Encriptación de imágenes mediante transformada de Arnold

Calamari, Santiago - Cipolatti, Edgardo
santi.calamari@gmail.com - edgardocipolatti@hotmail.com

***Resumen*—En este trabajo se describen y comparan la velocidad de procesamiento de imágenes para procesadores de un núcleo y para procesadores multinúcleos. Para poder realizar las correspondientes comparaciones se encriptaron y desencriptaron imágenes utilizando la transformada de Arnold.**

***Palabras clave*— single-core, multi-core, encriptar, desencriptar, Transformación de Arnold, período.**

I. INTRODUCCIÓN

El procesamiento paralelo consiste en un grupo de técnicas utilizadas para proporcionar tareas simultáneas de procesamiento de datos cuya finalidad consiste en aumentar la velocidad computacional de un sistema de computadoras. Se basa en un procesamiento concurrente de datos para conseguir un menor tiempo de ejecución. El procesamiento paralelo implica sucesos paralelos, es decir, sucesos que ocurren en múltiples recursos durante el mismo intervalo de tiempo; sucesos simultáneos, es decir que ocurren en el mismo instante y sucesos pipeline que ocurren en lapsos superpuestos.

El procesamiento paralelo ofrece una gran ventaja en cuanto a costos computacionales, permitiendo ejecutar procesos en donde cada procesador se encarga de diferentes tareas acelerando el tiempo y la forma de cálculo.

Por otro lado, el procesamiento en GPU o unidad de procesamiento gráfico, un procesador especializado que ejecuta rápidamente comandos para manipular y mostrar imágenes. La computación acelerada por GPU le ofrece un rendimiento más rápido a través de una amplia gama de aplicaciones de diseño, animación y vídeo.

Para poder realizar comparaciones entre estas formas de procesamiento, se utilizó la transformada de Arnold, que permite encriptar y desencriptar imágenes en el dominio espacial.

En el presente trabajo se realizaron diferentes casos de prueba, encriptando y desencriptando imágenes y videos con el fin de comparar la velocidad de procesamiento para single-core y multi-core.

II. MATERIALES Y MÉTODOS

En esta sección se describirán las formas de trabajo de cada tipo de procesamiento, como así también la manera en que la transformada de Arnold encripta y desencripta las imágenes.

El procesamiento single-core, consiste en procesar las imágenes en cuestión utilizando solamente un núcleo del procesador. En definitiva, cada instrucción o tarea se realiza de forma secuencial, esperando que una termine para que comience la siguiente. En la sección Casos de prueba, se describe la manera en que las imágenes fueron procesadas, tanto para single-core como multi-core.

El procesamiento multi-core o multinúcleo permite llevar a cabo distintos procesos en paralelos. Un proceso paralelo es aquel que se realiza al mismo tiempo que otro, siendo ejecutados ambos de modo simultáneo.

El proceso en paralelo tiene muchas ventajas sobre el sistema secuencial clásico. Permite salvar el conocido cuello de botella, algo que sucede con frecuencia en procesos secuenciales: si un proceso funciona lentamente, los demás deben esperar a que se termine para ejecutarse, con lo cual el rendimiento del ordenador se verá afectado en gran medida. Utilizando un sistema en paralelo, aunque un proceso sea lento, el resto continúan ejecutándose, lo que permite evitar este efecto. Sin embargo, el proceso en paralelo tiene un gran inconveniente: es mucho más complejo, básicamente porque para que un sistema trabaje en paralelo debes indicarle a donde derivar cada proceso, y esta tarea requiere ser hecha de antemano, o al menos tomar decisiones en el momento previo a ejecutar los procesos.

Para impedir lo último dicho, se realizaron diferentes casos de prueba con distintas imágenes, procesando las mismas de manera diferente contemplando varios enfoques para comparar la velocidad de procesamiento. Para más información ver la sección de casos de prueba.

Para el proceso de encriptación y desencriptación de imágenes se utilizó la transformada de Arnold. Dicha transformada trabaja en el dominio espacial, mezclando la posición de los pixeles. Dicha transformada se encuentra definida por:

$$A^M : \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \mathrm{mod}\left(\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}, M \right),$$

donde (x,y) e (u,y) son las coordenadas de la posición de un pixel de la imagen antes y después de la transformada. El operador mod es el módulo resto de la división entera. El periodo p de la transformada depende del tamaño M de la imagen. La imagen entera es codificada por el número iterativo n y la imagen puede ser recuperada por la inversa de la transformación de Arnold dada por k =p-n donde n y k son las claves para decodificar la imagen. Es crítico para imágenes en color mantener el color del píxel cuando realizamos las transformaciones de posición. La única restricción presente en la transformada de Arnold es determinar el periodo p para una imagen dada. Dicho periodo no es lineal, y está determinado por:

$$\mathrm{Period} = \min\{n | \{\mathrm{ART}[O(x,y), N]\}^n = O(x,y)\}$$

donde min denota el mínimo valor, n es un positivo entero que denota el número de iteraciones. De esta forma, si N es mayor que dos, el periodo usualmente satisface:

$$Periodo \leq \frac{N2}{15}$$

Por otra parte, como propuesta adicional, se logró desarrollar un algoritmo que expande el alcance de la transformada de Arnold. Más precisamente, se logró encriptar y desencriptar imágenes cuya estructura sea rectangular recordando que la transformada de Arnold solo trabaja con imágenes cuadradas. Este desarrollo radica en dividir la imagen rectangular en cuadrados más pequeños, con cierto lado determinado por el usuario para luego aplicarle la transformada de Arnold a cada uno.

Este algoritmo solo contempla cuadrados de tamaño cuyos lados son: 128, 120, 100, 80, 60, 32, 25, 16, 4.

En la figura 3 y 4 puede visualizarse el trabajo de dicho algoritmo.

III. CASOS DE PRUEBA

En esta sección se describen los casos de prueba generados tanto para videos como para imágenes.

Para el caso de pruebas de imágenes, se utilizó una base de datos llamada nirscene1, que contiene 477 imágenes con una resolución de 1024 por 768 pixeles. Para poder procesar las mismas, se recortaron con dimensiones cuadradas de 500 por 500 pixeles.

El primer caso de prueba (CS1), consistió en cargar todas las imágenes de la base de datos en un vector, y luego a través de un procesamiento single-core, encriptar y desencriptar cada imagen tomando como referencia el tiempo computacional generado. Lo mismo se realizó con un procesamiento multicore, donde en este caso, se asignó a

Figura 2: Imagen original, encriptada y desencriptada

cada hilo una imagen de la base de datos. De esta manera cada hilo, encriptaba y desencriptaba una imagen diferente.

El segundo y tercer caso de prueba con imágenes realizaba el mismo procedimiento que el caso de prueba uno, solo que para el caso de prueba dos (CS2), se duplicaron las imágenes de la base de datos, y para el caso de prueba tres (CS3) se triplicaron las imágenes. Con esto se buscaba que ambos casos de prueba tuvieran más imágenes para procesar, y en caso de que los tiempos computacionales sean diferentes, poder visualizar los mismos y sacar conclusiones futuras.

Para poder obtener un número mayor de variables para comparar resultados, se implementaron procesamientos single-core y multi-core encriptando y desencriptando videos.

De esta manera, el caso de prueba cuatro (CS4) consistió en procesar cada frame de un video .MP4. Para realizar procedimientos más complejos, se capturó cada frame del video y se lo dividió en cuatro sub-imágenes del mismo tamaño. Para procesamiento single-core se encripto y desencripto cada sub-imagen de forma secuencial y para procesamiento multi-core, en cada hilo se utilizó la transformada de Arnold en cada sub-imagen.

Por último el caso de prueba cinco (CS5), consistió en utilizar cuatro vídeos, dos videos cuya resolución es 480p y dos videos cuya resolución es de 720p. Esta prueba se basa en procesar diferentes videos a la vez, es decir, encriptar y desencriptar una lista de videos. Por lo tanto, para procesamiento single-core se procesaron los videos en forma de cascada, y para procesamiento multi-core se procesó en cada hilo un video.

IV. RESULTADOS

En la figura 1 se puede observarse los tiempos de ejecución para single-core y multi-core para los tres primeros casos de prueba.

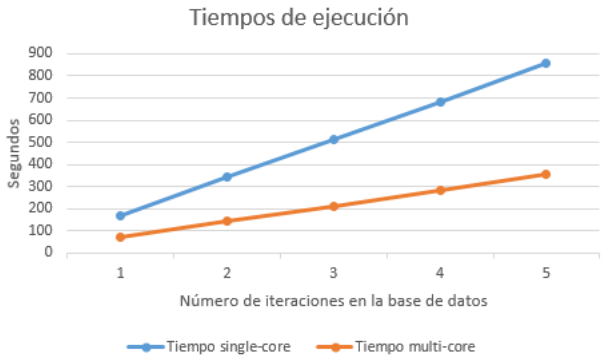

Figura 1: Comparación de tiempos de ejecución

En la figura 2 se observa a la izquierda la imagen original, en el medio la imagen encriptada y a la derecha la imagen desencriptada utilizando la transformada de Arnold.

Para el caso de prueba cuatro, no se obtuvieron buenos resultados, ya que el procesamiento multi-core devolvió un tiempo de ejecución de 205 segundos o 3 minutos y 25 segundos mientras que el procesamiento single-core obtuvo un tiempo de ejecución igual 100 segundos o 1 minuto y 50 segundos.

Para el caso de prueba cinco, se obtuvieron buenos resultados, ya que el procesamiento multi-core permite realizar tareas al mismo tiempo, logrando que los tiempos de ejecución sean menores y que los recursos utilizados se aprovechen de mejor manera. El tiempo de ejecución para este caso de prueba es 734.731 segundos o 11 minutos y 58 segundos, mientras que en multicore tenemos 379 segundo o 6 minutos y 19 segundos.

Por último en las tres últimas figuras (3, 4 y 5), se visualiza la mejora del algoritmo de Arnold, trabajando con imágenes rectangulares.

Figura 3: Imagen original

Figura 4: Imagen encriptada por cuadrados

Figura 5: Imagen original y encriptada

V. CONCLUSIONES

Como conclusión podemos rescatar que la utilización de multicore es mejor en la mayoría de los casos. Como se observa en los primeros 3 casos de prueba el crecimiento de tiempos a medida que se incrementa la cantidad de imágenes es lineal. Por otro lado, cuando encriptamos videos obtuvimos diferentes resultados dependiendo de la forma de procesarlos como se explicó anteriormente.

Para el caso de la imágenes rectangulares, cuando se encripta una imagen muy grande con cuadrados muy chicos, se percibe un efecto de cuadriculado y aun así se puede visualizar o interpretar la imagen original, cosa que no pasa cuando los cuadrados de encriptación son mayores. Como trabajo futuro y forma de mejorar aún más la encriptación de imagen se podría trabajar con distintos tamaños de cuadrados para encriptar una imagen rectangular y así dificultar la desencriptación.

Por otro lado, como se mencionó en la introducción, la GPU ofrece grandes rendimientos en imágenes pero queda pendiente su implementación y testeo sobre la misma dado que no contamos con CUDA cores, siendo éste, el único ambiente soportado por OpenCV.

REFERENCIAS

[1] W. Chen, C. Quan, C.J. Tay, *Optical color image encryption based on Arnold transform and interference method*, 2009.

[2] Min Li. Ting Liang. Yu-jie He. Hansen, Arnold Transform Based Image Scrambling Method,

[3] A. Pant, A. Arora, S. Kumar, R P Arora, Sophisticated Image Encryption Using OpenCV, 2012

[4] Cnyan Meng, Xinghui Zhang, Video Encryption Based on OpenCv, In: 2nd IEEE International Workshop on Database Technology and Applications ,2010, pp .1-4