecer una coincidencia exacta. (Arnold W.M. Smeulders, 2000)

La importancia de los elementos visuales depende de la subjetividad del observador, por lo que es necesario desarrollar criterios de búsqueda capaces de manejar imprecisiones y la falta de información en las descripciones que propongan los usuarios. También depende del contexto de la aplicación, así que el hecho de que se conozca de antemano puede ayudar.

En la Tabla 1 se muestran, de forma resumida, los sistemas analizados y los conjuntos de descriptores separados por su nivel de información asociado.

Sistema	Descriptores	
	Bajo	Medio
Netra	color, textura	forma, regiones, localización espacial
PickToSee	Color, contornos	invariantes geométricos
CANDID	color, textura	forma, firmas e histogramas globales
Chabot	color	textuales (fechas, localización geográfica), vectores de longitud variable
WebSeek	color	texturales, configuraciones espaciales, regiones, histogramas globales
QBIC	color, textura	textuales, forma, rectángulos de color orientados, regiones, posiciones espaciales
PhotoBook	color, textura	forma, regiones

Tabla 1 Descriptores de Sistemas de recuperación de Imágenes

1.4 Análisis de sistemas existentes

Robustos métodos automatizados de detección de duplicados de imágenes están tomando más atención recientemente debido al crecimiento exponencial de contenido multimedia en la web. La cantidad de datos de multimedia es inviable para vigilarlos manualmente. Además, violaciones de derechos de autor y piratería de datos son cuestiones importantes en muchas áreas que incluyen la gestión de derechos digitales. Actualmente existen diversos sistemas que se enfocan en la recuperación de la información visual, entre ellos podemos mencionar:

- AMORE.
- VisualSeek.
- QBIC.
- SQUID.

1.4.1 AMORE

El sistema AMORE (*Advanced Multimedia Oriented Retrieval Engine*) (ver Anexo 3.)

fue realizado por NEC USA Inc. Permite la recuperación de imágenes vía web. Las consultas en AMORE pueden ser formadas por palabras claves, por especificación de imágenes similares o por combinación de ambas. La principal característica de esta herramienta es segmentar una imagen en ocho regiones de colores homogéneos donde estas regiones son usadas directamente para la indexación. Las consultas de este sistema se realizan de la siguiente manera:

1. Un usuario selecciona una categoría de imágenes, puede ser seleccionada de manera aleatoria o por medio de una palabra clave.
2. El sistema muestra las imágenes más similares a la consulta.
3. El usuario puede indicar la importancia relativa de color y forma.

Para la similitud de forma primero se busca una correspondencia entre regiones de la imagen, basándose en el número de píxeles que se traslapan. Para la similitud de color entre dos regiones se usa la distancia HLS (*Hue, Saturation, Lightness*).

La recuperación se basa en:

- Segmentación de la imagen en regiones homogéneas de color.
- *Keywords* para recuperación semántica usando un modelo de vector del espacio.

1.4.2. VisualSeeK

VisualSeek (Chang, 1996) es un sistema basado en web propuesto por la "*Image and ATV Lab of Columbia University*" (ver anexo 2). En este sistema el usuario hace peticiones de imágenes por la descripción de arreglos espaciales de región de color. La similitud de imágenes en VisualSeek está dada por arreglos de regiones similares de color. Las características de este sistema son:

- Unión entre consultas basado en contenidos de imágenes
- Automatización directa del color
- Indexación hacia el color del objeto

Las consultas de este sistema se realizan de la siguiente manera:

1. El usuario hace un bosquejo del número de regiones, posiciones y dimensiones dentro de una tabla.
2. El usuario selecciona los límites para cada región.

3. El sistema regresa las mejores imágenes coincidentes con la consulta.
4. Finalmente el sistema muestra de forma decreciente las mejores imágenes similares según la consulta por el usuario.

La recuperación se basa en:

- Localización en el espacio.
- Regiones en la imagen basadas en color.

1.4.3. QBIC

Este sistema de consulta por contenido de imágenes fue desarrollado en el centro de investigación de la IBM en Almadeen. QBIC (ver anexo 1) es un sistema muy completo, ofrece la posibilidad de recuperar imágenes basadas en el contenido y basada en visualización por vídeos. Para imágenes estáticas las funciones disponibles son:

- Búsqueda por contenido semántico.
- Búsqueda por similitud de color global.
- Búsqueda por similitud de color regional y de textura.
- Búsqueda por forma.
- Solicitud por relaciones espaciales.

En el caso de video las funciones son:

- Segmentación automática.
- Extracciones de *frame* llave.

Este sistema utiliza como medida de distancia la denominada distancia euclidiana para evaluar la similitud de histogramas de color. Descriptores numéricos de coerción, contraste y dirección son extraídos de imágenes de escala de grises después de una conversión de imágenes de color. Las consultas por textura pueden ser combinadas con las consultas por color o por forma. Todas las formas están representadas de forma binaria. Un conjunto de 22 características es usado para su representación, entre ellas podemos mencionar:

- Área.
- Circularidad.
- Excentricidad.

En QBIC la calidad de la recuperación es altamente dependiente de la naturaleza de la

imagen, por lo general es efectivo. En el mismo consideran propiedades globales de forma (por ejemplo: redondez o que tan cuadrada es la forma de la botella) pero no es satisfactorio para evaluar la similitud de propiedades locales. Si las imágenes son simples, con un pequeño número de objetos y un fondo también sencillo, los objetos son automáticamente identificados al usar un algoritmo de segmentación no-supervisada. Si el algoritmo es más complejo, incluso con un fondo mas contrastado, se identifican regiones para señalarlas, usando una herramienta para que el usuario selecciones la región deseada. Ver anexo 3

La recuperación se basa en:

- Histograma de color
- Especificación de regiones basadas en color

Todos estos sistemas funcionan bien, sin embargo no permiten una búsqueda más específica. Muchas veces una consulta es confusa y con imágenes complejas, es por eso que en muchas ocasiones producen resultados indeseados.

1.5 Métodos existentes

A continuación se analizarán algunos métodos existentes para recuperar información visual. Algunos de estos métodos son usados en los sistemas existentes hasta el momento para este propósito.

1.5.1 Momentos

Los momentos son propiedades numéricas que se pueden obtener de una determinada imagen. Describen una región, en cuanto a la organización de sus píxeles, mediante descriptores de alto orden como el área, compactación e irregularidad. Los momentos son descriptores globales de una forma. En el análisis de imágenes son momentos estadísticos y fueron introducidos por Hu en 1962 (Hu, 1962). Los momentos están más asociados con el reconocimiento estadístico de patrones que con los modelos de visión. Esto se debe a que los modelos de visión asumen una vista no concluida de la forma. Las imágenes a procesar siempre tiene que estar umbralizadas, la forma óptima requiere de un simple objeto en el campo de visión. Son momentos estadísticos, en la teoría de reconocimiento de patrones sus características son derivadas de los momentos centrales.

Los *momentos Cartesianos en 2D* están asociados con un orden que inicia con valores pequeños (el mínimo es cero) hasta ordenes mayores. El momento de orden *p* y *q,* m_{pq}, de una función I(x, y) es definido como:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q I(x,y)dxdy$$

En el caso de las imágenes, la forma discreta es:

$$m_{pq} = \sum_x \sum_y x^p y^q I(x,y)\Delta A$$

En donde I(x, y) es una función que representa a la imagen en la cual si el píxel en las coordenadas (x, y) está dentro de la forma toma el valor de 1, en caso contrario toma el valor de 0. ΔA es el área de un píxel, de forma que si el área medida está en píxeles tomara el valor de 1. (Mery, 2006)

Estos descriptores poseen la propiedad de singularidad, ya que si la función satisface ciertas condiciones, los momentos de todos los órdenes existen. El momento de orden cero es equivalente a la función de "Área" descrita en la sección anterior. Mientras que los dos momentos de primer orden, m_{01} y m_{10}, son dados por:

$$m_{10} = \sum_x \sum_y xI(x,y)\Delta A \qquad m_{01} = \sum_x \sum_y yI(x,y)\Delta A$$

Para imágenes binarias estos valores representan las coordenadas del centro de la imagen, por lo que el *centro de masa* (x, y) puede ser calculado como el radio del primer orden al componente de orden cero:

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}}$$

Debido a que el mayor interés al emplear estos momentos está conforme a la invariancia en cuanto a la posición, tamaño y rotación; se pueden definir nuevos momentos por medio de la estimación al centro de la forma y que poseen invariancia a la traslación, denominados *momentos centralizados* μ_{pq}, los cuales pueden ser definidos como:

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x,y)\Delta A$$

Para obtener una invariancia al escalado se requieren los *momentos centralizados*

normalizados, η_{pq}, que se definen como:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \qquad \gamma = \frac{p+q}{2} + 1 \quad \forall p+q \geq 2$$

Los 7 momentos invariantes a la rotación de HU pueden ser calculados como: (Suk, 2006)

$$M1 = \eta_{20} + \eta_{02}$$
$$M2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$M3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$M4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$M5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) + ((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} - \eta_{03})^2) + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2)$$
$$M6 = (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$M7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{12} + \eta_{30})^2 - (\eta_{21} + \eta_{03})^2)$$

Los dos primeros de ellos, M1 y M2, son momentos de segundo orden (p + q=2); los restantes son momentos de tercer orden, p + q=3. Los momentos de primer orden no son considerados debido a que tiene valor de cero. El último momento, M7, fue introducido por Hu para la invariancia a imágenes reflejadas.

1.5.2 Wavelets

Los Wavelets son funciones matemáticas que cortan datos a diferentes componentes de frecuencia y entonces se estudia cada componente con una solución emparejada con su escala. Los Wavelets tienen ventajas sobre los métodos tradicionales de Fourier en cuanto a analizar situaciones físicas donde la señal contiene discontinuidades y puntos agudos. Los Wavelets fueron desarrollados independientemente en los campos de matemáticas, física cuántica, ingeniería electrónica y geología sísmica. El intercambio entre estos campos durante los últimos diez años ha conducido a nuevas aplicaciones usando Wavelets como la compresión de imágenes, turbulencia, visión humana, radares y predicción de terremotos.

El análisis de Fourier es un campo que se encuentra dentro del centro de las matemáticas aplicadas. Estas técnicas son una base para los wavelets y además los aspectos computacionales tanto de las trasformaciones de Fourier como de las series

de Fourier son bastante atractivos, ya que cuentan con características ortogonales y brindan una simple representación basada en dos funciones elementales: seno de *x* y coseno de *x*. La gran desventaja de las transformadas de Fourier es que únicamente tiene resolución de frecuencia y no tienen resolución de tiempo. Esto quiere decir que aunque podamos determinar todas las frecuencias presente en una señal, no sabemos cuando están presentes.

El análisis de wavelets (Jensen, 2001) permite a los investigadores aislar y manipular tipos de patrones específicos, ocultos en cantidades ingentes de datos de forma muy parecidas a como nuestros ojos observan los árboles o nuestros oídos pueden elegir el sonido de una flauta en una sinfonía.

La transformada de Wavelets a partir de una representación espacial en puntos de la imagen obtiene una serie de componentes que se van refinando para crear una representación multiescala de la imagen. A partir de los coeficientes en que se ha descompuesto la imagen es posible calcular una representación numérica de características que se manifiesten en la imagen de manera que sea posible aplicar un esquema de ordenación en base a un criterio de similitud con la consulta inicial.

Ejemplos de wavelets son: *Coiflet, Morlet, Mexican Hat, Daubechies y Haar*. De éstos, *Haar* es el más simple y más utilizado, mientras que *Daubechies* tiene estructuras fractales y son de vital importancia para las actuales aplicaciones de wavelets. Estos dos tipos de wavelets se muestran a continuación (Siddique, 2002):

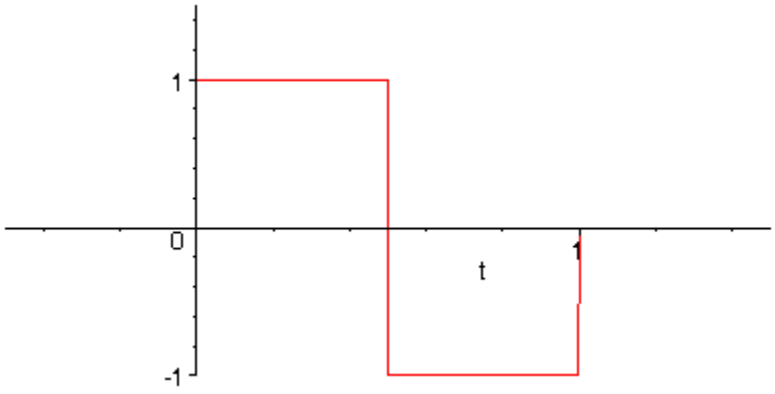


Fig. 3 Ejemplo de *Haar* wavelet

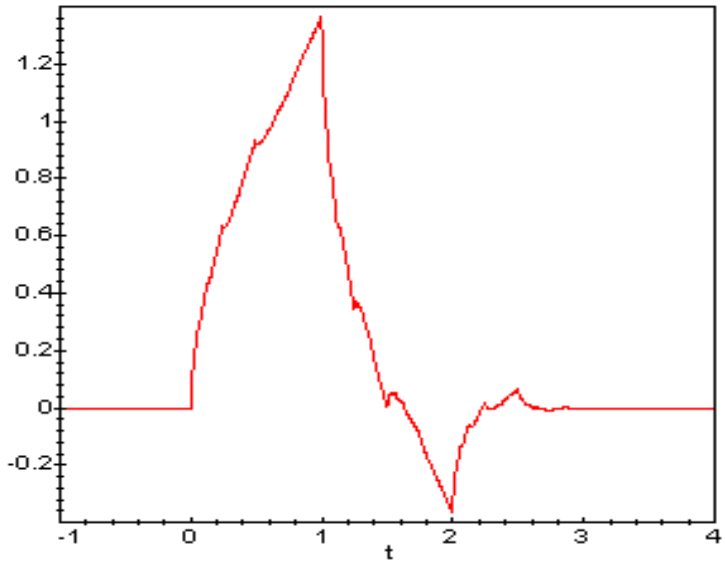


Fig. 4 Ejemplo de *Daubechies* wavelet

Las wavelets han demostrado ser una efectiva herramienta para análisis de imágenes debido al hecho de ser capaces de capturar la información espacial junto a las características visuales. Los problemas derivados de la necesidad de invariancia frente a transformaciones geométricas también han sido tratados y resueltos satisfactoriamente.

1.5.2.1 Propiedades de los Wavelets

Dentro las propiedades más importantes de los Wavelets se pueden mencionar las condiciones de admisibilidad y regularidad, dichas propiedades son las que le dan a los Wavelets su nombre.

Una propiedad fascinante de las wavelets es que eligen automáticamente las mismas características que nuestros ojos. Los coeficientes de las wavelets que quedan aun tras la cuantización corresponden a píxeles que son muy distintos a sus vecinos en el borde de los objetos de la imagen. Por tanto las wavelets recrean la imagen principalmente trazando bordes, que es exactamente los que hacen los humanos cuando esbozan un dibujo. De hecho, algunos investigadores han sugerido que la analogía entre la visión humana y los wavelets no es accidental, y que nuestras neuronas filtran las señales visuales de forma parecida a los wavelets.

La condición de admisibilidad se puede usar para primero analizar y después construir una señal sin pérdida de información. Además de estas propiedades, los Wavelets tienen condiciones de regularidad y estas dicen que las funciones de Wavelets deberían de tener concentración y suavidad tanto en dominios de frecuencia como de tiempo.

1.5.3 FFT

Fast Fourier (FFT), (Winograd, 1978) es un algoritmo eficiente para calcular la trasformación discreta de *Fourier* (DFT por sus siglas en ingles) y su inversa. Tiene una gran importancia puesto que se aplica en una amplia variedad de aplicaciones para el procesamiento digital de señales, para resolver diferenciales parciales, o en algoritmos para multiplicar números enteros grandes rápidamente. La transformación rápida de *Fourier* no es una nueva transformación sino un algoritmo que permite reducir el tiempo de cálculo de la trasformación discreta o DFT.

La necesidad de precisión y el aumento de la banda de análisis conllevan al aumento de frecuencias de muestreo y por tanto del número de muestras, multiplicaciones y sumas a realizar, lo cual retrasa la presentación en tiempo real de la señal. Sin embargo muchos de los coeficientes son redundantes y el cálculo no puede ser exacto.

El método más famoso de FFT es el popularizado por *J.W.Cooley y JW.Turkey* en 1965 el cual lleva el nombre de sus autores. El algoritmo FFT se basa en una técnica *"Divide y Vencerás"* que recursivamente rompe un DFT de cualquier tamaño compuesto de n = n1n2 en más pequeños DFT's de tamaños n1 y n2 junto con O(n) multiplicaciones por las raíces complejas de la unidad tradicional las que se denominan "*twiddle factors".*

1.5.4 CSS

Representación de formas, es uno de los aspectos más difíciles en la visión computacional. Este tema ha sido difícil puesto que la forma es más compleja que el color y la textura. El color y la textura pueden ser representados por unos pocos parámetros pero en el caso de la forma se necesitan cientos de parámetros para que sea representada explícitamente. Sin embargo, los resultados obtenidos al hacer uso de técnicas de representación de formas son satisfactorios.

CSS por sus siglas en ingles (*Curvate Scale Space*) (Sadegh Abbasi, 1999) es un método de comparación de imágenes que utiliza la forma como principal descriptor. Básicamente el procedimiento es procesar cada límite de una imagen CSS y entonces encontrar el punto máximo de los contornos CSS que son usados como descriptores de formas para comparar objetos. Una imagen CSS es una organización multiescala de los puntos de inflexión del contorno donde se desarrolla. Las coordenadas de esos puntos junto con el *aspect ratio* de la imagen CSS (número de columnas y números de renglones), la excentricidad, la circularidad y el nombre de la imagen original constituyen características que representan al objeto.

Para recuperar imágenes similares de una base de datos el usuario puede poner una imagen y pedirle al sistema que recupere las imágenes similares. El sistema procesa la imagen CSS de entrada, encuentra su máximo y después la compara. Las características recuperadas de una imagen CSS, al compararla, son los puntos máximos de sus contornos de cero. Comparar dos imágenes CSS consiste en encontrar el cambio horizontal óptimo de los máximos en una de las imágenes CSS que rendirán la mejor superposición posible con los máximos de las otras imágenes CSS. El costo de comparación entonces se define como la suma en parejas de distancias (en CSS) entre los pares correspondientes de máximos.

Una vez comparadas las imágenes el sistema asigna un valor de comparación a cada imagen candidata en la base de datos que es similar a la de la entrada que insertó el usuario. La imagen candidata es aquella en la que su *aspect ratio*, excentricidad y circularidad caen en el intervalo de la imagen de entrada.

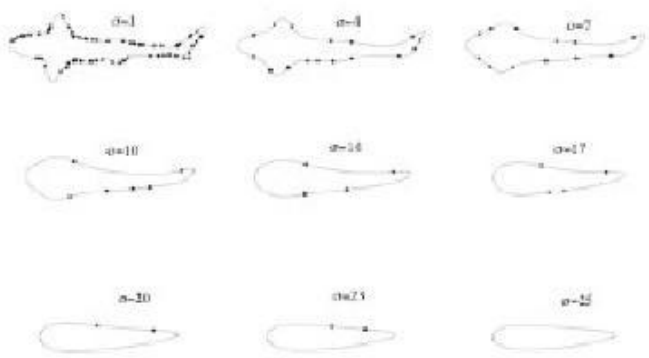


Fig. 5 Evolución de curvas con σ =25

Este método brinda las siguientes ventajas (Baojiang Zhong, 2007):

1. Es robusto con respecto al ruido, escala y cambio de orientación.
2. Retiene la información local de la forma de la imagen de entrada. Cada convexidad de la forma de la imagen tiene su propio contorno correspondiente de la imagen CSS.
3. Es rápido en respuesta a una consulta. El sistema procesa la imagen CSS de la entrada y extrae sus máximos. Entonces, el sistema compara el vector característico extraído con todas aquellas imágenes candidatas en la base de datos.
4. Es confiable .Varios sistemas, utilizan este método y los resultados obtenidos son satisfactorios.

1.6 Lenguajes de programación:

Uno de los objetivos de nuestro trabajo de diploma es implementar los algoritmos propuestos en el lenguaje *Python* para su posterior muestreo y prueba en el producto Pyxel, por lo que no es necesario hacer un exhaustivo análisis de lenguajes de programación existentes para escoger uno candidato.

1.6.1 Lenguaje *Python*

Python (htt2)es un lenguaje de programación sencillo, pero a la vez potente y muy similar a otros lenguajes existentes como Perl o Java. Fue creado por Guido Van Rossum y lanzado por primera vez en 1990. Actualmente es administrado por la *Python*

Software Foundation.

Es un lenguaje de código abierto, orientado a objetos y basado en scripts (interpretado). Esto último quiere decir que el código es ejecutado en un intérprete en lugar de ser compilado. Además cuenta con un modo interactivo que permite ejecutar código en tiempo real; según se ingresan las sentencias en el intérprete se ejecutan y se visualizan los resultados. Al ser un lenguaje interpretado ahorra una gran cantidad de tiempo en el desarrollo del programa ya que no necesita compilar ni enlazar el código.

Otra característica de *Python* es que permite dividir el código en diferentes módulos que se pueden reutilizar en otros programas implementados en *Python* simplemente importándolos. Viene además con diferentes módulos estándar que proporcionan multitud de aplicaciones para utilizar de base para la creación de programas. Además existen una gran variedad de módulos adicionales que pueden ser descargados y utilizados fácilmente. Algunos de estos módulos han sido utilizados para la realización de la presente aplicación:

- **Pywavelet:** *Python Wavelet*, facilita todo el trabajo con las diferentes transformadas wavelets.
- **Numpy**: *Numerical Python*, proporciona herramientas para la creación y manipulación de arrays.
- **PIL**: *Python Imaging Library*, encargada del procesamiento de las imágenes.
- **Pyro**: *Python Remote Object,* facilita e implementa el trabajo de algoritmos en paralelo.
- **Python OpenCV**: Facilita el trabajo con imágenes y la extracción de características importantes para su descripción.

Por último, una de las más importantes características de *Python* es que se trata de un lenguaje de muy alto nivel y que por lo tanto incluye tipos de datos muy complejos que permiten expresar operaciones bastante complejas con sentencias relativamente simples. Esto, unido a que no precisa de declarar variables ni argumentos y a que utiliza la identación, hacen de *Python* un lenguaje muy sencillo, compacto y legible.

1.7 Programación paralela

La programación paralela es una técnica de programación en la que muchas instrucciones se ejecutan simultáneamente. Se basa en el principio de que los problemas grandes se pueden dividir en partes más pequeñas que pueden resolverse de forma concurrente ("en paralelo"). Existen varios tipos de computación paralela: (Blaise Barney)

- **Paralelismo a nivel de instrucción:** es cuando las instrucciones de una secuencia son independientes y por tanto pueden ejecutarse simultáneamente en distintas unidades de procesamiento, cada procesador realiza su conjunto de instrucciones por separado colaborando para dar el resultado final.
- **Paralelismo de datos:** consiste en subdividir el conjunto de datos de entrada a un programa, de manera que a cada procesador le corresponda un subconjunto de esos datos. Cada procesador efectuará la misma secuencia de operaciones que los otros procesadores sobre su subconjunto de datos asignado. En resumen: se distribuyen los datos y se replican las tareas. Idealmente esta ejecución simultánea de operaciones resulta una aceleración neta global del cómputo. El paralelismo de datos es un paradigma suficientemente adecuado para operaciones sobre vectores y matrices debido a que muchas de ellas consisten en aplicar la misma operación sobre cada uno de sus elementos.
- **Paralelismo de tareas:** consiste en asignar distintas tareas a cada uno de los procesadores de un sistema de cómputo. En consecuencia cada procesador efectuará su propia secuencia de operaciones. En su modo más general, el paralelismo de tareas se representa mediante un grafo de tareas que se subdivide en subgrafos que son luego asignados a diferentes procesadores. De la forma como se corte el grafo depende la eficiencia del paralelismo resultante.

1.7.1 Clúster

El término clúster se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora. Hoy en día juegan un papel importante en la solución de problemas de las ciencias, las ingenierías y del comercio moderno.

Los clústeres han evolucionado en apoyo de actividades que van desde aplicaciones de súper-cómputo, software de misiones críticas, servidores Web y comercio electrónico hasta bases de datos de alto rendimiento, entre otros usos.

El cómputo con clúster surge como resultado de la convergencia de varias tendencias actuales que incluyen la disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta velocidad, el desarrollo de herramientas de software para cómputo distribuido de alto rendimiento, así como la creciente necesidad de potencia computacional para aplicaciones que la requieran.

Simplemente, un clúster es un grupo de múltiples ordenadores unidos mediante una red de alta velocidad, de tal forma que el conjunto es visto como un único ordenador, más potente que los comunes de escritorio. Los clústeres son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.

De un clúster se espera que presente combinaciones de los siguientes servicios:

1. Alto rendimiento
2. Alta disponibilidad
3. Equilibrio de carga
4. Escalabilidad

La construcción y agrupación de los ordenadores del clúster es muy fácil y económica debido a su flexibilidad:

Pueden ser **homogéneos** cuando tienen todas las estaciones de trabajo la misma configuración de hardware y sistema operativo, **semi-homogéneos** cuando tienen diferente rendimiento pero con arquitecturas y sistemas operativos similares o **heterogéneos** cuando tienen diferente hardware y sistema operativo, lo que hace más

fácil y económica su construcción.

Para que un clúster funcione como tal no basta solo con conectar entre sí los ordenadores, sino que es necesario proveer un sistema de manejo del clúster que se encargue de interactuar con el usuario y los procesos para optimizar el funcionamiento.

Cabe aclarar que a la hora de diseñar un clúster los nodos deben tener características similares, es decir, deben guardar cierta similitud de arquitectura y sistemas operativos, ya que si se conforma un clúster con nodos totalmente heterogéneos (existe una diferencia grande entre capacidad de procesadores, memoria, capacidad en disco) será ineficiente debido a que el middleware delegará o asignará todos los procesos al nodo de mayor capacidad de cómputo y solo distribuirá cuando este se encuentre saturado de procesos; por eso es recomendable construirlo con un grupo de ordenadores los más similares posible.

El middleware es un software que generalmente actúa entre el sistema operativo y las aplicaciones con la finalidad de proveer a un clúster lo siguiente:

- Una interfaz única de acceso al sistema, denominada SSI (*Single System Image*), la cual genera la sensación al usuario de que utiliza un único ordenador muy potente;
- Herramientas para la optimización y mantenimiento del sistema, migración de procesos, *checkpoint-restart* (congelar uno o varios procesos, mudarlos de servidor y continuar su funcionamiento en el nuevo host), balanceo de carga, tolerancia a fallos, entre otros.
- Escalabilidad ya que debe poder detectar automáticamente nuevos servidores conectados al clúster para proceder a su utilización.

1.7.2 Software para la Computación en Paralelo

La paralelización de algoritmos es una tendencia actual que tiene como objetivo acelerar el tiempo de procesamiento de un programa a fin de hacer concurrentes sentencias del mismo, teniendo gran aceptación y resultados en la actualidad donde la carga computacional es grande en muchos ámbitos de la informática. Esta es la forma en que se trabaja en el desarrollo de los procesadores modernos porque resulta más difícil incrementar la capacidad de procesamiento con un único procesador que

aumentar su capacidad de cómputo mediante la inclusión de unidades en paralelo obteniéndose así la ejecución de varios flujos de instrucciones dentro del procesador.

1.7.2.1 PVM (*Parallel Virtual Machine*)

PVM es una librería que procura englobar cualquier red de ordenadores bajo el concepto de "Máquina Virtual Paralela". Inicialmente se utilizó para ejecutar aplicaciones paralelas en plataformas de bajo coste, su gran aceptación la ha convertido en un estándar de *facto,* siendo ofrecido actualmente por casi todos los fabricantes de supercomputadoras.

El concepto de Máquina Virtual Paralela se aplica a un conjunto de ordenadores interconectados los cuales gracias a PVM podrán verse como una única maquina. Dentro de esta máquina virtual los distintos procesos se comunicaran mediante las primitivas de paso de mensaje ofrecidas por PVM y se podrá gestionar la creación o destrucción de procesos.

PVM puede utilizarse tanto en máquinas masivamente paralelas como en multiprocesadores en general o sencillamente sobre un grupo de máquinas heterogéneas interconectadas por algún tipo de red. En este último caso será PVM el encargado de traducir las posibles diferencias en los formatos de los datos comunicados entre las distintas máquinas.

PVM esta implementado en C, aunque puede utilizarse desde los lenguajes C++ y FORTRAN mediante el uso de *stubs*(Rutinas cuyo código solo se encarga de llamar a otras rutinas haciendo una posible traducción entre sus parámetros a las rutinas de C).

Una característica de PVM es que no solo se ocupa de la parte de comunicaciones de los programas en paralelo sino que permite gestionar de manera dinámica tanto la configuración de la Máquina Virtual como la creación y destrucción de los procesos

1.7.2.2 MPI (*Message Passing Interface*)

MPI es un protocolo de comunicación entre computadoras. Más específicamente para la comunicación entre los nodos que ejecutan un programa en un sistema de **memoria distribuida**. Las implementaciones en MPI consisten en un conjunto de bibliotecas de rutinas que pueden ser utilizadas en programas escritos en *C*, C++, Fortran entre otros. Es un estándar que define operaciones de paso de mensaje a utilizar en cualquier plataforma que pueda ser programada con este paradigma. Existen diversas implementaciones, las más utilizadas son MPICH y LAM.

MPI permite el trabajo con tipos de datos básicos y nuevos tipos que pueden definirse con algunas de sus operaciones. Tiene disimiles, es práctico, portable, eficiente y flexible.

1.7.2.3 Pyro (*Python Remote Objects*)

Pyro es un avanzado y potente sistema distribuidor de objetos totalmente escrito en *Python*. Es pequeño, sencillo y extremadamente portátil. Está diseñado para ser muy fácil de usar. Es muy sencillo de implementar ya que solo es necesario agregarle clases normales de *Python* porque todo el código de comunicación de redes es abstraído y oculto de su aplicación. Pyro ofrece un servidor de nombres (NS por sus siglas en inglés) que mantiene un registro de la ubicación de los objetos. La ubicación del NS puede ser descubierta por un mecanismo de difusión, y varias otras maneras, si su red no es compatible con la radiodifusión. Realiza un trabajo eficiente con objetos remotos a través de sus característicos proxys, excepciones remotas, acceso remoto a atributos, re-conexión automática en caso de fallos de conexión de red, y una muy buena documentación detallada, establece sus propios protocolos de comunicación dentro de la red LAN que lo hace muy eficiente y seguro.

1.8 Sistemas Gestores de Bases de Datos (SGBD)

Un SGBD (en inglés: *DataBase Management System*) puede definirse como un paquete generalizado de software que se ejecuta en un sistema computacional anfitrión centralizando los accesos a los datos y actuando de interfaz entre los datos físicos y el usuario. Las principales funciones que debe cumplir un SGBD se relacionan con la

creación y mantenimiento de la base de datos, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad. Los SGBD permiten al programador convencional ahorrarse horas de trabajo dedicadas a la seguridad, gestión de los datos, chequeo de errores, entre otros.

Entre los SGBD comúnmente utilizados en el mundo tenemos MySQL, Oracle, PostgreSQL, Microsoft SQL Server, SQLite, entre otros. Todos estos presentan un enfoque relacional con un buen basamento matemático centrado en el Álgebra Relacional.

1.8.1 SQlite

SQLite es un sistema de gestión de base de datos relacional que está contenida en una relativamente pequeña biblioteca en C de aproximadamente 500 KB. SQLite es un proyecto de dominio público creado por D Richard Hipp.

A diferencia de los sistemas de gestión de base de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones lo que reduce la latencia en el acceso de la base de datos debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos) son guardados como un solo fichero estándar en la maquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

1.8.2 Análisis de la selección del SGBD.

Luego de haber realizado un estudio detallado de los principales SGBD y valorando el entorno en que se va a desempeñar la aplicación, así como la complejidad que presenta el modelo relacional de la misma, se optó para la implementación del sistema utilizar SQLite por las siguientes razones:

- **Tamaño**: SQLite tiene una pequeña memoria y una única biblioteca es necesaria

para acceder a las bases de datos lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

- **Rendimiento de base de datos**: SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- **Portabilidad**: se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.
- **Estabilidad**: SQLite es compatible con ACID, reunión de los cuatro criterios Atomicidad, Consistencia, Aislamiento y Durabilidad.

1.9 Metodologías de desarrollo. Análisis de su selección

El objetivo de un proceso de desarrollo de software es elevar la calidad de este a través de la transparencia y control sobre su desarrollo, logrando así que dichas pautas sean reproducibles en cada proyecto independientemente de su magnitud. Actualmente existe una gran cantidad de procesos de desarrollo agrupados en dos tendencias: los métodos ágiles y los métodos pesados.

Debido a la envergadura del trabajo no sería conveniente usar RUP (Kruchten, 2003) puesto que no está regido por ningún caso de uso ni va dirigido a un usuario final como tal sino que es más bien un desarrollo de componentes para uso interno del proyecto Pyxel. No es de vital importancia el control de los procesos mediante rigurosas definiciones de roles, actividades o artefactos. Puesto que no es un gran proyecto en cuanto a recursos y tiempo aunque se exija alta calidad, es que surge la necesidad de usar una metodología acorde con su magnitud sin renunciar a las prácticas esenciales para asegurar la calidad del proyecto.

Dadas las características del proyecto “Informatización de la prensa” (Ailyn Alfonso González, 2008), se decidió usar como metodología ágil una variante personalizada de SCRUM (ver anexo 5). Donde se seleccionaron las características más adecuadas para aplicar al desarrollo de este trabajo que cuenta con un grupo reducido de desarrolladores.

Esta metodología está diseñada para adaptarse a posibles cambios que puedan ocurrir en los requerimientos del sistema a lo largo de su desarrollo. Los requerimientos y las

prioridades se revisan y ajustan durante el desarrollo del software en intervalos muy cortos y regulares, adaptando así en tiempo real el producto que se está construyendo a las necesidades del cliente. Scrum se basa en los principios de inspección continua, adaptación, autogestión e innovación en donde se obtienen entregables que hacen visibles los avances alcanzados, logrando así estimular y comprometer al cliente con el proyecto, ya que nota el crecimiento del producto.

Capitulo 2: Algoritmos propuestos para la detección de imágenes duplicadas

Para guiar el desarrollo de los algoritmos de detección de duplicados hacia el producto Pyxel es necesario comprender su contexto e identificar las condiciones y capacidades que debe cumplir. En este capítulo se definen los algoritmos para la detección de duplicados a implementar describiendo como va a funcionar y destacando sus características distintivas. Se realiza además una evaluación de los algoritmos mediante la obtención y análisis de resultados experimentales.

2.1 Algoritmos propuestos

A continuación se proponen algunos métodos de detección de imágenes duplicadas para la identificación de copias modificadas de la misma imagen en un gran repositorio. A estos algoritmos se les han hecho modificaciones para que tomen en cuenta aspectos tales como la rotación, el escalado y la traslación.

Debido a que en muchas ocasiones los duplicados no son réplicas idénticas de las imágenes en el repositorio, pero si son versiones de procesado digital de la imagen original, los métodos estándares de hash no funcionan. Aquí, "duplicado" se refieren a versiones modificadas digitalmente de la imagen después de manipulaciones realizadas usando cualquier herramienta de procesamiento de imágenes.

El tiempo real de recuperación en un gran archivo de imágenes exige necesariamente sistemas robustos en términos de:

- La eficiencia: puntualidad.
- La Exactitud: precisión y *recall*.
- La Escalabilidad: la propiedad del alojamiento de los cambios significativos en el volumen de datos sin afectar el rendimiento del sistema.

Por un lado se quiere alcanzar alta precisión lo cual a menudo significa alto costo computacional. Por otro lado se busca reducir el tiempo de procesamiento a fin de que más imágenes se puedan comparar en un tiempo limitado. Para cumplir estos requisitos se necesita un buen vector característico que pueda diferenciar imágenes y pueda encontrar las imágenes sospechosas con precisión. En este documento se presentan dos servicios de detección de copias desarrollados en la Universidad de las

Ciencias Informáticas. Estos servicios serán llamados “WaveDuplicates” y “HUDuplicates” tomando en cuenta los métodos de descripción utilizados para su desarrollo, propósito wavelets y momentos Hu respectivamente.

2.2 Arquitectura

Existen varias maneras para realizar la recuperación de imágenes. Una de ellas es basándose en descripciones de texto, que consiste en indexar las imágenes partiendo de anotaciones textuales y ejecutando consultas basadas en texto; otra es basándose en consultas ejemplos de imágenes, recuperando las imágenes que mejor concuerdan con la imagen a consultar. En este tipo de recuperación el usuario alimenta al sistema con una imagen que representa lo que quiere recuperar de la base de datos.

Los algoritmos propuestos en este trabajo van a recibir como entrada imágenes brindadas por los usuarios y a partir de esto encontrarán sus duplicados en la base de datos. Si el duplicado es encontrado, retornara una lista de los ID de las Imágenes sospechosas como duplicadas. Estos algoritmos no almacenan las imágenes en la base de datos, sino que después de procesadas las imágenes se almacenan los vectores característicos y sus ID en una simple Base de Datos desarrollada con SQLite.

El sistema sobre el que se está trabajando tiene la estructura que se muestra en las figuras.6 y 7. Para realizar su estructura se han identificado los bloques que lo constituyen.

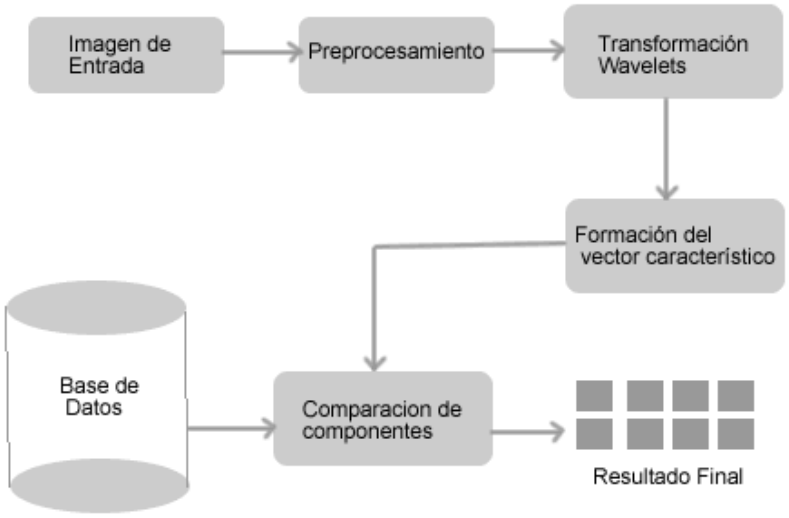


Fig. 6 Estructura Básica del algoritmo WaveDuplicates

Muy similar a la estructura del algoritmo WaveDuplicates es la estructura básica del HuDuplicates como se puede observar en siguiente figura.

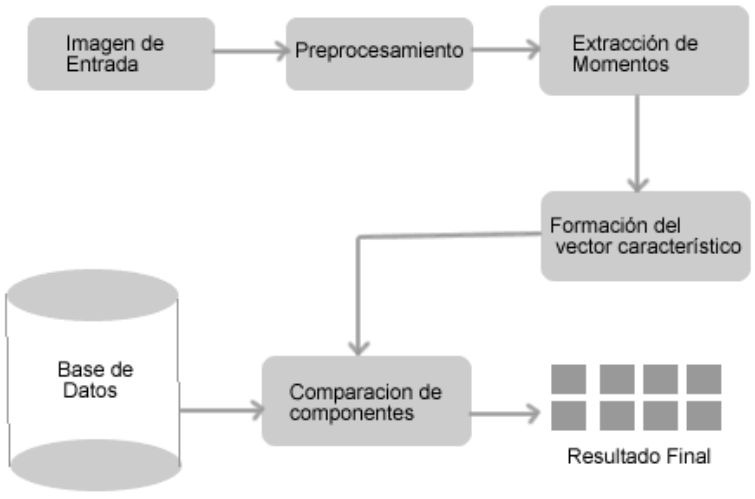


Fig. 7 Estructura Básica del algoritmo HuDuplicates

2.3 Vector característico

Un vector característico debe cumplir la condición de ser pequeño y un buen representador de la imagen. Debe ser capaz de diferenciar las imágenes en una base de datos y por consiguiente poder relacionar su semejanza. Se hizo uso de *Daubechies' Discrete Wavelet Transform (DWT)* (Edward Y. Chang, 1998) *y los momentos HU* (Mery, 2006) para captar la representación visual de la imagen.

Se presentan tres pasos para preparar el vector característico de la imagen. Primero modificamos la imagen a un tamaño predeterminado. Luego aplicaremos los métodos antes mencionados y por último almacenamos los resultados como el vector característico.

2.3.1 Pre-procesamiento de la imagen

Muchos formatos de imagen se encuentran actualmente en uso por ejemplo: GIF, JPEG, TIFF y PPM son los formatos más utilizados. Existe una larga lista de formatos de imágenes, lo que constituye un problema pues estos formatos se leen de formas diferentes y se debe tener una biblioteca que admita al menos la lectura de cada uno de ellos. Hasta el momento trabajamos con los formatos JPG, PGN, TIFF, entre otros. Estos formatos se encuentran entre los más usados en Internet, por lo que no se precisa el uso, por el momento, de ninguna biblioteca para leer otro formato de la imagen. Producto de que en una base de datos de imágenes puede existir diferentes tamaños de imágenes, primero redimensionamos la imagen a un tamaño adecuado al método que se va a utilizar para hallarle sus duplicados en la base de datos.

2.3.1.1 Pre-procesamiento de la imagen para HUDuplicates

A las imágenes que serán procesadas con este algoritmo se le realizará primeramente una conversión a escala de grises y posteriormente serán rescaldadas a un tamaño de 800 X 800. Además de estas transformaciones se hace necesaria la reducción de ruidos en la imagen para obtener mejores resultados.

2.3.1.2 Pre-procesamiento de la imagen para WaveDuplicates

En el pre procesamiento de la imagen para WaveDuplicates se reescala la imagen en un tamaño de 256 X 256, en el espacio de color RGB (rojo, verde, azul). Se usa 256

x256 porque este es un tamaño estándar en las imágenes en Internet y también por el hecho de que las transformaciones wavelet necesitan que los lados de las imágenes sean potencia de 2. Luego se trasforma la imagen reescalada a otro espacio de color puesto que lo que se necesita es la percepción real del color de distancia, la cual difiere de la del espacio de color RGB. (Edward Y. Chang, 1998)

$$C1 = (R + G + B)/3$$

$$C2 = (R + (255 – B))/2$$

$$C3 = (R + 2 * (255 - G) + B)/4$$

Se definen los nuevos colores de los píxeles basados en los valores RGB del píxel original. Se definen tres nuevos componentes de color para proveer una propiedad de correlación perceptible.

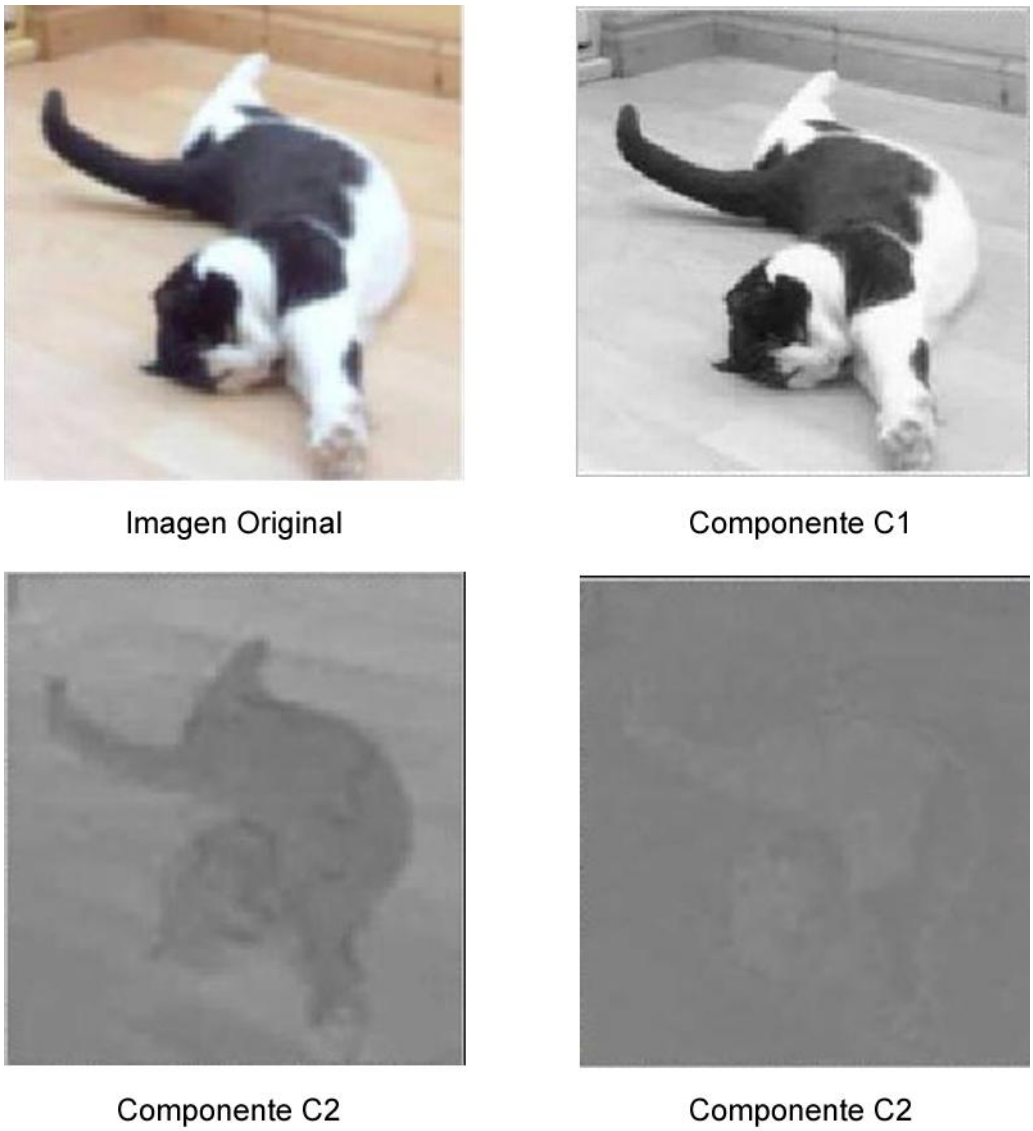


Fig. 8 Componentes de color extraídos de la imagen original

2.3.1.3 Modelos de color RGB

La descripción **RGB** (del inglés *Red, Green, Blue*; "rojo, verde, azul") de un color hace referencia a la composición del color en términos de la intensidad de los colores primarios con que se forma: el rojo, el verde y el azul. Es un modelo de color basado en la síntesis aditiva con el que es posible representar un color mediante la mezcla por adición de los tres colores da luz primarios. Se debe tener en cuenta que el modelo de color RGB no se refiere literalmente a lo que es rojo, verde o azul, por esta razón los mismos valores RGB pueden mostrar colores notablemente diferentes en diferentes dispositivos que usen este modelo de color. Aunque utilicen un mismo modelo de color,

sus espacios de color pueden variar considerablemente.Es el más común en sistemas de proceso digital de imágenes (escáner, cámaras, monitores, entre otros)

Inconvenientes del modelo de color RGB:

- Alta correlación entre sus componentes

- No intuitivo psicológicamente

- No uniforme: la distancia en el espacio RGB no se corresponde con la distancia perceptiva.

2.3.2 Extracción de características de la imagen

Tradicionalmente se ha usado un histograma de color (un histograma por cada componente: rojo, verde y azul) para caracterizar una imagen. El problema en esto es que un histograma global preserva la información de color contenida en una imagen pero no la información de la localización de ese color. Debido a que este método no puede decir donde los colores están concentrados, dos imágenes teniendo similares histogramas de color pueden tener muy diferentes semánticas.

La representación basada en textura y la representación basada en formas son utilizadas como técnicas de caracterización de imagen, técnicas que tienen una sustancial limitación cuando del propósito general de una base de datos de imágenes se trata. El algoritmo de representación basado en formas solo trabaja efectivamente con imágenes con fondos uniformes y la de textura no es apropiada para imágenes que no representen texturas. Por otro lado la representación basada en formas requiere un intenso cálculo por tanto no es recomendable para procesar un largo número de imágenes.

Para presentar la semánticas en imágenes, incluyendo configuración de objetos y variación local de color, es usado *Daubechies' wavelet coefficients* (Edward Y. Chang, 1998). A diferencia de las técnicas mencionadas, wavelets ofrece una buena localización espacial y de frecuencia. En el dominio espacial es capaz de capturar los cambios de color de un píxel a otro y en el dominio de frecuencia puede capturar la intensidad de los cambios de color en mínimo detalle.

Se aplica una rápida trasformación wavelets a la pre-procesada imagen para obtener

su semántica. Los coeficientes de baja frecuencia son almacenados como el vector característico de la imagen.

Los momentos se emplean como consultor de objetos en la imagen ya que son invariantes a la traslación, rotación y escalado, permitiéndole ser un gran representador del contenido visual de una imagen.

2.3.2.1 Seleccionando características usando Wavelets y Momentos

La extracción de características juega un papel importante en los algoritmos propuestos para la detección de imágenes duplicadas pues nos permite contar con un descriptor de la imagen para su comparación. A cada imagen se le asocia un vector de características que incluye información de la imagen. Se aplica *Daubechies' Discrete Wavelet Transform (DWT)* a sus componentes y se extrae el vector característico basado en coeficientes. Se aplicarán trasformaciones wavelets recursivamente para reducir los coeficientes a 16 x16 en los tres espacios de color respectivamente con el fin de almacenar los coeficientes más importantes.

La siguiente figura muestra un ejemplo donde la trasformación wavelets es aplicada en uno de los espacio de color tres veces para reducir el número de coeficientes a uno octavo. (La esquina izquierda de la figura)

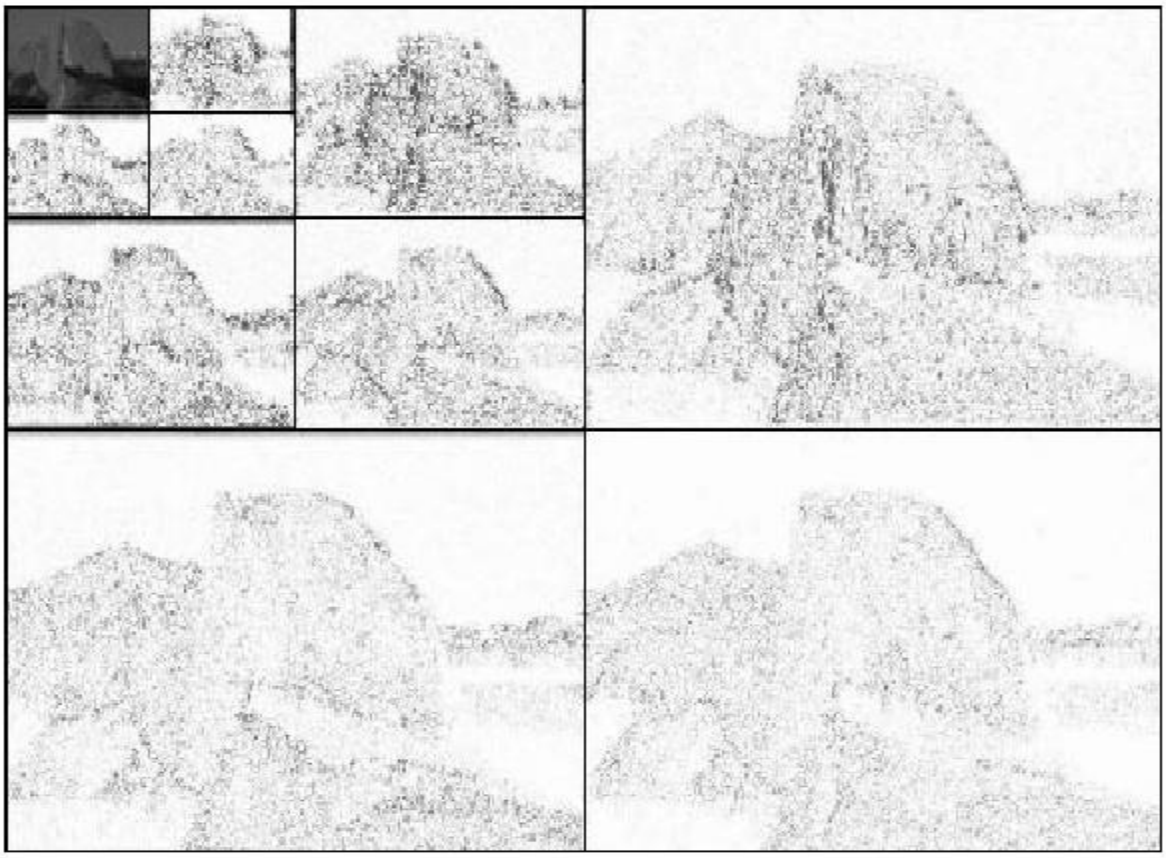


Fig. 9 Ejemplo de transformación wavelets

El uso de los Momentos de HU representa un buen descriptor de imágenes con el cual se le pueden calcular los píxeles de una región de la imagen. Los momentos de HU son invariantes a la traslación, rotación y escalamiento. Esto quiere decir que dos regiones que tengan la misma forma pero que sean de distinto tamaño y que estén ubicadas en posiciones y orientaciones distintas en la imagen tendrán momentos de HU iguales y por consiguiente pueden declarar a dos imágenes similares. El vector de características debe extraer la información considerada relevante para quedar incluida en la base de datos, por ello lo componen los momentos HU de la imagen.

2.4 Distancia euclidiana

Cuando el usuario envía una consulta se calcula el vector de características y se hace coincidir con los vectores de características pre-calculados de las imágenes de la base de datos. La distancia euclidiana es la usada para medir la similitud en la fase de comparación.

Algoritmo de Distancia Euclidiana:

1. Por cada imagen i en la base de datos se obtienen su vector característico
2. Se calcula la distancia euclidiana entre los dos vectores de características usando: $D_i = \sum_{k=1}^{k} (x_k - y_{ik})^2$
3. Se incrementa i y se repite el paso 1.

Usando este algoritmo, una vez realizada la consulta se procede a buscar en la base de datos de imágenes. La distancia euclidiana se calcula entre el vector de la imagen dada y los vectores de todas las imágenes almacenados en la base de datos. Este proceso se repite hasta que todas las imágenes en la base de datos se han comparado con la imagen de la consulta. Si una imagen en la base de datos difiere de la consulta de imágenes demasiado cuando se comparan los vectores de características, se descarta.

2.5 Algoritmo paralelo

Para el desarrollo de este trabajo se utiliza Pyro(abreviatura para *Python* y objetos a distancia). El objetivo de esta sección es aplicar los conceptos planteados en el capítulo anterior, profundizando en el modelo de arquitectura y estableciendo las estrategias de implementación de la paralelización, aplicada a los algoritmos ya antes planteados.

El paralelismo de datos es el paralelismo que utiliza múltiples unidades funcionales o procesos para aplicar la misma operación de forma simultánea en un conjunto de datos. En la práctica hay más datos que procesadores y por lo tanto es necesario dividir los de datos, esto da origen a dos estrategias de paralelismo de datos:

- **Paralelismo Pre-Itinerado:** El número de datos por unidad funcional se determina antes del procesamiento
- **Paralelismo Auto-Itinerado**: La asignación ocurre en tiempo de corrida. Se mantiene una "*Ready list*" (lista de trabajos a realizar).

La estrategia escogida para el acabado del algoritmo paralelo es la forma de Auto-Itinerado, ya que se asigna en tiempo real de ejecución que nodo es el encargado de procesar la imagen dada.

2.6 Diseño de la Base de Datos

Uno de los pasos cruciales en la construcción de una aplicación que maneje datos es sin duda el diseño de una base de datos. En la base de datos que se utiliza en la aplicación que se está realizando como propuesta de solución, se toman varias consideraciones, entre las que se encuentran la velocidad y la facilidad al acceso de la información para extraerla. La misma no es de gran complejidad puesto que cuenta con una sola tabla de solo 2 campos, el diseño se muestra en la siguiente figura.

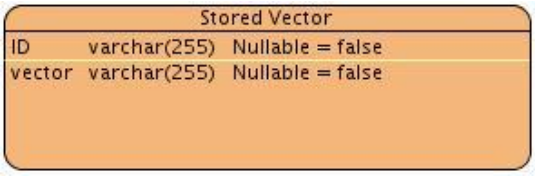


Fig. 10 Modelo de datos persistentes

2.7 Implementación

La implementación se inicia a partir de los resultados obtenidos en el diseño y se implementa el sistema en términos de componentes. Los componentes conforman lo que se conoce como un modelo de implementación al describir los componentes a construir, su organización y dependencia entre nodos físicos en los que funcionara la aplicación. Entre los componentes podemos encontrar:

- **wavelet.py:** módulo encargado del trabajo con las transformadas wavelets.
- **hu.py:** módulo encargado del trabajo con los momentos de hu.
- **distributed.py:** módulo encargado del trabajo en paralelo de los algoritmos.
- **distancia.py:** módulo encargado del trabajo con las distancia entre vectores.

- **_common.py:** módulo que implementa métodos comunes entre los distintos algoritmos.

Estos componentes están estrechamente relacionados con las librerías utilizadas para el desarrollo de la aplicación como se muestra en la fig. 11.

2.7.1 Diagrama de Componentes

El diagrama de componente ilustra los componentes que serán usados para construir el sistema. UML define cinco estereotipos estándar que se le aplican a los componentes: ejecutable, bibliotecas, tabla, archivo y documento. Los distintos componentes se pueden agrupar en paquetes según un criterio lógico y con vista a simplificar su implementación.

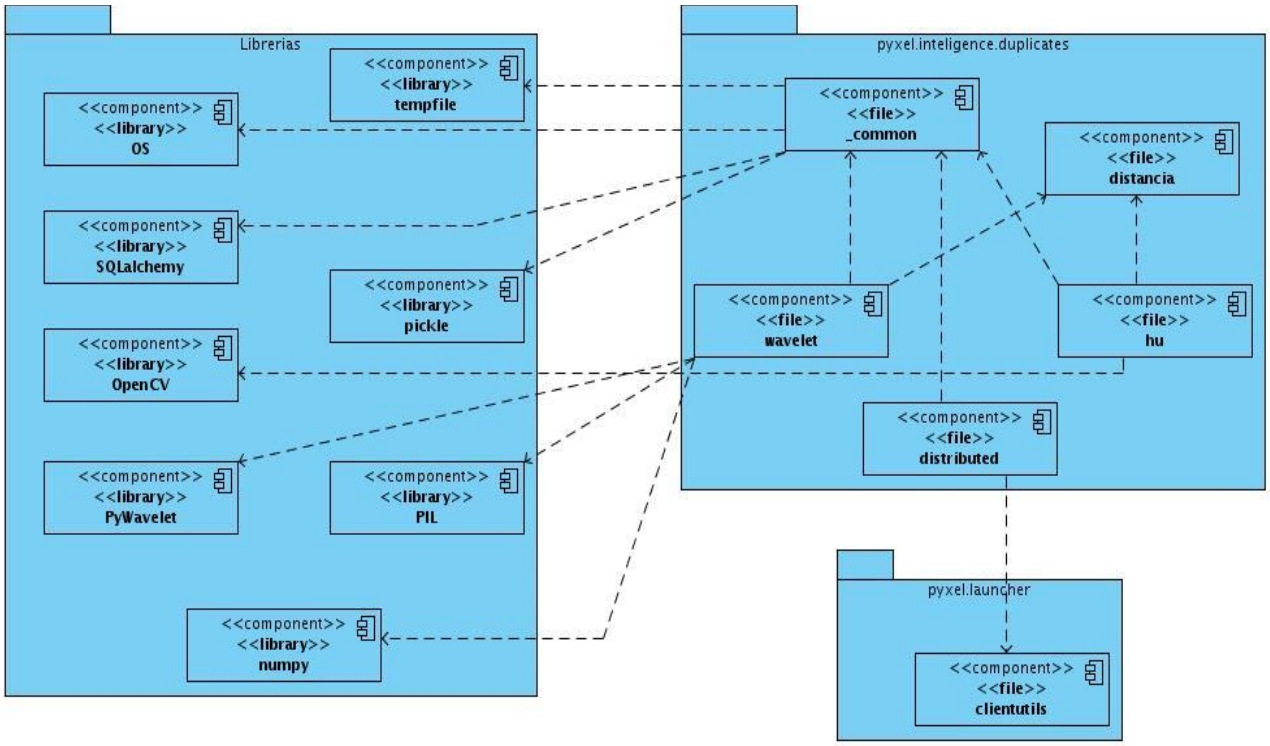


Fig 11 Diagrama de componentes

2.7.2 Experimentos y pruebas

El enfoque principal de los experimentos realizados en esta etapa es asegurarse de que la función de los vectores característicos es adecuada para la detección de copias de la imagen. La implementación fue hecha en *Python,* utilizando una plataforma Unix, en estaciones de trabajo Debian Linux, Inte(R) Core(TM)2 Duo CPU 2.20Hz. El uso de

una rápida transformación wavelets para extraer los vectores de características con WaveDuplicates para una colección de prueba de 300 imágenes a color requiere aproximadamente de 6 minutos. En el caso de la extracción de los vectores con HUDuplicates requiere de un tiempo mucho menor, aproximadamente 35 minutos. Estos tiempos son mejorados grandemente con el uso de la versión paralela de los algoritmos, siendo el ejemplo de que WaveDuplicates para la misma colección de imágenes y con tres nodos de trabajo requiere aproximadamente de cuatro minutos, tiempo que se reduce a medida crezca el numero de nodos brindando el servicio de detección de duplicados.

Para llevar a cabo los experimentos se hizo la selección de una colección de imágenes donde se insertaron versiones modificadas de imágenes dadas en la colección, donde se incluyeron transformaciones de rotación, escalado, cambios en la iluminación conversión a blanco y negro entre otras

Para la realización de las pruebas con motivo de puntualizar la calidad de los algoritmos se usaron las métricas de precisión y *recall* (Baeza-Yates, y otros, 1999) muy usadas en la actualidad para garantizar la calidad de los sistemas de recuperación de información en sentido general.

En sentido general se tuvieron en cuenta las siguientes variables para el cálculo de las métricas.

C, Colección de pruebas

TP, Verdaderos positivos de una consulta

TN = C – TP, Verdaderos negativos de una consulta

FP, Falsos positivos de una consulta

FN, Falsos negativos de una consulta

Prueba 1

Precisión = TP/TP+FP

Recall = TP/TP+FN

Algoritmo	TP+FN	TP	TP+FP	Precisión	Recall
WaveDuplicates	12	7	7	1	0.583
HuDuplicates	12	6	6	1	0.5

Prueba 2

Precisión = TP/TP+FP

Recall = TP/TP+FN

Algoritmo	TP+FN	TP	TP+FP	Precisión	Recall
WaveDuplicates	4	4	4	1	1
HuDuplicates	4	4	4	1	1

Prueba 3

Precisión = TP/TP+FP

Recall = TP/TP+FN

Algoritmo	TP+FN	TP	TP+FP	Precisión	Recall
WaveDuplicates	12	5	6	0.833	0.416
HuDuplicates	12	4	4	1	0.333

Prueba 4

Precisión = TP/TP+FP

Recall = TP/TP+FN

Algoritmo	TP+FN	TP	TP+FP	Precisión	Recall
WaveDuplicates	5	5	5	1	1
HuDuplicates	5	5	6	1	0.833

Se considera que las mejoras se podrán realizar al menos en este aspecto:

- Algunas imágenes que no se asemejan a la semántica de la consulta también se incluyen como los sospechosos. Esto se debe al rango de confianza propuesto. Al aumentar el rango de confianza se obtienen más resultados verdaderos positivos de imágenes que contienen transformaciones más complejas pero se corre el riesgo de que imágenes no relacionadas con la consulta sean inmiscuidas como sospechosas. Esto ocurre con las imágenes de rico contenido y con varias trasformaciones

Conclusiones

Al lograr los objetivos trazados en el presente trabajo, se implementaron algoritmos de detección automática de imágenes duplicadas lo cual ayudará grandemente a la gestión de imágenes en la prensa cubana. Después de un profundo estudio se escogieron como candidatos los métodos de cálculo de Momentos de HU a las imágenes y el cálculo de las transformadas Wavelets a las mismas, los cuales se consideran satisfactorios en cuanto a velocidad de procesamiento, factibilidad de implementación, tiempo y recursos. Estos algoritmos aun están en fase experimental pero son completamente funcionales.

Recomendaciones

- Se recomienda la implementación de estos algoritmos en otros lenguajes de programación tales como C a fin de hacer comparaciones en cuanto a velocidad de procesamiento y tiempo de ejecución.
- Integrar estos algoritmos como servicios en el archivo fotográfico Pyxel para la gestión de imágenes.
- Investigar sobre otras formas de realizar la detección de imágenes de duplicados a fin de garantizar mejores resultados.

[En línea] http://opencv.willowgarage.com/wiki/.

[En línea] http://pyro.sourceforge.net/.

[En línea] http://numpy.scipy.org/.

[En línea] http://www.python.org/.

[En línea] http://pypi.python.org/pypi/PyWavelets/.

Acosta, Manuel Vázquez. 2009. Gforge Facultad X: Pyxel. [En línea] 2009. http://gforge.f10.uci.cu/projects/pyxel/.

Ailyn Alfonso González, Joel Armada Herrera. 2008. *Trabajadores. Creación de un sitio modelo para el desarrollo de portales web utilizando el CMS Plone.* 2008.

Arnold W.M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, y Ramesh Jain. 2000. *Content-Based Image Retrieval at the End of the Early Years.* s.l. : IEEE Transactions, 2000.

Baeza-Yates, R. y Ribeiro-Neto, B. 1999. *Modern Information Retrieval.* s.l. : ACM Press, 1999.

Baojiang Zhong, Wenhe Liao. 2007. *Direct Curvature Scale Space: Theory and Corner Detection.* 2007.

Blaise Barney, Lawrence Livermore. Introduction to Parallel Computing. [En línea] https://computing.llnl.gov/tutorials/parallel_comp/.

Chang, J. R. Smith and S.-F. 1996. *Visualseek: A fully automated content-based image query system.* 1996.

Edward Y. Chang, James Ze Wang, Chen Li, and Gio Wiederhold. 1998. *A Replicated Image Detector for the World-Wide Web.* s.l. : Stanford University, 1998.

Etemad Kamran, David S. Doermann, y Rama Chellapa. 1997. *Multiscale Document Page Segmentation Using Soft Decision Integration.* s.l. : IEEE Transactions, 1997.

Hu, M. K. 1962. *Visual Pattern Recognition by Moment Invariants.* 1962.

Janoff, Linda Rising and Norman S. 2007. *The Scrum Software Development Process for Small Teams.* 2007.

Jensen, la Cour-Harbo. 2001. *Ripples in Mathematics.* 2001.

Kruchten, P. 2003. *The Rational Unified Processj An Introduction. Third Edition.* 2003.

M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang. 1995. *Query by image and*

video content: the qbic system. 1995.

M.C. Aranda, J. Galindo, A. Urrutia. 2002. *Museos Digitales en Internet: Modelo EER Difuso y Recuperación de Imágenes Basada en su Contenido.* 2002.

Manjunath, W. Y. Ma y B. S. 1997. *NeTra: A Toolbox for Navigating Large Image Databases.* 1997.

Manuel Agustí i Melchor, Jose Miguel Valiente González. *Bases de datos para Multimedia: Recuperación por Contenido.*

Mery, Domingo. 2006. *Extracción de Características.* s.l. : Departamento de Ciencia de la Computación Universidad Católica de Chile, 2006.

Sadegh Abbasi, Farzin Mokhtarian, Josef Kittler. 1999. *Curvature scale space image in shape similarity retrieval.* 1999.

Schwaber, Ken. 2004. *Agile Project Management with Scrum.* s.l. : Microsoft Press, 2004.

Siddique, Sharmin. 2002. *A Wavelet Based Technique for Analysis and Classification of Texture Images.* 2002.

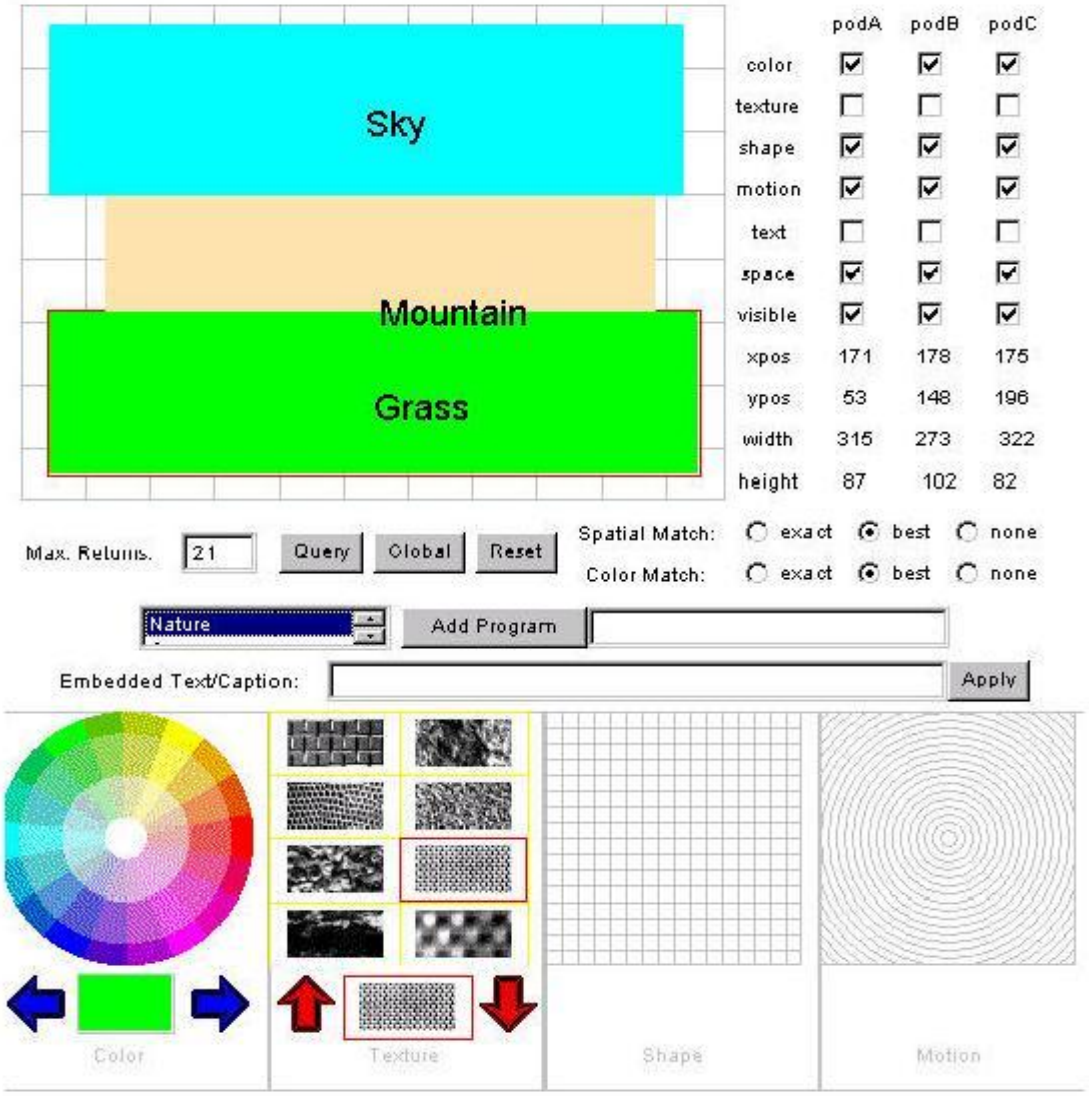
Suk, J. Flusser and T. 2006. *Rotation Moment Invariants for Recognition of Symmetric Objects.* 2006.

Winograd, S. 1978. *On computing the discrete Fourier transform.* 1978.

Anexo 1: Sistema QBIC

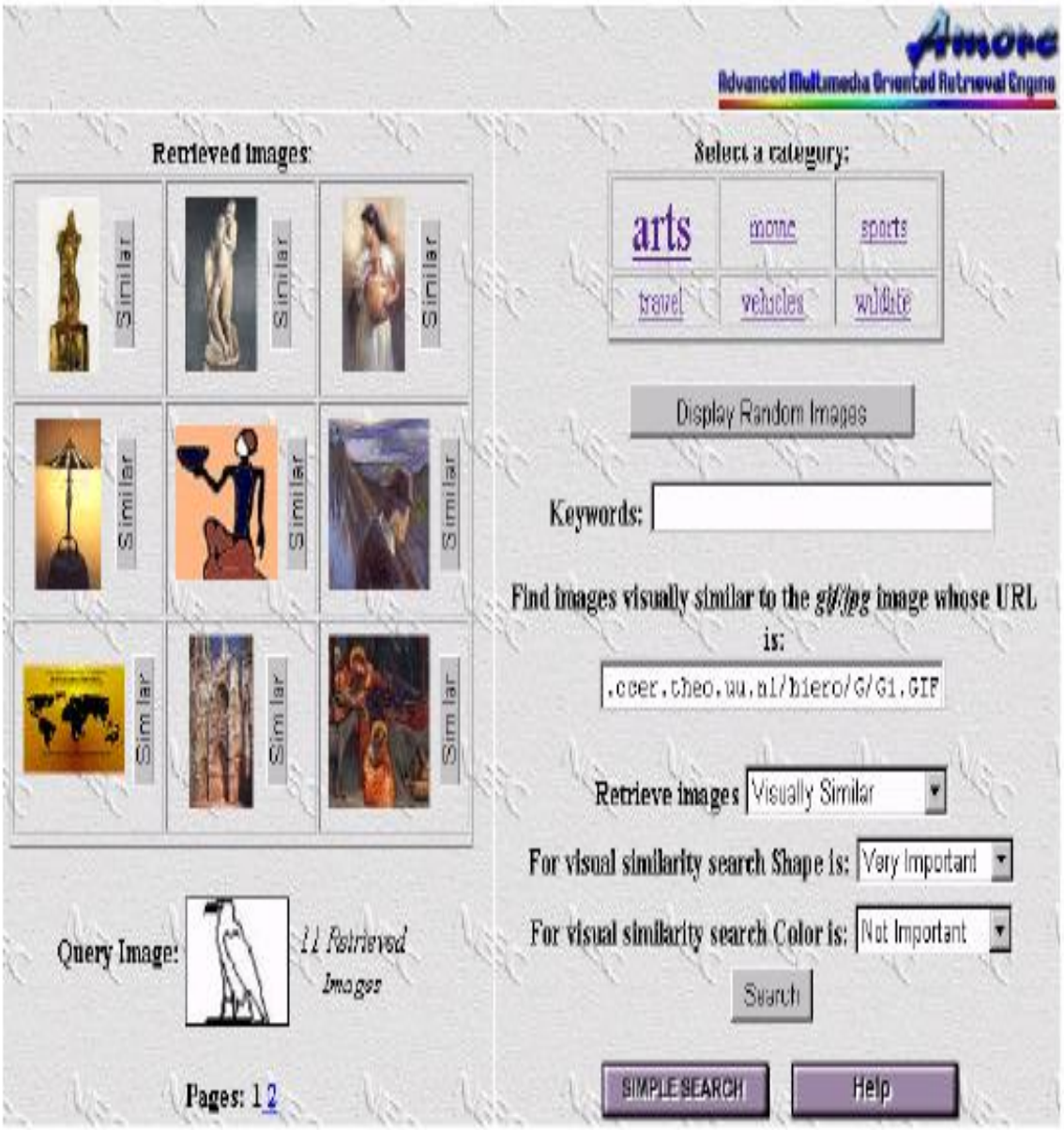
Anexo 1 QBIC

Anexo 2: Sistema VisualSeek



Anexo 2 VisualSeek

Anexo 3: Sistema AMORE



Anexo 3 AMORE

Anexo 4: Sistemas de recuperación de imágenes

Sistema de recuperación de información visual	Características de búsqueda	Organismo
a-LISP (Automatic linguistic indexing of pictures)	- Color promedio de bloques. - Textura, a través de transformadas Wavelets.	The Pennsylvania State University
Amore (Advanced Multimedia Oriented Retrieval Engine)	- Regiones homogéneas de colores. - Keywords para recuperación semántica usando un modelo de vector de espacio.	NEC USA Inc. C&C Research Laboratories.
CIRES (Content-Based Image Retrieval System)	- Histograma de color (espacio de colores). - Textura.	University of Texas at Austin.
C-Bird (Content-Based Image Retrieval from Digital Libraries)	- Color. - Textura. - Color más frecuente. - Vector de orientación más frecuente. - Vector de cromaticidad.	Escuela de Ciencias de la Computación de la Universidad de Simon Fraser.
Excalibur Visual RetrievalWare	- Aspecto de proporción. - Brillo. - Color local y global. - Forma. - Textura. - Orientación relativa. - Curvatura. - Histograma de color.	Excalibur Technologies ahora Convera Corporation.
PhotoBook	- Forma. - Textura. - Descripción textual.	MIT MediaLab.
QBIC (Query By Image Content)	- Promedio de color. - Texturas. - Contraste. - Area. - Circularidad. - Excentricidad. - Momentos invariantes.	IBM
Retrievr	- Transformadas Wavelet para obtener coeficientes de color	University of Washington System One Inc.
SIMPLIcity (Semantics-sensitive Integrated Matching for Picture Libraries)	- Histograma de color. - Capas de colores. - Regiones de colores. - Texturas. - Forma y espacios.	The Pennsylvania State University
SQUID (Shape Queries Using Image Databases)	- Excentricidad. - Circularidad. - Forma. - Proporción de curvatura.	CVSSP (Centre for Vision, Speech and Signal Processing).
VisualSEEK	- Color y textura. - Regiones de colores.	Columbia University
WebSEEK	- Keywords de tags de imágenes web. - Histograma de color.	Columbia University

Anexo 4 Sistemas de recuperación de imágenes

Anexo 5: Metodología SCRUM:

La metodología Scrum es una metodología diseñada principalmente para grupos de trabajo pequeños, tiene mucho éxito en el desarrollo del software de la actualidad, define cada proceso como iterativo e incremental. En esta metodología se divide por *Sprints* el desarrollo del trabajo. Durante cada *Sprint* el grupo de trabajo crea un incremento del software potencialmente entregable. Los objetivos de cada Sprint son determinados en el *Scrum Planning* durante esta reunión, el *Product Owner* identifica los elementos, que quiere ver completados y los hace del conocimiento del equipo. Posteriormente el equipo determina la cantidad de ese trabajo que puede comprometerse a completar durante el siguiente Sprint. Este ciclo se repite a lo largo de todo el proyecto. (Kruchten, 2003)

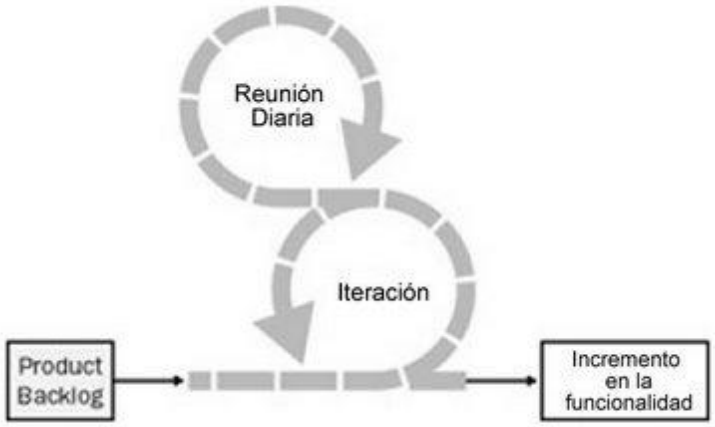


Figura 1: Esqueleto de Scrum

Esta metodología opera de esta manera:

1. Al comienzo de la iteración el equipo revisa que es lo que debe hacer.
2. Luego selecciona lo que cree que puede hacer para tener un incremento y un potencial prototipo funcional al término de la iteración.
3. El equipo se separa y hace su mejor esfuerzo por el resto de la iteración.
4. Al terminar la iteración el equipo presenta el incremento de la funcionalidad que construyó de manera que los otros miembros del equipo puedan revisar las funcionalidades y hacer las modificaciones necesarias al proyecto.

Roles en Scrum

En Scrum se definen varios roles, estos están divididos en dos grupos: cerdos y

gallinas. Los nombres de los grupos están inspirados en el chiste sobre un cerdo y una gallina que se relata a continuación. (Janoff, 2007)

> *Un cerdo y una gallina se encuentran en la calle. La gallina mira al cerdo y dice: "Hey, ¿por qué no abrimos un restaurante?" El cerdo mira a la gallina y le dice: "Buena idea, ¿cómo se llamaría el restaurante?" La gallina piensa un poco y contesta: "¿Por qué no lo llamamos "Huevos con jamón?" "Lo siento pero no", dice el cerdo, "Yo estaría comprometido pero tú solamente estarías involucrada".*

De esta forma, los *cerdos* están comprometidos a construir software de manera regular y frecuente, mientras que el resto son *gallinas*, interesados en el proyecto pero realmente irrelevantes porque si éste falla no son los que se habían comprometido a sacarlo adelante. Las necesidades, deseos, ideas e influencias de los roles *gallina* se tienen en cuenta, pero no de forma que pueda afectar, distorsionar o entorpecer el proyecto Scrum. (Janoff, 2007)

Roles de Cerdo

- *Product Owner*
- *ScrumMaster* (o Facilitador)
- Equipo de trabajo

Roles de Gallina

- Usuarios
- *Stakeholders* (Clientes, Proveedores)
- *Managers*