Paralelización del algoritmo de análisis de similitud de imágenes basado en contenido. Escalabilidad y eficiencia en queries simples y múltiples.

Armando E. De Giusti
Marcelo Naiouf
Laura C. De Giusti

L.I.D.I., Facultad de Informática, Universidad Nacional de La Plata
La Plata, Argentina, 1900
{degiusti, mnaiouf, ldgiusti}@lidi.info.unlp.edu.ar

Resumen

Los métodos tradicionales que analizan similitud de imágenes basados en una única firma fallan cuando las imágenes comparadas tienen sólo algunas regiones similares, o cuando contienen patrones similares trasladados o en escalas diferentes. En estos casos, la efectividad del análisis de similitud puede incrementarse usando firmas múltiples, correspondientes a diferentes regiones en las imágenes comparadas.

Una solución aceptada procesa imágenes basadas en múltiples firmas, usando los coeficientes de la transformada de wavelet. Los resultados experimentales sobre conjuntos de datos reales confirman la efectividad del algoritmo y verifican la relación teórica entre las principales componentes del tiempo de procesamiento y el tamaño de la imagen.

Como los algoritmos secuenciales requieren cálculo intensivo, el tiempo total de procesamiento (Tpt) se incrementa exponencialmente con el tamaño de la imagen. Para solucionar este problema se propone una arquitectura y un algoritmo paralelo, basado en procesadores homogéneos. El speedup teórico, escalabilidad y eficiencia se obtienen como funciones basadas en el número de procesadores, para realizar queries (búsquedas) simples (entre dos imágenes).

Finalmente se extiende la arquitectura y el algoritmo para múltiples queries, buscando la relación óptima entre performance y costo. Además, son mencionado algunos problemas de la arquitectura real y ventajas de utilizar una arquitectura con memoria distribuida y compartida.

PALABRAS CLAVES: Procesamiento de imágenes, Algoritmos paralelos, Similitud, Speedup, Escalabilidad, Eficiencia.

1. INTRODUCCIÓN

El procesamiento de imágenes es un área de creciente interés dentro de la Ciencia de la Computación. Los avances tecnológicos en la velocidad de los procesadores, las facilidades de comunicación, multimedia, capacidades gráficas e Internet favorecen a una cantidad de aplicaciones reales tales como data mining de imágenes, procesamiento de imágenes médicas, predicción del clima y la agricultura, producción de video y TV, o E-commerce [1].

El análisis de similitud de imágenes basado en contenido es una derivación de los problemas de reconocimiento de patrones tradicionales. Se puede descomponer el problema de la siguiente manera:

a Obtener una representación compacta de la imagen y cada "una" de las regiones que contienen objetos que podrían considerarse representativos [2] [3].

b Establecer un modelo de similitud a fin de tener une métrica asociada con la porción del área de dos imágenes que son "similares" [4].

c Reconocer objetos similares a diferente escala o en posiciones distintas en las dos imágenes.

Los enfoques típicos para análisis de similitud basado en contenido computan algún tipo de "rasgo firma" basados en histogramas de color, textura o coeficientes funcionales como la transformada de wavelet [5] [6] [7]. Este tipo de soluciones, basadas en un atributo de la imagen fallan cuando las imágenes tienen objetos o regiones similares pero con diferentes tamaños o posiciones. Una alternativa tradicional es la segmentación de imágenes (un paso tradicional en los métodos de reconocimiento de patrones) [8] [9] en donde objetos individuales son "extraídos" y comparados. Esto es caro en tiempos de procesamiento y muy difícil de utilizar con objetos escalados [10].

Modelos de similitud de imágenes Multi-Wavelet

El uso de Haar Wavelets [11] y de ventanas de diferentes tamaños son las ideas principales presentadas en WALRUS [12] [13]. La transformada de wavelet es utilizada sobre múltiples ventanas deslizantes para descomponer jerárquicamente una imagen completa, con lo cual las regiones y características de los objetos son capturadas. Como se muestra en [14], la transformada de wavelet es robusta con respecto a desplazamientos de la intensidad de colores y también tiene en cuenta la información de textura y forma; el uso de ventanas deslizantes soluciona los problemas de escalado y traslación, por lo cual se pueden obtener resultados eficientes en similitud de imágenes con la representación multi-wavelet.

El modelo de similitud de WALRUS considera que dos imágenes Q y T son similares si existe un conjunto de pares similares Q_i, T_i tal que el área de matching de las regiones Q_i, T_i, comparada con el área total de las dos imágenes Q y T está por encima de un límite δ especificado por el usuario. El uso de coeficientes de wavelet sobre ventanas deslizantes permite construir un árbol jerárquico para las imágenes comparadas, en el cual las regiones similares Q_i, T_i son nodos donde los coeficientes de

wavelet difieren en menos de un límite δ_i especificado. La Fig. 1 muestra dos imágenes consideradas similares con este modelo.

$$\delta \leq \frac{Area\left(\bigcup_{i=1}^{k} Q_i\right) + Area\left(\bigcup_{i=1}^{k} T_i\right)}{Total \quad Area\left(Q+T\right)} \qquad \textbf{Eq. 1}$$

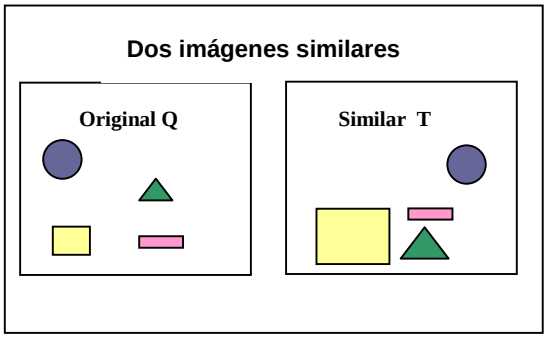

Fig. 1

En [15] y [17] hay resultados de trabajos experimentales para algoritmos secuenciales usando imágenes (RGB) conteniendo figuras geométricas con diferentes niveles de similitud. La implementación que se presenta se realizó en un único transputer utilizando una distancia fija de 2 para las ventanas deslizantes, y extiende el cálculo de los coeficientes de wavelet desde ventanas de 2x2 hasta formar una única ventana con toda la imagen.

Las imágenes relacionadas semánticamente, con objetos trasladados o escalados tienen un reconocimiento efectivo, como se detalla en [14]. Dependiendo de la definición de δ, la precisión en los resultados varía entre 70% y 90%. El algoritmo también es eficiente en determinar imágenes que no están relacionadas (imágenes "diferentes" o "no similares"), y para una similitud de δ, la efectividad es cercana al 100%. Resultados análogos son reportados en [13] con otro tipo de imágenes.

Nuestra contribución

Se estudió la complejidad en tiempo del modelo de similitud de imágenes multi-wavelet sobre un procesador secuencial. Se analizaron las componentes de tiempo en el algoritmo secuencial, mostrando que el cómputo dinámico de los coeficientes wavelet para las ventanas deslizantes (Tsw) eran la componente principal del tiempo de procesamiento total (Tpt). Se estudió la relación teórica entre el tiempo de procesamiento total y el tamaño de la imagen y se verificó con trabajo experimental [15][17].

Utilizando estos resultados, se analizan una arquitectura y un algoritmo paralelo, estudiando el problema de escalabilidad con un número diferente de procesadores e incrementando el tamaño de la imagen. Se obtuvieron el speedup teórico y la eficiencia. Para estos resultados, se investigó una expresión de los procesadores ociosos en la arquitectura propuesta, para una imagen con un tamaño fijo (1024 x 1024 pixels).

Por último, se analizaron múltiples búsquedas (queries de imágenes) para la similitud de imágenes en una base de datos. Mejorando la arquitectura propuesta para reducir los procesadores ociosos, se obtuvo un mejor speedup y eficiencia global.

Además se presentan algunos inconvenientes de la implementación real e ideas en el futuro utilizando una arquitectura MIMD con memoria compartida y distribuida [16].

2. ANÁLISIS DE LA SOLUCIÓN SECUENCIAL

Esquema del Algoritmo

Dadas dos imágenes I_1 e I_2.

Para cada imagen I_I
*Crear y analizar las estructuras de datos. { Tiempo = **Tds** }*
*Leer los datos de la imagen. {Tiempo = **Trd** }*
*Almacenar cada componente RGB. {Tiempo = **Tli** }*
Para cada componente RGB
*Computar los coeficientes wavelet p/ c/ vent deslizante.Insertarlos en el árbol.{Tiempo = **Tsw**}*

Para cada imagen I_I
Para cada componente RGB
*Computar pixels agrupados { Tiempo = **Tjp** }*
*Calcular la cantidad de clusters comunes entre las dos imágenes. { Tiempo = **Tcc** }*
*Computar el porcentaje de similitud para c/ comp. RGB y determinar la similitud {Tiempo = **Tfr** }*

$$\therefore Tpt = Tds + Trd + Tli + Tsw + Tjp + Tcc + Tfr$$

Componentes de tiempo en el algoritmo secuencial.

Tds, Trd y Tli son tiempos secuenciales los cuales dependen linealmente del tamaño de la imagen, y no tendrán cambios en una solución paralela en un multiprocesador con memoria distribuida; estos no tienen efecto en la complejidad del algoritmo paralelo. Tfr es un cálculo simple para obtener el nivel de similitud entre dos imágenes utilizando la fórmula [Eq. 1].

El más significativo es Tsw (tiempo de calcular y almacenar los coeficientes de las ventanas deslizantes) y Tcc (tiempo de calcular los clusters coincidentes). En [17], Tsw, Tjp y Tcc son estudiados en función del tamaño de la imagen para obtener una expresión del tiempo de complejidad que permita paralelizar los componentes más críticos de la solución paralela. Entonces, considerando imágenes de nxn y ventanas deslizantes a una distancia de 2, se tiene:

$$Tsw = \sum_{i=2}^{\log_2 n} \left(\left(\frac{n-2^i}{2} + 1 \right)^2 \left(2^{2i} - 4\right) T_c \right) + T_{insert}$$

donde T_c es el tiempo de copiar un pixel.
y T_{insert} es el tiempo para insertar los elementos en un árbol para el caso peor:

$$T_{insert} = \frac{k^2 - k}{2} T_c + kT_{assign} \qquad con \;\; k = \sum_{i=2}^{\log_2 n} \left(\frac{n-2^i}{2} + 1 \right)$$

$$Tjp = k * \left(4*4\frac{k}{2} T_{compare} + T_{add} \right) + T_{assign}$$

$$Tcc = k * \left(\left(\log_2 k * T_{compare} + T_{assign} \right) + 2 * \left(k + T_{agg} \right) \right)$$

donde,
T_{assign} es el tiempo de asignar un valor.
T_{add} es el tiempo de sumar dos valores.
$T_{compare}$ es el tiempo de comparar 2 matrices.
T_{agg} es el tiempo de insertar un elemento en la lista.

Está claro que Tsw es el componente fundamental en la complejidad del tiempo, y puede ser representado como una función dependiendo del tamaño de la imagen. La Tabla 1 muestra Tsw para imágenes desde 4x4 a 4096x4096 pixels, y la Fig. 2 muestra la complejidad del tiempo teórico para Tsw, Tjp y Tce.

	16x16	32x32	64x64	128x128	256x256	512x512	1024x1024	2048x2048	4096x4096
Tsw	7032	1.03E6	1.55xE7	2.4xE7	3.76E8	6.02xE9	9.45xE10	1.51xE12	2.39xE13

Tabla 1

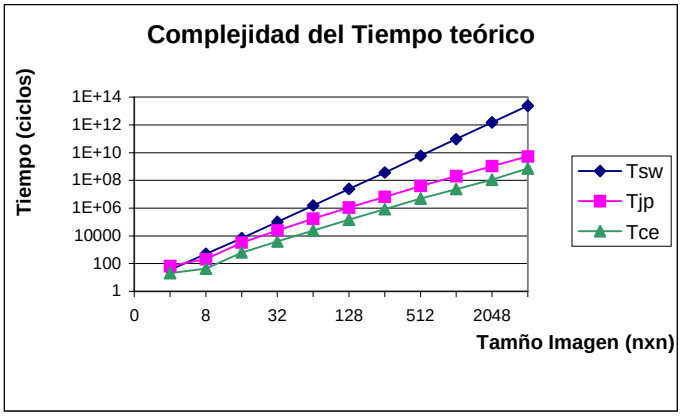

Fig 2

La expresión teórica deTsw es $O(10^{\log_2 n})$

Como se detalla en [17], los resultados experimentales con transputers e imágenes reales muestran una significativa coincidencia entre la expresión teórica Tsw y el tiempo de procesamiento.

3. SOLUCIÓN PARALELA

Utilizando una grilla de P procesadores, el algoritmo secuencial puede ser paralelizado en $\log_2(n)$ pasos. En cada paso se calculan los coeficientes 2^i x 2^i, $\forall i = 1..\log_2(n)$. Este esquema permite una nueva expresión para Tsw:

$$Tswp = \sum_{i=1}^{\log_2 n} \left(\frac{\left(\frac{n-2^i}{2}+1\right)^2}{min\left\{P, \left(\frac{n-2^i}{2}+1\right)^2\right\}} \left(2^{2i}-4\right) T_c \right) + T_{insert}$$

Utilizando una grilla de $P = \frac{n}{2} x \frac{n}{2}$ procesadores, la cantidad de procesadores es mayor o igual que la cantidad de ventanas V_i a calcular en la etapa *i*. Esto no ocurre para grillas con menor cantidad de procesadores; esto es, si $P < V_i$ en una etapa dada, un mismo procesador debe calcular más de una ventana.

Aunque usar una grilla de $P = \frac{n}{2} x \frac{n}{2}$ procesadores resulta en mayores valores de speedup, utilizar una grilla con menor cantidad de procesadores es una solución más realista. Reducir el número de procesadores afectará el speedup teórico, pero reducirá la comunicación lo cual es un factor crítico en la implementación real.

La Tabla 2 muestra los tiempos paralelos para diferentes configuraciones de grilla y tamaños de imagen. La Tabla 3, y las Figs. 3 y 4 muestran el speedup para diferentes configuraciones de grilla y tamaños de imagen.

Tam / Grilla	16x16	32x32	64x64	128x128	256x256	512x512	1024x 1024	2048x 2048	4096x 4096
$\log_2$n x $\log_2$n	1147.5	7050.2	5.51xE4	5.38xE5	6.08xE6	7.43xE7	9.51xE8	1.25xE10	1.67xE11
n/2 x n/2	972	4032	16308	65448	262044	1048464	4194180	16777080	6.71xE7
n/4 x n/4	1147.5	4618.6	18302.5	72562.32	288553	1.15xE6	4.59xE6	1.83xE7	7.33xE7
n/8 x n/8	2322	9295.4	36382.4	142829.49	5.64xE5	2.24xE6	8.93xE6	3.56xE7	1.42xE8
n/16 x n/16	7020	27999	108702	423898.2	1.66xE6	6.60xE6	2.38xE7	1.04xE8	3.73xE8
$\log_2$n/2 x logn/2	2322	9294.7	73988.6	715376.97	7.88xE6	9.39xE7	1.17xE9	1.37xE10	1.99xE11

Tabla 2

Tam Grilla	16x16	32x32	64x64	128x128	256x256	512x512	1024x 1024	2048x 2048	4096x 4096
$\log_2 n$ x $\log_2 n$	6	14	28	45	62	81	99	120	142
n/2 x n/2	7	25	95	367	1437	5723	22552	90004	357495
n/4 x n/4	6	22	84	330	1303	5216	20595	82273	326954
n/8 x n/8	3	11	42	168	666	2676	10587	42338	168341
n/16 x n/16	1	3	14	57	225	908	3965	14393	64188
$\log_2 n/2$ x logn/2	3	11	21	33	48	64	80	95	119

Tabla 3

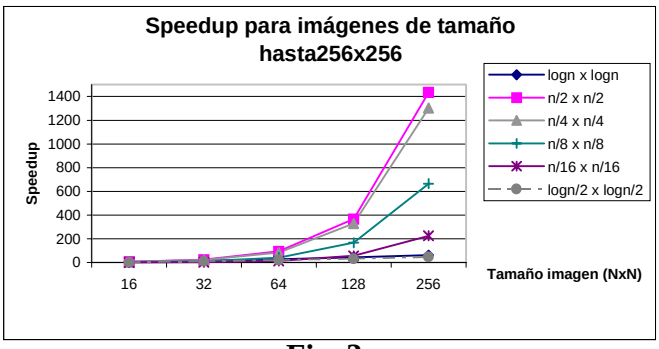

Fig. 3

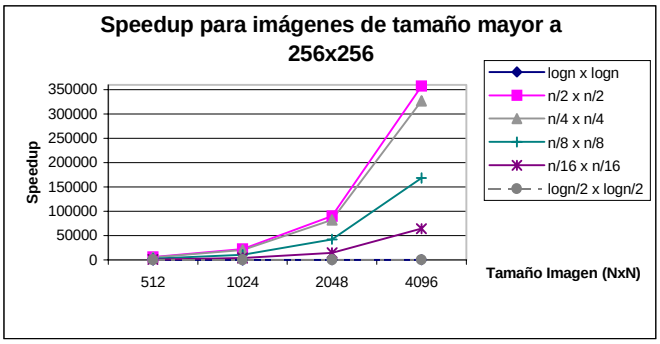

Fig. 4

Reducir la grilla de procesadores puede ser beneficioso para el speedup con un menor costo y menor complejidad en las comunicaciones.

4. EFICIENCIA DE LA SOLUCIÓN PARALELA

La eficiencia puede ser definida como:

$$Eficiencia = \frac{Speedup}{P}$$

Intuitivamente, una mayor eficiencia implica menos procesadores libres a través de los $\log_2(n)$ ciclos del algoritmo. La Fig. 5 muestra la eficiencia para las diferentes configuraciones y tamaños de imágenes.

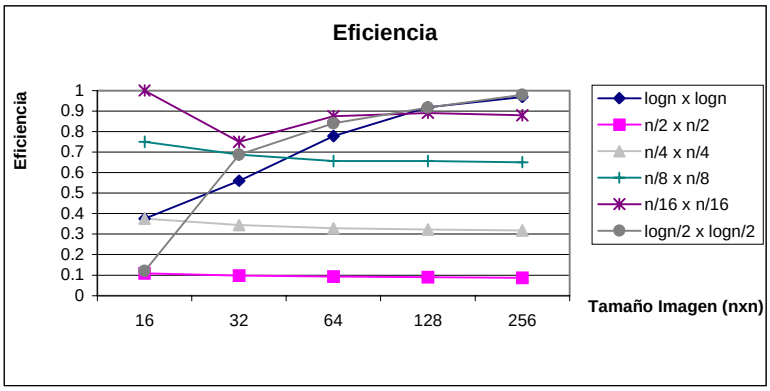

Fig. 5

Se puede observar que los peores valores de eficiencia corresponden a las configuraciones $P = \frac{n}{2} x \frac{n}{2}$, $P = \frac{n}{4} x \frac{n}{4}$ y $P = \frac{n}{8} x \frac{n}{8}$ como era esperable.

5. ESTUDIO DE LOS PROCESADORES OCIOSOS PARA UNA COMPLEJIDAD FIJA

Sea un tamaño de imagen fijo de 1024x1024 pixels (n=1024). Dado que el algoritmo comprende $\log_2(n)$ etapas, en este caso este número es 10. En cada etapa, hay una cantidad de ventanas a calcular. Luego, se puede definir un arreglo *V* donde *V*[*i*] = número de ventanas cuyos coeficientes deben calcularse en la etapa i ($\forall i = 1..\log_2 n$).

El número total de procesadores libres es:

$$P_L = \sum_{i=1}^{\log_2 n} (P - V[i]) \qquad \textit{donde P es el número de procesadores libres}$$

Dado que siempre *V*[*1*] = *P*, la fórmula anterior puede reescribirse como

$$P_L = \sum_{i=2}^{\log_2 n} (P - V[i]) \qquad \forall P \geq V[i]$$

Si *P* < *V*[*i*], entonces cada procesador tiene asignada una carga mayor al cálculo de los coeficientes de una ventana, en cuyo caso no se contabilizan "procesadores libres" en la etapa *i*. Nótese que si *V*[*i*] no es múltiplo de *P*, habrá un cierto porcentaje de procesadores no ocupados que se han despreciado en una primera aproximación.

La Tabla 4 muestra P_L y la cantidad de procesadores ociosos en cada etapa.

Procesadores	P_L	Procesadores ociosos para cada etapa $Et_i \forall i= 2..\log_2 n$									
		Et_1	Et_2	Et_3	Et_4	Et_5	Et_6	Et_7	Et_8	Et_9	Et_{10}
$\log_2 n/2$ x $\log_2 n/2$ (81)	80	0	0	0	0	0	0	0	0	0	80
$\log_2 n$ x $\log_2 n$ (100)	99	0	0	0	0	0	0	0	0	0	99
n/16 x n/16 (4096)	4095	0	0	0	0	0	0	0	0	0	4095
n/8 x n/8 (16384)	16383	0	0	0	0	0	0	0	0	0	16386
n/4 x n/4 (65536)	65535	0	0	0	0	0	0	0	0	0	65535
n/2 x n/2 (262144)	689823	0	1023	3063	7119	15135	30783	60543	113919	196095	262143

Tabla4

6. MÚLTIPLES CONSULTAS

En aplicaciones de consultas por contenido sobre grandes Bases de Datos de imágenes existe la necesidad de comparar una imagen contra otras *M* para analizar a cuáles es similar. Esto lleva a que el problema se convierta en una secuencia de comparaciones de similitud de imágenes.

Optimización para una complejidad fija de la imagen por modificación en la arquitectura

Para evaluar mejoras en speedup y eficiencia para secuencias de imágenes, se analiza una modificación en la arquitectura consistente en agregar procesadores.

Sea una consulta de similitud sobre una secuencia de *M* imágenes de nxn pixels. Luego, el tiempo secuencial está dado por $T_{sec} = M*Tsw$. Por ejemplo, para M=100 y n=1024, T_{sec} = 9.45xE12. Por otra parte, la Tabla 5 muestra los tiempos paralelos para las diferentes grillas, M=100 y n=1024.

Grilla / Tam	10x10	512x512	256x256	128x128	64x64	9x9
1024x1024	9.51xE10	4.19xE8	4.59xE8	8.93xE8	2.38xE9	1.17xE11

Tabla 5

Como puede verse en la Tabla 4, la última etapa de cada análisis de similitud es crítica porque sólo se utiliza un procesador para calcular la ventana de 1024x1024, por lo que *P*-1 procesadores se encuentran ociosos. Si el procesamiento del query se considera un pipe donde se agrega una *etapa de procesamiento* de manera que ésta es la encargada de realizar el cálculo del paso $\log_2(n)$ para la imagen M_j mientras se liberan los *P* procesadores para ser utilizados en la etapa 1 de la imagen M_{j+1} (512x512 ventanas de 2x2 en este caso), se obtendrá una mejora en el Speedup global, sin un costo adicional importante.

Hablamos de una *etapa de procesamiento* y no de *un* procesador adicional, porque según los tiempos efectivos de cálculo para cada grilla de P procesadores, debemos asegurar no retrasar el pipe. Es así que se muestran en la Tabla 6 los resultados del Tiempo paralelo promedio con las diferentes grillas de procesadores y se indica si la *etapa de procesamiento* requiere 1 o 3 procesadores reales.

	n/2 x n/2 (+3 proc)	n/4 x n/4 (+3 proc)	n/8 x n/8 (+1 proc)	n/16 x n/16 (+1 proc)	log n x logn (+1 proc)	logn/2 x logn/2 (+1 proc)
TPprom	107992.16	1478512.8	5819679.66	24740140.18	948373857.4	1170824541

Tabla 6

La Tabla 7 compara el speedup obtenido con la arquitectura paralela original y el speedup promedio que resulta del query de 100 imágenes con la arquitectura modificada. Se puede apreciar una sensible mejoría, sobre todo en las grillas con mayor número de procesadores.

	n/2 x n/2	n/4 x n/4	n/8 x n/8	n/16 x n/16	log n x logn	logn/2 x logn/2
Sporiginal	22552	20595	10587	3965	99	80
Spprom	87506	63915	16238	3980	99	80

Tabla 7

La Tabla 8 muestra la eficiencia obtenida con la arquitectura paralela original y la eficiencia promedio que resulta del query de 100 imágenes con la arquitectura modificada. Se puede apreciar una sensible mejoría en las grillas con mayor número de procesadores, mientras se observa una leve disminución para las arquitecturas con menor cantidad de procesadores.

	n/2 x n/2	n/4 x n/4	n/8 x n/8	n/16 x n/16	log n x logn	logn/2 x logn/2
Ef.Original	0.086	0.314	0.646	0.968	0.99	0.987
Ef. Prom	0.333	0.975	0.991	0.971	0.98	0.975

Tabla 8

7. CONCLUSIONES y LINEAS DE TRABAJO FUTURO

Los resultados experimentales sobre distintas clases de imágenes confirman la efectividad de los modelos de similitud basados en firmas múltiples, así como la ley exponencial de crecimiento del tiempo de procesamiento con el tamaño de la imagen.

El estudio de posibles arquitecturas paralelas (y su algoritmo asociado) permitió establecer el speedup y eficiencia alcanzables. El empleo de un modelo multiprocesador tipo MIMD con procesadores homogéneos y memoria distribuida tiene como principal inconveniente el costo de comunicaciones (que no se ha analizado en este trabajo).

Un resultado importante de este trabajo es el análisis del speedup global, asociado con el query de una secuencia de imágenes y su mejora con una pequeña modificación en la arquitectura paralela original.

Actualmente se está trabajando en la migración del algoritmo a una arquitectura multiprocesador con memoria compartida distribuida (SGI Origin 2000). Esta solución debiera mejorar los tiempos de comunicación entre procesos.

8. REFERENCIAS

[1]González R., Woods R., "Digital Image Processing", Addison-Wesley, 1996.

[2]Guibas L.,Rogoff B., Tomasi C. "Fixed window image descriptors for image retrieval", Proceeding Series of Storage and Retrieval for Images and Video Databases III. pp. 352-362-187, 1995.

[3]Jahne Bernd "Digital Image Processing", Springer, 1997.

[4]Zhang T., Ramakrishnan R., Livny M., "BIRCH: An Efficient Data Clustering Method for Very Large Databases", Proceedings of the ACM SIGMOD Conference on Management of Data. pp. 103-114, Montreal, Canada, June 1996.

[5]Niblack W. "The qbic project: Query image by content using color, texture and shape", Storage and Retrieval for Image and Video Databases. pp. 173-187, San Jose 1993.

[6]Jacobs C., Finkelstein A, Salesin D. "Fast multiresolution image querying", Proceedings of SIGGRAPH annual Conference Series. pp. 277-286, 1995.

[7]Castro L., Castro S., "Wavelets y sus Aplicaciones", Proceedings of the I Argentine Congress of Computer Science, Bahia Blanca, Argentina 1995. pp. 195-204.

[8]Hussain Z., "Digital Image Processing" Ellis Horwood, 1991.

[9]Lanzarini,De Giusti, "Class based Clustering" IWISPA 2000, 1st. Int'1 Workshop on Image and Signal Processing and Analysis. IEEE CAS & ASSP Croatia., vol. 21, no. 10, pp.1163-1175, 2000.

[10]Smith J. "Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis", PhD Thesis, Graduate School of Arts and Sciences, Columbia University, Feb 1997.

[11] Kalid S., "Introduction to Data Compression" Morgan Kaufmann, 1996.

[12] Natsev A., Rastogi R., Shim K., "WALRUS: A Similarity Retrieval Algorithm for Image Databases", Proceedings of the ACM SIGMOD 1999 Philadelphia. pp. 395-405.

[13] Natsev A., Rastogi R., Shim K., "WALRUS: A Similarity Retrieval Algorithm for Image Databases", Technical report Bell.

[14] De Giusti L., Tarrío D. "Algoritmo de Análisis de Similitud de Imágenes", Graduate Thesine. Universidad de La Plata, Argentina. Dec 2000.

[15] De Giusti A, Naiouf M., De Giusti L "Experimental measures in image similarity analysis", LIDI Technical Report. Universidad de La Plata, Argentina. March 2001.

[16] www.setcip.gov.ar/config_clementina2.htm.

[17] De Giusti L, Naiouf Marcelo, De Giusti A. "Similarity analisys for image databases. A parallel solution". SCI2001. Florida E.E.U.U. Julio2001. pp 226-231.