

## CONSTRUCCIÓN DE UN ALGORITMO EN PARALELO PARA LA TRANSFORMADA RÁPIDA DE FOURIER

MARIO A. HIGUERA G., (\*)  
HUMBERTO SARRIA, (\*\*)  
DIANA FONSECA, (\*\*)  
JOHN IDARRAGA (\*\*\*)

---

RESUMEN. Presentamos un código para el cálculo en paralelo de la Transformada Rápida de Fourier (FFT), sobre un cluster de  $2^n$  computadores. El código ha sido utilizado para obtener la transformada de una imagen astronómica, a la cual se le aplicó un filtro, para eliminar cierto tipo de frecuencias. Luego, se recupera una imagen filtrada, haciendo uso de la transformada inversa de Fourier.

PALABRAS CLAVE: Transformada Rápida de Fourier, message passing interface (MPI), procesamiento en paralelo.

### 1. INTRODUCCIÓN

En los últimos años se han estado implementando redes de computadores de bajo costo, para el procesamiento en paralelo. La mayoría de las implementaciones denominadas “Clusters” (grupo de computadores), se han desarrollado en estaciones de trabajo bajo entornos de Unix (ver [1]); sin embargo, hoy en día, se

---

(\*) Mario Higuera, Observatorio Astronómico Nacional,  
(\*\*) Humberto Sarria, Departamento de Matemáticas,  
Diana Fonseca, estudiante, Departamento de Matemáticas,  
(\*\*\*) John Idarraga, estudiante de maestría, Departamento de Física,  
Facultad de Ciencias. Universidad Nacional de Colombia.  
e-mail: ahiguera@ciencias.unal.edu.co, hsarria@ciencias.unal.edu.co,  
dfonseca@ciencias.unal.edu.co, idarraga@ciencias.unal.edu.co  
Apartado Aéreo 2584. Bogotá, Colombia.

construyen redes que incluyen procesadores G3-G4 de Motorola/IBM/Apple en computadores PowerMacintosh, administradas bajo el sistema operativo MacOS X, (ver [2]), también es común, el uso de procesadores Pentium y AMD bajo los sistemas operativos Linux o Windows.[1]

El Observatorio Astronómico Nacional, ha implementado un cluster, motivado por los desarrollos alcanzados por Decyk y sus colaboradores (ver [3]), con el fin de estudiar el espectro lumínico de algunas imágenes astronómicas. Uno de los métodos usados más rápido y efectivo para el análisis masivo de información, pasa por el cálculo de la FFT de la imagen. Este procedimiento permite identificar rápidamente, el espectro de una imagen.

Aplicando algunos filtros a este espectro y usando luego la inversa de FFT, es posible enfatizar algunas características de la imagen.

## 2. TRANSFORMADA RÁPIDA DE FOURIER

Sea  $f : \mathbb{R} \rightarrow \mathbb{C}$  una función continua, donde  $\mathbb{R}$  representa el conjunto de los números reales y  $\mathbb{C}$  el conjunto de los números complejos. La *Transformada de Fourier* de  $f$ , está dada por la ecuación

$$(1) \quad H(\nu) = \int_{-\infty}^{\infty} f(t)e^{-2\pi i\nu t} dt,$$

donde  $i$  denota la unidad imaginaria y  $\nu$  la frecuencia [7]. En la mayoría de las situaciones prácticas, la función  $f$  está dada en forma discreta como una colección finita de valores  $f(x_0), f(x_1), \dots, f(x_{N-1})$  con  $N \in \mathbb{N}$ , en donde  $\mathbb{N}$  es el conjunto de los números naturales y  $\{x_0, x_1, \dots, x_{N-1}\}$  es una partición uniformemente distribuida sobre un intervalo real  $[a, b]$ , con  $x_k = a + \frac{b-a}{N}k$ ,  $0 \leq k \leq N-1$ .

En problemas que implican cálculo numérico, en lugar de la ecuación (1) se usa la suma,

$$(2) \quad H(k) = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j)e^{-2\pi i k j / N},$$

denominada *Transformada Discreta de Fourier* (DFT)<sup>1</sup> de  $f$  sobre el intervalo  $[a, b]$ . Si  $f$  es una función definida sobre el intervalo  $[0, 2\pi]$  de valor real con periodo  $2\pi$ , los valores  $H(k)$ , para  $k = 0, \dots, N-1$ , pueden interpretarse como los coeficientes  $c_k$ ,

---

<sup>1</sup> Discrete Fourier Transform.

$$(3) \quad c_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) e^{-i \frac{2\pi k}{N} j}, \quad 0 \leq k \leq N-1$$

de un *polinomio exponencial*,

$$(4) \quad P(x) = \sum_{k=0}^{N-1} c_k e^{ikx},$$

donde  $x \in \mathbb{R}$ ,  $c_k \in \mathbb{C}$ , y  $k = 0, 1, \dots, N-1$ , el cual permite interpolar a  $f$  en los valores  $f(x_0), f(x_1), \dots, f(x_{N-1})$ .

La transformada discreta de Fourier de  $f$  sobre la partición  $\{x_0, x_1, \dots, x_{N-1}\}$ , se define como el operador

$$DFT : \mathbb{C}^N \longrightarrow \mathbb{C}^N$$

tal que

$$[c_0, c_1, \dots, c_{N-1}]^T = DFT\left([f(x_0), f(x_1), \dots, f(x_{N-1})]^T\right).$$

El algoritmo que calcula la DFT se denomina *Transformada Rápida de Fourier* (FFT)<sup>2</sup> y reduce significativamente el tiempo que le toma a un algoritmo directo calcular los coeficientes  $[c_0, c_1, \dots, c_{N-1}]$ .

La FFT, también puede ser usada para evaluar eficientemente el polinomio  $P(x)$ . El número de operaciones efectuadas por un algoritmo directo para el cálculo de  $P(x)$  es de orden  $N^2$ , mientras que FFT efectúa  $N \text{Log}_2 N$  operaciones para obtener los mismos resultados. Véase el cuadro 1, con el fin de comparar estas dos cantidades para valores grandes de  $N$ .

$N$	$N^2$	$N \text{Log}_2 N$
2048	4194304	22528
8192	67108864	106496
32768	1073741824	491520

CUADRO 1. Comparación para valores grandes de  $N$ .

La FFT basa su potencia de cálculo en las siguientes fórmulas recursivas

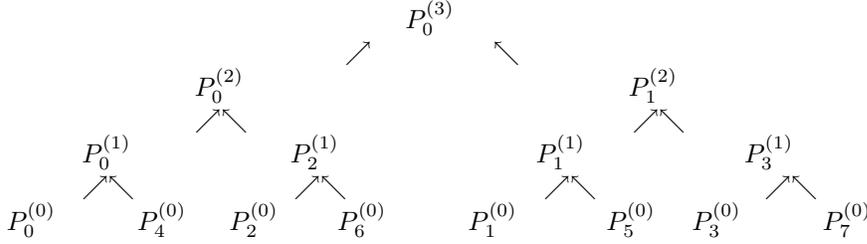
$$(5) \quad P_k^{(n+1)}(x) = \frac{1}{2}(1 + e^{i2^n x})P_k^{(n)}(x) + \frac{1}{2}(1 - e^{i2^n x})P_{k+2^m-n-1}^{(n)}\left(x - \frac{\pi}{2^n}\right),$$

<sup>2</sup> Fast Fourier Transform.

en donde  $P_k^{(0)}(x_k) = f(x_k)$  para  $k = 0, 1, \dots, 7, \dots$ ,  $0 \leq n < m$  y  $0 \leq k \leq 2^{m-n-1} - 1$ . Puede probarse que

$$P(x) = P_0^{(\log_2 N)}(x).$$

El siguiente grafo piramidal, muestra cómo se efectúa el cálculo de los coeficientes de un polinomio exponencial, que interpola los ocho valores:  $f(x_0), f(x_1), \dots, f(x_7)$ . ( $2^m = N = 8, m = 3$ ).



El cálculo se inicia en el **nivel** inferior del grafo (indicado por el superíndice de los polinomios), este proceso se realiza de manera recursiva: Para  $n = 0$ , cada polinomio  $P_k^{(1)}$ ,  $0 \leq k \leq 3$ , relaciona dos datos (según la ecuación (5)) que provienen del nivel cero. Nótese que  $P_k^{(1)}(x_{k+2j}) = f(x_{k+2j})$ ,  $0 \leq k \leq 3$  y  $j = 0, 2$ .

Para  $n = 1$ , los polinomios  $P_k^{(2)}$ ,  $0 \leq k \leq 1$ , se originan combinando los resultados obtenidos de sus ramas inmediatamente inferiores, de esta manera puede observarse que  $P_k^{(2)}(x_{k+2j}) = f(x_{k+2j})$ , para  $0 \leq k \leq 1$  y  $j = 0, 1, 2, 3$ .

Finalmente, para  $n = 2$ , se calcula el polinomio  $P_0^{(3)}(x)$ .

Vale la pena aclarar que el algoritmo se encarga de hallar sólo los coeficientes de estos polinomios. Si  $u$  es un coeficiente del polinomio  $P_k^{(n)}(x)$  y  $v$  es el coeficiente correspondiente en el polinomio  $P_{k+2^{m-n-1}}^{(n)}(x - \frac{\pi}{2^n})$ , para algún  $n$  y algún  $k$  fijos ( $0 \leq n < m$ ,  $0 \leq k \leq 2^{m-n-1} - 1$ ); entonces por la ecuación (5), los coeficientes de  $P_k^{(n+1)}(x)$  vienen dados por  $\frac{u + ve^{-\frac{i\pi j}{2^n}}}{2}$ ,  $\frac{u - ve^{-\frac{i\pi j}{2^n}}}{2}$ ,  $j = 0, \dots, 2^n - 1$ .

El código en paralelo trabaja con  $2^l$  procesadores,  $l \geq 0$  y hace uso del grafo piramidal mostrado anteriormente; distribuyendo inicialmente, todos los procesadores en un nivel específico del árbol, de tal modo que cada uno efectúa los cálculos de todas las ramas que llegan a su nodo. En la siguiente etapa del cálculo, se pasa al nivel inmediatamente superior y se dejan trabajando sólo  $2^{l-1}$  procesadores. Los restantes  $2^{l-1}$  procesadores transfieren la información

obtenida, a los procesadores que continúan trabajando. Este proceso se efectúa sucesivamente, para cada etapa, hasta llegar al nivel superior del grafo, en donde sólo se usa un único procesador para efectuar los cálculos finales.

Para una mejor comprensión del proceso, se desarrollará un ejemplo, que usa cuatro procesadores para calcular un polinomio exponencial que interpola ocho datos, que serán guardados inicialmente en el vector  $C = [C(0), \dots, C(7)]^T$ .

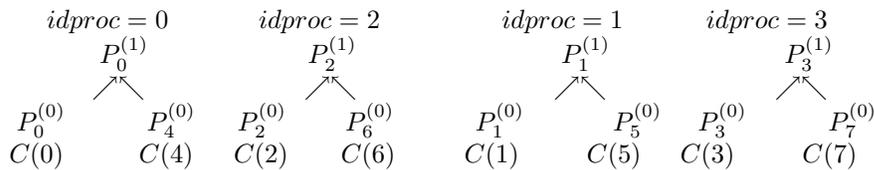
Las variables básicas que se usan dentro del código, se presentan en el Cuadro 2.

<i>Variable</i>	<i>Descripción</i>
$nproc$	Número de procesadores.
$idproc$	Identificación correspondiente a cada procesador. Se cumple que $0 \leq idproc < nproc$ , el procesador 0 representa al procesador MAESTRO.
$N$	número de datos.
$C$	Vector donde se guarda los datos de entrada y de salida.
$Z$	Vector que contiene los factores $e^{-\frac{2\pi i}{N}j}$ , con $0 \leq j \leq N - 1$ .
$m$	Número de niveles del árbol calculado como $\left\lceil \frac{\text{Log}(N)}{\text{Log}(2)} \right\rceil$ .
$n$	Número de nivel en el árbol para cada paso del proceso, $0 \leq n < m$ .
$m_1$	Número del nivel en el árbol en donde actúan todos los procesadores, viene dado por la fórmula: $m - \left\lceil \frac{\text{Log}(nproc)}{\text{Log}(2)} \right\rceil$ .
$nproc\_sup$	Número de procesadores que operan en algunos niveles.

CUADRO 2. Variables usadas en el algoritmo.

Entonces, para este ejemplo tenemos que  $nproc = 4$ ,  $N = 8$ ,  $m = 3$ ,  $m_1 = 1$ .

Cada procesador trabaja con su rama respectiva, es decir:



El procesador identificado con la etiqueta  $idproc$ , genera el vector de coeficientes de  $P_{idproc}^{(n)}$  para  $0 \leq idproc < nproc = 4$ . La parada del algoritmo se realiza cuando  $n = m_1 = 1$ . En este punto, el procesador  $idproc = 2$  transmite su vector de datos al procesador  $idproc = 0$ , y el procesador  $idproc = 3$  transfiere su vector de datos al procesador  $idproc = 1$ . Antes de efectuar la transferencia, cada procesador debe almacenar la información a transferir en una matriz de dos columnas, en donde se guarda la parte real en la primera columna y la parte imaginaria en la segunda columna. Esto es necesario ya que el comando de envío no reconoce un tipo de datos complejos en lenguaje C.

En este momento los procesadores,  $idproc = 2$  e  $idproc = 3$ , salen del proceso. Además,  $nproc\_sup = \frac{nproc}{2} = 2$ , es decir, en el nivel  $n = 2$  el número de procesadores se reduce a la mitad. De aquí en adelante, cada vez que se avance de nivel en el grafo, la mitad de los procesadores dejan de trabajar y la variable  $nproc\_sup$  desciende a la mitad.

Los procesadores que reciben la información, la guardan en el vector  $C$ , cuidando no sobrescribir sobre los datos ya calculados.



En el nivel  $n = 2$  del proceso,  $idproc$  produce el vector de coeficientes del polinomio  $P_{idproc}^{(n)}$ , para  $0 \leq idproc < nproc\_sup = 2$ . La acción anterior se repite mientras  $nproc\_sup \geq 1$ , es decir mientras exista al menos un procesador efectuando cálculos. Por consiguiente, el procesador identificado con  $idproc = 0$  recibe la comunicación de  $idproc = 1$ , el procesador MAESTRO efectúa el último cálculo que determina los coeficientes del polinomio  $P_{idproc}^{(n)}$ . En este instante ( $idproc = 0$ ,  $n = m = 3$ ) y  $nproc\_sup = \frac{1}{2}$ , en consecuencia el proceso termina.

## 3. LA TRANSFORMADA INVERSA

Se puede afirmar que el polinomio exponencial dado en (4) calcula la Transformada Discreta Inversa (IDFT)<sup>3</sup>, la cual se define como el operador

$$IDFT : \mathbb{C}^N \longrightarrow \mathbb{C}^N$$

tal que

$$[f(x_0), f(x_1), \dots, f(x_{N-1})]^T = IDFT([c_0, c_1, \dots, c_{N-1}]^T).$$

Se utiliza la FFT para encontrar la IFFT<sup>4</sup>. Definiendo  $x_k = \frac{2\pi k}{N}$  para

$0 \leq k \leq N-1$ , se tiene que al evaluar  $p$  en los puntos  $2\pi - x_k = 2\pi - \frac{2\pi k}{N}$

$$\begin{aligned} p(2\pi - x_k) &= \sum_{j=0}^{N-1} c_j e^{ij(2\pi - x_k)} = \sum_{j=0}^{N-1} c_j e^{ij2\pi} c_j e^{-ijx_k} = \sum_{j=0}^{N-1} c_j e^{-ijx_k} \\ (6) \quad &= N \left( \frac{1}{N} \sum_{j=0}^{N-1} c_j e^{-ijx_k} \right). \end{aligned}$$

De este modo se obtiene la IFFT hallando la FFT multiplicada por  $N$  y

$$p(2\pi - x_k) = p\left(2\pi - \frac{2\pi k}{N}\right) = p\left(2\pi \frac{N-k}{N}\right),$$

así la transformada inversa resulta en el orden contrario.

## 4. CÁLCULO DE LA COMPLEJIDAD COMPUTACIONAL DE LA FFT EN PARALELO

Si el número de datos es  $N = 2^m$ , ( $m \geq 1$ ) y el número de procesadores es  $2^l$  ( $l \geq 0$ ), el orden de multiplicaciones (complejas) efectuadas hasta el nivel  $m_1$  del árbol, corresponde a las multiplicaciones que hace cada procesador con el fin de calcular los coeficientes de los polinomios de interpolación del subárbol correspondiente. El número de multiplicaciones llevadas a cabo hasta el nivel  $n$  del árbol, es  $n2^m$  para  $0 \leq n \leq m$ . En particular, cuando  $n = m_1$  se tiene que  $O(m_1(2^m))$ . Como son  $2^l$  procesadores, la complejidad desciende a

$$\frac{m_1(2^m)}{2^l} = (m_1)2^{m-l} = m_1(2^{m_1}),$$

<sup>3</sup> Inverse Discrete Fourier Transform.

<sup>4</sup> Inverse Fast Fourier Transform

para  $n = m_1$ .

Únicamente hace falta sumar el número de multiplicaciones que se realizan en los niveles  $n$ , para  $m_1 < n \leq m$ . En cada nivel este número está dado por  $2^m$ . Teniendo en cuenta que para cada  $n$ ,  $m_1 < n \leq m$ , el número de procesadores se reduce a la mitad, entonces se requieren

$$\sum_{i=1}^l 2^{m_1+i}$$

multiplicaciones complejas a partir del nivel  $m_1 + 1$  hasta cuando  $n = m$ .

Concretando, el orden de complejidad es

$$O\left((m-l)2^{m-l} + \sum_{i=1}^l 2^{m-l+i}\right).$$

Calculando la razón entre el orden de multiplicaciones del algoritmo secuencial y el del algoritmo en paralelo se observa que

$$\begin{aligned} \frac{N \text{Log}_2 N}{(m-l)2^{m-l} + \sum_{i=1}^l 2^{m-l+i}} &= \frac{m2^m}{(m-l)2^{m-l} + \sum_{i=1}^l 2^{m-l+i}} \\ &= \frac{m}{(m-l)2^{-l} + \sum_{i=1}^l 2^{-l+i}} \end{aligned}$$

lo que indica que el algoritmo en paralelo es  $\frac{m}{(m-l)2^{-l} + \sum_{i=1}^l 2^{-l+i}}$  veces más rápido que el secuencial. En el Cuadro 3 se muestra el número de multiplicaciones requeridas en el desarrollo de la FFT con cuatro procesadores. Es importante mencionar que en este análisis no se estima el tiempo utilizado en el proceso de comunicación.

$N$	$N^2$	$N \text{Log}_2 N$	$PfP$	$RSP \text{ FFT}$
512	262144	4608	1664	2.77
2048	4194304	22528	7680	2.93
8192	67108864	106496	34816	3.06
32768	1073741824	491520	155648	3.16

CUADRO 3.  $PfP$ : Multiplicaciones realizadas en paralelo con cuatro procesadores;  $RSP \text{ FFT}$ : Razón de las multiplicaciones entre la FFT secuencial y paralela.

## 5. LA TRANSFORMADA DISCRETA BIDIMENSIONAL 2D-DFT

Para dos variables el muestreo se hace en el plano  $xy$  donde los valores están igualmente espaciados sobre las rectas paralelas al eje  $x$  y las rectas paralelas al eje  $y$ . Se define a  $\mathcal{M}_{N \times M}(\mathbb{C})$  como el conjunto de matrices de tamaño  $N \times M$  cuyas entradas son números complejos. Sea  $f(x, y) : \mathbb{R}^2 \rightarrow \mathbb{C}$ . La transformada discreta de Fourier bidimensional de  $f(x, y)$  para  $0 \leq x \leq N-1$ ,  $0 \leq y \leq M-1$ , se define como el operador

$$2D - DFT : \mathcal{M}_{N \times M}(\mathbb{C}) \longrightarrow \mathcal{M}_{N \times M}(\mathbb{C})$$

$$\left( f(x, y) \right)_{N \times M} \mapsto \left( F(u, v) \right)_{N \times M},$$

donde

$$(7) \quad F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x, y) e^{-i2\pi \left( \frac{ux}{N} + \frac{vy}{M} \right)}$$

para  $u = 0, 1, \dots, N-1$ ,  $v = 0, 1, \dots, M-1$ . Y se establece la 2D-IDFT (Transformada Inversa de Fourier bidimensional) como

$$2D - IDFT : \mathcal{M}_{N \times M}(\mathbb{C}) \longrightarrow \mathcal{M}_{N \times M}(\mathbb{C})$$

$$\left( F(u, v) \right)_{N \times M} \mapsto \left( f(x, y) \right)_{N \times M},$$

donde

$$(8) \quad f(x, y) = \frac{1}{M} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F(u, v) e^{i2\pi \left( \frac{ux}{N} + \frac{vy}{M} \right)}$$

para  $x = 0, 1, \dots, N-1$ ,  $y = 0, 1, \dots, M-1$ .

La ecuación (7) puede expresarse como

$$(9) \quad F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x, y) e^{-i2\pi \frac{ux}{N}} e^{-i2\pi \frac{vy}{M}} = \frac{1}{N} \sum_{x=0}^{N-1} F(x, v) e^{-i2\pi \frac{ux}{N}},$$

donde  $F(x, v) = M \left( \frac{1}{M} \sum_{y=0}^{M-1} f(x, y) e^{-i2\pi \frac{vy}{M}} \right)$ . De esta manera se puede usar sucesivamente la DFT unidimensional sobre filas y columnas de la matriz.

La implementación de la 2D-FFT (Transformada Rápida de Fourier bidimensional) es la siguiente,

FFT sobre filas  $\rightarrow$  Multiplicación por  $M \rightarrow$  FFT sobre columnas [10], donde  $N$  y  $M$  son potencias de dos.

## 6. APLICACIÓN

Una aplicación de uso frecuente está dirigida hacia el procesamiento de imágenes. Matemáticamente, una imagen puede representarse mediante una función

$$\begin{aligned} f : \mathbb{R}^2 &\longrightarrow Z \\ (x, y) &\longmapsto z = f(x, y), \end{aligned}$$

donde  $(x, y)$  representa las coordenadas de un punto de la imagen con intensidad de luz  $f(x, y)$  y  $Z$  es el conjunto de los números enteros. Una imagen digital es una imagen discretizada en  $x$ ,  $y$  y  $f(x, y)$ , es decir  $x, y, f(x, y) \in \mathbb{Z}^+ \cup \{0\}$ . Se puede representar una imagen digital por una matriz  $(P_{x,y})_{N \times M}$  cuyos elementos son los puntos o pixels (del inglés picture elements) sobre la pantalla de un monitor.

El algoritmo llena con ceros sobre filas y columnas para completar la próxima potencia de 2.

La Transformada de Fourier se representa por medio de su espectro

$$(10) \quad \sqrt{Re[C(x, y)]^2 + Im[C(x, y)]^2},$$

donde  $Re[C(x, y)]$  es la parte real e  $Im[C(x, y)]$  es la parte imaginaria del elemento  $(x, y)$  de la Transformada de la imagen.

Un filtro de imágenes es un operador  $H : \mathbb{Z}^2 \longrightarrow R$  que permite modificar los valores de intensidad lumínica de cierta imagen. Por el Teorema de Convolución se puede aplicar un filtro y hacer el producto del mismo con la parte real e imaginaria de la Transformada. La tarea de un filtro es variada, puede ser tanto introducir como eliminar ruido de una imagen, esto depende del objetivo que se quiera. En el algoritmo el filtro que se implementó tiene la siguiente forma:

$$(11) \quad F(x, y) = \begin{cases} 0 & \text{si } \sqrt{(x - c_1)^2 + (y - c_2)^2} \leq b \\ 1 & \text{en caso contrario,} \end{cases}$$

donde  $(c_1, c_2)$  es el centro de la imagen y  $b$  es una constante positiva, [10].

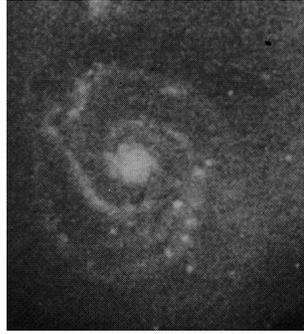


FIGURA 1. Imagen galáctica original

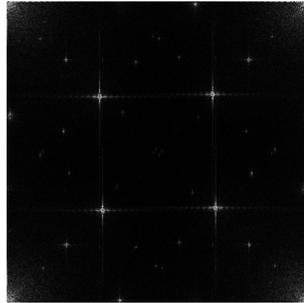


FIGURA 2. Espectro de Fourier

Para terminar el proceso, se calcula la 2D-IFFT a este producto y se obtiene la imagen filtrada. En la figura 3 se observa el efecto de suavizado que se logró con este filtro sobre la imagen de la galaxia en espiral NGC5194 (Galaxia Whirlpool) localizada en la constelación “Los perros de Caza”. La imagen tiene un tamaño de  $460 \times 506$  pixels, fue tomada de [11] y el parámetro de filtro es  $b = 340$ .

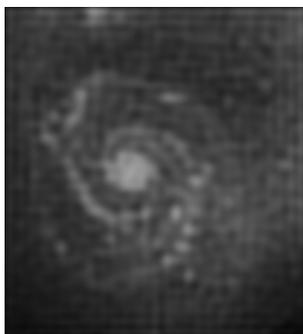


FIGURA 3. Imagen filtrada

#### BIBLIOGRAFÍA

- [1] <http://www.topclusters.org/>
- [2] <http://exodus.physics.ucla.edu/appleseed/appleseed.html>
- [3] Decyk Viktor K., Dager Dean E. and Kokelaar Pieter R. *How to build an AppleSeed: A Parallel Macintosh Cluster for Numerically Intensive Computing. Presentada en la Conferencia Internacional sobre Simulación de Plasma en Banff, Alberta, Canadá en mayo de 2000. phys.* **79B**, 239 (1978).
- [4] <http://www.absoft.com/>
- [5] <http://exodus.physics.ucla.edu/appleseed/dev/Developer.html>
- [6] <http://www.observatorio.unal.edu.co/invest/cluster/index.html>
- [7] William H. Press, Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling. *Numerical Recipes in C. The Art of Scientific Computing.* Cambridge University Press, 1987.
- [8] D. Kincaid, W. Cheney. *Análisis Numérico.* Adison Wesley Iberoamericana. 421-434.
- [9] Fonseca Diana. *Construcción y análisis de un algoritmo en paralelo para la Transformada Rápida de Fourier y una aplicación al procesamiento de imágenes.* Tesis de grado. Departamento de Matemáticas, Universidad Nacional de Colombia, Bogotá, 2000.
- [10] Rafael C. González, Richard E. Woods. *Tratamiento digital de imágenes.* Adison Wesley, Iberoamericana, 1996.
- [11] Robert A. Schowengerdt. *Techniques for Image Processing and Classification in Remote Sensing.* Academic Press, 1983.

AGRADECIMIENTOS. Deseamos expresar nuestro más sincero agradecimiento al profesor Viktor K. Decyk, (Universidad de los Angeles en California, UCLA), por su apoyo y útiles sugerencias; y a la División de Investigación de la Sede de Bogotá, Universidad Nacional de Colombia por el apoyo financiero, código DIB 803577.