# ANTEPROYECTO FIN DE CARRERA

Alberto Monescillo Álvarez

2 de Abril de 2009

**TÍTULO:** Paralelización de algoritmos para el procesado de imágenes de teledetección

**DEPARTAMENTO:** Electrónica

**AUTOR:** Alberto Monescillo Álvarez

**DIRECTOR:** Felipe Bertrand Galarza

**TUTOR:** Javier Macías Guarasa

## 1. Introducción

La teledetección es la observación y medida de objetos sobre la superficie terrestre sin estar en contacto con ellos. Se basa en la detección y registro de la energía reflejada o emitida desde un objeto, y el procesado, análisis y aplicación de esa información [16]. El dispositivo que detecta esta radiación electromagnética se llama sensor remoto o simplemente sensor. El uso actual de la teledetección abarca una gran cantidad de campos, como los ámbitos de la cartografía, agricultura, conservación de bosques, meteorología, climatología, militar, etc.

La agencia espacial europea (ESA), ha desarrollado diferentes misiones basadas en la teledetección, que permiten realizar una observación de la superficie terrestre. Entre ellas cabe destacar Meteosat, ERS y ENVISAT, que proporcionan imágenes de nuestro planeta [3]. En función del tiempo que se tarde en procesar los datos capturados por el sensor y generar las correspondientes imágenes, la ESA clasifica los servicios en:

- Near Real Time (NRT)
- 48 hours
- 1 month

Es por ello que hoy en día uno de los grandes objetivos en el ámbito de la teledetección es implementar algoritmos que sean capaces de reducir el tiempo necesario por la cadena de procesado para generar las imágenes, y de esta forma aproximarse cada vez más a un sistema de tiempo real.

Con la tecnología actual existe un amplio abanico de sensores que permiten realizar una observación de la tierra [11]. Los más utilizados son:

- Instrumentos ópticos de alta resolución
- Radar de Apertura Sintética (SAR)
- Instrumentos Radiométricos Pasivos

Una vez que los datos han sido capturados por el sensor, estos deben pasar por una cadena de procesado, a través de la cual se van aplicando una serie de algoritmos para corregir posibles errores y obtener imágenes de mayor calidad. Las principales etapas que intervienen en esta cadena de procesado son [13]:

- Ecualización
- Reducción de ruido
- Calibración absoluta
- Corregistración
- Ortorectificación

Cuando los datos han pasado por todas estas etapas y se ha generado la imagen, ésta puede ser utilizada por diversas aplicaciones en función del fin que se persiga. En nuestro caso, el usuario que va a hacer uso de estas imágenes será la plataforma del proyecto España Virtual [10].

Para mejorar los tiempos empleados por la cadena de procesado existen diferentes alternativas. Una de las más destacadas consiste en realizar una paralelización de la aplicación [14], dentro de la cual existen varias opciones:

- Emplear las GPUs de las tarjetas gráficas.
- Emplear procesadores de múltiples núcleos.
- Emplear clusters.

En este proyecto nos vamos a centrar en desarrollar las dos primeras técnicas de paralelización.

## 2. Objetivos

El objetivo principal de este proyecto es optimizar una serie de algoritmos de procesado de imágenes de teledetección mediante técnicas de paralelización [11], [14], con el fin de obtener un sistema de baja latencia, es decir, capaz de procesar una gran cantidad de datos a alta velocidad, y por lo tanto poder generar las imágenes en un tiempo reducido.

Para ello se partirá de los algoritmos implementados en el proyecto AIPC (Autonomous Image Processing Chain) de Deimos Space, que implementa todas las etapas necesarias para procesar los datos capturados por el sensor. Para mejorar las prestaciones de estos

algoritmos se van a implementar dos técnicas de paralelización, que consisten en modificar el código original de los algoritmos para poder optimizar la aplicación.

Por lo tanto se puede decir que el objetivo principal de este proyecto se apoya en dos elementos tecnológicos:

- Utilizar la tecnología CUDA de las tarjetas gráficas NVidia para realizar la paralelización.
- Utilizar el estándar OpenMP, que permite realizar la paralelización empleando procesadores de múltiples núcleos.

## 3. Planteamiento general del trabajo

Como se describió en el apartado 1, existen diferentes tipos de sensores que permiten capturar los datos para poder generar las imágenes. Las características mas importantes de los más utilizados son:

- Instrumentos ópticos de alta resolución: Se caracterizan por su resolución y la banda de frecuencias en que trabajan. La resolución determina el detalle de la imagen, mientras que la banda de frecuencias determina los fenómenos físicos que va a capturar. Las bandas de frecuencias que se suelen utilizar son: azul, verde, roja, infrarrojo cercano o infrarrojo de onda corta.
- Radar de Apertura Sintética (SAR): Está basado en tecnología RADAR. Es, por lo tanto, un instrumento que usa ondas de radio para estimar la distancia y posición de objetos remotos. La peculiaridad del SAR es que, en lugar de usar una gran antena con muy buena direccionalidad, y por lo tanto muy buena resolución, usa una antena pequeña con mala direccionalidad, pero obtiene muestras desde muchas posiciones distintas, consiguiendo una mayor resolución.
- Instrumentos Radiométricos Pasivos: Suelen operar en la banda de las microondas, ya que a estas frecuencias se producen muchos fenómenos y propiedades físicas de interés. La diferencia fundamental con los instrumentos ópticos es su resolución, y que la detección debe realizarse con antenas.

El sensor seleccionado para desarrollar este proyecto será un sensor óptico.

Los datos capturados por este sensor serán procesados a través de diferentes etapas, en las que se aplican distintos algoritmos [4], [13]. Como se comentó en el apartado 2, estos algoritmos ya han sido implementados en el proyecto AIPC de Deimos Space.

El esquema de la cadena de procesado es el mostrado en la figura.

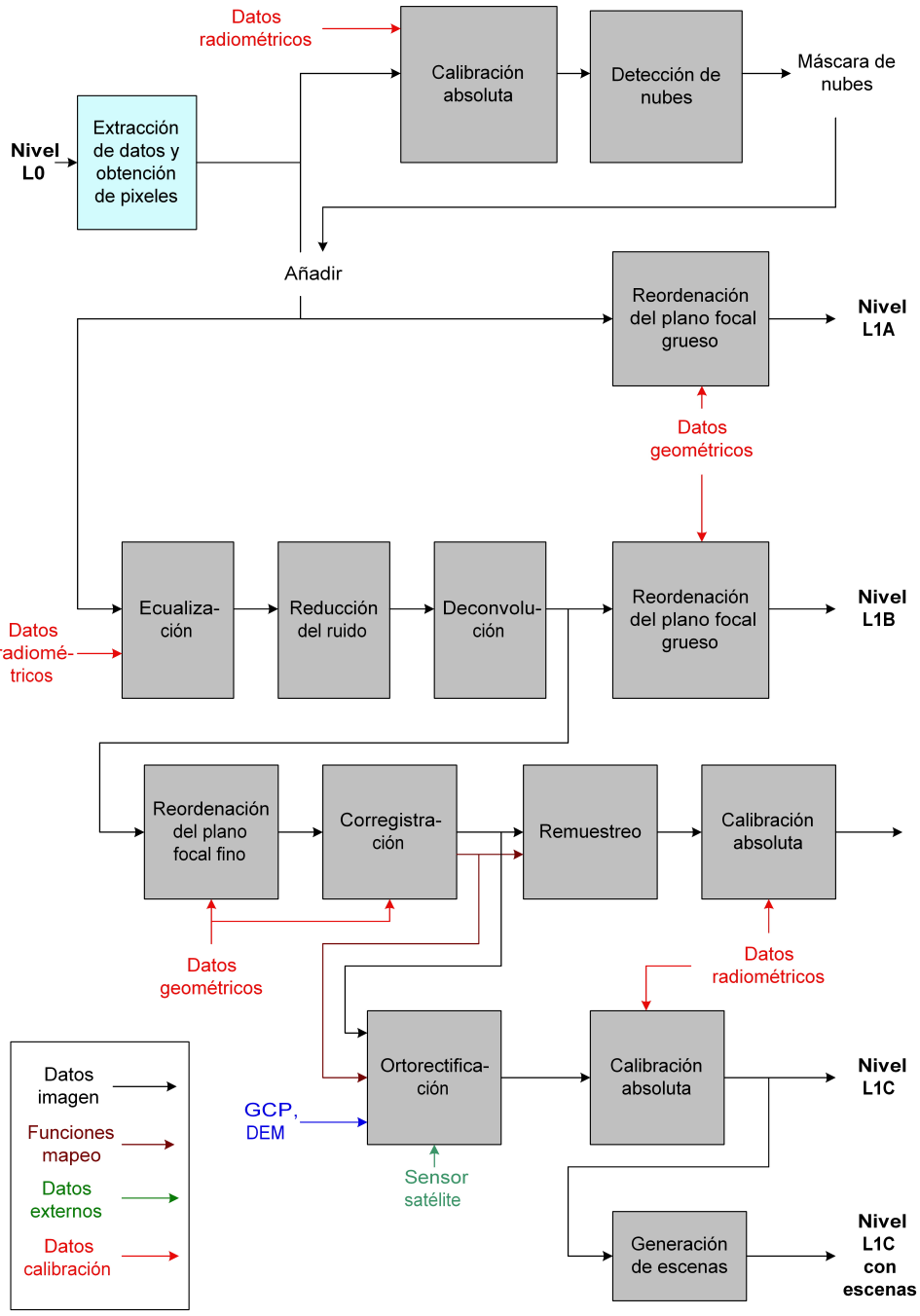


**Figura 1:** Cadena de procesado de imágenes

Como se puede observar, las diferentes etapas se agrupan en niveles, donde cada uno engloba una serie de funcionalidades:

- Nivel L0: Imágenes en crudo capturadas por el sensor, ordenadas cronológicamente. Estos datos incluyen marcas de tiempo.
- Nivel L1A: Imágenes del nivel anterior a las que se añade información adicional acerca

de la posición y orientación del satélite, que serán usadas para aplicar correcciones.

- Nivel L1B: Datos ecualizados y filtrados para eliminar efectos no deseados (ruidos).
- Nivel L1C: Imágenes ortorectificadas y remuestreadas [12], [9].

Estos niveles son básicos, ya que son necesarios en casi todo tipo de aplicaciones basadas en procesado de imágenes, independientemente del fin para el cual haya sido diseñada la aplicación.

Como se mencionó en el apartado 1, existen diferentes técnicas de paralelización que permiten mejorar las prestaciones proporcionadas por los algoritmos utilizados a lo largo de la cadena de procesado.

- Usar las GPUs de las tarjetas gráficas. En este sector hay tecnología tanto de NVidia como de ATI. Se basa en el uso de librerías y compiladores específicos proporcionados por los fabricantes, mediante los cuales los programadores de aplicaciones pueden utilizar los cientos de nodos de procesamiento de estas tarjetas para realizar cálculos de carácter matemático a gran velocidad y con un coste reducido.
- Usar procesadores de múltiples núcleos. Existen varias alternativas dentro de esta tecnología, como el empleo de semáforos, hilos o el estándar OpenMP.
- Usar clusters, es decir, un conjunto de ordenadores conectados por una malla de comunicaciones de altas prestaciones, que permiten realizar una ejecución en un entorno masivamente paralelo utilizando el estándar MPI.

En este proyecto vamos a realizar la optimización de los algoritmos empleando las siguientes técnicas de paralelización:

- Tecnología CUDA basada en las tarjetas gráficas NVidia [18], [6], [8]:

  Esta tecnología está basada en lenguaje C. Las aplicaciones se desarrollan usando este lenguaje de alto nivel, y posteriormente se pasa a un nivel mas bajo, añadiendo una serie de extensiones o instrucciones específicas [7] sobre el código original para conseguir paralelizar la aplicación, y que parte del código sea procesado por los multiprocesadores incluídos en la tarjeta gráfica (GPUs) en vez de por la CPU. De esta forma se consigue realizar una gran cantidad de cálculos complejos a una mayor velocidad y con un coste reducido, lo que supone obtener un mayor ancho de banda y mejorar los tiempos de procesado.

  Para realizar este proyecto se utilizará el modelo de tarjeta gráfica NVidia GeForce GTX 280, que dispone de 30 multiprocesadores y puede trabajar con datos en doble precisión. Se utilizarán las dos librerías proporcionadas por CUDA para realizar cálculos complejos:

  - BLAS [5]
  - FFT

- Estándar OpenMP para procesadores de múltiples núcleos [15], [1]:

  Esta tecnología se basa en desarrollar aplicaciones en lenguaje C, y posteriormente añadir una serie de directivas específicas [17] para realizar la paralelización.

  Este estándar presenta una gran ventaja respecto a CUDA, y es que OpenMP es poco intrusivo, es decir, hay que realizar pocas modificaciones sobre el código original. Por el contrario, al ser un sistema de memoria compartida, el ancho de banda del bus no se incrementa aunque tengamos mas procesadores.

Una vez que se hayan probado y verificado el correcto funcionamiento de ambas técnicas de paralelización, éstas podrán ser usadas en proyectos futuros para la generación de imágenes en un tiempo reducido.

# 4. Fases del desarrollo

Las fases que se van a seguir a lo largo del proyecto son:

- Familiarización con el entorno de trabajo y las distintas aplicaciones software necesarias para desarrollar el Proyecto Fin de Carrera.
- Análisis y documentación acerca de la cadena de procesado que se utiliza en el tratamiento de imágenes de teledetección, y más en concreto en los algoritmos utilizados en el proyecto AIPC de Deimos Space [11], [14], [4], [13].
- Formación y documentación sobre la tecnología CUDA de NVidia para realizar paralelización [18], [7].
- Optimización de los algoritmos de procesado mediante la tecnología CUDA y su posterior evaluación mediante diversas pruebas.
- Comprobación del correcto funcionamiento a partir de las imágenes generadas por los algoritmos originales y las generadas por los algoritmos optimizados con CUDA.
- Formación y documentación sobre el estándar OpenMP para realizar paralelización [15], [17].
- Optimización de los algoritmos de procesado mediante el estándar OpenMP y su posterior evaluación mediante diversas pruebas.
- Comprobación del correcto funcionamiento a partir de las imágenes generadas por los algoritmos originales y las generadas por los algoritmos optimizados con OpenMP.
- Documentación del proyecto.

# 5. Herramientas y recursos

Las herramientas necesarias para la elaboración del proyecto son:

- PC compatible
- Sistema operativo GNU/Linux
- Sistema operativo Windows XP
- Estación de trabajo (servidor) 5046A-XB con las siguientes especificaciones:
  - Intel Xeon Harpertown E5405 Quad-Core Processor / 2.0Ghz / 12MB L2 Cache / 1333 Mhz FSB.
  - 16GB DDR2 RAM
  - RAID-1 w/ 2 x SEAGATE HD 500GB SATA-II 7200 rpm 32MB Cache
- Tarjeta gráfica NVidia GeForce GTX 280
- Aplicación Xmanager para acceso remoto
- Procesador de textos LaTeX[2]
- Lenguaje de programación C/C++
- Tecnología CUDA
- Estándar OpenMP
- Compilador C/C++ gcc
- Compilador para CUDA nvcc

Otros recursos necesarios para la elaboración del proyecto son:

- Base de datos con imágenes de prueba
- Algoritmos de procesado implementados en el proyecto AIPC de Deimos Space

## Referencias


[1] Sun One Studio 8. *OpenMP API Users Guide*, 2003.

[2] Francisco J. Blancas Peral and María Cortada García. *Guía rápida para el nuevo usuario de LATEX*, 2004.

[3] European Space Agency (ESA), (2009). http://www.esa.int/esaCP/index.html.

[4] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Addison-Wesley, 2003.

[5] NVidia. *CUBLAS Library*, 2008.

[6] NVidia. *CUDA Getting Started*, 2008.

[7] NVidia. *CUDA Programming Guide*, 2008.

[8] NVidia. *CUDA Reference manual*, 2008.

[9] Deimos Space. *Algorithmic Models: Modules E-G*, 2008. Internal Reference: AIPC-DMS-TEC-TNO04.

[10] Deimos Space. *España Virtual*, 2008. Internal Reference: EVIRTUAL-DMS-EST-CO-PT2-T2-1.

[11] Deimos Space. *Estudio del Estado del Arte en Sistemas Grid Específicos para Algoritmos Espaciales*, 2008. Internal Reference: EVIRTUAL-DMS-EST-CO-PT2-T2-1.

[12] Deimos Space. *Requirements Baseline: co-registration, ortho-rectification and mosaicking*, 2008. Internal Reference: AIPC-DMS-TEC-TNO02.

[13] Deimos Space. *SYSTEM DESIGN DOCUMENT*, 2008. Internal Reference: AIPC-DMS-TEC-ADD01.

[14] Deimos Space. *Definición del Activo Experimental Integrado*, 2009. Internal Reference: EVIRTUAL-DMS-AEI-CO-PT4-T1-0.

[15] The OpenMP API specification for parallel programming, (2009). http://openmp.org/wp/.

[16] UNED. *Curso de teledetección*.

[17] Joel Yliluoma. *Guide into OpenMP: Easy multithreading programming for C++*, 2007.

[18] NVidia CUDA Zone, (2009). http://www.nvidia.es/object/cudawhatises.html.