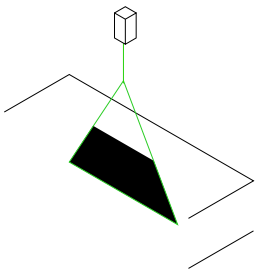
PYV: A PIV processing software in Python

Objectives

The aim of this work is to develop a full tool to pre-process and post-process PIV (Particle Image Velocimetry) images.

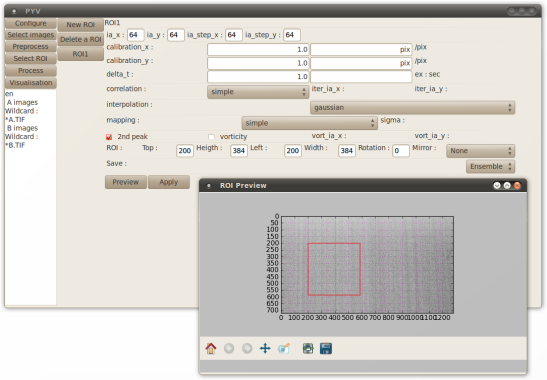
Particle Image Velocimetry (PIV) method

PIV is an optical non intrusive flow velocity measurement method. A CCD camera records two pictures (A and B) of a plane of a seeded flow illuminated by two successive laser layer pulses. Recordings are then divided into small interrogation areas (IA). For each IA an inter-correlation is applied between the IA A and the IA B that gives the local displacement. Knowing both laser pulse delay Δt and the spatial calibration allows to obtain velocity vectors field.



Presentation of PYV

PYV is a software programmed in Python language. It is used to preprocess and process PIV over a large range of PIV pairs of images. PYV has a Graphical User Interface (GUI).

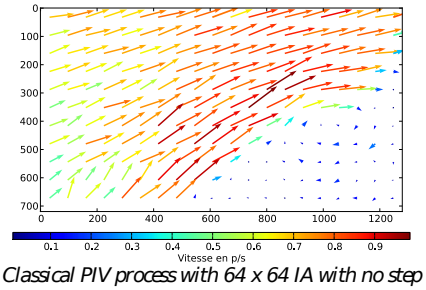


Regions of interest (ROI) can be cropped, images can be flipped and/or rotated. Several parameters allow the user to apply different kinds of PIV processing: The user can choose the correlation scheme (single or iterative). It is possible to configure the Interogation Areas (IA) mesh, so that one can obtain a high spatial repartition of vectors (up to one vector per pixel). An advantage of PYV is that several ROI can be defined on a single image with different parameters. The user can apply different kinds of processing to the same flow, taking into account its particularities.

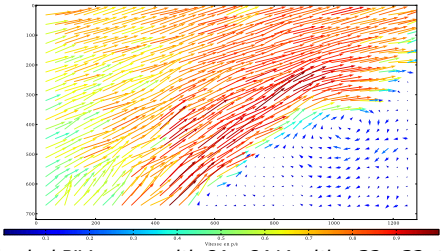
Librairies

- The GUI is built with *wxpython*.
- Calculations are done using *numpy* and *scipy*.
- Image processing uses *PIL* and *scipy.ndimage*.
- Prints are displayed with *matplotlib*.

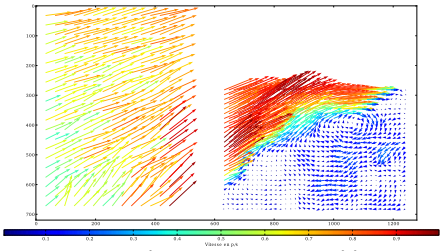
Examples



Classical PIV process with 64 x 64 IA with no step



Classical PIV process with 64 x 64 IA with a 32 x 32 step



Two different ROI on a same image: a PIV process with 64 x 64 IA with a 32 x 32 step; a PIV process with 64 x 64 IA with a 16 x 16 step

Perspectives

- New functions will be implemented such as spatial calibration depending on the area of the image.
- The code has been developped to work on a cluster without GUI. A next step will be to offer a public web interface, so users will process their images with FEMTO-ST cluster.

contacts

Dimitri Bonnet, Magali Barthès, Laurent Girardot, Yannick Bailly

mail: dimitri.bonnet@gmail.com
magali.barthes@univ-fcomte.fr
laurent.girardot@univ-fcomte.fr
yannick.bailly@univ-fcomte.fr