Last edit: February 07, 2024

CalVi

Calibration Visualizer

User Guide

PaIRS – version 0.1.13

email: etfd@unina.it

Experimental Thermo-Fluid Dynamics (ETFD) group,
Department of Industrial Engineering (DII)
University of Naples “Federico II”, Naples, Italy

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

DIPARTIMENTO DI
INGEGNERIA
INDUSTRIALE

Contents

1. Introduction

PaIRS-UniNa is a project developed by the Experimental Thermo Fluid-Dynamics (ETFD) group of University of Naples "Federico II" since 2000. It is aimed to provide fast and efficient tools for digital particle image velocimetry (PIV) analysis in research and industrial applications.

PaIRS-UniNa is based on a C library (**PaIRS-PIV**) and relies on a graphical user interface (**PaIRS**) that is developed via PySide6 and makes the use of PaIRS-PIV easy and intuitive. PaIRS-PIV includes several modules that allow to process double-frame or single-frame 2D planar PIV images as well as stereoscopic and tomographic PIV or Lagrangian particle tracking velocimetry (4D PTV) measurements. The current release of PaIRS-UniNa features only the module for the 2D planar PIV analysis.

PaIRS-UniNa is supported by Python 3.8+ and is compatible with all the operating systems, however, the PaIRS-PIV library relies on OpemMP library, which must be installed on the macOS platform (see section Installation). On the other side, PaIRS requires, among other packages, SciPy and matplotlib.

CalVi is the calibration module of **PaIRS-UniNa** and allows accurate optical calibration of single and multiple camera bundles with the camera models mostly used in the PIV community: polynomials, rational functions and the pinhole camera model. Among the other features, it supports camera calibration procedures working with unknown positions and orientations of the calibration target and the integration of the pinhole camera model with a refractive correction model for cylindrical geometries (based on ray-tracing and Snell’s law).

2. Related works

Please cite the following works if you are intended to use CalVi for your purposes:

[1] Paolillo, G., & Astarita, T. (2020). "Perspective camera model with refraction correction for optical velocimetry measurements in complex geometries". *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 44(6), 3185-3196**.** doi: 10.1109/TPAMI.2020.3046467.

[2] Paolillo, G., & Astarita, T. (2021). "On the PIV/PTV uncertainty related to calibration of camera systems with refractive surfaces". *Measurement Science and Technology*, 32(9), 094006. doi: 10.1088/1361-6501/abf3fc.

If you use PaIRS also acknowledge the following works::

[1] Astarita, T., & Cardone, G. (2005). "Analysis of interpolation schemes for image deformation methods in PIV". *Experiments in Fluids*, 38(2), 233-243. doi: 10.1007/s00348-004-0902-3.

[2] Astarita, T. (2006). "Analysis of interpolation schemes for image deformation methods in PIV: effect of noise on the accuracy and spatial resolution". *Experiments in Fluids*, vol. 40 (6): 977-987. doi: 10.1007/s00348-006-0139-4.

[3] Astarita, T. (2007). "Analysis of weighting windows for image deformation methods in PIV." *Experiments in Fluids*, 43(6), 859-872. doi: 10.1007/s00348-007-0314-2.

[4] Astarita, T. (2008). "Analysis of velocity interpolation schemes for image deformation methods in PIV". *Experiments in Fluids*, 45(2), 257-266. doi: 10.1007/s00348-008-0475-7.

[5] Astarita, T. (2009). "Adaptive space resolution for PIV". *Experiments in Fluids*, 46(6), 1115-1123. doi: 10.1007/s00348-009-0618-5.

3. Installation

All PaIRS-UniNa wheels are distributed under LGPLv3+ licences. For PaIRS installation, you must first download and install Python. You can follow the tutorial at https://realpython.com/installing-python/ to install or update the official Python distributions on Windows, macOS or Linux. After that, the PaIRS installation can be performed from a command-line application with:

```
python -m pip install PaIRS-UniNa
```

On macOS and Linux `python` must be replaced by `python3`.
Normally the OpenMP library is not preinstalled in MacOs. A possible way to install this library is:

```
curl -O https://mac.r-project.org/openmp/openmp-12.0.1-darwin20-Release.tar.gz
sudo tar fvxz openmp-12.0.1-darwin20-Release.tar.gz -C /
```

Please, notice that on Windows 7 you should install Python 3.8 and PySide 6.1.3 version. For this purpose, after the Python and PaIRS installations (as above explained), execute the following commands:

```
python -m pip uninstall PySide6
python -m pip install PySide6==6.1.3
```

4. Run

From command prompt

It is possible to run CalVi directly from the command prompt with:

```
python -m PaIRS_UniNa -calvi
```

CalVi automatically saves and stores its configuration upon exit and starts from the latter at the next run. If any trouble with loading the last configuration file (saved in the package folder) occurs, the user is suggested to execute a clean run of CalVi via the following command:

```
python -m PaIRS_UniNa -calvi -c
```

A debug mode is also available for developers. It can be accessed via:

```
python -m PaIRS_UniNa -calvi -d
```

After the above command, the user will be asked to enter a password. Interested users can ask the password to the authors by sending an email to: etfd@unina.it. The debug mode can be turned on/off at any time via the keyboard sequence: `Alt+Shift+D`.
On macOS and Linux `python` must be replaced by `python3`.

In Python environment

CalVi can be run also in Python environment. For this purpose, open a command prompt and launch Python:

```
python
```

and then use the following commands for normal run:

```
>>> from PaIRS_UniNa import CalVi
>>> CalVi.run()
```

For clean mode:

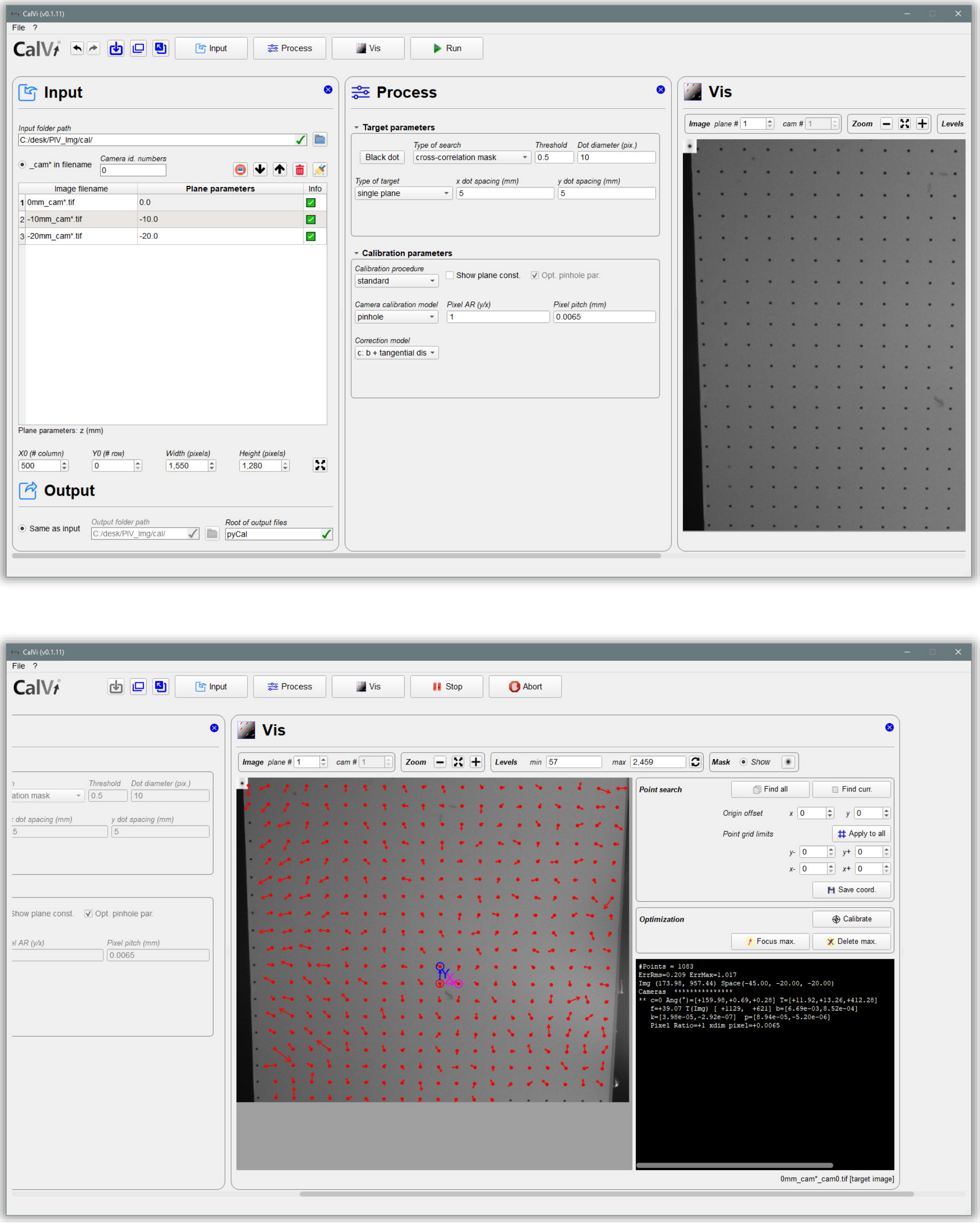
```
>>> CalVi.cleanRun()
```

while for debug mode:

```
>>> CalVi.debugRun()
```

5. Main Window

CalVi main window looks like in Figure 1.



```
#Points = 1083
ErrRms=0.209 ErrMax=1.017
Img (173.98, 957.44) Space(-45.00, -20.00, -20.00)
Cameras  ***************
** c=0 Ang(°)=[+159.98,+0.69,+0.28] T=[+11.92,+13.26,+412.28]
   f=+39.07 T(Img) [ +1129,  +621] b=[6.69e-03,8.52e-04]
   k=[3.98e-05,-2.92e-07]  p=[8.94e-05,-5.20e-06]
   Pixel Ratio=+1 xdim pixel=+0.0065
```

Figure 1 – CalVi main window.

Several tabs are included in the main window and allow the user to specify the image and process settings before launching the interactive procedure to calibrate the selected camera model. The Input/Output and Process tabs are used to specify the parameters of the image files and the output files produced by CalVi (including their full paths) and the settings of the calibration process to be performed. The Vis tab is the core of the calibration process and here the user can play to identify the control points in the target images and optimize the mapping functions of the selected camera model; log window is included in this tab to visualize output from the PaIRS-PIV library and further error/warning messages. Finally, the Button bar is used to hide/show or dock/undock the other above-mentioned tabs and to manage the interface geometry. CalVi has also a Menu bar, where further functions are provided.

6. Menu bar

The Menu bar consists of the section , File and Help (?), as show in Figure 2.

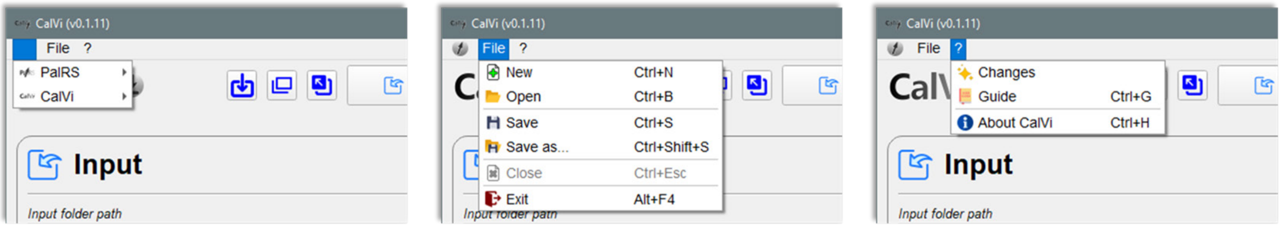


Figure 2 – Menu bar.

From this section, new instances of PaIRS and CalVi can be launched:

- *Normal run*: launch a new instance of PaIRS/CalVi starting from the last saved configuration;
- *Clean run*: launch a new instance of PaIRS/CalVi in clean mode;
- *Debug run*: launch a new instance of PaIRS/CalVi in debug mode.

File

The following functions are provided:

- *New* (`Ctrl+N`)[1]: save the current project and start a new one saved to a specified location;
- *Open* (`Ctrl+B`): load a previously saved project (extension of the file is '.calvi_cfg');
- *Save* (`Ctrl+S`): save the current project;
- *Save as* (`Ctrl+Shift+S`): save the current project to a specified location (extension of the file is '.calvi_cfg');
- *Close* (`Ctrl+Esc`)*:* close the current project and turn the interface to its initial state;
- *Exit* (`Alt+F4`): exit the application.

Help (?)

The following functions are provided:

- *Changes*: relevant updates with respect to the previous versions of PaIRS-UniNa are shown;
- *Guide* (`Ctrl+G`): if you are reading this document, you well know this function;
- *About CalVi* (`Ctrl+H`): display information about the release of CalVi in use and the authors and their work.

[1] Please, notice the Macintosh equivalent of the Windows/Linux `Ctrl` key is the `⌘(Command)` key

7. Button bar

The Button bar (Figure 3) is used to hide/show or dock/undock the Input/Output, Process and Vis tabs. By clicking on the corresponding button, it is possible to show one of these tabs and/or focus on it. The tabs currently hidden are denoted by a padlock symbol (🔒).

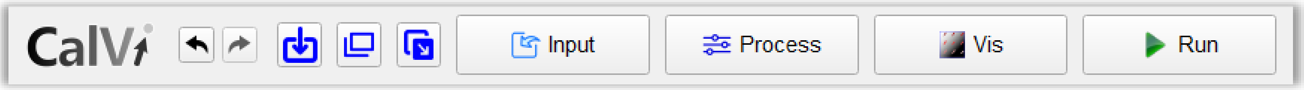


Figure 3 – Button bar.

The following buttons are present in the bar:

↶ Undo (`Ctrl+Z`): undo last changes and access last configuration of settings before the present one.

↷ Redo (`Ctrl+Y`): redo last changes and access last configuration of settings after the present one.

Import saved project (Ctrl+O): import a calibration result file previously saved in the disk.

Change sizes (Ctrl+Space): set the default geometry configuration (a clean run of CalVi starts with it).

Undock/Dock button (`Ctrl+U`): by clicking on it, it is possible to unpack the main window in separated windows each containing a different tab.

Input — Show/focus on Input (`Ctrl+I`). Input can be closed clicking on ⊗ (`Alt+I`).

Process — Show/focus on Process (`Ctrl+P`). Process can be closed clicking on ⊗ (`Alt+P`).

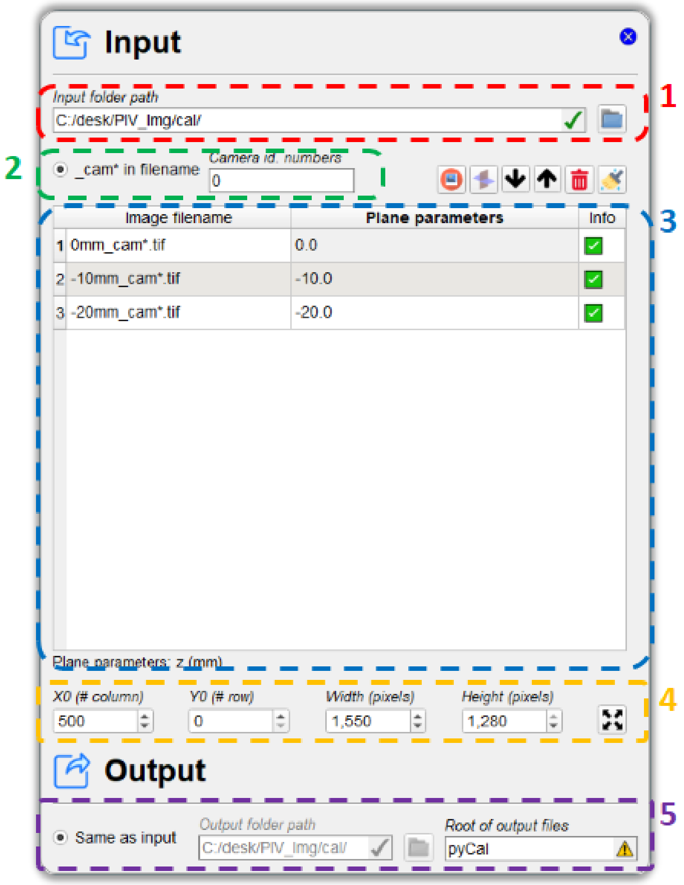
Vis — Show/focus on Vis (`Ctrl+K`). Vis can be closed clicking on ⊗ (`Alt+K`).

▶ Run
❚❚ Stop — Start up or stop (`Ctrl+Return`) the calibration procedure.

Abort — Abort (`Ctrl+Del`) the calibration procedure without saving results.

8. Input/Output

The Input tab allows the user to specify the set of images to process. Images are opened via Pillow, a Python package that supports over than 30 different file formats. For more details, visit the webpage https://pillow.readthedocs.io/en/stable/handbook/image-file-formats.html.



Input folder path: C:/desk/PIV_Img/cal/

● _cam* in filename Camera id. numbers: 0

	Image filename	Plane parameters	Info
1	0mm_cam*.tif	0.0	✓
2	-10mm_cam*.tif	-10.0	✓
3	-20mm_cam*.tif	-20.0	✓

Plane parameters: z (mm)

X0 (# column): 500 Y0 (# row): 0 Width (pixels): 1,550 Height (pixels): 1,280

● Same as input Output folder path: C:/desk/PIV_Img/cal/ Root of output files: pyCal

Figure 4 – The Input/Output tab.

In the Input/Output tab (Figure 4) the following parameters can be specified:

1. the path of the input folder containing the target image files: the image files related to the same calibration process must be contained in the same folder. The path of the input folder can be located either by inserting its name in the corresponding line text box (*Input folder path*) or clicking the button 📁 and using the dialog window generated;
2. the identification numbers of the camera in the image filenames: in the current release sets of target images related to multiple camera bundles can be specified only by including the pattern `_cam*` in the filename and activating the radio button *_cam* in the filename*. In particular, the camera images corresponding to the same target position and orientation must have the same filename containing the pattern `_cam*` with the asterisk being replaced by the camera identification number. The camera identification number must be indicated in the dedicated line edit box. An example of valid set of target images for the calibration of a 4-cameras bundle is the following:

```
0mm_cam0.tif
-10mm_cam0.tif
-20mm_cam0.tif
0mm_cam1.tif
-10mm_cam1.tif
-20mm_cam1.tif
0mm_cam2.tif
-10mm_cam2.tif
-20mm_cam2.tif
…
```

(the files are not ordered in an alphabetical way)

3. the order and the geometric parameters of the target plane configurations: for each target position and orientation the plane parameters must be specified in the second column of the image file table. The number and type of parameters depend on the type of calibration to be performed; depending on the case, the must specify only the z coordinate of the target plane position (standard calibration, in which the planar target is shifted along the normal direction) or the Euler angles and the origin translation related to the target plane position (calibration with unknown plane positions or calibration to estimate the equation of the plane, in which the planar target can be placed in an arbitrary way). The Euler angles and the origin translation related to the target plane position are simply an initial guess for the optimization of the mapping function parameters. The Euler angles β, α, γ are the rotations respectively around the x, y and z-axes; the rotation matrix used is:

$$\begin{bmatrix} \cos\ \alpha \cos\gamma & \cos\ \alpha \sin\gamma & -\sin\alpha \\ \sin\beta \sin\alpha \cos\gamma - \cos\beta \sin\gamma & \sin\beta \sin\alpha \sin\gamma + \cos\beta \cos\gamma & \cos\ \alpha \sin\beta \\ \cos\beta \sin\alpha \cos\gamma + \sin\beta \sin\gamma & \cos\beta \sin\alpha \sin\gamma - \sin\beta \cos\gamma & \cos\ \alpha \cos\beta \end{bmatrix}$$

The image file table is provided with the following buttons:

Add image files to the current calibration process (`Ctrl+D`).

Import plane parameters from file and assign to the current target file (`Ctrl+FD`).

↓ ↑ Move items in the list down/up thus changing the list sorting (`Ctrl+Down`, `Ctrl+Up`).

Delete the item permanently (`Delete` or `Backspace`).

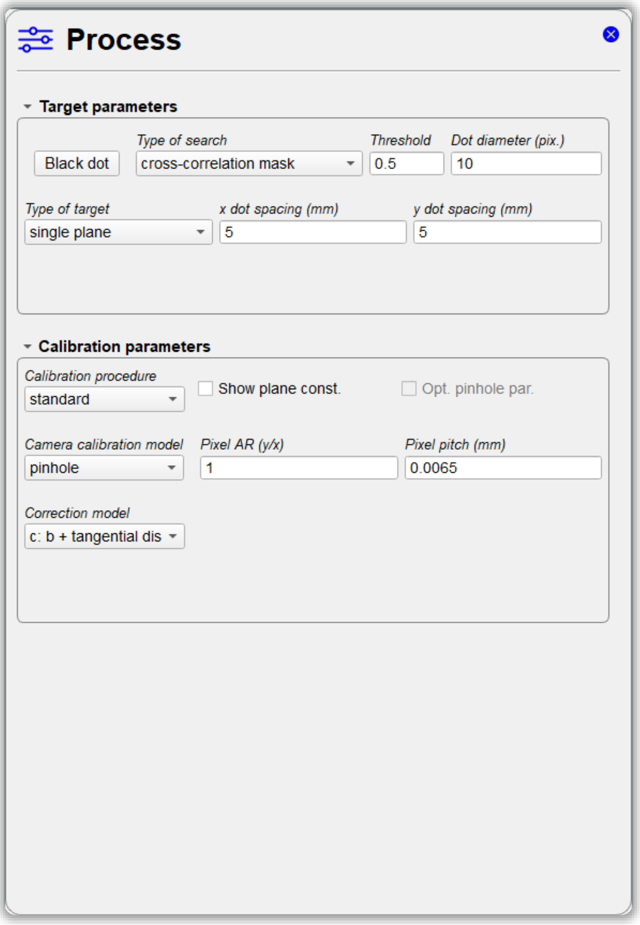
Clean the whole queue permanently (`Ctrl+Shift+T`).

4. the image area to be processed: the four spin boxes *X0 (# column)*, *Y0 (# row)*, *Width (pixels)* and *Height (pixels)* allow to crop the images. *X0* and *Y0* are the first column and the first row of the region of interest, while *Width* and *Height* are the number of the columns and rows of this region, respectively. The result of the image cropping can be easily visualized in the Vis tab;

5. the output file specifications: in the Output tab the user can specify the location of the output folder (this can be chosen to be the same as the input folder or different) and the root of the output filenames.

9. Process

In the Process tab (Figure 5) the user can setup the settings for the calibration process.



Process

Target parameters

Black dot

Type of search: cross-correlation mask

Threshold: 0.5

Dot diameter (pix.): 10

Type of target: single plane

x dot spacing (mm): 5

y dot spacing (mm): 5

Calibration parameters

Calibration procedure: standard

□ Show plane const.

□ Opt. pinhole par.

Camera calibration model: pinhole

Pixel AR (y/x): 1

Pixel pitch (mm): 0.0065

Correction model: c: b + tangential dis

Figure 5 – The Process tab.

The Process tabs consists of two collapsible boxes in which the target and the calibration process parameters can be specified.

Target parameters

It is worth noticing that the calibration performed in CalVi has been designed to work with calibration targets consisting of a plate with a regular Cartesian grid of circular dots. In this box, the parameters of the control dots and their grid can be specified and, more specifically, the user can choose (refer to Figure 6):

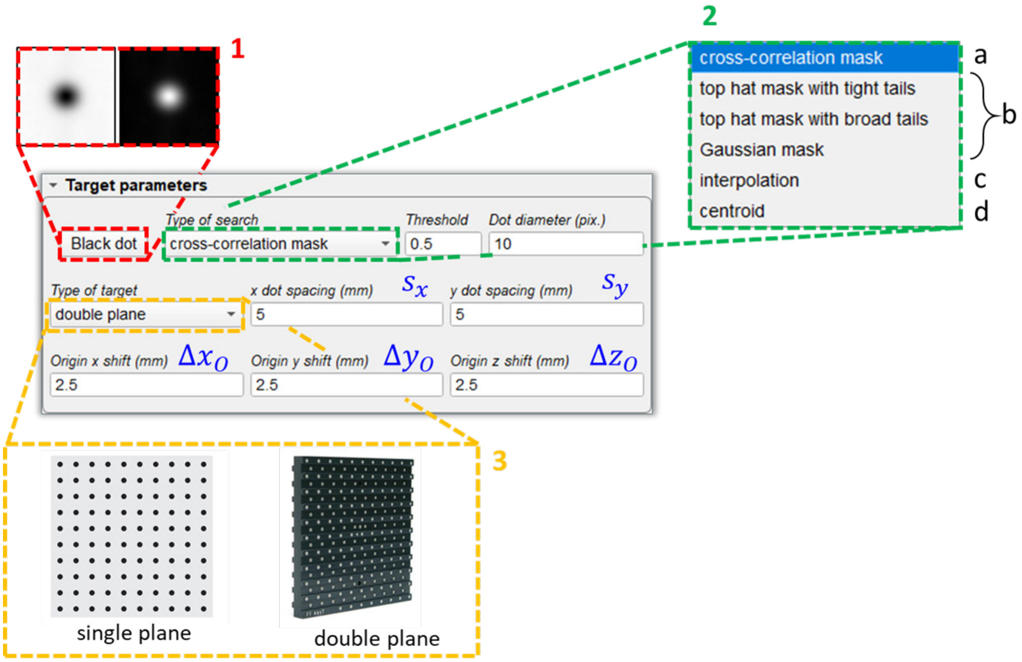
1. if the control points are black dots on a light background or white dots on a dark background. ;
2. the type of dot search: CalVi uses both a template matching algorithm and interpolation or centroid based methods for the detection of the markers in the images of the calibration target. The template (mask) can be generated in different ways:
    a. autocorrelation mask: the template is generated via autocorrelation of a portion of the image that can be interactively selected by the user (using the mouse central button). Before starting the calibration process, the autocorrelation mask is determined based on the whole target image. Such a method yields a subpixel accuracy in most situations and is the method suggested by the authors;

b. function-based masks (top-hat with tight Gaussian-like tails, top-hat with broad Gaussian-like tails and Gaussian): these masks have been observed to offer better results in the case of large dot images with a moderately variable distribution of the intensity levels;

The identification of the control points via the template-matching method is based on the computation of the local correlation map. Only the control points characterized by a peak in the correlation map greater than the value specified in the *Threshold* edit box are stored. The mask is square and its size is equal to 2.5 times the value indicated in *Dot diameter* edit box. This is also the size of the computed local correlation map. The user can easily visualize the mask in the Vis tab with the appropriate image tools.
The dot search can be carried out also in two further ways:

c. interpolation: sub-pixel parabolic interpolation around the local maximum (white dot) or minimum (black dot). The local peak is identified considering only the intensity levels beyond the value specified in the *Threshold* edit box and in a square area of edge equal to 2.5 times the value indicated in *Dot diameter* edit box;

d. centroid: determination of the centroid of the distribution of the absolute image levels above the threshold specified in the *Threshold* edit box in the square area of edge equal to 2.5 times the value indicated in *Dot diameter* edit box.



Target parameters

Black dot

Type of search: cross-correlation mask

Threshold: 0.5

Dot diameter (pix.): 10

Type of target: double plane

x dot spacing (mm): 5

y dot spacing (mm): 5

Origin x shift (mm): 2.5

Origin y shift (mm): 2.5

Origin z shift (mm): 2.5

Figure 6 – Target parameters box.

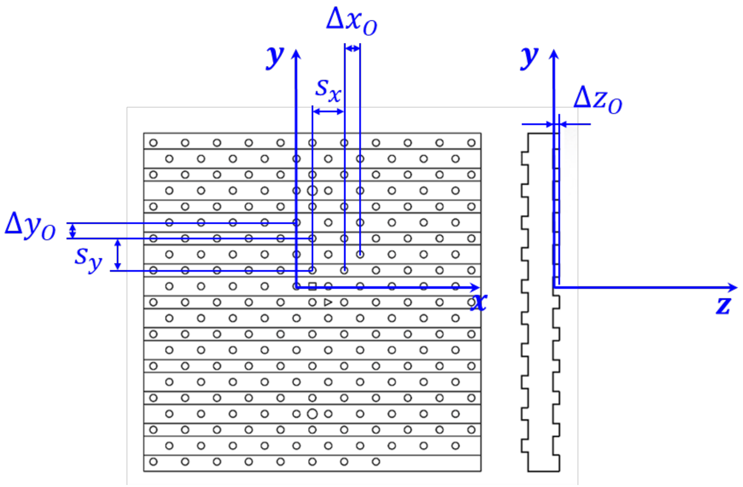
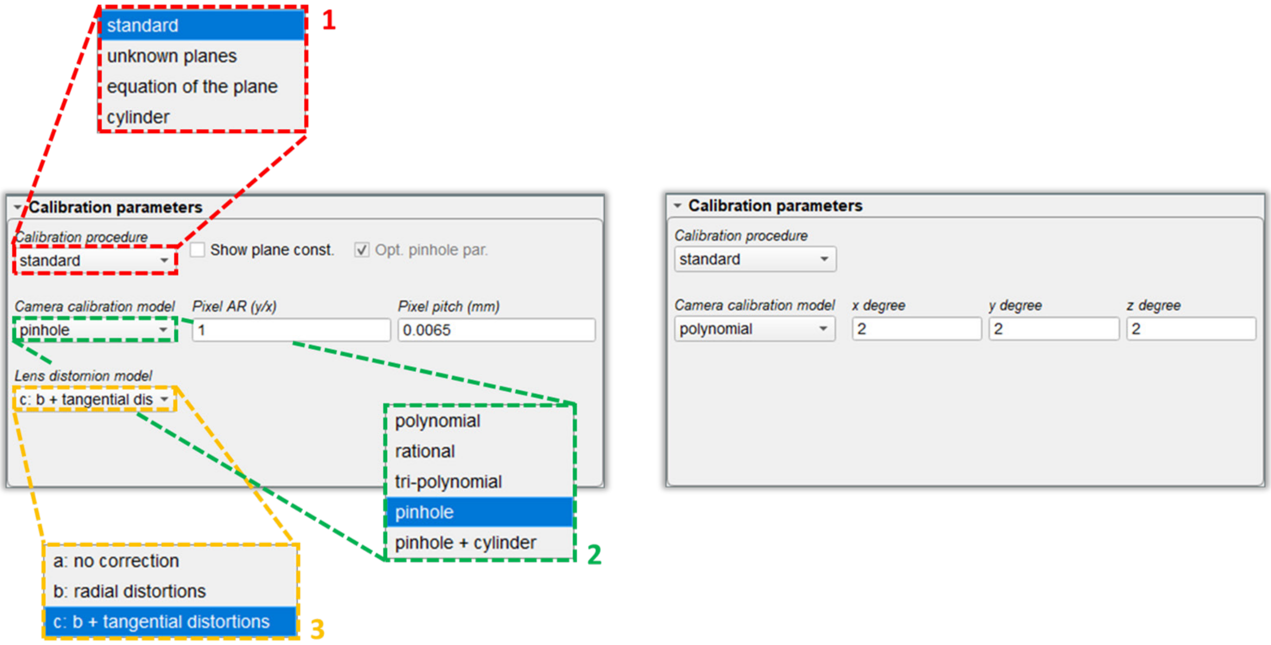


Figure 7 – Definition of dot spacings and origin shifts for a double-plane target.

3. the type of target: single plane or double plane. The dot spacings s_x and s_y along the x and y-directions and the origin shifts Δx_O, Δy_O and Δz_O in the x, y and z directions must be specified in the corresponding edit boxes. Their definition is intuitive and clarified in Figure 7 for a double-plane target.

Calibration parameters



Calibration parameters

Calibration procedure: standard
□ Show plane const.
☑ Opt. pinhole par.
Camera calibration model: pinhole
Pixel AR (y/x): 1
Pixel pitch (mm): 0.0065
Lens distortion model: c: b + tangential dis

Calibration parameters

Calibration procedure: standard
Camera calibration model: polynomial
x degree: 2
y degree: 2
z degree: 2

Figure 8 – Calibration parameters box.

In this box the user can choose (refer to Figure 8):

1. the type of calibration procedure: see Table 1 for more information in this regard;
2. the camera calibration model: CalVi supports accurate calibration with the camera models mostly used in the PIV community: polynomials, rational functions and the pinhole camera model. The reader is referred to reference [1] for more information about the definition of the polynomial and rational camera models (also denoted as analytical camera models). CalVi also supports the integration of the pinhole camera model with a refractive correction model for cylindrical geometries. The latter relies on a ray tracing procedure and correctly represents the refraction

of the lines-of-sight (LOSs) across the cylindrical interfaces via Snell's law, as explained in [2]. The camera models are formulated in Table 2. In the case of the analytical camera models, the user is asked to specify the degree of the polynomial functions N_x, N_y; N_z, whereas in the case of the pinhole camera model the pixel size d and the pixel aspect ratio χ must be specified;

3. the lens distortion corrections for the pinhole camera model:

   a: no correction: the pinhole reduces to a linear model (DLT, direct linear transformation);

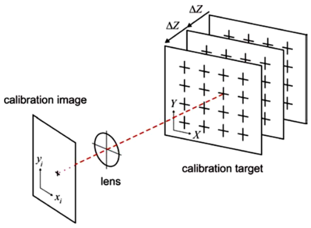
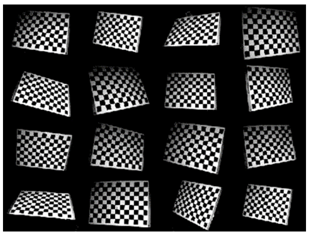
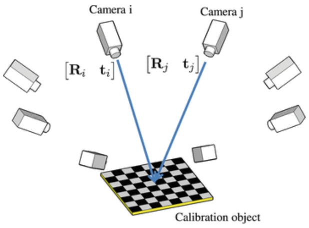
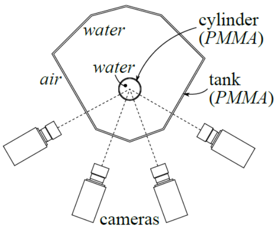
   b: radial distortions: the correction terms for the radial distortions are included;

   c: b + tangential distortions: all the correction terms for both the radial and the tangential distortions are included.

   When the pinhole + cylinder camera model is selected, further correction models are available:

   d: c + cylinder origin: also the position of the cylinder is optimized in the calibration procedure;

Table 1 - Type of calibration procedure.

Type	Description
standard  	The calibration target is translated along the z direction (normal to the target plane) in several known positions, accurately measured by the user. The z coordinate of each plane position must be specified in the *Plane parameters* column of the image file table included in the Input tab. CalVi can also determine a-posteriori all the six plane parameters (Euler angles and origin shifts) for each target position, both in case of a single camera and in case of a multiple camera bundle. To visualize such parameters the user must activate the *Show plane const.* check box.
unknown planes (calibration per planes) 	This type of calibration, also defined as "calibration per planes", does not need a perfect alignment of the target during the image acquisition. The user should perform the following steps during the image acquisition: 1. place the target in a reference position and acquire a first set of images with all the cameras. The reference position of the target, together with the choice of the origin dot in the image (during the calibration process performed in Vis), fixes the world reference frame; 2. rotate the target around the x or y-axis of about 4-8 degrees and acquire a second set of images with all the cameras. It should not be necessary to be very precise, a translation shift of the target should not be important at this stage; 3. rotate in the opposite direction the target and acquire a third set of images with all the cameras; 4. assign another rotation or simply translate the target along the z-direction and acquire a fourth set of images with all the cameras; 5. you may acquire also other images of the target in different positions. When selecting this type of calibration the generated calibration files contain also the constants of the planes.
equation of the plane  Camera i Camera j $[\mathbf{R}_i \ \ \mathbf{t}_i]$ $[\mathbf{R}_j \ \ \mathbf{t}_j]$ Calibration object	In this case the goal of the calibration is to determine the plane constants of the target position starting from a previous camera calibration and from a set of images of the target in the considered unknown position. It is advised to perform this kind of calibration with at least 3 cameras.
cylinder  water cylinder (PMMA) air water tank (PMMA) cameras	This type of calibration procedure is used to optimize the cylinder parameters of the "pinhole + cylinder" camera model by using multiple cameras. It requires a pinhole camera calibration for each camera of the bundle as a starting point. The user can choose whether to optimize the pinhole camera parameters for each camera and/or the plane constants of the target positions recorded in the images. The plane constants and the cylinder parameters are common to all the cameras. Please see section 12 for more information about the procedure to correctly calibrate the "pinhole + cylinder" camera model.

e: d + cylinder rotation: also the angular orientation of the cylinder is optimized in the calibration procedure;

f: e + cylinder radius and thickness: also the internal and external radius of the cylinder are optimized in the calibration procedure;

g: f + refractive index ratio: also the ratio of the refractive indexes of the cylinder wall and the surrounding fluid is optimized in the calibration procedure.

Table 2 – Type of camera model; see Paolillo and Astarita (IEEE 2020) [1] and (MST 2021) [2] for more details.

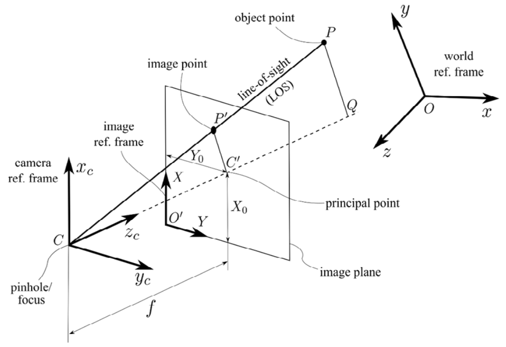
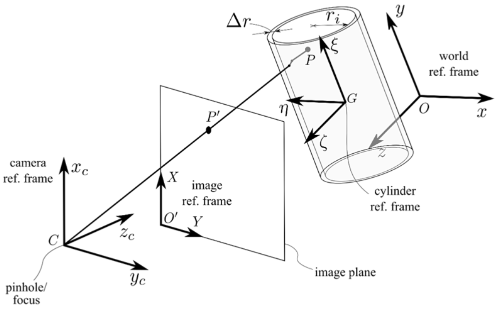
Type	Formulation
polynomial	$X = \sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} a_{ijk} x^i y^j z^k\,, \qquad Y = \sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} b_{ijk} x^i y^j z^k$ $N_y^* = \min(N_y, N_{max} - k);\ N_x^* = \min(N_x, N_{max} - k - j);\ N_{max} = \max(N_x\,, N_y, N_z).$
rational	$X = \frac{\sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} a_{ijk} x^i y^j z^k}{\sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} c_{ijk} x^i y^j z^k}, \qquad Y = \frac{\sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} b_{ijk} x^i y^j z^k}{\sum_{k=0}^{N_z} \sum_{j=0}^{N_y^*} \sum_{i=0}^{N_x^*} c_{ijk} x^i y^j z^k}$ with $c_{000} = 1$ $N_y^* = \min(N_y, N_{max} - k);\ N_x^* = \min(N_x, N_{max} - k - j);\ N_{max} = \max(N_x\,, N_y, N_z).$
tri-polynomial	$X = \sum_{k=0}^{N_z} \sum_{j=0}^{N_y} \sum_{i=0}^{N_x} a_{ijk} x^i y^j z^k\,, \qquad Y = \sum_{k=0}^{N_z} \sum_{j=0}^{N_y} \sum_{i=0}^{N_x} b_{ijk} x^i y^j z^k$
pinhole	$c\underline{u}_i = FT_c\underline{x}$ $\underline{X} = U\left(B\underline{u}_i + \delta\underline{u}(\underline{u}_i)\right)$ $\underline{u}_i = [u_i, v_i, 1]^T$ ideal coordinates in the camera reference frame $\underline{x} = [x, y, z, 1]^T$ coordinates in the world reference frame  object point P y world ref. frame image point line-of-sight (LOS) Q O x z P' image ref. frame Y_0 C' camera ref. frame x_c X principal point X_0 O' Y C z_c image plane y_c pinhole/ focus f $T_c = \begin{bmatrix} R_c & \underline{x}_{c0} \\ \underline{0}^T & 1 \end{bmatrix};\ F = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 1 \end{bmatrix};\ B = \begin{bmatrix} 1 + b_1 & b_2 & 0 \\ b_2 & 1 - b_1 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ U = \begin{bmatrix} \frac{\chi}{d} & 0 & X_0 \\ 0 & \frac{1}{d} & Y_0 \\ 0 & 0 & 1 \end{bmatrix}$ R_c = Euler rotation matrix, $\underline{x}_{c0}$ = coordinates of the origin in the camera reference frame, f = effective pinhole focal length, b_1, b_2 = linear distortion coefficient, d pixel size in the Y direction, χ = pixel aspect ratio, X_0, Y_0 = image coordinates of the principal point C' $\delta u(u, v) = u(k_1 r^2 + k_2 r^4) + p_1(r^2 + 2u^2) + 2p_2 uv$ $\delta v(u, v) = v(k_1 r^2 + k_2 r^4) + 2p_1 uv + p_2(r^2 + 2u^2)$ $r^2 = u^2 + v^2$, k_1, k_2 = coefficients for the radial distortion, p_1, p_2 = coefficients for the tangential distortion.

Table 2 – Type of camera model; see Paolillo and Astarita (IEEE 2020) [1] and (MST 2021) [2] for more details.

pinhole + cylinder	$\phi\left(\underline{\xi}_A;\ \underline{\xi}_C, r_i, \Delta r, \rho\right) = T_b\underline{x}$ $c\underline{u}_i = F T_c T_b' \underline{\xi}_A$ $\underline{X} = U\left(B\underline{u}_i + \delta\underline{u}(\underline{u}_i)\right)$ $\phi\left(\underline{\xi}_A;\ \underline{\xi}_C, \underline{g}, \rho\right)$ refractive distortion function $\underline{\xi}_A = [\xi_A, \eta_A, \zeta_A, 1]^T$ coordinates of the dewarped point in the cylinder (body) reference frame  Δr r_i y ξ P world ref. frame G η O x P' ζ z camera ref. frame x_c X image ref. frame cylinder ref. frame O' Y C z_c image plane pinhole/ focus y_c $\underline{u}_i = [u_i, v_i, 1]^T$ ideal coordinates in the camera reference frame $\underline{x} = [x, y, z, 1]^T$ coordinates in the world reference frame $\underline{\xi}_C = [\xi_C, \eta_C, \zeta_C, 1]^T$ coordinates of the focus C in the cylinder reference frame $\underline{\xi} = T_b\underline{x} = \begin{bmatrix} R_b & \underline{x}_{b0} \\ \underline{0}^T & 1 \end{bmatrix}\underline{x};\quad \underline{x} = T_b'\underline{\xi} = \begin{bmatrix} R_b^T & -R_b^T\underline{x}_{b0} \\ \underline{0}^T & 1 \end{bmatrix}\underline{\xi}$ R_b = Euler rotation matrix from world to cylinder frame; $\underline{x}_{b0}$ = coordinates of the origin in the cylinder (body) reference frame; r_i = cylinder internal radius; Δr = cylinder wall thickness; $\rho = n_f/n_c$ = ratio of the refractive index of the fluid to that of the cylinder material (the fluid inside and outside is supposed to be the same). $T_c = \begin{bmatrix} R_c & \underline{x}_{c0} \\ \underline{0}^T & 1 \end{bmatrix};\ F = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 1 \end{bmatrix};\ B = \begin{bmatrix} 1+b_1 & b_2 & 0 \\ b_2 & 1-b_1 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ U = \begin{bmatrix} \frac{\chi}{d} & 0 & X_0 \\ 0 & \frac{1}{d} & Y_0 \\ 0 & 0 & 1 \end{bmatrix}$ R_c = Euler rotation matrix, $\underline{x}_{c0}$ = coordinates of the origin in the camera reference frame, f = effective pinhole focal length, b_1, b_2 = linear distortion coefficient, d pixel size in the Y direction, χ = pixel aspect ratio, X_0, Y_0 = image coordinates of the principal point C' $\delta u(u,v) = u(k_1r^2 + k_2r^4) + p_1(r^2 + 2u^2) + 2p_2uv$ $\delta v(u,v) = v(k_1r^2 + k_2r^4) + 2p_1uv + p_2(r^2 + 2u^2)$ $r^2 = u^2 + v^2$, k_1, k_2 = coefficients for the radial distortion, p_1, p_2 = coefficients for the tangential distortion.

10. Vis

The Vis tab (Figure 9) is the core of the calibration process and allows the user to perform the interactive calibration once the *Run* button in the button bar has been clicked.

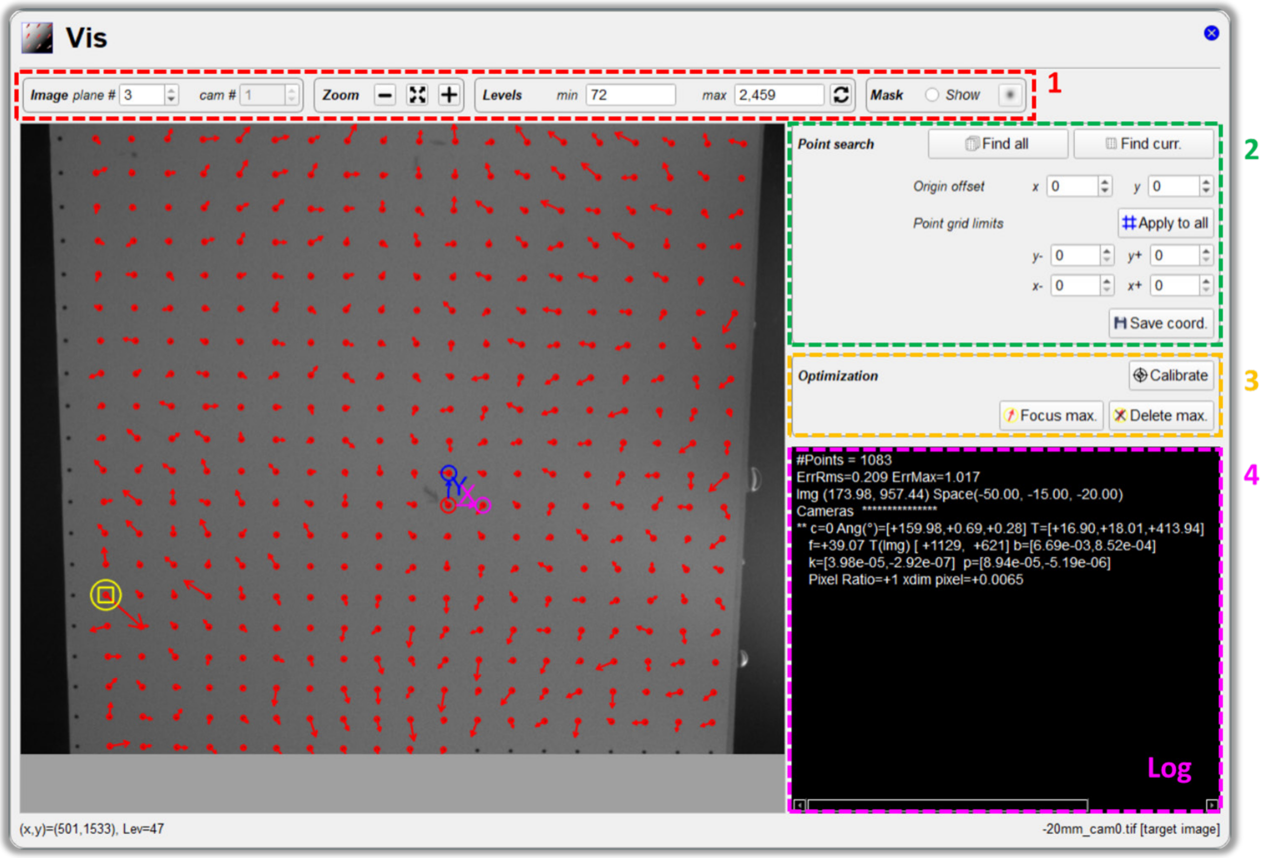


Image plane # 3 cam # 1

min 72 max 2,459

x 0 y 0

y- 0 y+ 0

x- 0 x+ 0

```
#Points = 1083
ErrRms=0.209 ErrMax=1.017
Img (173.98, 957.44) Space(-50.00, -15.00, -20.00)
Cameras  ***************
** c=0 Ang(°)=[+159.98,+0.69,+0.28] T=[+16.90,+18.01,+413.94]
   f=+39.07 T(Img) [ +1129,  +621] b=[6.69e-03,8.52e-04]
   k=[3.98e-05,-2.92e-07]  p=[8.94e-05,-5.19e-06]
   Pixel Ratio=+1 xdim pixel=+0.0065
```

(x,y)=(501,1533), Lev=47

-20mm_cam0.tif [target image]

Figure 9 – The Vis tab.

It consists of four blocks:

1. the image plot tool, where the user can select the plane number and the camera number for the target image to be displayed, set the zoom and the level limits for the visualization and show/hide the mask;
2. the control point search box, which allows to find point coordinates in all the target images or only in the currently visualized image, assign an origin offset, limit the grid of dots and save the image coordinates to the disk;
3. the optimization box, where the user can start the optimization of the mapping function parameters and detect and remove the dots with the largest reprojection errors to refine the calibration parameters;
4. the log window, where instructions for the use of the interactive calibration tool, the output from the calibration process and potential errors are provided.

The following section explains how to use the interactive calibration tool.

11. Use of the interactive calibration tool

Once clicked the *Run* button in the button bar the user is driven through the interactive calibration process via the instructions displayed in the log window in the Vis tab. The calibration process consists of the following steps.

Spot detection

This phase starts upon clicking the *Run* button in the button bar or the *Find all* button in the control point search box in the Vis tab. In this phase, the user must choose the spot that defines the origin of the world reference frame (central point), by simply using the left button of the mouse to select a spot. If a red circle is drawn around the spot, the image coordinates of the origin have been correctly found. Subsequently, with the same procedure, the user must select the point along the x (purple square, defined as: *right point*) and y (blue circle defined as: *upper point*) axis (see Figure 10).

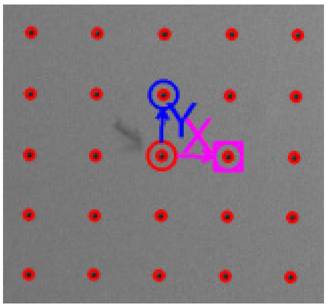
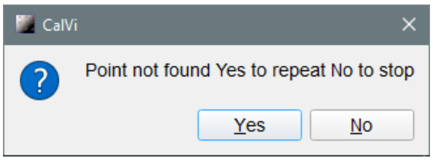


Figure 10 – Detection of the central point and the x and y-axis.

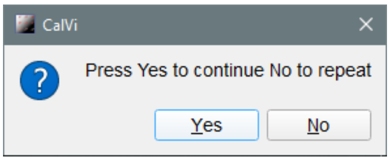
CalVi immediately identifies the spots on the image and writes the number of points in the log window. If CalVi fails in finding one spot, a pop-up dialog appears where the user is asked whether to retry or to stop and exit (Figure 11). In this case the user may point the spot better. Normally the pop-up is shown in the left bottom part of the main window and, to speed up the procedure, in Windows and Linux the mouse pointer is moved automatically onto the most probable answer and back to the initial position once the button is clicked. This behavior initially is a little unsettling but in the long run reduces the time spent in clicking.



Point not found Yes to repeat No to stop

Figure 11 – Pop-up dialog upon CalVi failure in detecting one spot.

After selecting the three control points on the first image defining the origin and the x and y-axis of the world reference frame, a pop-up dialog appears on the bottom right part of the screen (Figure 12). CalVi shows all the found spots with red circles and if the spots have been correctly identified the user should just press the *Yes* button (or the return button on the keyboard) to go on with the second image target. If the spots are not found correctly, the user can press the *No* button in order to restart the procedure.



Press Yes to continue No to repeat

Figure 12 - Pop-up dialog after the detection of the first three control points.

After having correctly found the spots on the first image target, CalVi moves to the second image and asks to point the spot that defines the origin on the second plane, necessary to specify the z-axis of the world reference frame (in the case of a double plane target this step is performed automatically, in some cases this is not possible and the user need to repeat the procedure for all the planes). Again, if the origin has been correctly identified, the user just needs to press the *Yes* button in the pop-up window to go on. CalVi shows all the found spots with red circles and again the user can simply confirm and go on.

From the third image target on CalVi automatically identifies the origin and the user needs only to press the *Yes* button in the pop-up window. At the end of the whole spot detection stage, CalVi shows the total number of points found.

The user can start again the spot detection procedure if they desire to change the position of the origin by using the *Find all* button or repeat the detection for a single target image by using the *Find curr*. button in the control point search box.

Calibration

Normally at this point, the user performs the calibration by simply pressing the *Calibration* button in the optimization box. If the user is only interested in the point position, they may just press the *Save coord.* button in the control point search box.

If the calibration takes too much time, the user can repress the same button (now called *Stop*) to terminate the process. At the end, arrows are shown to identify the reprojection error associated to each control point. It is possible to visualize and/or remove the point with largest reprojection error by pressing the corresponding buttons, i.e., the *Focus max* and *Delete max* buttons in the optimization box, or by using the context menu. The control point with maximum error is identified with a yellow square plus a yellow circle. If needed, the user can perform further iterations of the optimization by pressing again the *Calibration* button.

Point removal

The control points can be removed in bulk by using the spin boxes within the *Point grid limits* section in the control point search box. By pressing the *Apply to all* button, CalVi uses the same limits for all the target images.

Selection of the area for the auto-correlation mask

When using the cross-correlation mask type, CalVi initially chooses a relatively large central area of the image to build the correlation mask. The mask can be visualized via the mask box in the image plot tool. In some cases, the mask area can be biased and could result in wrong positioning of the spots or even in further difficulties in finding the spots. In these cases, the user can choose a new mask area by clicking on the image with the central mouse button and dragging till the final position. When a user-defined area for building the auto-correlation mask is used, a blue rectangle indicating such an area will be drawn on the target image. The mask area is normally different for each plane, but if it is chosen during the origin search of the first, the same mask area is applied to all the target images of the same camera. The user can choose a different mask area whenever needed.

Origin not found or not in the image

If the origin is not found or is not in the image the automatic procedure fails and a pop-up dialog warns the user that the origin has not been found. In these cases, the user, after completing the initial procedure, should first select the plane using the *plane #* spin box in the image plot tool, then press the button *Find curr.* button in the control point search box and finally define a new reference frame in the image. If needed the selected virtual origin can be moved, spot by spot, to the correct position by using the spin boxes within the *Origin shift* section in the control point search box; a double green circle identifies the shifted origin while assigning these parameters. The origin can also be placed outside of the image. Moreover, the origin can be changed *a posteriori* in all the target images.

12. Calibration of pinhole + cylinder camera model

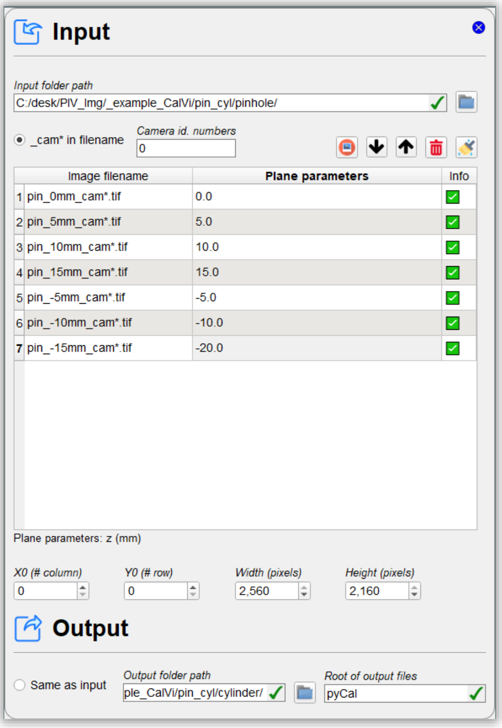
In the present section, three different strategies for the calibration of the pinhole + cylinder camera model are presented. This type of calibration is less robust than the classical ones and the nonlinear optimization of the calibration parameters may not converge easily, so the following are guidelines but, if needed, a more specific approach can be advised by the UniNa team. The recommended procedure (procedure 1) consists in two separate steps: a calibration of the pinhole camera model for each camera of the bundle based on a set of images of the target placed in the field of view in absence of the transparent cylindrical body; an estimation of the cylinder parameters based on a set of images of the target placed in the field of view in presence of the cylinder. Procedure 2 should be followed if it is not possible to record a set of images of the target in absence of the cylinder; in this case both the calibration of the pinhole camera model and the estimation of the cylinder parameters can be performed using the set of images of the target visualized through the cylinder. Procedure 3 is only recommended in the case of a single camera setup and it consists in simultaneous calibration of both the pinhole and the cylinder parameters based on a set of images of the target in presence of the cylinder.

Procedure 1 (recommended): separate pinhole camera model calibration and cylinder parameter estimation with different sets of target images

Step 1: pinhole camera model calibration

This step consists of a calibration of the pinhole camera model for each camera of the bundle. This can be performed either in a standard way (target moved in known positions) or with images of the target in unknown positions (calibration per planes). The images of the target must be recorded i**n absence of the cylinder** in the field of view and with the cameras arranged in their final configuration.

In case of standard calibration, the user must record a sufficient number of images of the target translated along the z direction (normal to the target plane). It is suggested to put the image files for each camera in the same folder, say the *pinhole* folder, and to select a different output folder for the calibration results, say the *cylinder* folder. The calibration of the pinhole camera model must be performed separately for each camera; for fastness and simplicity, it is recommended to save the image files for each camera with names containing the pattern `_cam*` in such a way that it is possible to switch to the calibration of another camera by simply changing the *Camera id. Number* in the Input tab. In the Calibration parameters box of the Process tab, the standard calibration procedure and the pinhole camera model (with an appropriate lens distortion model) must be selected. Alternatively, the pinhole + cylinder camera model can be selected, but the *n ratio* value must be set to 1. The following figures show an example of how to set the parameters in the Input tab and the Calibration parameters box of the process tab for the present case. In the below example, the user will run the calibration for the camera 0 and then by simply changing the Camer id. Number they will move to camera 1, 2, and so on.



Input

Input folder path: C:/desk/PIV_Img/_example_CalVi/pin_cyl/pinhole/

● _cam* in filename — Camera id. numbers: 0

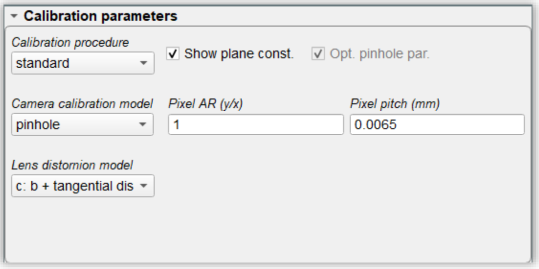
	Image filename	Plane parameters	Info
1	pin_0mm_cam*.tif	0.0	✓
2	pin_5mm_cam*.tif	5.0	✓
3	pin_10mm_cam*.tif	10.0	✓
4	pin_15mm_cam*.tif	15.0	✓
5	pin_-5mm_cam*.tif	-5.0	✓
6	pin_-10mm_cam*.tif	-10.0	✓
7	pin_-15mm_cam*.tif	-20.0	✓

Plane parameters: z (mm)

X0 (# column): 0 — Y0 (# row): 0 — Width (pixels): 2,560 — Height (pixels): 2,160

Output

○ Same as input — Output folder path: ple_CalVi/pin_cyl/cylinder/ — Root of output files: pyCal



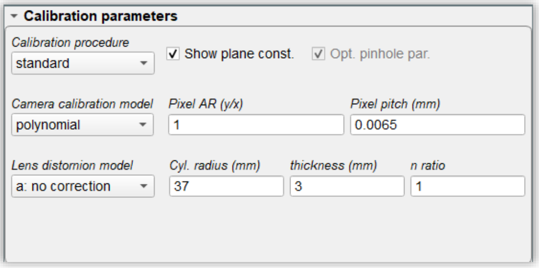
Calibration parameters

Calibration procedure: standard — ☑ Show plane const. — ☑ Opt. pinhole par.

Camera calibration model: pinhole — Pixel AR (y/x): 1 — Pixel pitch (mm): 0.0065

Lens distortion model: c: b + tangential dis

Alternatively…



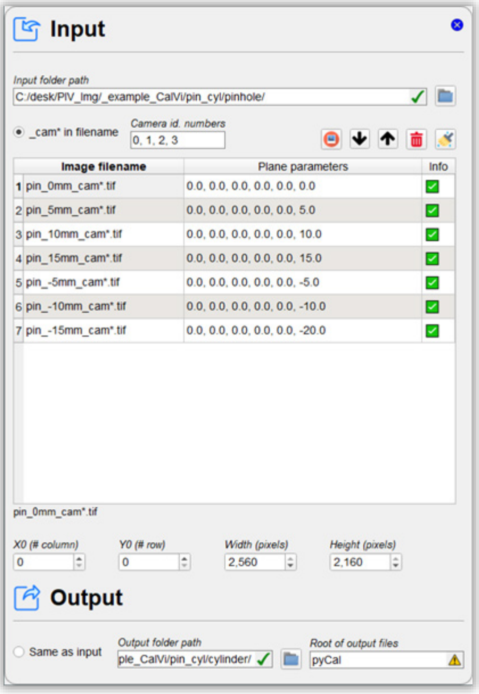
Calibration parameters

Calibration procedure: standard — ☑ Show plane const. — ☑ Opt. pinhole par.

Camera calibration model: polynomial — Pixel AR (y/x): 1 — Pixel pitch (mm): 0.0065

Lens distortion model: a: no correction — Cyl. radius (mm): 37 — thickness (mm): 3 — n ratio: 1

In the case of calibration per planes, all the cameras can be calibrated in a single step. Thus, the differences with respect to the previous case are: in the Camera id. numbers in the Input tab all the camera numbers must be specified (please notice that in this case it is mandatory to save images of the target with the pattern _cam* in the filename); the initial guess values for the plane parameters must be appropriately assigned in the image file table contained in the Input tab; the calibration procedure in the Calibration parameters box of the Process tab must be set to "unknown planes". The following figures show an example of how to set the parameters in the Input tab and the Calibration parameters box of the process tab for the present case:



Input

Input folder path: C:/desk/PIV_Img/_example_CalVi/pin_cyl/pinhole/

● _cam* in filename — Camera id. numbers: 0, 1, 2, 3

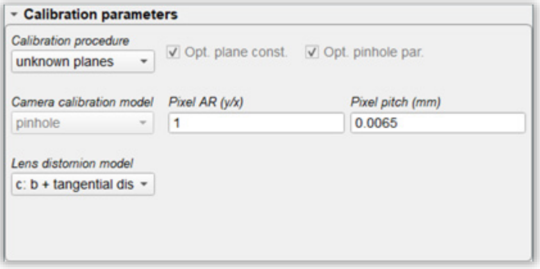
	Image filename	Plane parameters	Info
1	pin_0mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 0.0	✓
2	pin_5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 5.0	✓
3	pin_10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 10.0	✓
4	pin_15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 15.0	✓
5	pin_-5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -5.0	✓
6	pin_-10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -10.0	✓
7	pin_-15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -20.0	✓

pin_0mm_cam*.tif

X0 (# column): 0 — Y0 (# row): 0 — Width (pixels): 2,560 — Height (pixels): 2,160

Output

○ Same as input — Output folder path: ple_CalVi/pin_cyl/cylinder/ — Root of output files: pyCal



Calibration parameters

Calibration procedure: unknown planes — ☑ Opt. plane const. — ☑ Opt. pinhole par.

Camera calibration model: pinhole — Pixel AR (y/x): 1 — Pixel pitch (mm): 0.0065

Lens distortion model: c: b + tangential dis

It is noticed that the calibration per plane is not recommended in the case of target positions consisting of planes all parallel to each other (as usually happens in the standard calibration).

Step 2: estimation of the cylinder parameters

In this step the cylinder parameters are estimated using all the cameras available. The images of the target must be recorded **in presence of the cylinder** in the field of view and with the cameras arranged in their final configuration. In particular, the target can be placed either inside the cylinder or outside of it in the rear part. In fact, the pinhole + cylinder camera model is capable of mapping the whole domain surrounding the cylindrical body.

For this step, the image files for each camera must be saved in the folder containing the output calibration files from the previous step (the *cylinder* folder). It is suggested to save the output files from the present step to a different output folder, in such a way that the calibration output files from the previous step are not overwritten. In this case it is mandatory to save images of the target with the pattern `_cam*` in the filename and the known and initial guess values for the plane parameters must be appropriately assigned in the image file table contained in the Input tab. If the plane parameters are not optimized in the calibration procedure, then their values must be assigned accurately in the latter table. When using a single target image in this step, like for instance the image of the target placed behind the cylinder, if the position of the target is not changed while placing the cylinder in the field of view, the user is advised to perform an intermediate step between step 1 and step 2, consisting in the determination of the equation of the plane for the target, based on the previously calibrated pinhole camera model. The thus determined plane constants should be then imported in the Input tab using the related button .

In the Calibration parameters box of the Process tab, the cylinder calibration procedure must be selected. If the pinhole parameters are optimized, the user must select a lens distortion model for the pinhole part. As concerns the cylinder parameter optimization, the known or initial guess values for the internal cylinder radius, the cylinder wall thickness and the ratio of the refractive indexes of the fluid and the wall material must be specified. Finally, the user must choose a strategy for the optimization of the cylinder parameters among the following options (in case of difficulties in the nonlinear optimization the user may start with option a and only after try to optimize also the other parameters):

a: cylinder origin and rotation: only the position and orientation of the cylinder are optimized, the geometrical and optical properties are not (their values are kept constat and equal to those assigned in the Process tab);

b: a + cylinder thickness: the position and orientation and the thickness of the cylinder are optimized;

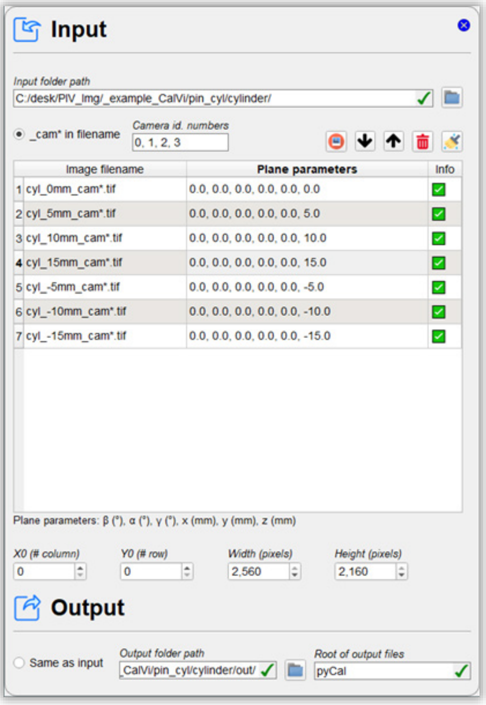
c: b + refractive index (n) ratio: all the cylinder parameters expect for the internal radius are optimized;

d: b + internal radius: all the cylinder parameters expect for the refractive index (n) ratio are optimized;

e: a + internal radius and n ratio: all the cylinder parameters expect for the thickness are optimized;

f: all cylinder parameters.

The following figures show an example of how to set the parameters in the Input tab and the Calibration parameters box of the process tab for the present case:



Input

Input folder path: C:/desk/PIV_Img/_example_CalVi/pin_cyl/cylinder/

● _cam* in filename

Camera id. numbers: 0, 1, 2, 3

	Image filename	Plane parameters	Info
1	cyl_0mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 0.0	☑
2	cyl_5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 5.0	☑
3	cyl_10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 10.0	☑
4	cyl_15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 15.0	☑
5	cyl_-5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -5.0	☑
6	cyl_-10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -10.0	☑
7	cyl_-15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -15.0	☑

Plane parameters: β (°), α (°), γ (°), x (mm), y (mm), z (mm)

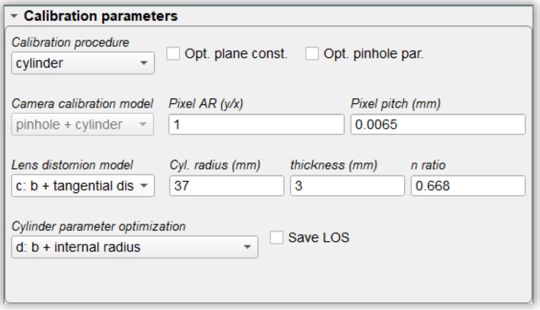
X0 (# column): 0
Y0 (# row): 0
Width (pixels): 2,560
Height (pixels): 2,160

Output

○ Same as input

Output folder path: CalVi/pin_cyl/cylinder/out/

Root of output files: pyCal



Calibration parameters

Calibration procedure: cylinder
□ Opt. plane const.
□ Opt. pinhole par.

Camera calibration model: pinhole + cylinder
Pixel AR (y/x): 1
Pixel pitch (mm): 0.0065

Lens distornion model: c: b + tangential dis
Cyl. radius (mm): 37
thickness (mm): 3
n ratio: 0.668

Cylinder parameter optimization: d: b + internal radius
□ Save LOS

When using images of the target outside of the cylinder and in its rear part, a relatively small number of images (also a single image) may be sufficient by virtue of the large distortions caused by the twofold refraction of the lines-of-sight across the front and the rear part of the cylindrical wall. In case the target is placed inside the cylinder it is recommended to record a greater number of images of the target in different positions and try to put the control points as close as possible to the cylindrical wall.

Procedure 2: separate pinhole camera model calibration and cylinder parameter estimation with the same set of target images

This procedure should be followed when it is not possible to record images of the target in absence of the cylinder, thus only images of the target visualized across the cylinder wall are available. It is again suggested to perform the calibration in two steps as in procedure 1. Some precautions should be adopted in the first step of the current procedure.

Step 1: pinhole camera model calibration

In this step, the pinhole camera model for each camera of the bundle must be calibrated. This task is now performed using the images of the target in presence of the cylinder, neglecting the optical distortion due to the latter. For this reason, it is not advisable to perform a calibration per planes (poorly accurate results would be obtained by neglecting the cylindrical wall). So, the user must perform a standard calibration of the pinhole camera model for each camera. The output folder for the calibration results of this step should coincide with the input folder (activating *Same as input* checkbox in the Input tab), since the images to be used in step 2 are the same as those employed in this step.

In the calibration process, the user should try to limit the control points employed for the optimization to those falling in the region around the cylinder axis, that is to say farther from the cylindrical wall (in all directions). In fact, the optical distortions from the cylindrical wall of the lines-of-sight reaching these points are reduced if the angle between the optical axis of the camera and the cylinder axis is small.

To obtain a more accurate result, the user can calibrate the pinhole + cylinder camera model for each camera of the bundle following the below procedure 3 (applied for single camera) and then pass to step 2.

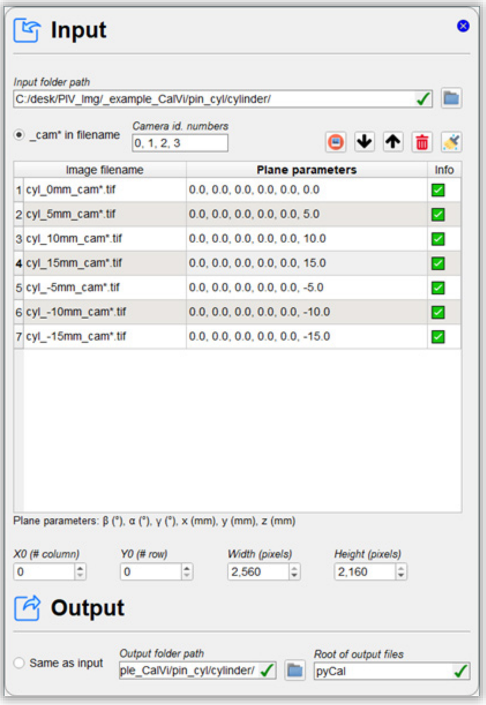
Step 2: estimation of the cylinder parameters

This step is identical to the step 2 of procedure 1, to which the reader is referred (see page 23).

Procedure 3: simultaneous pinhole camera model calibration and cylinder parameter estimation

This procedure is based on consists the simultaneous calibration of both the pinhole and the cylinder parameters based on a set of images of the target in presence of the cylinder. This can be performed with a standard calibration procedure and selecting the pinhole + cylinder camera calibration model. The known or initial guess values for the internal cylinder radius, the cylinder wall thickness and the ratio of the refractive indexes of the fluid and the wall material must be specified. In this case, the user cannot choose a strategy for the optimization of the cylinder parameters among the options available for the cylinder calibration procedure as in step 2 of the above-described procedure 1 and 2. The cylinder parameters to be optimized can be selected from the *Lens distortion model* combo choosing among the options reported in bullet 3 at page 14 of the present guide. However, also in the case of a single camera bundle, this calibration procedure can be used as step 1 of procedure 2 and refined with a following ad-hoc estimation of the cylinder parameters (step 2 of procedure 2).

The following figures show an example of how to set the parameters in the Input tab and the Calibration parameters box of the process tab for the present case:



Input

Input folder path
C:/desk/PIV_Img/_example_CalVi/pin_cyl/cylinder/ ✓

● _cam* in filename
Camera id. numbers
0, 1, 2, 3

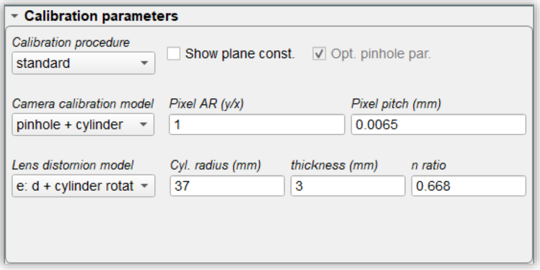
	Image filename	Plane parameters	Info
1	cyl_0mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 0.0	☑
2	cyl_5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 5.0	☑
3	cyl_10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 10.0	☑
4	cyl_15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, 15.0	☑
5	cyl_-5mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -5.0	☑
6	cyl_-10mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -10.0	☑
7	cyl_-15mm_cam*.tif	0.0, 0.0, 0.0, 0.0, 0.0, -15.0	☑

Plane parameters: β (°), α (°), γ (°), x (mm), y (mm), z (mm)

X0 (# column) 0
Y0 (# row) 0
Width (pixels) 2,560
Height (pixels) 2,160

Output

○ Same as input
Output folder path ple_CalVi/pin_cyl/cylinder/ ✓
Root of output files pyCal ✓



Calibration parameters

Calibration procedure standard
□ Show plane const.
☑ Opt. pinhole par.

Camera calibration model pinhole + cylinder
Pixel AR (y/x) 1
Pixel pitch (mm) 0.0065

Lens distortion model e: d + cylinder rotat
Cyl. radius (mm) 37
thickness (mm) 3
n ratio 0.668

13. Use of the calibrated mapping functions

After calibrating the camera system with the aid of the CalVi interface, the PaIRS-PIV library can be imported in a Python environment to read the output calibration files and compute the image coordinates of the projections of physical points with assigned world coordinates.

The following code reports an example of such an use of the PaIRS-PIV library:

```
import numpy as np
from PaIRS_UniNa.PaIRS_PIV import MappingFunction
mapFun=MappingFunction()
calFileName=['calib\pyCal1.cal','calib\pyCal2.cal']
mapFun.readCal(calFileName) # readCal reads the calibration files
numPoints=4 # let's use 4 points
# selecting double (np.float64) or single (np.float32 faster) precision
type=np.float64
# selecting the point in the world reference system
points=np.array([[1,2,3]]*numPoints,dtype=type,order='C')+\
       np.array(np.arange(numPoints)*0.001,dtype=type)[:,None]
cam=-1 # selecting the camera (starting from 0);
# if cam is equal to -1 then the mapping function is evaluated for all the cameras
X=None  # Output vector should be already allocated or None
X1=mapFun.worldToImg(points,cam,None)
#in output X1 is equal to X if correctly allocated
print (f'id(X)={id(X)} id(X1)={id(X1)} same Id={id(X)==id(X1)}')
#print (X1)

type=np.float32
points=np.array([[1,2,3]]*numPoints,dtype=type,order='C')+\
       np.array(np.arange(numPoints)*0.001,dtype=type)[:,None]
cam=0
# a variable to be passed should be allocated as shown in the following
if cam == -1: X=np.zeros([mapFun.nCam, numPoints,2],dtype=type,order='C')
else: X=np.zeros([ numPoints,2],dtype=type,order='C')

X1=mapFun.worldToImg(points,1,X)
print (f'id(X)={id(X)} id(X1)={id(X1)} same Id={id(X)==id(X1)}')
#print (X1)

# of course you may use the output variable as an input for a successive call
X2=mapFun.worldToImg(points,1,X1)
print (f'id(X2)={id(X2)} id(X1)={id(X1)} same Id={id(X2)==id(X1)}')
#print (X1)

# if you need just one point simply pass to worldToImgPoint
# a list with the three coordinates
# in output you get a point in the Img reference system
# in this case cam cannot be -1
point=mapFun.worldToImgPoint([0,0,0],cam)
print(f'{point.x} {point.y}')

# example of mapping function inversion (from image to world coordinates)
# selecting type
```

```
for Type in [np.float64, np.float32]:
  print(f'Type={Type} ******************************************')
  for cam in [-1,0]:
    print(f'------------  cam={cam}  z ')

points=np.array([[1,2,3]]*numPoints,dtype=Type,order='C')+np.array(np.arange(numPoints)*0.001,dtype=Type)[:,None]
    X1=mapFun.worldToImg(points,cam,None)# In output X1 is equal to X if correctly allocated
    z=np.array(points[:,2].T,order='C')
    x=mapFun.imgToWorld(X1,cam,z)

    print (x-points)
    xx=np.copy(points)
    #adding some noise
    maxNoise=1
    xx[:,0:2]+=np.random.default_rng().random([xx.shape[0],2])*maxNoise
    if cam ==-1:
      xx=np.tile(xx,[2,1,1])
    x=mapFun.imgToWorld(X1,cam,xx)
    print(f'cam={cam} Type={Type} x,y,z')
    print (x-points)
```

14. Appearance

It is possible to set the light or dark mode of the CalVi interface by clicking on the logo **CalVi** and selecting the favorite option, as shown in Figure 13.

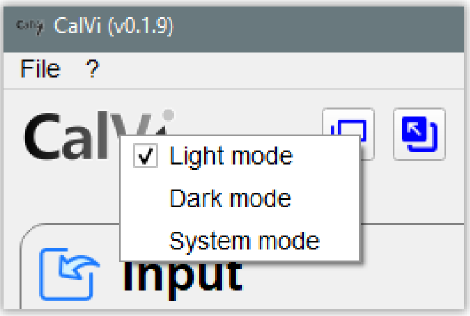
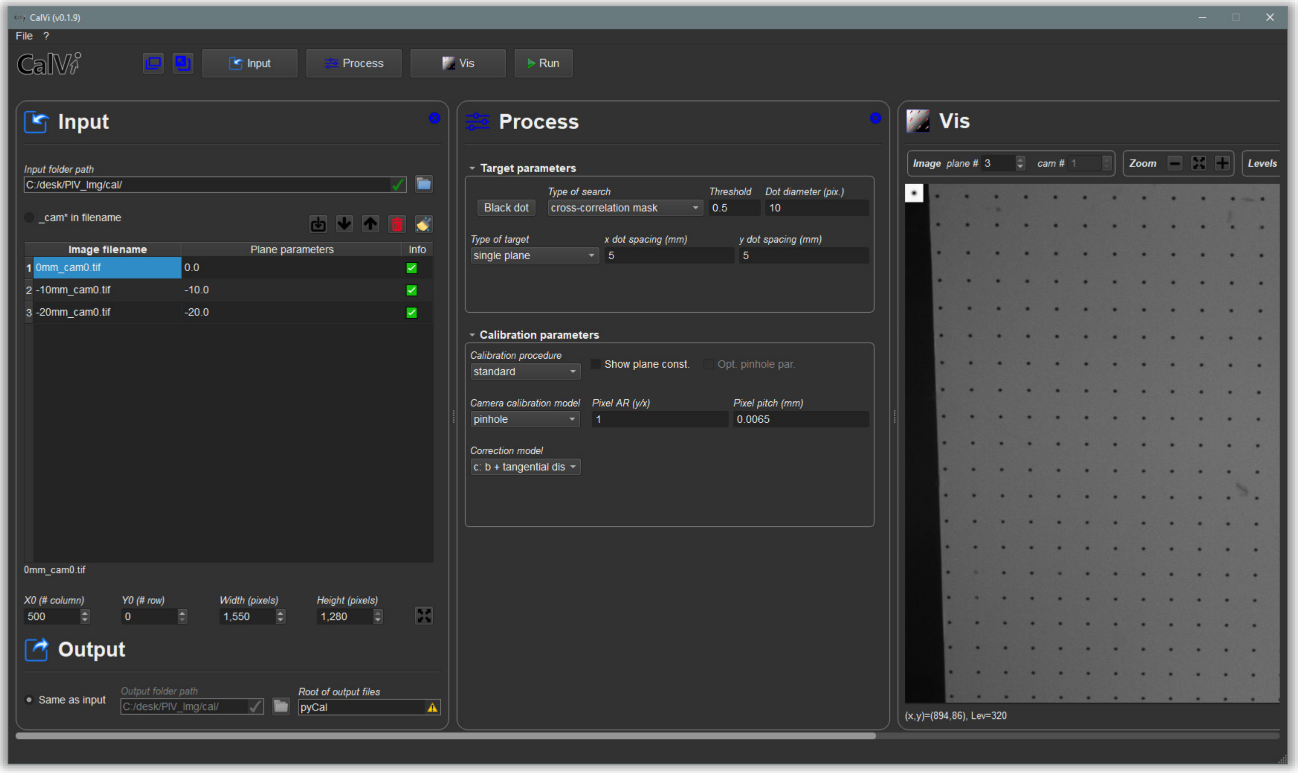


Figure 13 – Color mode context menu.

If you cannot wait for seeing the dark skin of CalVi, give a look at Figure 14.



Input folder path: C:/desk/PIV_Img/cal/

○ _cam* in filename

	Image filename	Plane parameters	Info
1	0mm_cam0.tif	0.0	✓
2	-10mm_cam0.tif	-10.0	✓
3	-20mm_cam0.tif	-20.0	✓

0mm_cam0.tif

X0 (# column): 500
Y0 (# row): 0
Width (pixels): 1,550
Height (pixels): 1,280

● Same as input
Output folder path: C:/desk/PIV_Img/cal/
Root of output files: pyCal

Target parameters
Black dot
Type of search: cross-correlation mask
Threshold: 0.5
Dot diameter (pix.): 10
Type of target: single plane
x dot spacing (mm): 5
y dot spacing (mm): 5

Calibration parameters
Calibration procedure: standard
□ Show plane const.
□ Opt. pinhole par.
Camera calibration model: pinhole
Pixel AR (y/x): 1
Pixel pitch (mm): 0.0065
Correction model: c: b + tangential dis

plane # 3
cam # 1

(x,y)=(894,86), Lev=320

Figure 14 – CalVi in the dark mode.

Moreover, it is possible to increase or decrease the font size of the graphical user interface via the following keyboard sequences:

`Ctrl+0`:	Set the default font size (14 pixels).
`Ctrl+1`:	Decrease the current font size by 1 pixel.
`Ctrl+9`:	Increase the current font size by 1 pixel.
`Ctrl+Shift+1`:	Set the font size to the minimum value allowed (8 pixels).
`Ctrl+Shift+9`:	Set the font size to the maximum value allowed (20 pixels).

References

[1] Paolillo, G., & Astarita, T. (2020). "Perspective camera model with refraction correction for optical velocimetry measurements in complex geometries". *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 44(6), 3185-3196**.** doi: 10.1109/TPAMI.2020.3046467.

[2] Paolillo, G., & Astarita, T. (2021). "On the PIV/PTV uncertainty related to calibration of camera systems with refractive surfaces". *Measurement Science and Technology*, 32(9), 094006. doi: 10.1088/1361-6501/abf3fc.

About the authors

Gerardo Paolillo received a Master's degree in Aerospace Engineering and a PhD degree in Industrial Engineering from Università di Napoli "Federico II" in 2015 and 2018, respectively.

He is currently a Research Associate in the Department of Industrial Engineering at Università di Napoli "Federico II".

His research interests lie in the area of experimental fluid mechanics, with focus on applications of unsteady jets to flow control and electronics cooling, investigation into dynamics of turbulent Rayleigh-Bènard convection and development of 3D optical velocimetry techniques.

Tommaso Astarita received a Master's degree in Aeronautical Engineering in 1993 and a PhD degree in Aerospace Engineering in 1997, both from Università di Napoli "Federico II".

He was Post-doc at the von Kàrmàn Institute for Fluid Dynamics and he is currently full Professor of Fluid Mechanics at Università di Napoli "Federico II".

His main research interests are dedicated to the experimental study of problems in the fields of fluid mechanics and convective heat transfer, in particular, the application and development of IR thermography and stereoscopic and tomographic PIV techniques for fluid mechanics problems.