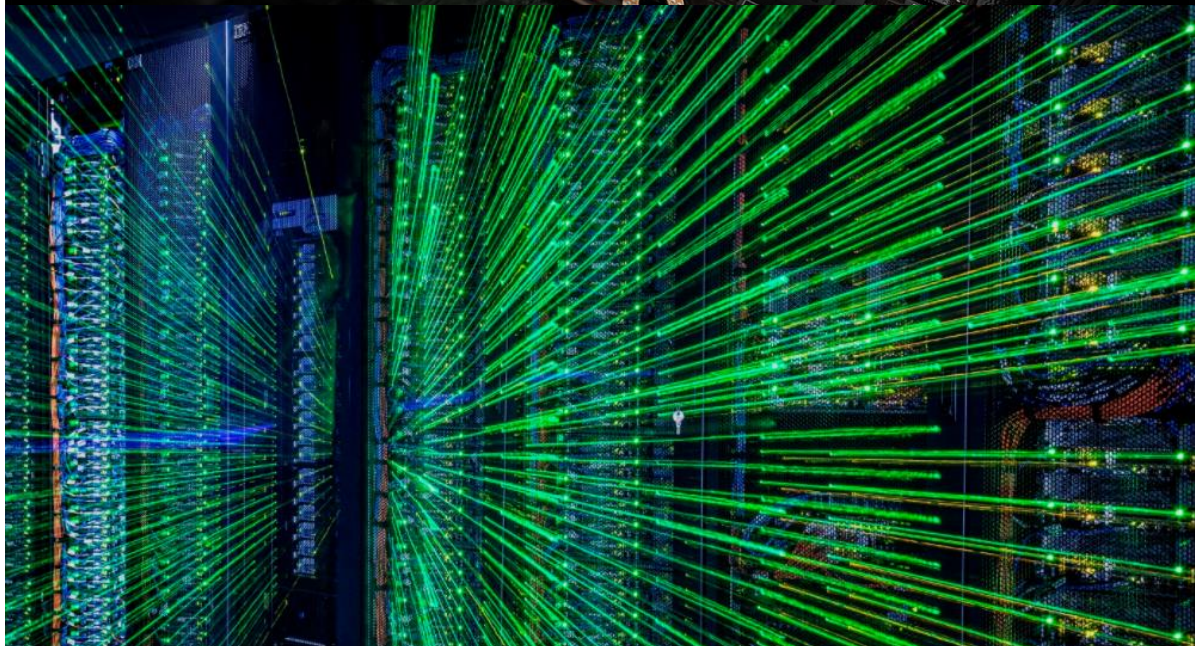
Departamento de Ingeniería de Computadores
Facultad de Informática de A Coruña
UNIVERSIDADE DA CORUÑA

TRABAJO FIN DE MÁSTER
MÁSTER INTERUNIVERSITARIO EN
COMPUTACIÓN DE ALTAS PRESTACIONES

Mapeo de imágenes multiespectrales sobre nubes de puntos 3D de alta densidad en GPU

Juan Manuel Jurado Rodríguez

Directores: Emilio José Padrón González
Francisco Ramón Feito Higueruela



Introducción

- Fusión de datos heterogéneos.
- Procesamiento de grandes volúmenes de datos.
- Manejo de modelos 3D.
- HPC en teledetección.
- El uso de la GPU para computación.

Motivación

- A partir del método:

Multispectral Mapping on 3D Models and Multi-Temporal Monitoring for Individual Characterization of Olive Trees

by J. M. Jurado *, L. Ortega, J. J. Cubillas and F. R. Feito

Computer Graphics and Geomatics Group of Jaén, University of Jaén, 23001 Jaén, Spain

* Author to whom correspondence should be addressed.

Remote Sens. **2020**, *12*(7), 1106; https://doi.org/10.3390/rs12071106

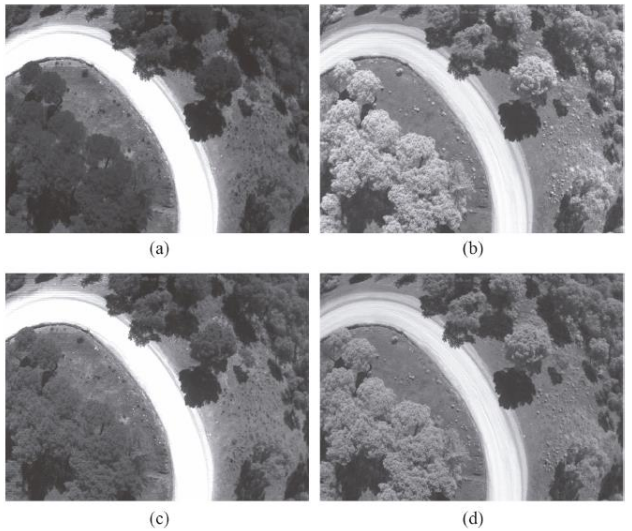
- Limitaciones:
  1. Bajo rendimiento con nubes de puntos superiores a 5 millones de puntos.
  2. No es posible el cálculo en remoto.
  3. Altas especificaciones hardware en la máquina cliente.

Objetivos

1. Desarrollo de un método acelerado por GPU para el mapeo de imágenes multiespectrales sobre grandes nubes de puntos.
2. Desarrollo de un método out-of-core no dependiente de las restricciones de memoria en el hardware.
3. Paralelización en GPU del método utilizando CUDA.
4. Optimizar el rendimiento de la versión secuencial.

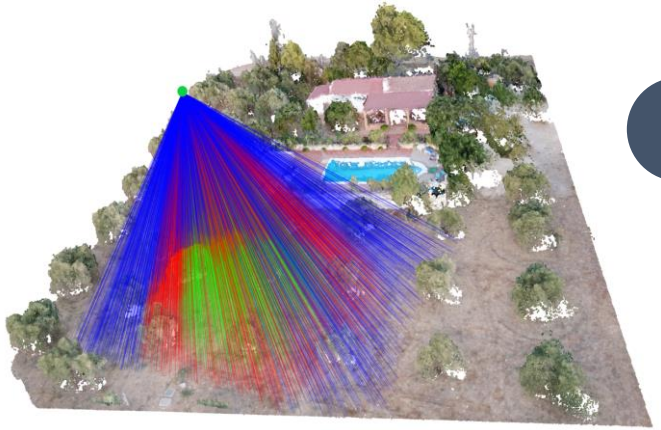
Conjunto de datos

- Nube de puntos de alta resolución (~66 millones)
- Conjunto de imágenes multiespectrales (180)

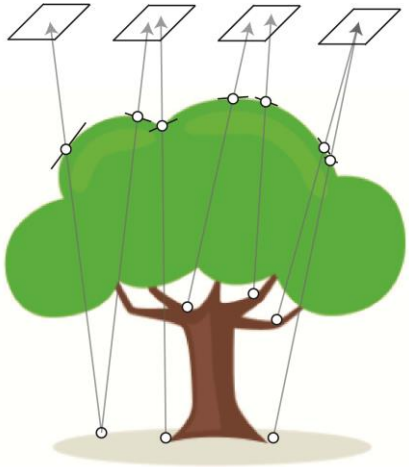


Método secuencial

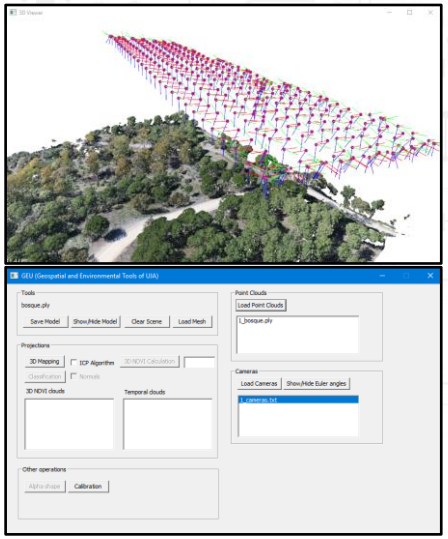
1 Mapeo de puntos 3D en el espacio de la imagen.



2 Cálculo de la oclusión



GEU
Geospatial and Environmental tools of University of Jaén



Método secuencial

Resultado:

Paralelización en CPU

- Se utilizó OpenMP para:
  1. Paralelizar la proyección de cada punto sobre el conjunto de cámaras desde donde es visible.

```
#pragma omp parallel for private (it, k, r, c, rot_m, res, temp, cam, id_point) shared(point_cameras, cloud)
for (it = 0; it < size; it++) {

    Projection* proj_point;
    unsigned int num_cameras = point_cameras[it].num_cameras;
    proj_point = (Projection*)malloc(num_cameras * sizeof(Projection));

    //Cámaras para cada punto
    for (k = 0; k < num_cameras; k++) {
```

  2. Paralelizar el cálculo de la oclusión para cada imagen.

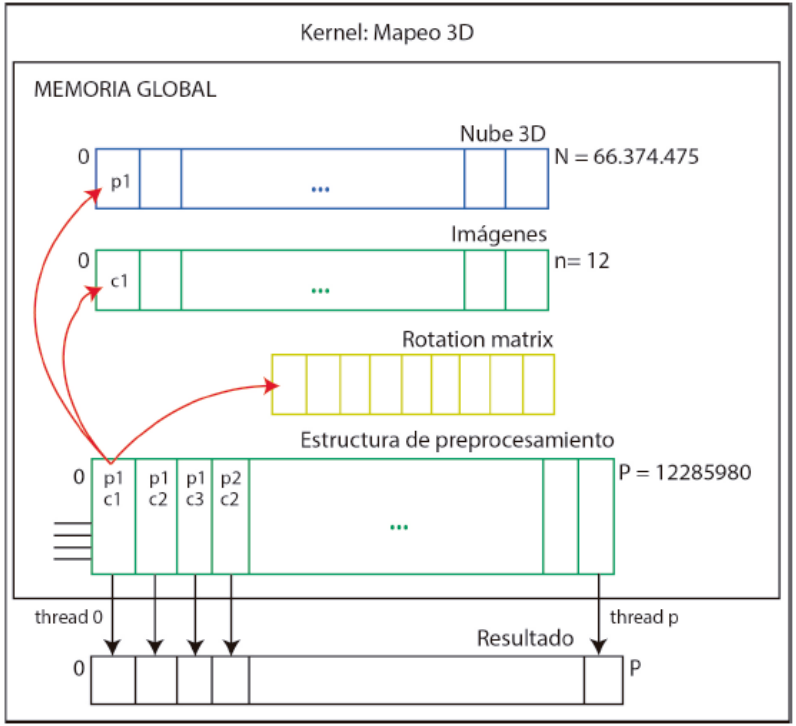
```
#pragma omp parallel for private(it, i, j, m_occlusion, indices, px, py, id_point, id_cam) shared(projections)
for (it = 0; it < num_images; it++) {

    //Se inicializa la matriz de oclusión a 0
    int numbytes = 960 * 1280 * sizeof(float);
    m_occlusion = (float*)malloc(numbytes);
    memset(m_occlusion, 0x77, numbytes);

    numbytes = 960 * 1280 * sizeof(c_p);
    indices = (c_p*)malloc(numbytes);
    memset(indices, -1, numbytes);
    for (i = 0; i < point_cameras.size(); i++) {
        for (j = 0; j < point_cameras[i].num_cameras; j++) {
```

Paralelización en GPU (v.1)

- Operación de mapeo 3D.
- Operación de test de oclusión.
- Características:
  - Alto nivel de paralelización en el Mapeo 3D.
  - Uso de una estructura de preprocesamiento.
  - Ordenación en CPU para el test de oclusión.
  - Bajo paralelismo de los datos en el cálculo de la oclusión.

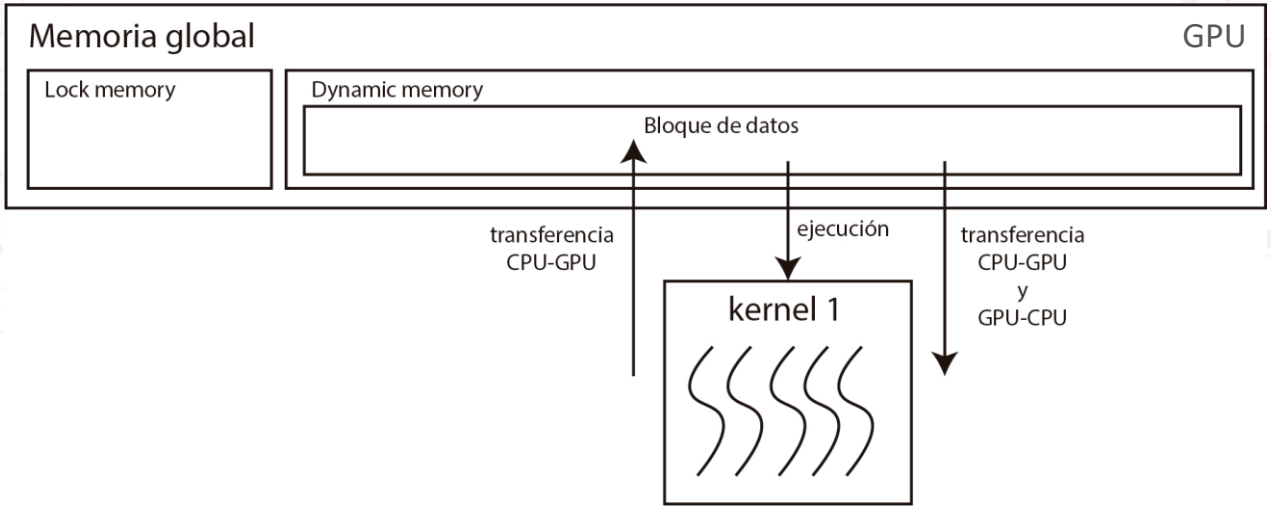


Paralelización en GPU (v.2)

- Se replantea el enfoque del algoritmo con el fin de:
  - Optimizar la salida de *kernel* de mapeo para el cálculo de oclusión.
  - Realizar la ordenación en GPU.
  - No utilizar una estructura de preprocesamiento.
  - Selección del punto 3D en cada pixel a partir de una operación de Reducción.

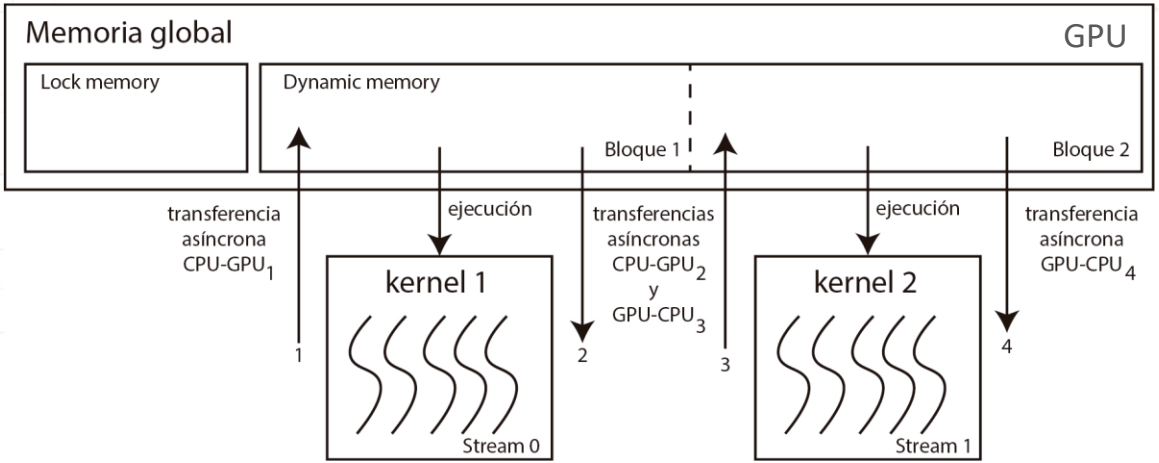
Paralelización en GPU (v.3.1 *out-of-core*)

- Características:
  - Partición de la nube de puntos en varios segmentos.
  - Transferencia y ejecución del *kernel* de forma síncrona.
  - Se produce un impacto negativo en el rendimiento para la transferencia de memoria.



Paralelización en GPU (v.3.2 *out-of-core*)

- Características:
  - Partición de la nube de puntos en varios segmentos.
  - Para cada iteración se crean dos bloques.
  - La transferencia de memoria y el cálculo en GPU se produce de forma asíncrona.



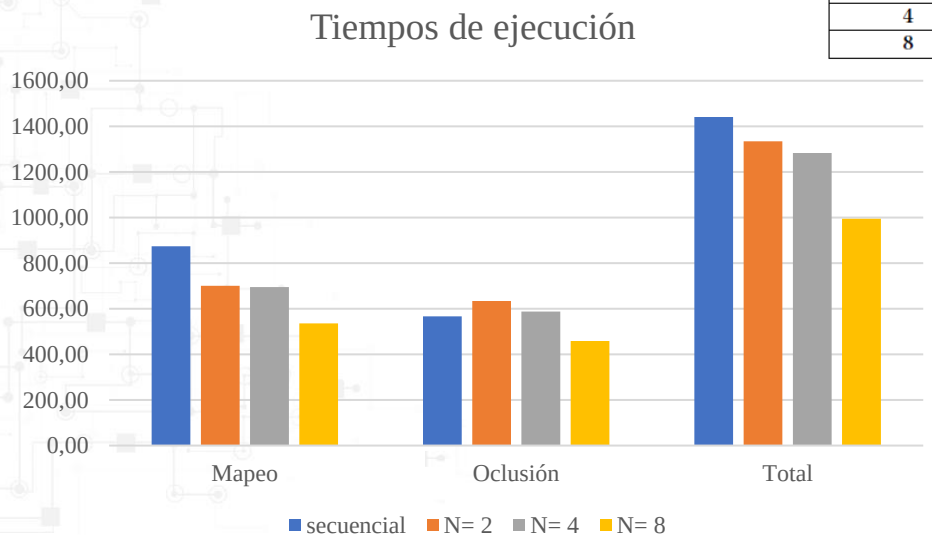
Estudio del rendimiento

- Máquina de test:
  - Procesador: Intel con 4 núcleos (cores) i7-4790 CPU @ 3.60GHz (hyperthreading activado con 8 cores virtuales).
  - Memoria RAM: 24 GB, Caché L1: 32 KB para datos y 32 KB para instrucciones, Caché L2: 256 KB, Caché L3: 8 MB.
  - GPU: Nvidia TITAN V (Nvidia Driver: 450.57) con 5120 núcleos y VRAM de 12 GB.
- Conjunto de test:
  - Nube de puntos: 66 millones de puntos
  - Número de imágenes: 12

Estudio del rendimiento

- Prueba 1: Aceleración con OpenMP.

	icc			gcc		
Nº de núcleos	Mapeo 3D	Oclusión	Total	Mapeo 3D	Oclusión	Total
2	759.53	434.29	1193.86	700.24	633.98	1334.25
4	767.38	420.18	1187.6	694.88	587.73	1282.65
8	792.32	391.21	1183.56	535.99	458.45	994.47



- Mejor tiempo: **994,47 ms**
- Aceleración global: **30.4 %**

Estudio del rendimiento

- Prueba 2: Aceleración con GPU.

	Mapeo 3D		Oclusión		Transferencia de memoria		Totales	
Hilos/Versión	v.2	v.3.1	v.2	v.3.1	v.2	v.3.1	v.2	v.3.1
32 hilos	90.37	95.01	352.52	418.94	535.73	4909.89	1366.48	5523.12
64 hilos	90.36	94.75	352.3	420.59	981.56	4582.51	981.06	5203.4
128 hilos	90.38	95.17	352.37	420.73	534.45	4554.2	980.186	5173.82

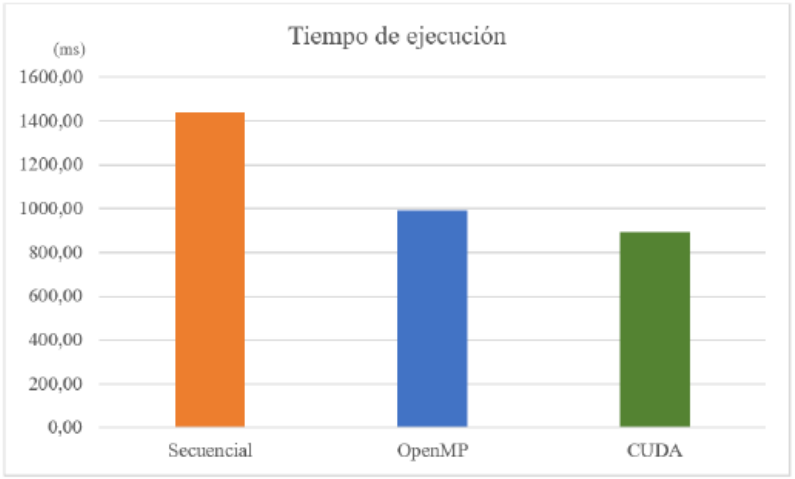
	Tiempo total	
Hilos/Versión	v.2.0	v.3.2
32 hilos	1366.48	1243.62
64 hilos	981.06	902.717
128 hilos	980.186	893.94

- Mejor tiempo: **893,94 ms**
- Aceleración global: **38 %**

Estudio del rendimiento

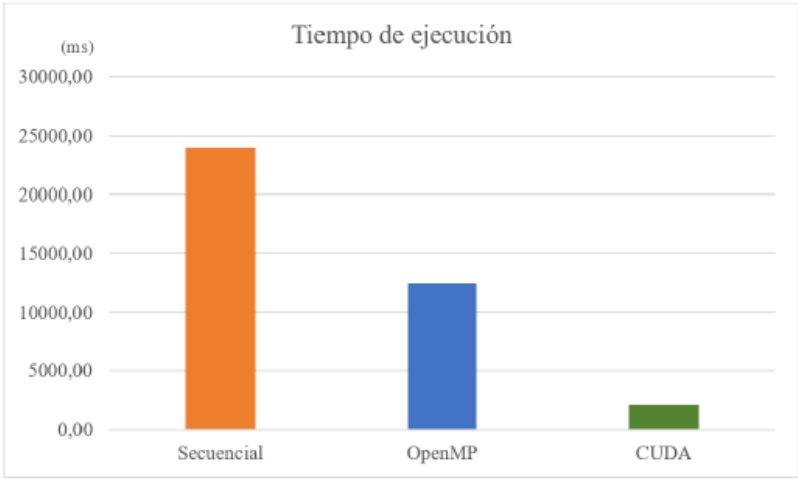
- Prueba 3: Comparación global

Conjunto de test



- OpenMP: **30% (~994 s)**
- CUDA: **38% (~890 s)**

Dataset completo



- OpenMP: **48% (~12 s)**
- CUDA: **91% (~2 s)**

Conclusiones y trabajo futuro

- Se alcanza una significativa aceleración del método en GPU.
- Se propone una solución out-of-core.
- Se posibilita al experto tener información espectral en un escenario 3D con una alta resolución geométrica.
- Próximamente se pretende ampliar la paralelización del método siguiendo un enfoque multi-GPU.
- Publicación del trabajo en una revista científica de alto impacto.

Gracias por su atención.