OpenPivGui

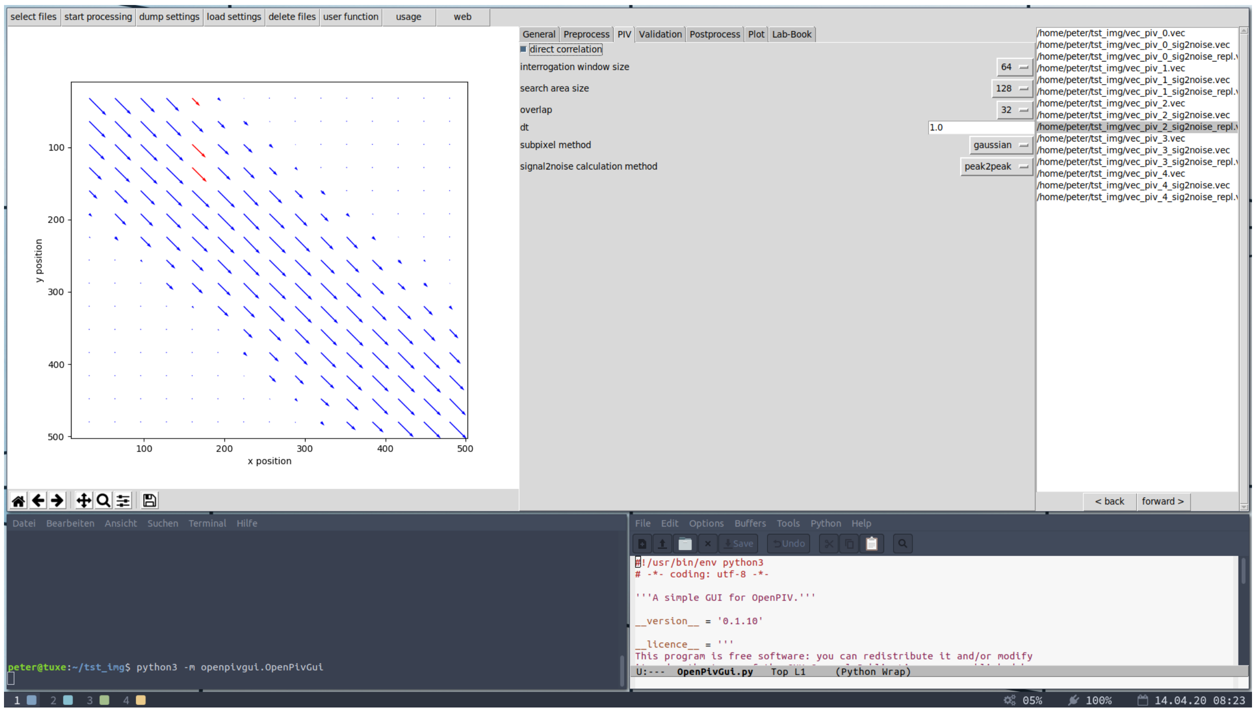
Release 0.2.9

OpenPivGui Community

Sep 25, 2020

CONTENTS

OpenPivGui is a graphical user interface, providing an efficient workflow for evaluating and postprocessing particle image velocimetry (PIV) images. OpenPivGui relies on the Python libraries provided by the OpenPIV project.



- [✓ direct correlation]
- interrogation window size: 64
- search area size: 128
- overlap: 32
- dt: 1.0
- subpixel method: gaussian
- signal2noise calculation method: peak2peak

```
/home/peter/tst_img/vec_piv_0.vec
/home/peter/tst_img/vec_piv_0_sig2noise.vec
/home/peter/tst_img/vec_piv_0_sig2noise_repl.v
/home/peter/tst_img/vec_piv_1.vec
/home/peter/tst_img/vec_piv_1_sig2noise.vec
/home/peter/tst_img/vec_piv_1_sig2noise_repl.v
/home/peter/tst_img/vec_piv_2.vec
/home/peter/tst_img/vec_piv_2_sig2noise.vec
/home/peter/tst_img/vec_piv_2_sig2noise_repl.v
/home/peter/tst_img/vec_piv_3.vec
/home/peter/tst_img/vec_piv_3_sig2noise.vec
/home/peter/tst_img/vec_piv_3_sig2noise_repl.v
/home/peter/tst_img/vec_piv_4.vec
/home/peter/tst_img/vec_piv_4_sig2noise.vec
/home/peter/tst_img/vec_piv_4_sig2noise_repl.v
```

```
peter@tuxe:~/tst_img$ python3 -m openpivgui.OpenPivGui
```

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

'''A simple GUI for OpenPIV.'''

__version__ = '0.1.10'

__licence__ = '''
This program is free software: you can redistribute it and/or modify
U:---  OpenPivGui.py   Top L1     (Python Wrap)
```

```
1  2  3  4                                    05%    100%    14.04.20 08:23
```

Fig. 1: OpenPivGui.

CHAPTER

ONE

INSTALLATION

1.1 Python Package Index

You may use `pip` to install OpenPivGui:

```
pip3 install openpivgui
```

1.2 Launching

Launch OpenPivGui by executing:

```
python3 -m openpivgui.OpenPivGui
```

CHAPTER

TWO

USAGE

2.1 Video Tutorial

Learn how to use and extend OpenPivGui in less than eight minutes in a video tutorial.

2.2 Workflow

1. Press the button »open directory« and choose a directory that contains some PIV images.
2. To inspect the images, click on the links in the file-list on the right side of the OpenPivGui window.
3. Walk through the riders, select the desired functions, and edit the corresponding parameters.
4. Press »start processing« to start the evaluation.
5. Inspect the results by clicking on the links in the file-list.
6. Use the »back« and »forward« buttons to inspect intermediate results. Use the »back« and »forward« buttons also to list the image files again, and to repeat the evaluation.
7. Use »dump settings« to document your project. You can recall the settings anytime by pressing »load settings«. The lab-book entries are also restored from the settings file.

2.3 Adaption

First, get the source code. There are two possibilities:

1. Clone the git repository:

   ```
   git clone https://github.com/OpenPIV/openpiv_tk_gui.git
   ```

2. Find out, where pip3 placed the source scripts and edit them in place:

   ```
   pip3 show openpivgui
   ```

In both cases, cd into the subdirectory `openpivgui` and find the main scripts to edit:

- `OpenPivParams.py`
- `OpenPivGui.py`

Usually, there are two things to do:

1. Adding new variables and a corresponding widgets to enable users to modify its values.

2. Adding a new method (function).

2.3.1 Adding new variables

Open the script `OpenPivParams.py`. Find the method `__init__()`. There, you find a variable, called `default` of type dict. All widgets like checkboxes, text entries, and option menus are created based on the content of this dictionary.

By adding a dictionary element, you add a variable. A corresponding widget is automatically created. Example:

```
'corr_window':              # key
     [3020,                 # index
     'int',                 # type
     32,                    # value
     (8, 16, 32, 64, 128),  # hints
     'window size',         # label
     'Size in pixel.'],     # help
```

In `OpenPivGui.py`, you access the value of this variable via `p['corr_window']`. Here, `p` is the instance name of an `OpenPivParams` object. Typing:

```
print(self.p['corr_window'])
```

will thus result in:

```
32
```

The other fields are used for widget creation:

- index: An index of 3xxx will place the widget on the third rider (»PIV«).
- **type:**
  - **`None`: Creates a new notebook rider.**
    * `bool`: A checkbox will be created.
    * `str[]`: Creates a listbox.
    * `text`: Provides a text area.
    * `float`, `int`, `str`: An entry widget will be created.
- hints: If hints is not `None`, an option menu is provided with `hints` (tuple) as options.
- label: The label next to the manipulation widget.
- help: The content of this field will pop up as a tooltip, when the mouse is moved over the widget.

2.3.2 Adding a new method

Open the script `OpenPivGui`. There are two main possibilities, of doing something with the newly created variables:

1. Extend the existing processing chain.
2. Create a new method.

Extend existing processing chain

Find the function definition `start_processing()`. Add another `if` statement and some useful code.

Create a new method

Find the function definition `__init_buttons()`. Add something like:

```
ttk.Button(f,
           text='button label',
           command=self.new_func).pack(
                   side='left', fill='x')
```

Add the new function:

```
def new_func(self):
    # do something useful here
    pass
```

2.3.3 Testing

Overwrite the original scripts in the installation directory (locate the installation directory by `pip3 show openpivgui`) with your altered version and test it. There are test images in the OpenPivGui Github repository, if needed.

2.4 Reusing code

The openpivgui modules and classes can be used independently from the GUI. The can be used in other scipts or jupyter notebooks and some can be called from the command line directly.

2.5 Troubleshooting

I can not install OpenPivGui. Try `pip` instead of `pip3` or try the `--user` option:

```
pip install --user openpivgui
```

Did you read the error messages? If there are complaints about missing packages, install them prior to OpenPivGui:

```
pip3 install missing-package
```

Something is not working properly. Ensure, you are running the latest version:

```
pip3 install --upgrade openpivgui
```

Something is still not working properly. Start OpenPivGui from the command line:

```
python3 -m openpivgui.OpenPivGui
```

Check the command line for error messages. Do they provide some useful information?

I can not see a file list. The GUI may hide some widgets. Toggle to full-screen mode or try to check the »compact layout« option on the »General« rider.

I do not understand, how the »back« and »forward« buttons work. All output files are stored in the same directory as the input files. To display a clean list of a single processing step, the content of the working directory can be filtered. The »back« and »forward« buttons change the filter. The filters are defined as a list of comma separated regular expressions in the variable »navigation pattern« on the »General« tab.

Examples:

`png$` Show only files that end on the letters »png«.

`piv_[0-9]+\.vec$` Show only files that end on `piv_`, followed by a number and `.vec`. These are usually the raw results.

`sig2noise_repl\.vec$` Final result after applying a validation based on the signal to noise ratio and filling the gaps.

You can learn more about regular expressions by reading the Python3 Regular Expression HOWTO.

I get »UnidentifiedImageError: cannot identify image file« This happens, when a PIV evaluation is started and the file list contains vector files instead of image files. Press the »back« button until the file list contains image files.

I get »UnicodeDecodeError: 'utf-8' codec can't decode byte 0xff in position 85: invalid start byte« This happens, when PIV evaluation is NOT selected and the file list contains image files. Either press the »back button« until the file list contains vector files or select »direct correlation« on the PIV rider.

CHAPTER

THREE

PARAMETERS

3.1 General

filenames None

number of cores Select amount of cores to be used for PIV evaluations.

sequence order step Select sequence order step for evaluation. Assuming >>skip<< = 1; >>1<< yields (1+2),(2+3) >>2<< yields (1+2),(3+4)

sequence order skip Select sequence order jump for evaluation. Assuming >>step<< = 1; >>1<< yields (1+2),(2+3) >>2<< yields (1+3),(2+4) >>3<< yields (1+4),(2+5) and so on…

compact layout If selected, the layout is optimized for full screen usage and small screens. Otherwise, the layout leaves some horizontal space for other apps like a terminal window or source code editor. This setting takes effect after restart.

base output filename Filename for vector output. A number and an acronym that indicates the process history are added automatically.

navigation pattern Regular expression patterns for filtering the files in the current directory. Use the back and forward buttons to apply a different filter.

settings for using pandas Individual settings for loading files using pandas.

skip rows Number of rows skipped at the beginning of the file.

decimal separator Decimal separator for floating point numbers.

column separator Column separator.

read header Read header. If chosen, first line will be interpreted as the header

specify own header names Specify comma separated list of column names.Example: x,y,vx,vy,sig2noise

3.2 Pre-Processing

region of interest Define region of interest.

x min Defining region of interest.

x max Defining region of interest.

y min Defining region of interest.

y max Defining region of interest.

invert image Invert image (see skimage invert()).

Gaussian filter Standard Gaussian blurring filter (see scipy gaussian_filter()).

sigma/kernel size Defining the size of the sigma/kernel for gaussian blur filter.

CLAHE filter Contrast Limited Adaptive Histogram Equalization filter (see skimage adapthist()).

kernel size Defining the size of the kernel for CLAHE.

clip limit Defining the contrast with 0-1 (1 gives highest contrast).

UnSharp high pass mask/filter A simple image high pass filter (see skimage un_sharp()).

perform before CLAHE Perform UnSharp high pass mask/filter before CLAHE.

filter radius Defining the radius value of the subtracted gaussian filter in the UnSharp high pass mask/filter (positive ints only).

clip limit Defining the clip of the UnSharp filter (higher values remove more background noise).

dynamic masking Dynamic masking for masking of images. Warning: This is still in testing and is not recommended for use.

mask type Defining dynamic mask type.

mask threshold Defining threshold of dynamic mask.

mask filter size Defining size of the masks.

3.3 PIV Evaluation

do PIV evaluation Do PIV evaluation, select method and parameters below. Deselect, if you just want to do some post-processing.

evaluation method extd: Direct correlation with extended size of the search area. widim: Window displacement iterative method. (Iterative grid refinement or multi pass PIV). windef: Iterative grid refinement with window deformation (recommended).

search area size Size of square search area in pixel for extd method.

interrogation window size Size of square interrogation windows in pixel (final pass, in pixel).

overlap Overlap of correlation windows or vector spacing (final pass, in pixel).

number of refinement steps Example: A window size of 16 and a number of refinement steps of 2 gives an window size of 64×64 in the fist pass, 32×32 in the second pass and 16×16 pixel in the final pass. (Applies to widim and windef methods only.)

correlation method Correlation method. Circular is no padding andlinear is zero padding (applies to Windef).

subpixel method Fit function for determining the subpixel position of the correlation peak.

signal2noise calculation method Calculation method for the signal to noise ratio.

dt Interframing time in seconds.

scale Interframing scaling in pix/m

invert u-component Invert (negative) u-component when saving RAW results.

invert v-component Invert (negative) v-component when saving RAW results.

3.4 Validation

signal to noise ratio validation Validate the data based on the signal to nose ratio of the cross correlation.

signal to noise threshold Threshold for filtering based on signal to noise ratio.

standard deviation validation Validate the data based on a multiple of the standard deviation.

standard deviation threshold Remove vectors, if the the sum of the squared vector components is larger than the threshold times the standard deviation of the flow field.

local median validation Validate the data based on a local median threshold.

local median threshold Discard vector, if the absolute difference with the local median is greater than the threshold.

global threshold validation Validate the data based on a set global thresholds.

min u Minimum U allowable component.

max u Maximum U allowable component.

min v Minimum V allowable component.

max v Maximum V allowable component.

3.5 Post-Processing

replace outliers Replace outliers.

replacement method Each NaN element is replaced by a weighed averageof neighbours. Localmean uses a square kernel, disk a uniform circular kernel, and distance a kernel with a weight that is proportional to the distance.

number of iterations If there are adjacent NaN elements, iterative replacement is needed.

kernel size Diameter of the weighting kernel.

smoothn data Smoothn data using openpiv.smoothn.

smoothn vectors Smoothn data with openpiv.smoothn. <each pass> only applies to windef

smoothn each pass Smoothn each pass using openpiv.smoothn.

smoothn robust Activate robust in smoothn (minimizes influence of outlying data).

smoothning strength Strength of smoothn script. Higher scalar number produces more smoothned data.

3.6 Plotting

plot type Select, how to plot velocity data.

vector scaling Velocity as a fraction of the plot width, e.g.: m/s per plot width. Large values result in shorter vectors.

vector line width Line width as a fraction of the plot width.

invalid vector color The color of the invalid vectors

valid vector color The color of the valid vectors

vector plot invert y-axis Define the top left corner as the origin of the vector plot coordinate sytem, as it is common practice in image processing.

histogram quantity The absolute value of the velocity (v) or its x- or y-component (v_x or v_y).

histogram number of bins Number of bins (bars) in the histogram.

histogram log scale Use a logarithmic y-axis.

profiles orientation Plot v_y over x (horizontal) or v_x over y (vertical).

Use pandas plot utility. If chosen, plots will be generated with pandas.

plot-type Choose plot-type. For further information refer to pandas.DataFrame.plot().

column name for x-data column name for x-data. If unknown watch labbook entry.

column name for y-data column name for y-data. If unknown watch labbook entry. For histogram only y_data are needed.

number of bins number of bins. This box is only used for plotting type scatter.

diagram title diagram title.

grid adds a grid to the diagram.

legend adds a legend to the diagram.

axis scaling scales the axes. logarithm scaling x-axis –> logx; logarithm scaling y-axis –> logy; logarithm scaling both axes –> loglog.

limits for the x-axis For implementation use (lower_limit, upper_limit).

limits for the y-axis For implementation use (lower_limit, upper_limit).