High Performance Computing
Departamento de Ingeniería en Informática
LAB recuperativo : SIMT- CUDA

1 Objetivo

El objetivo de este laboratorio es implementar un proceso llamado *gridding*, que se utiliza en astronomía, con CUDA.

2 Interferometría

2.1 El observatotio ALMA

Las antenas del observatorio ALMA se ubican en el llano de Chajnantor a 4800 metros sobre el nivel del mar, en el Desierto de Atacama. Estas antenas miden señales de radio frecuencia en longitudes de onda milimétricas y submilimétricas. La señal capturada por las antenas, entonces, no es la luz visible emitida por el objeto, sino la radiación que esa luz produce al interactuar con el gas y polvo que se encuentra alrededor del objeto. Es decir, es necesario transformar esta señal de RF a una señal de intensidad. Este proceso se llama síntesis de imágenes.

Para ser más específicos, la señal capturada por las antenas corresponde a mediciones en el plano de Fourier. La figura 1 muestra una típica distribución de posiciones de muestreo en el plano uv (plano Fourier) de una cierta configuración de antenas. Cada punto representa una medición, llamada **visibilidad**. Aunque no es relevante para este lab, diremos que cada par de antenas produce una visibilidad, y a medida que pasa el tiempo y gira la tierra, el punto se mueve en el plano uv.

Como se puede observar, los puntos no cubren completamente el plano uv. Más aún, el muestreo es irregular, en el sentido que no están regularmente espaciados en ambas direcciones.

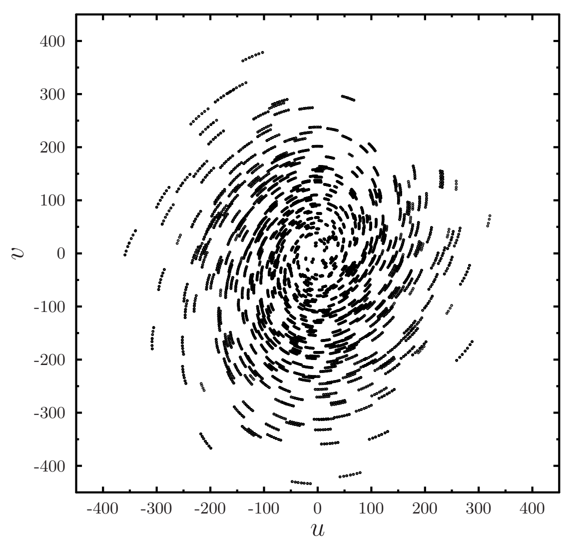

Figure 1: Típico muestreo del plano uv por ALMA.

2.2 Conceptos básicos de Fourier

Sea $I(x, y)$ una imagen bidimensional de $N \times N$ píxeles, donde cada pixel representa el nivel de intensidad en la posición (x, y). Luego, la transformada discreta de Fourier de esta imagen es:

$$V(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} I(x, y) \exp(-2\pi j(ux + vy)/N) \tag{1}$$

y su transformada inversa

$$I(x, y) = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \exp(+2\pi j(ux + vy)/N) \tag{2}$$

ALMA mide $F(u, v)$, pero la imagen que deseamos es $I(x, y)$. Si tuvieramos el plano uv completo, es decir los valores $F(u, v)$ para todos los puntos uv y éstos correspondiesen a una grilla regular, entonces la síntesis de imagen sería trivial, pues bastaría aplicar la transformada inversa a los datos $V(u, v)$ y obtendríamos $I(x, y)$.

Las ecuaciones anteriores son válidas cuando la imagen y su transformada de Fourier están muestrados en una grilla regular. Es decir, los píxeles están equidistantemente espaciados en la dirección x y en la dirección y. Suponga que la distancia entre los píxeles de la imagen $I(x, y)$ es Δx y Δy. Entonces, la distancia en los puntos en su transformada $V(u, v)$ es:

$$\Delta u = \frac{1}{N\Delta x} \quad \Delta v = \frac{1}{N\Delta y} \tag{3}$$

La siguiente figura representa la relación de estas ecuaciones. Note que la regularidad de muestreo también significa que tanto I como V pueden ser almacenados en matrices cuadradas. La siguiente figura muestra una imagen junto a su transformada de Fourier en grillas regulares. Note que $V(u, v)$ es una matriz compleja, es decir cada punto posee un componente real y otro imaginario, y lo que se visualiza es la magnitud de este valor. Las imágenes fueron creadas con Matlab con los siguiente commandos:

```
>> f = fopen('HLtau.raw', 'r');
>> I = fread(f, 'float');
>> fclose(f);
>> I = reshape(I, N, N);
>> I = I';
>> V = fft2(I);
>> F = fftshift(fftshift(V));
>> imagesc(I); axis('square'); colormap(gray)
>> figure
>> imagesc(log(abs(V)+1)); axis('square'); colormap(gray)
```

Primero se abre y lee el archivo de una imagen. Como la imagen se lee como un vector, éste se re-forma como una imagen cuadrada de $N \times N$. La operación `I = I'` simplemente traspone la imagen. Esto es necesario pues matlab lee por columns y no por filas. Luego, se calcula la transformada de Fourier con la Fast Fourier Transform (`fft2`). Para que el origen del plano uv quede en el centro, se aplica `fftshift`. Finalmente se visualiza el módulo de la transformada.

3 Gridding

Lamentablemente, los datos que ALMA captura no son completos ni regulares. Como se puede apreciar en la Figura 1, los puntos (u, v) pueden estar en cualquier posición y no necesariamente están equiespaciados uno del otro. Para que esto quede más claro, la Figura 2 muestra un zoom del plano uv junto con una grilla regular. Por lo tanto los puntos uv no me sirven para aplicar la inversa de Fourier y obtener la imagen.

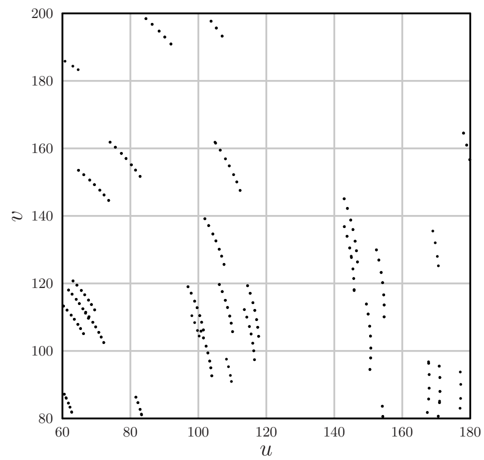

Figure 2: Zoom en el plano uv.

El proceso de gridding consiste en construir datos regularmente espaciados en el plano Fourier, a partir de los datos que entrega ALMA. Hay muchas formas de hacer esto, pero la más sencilla es sumar todos los valores de las visibilidades que se encuentran en una vecindad de cada punto de la grilla regular.

A continuación, explicamos con detalle esta operación.

Asuma que $\Delta u = \Delta v$. Suponga además que queremos construir una grilla regular grideada de tamaño $N \times N$, tal que las posiciones regulares corresponden a

$$(u_i, v_j) = (i\Delta u, j\Delta v) \quad i, j = -(N/2 - 1), -(N/2 - 2), \ldots, -1, 0, 1, \ldots . N/2 - 1$$

Suponga que los datos de ALMA corresponden a una lista de Z visibilidades del tipo

$$L = \{(u_k, v_k), V(u_k, v_k).r, V(u_k, v_k).i; k = 1, \ldots Z\}$$

donde

1. (u_k, v_k) corresponde a la coordenada en el plano uv de la visibilidad k
2. $V(u_k, v_k).r$ es el componente real del valor de la visibilidad k
3. $V(u_k, v_k).i$ es el componente imaginario del valor de la visibilidad k

La diferencia entre (u_i, v_j) y (u_k, v_k) es que u_k y v_k pueden ser cualquier valor, en cambio u_i y v_j son solo valores que corresponden a la grilla regular. En la Figura 2 las posiciones en las intersecciones de las líneas son las posiciones en la grilla regular, y las posiciones marcadas con puntos son posiciones de

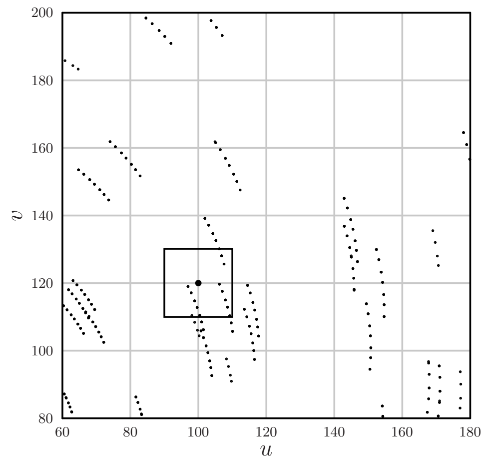

Figure 3: Ejemplo de vecindad que define los puntos que se suman para el punto de la grilla regular.

las visibilidades de la lista L. Entonces, a partir de los valores en los puntos deseamos estimar el valor en las intersecciones. En la Figura 3 se muestra la vecindad del punto de la grilla regular $(100, 120)$. Todos los puntos uv que están dentro de esa vecindad cuadrada deben sumarse para formar el valor del punto en la grilla. Note que para este ejemplo $\Delta u = \Delta v = 20$.

4 El programa

La invocación al programa desde línea de comando será la siguiente

```
$ ./gridding -i datosuv.raw -z numerodedatos -d deltax -N tamañoimagen -o archivo
```

El nombre del archivo con los datos es indicado con la opción `-i`. Este archivo es binario y contiene la información para cada visibilidad en precisión simple. El número de visibilidades se indica con la opción `-z`. Así, el archivo de entrada contiene $4 \times 4 \times Z$ bytes. Cada dato se compone de la coordenada u, la coordenada v, el componente real y el componente imaginario.

La opción `-d` especifica el valor Δx en arco segundos, y asumimos que $\Delta x = \Delta y$. Este valor debe ser convertido a radianes antes de calcular Δu. La opción `-N` especifica el número de filas y de columnas de la imagen $I(x, y)$ y también de la imagen en el plano Fourier grideado. Asumimos que son matrices cuadras. Con Δx y N podemos calcular Δu.

Para simplificar, el program solo debe calcular el plano de Fourier grideado, centrado en $(N/2, N/2)$, es decir $V[N/2, N/2]$ corresponde al componente de frecuencia $(0, 0)$. Como $V(u, v)$ es complejo entonces se deben escribir dos archivos `archivoreal.raw` y `archivoimg.raw` conteniendo los componentes real e imaginario en formato binario.

Para chequear que la solución es correcta, se pueden leer estos dos archivos en Matlab y crear una matriz compleja. Luego, shiftear la imagen tal que el componente de frecuencia $(0, 0)$ quede en la posición $[1, 1]$ (posición Matlab) de la matriz (usar `fftshift()`). Luego, se calcula la inversa de Fourier y obtenemos la imagen $I(x, y)$ que podemos visualizar con `imagesc()`.

Construya un programa en CUDA que implemente una solución a este laboratorio.

5 Entregables

Tarree, comprima y envíe a `fernando.rannou@usach.cl` al menos los siguientes archivos:

1. `Makefile`: archivo para make que compila los programas
2. `gridding.cu`: archivo con el código fuente. Puede incluir otros archivos fuentes.
3. `gridding.pdf`: informe breve de su solución. Estrategia de paralelizaciín y rendimiento computacional.