ASISTENTE DE CONDUCCIÓN NOCTURNA EN SISTEMA EMBEBIDO CPU-GPU, UTILIZANDO LA TRANSFORMADA WAVELET DISCRETA DE HAAR

NIGHT DRIVING ASSISTANT IN A SYSTEM EMBEDDED CPU-GPU, USING THE HAAR DISCRETE WAVELET TRANSFORM

M. Fernando Maldonado Ramírez
Universidad Autónoma Metropolitana, Azcapotzalco, México
manuelfernando.mrmz@gmail.com

Oscar Alvarado Nava
Universidad Autónoma Metropolitana, Azcapotzalco, México
oan@azc.uam.mx

Eduardo Rodríguez Martínez
Universidad Autónoma Metropolitana, Azcapotzalco, México
erm@azc.uam.mx

Andrés Ferreyra Ramírez
Universidad Autónoma Metropolitana, Azcapotzalco, México
fra@azc.uam.mx

Recepción: 5/noviembre/2019 **Aceptación:** 23/noviembre/2019

Resumen

Durante la noche el tráfico vehicular disminuye considerablemente, sin embargo, un número significativo de accidentes ocurren durante este lapso de tiempo, un factor importante a considerar es la disminución visibilidad debido a la mala iluminación. Se presenta el diseño y desarrollo de un sistema de asistencia para conducción nocturna en un sistema embebido. Mediante la Transformada Wavelet Discreta de Haar, se fusionó una imagen convencional con una imagen infrarroja, para así generar una imagen con más información visible para el conductor. Para que estas imágenes fusionadas sean realmente un asistente de conducción, deben ser generadas en el menor tiempo posible, por ello se realizó la paralelización del programa y se ejecutó en un sistema paralelo embebido CPU-GPU. En este sistema paralelo se realizaron diversos experimentos para generar imágenes fusionadas

con diferentes resoluciones, obteniendo aceleraciones en el tiempo de procesamiento de hasta 69x.

Palabras Claves: Fusión de imágenes, GPU, transformada Wavelet discreta, sistemas embebidos.

Abstract

At night, the vehicular traffic decreases considerably, however, a significant number of accidents occurs, one problem is the decreased visibility due to poor lighting. This paper presents the design and development of a night driving assistance system. Through the Haar Discrete Wavelet Transform, a conventional image was merged with an infrared image, in order to generate an image with more information visible to the driver. For these merged images to really be a driving assistant, they must be generated in the shortest possible time, so the program was parallelized and executed in a parallel CPU-GPU embedded system. In this parallel system, several experiments were carried out to generate fused images with different resolutions, obtaining accelerations in the processing time of up to 69x.

Keywords: *Image fusion, GPU, discrete Wavelet transform, embedded systems.*

1. Introducción

Conducir un automóvil durante la noche o en condiciones meteorológicas desfavorables puede ser de alto riesgo debido a la falta de iluminación tanto del camino como de señalamientos viales. En estas condiciones, la información recibida por el ojo humano y por cámaras de video convencionales está limitada a la iluminación provista por los faros del vehículo y por la luz artificial disponible. Con una mala iluminación, peatones y animales pueden ser difíciles de detectar, pero pueden ser fácilmente detectados con una cámara infrarroja, sin embargo, una imagen infrarroja no proporciona la información necesaria para conducir un vehículo por la noche, ya que los señalamientos viales como las marcas de carril o los pasos peatonales no son detectados por cámaras infrarrojas.

En la figura 1 se muestran imágenes de la misma escena tomadas con una cámara infrarroja y con una cámara convencional respectivamente.

a) *Imagen Infrarroja* *b) Imagen convencional o visible*

Figura 1 Imágenes fuente.

La fusión de imágenes convencionales e infrarrojas se presenta como una solución para crear un asistente de conducción, ya que resalta la información de cada imagen y las mezcla para generar una nueva imagen que ayuda tanto a distinguir señalamientos viales como a detectar peatones o animales en la vialidad. Debido sus características de multiresolución la Transformada Wavelet Discreta de Haar o DHWT (*Discrete Haar Wavelet Transform*) se ha utilizado para la fusión de imágenes, en [Herrington, 2005] se presenta el desarrollo en de un asistente de conducción nocturna utilizando la DHWT. El sistema fue implementado en un sistema de cómputo de ejecución secuencial.

La fusión de imágenes convencionales con imágenes infrarrojas utilizando la Transformada Wavelet Discreta, se ha utilizado en otros campos y aplicaciones, por ejemplo, en [Pal, 2017] se presenta la fusión de imágenes para el reconocimiento facial.

Transformada Wavelet Discreta

La parte fundamental de la Transformada Wavelet Discreta (DWT, *Discrete Wavelet Transform*) es la descomposición multiresolución de señales, lo cual resulta muy ventajoso cuando se aplica a imágenes, ya que objetos pequeños necesitan resoluciones altas, mientras que los objetos grandes necesitan bajas resoluciones.

A su vez, el enfoque general de la descomposición multiresolución es el crear un componente de aproximación con una función de escalamiento (filtro pasa-bajas) y un componente de detalle con funciones wavelets (filtro pasa-altas).

La DWT permite convertir una señal que se encuentra en el dominio del tiempo, a una señal en el dominio de frecuencia y regresar la a su dominio original [Pal, 2017], [Kannan, 1997]. Con la DWT es posible localizar el momento en el que ocurre cierto evento en el dominio del tiempo, es decir, la correlación entre una señal dada y una señal base se puede determinar en varios instantes de tiempo [Aymaz, 1997]. Además, el espectro de la señal está dividida en partes desiguales, lo cual es muy adecuado para el análisis de la señal, ya que un ancho de banda amplio e intervalos de tiempo cortos son apropiados para detectar un componente de alta frecuencia, mientras que un ancho de banda corto e intervalos de tiempo largos son adecuados para localizar un componente de baja frecuencia [Sundararajan, 2015].

La DWT trabaja con señales discretas. Una señal discreta es una función en el tiempo con valores que ocurren en un instante de tiempo finito, como se como se muestra en la ecuación 1.

$$f = \left(f_1, f_2, f_{3,\dots}, f_N\right) \tag{1}$$

Donde N es un número entero par positivo el cual representa el tamaño o longitud de f y def_1 a $f_N$son los N números reales de la señal discreta f.

Transformada Wavelet Discreta de Haar

La Transformada Wavelet Discreta de Haar (DHWT, *Discrete Haar Wavelet Transform*) es la familia de Wavelet más simple y es la base para familias de Wavelets complejas. La DHWT descompone una señal discreta en dos subseñales de la mitad del tamaño que la señal original. La primera de estas es un promedio o tendencia, la segunda por su parte es una diferencia o fluctuación [Walker, 2008]. La sub-señal de tendencia $a = \left(a_1, a_2, a_{3,\dots}, a_{N/2}\right)$ de f está compuesta por un conjunto de promedios, los cuales son calculados de la siguiente manera: el primer valor, a_1 es calculado tomando el promedio del primer par de valores de f, y después multiplicando del siguiente par dicho promedio por$\sqrt{2}$. El valor a_2 es el

resultado de tomar el promedio de valores de f y de igual forma, multiplicando el resultado por $\sqrt{2}$. Así, los valores de a son obtenidos tomando el promedio de pares sucesivos de valores de f, y después multiplicando esos promedios por $\sqrt{2}$. La ecuación 2 se utiliza para obtener los valores de a.

$$a_m = \frac{f_{2m-1} + f_{2m}}{\sqrt{2}} \; para \; m = 1,2,3 \ldots N/2 \qquad (2)$$

En contraste, la subseñal de fluctuación de f, denotada por $d = (d_1, d_2, d_{3,\ldots}, d_{N/2})$, está formada por una serie de restas o diferencias [Nickolls, 2010]. El primer valor de la señal de fluctuación, d_1 , es obtenido tomando el promedio de la diferencia entre el primer par de valores de f, y multiplicando ese promedio resultante por $\sqrt{2}$. Del mismo modo es obtenido el valor d_2, tomando el promedio de la diferencia entre el segundo par de valores de f, y multiplicando el resultado por $\sqrt{2}$. La ecuación 3 representa el procedimiento descrito.

$$a_m = \frac{f_{2m-1} + f_{2m}}{\sqrt{2}} \; para \; m = 1,2,3, \ldots, N/2 \qquad (3)$$

Transformada Wavelet Discreta de Haar bidimensional

Una imagen es un conjunto de valores bidimensionales por lo que será necesario aplicar la DHWT en cada dimensión [Li, 2003]. Se denota DHWT-1D como la transformada unidimensional y DHWT-2D como la trasformada bidimensional. La DHWT-2D puede ser aplicada a una imagen, siempre y cuando el número de filas y columnas de la imagen sea par. Para obtener la DHWT-2D de una imagen f, es necesario realizar la DHWT-1D en cada fila de f, generando así una nueva imagen y sobre la imagen generada en el paso anterior, se aplica DHWT-1D pero ahora en cada una de las columnas de f. La DHWT-2D de una imagen puede ser representada como se muestra en figura 2.

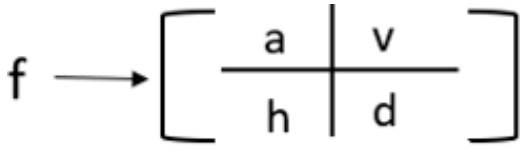

Figura 2 Representación de la DHWT-2D.

Donde h,d,a y v son subimágenes con $M/2$ filas y $N/2$ columnas. La subimagen a es el resultado de obtener la tendencia a lo largo de las filas y columnas, dando como resultado una versión de menor resolución que la imagen original f. La subimagen h es creada calculando la tendencia a lo largo de las filas y la fluctuación a lo largo de las columnas. Como consecuencia, en cualquier parte dentro de una imagen en dónde haya bordes horizontales, las fluctuaciones detectarán dichos bordes. Esta subimagen es la fluctuación horizontal porque tiende a enfatizar los bordes horizontales de una imagen.

La subimagen v es similar a la subimagen h, excepto que en la subimagen v la fluctuación es calculada a lo largo de las filas y la tendencia a lo largo de las columnas, obteniendo así una subimagen que resalta más los bordes verticales dentro de la imagen original, por lo que es la fluctuación vertical.

Por último, la subimagen do fluctuación diagonal, la cual a diferencia de las demás tiende a enfatizar las características diagonales de la imagen original, ya que es creada calculando la fluctuación de las filas y columnas de la imagen fuente.

Sistema embebido CPU-GPU

Actualmente es común encontrar en estaciones de trabajo o computadoras personales, como el sistema cómputo que se muestra en la figura 3.

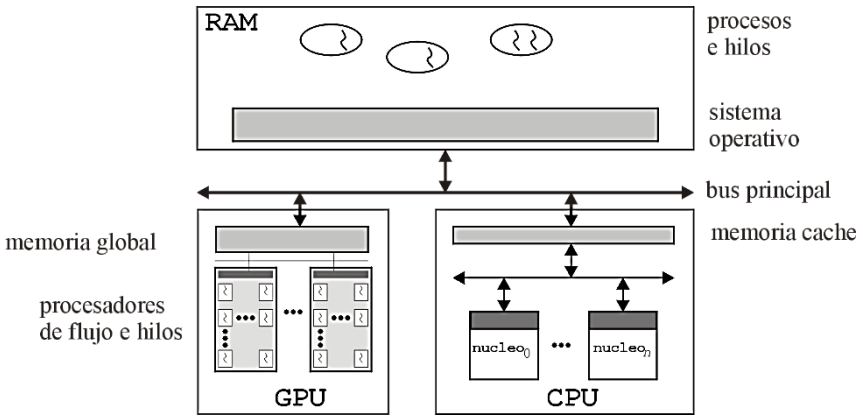

Figura 3 Sistema de cómputo con un CPU multinúcleo y un GPU.

Este sistema cómputo consiste en un CPU multinúcleo interactuando a través de la memoria principal (RAM) con una unidad de procesamiento de gráficas o GPU

(*Graphics Processing Unit*). Una aplicación implementada en este tipo de sistema podría beneficiarse de la gran cantidad de unidades procesamiento disponibles y de los diferentes tipos de memoria que ambas unidades ofrecen. Los hilos ejecutados en los núcleos de los CPUs podrían trabajar de manera colaborativa o paralela en una aplicación, disminuyendo su tiempo de ejecución. El nivel de paralelismo puede aumentar si las partes más costosa en cómputo de la aplicación lo realizan los hilos del GPU, al haber más unidades de ejecución, se pueden crear más hilos en un GPU [Owens, 2011]. El desarrollo tecnológico ha permito que un sistema como el mostrado en la figura 3 sea inmerso, empotrado o embebido en un chip. A este tipo de sistemas embebidos también se les conoce como sistema en chip o SOC (System On Chip). El desarrollo de sistemas como un SOC genera múltiples ventajas, destacando el bajo consumo de energía, la seguridad y el gran potencial para mejorar el rendimiento de las aplicaciones [Nickolls, 2010].

2. Métodos

Sistema embebido Tegra X1

El programa para la fusión de imágenes se desarrolló para ser ejecutado en el sistema de cómputo CPU-GPU embebido Tegra X1 de la compañía Nvidia, el cual está montado en una tarjeta Jetson TX1. Un esquema simplificado de este sistema se muestra en la figura 4.

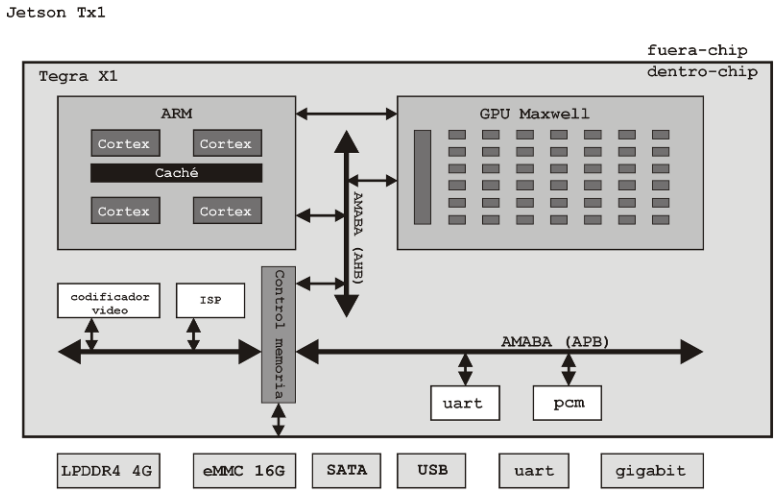

Figura 4 Sistema CPU-GPU embebido Tegra X1.

El sistema embebido Tegra X1 integra un CPU ARM de 4 núcleos con capacidad operativa de 32 bits y 64 bits, un GPU con arquitectura Maxwell de 256 núcleos o procesadores de flujos, un doble procesador de imágenes ISP (Image Service Processors), sistemas de procesamiento dedicado para audio, video e imágenes multifunción y varios sistemas de comunicación tanto alámbricos e inalámbricos. Los elementos son conectados por un sistema de buses de tipo AMABA (Advanced Microcontroller Bus Architecture), tanto para periféricos de baja velocidad APB (AMBA Pefipherial Bus), como dispositivos de alta velocidad AHB (AMBA High-speed Bus). Este sistema embebido es incluido como un módulo central de en una tarjeta para interactuar con otros módulos de propósito específico.

El sistema Jetson TX1 es considerado un sistema en módulos, SOM (S*ystem on Module*), en este caso, los módulos o componentes son utilizados por el módulo principal, el Tegra X1. Algunos de estos componentes son los módulos RAM de tipo LPDDR (*Low-Power* DDR SDRAM), módulos eMMC (*embedded MultiMediaCard*), módulos de red alámbrica Gigabit ethernet e inalámbrica Wifi, varios conectores UART, entre otros.

Módulos para Fusión de Imágenes

En la figura 5 se presenta un esquema de los módulos que conforman el sistema de fusión de imágenes.

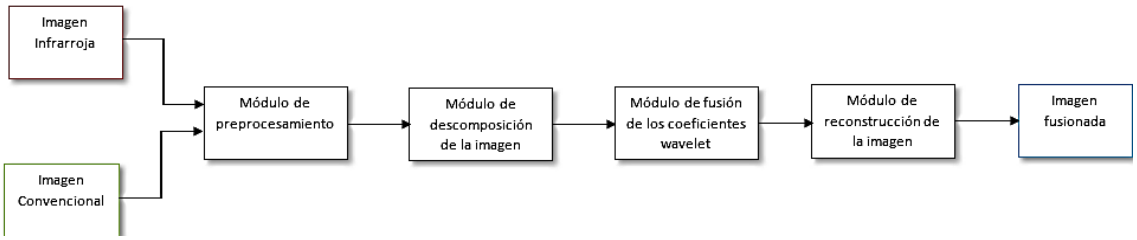

Figura 5 Módulos para el desarrollo de fusión de imágenes por medio DHWT-2D.

Con el propósito de mejorar el tiempo de procesamiento de la fusión de imágenes, se realizó un análisis para determinar qué partes de la aplicación secuencial se podrían ejecutar en paralelo y con ello generar estrategias de ejecución paralela entre los hilos del CPU con los hilos del GPU [Sanders, 2011]. El análisis realizado

arrojó como resultado que los siguientes módulos fueran diseñados para llevar a cabo sus operaciones en paralelo: módulo de descomposición de la imagen, módulo de fusión de los coeficientes de Wavelet y módulo de reconstrucción de la imagen.

El **módulo de pre-procesamiento** comprobará que las imágenes, tanto la convencional como la infrarroja, cumplan con los requerimientos para ser procesadas por los módulos siguientes. Las imágenes deben tener la misma resolución y esta deberá ser un número par. En ambos casos las imágenes deben estar en escalar e grises y cada píxel debe ser representado por un byte. Las imágenes preprocesadas de esta forma serán las imágenes fuente.

Como se explicó en la Introducción, para calcular la DHWT-2D de una imagen se debe aplicar la DHWT-1D a cada una de las filas de la imagen original y a la imagen resultante se le aplica nuevamente la DHWT-1D pero sobre las columnas, obteniendo así una imagen descompuesta en cuatro subbandas. Los algoritmos implementados en el **módulo de descomposición** de la imagen, figura 6.

```
For row=0 → row< Height do
    For col=0 → col< ( Width /2 ) do
        Fluctuacion = ( ValorPixelAcutal – ValorPixelSiguiente )/2
        Tendencia = ( ValorPixelAcutal + ValorPixelSiguiente )/2
        imgReturn[ col ] =Tendencia
        imgReturn[ col + ( Width /2 ) ] = Fluctuacion
```

a) *DHWT-1D para filas*

```
For col=0 → col< Width do
    For row=0 → row < (Height /2) do
        Fluctuacion = ( ValorPixelAcutal – ValorPixelSiguiente )/2
        Tendencia = ( ValorPixelAcutal + ValorPixelSiguiente )/2
        imgReturn[ row ] =Tendencia
        imgReturn[ row + ( Height/2 ) ] = Fluctuacion
```

b) DHWT-1D para columnas

Figura 6 Algoritmos del módulo de descomposición.

Este módulo fue diseñado para llevar a cabo los cálculos de la DHWT-2D en paralelo, para ello se dividió el procedimiento en dos submódulos, el primero calculará la DHWT-1D de las filas, mientras que en el segundo calculará la DHWT-1D de las columnas. Cada submódulo recibirá el número de bloques y el número de hilos por bloque que se ejecutarán para llevar a cabo el cálculo en el GPU. De acuerdo a la resolución de la imagen y a los recursos del GPU, se pueden plantear diferentes distribuciones de bloques e hilos por bloque. En general, se pueden plantear las siguientes ecuaciones para la distribución del trabajo por bloques bidimensionales de hilos sobre una imagen de resolución ($WIDHT \times HEIGH$). Para el sub-bloque de filas, ecuación 4 y para sub-bloque de columnas, ecuación 5.

$$BloquesenY = \left(\frac{HEIGH}{hilos_y}\right) \tag{4}$$

$$BloquesenY = \left(\frac{HEIGH/2}{hilos_y}\right) \tag{5}$$

De esta forma, para el cálculo de la DHWT-1D de las filas de una imagen de 512×512 píxeles con bloques bidimensionales de 32×32 hilos, habrá 128 bloques, 8 bloques en X y 16 bloques en Y, cada uno con 1024 hilos. En total se tendrán 13,1072 hilos trabajando de manera simultánea. Con esta distribución de procesamiento, cada hilo se encargará de calcular tanto la tendencia como la fluctuación. Si se desea que un solo hilo calcule la tendencia y otro hilo la fluctuación, se deberá crear el doble de hilos. Antes de hacer un cambio de este tipo se debe considerar la arquitectura del GPU. La distribución del trabajo por bloques bidimensionales de hilos para el cálculo de la DHWT-1D de las filas se muestra gráficamente en la figura 7 y para el cálculo de la DHWT-1D de columnas se muestra gráficamente en la figura 8. Una vez descompuesta la imagen fuente infrarroja y la imagen fuente convencional, el **módulo de fusión de coeficientes wavelet** procederá con la selección y combinación de los coeficientes Wavelet de cada una de las imágenes de acuerdo a las reglas del proceso de fusión.

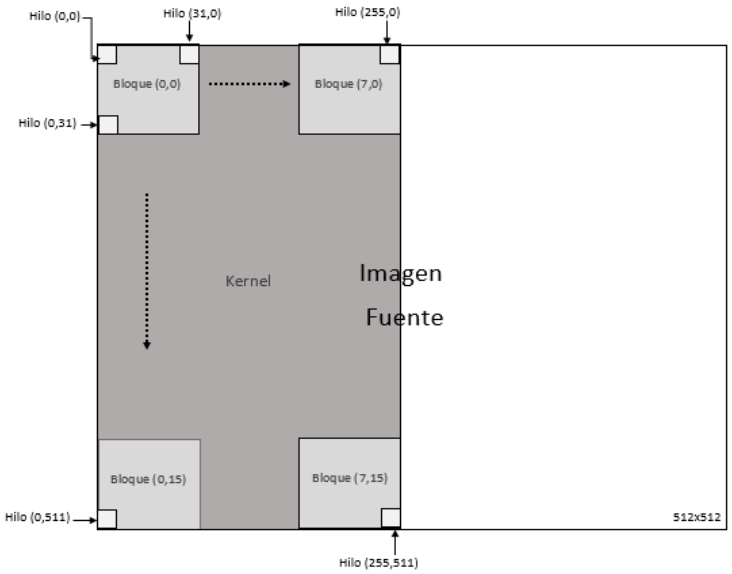

Figura 7 Bloques bidimensionales de hilos para DWHT-1D sobre filas.

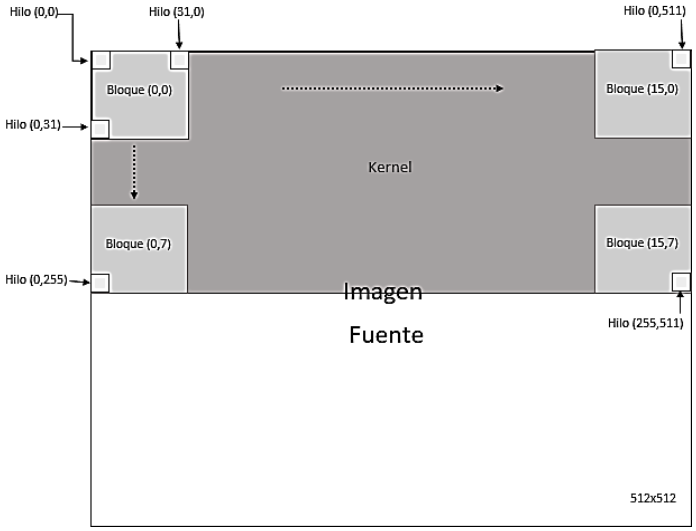

Figura 8 Bloques bidimensionales de hilos para DWHT-1D sobre columnas.

Para las regiones de bajas frecuencias se obtiene el promedio de los coeficientes Wavelet de las imágenes fuente. Para las regiones de altas frecuencias se comparan los coeficientes Wavelet de las imágenes fuente y se seleccionando la de mayor valor.

Al aplicar estas reglas se obtiene como resultado una nueva imagen con regiones de altas y bajas frecuencias que contienen coeficientes tanto de la imagen infrarroja como de la imagen convencional. Para esta implementación se crea la distribución de procesamiento mostrada en la ecuación 6.

$$\left(\frac{WIDHT}{hilos_x}\right) \times \left(\frac{HEIGHT}{8 * hilos_y}\right) \tag{6}$$

La representación gráfica de los bloques e hilos se puede observar en la figura 9.

Para la fusión se ha lanzado un hilo para cada uno de los píxeles de la imagen fuente, esto se debe a que cada hilo es el encargado de fusionar los coeficientes wavelet de la imagen infrarroja con los de la imagen convencional, colocando el resultado de esta fusión en el píxel correspondiente a cada hilo.

De manera similar al cálculo la DHWT-2D, se obtiene la transformada inversa, DHWIT-2D, la cual también consta de dos módulos, el primero calcula la inversa de la transformada en cada una de las filas, mientras que el segundo determinará inversa de la transformada en cada una de las columnas.

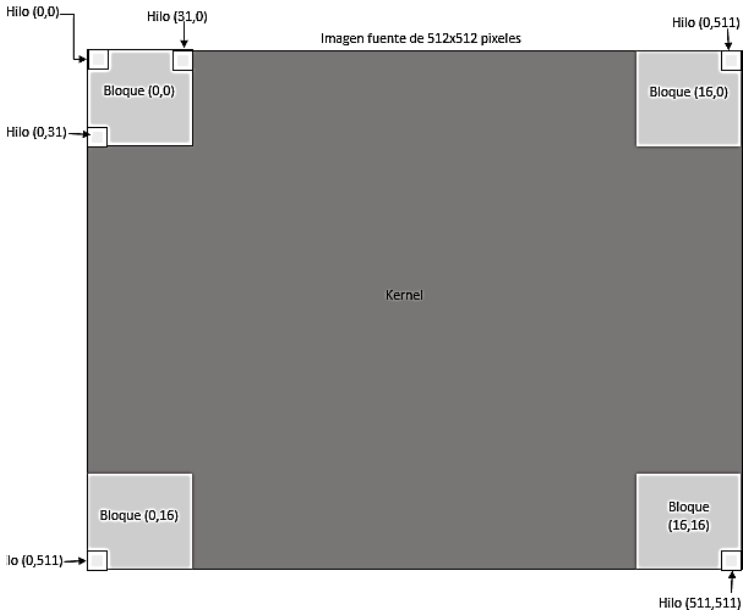

Figura 9 Bloques bidimensionales de hilos para fusión de los coeficientes de Wavelet.

El **módulo de reconstrucción de la imagen** se encarga de llevar estas operaciones siguiendo los algoritmos que se muestran en la figura 10.

La distribución de procesamiento de los módulos de reconstrucción, es idéntica a de los módulos para descomposición de la imagen, así la distribución para la DHWT-1D de las filas es igual al de la figura 7, mientras que para la DHWT-1D de las columnas es idéntico al de la figura 8.

```
For row=0 ⟶ row< Height do
    For col=0 ⟶ col< (Width /2) do
        Tendencia = imgSource[col]
        Fluctuacion = imgSource[col+( Width /2 )]
        imgReturn[ col ] = (Tendencia + Fluctuacion)/2
        imgReturn[ col +1] = (Tendencia - Fluctuacion)/2
```

a) *DHWT-1D para filas*

```
For col=0 ⟶ col< Width do
    For row=0 ⟶ row < (Height /2) do
        Tendencia = imgSource[row]
        Fluctuacion = imgSource[row+( Height /2 )]
        imgReturn[ row ] =(Tendencia + Fluctuacion)/2
        imgReturn[ row + ( Height/2 ) ] = (Tendencia - Fluctuacion)/2
```

b) DHWT-1D para columnas

Figura 10 Algoritmos del módulo de reconstrucción.

3. Resultados

En la figura 11 se presenta la imagen obtenida al fusionar las imágenes convencional e infrarroja de la figura 12, las cuales serán utilizadas para realizar un análisis de los resultados.

Figura 11 Imagen fusionada.

a) *Imagen convencional o visible* *b) Imagen infrarroja*

Figura 12 Características importantes de las imágenes fuente.

Sistemas de ejecución

Se realizaron varios experimentos sobre dos sistemas de cómputo, los cuales tienen componentes similares, pero estructuralmente diferentes. El primero es una estación de trabajo (ET), como el mostrado en el esquema de la figura 3. La ET cuenta con un CPU Intel i7 con 4 núcleos, 4 GB de DDR-RAM y una tarjeta de procesamiento de gráficas Quadro K420 el cual contiene un GPU con 192 núcleos con arquitectura Kepler. El segundo es la tarjeta de desarrollo Jetson TX1 la cual tiene integrada el sistema embebido Tegra X1, mostrado en el esquema de la figura 4. En la tabla 1 se muestra una comparativa de sus principales recursos.

Tabla 1 Comparativa de recursos de los sistemas de cómputo utilizados.

	CPU [núcleos]	RAM principal	GPU-Arquitectura [núcleos]	RAM GPU	Memoria BW
ET	Intel i7 [4]	4GB	Kepler [192]	1GB	29 GB/s
Tegra X1	ARM [4]	16GB	Maxwell [256]	4GB	16 GB/s

Tiempo de Ejecución

Los tiempos de ejecución de la implementación secuencial y paralela de ambos sistemas se muestran en la tabla 2 y en la figura 13 su gráfica. El tiempo secuencial fue tomado en el GPU Quadro K20 de la estación de trabajo al lanzar un solo hilo para realizar todas las operaciones. Estos tiempos son el resultado de fusionar imágenes de diferentes resoluciones.

Tabla 2 Comparativa del tiempo de ejecución en tres arquitecturas diferentes.

Resolución	Secuencial [s]	ET, Quadro K420 [s]	Tegra X1 [s]
512x512	0.409	0.060	0.094
1021x1024	1.626	0.113	0.122
2048x2048	6.960	0.328	0.239
4096x4096	28.731	0.845	0.708
8192x8192	145.775	2.777	2.104

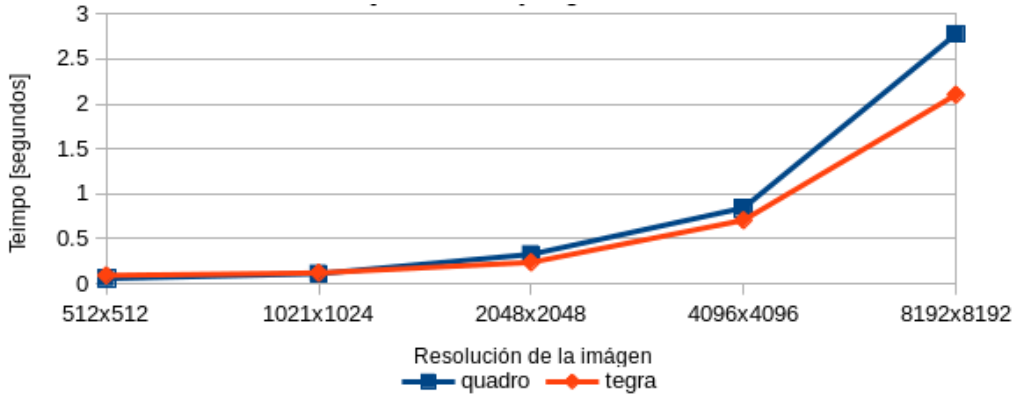

Figura 13 Gráfica de los tiempos de ejecución de paralela.

El tiempo de ejecución considera los tiempos de procesamiento tanto del CPU como del GPU. El CPU se encargó del preprocesamiento de la imagen y la comunicación con el GPU. El GPU se realizó la descomposición de la imagen, la fusión de coeficientes Wavelet y la reconstrucción de la imagen.

Puede notarse tanto en la tabla 2 como en la figura 13, que los tiempos de ejecución del sistema embebido Tegra X1 son mejores en alrededor de un 30% sobre la estación de trabajo y respecto al tiempo secuencial el sistema Tegra X1 tiene una aceleración del 69x.

4. Discusión

Para realizar análisis de los resultados obtenidos, se han resaltado con círculos rojos numerados las características más importantes una imagen convencional y una imagen infrarroja de la misma escena, mostradas en imágenes de figura 12.
En la figura 12a se muestra la imagen convencional con algunas zonas marcadas con círculos rojos numerados del 1 al 4. En los círculos 1 y 2 se pueden apreciar los señalamientos de tránsito, sin embargo, en los círculos 3 y 4 las personas que transitan por las aceras no se alcanzan a distinguir. En la figura 12b se muestra la imagen infrarroja con la misma escena de la imagen mostrada en la figura 12a y con las mismas zonas marcadas con círculos. Se puede observar que en los círculos marcados como 1 y 2 los señalamientos de tránsito no se logran percibir, sin embargo, en los círculos 3 y 4 de la imagen infrarroja las personas que transitan son fácilmente identificables. Identificados los principales elementos de cada una de las imágenes fuente, en la figura 14 se muestra el resultado de la fusión imágenes.

Figura 14 Imagen fusionada.

En los círculos 1 y 2 de la figura 14 se pueden apreciar que los señalamientos de tránsito no se pierden como sucede en la imagen infrarroja y, en los círculos 3 y 4 las personas que transitan en las aceras son fácilmente identificables.

Por otro lado, en cuanto al tiempo de ejecución para la fusión de imágenes, es evidente que las versiones paralelas ejecutadas en los GPU fueron muy superiores a la implementación secuencial, lo cual cumple con el objetivo de generar una imagen fusionada en menor tiempo, cercano a la ejecución en tiempo real. Respecto a la comparativa entre la estación de trabajo y el sistema embebido, el sistema embebido obtuvo los mejores tiempos esto debido principalmente a la integración en un chip mejora las comunicaciones entre los sistemas embebidos.

5. Conclusiones

A partir de una imagen convencional y una imagen infrarroja, es posible obtener una imagen fusionada que asista la conducción nocturna, utilizando la Transformada Wavelet Discreta de Haar. Las operaciones para la fusión pueden ejecutarse en paralelo en un GPU para mejorar su tiempo de ejecución. Si el sistema de ejecución paralelo se encuentra como un sistema embebido, se tienen mayores ventajas sobre un sistema convencional, no solamente mejora el tiempo de ejecución, sino que además se tiene un menor consumo de energía, lo cual permite que a la aplicación pueda ser fácilmente integrada a un automóvil o ser un sistema de visión portable.

El sistema puede mejorar tanto en visualización como tiempo en tiempo de ejecución. Para mejorar la visualización, se pueden crear reglas de selección de coeficientes de Wavelet más elaboradas que nos permitan apreciar los elementos principales de las imágenes fuente con mayor claridad. Los experimentos se realizaron con imágenes en escala de grises, lo cual limita los detalles en la imagen final fusionada.

Al utilizar imágenes a color se aprovechar de una mejor manera las características para generar imágenes fusionadas con mayores detalles y claridad. El tiempo de procesamiento puede mejorar al utilizar diferentes estrategias de paralelización, utilizando diferentes recursos que los GPUs ofrecen, como combinar accesos entre

la memoria global, la memoria compartida y la memoria de textura, a costa de algoritmos más complejos por razones de dependencia de datos.

6. Bibliografía y Referencias

[1] Aymaz S. y Köse C., Multi-focus image fusion using stationary wavelet transform (swt) with principal component analysis (pca), en Department of Computer Engineering, Karadeniz Technical University, Trabzon,Turquía, ene 1997.

[2] Herrington W. F., Horn B. K. P. y Masaki I., Application of the discrete haar wavelet transform to image fusion for nighttime driving, en IEEE Proceedings. Intelligent Vehicles Symposium, 2005, Las Vegas, NV, USA, pp. 273–277, 2005.

[3] Kannan K. y Arumuga S., Optimal decomposition level of discrete wavelet transform for pixel based fusion of multi-focused images, en Department of Computer Science, St. Hindu College, Tamilnadu, India, enero 1997.

[4] Li J. P., Proceedings of the Third International Conference on Wavelet Analysis and its Applications. Singapore: World Scientific, 2003.

[5] Sanders J. y Katndrot E., CUDA by example; an introduction to general-purpose GPU programming. Boston, MA: Addison-Wesley, 2011.

[6] Nickolls J. y Dally W. J., The gpu computing era, in IEEE Micro, vol. 30, no. 2, pp. 56–69, March-April 2010.

[7] Owens J. D., Houston M., Luebke D. y Green S., J. Stone J. C. Phillips, Gpu computing, Proceedings of the IEEE, vol. 96, no. 5, pp. 879–899, May 2008.

[8] Pal A. R. y Singha A., A comparative analysis of visual and thermal face image fusion based on different wavelet family, en 2017 International Conference on Innovations in Electronics, Signal Processing and Communication (IESC), pp. 213–218, Shillong, 2017.

[9] Sundararajan D., Discrete Wavelet Transform:A Signal Processing Approach. ton,WA: John Wiley Sons, Incorporated, 2015.

[10] Walker J. S., A primer on wavelets and their scientific applications. Chapman Hall/CRC, 2008.