Py2DIC:
Python software for 2D Digital Image Correlation (DIC)

Contents

1 Py2DIC software

Py2DIC is a free open-source software developed at the Geodesy and Geomatics Division of Sapienza University of Rome for 2D Digital Image Correlation (DIC). It is based on the template matching method and convolution algorithm to compute 2D displacement and strain fields. The software compares a series of images of a planar surface collected at different stages of deformation and it tracks the pixel movement inside the Area of Interest (AOI) using template matching algorithms (Figure 1.1).

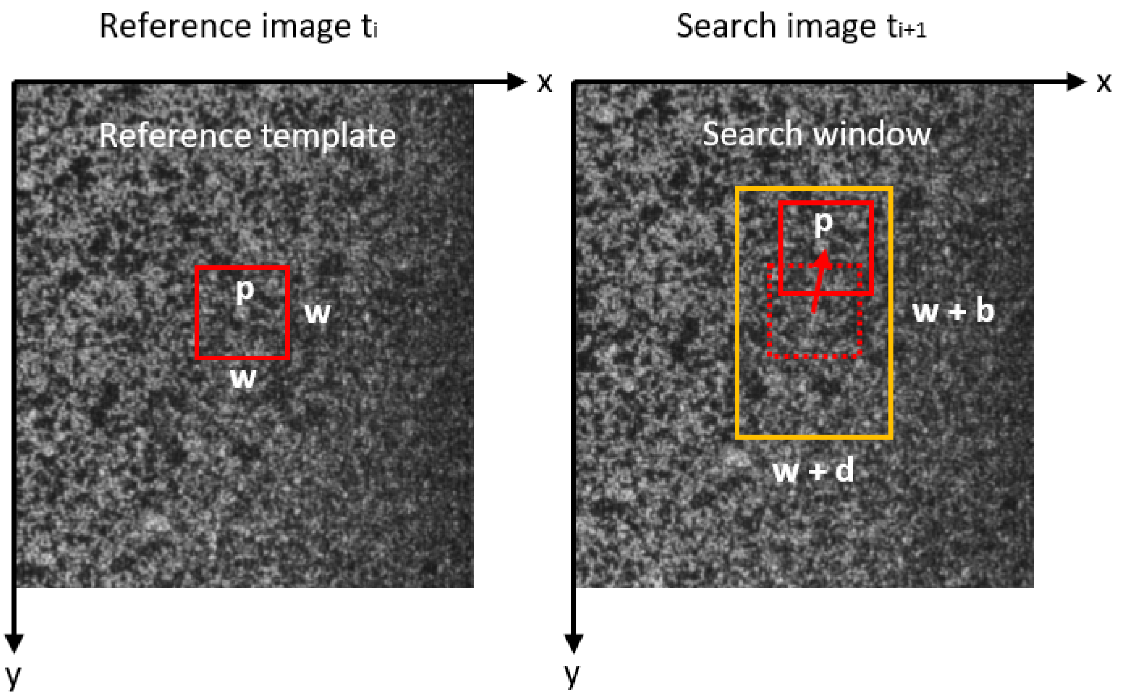

Figure 1.1: Template matching.

Once the displacements are computed, a Gaussian filter is applied to smooth the displacements estimated at each pixel to reduce the noise. Then, the smoothed displacements are differentiated using the centred difference approximation, for which the weights (up to the eighth order of accuracy) are generated following the approach described in [3]. Finally, the Green Lagrangian strains are computed. For details on Py2DIC displacement and strain field computation refer to [1].

The software is completely written in Python and the source code is freely available at http://github.com/Geod-Geom/py2DIC. Py2DIC is provided with a Graphical User Interface (GUI) and leverages the potentialities of OpenCV [2]. The software allows users to set the main input parameters for displacement and strain computation such as template, search window and multilook (see Figure 1.2).

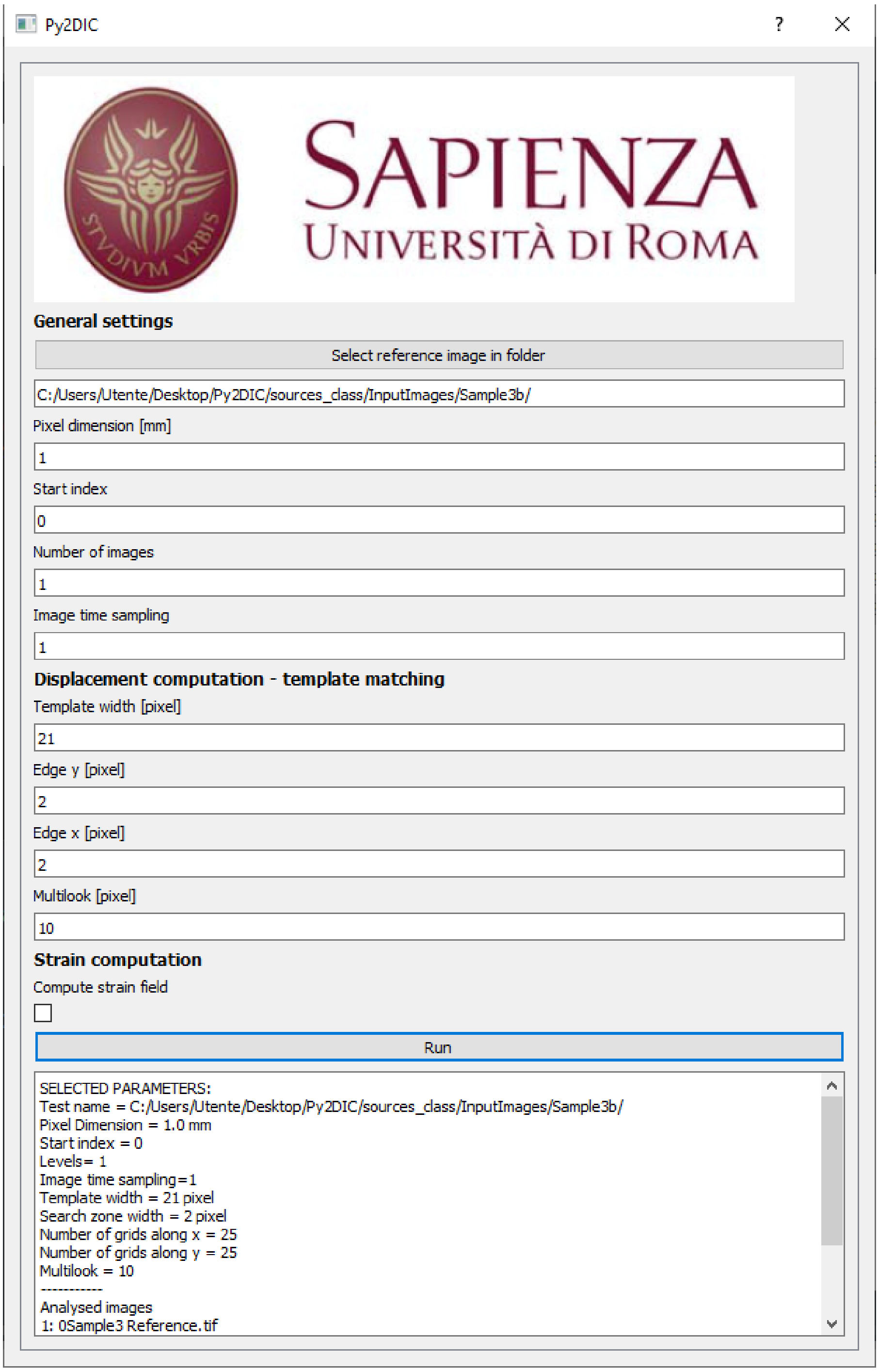

Figure 1.2: Py2DIC GUI.

At the end of the processing, Py2DIC returns the displacement and strain fields (Green Lagrangian strains) inside the AOI. An example of Py2DIC outcomes is shown in Figures 1.3 and 1.4. The results present the horizontal and vertical displacement fields (Figure 1.3) and the central vertical section of the sample surface (Figures 1.4).

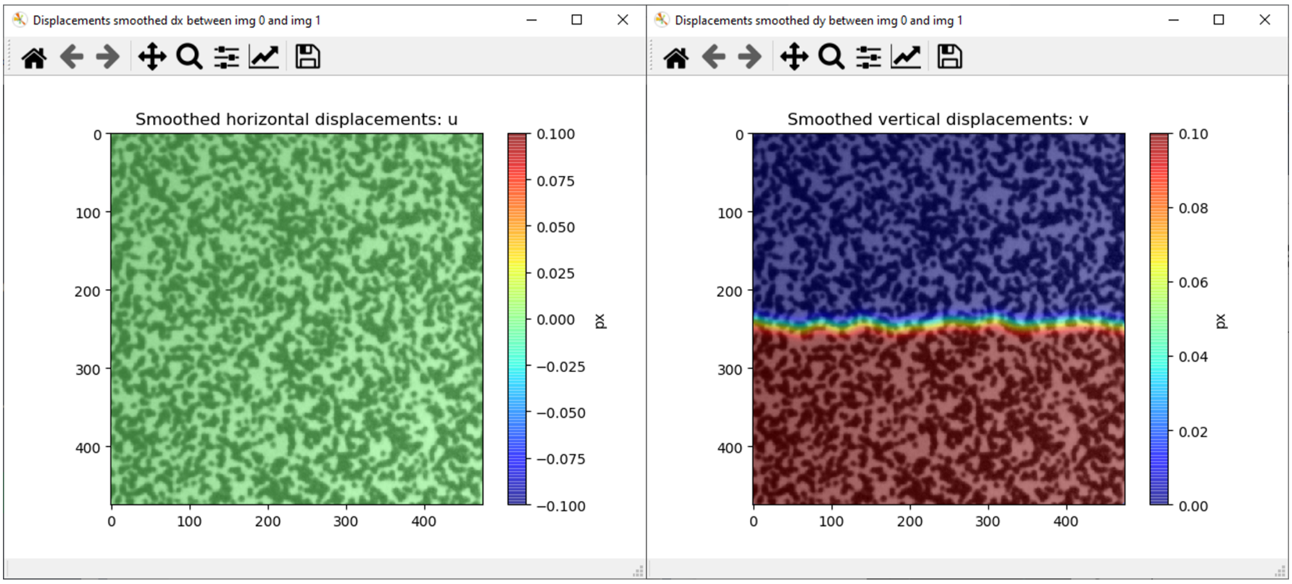

Figure 1.3: Py2DIC outcomes.

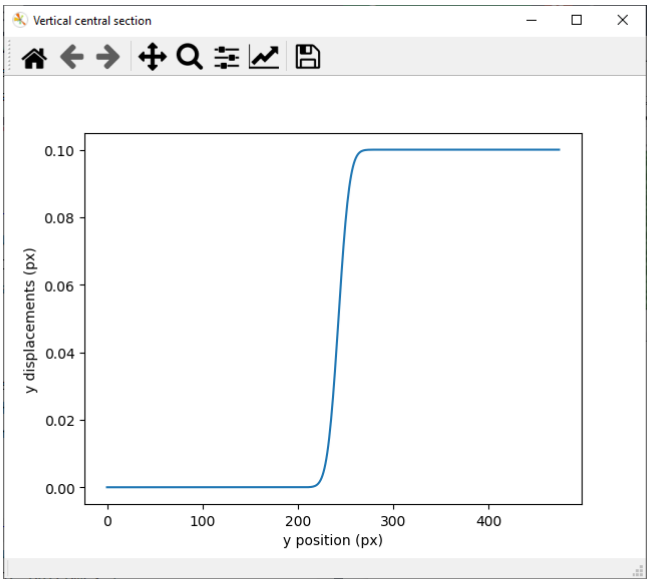

Figure 1.4: Py2DIC outcomes.

2 Py2DIC installation

- Install Anaconda following the instructions available at https://docs.anaconda.com/anaconda/install/
- Open the Anaconda Prompt

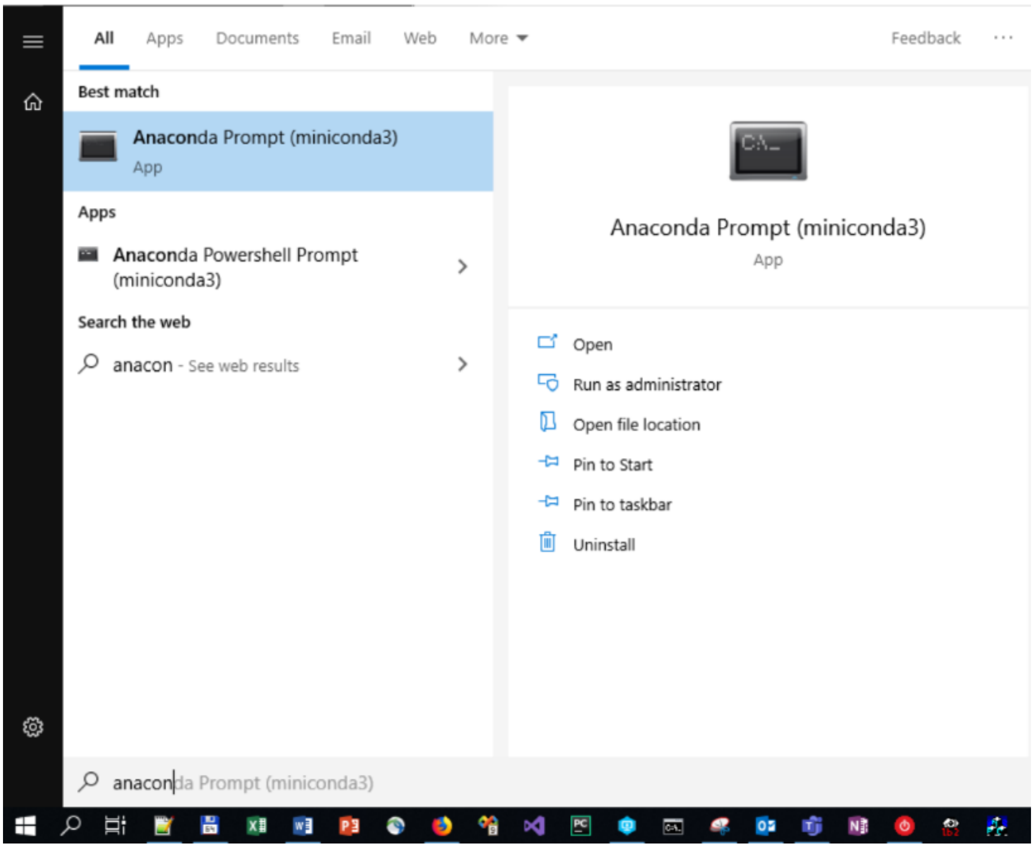

Figure 2.1: Open the Anaconda Prompt.

- Create an environment named py2DIC in the Anaconda Prompt using the command: *conda create --name py2DIC*

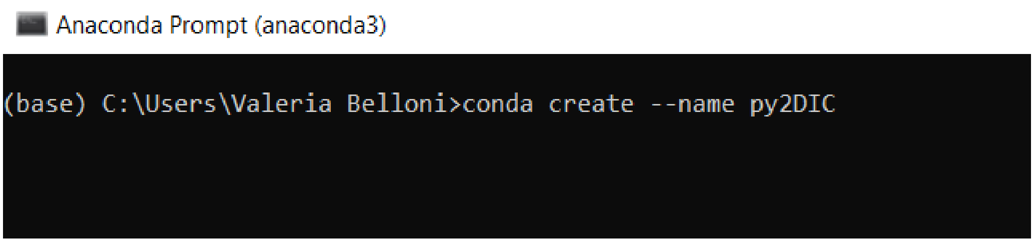

Figure 2.2: Create the environment in the Anaconda Prompt.

- Activate the environment in the Anaconda Prompt using the command:
  conda activate py2DIC

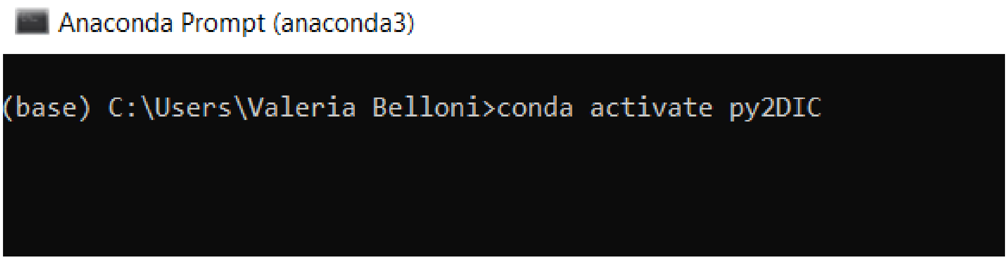

Figure 2.3: Activate the environment in the Anaconda Prompt.

- Install all the dependencies in the Anaconda Prompt using the following commands:
  conda install conda-forge::opencv
  conda install conda-forge::matplotlib
  conda install anaconda::scipy
  conda install conda-forge::joblib
  conda install astropy

  If conda installation does not work for Opencv use:
  pip install opencv-python

3 Py2DIC usage

You can run the software through the provided batch file or the Anaconda Prompt.

3.0.1 Run via batch file

- Open the batch file (run_py2DIC.bat) available in the folder, change the paths highlighted in Figure 3.1 and save the file. The first path refers to the location of the conda environment, the second is the path to the folder downloaded from GitHub
- Double-click the batch file (run_py2DIC.bat) to launch the Py2DIC GUI (Figure 3.2)

```
@echo OFF
rem How to run a Python script in a given conda environment from a batch file.

rem It doesn't require:
rem - conda to be in the PATH
rem - cmd.exe to be initialized with conda init

rem Define here the path to your conda installation
set CONDAPATH="C:\Users\Utente\anaconda3\"
rem Define here the name of the environment
set ENVNAME=py2DIC

rem The following command activates the base environment.
rem call C:\ProgramData\Miniconda3\Scripts\activate.bat C:\ProgramData\Miniconda3
if %ENVNAME%==base (set ENVPATH=%CONDAPATH%) else (set ENVPATH=%CONDAPATH%\envs\%ENVNAME%)

rem Activate the conda environment
rem Using call is required here, see: https://stackoverflow.com/questions/24678144/conda-environments-and-bat-files
call %CONDAPATH%\Scripts\activate.bat %ENVPATH%

rem Run a python script in that environment
REM Start the FEED2 GUI
python.exe "C:\Users\Utente\Desktop\Py2DIC\sources_class\GUI.py"
IF %ERRORLEVEL% NEQ 0 PAUSE
rem Deactivate the environment
call conda deactivate

rem If conda is directly available from the command line then the following code works.
rem call activate someenv
rem python script.py
rem conda deactivate

rem One could also use the conda run command
rem conda run -n someenv python script.py
```

Figure 3.1: Batch file for running Py2DIC.

3.0.2 Run via Anaconda prompt

- Open the Anaconda Prompt and activate the py2DIC environment:
  conda activate py2DIC
- Navigate to the path of the Py2DIC folder using the command:
  cd "path to GUI.py"
- Use the following command to activate ipython:
  ipython
- Use the following command to launch the GUI (Figure 3.2):
  run GUI.py

3.0.3 Py2DIC settings

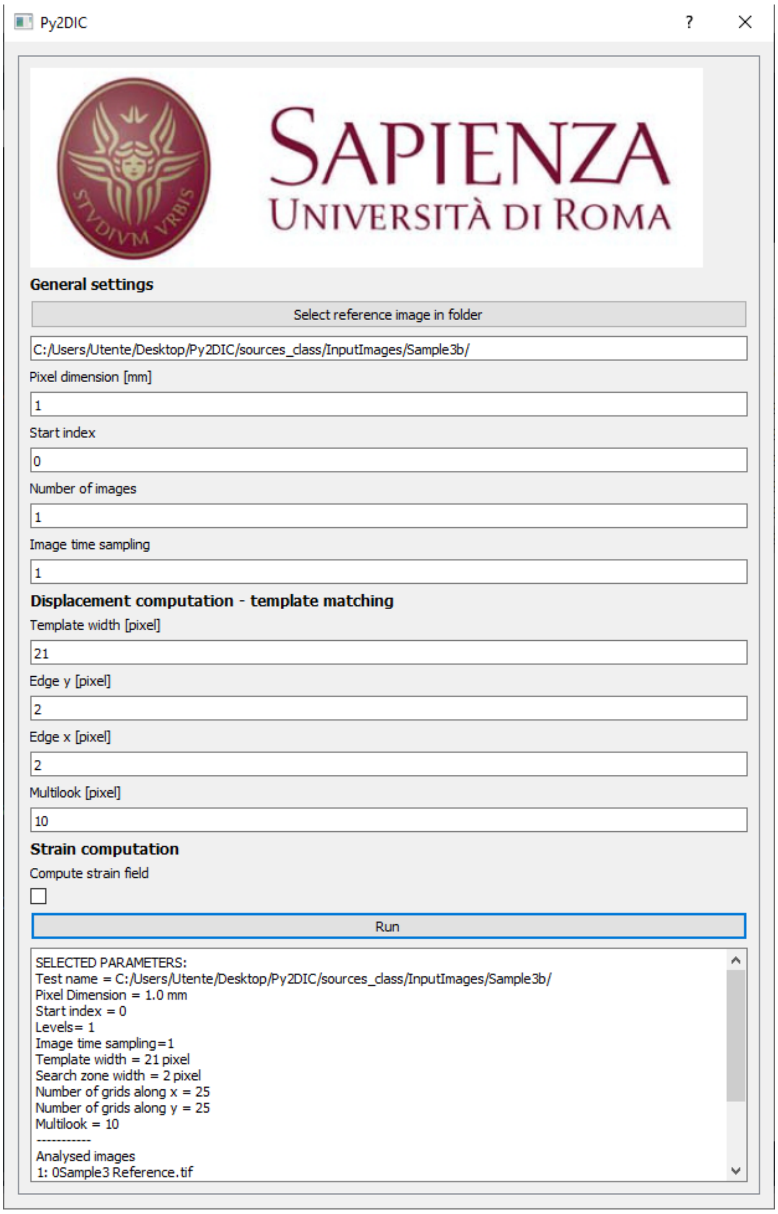

Figure 3.2: Grafical User Interface.

- Select the reference image in the folder using the available button
- Select the pixel dimension to convert the results from pixel to mm. If not available keep the default value 1. In this case, the results are provided in pixel units
- Start index refers to the first image. If set to 0 the reference image is the first image of the folder

- Number of images is set to 1 to process a single image pair. To process more than one image incrementally, increase this parameter
- Image time sampling enables the selection of the deformed image. If the start index is 0, the number of images is 1 and image time sampling is set to 1 the deformed image is the second image of the folder. If the start index is 0, the number of images is set to 1 and image time sampling is set to 2 the deformed image is the third image of the folder
- Select the template matching parameters. The template should be selected according to the features of the images
- Edges should be larger than the maximum expected displacements
- The multilook is used to reach the sub-pixel level and is 10 by default
- Decide if you want to compute the strains
- Once the parameters are set, run the program using the *Run* button

4 Py2DIC test

4.1 Samples from DIC Challenge

The sets of input images available to test Py2DIC are part of the Society for Experimental Mechanics (SEM) 2D DIC challenge datasets https://www.idics.org/challenge/. The DIC challenge provides common image datasets that can be used to validate and improve both commercial and academic DIC software solutions. A detailed explanation of the dataset, processing and results of the DIC challenge can be found at https://link.springer.com/article/10.1007/s11340-017-0349-0.

4.1.1 Sample 16

Sample 16 is a rigid-body experimental translation dataset. To process Sample 16 data, download the Sample16.zip folder at https://drive.google.com/drive/folders/1tNUKPJ7UJOm23JhERtkrIy5gSBiwV3Dj. Select the reference image (50pix-test1-0000_0-Dec-Avg 00000.tif) and the deformed image (50pix-test1-0050_0-Dec-Avg 00050.tif) from the Sample16.zip folder. Copy the two images into a new folder. Run Py2DIC and select the reference image. Set the following parameters and keep the other parameters by default:

- Template = 11 pixel
- Edges = 2 pixel
- Multilook = 100
- Do not computer strain

The expected results are shown in Figure 4.1. For a proper comparison change the min and max values of the colorbar as suggested in the figure.

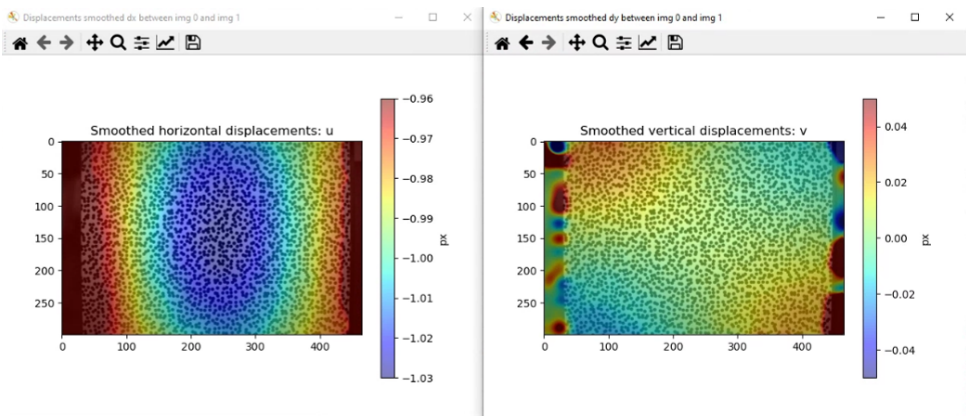

Figure 4.1: Py2DIC outcomes.

4.1.2 Sample 12

Sample 12 is an experimental set of 12 images (resolution of 400 × 1040 pixels) of a steel plate with a hole in the middle being loaded in tension. Regarding the specimen, a painted speckle pattern was used during the tensile test. As this is an experimental image series, there is no ground truth data but it is useful for round-robin type tests where different codes for DIC. To process Sample 16 data, download the Sample12.zip folder at https://drive.google.com/drive/folders/1tNUKPJ7UJOm23JhERtkrIy5gSBiwV3Dj. Select the reference image (ohtcfrp_00.png) and the deformed image (ohtcfrp_11.png) and copy the images into a new folder. Keep everything by default and run the program.
The expected results are shown in Figure 4.2. The expected results are shown in Figure 4.1. For a proper comparison change the min and max values of the colorbar as suggested in the figure.

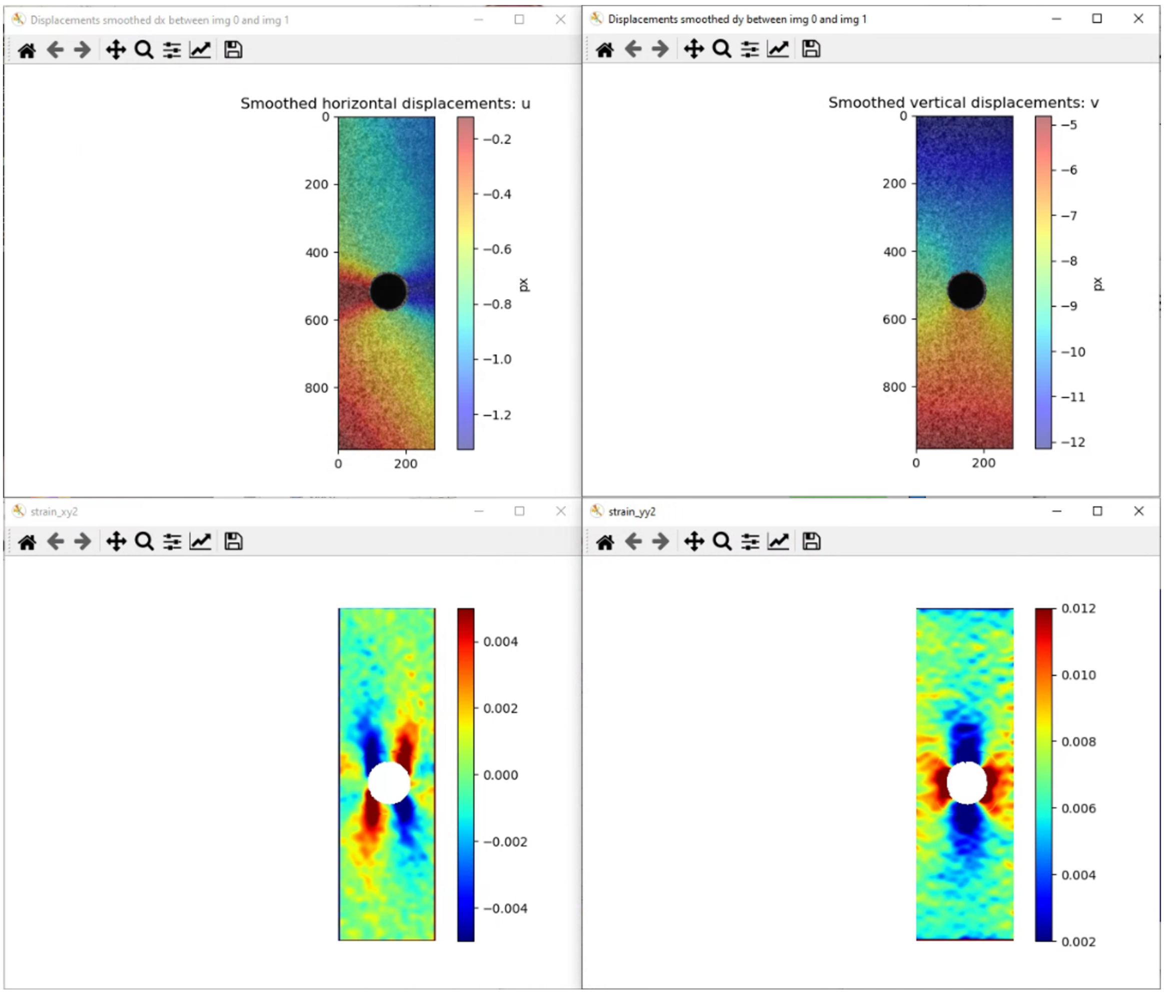

Figure 4.2: Py2DIC outcomes.

4.1.3 Sample 3b

Sample 3b and Sample 3 are examples of good contrast and low noise images shifted using the FFT function. Sample 3 is a rigid body shift of 0.1 pixels per image and will nicely reveal both the DIC interpolation bias and variance. Sample 3b (resolution of 500 × 500 pixels) uses the same FFT shift, however, only the bottom half of the image is shifted. Sample 3b can be used to look at the effect of discontinuities in the results [4]. To process Sample 3b data, download the Sample16.zip folder at https://drive.google.com/drive/folders/1tNUKPJ7UJOm23JhERtkrIy5gSBiwV3Dj. Select the reference image (0Sample3 Reference.tif) and the deformed image (Sample3 Half01.tif). Rename the image Sample3 Half01.tif as 2Sample3 Half01.tif and copy the images into a new folder. Set the following parameters and keep the other parameters by default:

- Template = 21 pixel
- Edges = 2 pixel
- Multilook = 10
- Do not computer strain

The expected results are shown in Figures 4.3 and 4.4.

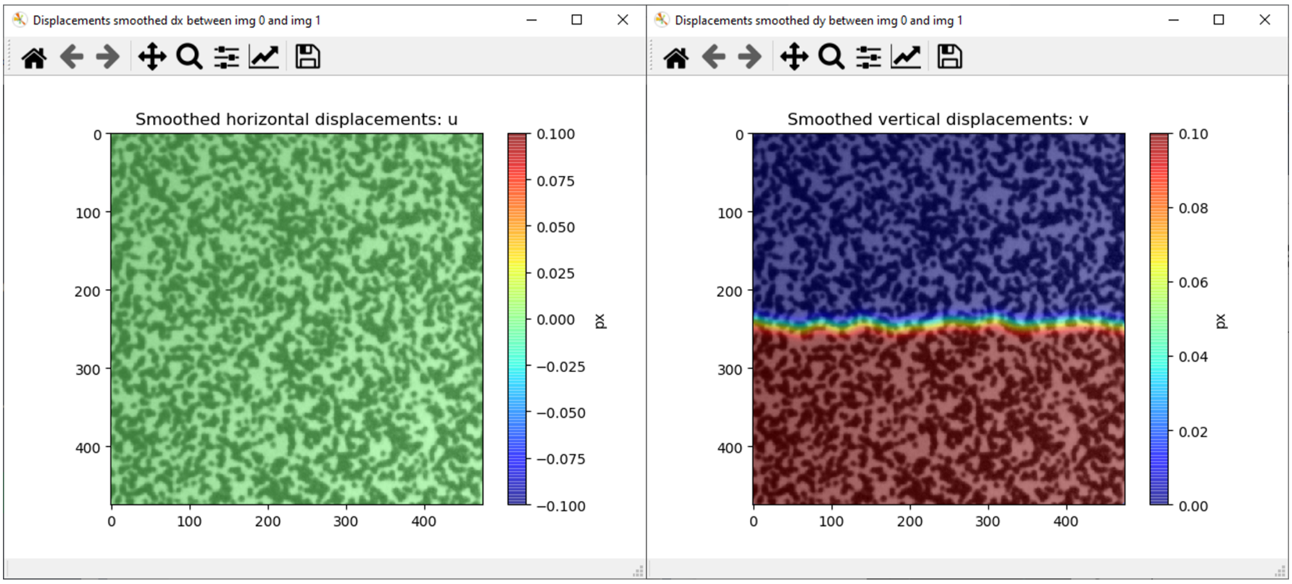

Figure 4.3: Py2DIC outcomes.

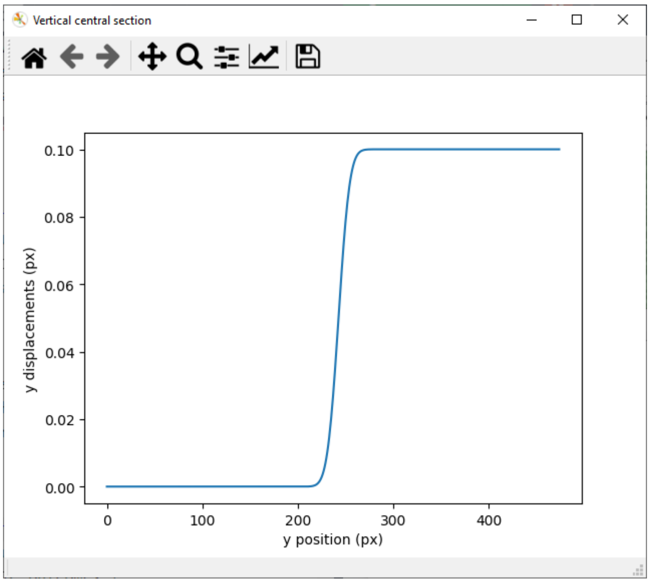

Figure 4.4: Py2DIC outcomes.