Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

Tipo de artículo: Artículo original
Temática: Procesamiento de imágenes
Recibido: dd/mm/aa | Aceptado: dd/mm/aa | Publicado: dd/mm/aa

Componente de software para el reconocimiento de armas en imágenes de rayos X

Software component for weapons recognition in X-ray images

Felipe Rodriguez Arias [1*], Leaned Quintana Vivanco [1], Frank Sanabria Macías [1], David Castro Piñol [1], Enrique Juan Marañón Reyes [1]

[1] Centro de Estudios de Neurociencias, Procesamiento de Imágenes y Señales, Universidad de Oriente, Cuba. Ave. Las Américas s/n e/ L e I. Santiago de Cuba, directorcenpis@uo.edu.cu

[*] Autor para correspondencia: farias@uo.edu.cu

Resumen

Las imágenes de rayos X constituyen una importante herramienta para aplicaciones de seguridad, especialmente en sitios donde la exploración de bultos en busca de objetos peligrosos es de vital importancia. Según los especialistas la inspección de equipajes mediante esta técnica presenta limitantes en cuanto a la eficiencia en el reconocimiento de objetos peligrosos entre otros factores. Esto es influenciado -en alguna medida- por las características propias de las imágenes de rayos X, las cuales son muy diferentes a las imágenes del espectro visible, afectando su interpretación. Es por ello que, tanto los métodos de adquisición, como los algoritmos para la mejora, procesamiento o reconocimiento de objetos en estas imágenes, se caractericen por costosas operaciones de cómputo que repercuten en altos tiempos de ejecución. La reducción del tiempo de ejecución en aplicaciones de seguridad puede ser un objetivo de alta prioridad, en sistemas que soportan la toma de decisiones rápidas, entre otros. Además, las técnicas de programación paralela han demostrado ser una alternativa viable para la solución rápida de problemas de visión computacional. En este trabajo se presenta el desarrollo de un componente de software escrito en C/C++ que describe un procedimiento basado en el algoritmo BoVW para la detección de armas cortas. La implementación emplea técnicas de programación paralela mediante OpenMP, logrando reducir hasta un 93.36% el tiempo de ejecución de una anterior implementación en MATLAB. También se prueba el método de clasificación en una nueva base de datos de imágenes de rayos X de equipos de inspección de energía dual.

Palabras clave: imágenes de rayos X, Saco de Palabras Visuales, computación paralela.

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

Abstract

The x-ray images constitute an important tool for security applications, especially on sites exploration of packages in for searching dangerous objects is very important. According to experts, the inspection of baggage by using the exploration of package techniques present limitations for the terms of efficiency in recognition of dangerous objects among other factors. This is influenced -in some aspects- by the characteristics of the X-ray images, which are very different to the visible spectrum images, affecting the performance of the techniques of exploration packages. For this reason, the acquisition methods, as well as for improving, processing or recognition of objects in these images, are characterize with expensive operations producing high runtimes. The reduction of the runtimes of the security applications can be a target of high priority for systems supported by fast decisions, among others. In addition, parallel programming techniques have proven to be a viable alternative for solving of problems in computer vision. In this paper, we propose a software component written in C/C++ that describes a procedure based on the BoVW algorithm handguns detection. For implementing the proposal, we use techniques of parallel programming with OpenMP, decreasing in 93.36% from a previous implementation of the proposal in MATLAB. The classification method is also tested in a new database of X-ray images, which were obtained by using dual-energy inspection equipment.

Keywords: *X-ray images, Bag of Visual Words, parallel computing.*

Introducción

En la actualidad, existe una alta especialización de los técnicos radiólogos en la utilización de equipos de rayos X de última generación. En contraste con esto, subsisten dificultades en la inspección de equipajes mediante esta técnica, por lo que cada vez más es deseable la implementación de herramientas informáticas que reduzcan la carga de trabajo de los especialistas (Bastan, Yousefi, & Breuel, 2011). Entre otros aspectos, resulta necesario una mejoría en la eficiencia del reconocimiento de objetos peligrosos y el aumento de la rapidez en el procedimiento de requisa, siempre teniendo como premisa que no se desea eliminar del todo la labor consiente del hombre sino humanizar, fortalecer y complementar su tarea (Mery, 2013).

En los últimos años, se han llevado a cabo investigaciones científicas que viabilizan la detección de objetos peligrosos en imágenes de rayos X. Los estudios de (Mery, Mondragon, Riffo, & Zuccar, 2013) y (Mery & Riffo, 2013) dan a conocer un método combinatorio que fusiona la estimación de estructuras internas y la detección de las partes de objetos prohibidos en un modelo geométrico utilizando equipos de múltiples vistas de energía simple con muy buenos resultados. Otro de los métodos empleados en el análisis de imágenes de rayos X es el algoritmo Saco de Palabras

Visuales o *Bag of Visual Words* (BoVW), propuesto en sus inicios por (Csurka, Dance, Fan, Willamowski, & Bray, 2004) para la búsqueda de imágenes por contenido. Más tarde, (Bastan, Yousefi, & Breuel, 2011; Bastan, Byeon, & M, 2013) y (Turcsany, Mouton, & Breckon, 2013) extienden la utilización de esta técnica a las imágenes de rayos X de sistemas de inspección de energía dual, de transmisión de múltiples vistas y de energía simple respectivamente, para el reconocimiento de materiales ilícitos. Otra contribución en esta área del conocimiento la realizó (Castro, Sanabria, Marañón, & Rodriguez, 2016) con la colaboración de especialistas de la Aduana General de la República (AGR) de Cuba. En esta última, se utilizan imágenes de equipos de rayos X de energía dual con el propósito de detectar armas cortas; además, se lleva a cabo un estudio de la influencia de diferentes *kernels*, funciones de pérdida y vocabularios en la clasificación de las muestras empleando el método BoVW.

La solución propuesta por (Castro, 2016) mostró ser eficiente respecto a la clasificación de armas cortas, sin embargo, la implementación llevada a cabo fue realizada sobre la plataforma MATLAB, lo cual dificulta la ejecución del método fuera de ese entorno y repercute en un pobre acoplamiento con otros sistemas en diferentes lenguajes de programación. Por otra parte, en diversas pruebas de rendimiento, para observar la velocidad de ejecución de esa propuesta, se pudo constatar que como promedio se empleaban 63 segundos para el análisis y clasificación de una nueva imagen de rayos X. Esta evaluación fue verificada en una PC Intel(R) Core(TM) i3 con 4 núcleos de procesamiento a 2.20 GHz y 4 GB de memoria RAM. Este tiempo de ejecución resulta insuficiente considerado que un método de detección de armas de este tipo pudiera ser empleado como herramienta de ayuda por los especialistas radiólogos justo en el proceso de inspección de equipajes (Bastan, Yousefi, & Breuel, 2011).

El presente trabajo tiene como objetivo la implementación de un componente de software en el lenguaje de programación C/C++, que sigue el procedimiento basado en el algoritmo BoVW descrito por (Castro, 2016). Además, se ha hecho especial énfasis en la reducción del tiempo de ejecución mediante técnicas de programación multiproceso. Estas permiten la utilización al máximo de las potencialidades de los procesadores con múltiples núcleos integrados en los equipos de cómputo de uso cotidiano hoy en día. También, se prueba la eficiencia en la clasificación de armas cortas de la nueva implementación sobre una base de datos de imágenes de rayos X de equipos de inspección de energía dual de amplia utilización en Cuba.

Método Saco de Palabras Visuales

El método Saco de Palabras Visuales está constituido por tres fases (figura 1), en su desarrollo se utilizan tanto algoritmos de clasificación supervisada como no supervisada. La primera fase se basa en la construcción de un vocabulario de palabras visuales empleando como fuente una base de datos de imágenes. Para ello, se extraen rasgos

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

visuales de todas las imágenes con algún método de extracción de características. Luego se utiliza un algoritmo de agrupamiento, usualmente *k-means*, que crea grupos de rasgos visuales similares entre sí. Los centros de cada grupo son llamados palabras visuales (Csurka, 2004; Tsai, 2012; Turcsany, 2013).

La segunda etapa tiene como objetivo el entrenamiento de un clasificador binario a través de histogramas de palabras visuales, el más utilizado en este caso es la Maquina de Soporte Vectorial o *Support Vector Machine* (SVM), (Vapnik & Vapnik, 1998). Los histogramas se construyen mediante una cuantificación vectorial de los rasgos extraídos de la imagen con el vocabulario de palabras visuales (Bastan, Yousefi, & Breuel, 2011; Castro, 2016).

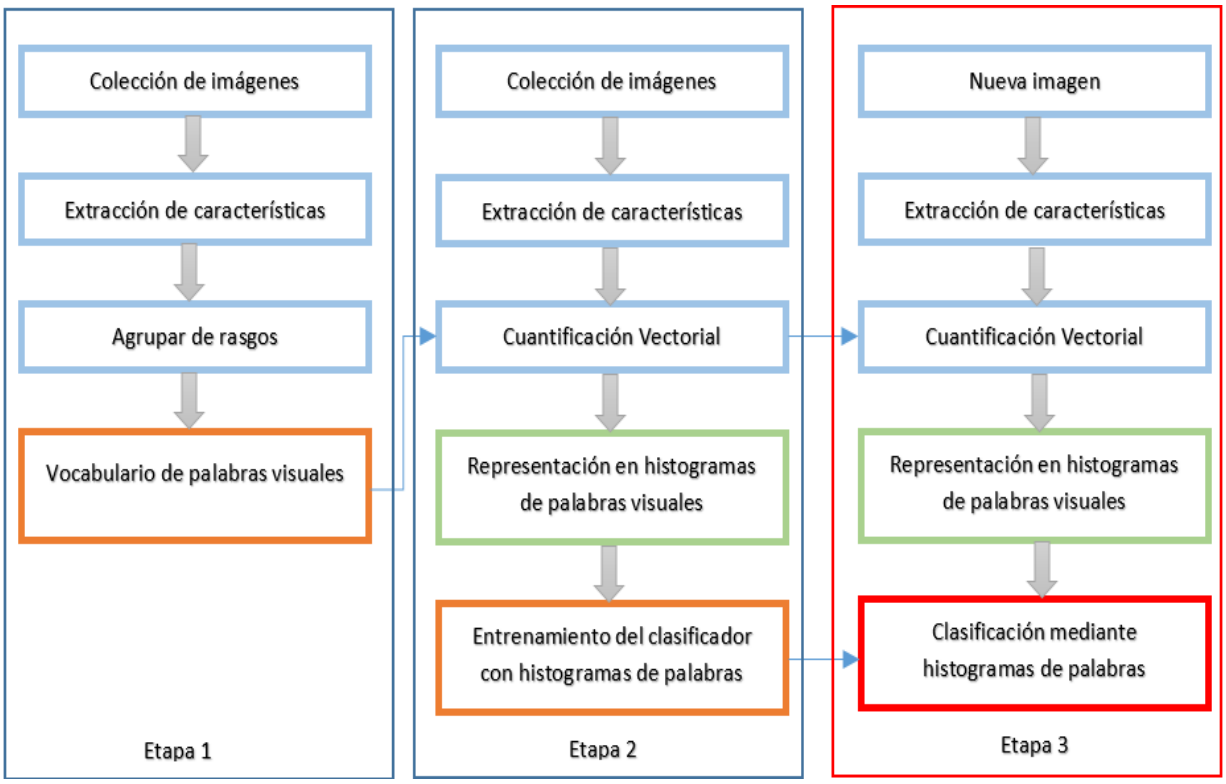

Figura 0. Esquema de clasificación basado en una representación BoVW

La tercera fase utiliza los datos de las dos etapas anteriores, el vocabulario de palabras visuales y el entrenamiento del clasificador basado en los histogramas del universo de imágenes. Es el proceso mediante el cual se logra la clasificación de una nueva imagen, no contenida en el proceso de entrenamiento. Para ello la nueva imagen es transformada en un histograma de palabras visuales, y comparada en el clasificador SVM quien determina si el histograma pertenece o no a la clase objetivo.

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

Materiales y métodos

El componente de software desarrollado implementa los procedimientos de la etapa 3 del esquema de clasificación de la figura 1, basado en la propuesta de (Castro, 2016). Para ello, se empleó el lenguaje de programación C/C++, con la ayuda de las bibliotecas de funciones VLFeat 0.9.18 (Vedaldi & Fulkerson, 2010) y OpenCV. La librería VLFeat permitió simplificar el cálculo de los descriptores de características de la imagen, la creación de los histogramas de palabras visuales y el proceso de clasificación mediante la SVM. En la ejecución se utiliza una ventana deslizante, (Jones & Viola, 2001) para la detección de los objetos, que evalúa el clasificador a lo ancho y largo de la imagen.

Con el propósito de mejorar los tiempos de ejecución de esta nueva implementación se explotaron las potencialidades que brinda la interfaz OpenMP para la programación multiproceso de memoria compartida (Chapman, Jost, & Der, 2008). La biblioteca OpenMP se compone de un conjunto de directivas de compilador que permiten especificar qué regiones de código van a ser paralelizadas, así como una biblioteca de rutinas en tiempo de ejecución y un conjunto de variables de entorno. Se basa en el modelo *fork-join* para obtener el paralelismo a través de múltiples hilos. Este modelo plantea la división del hilo maestro en hilos esclavos que se ejecutan concurrentemente, distribuyéndose las tareas sobre estos hilos. Estos hilos acceden a la misma memoria, aunque es posible gestionar estos accesos generando espacios de memoria privada (Chapman, Jost, & Der, 2008).

Descripción de los datos del entrenamiento del clasificador SVM

El resultado final del proceso de entrenamiento, ejecutado en las dos primeras etapas del método BoVW está determinado por dos matrices de datos numéricos, que representan el vocabulario de palabras visuales y el entrenamiento del clasificador SVM. Para su obtención se siguieron las pautas descritas por (Castro, 2016) para la clasificación de armas cortas en imágenes de rayos X. Esta información es utilizada por el componente de software en el proceso de clasificación de un histograma de palabras visuales, con el propósito de conocer si los descriptores de una ventana deslizante pertenecen o no al tipo de objeto que se busca, en este caso armas cortas.

El vocabulario está conformado por una matriz de 1000 filas por 384 columnas que caracterizan los 1000 representantes obtenidos por el agrupamiento de características mediante el algoritmo *kmeans* (cada uno representa una palabra visual). Los valores de las columnas identifican los coeficientes calculados mediante el algoritmo PHOW (Bosch, Zisserman, & Munoz, 2007) para cada representante en cada uno de los canales HSV de la imagen (128 x 3).

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

El clasificador SVM está representado por un vector de pesos de 5000 elementos, ya que la utilización del *kernel* homogéneo aditivo X^2 elevó en un factor de 5 la dimensión de los datos de entrada (1000 palabras visuales). Además, se obtiene el valor del *bias* que representa el término independiente de la ecuación SVM (Castro, 2016).

Rendimiento de computadores paralelos

Una forma de medir la calidad de un sistema paralelizado consiste en comparar la velocidad conseguida en el sistema paralelo (con N procesadores) con la velocidad conseguida con un solo procesador (Berstein, 1966). Para ello se define la ganancia de velocidad o aceleración de un sistema de N procesadores como:

$$S(N) = t(1)/t(N) \quad [1.1]$$

Donde $t(1)$ es el tiempo empleado para ejecutar el proceso si se utiliza un solo procesador y $t(N)$ es el tiempo empleado para ejecutarlo en el sistema paralelo con N procesadores. En condiciones ideales, $t(N) = 1/N$ con lo que, en esas mismas condiciones, la ganancia de velocidad dada por la ecuación 1.1 será

$$S(N)_{ideal} = \frac{t(1)}{t(N)} = \frac{1}{(1/N)} = N \quad [1.2]$$

Una forma de medir el rendimiento de un sistema resulta de comparar la ganancia de velocidad del mismo con la ganancia de velocidad ideal dada por 1.2, a esta medida se le denomina eficiencia:

$$E(N) = \frac{S(N)}{S(N)_{ideal}} = \frac{S(N)}{N} = \frac{\frac{t(1)}{t(N)}}{N} = \frac{t(1)}{Nt(N)} \quad [1.3]$$

La ley de Amdahl (Amdahl, 1967) pone un límite superior a la ganancia en velocidad, y por tanto también a la eficiencia, de un sistema paralelo atendiendo al hecho, de que los procesos suelen tener partes que no pueden ser ejecutadas en paralelo, sino solo de forma secuencial pura. De esta forma puede verse que:

$$S(N)_{max} = \frac{N}{1+(N-1)*f} \quad [1.4]$$

$$E(N)_{max} = \frac{1}{1+(N-1)*f} \quad [1.5]$$

Llamando f a la fracción de tiempo que no se puede paralelizar sobre el total del tiempo de ejecución.

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

Resultados y discusión

En esta sección se describen los resultados obtenidos en distintos experimentos sobre el componente de software desarrollado. Primeramente, se examina el proceso de clasificación, donde se utiliza una base de datos de imágenes de equipos de rayos X de energía dual. Luego, se presenta el diagrama de bloques de la implementación creada, así como la dependencia de datos existente entre cada uno de los componentes que facilita la ejecución en paralelo del procedimiento. Se realiza un estudio del comportamiento en distintos procesadores y entornos de hardware y se calcula la medida de la mejora obtenida en cada arquitectura de hardware mediante la Ley de Amdahl.

Evaluación de la clasificación

La base de datos de prueba está compuesta por 172 imágenes de rayos X que pertenecen a 8 modelos de equipos de inspección NUCTECH, de amplia utilización en Cuba. Se caracterizan por contener al menos un arma corta y presentar un tamaño promedio de 573x515 píxeles. Las armas en las imágenes de prueba aparecen en diversas condiciones con diferentes niveles de complejidad para la detección, que fueron catalogadas en los siguientes grupos: 1: objetos con oclusión propia, 2: solapadas con objetos metálicos, 3: partes no metálicas, 4: distorsión geométrica de la adquisición, 5: parcialmente desarmada y 6: visible frontal simple (figura 2). Existen también armas que contienen una combinación de las situaciones que se describen.

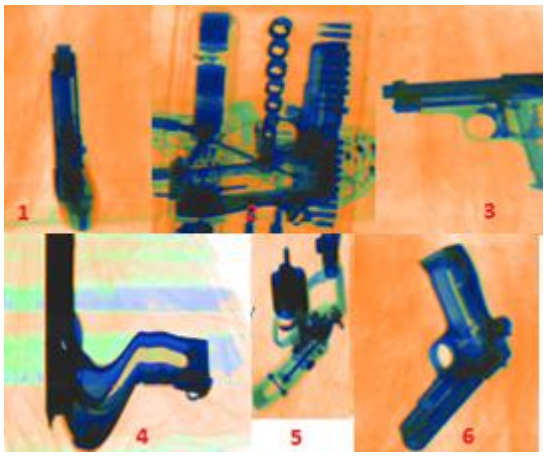

Figura 2: *Distintas situaciones de las armas*

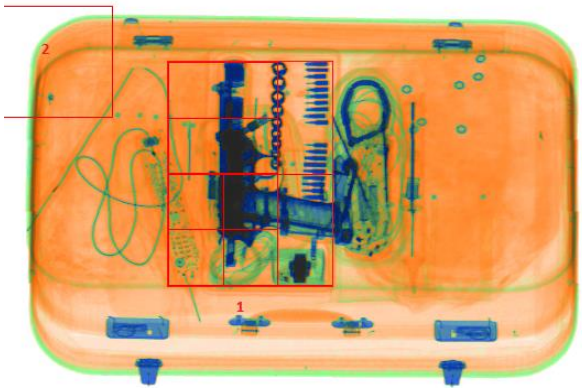

Figura 3: *Escena obtenida en una inspección.*
1: Múltiple detecciones sobre un objeto, 2: Detección falsa positiva

Se comprobó que como promedio el componente de software ejecuta 116 ventanas deslizantes por imagen evaluando el clasificador a lo largo y ancho de esta. Cada ventana deslizante tiene un tamaño de 100x100 píxeles y se mueve con

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

un paso de 50 píxeles. De las 172 imágenes evaluadas se etiquetaron correctamente 160, para un 93.02% de reconocimiento de armas cortas como se observa en la tabla 1. Del total de ventanas evaluadas (19952), calculadas a partir de multiplicar la cantidad de imágenes de la base de datos por el promedio de ventanas deslizantes evaluadas en cada imagen; son marcadas como ventanas falsas positivas 149, lo cual representa el 0.74% (figura 3). Los resultados obtenidos confirman la viabilidad del método propuesto por (Castro, 2016) sobre una nueva base de datos de imágenes de rayos X de equipos de inspección de energía dual, marca NUCTECH, utilizados por el sistema de aduanas cubano.

Tabla 1. Evaluación en el reconocimiento de armas cortas

Imágenes	Imágenes clasificadas	%	Ventanas evaluadas	Ventanas Falsas Positivas
172	160	93.02	19952	149

Análisis del desempeño

En la evaluación del desempeño se emplearon técnicas de programación paralela para determinar la ganancia de velocidad en el tiempo de ejecución del componente de software. Las pruebas se realizaron en una computadora personal Core i3 de 4 núcleos a 3.1GHz, 4GB de RAM con sistema operativo Windows7. Se determinó el tiempo de ejecución en las 19952 ventanas deslizantes de las 172 imágenes de la base de datos y se obtuvo el promedio del tiempo para una sola ventana.

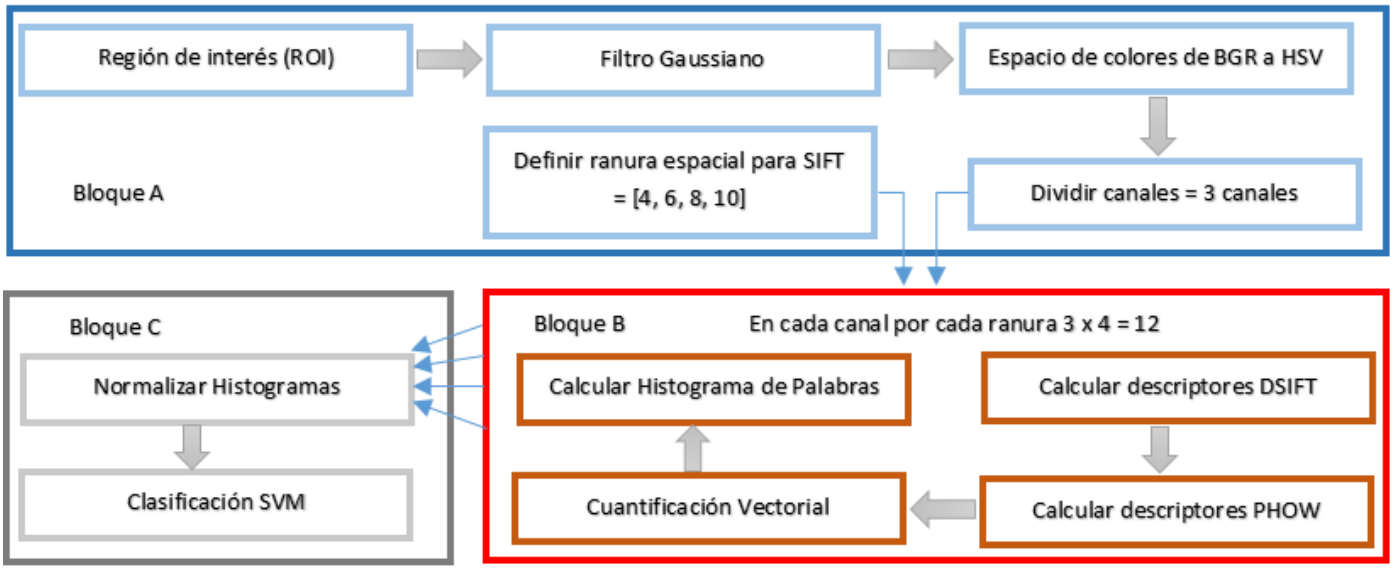

Figura 4. Diagrama de bloques ampliado de los procedimientos implementados en el componente de software

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

La figura 4 muestra el diagrama de bloques de la implementación, realizada con algunos ajustes para permitir la división del problema en partes independientes para su análisis. El bloque A corresponde a las tareas de inicialización de variables, donde se preparan las estructuras de datos necesarios para ejecutar el bloque B. Después, en el bloque C, se construye el histograma de palabras visuales a partir de la normalización de los histogramas obtenidos en cada canal de la imagen por cada ranura espacial del DSIFT (Lowe, 2004) y se entrega al clasificador para su evaluación. El bloque B se encarga de la detección de la región de interés para el cálculo de los descriptores de la imagen, luego asocia cada descriptor obtenido con el representante más cercano del vocabulario de palabras visuales, finalmente construye el histograma de ocurrencias por cada palabra visual. La complejidad del bloque B es $O\ (C * D * W) = O(N^3)$ donde C es la cantidad de descriptores obtenidos, D es la dimensión del vector del descriptor y W es el número de palabras visuales del vocabulario. Teniendo en cuenta que el bloque B se ejecuta varias veces para una instancia de una ventana deslizante la complejidad resultante sería $O(M * N^3)$, donde M es el número de llamadas al procedimiento.

El diseño de los procedimientos internos del componente de software se basó principalmente en las etapas de partición (descomposición de la computación de tareas) y mapeo (asignación de tareas a los procesadores); descritas por (Foster, 1995) que proporcionan un esquema para la adecuación de algoritmos a un entorno paralelo. Se observa entonces que el bloque B efectúa un proceso repetitivo en el que no existen dependencias de datos entre una y otra iteración, lo que facilita su ajuste a un ambiente de ejecución paralelo. En la etapa de partición se empleó una descomposición del dominio aprovechando la independencia de los datos entre cada una de las iteraciones del bloque B. Luego, en la asignación, las tareas y datos asociados a cada iteración son asignados a un procesador tratando de maximizar la utilización de los procesadores y de reducir el costo de comunicación. Se escogió la asignación estática consiste en distribuir las tareas entre los procesadores al iniciar la ejecución del algoritmo esto debido a que la carga de procesamiento se encuentra en la iteración del bloque B y no en la cantidad de iteraciones a ejecutar que son relativamente pocas, sólo 12.

En la tabla 2 se muestra el comportamiento de los tiempos de ejecución en milisegundos de los diferentes bloques y se observa la ganancia de velocidad cuando se aumenta el número de procesadores empleados en el procesamiento de los datos. Considerando la información de la tabla 2 se obtiene un promedio de ejecución para el análisis de una imagen en la versión secuencial de 12.08 segundos (0.1042 segundos multiplicado por 116 promedio de ventanas deslizantes). Esto representa 50.92 segundos menos que la versión de MATLAB, mejorando en un 80.82% la velocidad de ejecución. Si se realiza la misma reflexión teniendo en cuenta la prueba con 4 hilos de ejecución, se

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

obtiene una ejecución promedio de 4.18 segundos por imagen, lo cual supera los resultados de la prueba de ejecución secuencial y constituye un 93.36% de mejora respecto a la versión de MATLAB en términos de tiempo.

Tabla 2. Tiempos de ejecución (ms) de los bloques en la etapa 3

Algoritmo	Secuencial	2 hilos	3 hilos	4 hilos
Bloque A	3,2	3,2	3,2	3,2
Bloque B	99,2	52,34	40,28	31
Bloque C	1,8	1,8	1,8	1,8
Total	104,2	57,34	45,28	36

Realizando un análisis mediante la ley de Amdahl en la tabla 2 se evidencia que cuando se efectúa la versión secuencial la fracción de tiempo de los bloques A y C (no paralelos) representan juntos el 5% del tiempo total y se mantienen inalterables en las otras versiones. La ley de Amdahl establece una cota superior en la ganancia máxima de velocidad cuando $f = 0.05$ y $N = 4$, que es $S(4)_{max} = \frac{4}{1+(4-1)*0.05} = 3.47$ y una eficiencia máxima $E(4)_{max} = \frac{1}{1+(4-1)*0.05} = 0.86$; luego se comparan estos valores con la ganancia y la eficiencia obtenida en la mejor de las pruebas. Empleando los 4 núcleos del procesador se logra una ganancia de velocidad $S(4) = \frac{t(1)}{t(4)} = \frac{99.2}{31} = 3.2$, con una eficiencia $E(4) = \frac{t(1)}{4t(4)} = \frac{99.2}{4*31} = 0.8$. La diferencia entre los valores máximos ideales y los obtenidos es pequeña, se encuentran en el orden de los 0.27 y 0.06 respectivamente, lo que indica que la implementación realizada aprovecha con efectividad los recursos de multiprocesamiento del procesador.

Luego de observar el comportamiento en un procesador *Core i3*, se realizaron pruebas en distintas configuraciones de *hardware* con el propósito de observar la ganancia de velocidad y la eficiencia de los procesadores en la ejecución de la sección paralela del método implementado (figuras 5, 6, 7). El procedimiento `omp_set_num_threads()` de OpenMP permite establecer un número mayor de hilos de ejecución que la cantidad de núcleos de procesamiento incluidos en un procesador, no obstante el algoritmo gana en velocidad hasta el número real de núcleos del procesador. De esta forma se observa una mejora en la aceleración hasta el procesador 2 en las configuraciones *Pentium 4, AMD A4 3305, Core2 Duo*, los cuales tienen 2 núcleos de procesamiento. En el caso del *Core2 Quad* y el *Core i3* se observan ganancias hasta los 4 hilos de ejecución. La figura 8 muestra en perspectiva los tiempos de ejecución de cada uno de los bloques en los diferentes procesadores.

Si bien las ganancias en velocidad y la eficiencia en las diferentes arquitecturas de hardware dependen de otros factores como la velocidad del disco duro, la memoria RAM, la velocidad y caché del CPU, el coprocesador

Revista Cubana de Ciencias Informáticas
Vol. 11, No. 2, Abril-Junio, 2017
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 162- 174

matemático o el sistema operativo empleado, los resultados alcanzados ofrecen una idea del comportamiento del algoritmo de forma general.

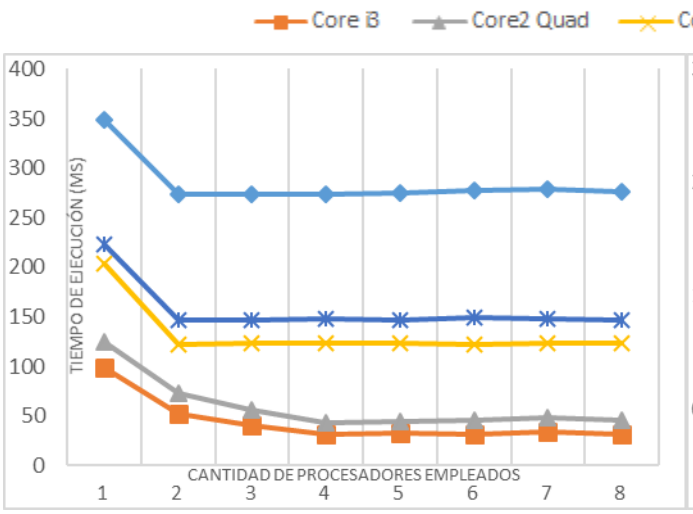

Figura 5. Tiempo de ejecución

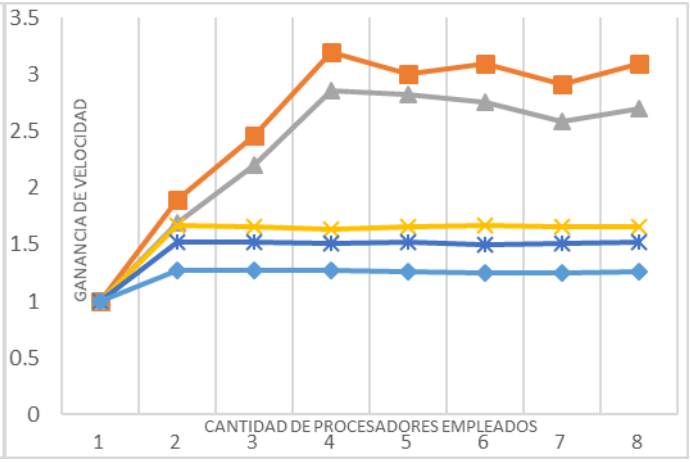

Figura 6. Ganancia de velocidad (Aceleración)

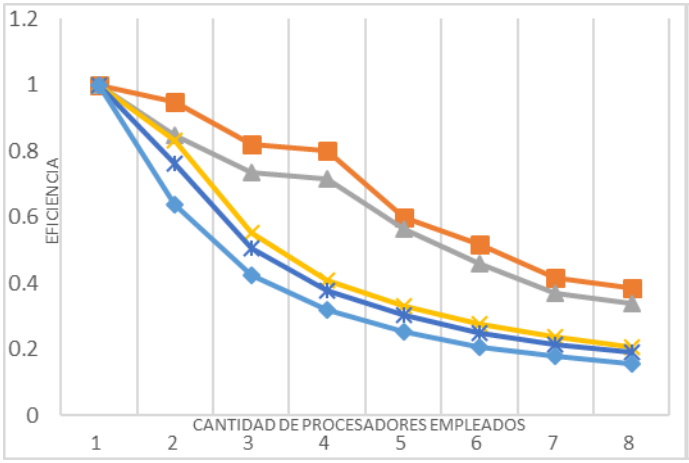

Figura 7. Eficiencia de los procesadores

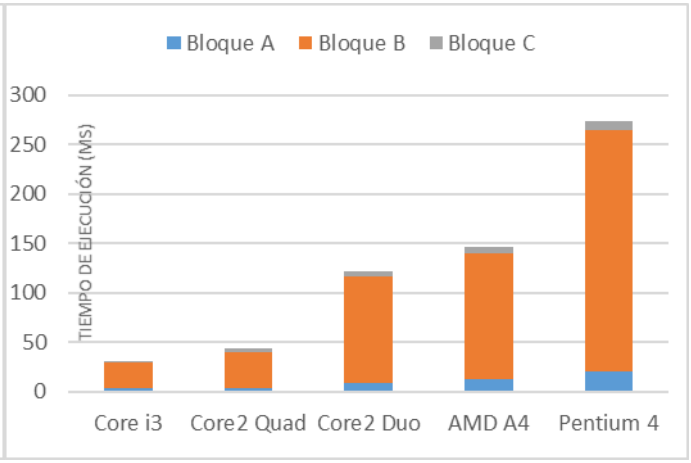

Figura 8. Tiempo de los bloques de ejecución

Conclusiones

- Se mostró el desarrollo de la implementación de un componente de software en C/C++ que sigue un procedimiento basado en BoVW para el reconocimiento de armas cortas en imágenes de rayos X.

- La adaptación del algoritmo original, escrito en MATLAB, a un entorno de ejecución paralelo con la utilización de las características de procesadores con múltiples núcleos, permite la obtención de mejores rendimientos mientras mayores sean las capacidades de procesamiento paralelo de los equipos.
- El carácter independiente de los datos en la construcción del histograma de palabras visuales favorece la utilización del paralelismo a nivel de datos como una alternativa eficiente para disminuir los tiempos de respuesta asociados.
- Las técnicas de programación paralela en memoria compartida facilitan la explotación del paralelismo de datos a través del uso de bucles iterativos para la distribución de tareas a los procesadores, propiciando un mejor aprovechamiento de las capacidades de cómputo.
- El componente de software logró la disminución del tiempo de ejecución en un 93.36% respecto a una implementación anterior, mediante el uso de la programación paralela en procesadores multi-núcleo.
- Se probó la implementación del método en una nueva base de datos de 172 imágenes de rayos X de energía dual, obtenida de equipos de inspección NUCTECH de amplia utilización en Cuba. Se alcanzó una clasificación satisfactoria del 93.02% en imágenes que contenían armas de fuego.

Referencias

AMDAHL, G. (1967). Validity of the Single Processor Aproach to Achieving Large Scale Computing Capabilities.

BASTAN, M., BYEON, W., & M, B. T. (2013). Object Recognition in Multi-View Dual Energy X-ray Images.

BASTAN, M., YOUSEFI, M. R., & BREUEL, T. M. (2011). Visual words on baggage X-ray images. *Springer*, (pp. 360-368).

BERSTEIN, A. (1966). Analysis of programs for parallel procesing. *IEEE Transactions on Computers, Oct.*, 15.

BOSCH, A., ZISSERMAN, A., & MUNOZ, X. (2007). Image classification using random forests and ferns.

CASTRO, D., SANABRIA, F., MARAÑÓN, E., & RODRIGUEZ, F. (2016). Reconocimiento de armas en imágenes de rayos X mediante Saco de Palabras Visuales. *Revista Cubana de Ciencias Informáticas, 1*, 161.

CHAPMAN, B., JOST, G., & DER, R. V. (2008). *Using OpenMP*. MIT.

CSURKA, G., DANCE, C., FAN, L., Willamowski, J., & Bray, C. (2004). Visual categorization with bags of keypoints., *1*, pp. 1-2.

FOSTER, I. (1995). Designing and Building Parallel Programs. *Chicago, Addison Wesley*, 430.

JONES, M., & VIOLA, P. (2001). Robust real-time object detection.

LOWE, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*(2), 91-110.

MERY, D. (2013). X-ray Testing: The State of the Art. *The e-Journal of Nondestructive Testing \& Ultrasonics (www. ndt. net)*, *18*(9).

MERY, D., & RIFFO, V. (2013). Automated Object Recognition in Baggage Screening Using Multiple X-ray Views.

MERY, D., MONDRAGON, G., RIFFO, V., & ZUCCAR, I. (2013). Detection of regular objects in baggage using multiple X-ray views. *Insight-Non-Destructive Testing and Condition Monitoring*, *55*(1), 16-20.

TSAI, C.-F. (2012). Bag-of-Words Representation in Image Annotation: A Review., *2012*, p. 19.

TURCSANY, D., MOUTON, A., & BRECKON, T. P. (2013). Improving feature-based object recognition for X-ray baggage security screening using primed visualwords. *IEEE*, (pp. 1140-1145).

VAPNIK, V. N., & VAPNIK, V. (1998). *Statistical learning theory* (Vol. 1). Wiley New York.

VEDALDI, A., & FULKERSON, B. (2010). VLFeat: An open and portable library of computer vision algorithms. *ACM*, (pp. 1469-1472).