Proyecto Final

Velocimetría por Imágenes de Partículas

Didier Muñoz Díaz

21 de mayo de 2019

1. Introduction

La técnica de Velocimetría por Imágenes de Partículas (VIP) es una técnica diseñada para estudiar el flujo de fluidos. Tiene la ventaja de ser poco invasiva en comparación con otras técnicas (prácticamente no modifica el comportamiento del fluido) mientras logra obtener un campo de velocidades en dos dimensiones (o más) con una gran flexibilidad para estudiar diversas geometrías y tipos de fluidos.

La técnica de Velocimetría por Imágenes de Partículas, como su nombre lo indica, consiste en obtener imágenes de un fluido en movimiento al cual se le han agregado partículas de tal forma que un procesamiento de estas imágenes revele información sobre la velocidad del flujo a través del movimiento de las partículas. La Figura 1 muestra un diagrama de cómo se obtienen las imágenes utilizando una cámara de alta velocidad y un plano de luz láser. En este tipo de configuración lo normal es tomar dos imágenes, una al tiempo t y otra al tiempo $t + \Delta t$, para poder ver el movimiento con mayor facilidad (hay técnicas para utilizar una sola imagen, pero en general no se obtienen tan buenos resultados).

1.1. Procesamiento de las imágenes

El procesamiento de las imágenes para obtener información del fluido se puede dividir en tres partes:

- Pre-procesamiento. Primero las imágenes se adecúan y preparan. En este primer paso se busca eliminar los imperfectos presentes en las imágenes, como por ejemplo artefactos indeseados o mala iluminación.
- Procesamiento. En esta etapa se obtiene la información de desplazamiento del fluido. En general, este proceso consiste en dividir las imágenes en pequeñas regiones, llamadas zonas de interrogación, donde se estima estadísticamente el desplazamiento promedio de las partículas a través de un algoritmo de correlación. Cada una de estas regiones, junto con su vector desplazamiento estimado, forman en conjunto un mapa de deplazamientos.

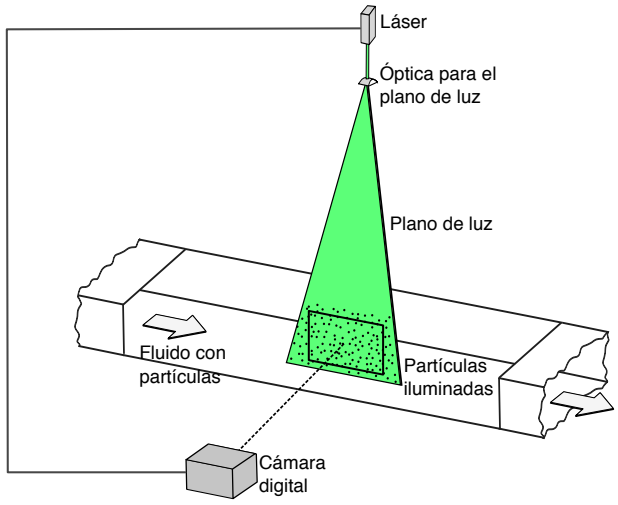


Figura 1: Diagrama de una configuración experimental típica para la obtención de las imágenes.

- Post-Procesamiento. Esta fase final se encarga de mejorar y corregir el mapa de desplazamientos. Generalmente el mapa de desplazamientos presenta zonas de interrogación donde el desplazamiento calculado es erróneo y es necesario identificar estos vectores y corregirlos.

Este proyecto se enfoca en la etapa de *procesamiento* donde utilizando la Transformada Rápida de Fourier se calculan rápidamente las correlaciones necesarias y se obtiene el mapa de desplazamientos.

La implementación se ha hecho en python. Utilizando numpy, PIL y multiprocessing.

2. Procesamiento de las imágenes

Parte central del procesamiento de las imágenes es el algoritmo de correlación que consiste de las siguientes partes:

1. Definir las zonas de interrogación. En este caso se define una malla regular y la idea es encontrar posteriormente un vector desplazamiento para cada una de las celdas. Las mallas se definen de la misma forma para las dos imágenes.

2. Correlacionar cada zona de interrogación en la imagen al tiempo t con una zona de interrogación correspondiente en la imagen al tiempo $t + \Delta t$. Para cada par de zonas se obtiene un *mapa de correlación* al calcular la correlación circular de ambas zonas. La Figura 2 explica un mapa de correlación y qué representa.

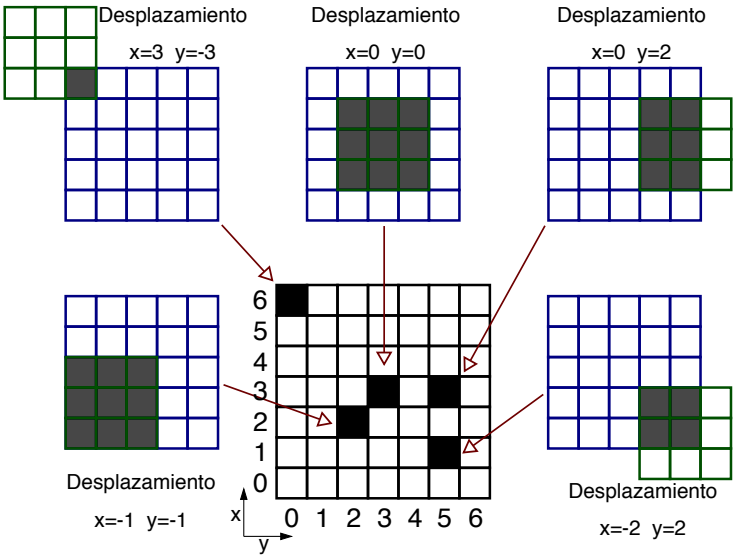


Figura 2: Mapa de correlación de dos zonas de interrogación mostrando qué significa cada entrada.

3. De cada mapa de correlación se obtiene un vector desplazamiento con una búsqueda del máximo del mapa de desplazamiento. La Figura 3 muestra un ejemplo de correlación de dos imágenes y del máximo que se necesita localizar. Finalmente estos vectores forman el mapa de desplazamientos.

2.1. Correlación

La correlación que se busca calcular se define de la siguiente forma:

Definición 1 (Correlación circular en dos dimensiones)**.** *Sean* $f(x,y)$ *y* $g(x,y)$ *dos imágenes definidas en* $\{0,1,\ldots,M-1\}\times\{0,1,\ldots,N-1\}$*. La* correlación circular *de éstas se define como la función:*

$$f(x,y)\mathring{\star}g(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)g(x+m,y+n)$$
$$x = 0,1,\ldots,M-1$$
$$y = 0,1,\ldots,N-1$$

Con $g(x+M,y+N) = g(x+M,y) = g(x,y+N) = g(x,y)$*.*

Nótese que esto implica considerar que la función $g(x)$ *se extiende periódicamente por un periodo más en ambas direcciones.*

Esta operación significa desplazar una función con respecto a la otra, luego multiplicar las funciones considerando extenderlas periódicamente cuando sea necesario y finalmente sumar estos productos.

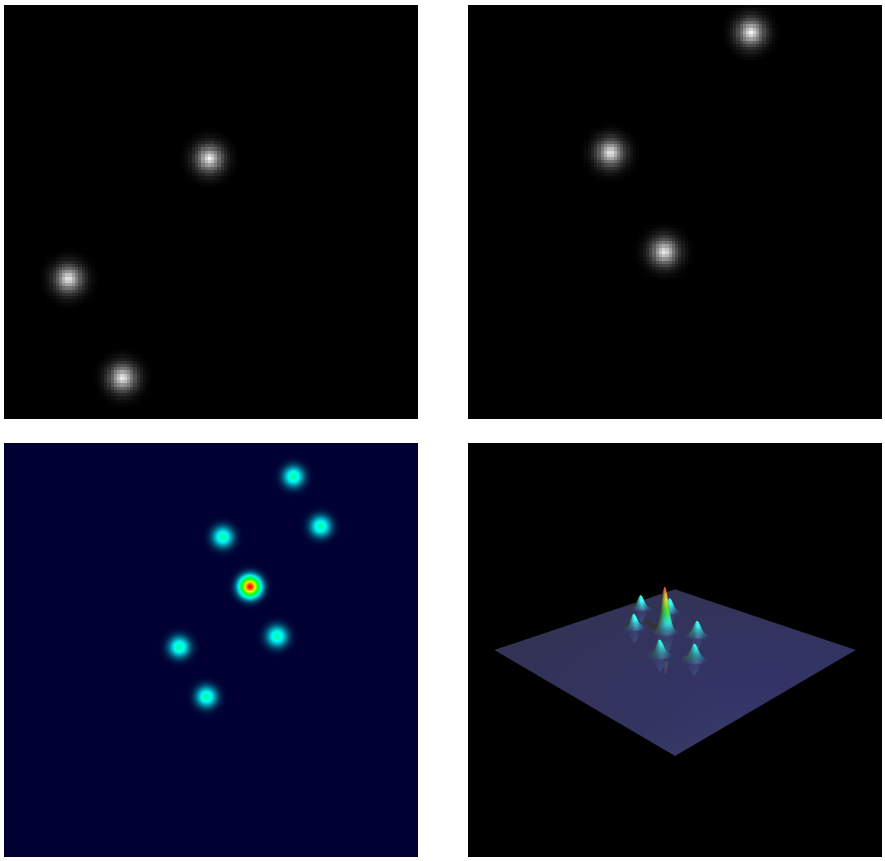


Figura 3: Ejemplo de mapa de correlación para dos zona de correlación. Se puede observar que el punto máximo de la correlación representa el desplazamiento que produce un mejor "empate" entre las dos zonas de interrogación.

La razón detrás de que sea esta correlación la que se busca calcular es que para hacer el procesamiento eficiente se utiliza el siguiente teorema de correlación que involucra a la transformada discreta de Fourier.

Teorema 1 (Teorema de Correlación)**.** *Sean* $f(x,y)$ *y* $g(x,y)$ *dos imágenes definidas en* $\{0,1,2,\ldots,M-1\}\times\{0,1,2,\ldots,N-1\}$*, con* $F(u,v)$ *y* $G(u,v)$ *sus Transformadas Discretas de Fourier respectivamente.*

Entonces:

- *La TDF de* $f(x,y) \mathring{\star} g(x,y)$ *es* $F^*(u,v)G(u,v)$*.*
- *La TIDF de* $F^*(u,v)G(u,v)$ *es* $f(x,y) \mathring{\star} g(x,y)$*.*

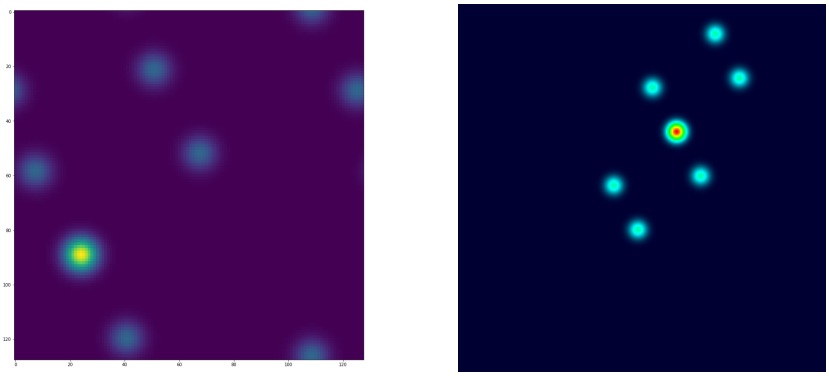
Donde $F^*(u,v)$ *significa el conjugado complejo de* $F(u,v)$*.*

Con este teorema y con la ventaja de la Transformada Rápida de Fourier que tiene una complejidad computacional asintótica de $O(MNlog(MN))$ para 2 dimensiones con M y N las dimensiones de la imagen, se puede lograr calcular el algoritmo de correlación de forma eficiente.

3. Implementación

En el archivo *PIV.ipynb* se muestra el código y el flujo de trabajo. En resumen las partes de las que está compuesto el proyecto son:

- Carga de imágenes con el módulo `Image` de `PIL`. Conversión de las imágenes cargadas a arreglos de `numpy`.
- División de las imágenes en zonas de interrogación con `numpy.vsplit` y `numpy.hsplit`.
- Correlación circular utilizando `numpy.fft`.
- Obtención de desplazamientos a partir de mapas de correlación utilizando `numpy.argmax`. Y adecuación del índice obtenido fijándose en que el mapa de correlación calculado a través del teorema de correlación considera funciones periódicas y que mueve la correlación mostrando primero la segunda mitad y luego la primera, en cada eje. La Figura 4 ilustra este comportamiento.
- Visualización y guardado de los resultados utilizando `matplotlib`.



(a) Mapa de correlación resultado de calcularse con el teorema de correlación.

(b) El mapa de correlación que se espera al calcularse directamente.

Figura 4: Comparación de mapas de correlación.

Además, se programó una versión que calcula los desplazamientos en paralelo utilizando la clase `process`. Esta versión está implementada en el archivo *PIV.py*

y se invoca desde la terminal, pasándole los nombres de las imágenes a procesar, las dimensiones de las zonas de interrogación y el nombre del archivo de salida.

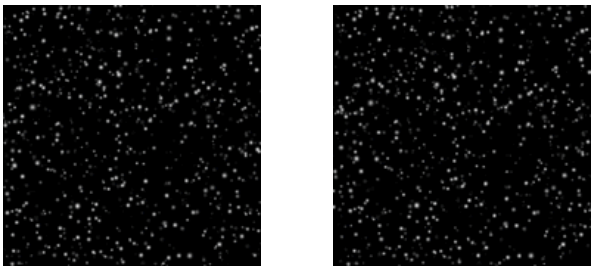
Se pensó en implementar la versión paralela con MPI pero al hacer pruebas el cálculo de los desplazamientos no parece ser lo más tardado del programa. Lo que más se lleva es la visualización de los datos y esto es algo que de todos modos se tiene que hacer en un único nodo.

Se utilizó `numba` para tratar de acelerar los cálculos.

4. Resultados

Se procesaron con éxito algunas imágenes. A continuación se muestran dos ejemplos:

Experimento 1 Un par de imágenes de prueba, con zonas de interogación de 22×22 pixeles.



(a) Imagen al tiempo t. (b) Imagen al tiempo $t + \Delta t$.

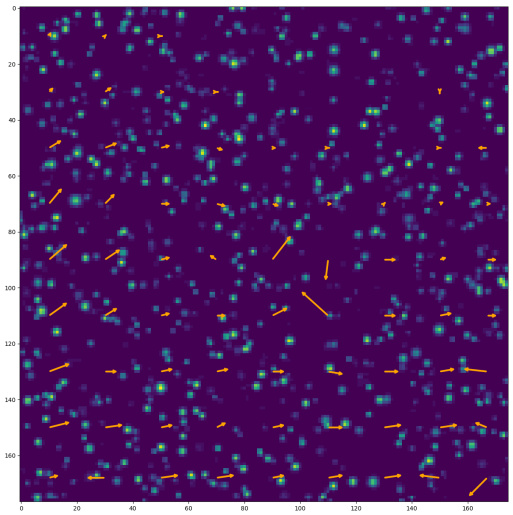
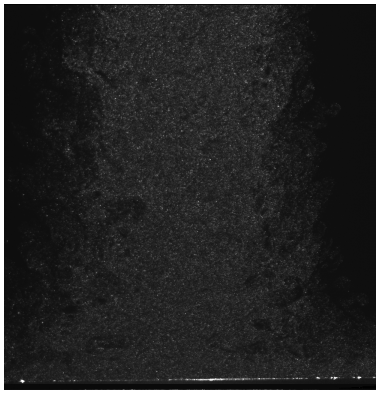
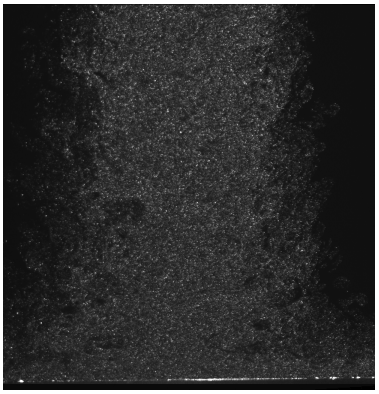


Figura 5: Mapa de desplazamientos.

Experimento 2 En este experimento las zonas de interrogación son de 40×40 pixeles.



(a) Imagen al tiempo t.



(b) Imagen al tiempo $t + \Delta t$.

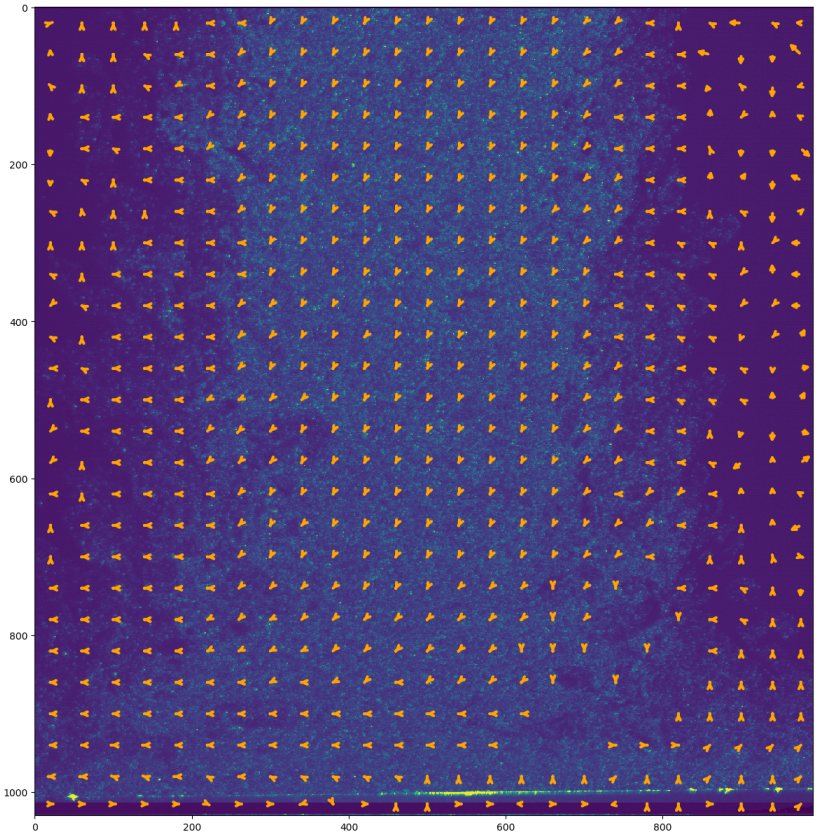


Figura 6: Mapa de desplazamientos.