Intelligent Classification of Large-Scale Remotely Sensed Hyperspectral Images Using Multi-GPU Computing

A. Castillo, *Member, IEEE,* J. Vázquez, *Member, IEEE,* J. Ortegón, *Senior Member, IEEE,* R. Carrasco, and J. Avilés

***Abstract*—Image classification is one of the most popular tasks used to analyze remote sensing signatures (RSS) of a geographical region from remotely sensed hyperspectral images. However, the high dimensionality of such hyperspectral images raises a series of new challenges. In this paper, a new approach for real-time intelligent classification of large-scale hyperspectral imagery which aggregates Fuzzy logic and the fused weighted order statistic (WOS) with the minimum distance to mean (MDM) techniques using commodity graphics processing units (GPUs) is addressed. Within this context, intelligent image processing methods are algorithmically adapted via parallel computing techniques and efficiently implemented in two NVIDIA Tesla C2075 GPUs. Experimental results demonstrate how such unification reduces drastically the computational load of the real-world hyperspectral classification tasks resulting in efficient numerical algorithms suitable for real-time multi-GPU-adapted implementation.**

***Index Terms*—remote sensing, GPU computing, Image processing.**

I. Introducción

El procesamiento de imágenes hiperespectrales de percepción remota (RS, por sus siglas en inglés) ha proporcionado ventajas y cambios muy significativos en el análisis de la información geográfica. Particularmente, en el procesamiento de imágenes geoespaciales se obtiene información sobre las características físicas, tales como agua, vegetación, suelo, minerales, entre otros. Sin embargo, cuando las imágenes RS son capturadas, éstas ya han sido contaminadas con diferentes tipos de ruido. Las fuentes de ruido dependen del detector utilizado, pero tres son comunes a la mayoría de los dispositivos optoelectrónicos: el ruido de disparo (shot noise), ruido oscuro (dark noise) y el ruido térmico (Johnson noise). El ruido de disparo se debe a la naturaleza aleatoria del proceso de detección del sensor. El ruido oscuro se asocia a la carga oscura (dark current) del circuito. Finalmente, el ruido térmico se asocia a la temperatura de los elementos de circuitos resistivos. Los tres tipos de ruido mencionado son estadísticamente independientes, por lo que las varianzas del ruido simplemente se suman. El ruido de disparo y el oscuro tienen una distribución de Poisson, pero a mayores niveles de señal se pueden aproximar por una distribución normal; por lo que el ruido total del sistema, generalmente, se asume que sigue una distribución normal. Así, aun cuando proviene de distintas fuentes, el ruido se puede considerar como aditivo e impide la correcta clasificación e interpretación [1]. Por lo anterior, una vez capturada la imagen es necesario que sea procesada con diferentes algoritmos con el fin de mejorar la calidad de la imagen adquirida [2].

En imágenes hiperespectrales el reto de procesamiento es aún mayor, debido a que las imágenes hiperespectrales se conforman de múltiples canales, los cuales corresponden a diferentes bandas espectrales (por ejemplo, 224 bandas) y que están asociadas a un rango de longitud de onda [3]. Por otra parte, la implementación de técnicas de procesamiento en paralelo, en conjunto con arquitecturas especializadas de cómputo de alto desempeño (HPC, por sus siglas en inglés), son necesarias para satisfacer la alta demanda de procesamiento de las imágenes hiperespectrales [4]. Sin embargo, en muchas aplicaciones basadas en HPC, aún quedan sin resolverse problemas a nivel teórico y de procesamiento de datos con arquitecturas aceleradoras especializadas.

Existen diversas formas de hacer la clasificación de imágenes de percepción remota multi e hiperespectrales [5], [6], [7]. En [5], López-Fandiño et al. presentaron una clasificación de imágenes hiperespectrales usando unidades de procesamiento gráfico (GPU, por sus siglas en inglés) y una máquina de aprendizaje extremo (ELM, por sus siglas en inglés); su ELM la expresan en términos de matrices para aprovechar la arquitectura de los GPU, y la comparan con una máquina de vectores de soporte, dado que éstas son la base para la ELM. Por su parte, Wu et al. [6] realizan la clasificación de imágenes hiperespectrales con GPUs a través de campos aleatorios de Markov (MRF, por sus siglas en inglés); ellos combinan la información espectral de un clasificador de regresión logística multinomial dispersa (SMLR) y la información espacial modelada por la función potencial asociada a un MRF. Sahar [7] hace una clasificación de imágenes hiperespectrales con un algoritmo de aprendizaje no supervisado (K-means) y se compara con ISODATA [8]; se elige generalmente ISODATA como referencia por su amplia divulgación e implementación disponible [9].

En este artículo se propone una nueva metodología para el mejoramiento y clasificación inteligente en tiempo real de imágenes hiperespectrales RS implementada en múltiples GPU. La

A. Castillo Atoche and J. Aviles Viñas are with the Department of Mechatronics, Universidad Autónoma de Yucatán, Mérida, México, 97310, e-mail: acastill,javiles@correo.uady.mx.

J. Vázquez Castillo and J. Ortegón Aguilar are with the Department of Engineering, Universidad Quintana Roo, Chetumal, México, 77019, e-mail: jvazquez,jortegon@uqroo.edu.mx.

R. Carrasco Álvarez is with the Department of Electronics Engineering, Universidad de Guadalajara, Guadalajara, México, 44430, e-mail: r.carrasco@academicos.udg.mx.

metodología introducida es la siguiente: (i) en la primera etapa se implementa un algoritmo de reconstrucción de la imagen, vía el uso del algoritmo de mínimos cuadrados restringidos ponderados (WCLS, por sus siglas en inglés), para eliminar los efectos de degradación del canal de comunicaciones y los efectos del ruido, (ii) la segunda etapa implementa un algoritmo de post-procesamiento para el mejoramiento de la imagen reconstruida usando difusión anisotrópica difusa, la cual preserva los bordes de las imágenes reconstruidas, (iii) la tercera etapa consiste en clasificar la imagen hiperespectral mediante la fusión de los estadísticos de orden ponderado (WOS, por sus siglas en inglés) y distancia mínima a la media (MDM, por sus siglas en inglés).

La metodología propuesta requiere de recursos computacionales significativos para aplicar las técnicas de procesamiento ya descritas, razón por la cual en este trabajo se propone la implementación de los algoritmos en múltiples GPUs en una arquitectura paralela HPC, y con ello acelerar las operaciones altamente complejas necesarias en el procesamiento de imágenes hiperespectrales de una región geográfica en particular. A pesar de que en la literatura se describen implementaciones HPC para imágenes hiperespectrales [5], [6], [7], en nuestro trabajo se presenta una implementación multi-GPU de los algoritmos descritos en la metodología.

El resto del artículo se organiza de la siguiente manera: en la sección II se describe el modelo de percepción remota y los algoritmos para la reconstrucción, el mejoramiento y la clasificación de las imágenes hiperespectrales. La implementación multi-GPU de los algoritmos se presenta en la sección III, mientras que la efectividad de los algoritmos y su desempeño son analizados mediante dos casos de estudio reales de procesamiento de imágenes en la sección IV. Finalmente, las conclusiones se presentan en la sección V.

II. Modelo de Percepción Remota para Imágenes Hiperespectrales

En esta sección, se describe brevemente el modelo de percepción remota adaptado para el procesamiento de imágenes hiperespectrales. En la Fig. 1 se ilustra el flujo de diseño para la reconstrucción, mejora y clasificación de las firmas espectrales del medio ambiente (vegetación, suelo, regiones hidrológicas, entre otros).

En el análisis de la Fig. 1, se puede observar que el primer bloque adquiere las imágenes hiperespectrales y el segundo, realiza la reconstrucción del patrón de potencia espacial-espectral (SSP, por sus siglas en inglés) de las señales dispersas adquiridas remotamente de la escena. En este estudio en particular, el reto científico consiste en solucionar el mejoramiento y extracción de firmas espectrales RS de la imagen hiperespectral utilizando HPC con múltiples GPUs. La propuesta consiste en una cadena de procesamiento (reconstrucción robusta, mejoramiento con difusión anisotrópica difusa y clasificación WOS-MDM) que permite generar mapas digitales de las características físicas en tiempo real.

A. Reconstrucción de la Imagen

Una vez que se adquiere la imagen hiperespectral, los datos $u(y) = s(y) + n(y)$ de cada banda se modelan como una

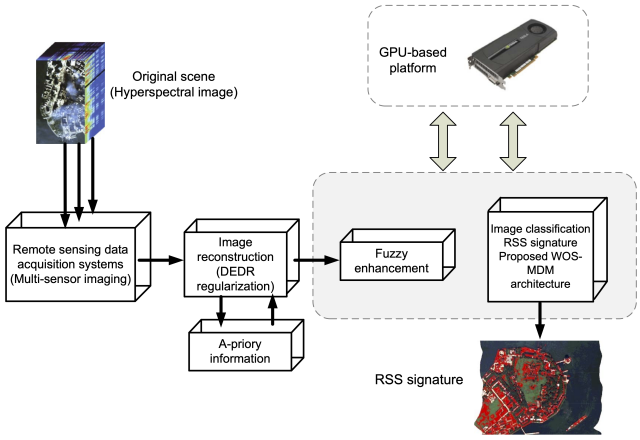

Fig. 1. Metodología para el procesamiento RS hiperespectral.

superposición de las señales s y ruido aditivo n [1]. El modelo de observación de datos RS para una instancia específica en el tiempo t está dada por

$$\mathbf{u} = \mathbf{S}e + \mathbf{n} \tag{1}$$

donde $\mathbf{u}$, $\mathbf{e}$ y $\mathbf{n}$ son vectores de dimensión finita del campo de observación u, especificados por el formato de modulación particular empleado en el sistema RS [10], [11]

La reconstrucción de la imagen se consigue utilizando el algoritmo ponderado limitado de mínimos cuadrados (WCLS) presentado en [11].

B. Difusión Anisotrópica Difusa

Debido a que el algoritmo WCLS actúa como un filtro pasa-bajos, se disminuye el ruido en la imagen reconstruida. Sin embargo, los bordes en la imagen reconstruida se hacen borrosos. Por lo tanto, en este trabajo se utiliza el método de difusión anisotrópica propuesta en [12] descrito mediante la ecuación diferencial:

$$\frac{\partial \hat{b}(\mathbf{r};t)}{\partial t} = \operatorname{div}\big(c(\mathbf{r};t)\nabla \hat{b}(\mathbf{r};t)\big), \tag{2}$$

para $\mathbf{r} = (x, y) \in \mathbf{R}$ una escena rectangular continua 2-D. En (2), $\frac{\partial \hat{b}(\mathbf{r};t)}{\partial t}$ representa la evolución del estimado reconstruido $\hat{b}(\mathbf{r};t)$, que proporciona la preservación de bordes en las regiones de la escena con un alto contraste en el gradiente, mientras que se suaviza por ventanas en las zonas homogéneas de la imagen. También, $c(\mathbf{r};t) = g(\left\|\nabla \hat{b}(\mathbf{r};t)\right\|)$ representa el coeficiente de difusión, $\nabla \hat{b}(\mathbf{r};t)$ denota el gradiente de la imagen y $g(\cdot)$ es la función de paro en los bordes, que se selecciona como una función decreciente del gradiente de la imagen reconstruida. La versión discreta de (2) puede ser implementada utilizando el enfoque difuso presentado en [13]:

$$\begin{aligned}\hat{b}_r^{[i+1]} &= \hat{b}_r^{[i]} \\ &+\lambda\Bigg[\frac{1}{d_x^2}\cdot c_N\cdot\gamma_N(\hat{b})+\frac{1}{d_x^2}\cdot c_S\cdot\gamma_S(\hat{b}) \\ &+\frac{1}{d_y^2}\cdot c_E\cdot\gamma_E(\hat{b})+\frac{1}{d_y^2}\cdot c_O\cdot\gamma_O(\hat{b}) \\ &+\frac{1}{d_d^2}\cdot c_{NE}\cdot\gamma_{NE}(\hat{b})+\frac{1}{d_d^2}\cdot c_{NO}\cdot\gamma_{NO}(\hat{b}) \\ &+\frac{1}{d_d^2}\cdot c_{SE}\cdot\gamma_{SE}(\hat{b})+\frac{1}{d_d^2}\cdot c_{SO}\cdot\gamma_{SO}(\hat{b})\Bigg]_r^{[i]}\end{aligned} \tag{3}$$

donde $c_N, c_S, c_E, c_O, c_{NE}, c_{NO}, c_{SE}, c_{SO}$ representan los coeficientes de difusión, d_x, d_y, d_d son las distancias entre los pixeles y γ_N a γ_{SO} representan los bordes en cada dirección como un resultado del sistema de lógica difusa. Una vez que la imagen ha sido reconstruida, es posible extraer las firmas de detección remotas (RSS) de interés. Esto se logra a través de la implementación del operador de extracción de firma en la imagen reconstruida obtenida por el algoritmo difuso, como sigue:

$$\hat{\Lambda}_{RSS} = \Lambda\{\hat{\mathbf{B}}_F\} \tag{4}$$

donde $\hat{\mathbf{B}}_F = L(\hat{\mathbf{b}}_F)$, $L(\hat{\mathbf{b}}_F)$ es el ordenamiento lexicografico de $\hat{\mathbf{b}}_F$ y $\hat{\mathbf{b}}_F$ es la escena reconstruida utilizando el algoritmo difuso. Λ es el operador de extracción de firmas que se implementa en este artículo por la fusión de los estadísticos de orden ponderadas (WOS, por las siglas de weighted order statistics) y la distancia mínima a las medias (MDM, por las siglas de minimum distance to means).

C. WOS

El método WOS se considera como una generalización del filtro de mediana calculada sobre las diferentes bandas espectrales de la imagen reconstruida. Este método se basa en conocimiento a priori acerca de la media de la firma espectral μ de c diferentes clases a clasificar. Dicha información puede resumirse en una matriz donde cada elemento (c, b) corresponde a la respuesta del posible elemento c-ésimo de la banda espectral b-ésima. En la primera etapa de la metodología WOS, se calcula la mediana en una ventana $\mathbf{W}$ centrada en el píxel de interés sobre cada banda de la imagen hiperespectral. La generación de la mediana se formaliza como

$$\gamma_{i,j,k} = \text{median}(\mathbf{W}_{i,j,k}), \tag{5}$$

donde $\mathbf{W}_{i,j,k} = (\pi_{i-1,j-1,k}, \ldots, \pi_{i,j,k}, \ldots, \pi_{i+1,j+1,k})$ es la ventana en la banda k-ésima centrada en el (i, j)-ésimo píxel y $\pi_(i, j, k)$ es el píxel en la posición (i, j) en la banda k.

A continuación, una vez que se obtienen las distancias $\Delta_{i,j,k}$, se genera la nueva imagen hiperespectral con los valores medios a priori correspondientes a su distancia mínima, formalizada como:

$$\begin{aligned}\mathbf{WOS}_{i,j,k} &= \mu_{r,k} \\ \text{for } r &= \min_{p\in[1,\ldots,c]}(\delta_p) \\ &= \min_{p\in[1,\ldots,c]}(|\gamma_{i,j,k}-\mu_{p,k}|)\forall i,k\end{aligned} \tag{6}$$

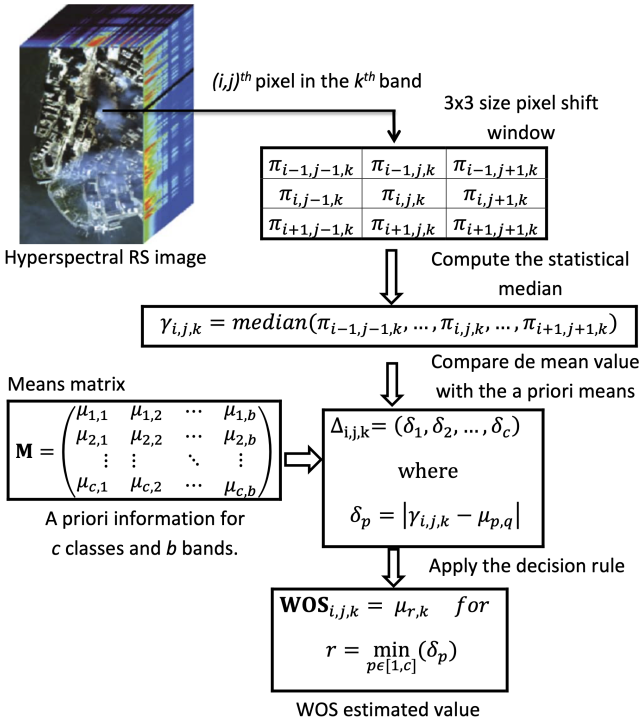

Fig. 2. Descripción gráfica del método WOS.

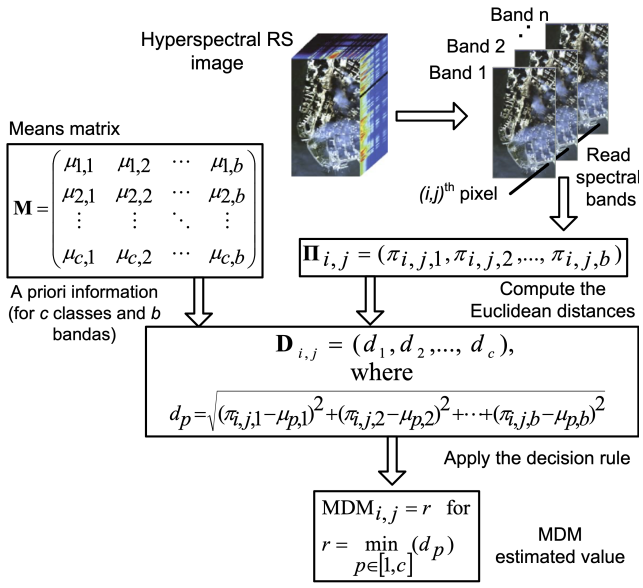

Fig. 3. Método de clasificación MDM.

donde $\mu_{r,k}$ es la media a priori para el elemento r-ésimo correspondiente a la k-ésima banda y p es una de las clases definidas. Este procedimiento se explica a detalle en la Fig. 2.

D. MDM

Este método se basa en el cálculo de la distancia euclidiana entre la respuesta espectral a priori de cada elemento y la respuesta espectral de cada pixel de la imagen dada por el método WOS. La Fig. 3 ilustra la metodología de diseño del método MDM.

A partir del análisis de la Fig. 3, se describe el vector $\Pi_{(i,j)}$ que se compone de cada (i,j)-ésimo píxel de la imagen WOS de cada banda espectral. Para calcular la salida del clasificador de MDM, la distancia euclidiana, entre cada entrada de $\Pi_{(i,j)}$ y las medias correspondientes de la matriz $\mathbf{M}$, se calcula como sigue

$$d_p = \sqrt{\sum_{l=1}^{b}(\pi_{i,j,l} - \mu_{p,l})^2} \tag{7}$$

donde b es el número de bandas espectrales de la imagen hiperespectral y $p = 1, \ldots, c$ es el número de clases (definido por el usuario). Después de esto, el método MDM proporciona un mapa de firmas RSS, en la que cada píxel está marcado con el elemento que tiene la distancia mínima euclidiana. Este criterio se puede establecer como

$$\begin{aligned} \mathbf{MDM}_{i,j} &= r \\ \text{for } r &= \min_{p\in[1,\ldots,c]}(d_p) \\ &= \min_{p\in[1,\ldots,c]}\left(\sqrt{\sum_{l=1}^{b}(\pi_{i,j,l} - \mu_{p,l})^2}\right) \end{aligned} \tag{8}$$

donde $r \in [1, \ldots, c]$ representa una de las clases RSS.

Una vez se ha descrito los algoritmos WOS y MDM, es posible implementarlos a través de técnicas de cómputo paralelo en dispositivos multi-GPU.

III. Implementación multi-GPU

En esta sección, se presenta la metodología para la implementación eficiente de la difusión anisotrópica difusa y los algoritmos WOS-MDM. Para las implementaciones se han usado dispositivos NVIDIA, librerías CUDA y su terminología. Teniendo en cuenta la agregación de las técnicas paralelas con multi-GPU, se identifican cuatro etapas en el diseño:

1) Implementación del método de difusión anisotrópica difusa con GPU;
2) Implementación del WOS con GPU;
3) Implementación del MDM con GPU;
4) Implementación multi-GPU de WOS-MDM-difuso.

A. Difusión Anisotrópica Difusa en GPU

El objetivo del núcleo (kernel) de difusión anisotrópica difusa es mejorar la imagen. En esta etapa, la primera operación es leer la imagen usando texturas con precisión subpíxel [14], [15] y tomar las diferencias en una vecindad de 3×3 para cada píxel de la imagen. Después, siguiendo la metodología de lógica difusa [16], se aplica el método de implicación (función `min`), una agregación (función `max`), y finalmente una deffuzzificación de los valores aplicando la difusión anisotrópica (3).

Es importante destacar que el procesamiento de un píxel es independiente de los otros; por lo tanto, los cálculos difusos son apropiados para una implementación en paralelo con GPU. En este sentido, cada GPU calculará la difusión anisotrópica difusa de una parte de la imagen.

Los valores de los píxeles resultantes son números flotantes, por lo tanto, con el fin de facilitar la visualización, son normalizados a valores entre 0 y 255 para cada canal. Esta tarea se realiza antes de mostrar la imagen resultante, pero es opcional en caso de un post-procesamiento.

Se utilizan funciones optimizadas disponibles en NVIDIA Performace Primitives [17], `nppsMinMax_3f` para obtener el mínimo y el máximo resultante del núcleo difuso, y `nppsNormalize_32f` para normalizar a valores entre 0 y 255.

B. Diseño de WOS Basado en GPU

En esta etapa, se implementa un núcleo CUDA para calcular el WOS de cada píxel de la imagen hiperespectral. Igualmente, WOS permite una paralelización en un esquema masivo. Con base en la Fig. 2, el núcleo WOS se pondrá en marcha una vez por cada banda, con tantos hilos (threads) como filas contiene la imagen hiperespectral, donde cada hilo ejecuta el WOS.

Es importante destacar, que para tener una implementación eficiente se traslapa la ejecución del núcleo WOS con las transferencias de datos [15]. Es decir, la GPU ejecuta el WOS de la k-ésima banda, mientras se trasmite la $(k+1)$-ésima banda.

Nuestra implementación uso 22 registros, con un tamaño de malla de (24,30,1) y el tamaño de bloque de (32,32,1), con una ocupación teórica de 66.7%. Las imágenes utilizadas tienen 224 bandas, por lo tanto, el núcleo se lanza 224 veces alcanzando 72.9%, 79% y 76% como ocupación mínima, máxima y media.

C. MDM Basado en GPU

En este apartado, se describe la implementación en GPU del algoritmo MDM. Se distinguen dos pasos: la distancia euclideana y la regla de toma de decisiones. Los procedimientos se describen a continuación:

1) Se calculan las distancias euclidianas, a través de un núcleo específico con una rejilla de $n \times m$ bloques, en el que cada bloque procesa un píxel. Los resultados se cargan en la memoria compartida de la GPU. Un algoritmo de reducción [14] se utiliza para tener el resultado final, la ocupación alcanzada por este núcleo es 57.1%, con un tiempo de ejecución de 274.8ms.
2) La regla de toma de decisiones también se implementa con un núcleo CUDA. En este paso, las distancias euclidianas resultantes son empleadas con tantos procesos como el número de píxeles de la imagen hiperespectral. Con este esquema paralelo, se tiene un tiempo de cálculo de 280 μs.

Por último, el pseudocódigo para implementar el algoritmo de WOS-MDM utilizando un único dispositivo GPU se presenta en la Fig. 4.

D. WOS-MDM-Difuso Multi-GPU

En este apartado, se describe la implementación del algoritmo WOS-MDM-difuso con múltiples GPU, donde la CPU debe realizar la distribución de la carga computacional entre

1: *% b denota el número de bandas espectrales %*
2: Asignar memoria lineal en la GPU para medios y un mapa RSS;
3: Asignar una matriz CUDA 3D para una imagen hiperespectral;
4: Asociar una textura en capas 2D a una matriz 3D ;
5: Copiar la memoria de los medios de la CPU a la GPU;
6: Crear b flujos; ;
7: **for** $i = 1 \rightarrow b$ **do**
8: Copiar de forma asincrónica la memoria de la banda i-ésima de la imagen hiperespectral desde la CPU a la matriz 3D en el flujo i-ésimo;
9: Llamar de forma asíncrona el núcleo WOS utilizando la banda i de la hiperespectral imagen en el flujo i-ésimo;
10: **end for**
11: Sincronizar y destruir el flujo b;
12: Llamar el núcleo MDM utilizando el WOS clasificada mapa RSS;
13: Copiar la memoria del mapa RSS de GPU para la CPU;
14: Desenlazar la textura en capas 2D ;
15: Libera la matriz 3D;
16: Libera la memoria asignada en la GPU.

Fig. 4. Pseudocódigo de la aplicación WOS-MDM propuesto.

1: *% k denota el número de GPU %*
2: *% m denota el número de filas de la imagen hiperespectral %*
3: **for** $i = 1 \rightarrow k$ **do**
4: Crear un hilo de la CPU para procesar la i-ésima porción de m / k filas de la imagen hiperespectral en el i-ésimo GPU;
5: **end for**
6: El hilo principal del CPU procesa las últimas m / k filas de la imagen hiperespectral;
7: Sincronizar todos los hilos de la CPU.

Fig. 5. PPseudocódigo de la partición espacial usando multi-GPU.

todos los dispositivos disponibles. Con el fin de lograr tal distribución, es necesario realizar una partición de los datos. Regularmente, se manejan tres tipos diferentes de partición de datos: espectral, espacial y mixta. Para la implementación propuesta se seleccionó la partición espacial debido a que el método MDM requiere que todas las bandas espectrales estén disponibles para cada píxel. El algoritmo de la Fig. 5 muestra la partición espacial utilizada para la implementación multi-GPU. Se crea un hilo de CPU para cada GPU disponible. Cada hilo, en su respectivo GPU, procesa una parte de la imagen hiperespectral siguiendo el algoritmo de la Fig. 4. Tenga en cuenta que, en este estudio, se utiliza la biblioteca *pthread* para la creación y manipulación de hilos.

IV. Resultados

En esta sección, se presenta el análisis de la implementación y el rendimiento de la propuesta. Se evalúa la arquitectura basada en el cómputo multi-GPU. Estos resultados se analizan

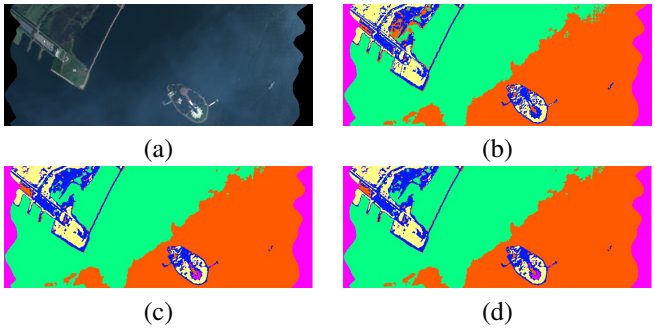

Fig. 6. Mapas RSS extraídos por el GPU con la clasificación WOS-MDM-difuso: (a) la composición en falso color del bajo Manhattan, (b) resultado ISODATA, (c) clasificación WOS-MDM-difuso, (d) las diferencias entre ISODATA y WOS-MDM-difuso.

desde el punto de vista de precisión de la clasificación hiperespectral y el tiempo consumido. La clasificación se compara con ISODATA [8] como referencia. Se utilizó MultiSpec [9] para obtener la clasificación ISODATA.

A. Datos de Imágenes Hiperespectrales de Gran Escala

Se presentan dos estudios de caso para demostrar la efectividad de la propuesta. A fin de que extraer mapas digitales de RSS desde escenas de imágenes hiperespectrales recogidas por el AVIRIS. El conjunto completo de datos seleccionado para ambos casos consiste en imágenes hiperespectrales de 271×689 píxeles, con 224 bandas espectrales. Se clasificaron cinco clases de RSS, que se refieren a una clasificación supervisada de 3 bits:

RSS en relación con las zonas de agua,
RSS relativa a las zonas de aguas profundas,
RSS relativa a las zonas de suelo,
RSS en relación con las zonas de vegetación,
RSS sin clasificación.

En el primer escenario, consideramos una imagen hiperespectral de Manhattan. La Fig. 6a muestra una composición en falso color de la imagen hiperespectral, utilizando las bandas de 664, 567 y 470 nm (representada como el rojo, verde y azul, respectivamente). Los resultados cualitativos del algoritmo ISODATA [8] y la clasificación WOS-MDM-difuso se muestran en las Figs. 6b y 6c, respectivamente. La Fig. 6d muestra las diferencias entre ambos resultados; las áreas negras son las coincidencias y los píxeles de colores representan las discrepancias que resultan del efecto de suavizado de la difusión anisotrópica difusa, pero se tiene un 85% de coincidencias entre ISODATA y la clasificación WOS-MDM-difuso.

En el segundo escenario de prueba, otra imagen hiperespectral de la zona de Manhattan es considerada. La Fig. 7a presenta una composición en falso color de la imagen hiperespectral. Figs. 7b y 7c, ilustran los algoritmos ISODATA y la clasificación WOS-MDM-difuso. La Fig. 7d muestra las diferencias entre ambos resultados.

B. Análisis de Rendimiento

En esta sección, se presenta el análisis de rendimiento en tiempo de la aplicación multi-GPU. La propuesta de

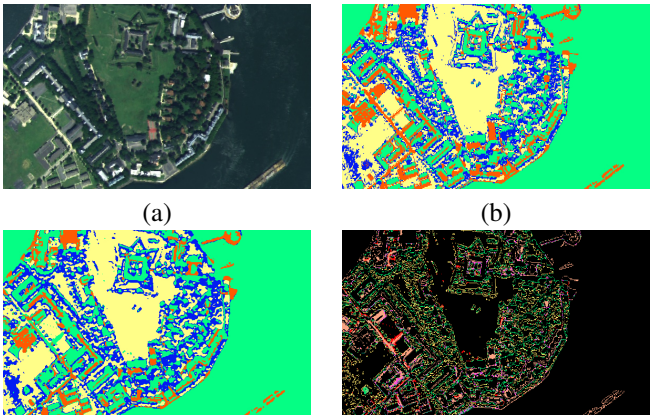

Fig. 7. Figura 5: Mapas RSS extraídos por el GPU con la clasificación WOS-MDM-difuso: (a) la composición en falso color del bajo Manhattan, (b) resultado ISODATA, (c) clasificación WOS-MDM-difuso, (d) las diferencias entre ISODATA y WOS-MDM-difuso.

TABLA I
TIEMPOS DE PROCESAMIENTO (EN SEGUNDOS) REGISTRADOS POR LAS IMPLEMENTACIONES DE MDM-WOS

Escena	Tipo	CPU	1 GPU	2 GPU
1	Multiespectral (3 bandas)	1.7526	0.0526	0.0317
	Hiperespectral (224 bandas)	66.4509	2.5935	1.3251
2	Multiespectral (3 bandas)	1.7272	0.0531	0.0324
	Hiperespectral (224 bandas)	65.7741	2.6206	1.3401

implementación multi-GPU de la clasificación difuso-WOS-MDM ha sido probada en una estación de trabajo (PC) con un procesador Intel Xeon E5603 QuadCore a 1.6 GHz y 24 GB de memoria RAM con dos GPU NVIDIA Tesla C2075, que cuentan con 448 núcleos CUDA a 1.15 GHz y un ancho de banda de 144 GB/s para la memoria. La Tabla I resume los resultados globales del tiempo utilizado con el enfoque propuesto. En el estudio comparativo, el mismo algoritmo se implementó de forma secuencial para CPU y usando la arquitectura multi-GPU. Se consideraron dos escenas de prueba, primero se procesaron 3 bandas (para tener un falso color), pero también se procesaron las 224 bandas de la imagen hiperespectral, vea Tabla I.

En particular, la implementación del algoritmo difuso-WOS-MDM utilizando la arquitectura multi-GPU, toma sólo 1.3401 segundos para la clasificación hiperespectral, en contraste con los 65.7741 segundos requeridos por la implementación serial en CPU, con una importante aceleración de 49.08 veces.

C. Discusión y Análisis de Diseños Alternativos

Con la precisión en la clasificación y el análisis de rendimiento previos; se presenta un análisis comparativo con diferentes sistemas y las estrategias algorítmicas en el contexto de la clasificación en tiempo real de imágenes de percepción remota. En [18], se hace una interesante revisión de diferentes plataformas de alto rendimiento (HPC) para aplicaciones de RS hiperespectrales. Particularmente, sus resultados experimentales demuestran que se usan ampliamente dispositivos como FPGA y GPU de bajo costo en muchas aplicaciones de RS hiperespectrales de tiempo (casi) real. En este respecto, la extracción de firmas de teledetección (RSS) utilizando el MDM, WOS se describe en [19]; sin embargo, solo se implementan para imágenes multiespectrales (sólo tres bandas RGB). Esta propuesta se implementa en una arquitectura paralela y está orientado para las imágenes hiperespectrales de gran escala de 689×271 píxeles con 224 bandas.

V. CONCLUSIONES

El principal resultado de este estudio se refiere al procesamiento de señales digitales para resolver un problema de clasificación en imágenes de percepción remota (imágenes de 689×271 píxeles, con 224 bandas espectrales) mediante la integración de la computación paralela con dispositivos GPU. En primer lugar, hemos establecido analíticamente el uso de técnicas difusas y la fusión de los estadísticos de orden ponderado (WOS) y la mínima distancia a la mediana (MDM), reduciendo el tiempo de procesamiento de las tareas de clasificación de imágenes de gran escala aplicando técnicas de computación en paralelo. Aquí, hemos deducido un esquema sobre la base de la combinación de multi-GPU y la CPU, en el que la difusión anisotrópica difusa y el algoritmo WOS-MDM se calcularon en paralelo en el CPU y dos GPU. En segundo lugar, hemos obtenido, a través de los resultados de los escenarios de prueba hiperespectrales, que con la propuesta de un esquema de co-diseño se puede cumplir los requisitos de procesamiento de imágenes en tiempo real.

REFERENCIAS

[1] M. T. Eismann and Society of Photo-optical Instrumentation Engineers., *Hyperspectral remote sensing*. SPIE, 2012.

[2] F. M. Henderson, A. J. Lewis *et al.*, *Principles and applications of imaging radar. Manual of remote sensing, volume 2.*, 3rd ed. John Wiley and sons, 1998.

[3] C.-I. Chang, *Hyperspectral data exploitation: theory and applications*. Wiley-Interscience, 2007.

[4] A. Plaza and C. I. Chang, *High performance computing in remote sensing*. Chapman & Hall/CRC, 2007.

[5] J. Lopez-Fandino, P. Quesada-Barriuso, D. B. Heras, and F. Arguello, "Efficient ELM-Based Techniques for the Classification of Hyperspectral Remote Sensing Images on Commodity GPUs," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2884–2893, jun 2015.

[6] Z. Wu, Q. Wang, A. Plaza, J. Li, L. Sun, and Z. Wei, "Parallel Spatial-Spectral Hyperspectral Image Classification With Sparse Representation and Markov Random Fields on GPUs," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2926–2938, jun 2015.

[7] A. E. Sahar, "Hyperspectral Image Classification Using Unsupervised Algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 4, 2016.

[8] N. Memarsadeghi, D. M. Mount, N. S. Netanyahu, and J. Le Moigne, "A fast implementation of the ISODATA clustering algorithm," *International Journal of Computational Geometry and Applications*, vol. 17, no. 01, pp. 71–103, 2007.

[9] D. Landgrebe and L. Biehl, "MultiSpec© | Home." [Online]. Available: https://engineering.purdue.edu/~biehl/MultiSpec/

[10] Y. Shkvarko, Perez-Meana, and A. Castillo Atoche, "Enhanced radar imaging in uncertain environment: a descriptive experiment design regularization paradigm," *International Journal of Navigation and Observation*, vol. 8, p. 11, 2008.

[11] A. Castillo Atoche, D. Torres Roman, and Y. Shkvarko, "Experiment design regularization-based hardware/software codesign for real-time enhanced imaging in uncertain remote sensing environment," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, p. 10, 2010.

[12] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 7, pp. 629–639, 1990.

[13] A. Castillo Atoche, R. Carrasco Alvarez, J. Ortegón Aguilar, and J. Vázquez Castillo, "A new tool for intelligent parallel processing of radar/SAR remotely sensed imagery," *Mathematical Problems in Engineering*, vol. 2013, pp. 2–10, Nov. 2013.
[14] J. Sanders and E. Kandrot, *CUDA by Example:An Introduction to General-Purpose GPU Programming*, 1st ed. Addison-Wesley Professional, 2011.
[15] NVIDIA, "CUDA C programming guide," 2011.
[16] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 3rd ed. Upper Saddle River, NJ: Pearson/Prentice Hall, 2008.
[17] NVIDIA, "NVIDIA performance primitives | NVIDIA developer zone," https://developer.nvidia.com/npp, [Online; accessed 2014-09-10].
[18] C. Lee, S. Gasster, A. Plaza, C.-I. Chang, and B. Huang, "Recent developments in high performance computing for remote sensing: A review," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 4, no. 3, pp. 508–527, 2011.
[19] I. E. Villalon-Turrubiates, "Remote sensing signatures extraction for hydrological resources management applications," in *Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on*. IEEE, 2009, pp. 567–570.

Alejandro Arturo Castillo Atoche obtuvo el grado Doctor en Ciencias (2010) y de Maestro en Ciencias (2002) por el Centro de Investigación y de Estudios Avanzados del I.P.N. en México, y es Ingeniero en Electrónica del Instituto Tecnológico de Mérida, México (2000). Actualmente es profesor en la Universidad Autónoma de Yucatán en el departamento de Mecatrónica. Sus áreas de interés son: aplicaciones para el procesamiento inteligente de imágenes/señales, sistemas en tiempo real, sistemas embebidos, sistemas multiprocesador en un chip (MPSoC), co-diseño HW/SW y cómputo paralelo con GPUs, entre otras.

Javier Vázquez Castillo es Doctor en Ciencias por el Centro de Investigación y de Estudios Avanzados del I.P.N. (2014) en México con la especialidad en telecomunicaciones, recibió el grado de Maestro en Ciencias por la misma institución en 2002 e Ingeniero en Electrónica con especialidad en sistemas digitales por el Instituto Tecnológico de Mérida, México (2000). Actualmente es profesor de la Universidad de Quintana Roo. Sus áreas de interés son: diseño en microelectrónica digital, tratamiento digital de señales, cómputo aritmético y diseño de dispositivos para telecomunicaciones.

Jaime Silverio Ortegón Aguilar es Doctor en Ciencias por el Centro de Investigación y de Estudios Avanzados del I.P.N. (2007) con la especialidad en Ingeniería Eléctrica, Maestro en Ciencias por el mismo centro (2002) en la especialidad en Ciencias de la Computación, Licenciado en Ciencias de la Computación por la Facultad de Matemáticas de la Universidad Autónoma de Yucatán, México (2000). Actualmente es profesor de la Universidad de Quintana Roo. Sus áreas de interés son: tratamiento digital de señales, visión por computadora y diseño de dispositivos para telecomunicaciones.

Roberto Carrasco Alvarez nació en la ciudad de México en 1981. Cursó sus estudios universitarios en el Instituto Tecnológico de Morelia obteniendo el título en ingeniero electrónico en 2004. Llevo a cabo sus estudios de maestría y doctorado en el Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV), en el área de telecomunicaciones, obteniendo los grados en los años 2006 y 2010 respectivamente. Actualmente es profesor investigador en el Centro Universitario de Ciencias Exactas e Ingenierías de la Universidad de Guadalajara. Sus líneas de investigación incluyen el procesamiento digital de señales y las comunicaciones digitales.

Jaime Francisco Avilés Viñas es Profesor Investigador del Departamento de Mecatrónica de la Facultad de Ingeniería de la Universidad Autónoma de Yucatán. Estudios en Ingeniería Mecánica (Instituto Tecnológico de Mérida), M.C. en Ingeniería Mecatrónica (Cenidet) y Doctorado en Robótica y Manufactura Avanzada (Cinvestav IPN Saltillo).