

# M\_CALIB user manual, v0.8.6

Martin Dubs, FMA

## Summary

M\_CALIB replaces the calibration of meteor spectra with laser calibration images using IRIS for the determination of laser point coordinates and EXCEL for the fitting of parameters for the distortion model. These parameters are required for the change from image coordinates to the orthographic projection, which is used to linearize meteor spectra. It is derived from `I_calib6.py`, described in [Calib Python manual 2.pdf](#). It provides a GUI for easier navigation and extended capabilities (conversion of AVI-files and peak sum of calibration images). The GUI was designed with PySimpleGUI

## History

The main changes from `I_calib` were:

- (in `s_calib`) Whereas the previous versions were limited to laser calibration lines (multiple orders of a single wavelength), the present version allows calibration of different spectral lines according to a line list which may contain suitable lines from a calibration lamp (e.g. Hg or HgAr lamp). The new versions allow both laser calibration and spectral line calibration. Both are described separately.
- In addition to a polynomial fit, the present version also allows the fit of an effective focal length, with approximation of the distortion function as a square root of radius see: [https://meteorspectroscopy.files.wordpress.com/2018/01/meteor\\_spectroscopy\\_wgn43-4\\_2015.pdf](https://meteorspectroscopy.files.wordpress.com/2018/01/meteor_spectroscopy_wgn43-4_2015.pdf) for details). This is particularly useful for longer focal length, where only a few calibration lines are available. The resulting fit is more stable and works also in cases where the polynomial fit gives unrealistic results.
- The naming of the configuration files has been modified. Instead of a single filename `m_set.ini`, different filenames, preferably `m_*.ini` can be used to distinguish different cameras, calibrations etc.
- For input images both `*.fit` and `*.png` are allowed (for backward compatibility the `*.BMP` format can also be used). The PNG image format was selected for two reasons: It is more compact than BMP and it can be used together with PySimpleGUI (<https://pysimplegui.readthedocs.io/en/latest/>), with which the GUI was created.

The goal is to simplify the analysis of meteor spectra. Together with `m_spec.py` (with a similar GUI), the analysis of meteor spectra with IRIS, EXCEL, ImageTools and SpectroTools were converted to Python, so that only one language will be used for the complete analysis. In a next step `m_calib` will be made a part of `m_spec`, so that only one script will be needed for the calibration and analysis of meteor spectra.

Python was chosen, because

- it contains all the necessary tools to do the analysis
- it finds widespread use in the astronomy community
- it is free
- it runs on different platforms

I was inspired to use Python by Giovanni Leidi, who was giving a talk about the use of Python for the analysis of spectra in the spectroscopy workshop at OHP 2018 (<https://www.shelyak.com/ohp-spectro-star-party-2018/>).

Note of caution: I am new to Python, so the script presented here may not be the best solution. Some things have been done in a complicated way, copying examples from different sources and trying to make it work. It certainly should be improved for clarity and safety of operation. Therefore I hope you will suggest improvements.

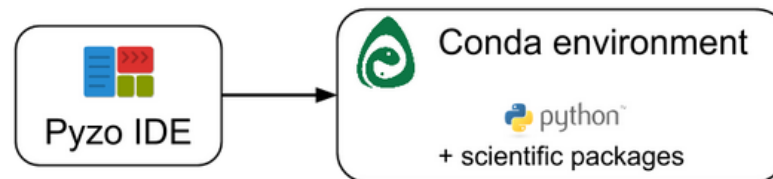
## Installation of Python

Python comes in different versions. Apart from the base version it requires additional packages, in particular:

Matplotlib  
Numpy  
Astropy  
Etc.

Some versions are not quite compatible. For that reason it is highly recommended to install a pre-packaged version, such as Anaconda, unless you are familiar with Python installing packages.

For running and editing Python scripts you need a development environment (IDE). I use PYZO, as recommended by Giovanni. It is installed from <http://www.pyzo.org/start.html>.



To get started with Pyzo, you need to install the Pyzo IDE (in which you write your code) and a Python environment (in which you run your code).

(for the actual writing of the script I use PyCharm, which allows a lot of checks and better editing of scripts, at the cost of a steeper learning curve, so I recommend you to start with pyzo or something similar)

The installation of Pyzo and Anaconda has been described elsewhere. See

[https://github.com/meteorspectroscopy/meteor-spectrum/blob/master/doc/M\\_spec%20Python%20manual\\_V09.pdf](https://github.com/meteorspectroscopy/meteor-spectrum/blob/master/doc/M_spec%20Python%20manual_V09.pdf) for details. Follow these instructions carefully, unless you are a Python expert.

In addition you have to install lmfit. In the Pyzo shell with the command:

```
>>> conda install -c conda-forge lmfit
```

Do not forget to download and install ffmpeg for the video conversion from

[www.ffmpeg.org/download.html](http://www.ffmpeg.org/download.html) or from <https://ffmpeg.zeranoe.com/builds/>

See the M\_spec-python-manual for details on installation.

## Laser calibration of meteor spectra

For the calibration of meteor spectra you need some calibration images, containing laser spectra with different orders in different positions of the image area. Ideally they cover most of the image area, in order to avoid extrapolation of the fitting functions. Details are described in the old manual

<https://meteorspectroscopy.files.wordpress.com/2018/01/processing-meteor-spectra-v151.pdf>

and are not repeated here. The theory is described in the following paper:

*A practical method for the analysis of meteor spectra*

*Martin Dubs, Peter Schlatter (WGN Journal, Ausgabe 43-4, 2015):*

[Meteor Spectroscopy WGN43-4 2015](#)

For simplicity, the spectra are in a single image, obtained earlier with ADD\_MAX2 in IRIS (get peak values from several images). In M\_CALIB, these images can be created directly from video files.

Calibration with a laser is particularly useful for wide angle lenses and/or low resolution spectra and is easier to interpret:



.\calib\caladd9.png

On each horizontal row 1 spectrum is displayed, with 4 laser lines in different orders, for 8 spectra in total. This was recorded with

WATEC 902 H2 ultimate image size: 720x576 pixels

(Tamron 12VG412ASIR ½", f: 4-12mm f/1.2, used at f: 8 mm)

Grating: 50x50 600 lines/mm Thorlabs

Calibration laser: 405 nm wavelength from eBay:

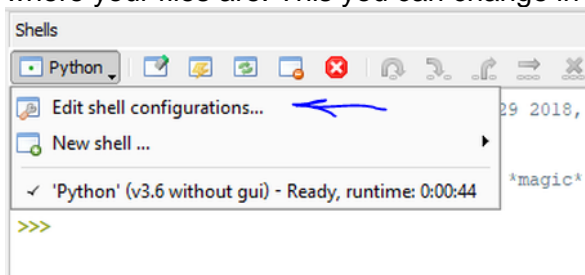
<http://www.ebay.com/itm/Diode-Laser-405nm-20mW-Violet-Purple-Laser-Dot-Module-w-cable-13x42mm-3-5-5V-/191174143732?ssPageName=ADME:L:OC:CH:3160>

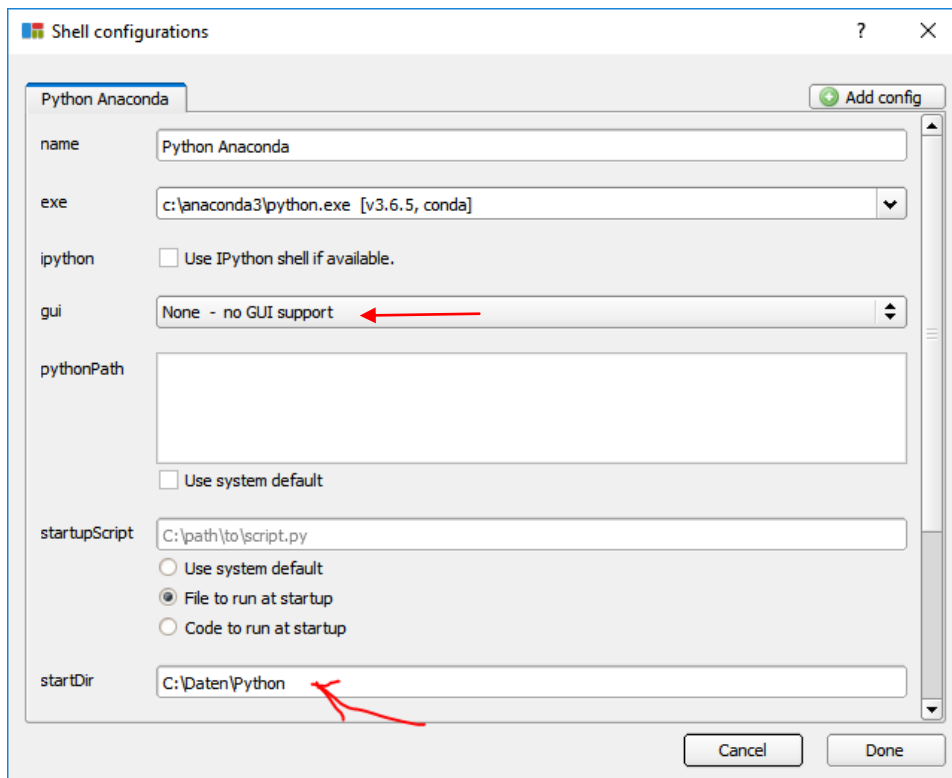
## Run the test example

For simplicity we start with the laser calibration as shown in the above figure with the image caladd9.fit or caladd9.png. The creation of this image will be shown later. This avoids uploading large video files. The files are in a folder calib.

Download the test example from the demo zip file or obtain a copy of the files from the author.

Put the Python script into the starting directory of Pyzo or make the startDir in Pyzo the directory where your files are. This you can change in Pyzo – Shell – edit shell configuration:



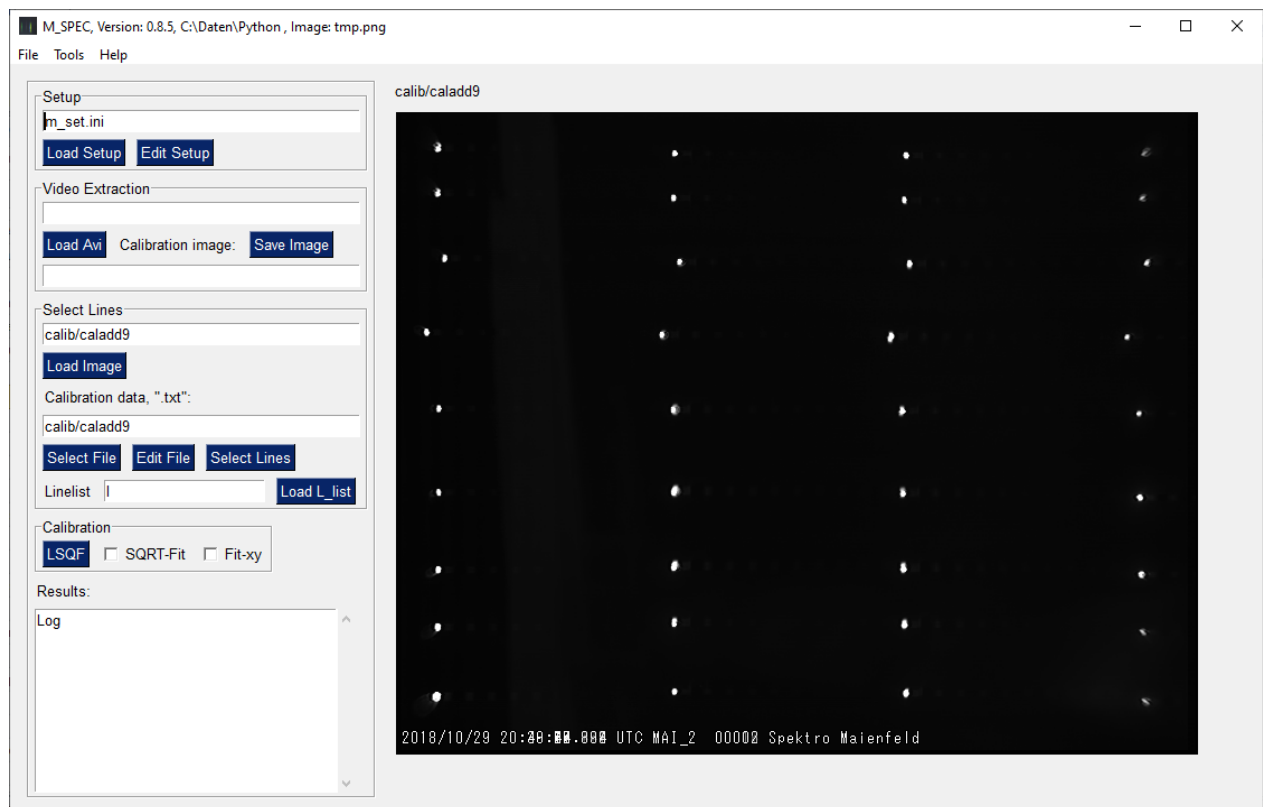


Note also that PYZO uses the Anaconda version of Python (at the moment I use v3.7.4) Put the calibration files in a subdirectory 'calib'. If the files are downloaded from a zip file, keep the directory structure as it is. This way the configuration works as it is supposed to do. Load the file m\_calib08.py with the main menu with File – Open File. You can inspect the script in the editor window. Start the script with Run, also from the Pyzo menu. You can also start the script from the shell window by entering run m\_calib08.py after the prompt and finish with ↵:

```
Shells
Python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:32:41) on Windows
(64 bits).
This is the Pyzo interpreter.
Type 'help' for help, type '?' for a list of *magic* commands.

>>> run m_calib08.py
```

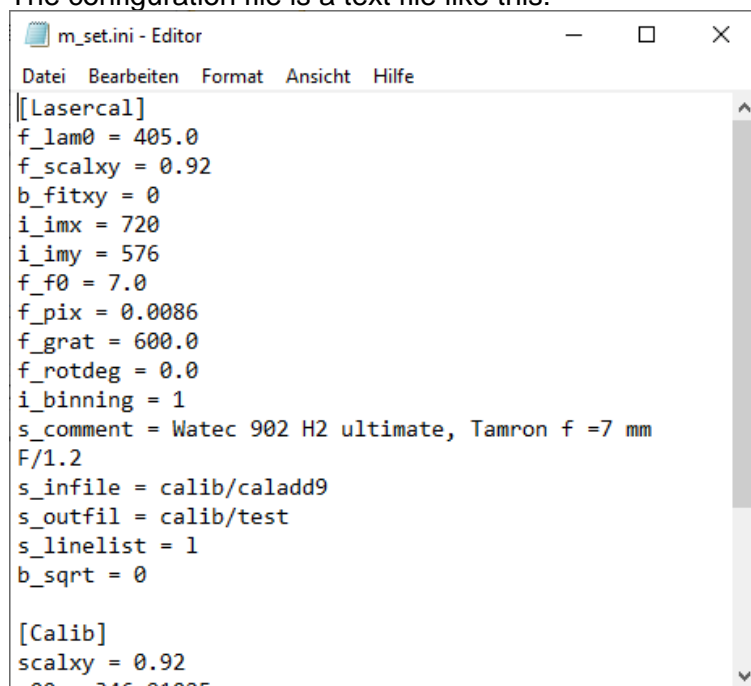
If you run the script for the first time, it will take a while, because all the necessary libraries have to be assembled. If the script does not run correctly, it is probably because some library is missing (see the manual [https://github.com/meteorspectroscopy/meteor-spectrum/blob/master/doc/M\\_spec%20Python%20manual\\_V09.pdf](https://github.com/meteorspectroscopy/meteor-spectrum/blob/master/doc/M_spec%20Python%20manual_V09.pdf) for details on the installation of library packages). While waiting, it may be a good idea to study the Pyzo documentation about the meaning of the other menu buttons in the shell and editor window. After the initialization the following window appears:



## Setup

The exact look may vary, depending on the content of the configuration file. At startup, it uses the configuration `m_set.ini`. Its content is updated at the end of each session with the button Exit, so you can close a session and resume the work later with the same configuration.

The configuration file is a text file like this:



The meaning of the parameters is as follows:

- **F\_lam0**: laser wavelength [nm] in case of a calibration with a laser, ignored for calibration lamp. The `f_` at the beginning indicates a floating point value

- **F\_scalxy** gives the ratio pixel height to pixel width of the camera (or the video capture device, 0.92 is the value determined from UFO analyzer.
  - **B\_fitxy** is a boolean value, 0 for False, do not use scalxy as a fit parameter, keep it fixed, 1 for True, obtain a better fit with scalxy as fit parameter. If scalxy deviates far from the expected value, set b\_fitxy to 0
  - The following parameters (imx, imy, f0, pix and grat) are image size, focal length [mm], pixel size [mm] and number of lines /mm of the grating [mm<sup>-1</sup>]. Its approximate values are used for the start of the fit, but the least square fit works with a wide range of values, takes a bit longer maybe.
  - **F\_rotdeg** is the tilt of the spectrum (deviation angle from x-axis, measured in degrees [°].
  - **F\_binning** indicates if the calibration image was binned. In that case the distortion parameters are converted to an unbinned image. The spectrum script works with the correct value if it given the correct value of binning (binning is useful for 4k videos, it speeds up the processing considerably, without losing much precision).
  - **S\_comment** gives a short description of the equipment, in case you use different, each with its own configuration file.
  - **S\_infile** gives the path and filename of the calibration image (without extension). At present images with extension PNG and FIT can be read. PNG is used because it is compatible with the user interface based on PySimpleGUI and FIT is used for internal calculations and as a standard astronomical image format. It contains detailed information about the processing in its header, important for the documentation.
  - **S\_outfil** (s\_for string) gives the path and name of the output of the line positions, as determined by the next step Select Lines (extension .txt added by the script).
  - **S\_linelist** (.txt) is used for the calibration of lamp or meteor spectra. It contains the (wavelength\*order) of the selected lines and a comment for each line. A value 'l' (lower case 'L') for the linelist means a laser calibration with lam0.
  - **B\_sqrt** is a boolean flag: 0 for polynomial fit, 1 for SQRT fit. The latter has one parameter less to fit, is more stable and useful for low distortion lenses and/or longer focal length.
  - Under the heading [Calib] follow the results of the calibration, to be discussed later.
- All these parameters may be edited with a text editor or easier within the GUI

### Select configuration

The actual configuration used by the script can be checked with the button 'Edit Setup' in the main window at top left.

Parameters

Settings

Lasercal

Lasercal

f\_lam0 405.0

f\_scalxy 0.92

b\_fitxy 0

i\_imx 720

i\_imy 576

f\_f0 7.0

f\_pix 0.0086

f\_grat 600.0

f\_rotdeg 0.0

i\_binning 1

s\_comment Watec 902 H2 ultimate, Tamron f=7

s\_infile calib/caladd9

s\_outfil calib/caladd9

s\_linelist l

b\_sqrt 0

m\_set.ini

SaveC Apply Cancel

Options

Parameter	Value
Zoom	1.0
<input checked="" type="checkbox"/> debug	
<input type="checkbox"/> fit-report	
<input checked="" type="checkbox"/> scale_win2ima	
Comment	
Watec	

Important in the setup is the field 's\_outfil', which gives the path and filename to the textfile, where the results of the line positions are stored. When starting a new calibration, it might be advisable to give it the same name or a similar name as the image file 'infile' or a name indicating specifics like date or camera setup. With 'Apply' the configuration is. A different name for the configuration file can be saved with SaveC. With Cancel the configuration remains as before Edit Setup was chosen, changes are ignored. Best to try it out, easier than explaining!

In addition the setup window allows to change some options.

Debug is mostly for testing purpose. Fit-report gives detailed information about the least square fit in the shell window. You probably do not need it either. If you check scale\_win2ima the image is scaled by the factor zoom and the window size changed that the image fits inside. Use a zoom value according to original image size and screen size. It works only after closing the program and restarting it (peculiar to PySimpleGUI). If this is not checked, the image fits the available space in the window, the zoom factor is ignored (For the determination of line position, the original image is used, with the binning as given by i\_binning).

### Select points of the different orders

Once the configuration has been set, we can load a calibration image for analysing with "Load Image"

If you would like to do another analysis, you can type the new filename in the input field Calibration data or select an existing file with "Select File"

Start the calculation of the line positions with 'Select Lines'. A new window appears:



- Start with the lowest order at the left and click with the mouse on the spot you would like to select:



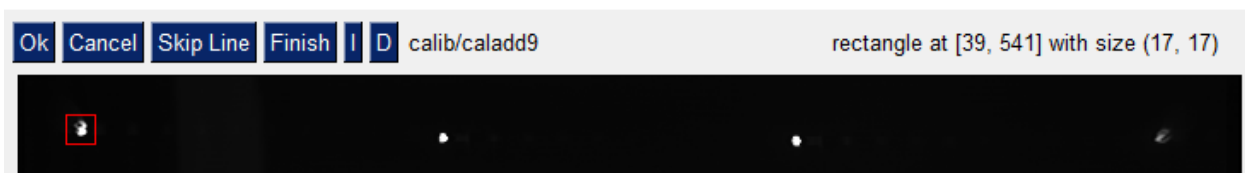
The program calculates a Gaussian fit of the spectrum and saves the result. A yellow circle shows the selected point. A label shows wavelength\*order. Continue with the next point. Since all lines have the same spacing, it is not important if the first point is actually the zero order. This is different with a calibration lamp, where the lines have to be selected precisely as give in the list.

- If you have selected all orders of one spectrum, press "OK", the coordinates and widths of the different orders will be saved to the text file.
- If you made a mistake, press 'Cancel', to start with the same or another spectrum
- With 'i' and 'd' you can adjust the image contrast
- With 'Skip Line' you can skip a line (not needed here, but useful if you do not see a line corresponding to the linelist, or when it overlaps another line)
- Press "Finish" again after the last spectrum

#### Note:

If you have problems with the selection of points, you may change the size of the rectangle, in which the Gaussian peak is searched and fitted (default is a rectangle with corners  $x_0, y_0 \pm 10$  pixels). You may draw a rectangle in the image by selecting one corner and draw with the mouse down to the opposite corner and release the button there:





If the rectangle is too small, you may miss the point when selecting it, if the rectangle is too large, the Gaussian fit may fail. You will get some error messages in the shell, try with another size! Drawing the rectangle does not select the point. Type 'r', then start a new spectrum. Notice, this is still a test version.

### Inspection of the calibration text file

With "Edit File" you can check the measurements.

It should look like this:

```

Edit Text File: C:/Daten/Python/calib/caladd9.txt

# 1 calib\caladd9.fit
40.30 540.48 4.33 6.49 0.00
251.81 535.69 3.91 4.33 405.00
458.63 533.89 3.97 4.75 810.00
672.12 536.11 6.26 6.16 1215.00
0.00 0.00 0.00 0.00 0.00
# 1 calib/caladd9.fit
46.07 442.10 3.50 5.42 0.00
256.12 438.87 3.75 4.87 405.00
461.64 437.10 3.84 5.83 810.00
673.80 438.04 3.75 4.61 1215.00
0.00 0.00 0.00 0.00 0.00
# 2 calib/caladd9.fit
40.84 308.04 3.45 4.98 0.00
251.01 307.05 4.81 6.65 405.00
455.05 305.95 4.25 6.39 810.00
666.76 303.64 3.64 3.84 1215.00
0.00 0.00 0.00 0.00 0.00
# 3 calib/caladd9.fit
40.28 164.32 3.93 5.25 0.00
251.50 167.78 4.90 7.33 405.00
456.15 165.90 4.27 7.47 810.00
668.87 160.09 4.38 5.13 1215.00
0.00 0.00 0.00 0.00 0.00
# 4 calib/caladd9.fit
38.80 51.05 5.95 7.08 0.00
251.52 55.64 3.78 4.67 405.00
458.53 54.31 4.17 6.06 810.00
672.53 46.99 5.85 5.73 1215.00
0.00 0.00 0.00 0.00 0.00

```

- The columns denote x-coordinate, y-coordinate, FWHM(x) and FWHM(y) of the spots, as determined by the Gaussian fit  
The last column is wavelength\*order (added for analysis of calibration spectra)
- The lines with # are comments and ignored for the fit.
- Each spectrum ends with a line of zeros as separator
- Additional lines of zeros in between are ignored, also spectra containing only one line (there is nothing to fit there)
- If you forgot one or more spectra, you can run the script again and add the missing spectra
- If you recorded one spectrum twice or entered wrong data points, you may erase these in the text file manually and save the file again for the following step.

This can be done directly in the Edit File window, followed by 'Save'

In the Results window it should say 'Finished, saved caladd/caladd9.txt'.

### Least square fit

For the determination of the distortion parameters, a linear dispersion law is postulated in the transformed image

( $x' = x_0 + i \cdot \lambda_{i0} \cdot \text{disp}_0$ ,  $i = 0, 1, 2, \dots$  for laser calibration, used in old version

In present version:

$x' = x_0 + \lambda_{i0} \cdot \text{disp}_0$ ,  $i = 0, 1, 2, \dots$ , where  $\lambda_{i0}$  = wavelength\*order of line[i] from line list, both for laser calibration and spectral line calibration

$y' = y_0$

from these coordinates the transformation to the image coordinates (as required in the coordinate transformation of the meteor images) is calculated by a transformation to polar coordinates around the image axis at coordinates ( $x_{00}, y_{00}$ ), rotation of the image by an angle  $\text{rotdeg}$  (converted to  $\text{rot}$ , angle in radians,  $\text{rot} = \text{rotdeg} \cdot \pi / 180$ ), transforming the radial coordinate by

$r = r' \cdot (1 + a_3 \cdot r'^2 + a_5 \cdot r'^4)$  (plus higher terms in the future),

transforming back to Cartesian coordinates with the axis remaining at ( $x_{00}, y_{00}$ )

For those interested in the details, here is the corresponding code from the Python function.

( $x_0, y_0$ s) correspond to ( $x', y'$ ).

$x_{y0} + xy_{00}$  corresponds to ( $x, y$ ) in the calibration image, to be fitted to the coordinates in the calibration file (testcal.txt in the example)

```
xy00 = [x00,y00]
xyl = [x0s+lam/disp0,y0s] - np.array(xy00) # coordinates of laser lines wrt, optical axis
r = np.sqrt(xyl[0]**2+xyl[1]**2) # polar coordinates
phi = np.arctan2(xyl[1],xyl[0]) # polar coordinates
phi += rot # apply rotation
r = r*(1 + a3*r**2 + a5*r**4) # transform radial coordinate
xyr = np.multiply(r,[np.cos(phi), np.sin(phi)/scalxy]) # return to cartesian coordinates
return (xyr + xy00)
```

-----

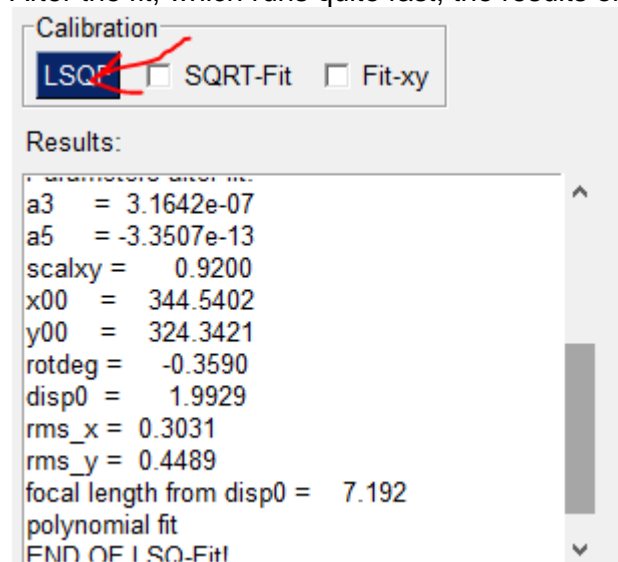
NB: For the sqrt-fit the following equation is used for  $r'$ :

$r = r' \cdot (1 + a_3 \cdot r'^2 + 1.5 \cdot a_3^2 \cdot r'^4)$

### Run the fit for the example caladd9.txt with m\_calib.py

Let us assume that you have edited the file caladd9.txt and corrected all the errors (wrong points, wrong order, etc.) and you would like to start the least square fit with these data. Select 'LSQF'.

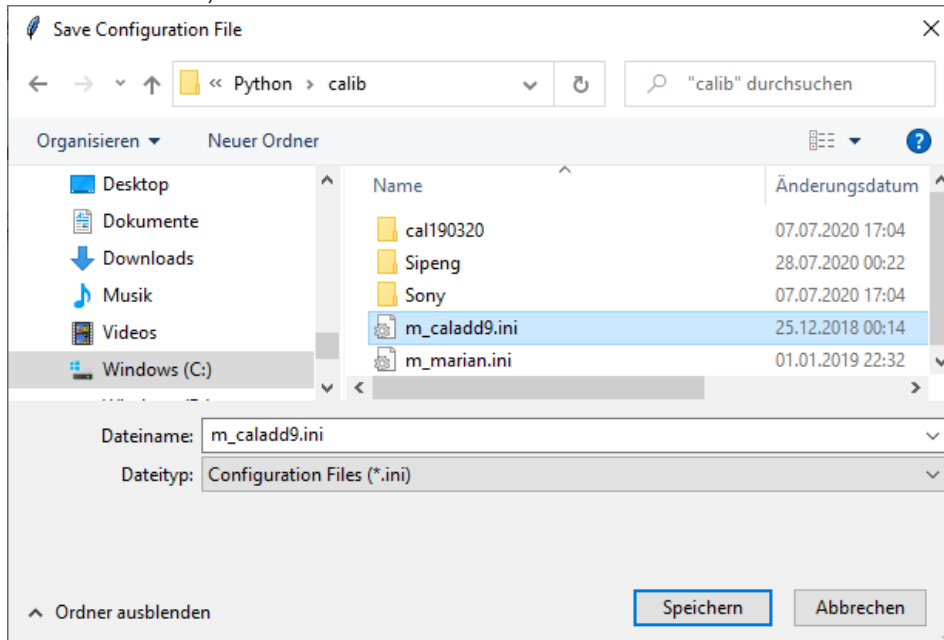
After the fit, which runs quite fast, the results of the least square fit appear in the log window:



At the same time they are stored in  $m\_cal.ini$  in the working directory (in order not to overwrite  $m\_set.ini$ , which may be used as a backup). If you like the result and want to use it with  $m\_spec$ , you may save the new configuration under  $m\_set.ini$  or another meaningful name (e.g.  $m\_set\_yymmdd.ini$  or  $m\_set\_sony24mm.ini$  etc.)

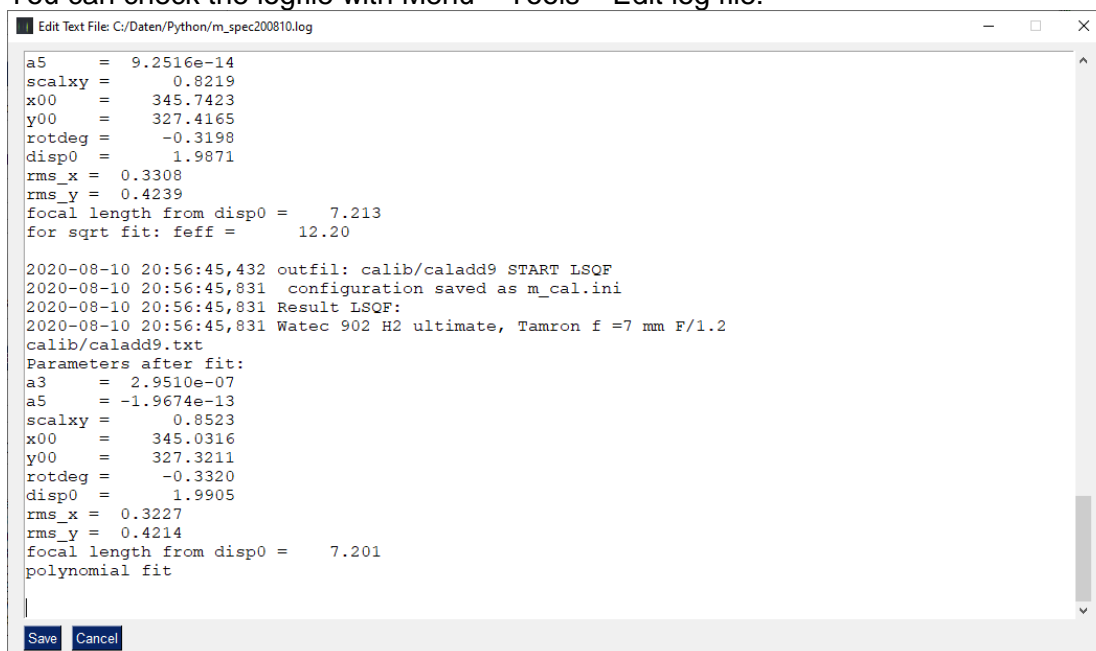
These parameters you may use to transform your meteor spectra to the orthographic projection. Notice that the fit runs quite fast, the Levenberg Marquard algorithm is very efficient for this kind of problem. In addition the results of the fit are shown in graphical form (helpful, if you have made any errors in the assignment of any lines).

I recommend to save the configuration under a suitable name, perhaps in the directory with the calibration data, such as caladd.ini:



The results are also stored in the Logfile m\_spec\_yymmdd.log

You can check the logfile with Menu – Tools – Edit log file:



(the filename is the same as for the script m\_spec.py, in view of a future integration of m\_calib into m\_spec).

In addition, the content of the result window is also written to a file Logyyyyymmdd\_hhmmss.txt for each session, which contains similar information in a shorter form. You may use either one.

## Use of spectrum lamp for calibration

For cameras with high dispersion, either with a long focal length or with a grating with large number of grooves/mm, not enough orders of the laser may fit on the detector size. Maybe a suitable laser is not at hand. In these cases it is possible to use a calibration lamp with several emission lines for calibration. Suitable lamps are low pressure Hg-lamps or Hg-Ar lamps. I have used successfully a lamp from Ocean Optics (<http://oceanoptics.com/product/hg-1/>) with several lines from UV to near-IR. An example is given in [https://meteorspectroscopy.files.wordpress.com/2018/01/meteor\\_spectroscopy\\_wgn43-4\\_2015.pdf](https://meteorspectroscopy.files.wordpress.com/2018/01/meteor_spectroscopy_wgn43-4_2015.pdf), although with low dispersion and somewhat problematic separation of spectral lines.



Therefore here an example with longer focal length is used (spectra recorded by Sipeng Yang). With a violet laser, only three orders fit on the image area, not enough for a good determination of the distortion parameters. With a green or red laser, only two orders fit in the image area, definitely not enough for a determination of the distortion parameters.

With the Hg-lamp, several spectral lines in addition to the zero order can be used for the fit:

Load image: calib/Sipeng/cal1\_peak\_5.png

5 spectra of a mercury argon lamp are shown, zero order at left, with first and order lines to the right. The spectra were recorded by Sipeng Yang<sup>1</sup>, thanks for using these files. Equipment:

1. Watec 902H2 ULTIMATE 1/2' CCD, 720\*576 Resolution
2. Lens: Hikvision HV0733D-6MP, F/0.95,  $f = 7-33\text{mm}$  zoom lens, operated at approx.  $f = 10\text{mm}$
3. Grating: Edmund 600l/mm transmission grating
4. Camera FOV: 30.4 degree

This is particularly useful for lenses with longer focal length and/or high resolution spectra and needs some knowledge about the spectrum of the lamp (which spectral line is where)

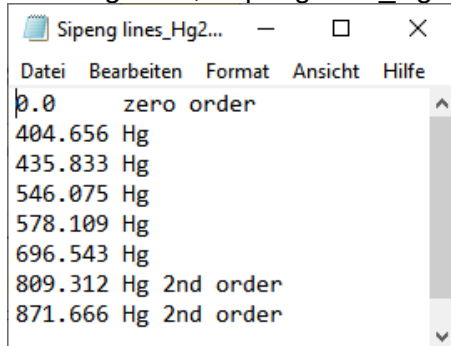
The bright line at left is the zero order, 5 Hg-lines 1<sup>st</sup> order are visible. Some second order lines are visible as well and can also be used for calibration.

### Line list

When working with a spectral lamp, a slightly different procedure from the laser calibration will be used. Instead of a single laser wavelength, a list with the wavelengths of the spectral lamp in different orders is required. This is a textfile with two columns, the first column contains the wavelength\*order (negative for negative orders, zero for zero order, wavelength for first order,

<sup>1</sup> Beijing National Observatory for Space Environment  
Institute of Geology and Geophysics, Chinese Academy of Sciences  
NO.19 Beitucheng Xilu Street, Chaoyang District, Beijing, China 100029

2\*wavelength for 2<sup>nd</sup> order etc. as used here also for the plotting of spectra. The 2<sup>nd</sup> column must be separated by a tab and contains text information (element, order or whatever you like to add). The script actually reads only the first column, but make sure you end the column with a tab for compatibility with other scripts, which will use this file. It can be created with a text editor (Windows Editor or Notepad++ which I prefer). For the image of Hg spectra above I used the following table, "Sipeng lines\_Hg2.txt":

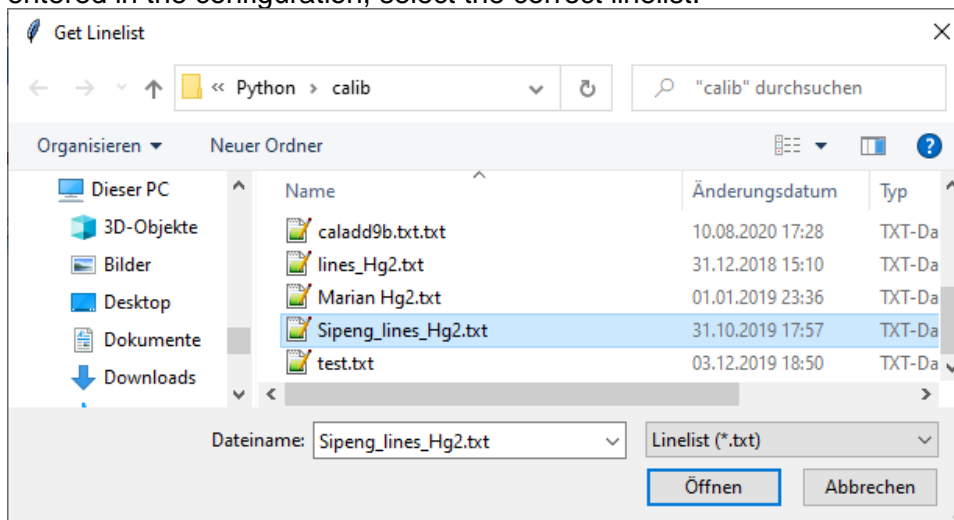


Wavelength (nm)	Element/Order
0.0	zero order
404.656	Hg
435.833	Hg
546.075	Hg
578.109	Hg
696.543	Hg
809.312	Hg 2nd order
871.666	Hg 2nd order

The line at 578.109 nm is actually the weighted average of two lines at 576.96 and 579.066 nm.

### Run the script for spectral lamp calibration

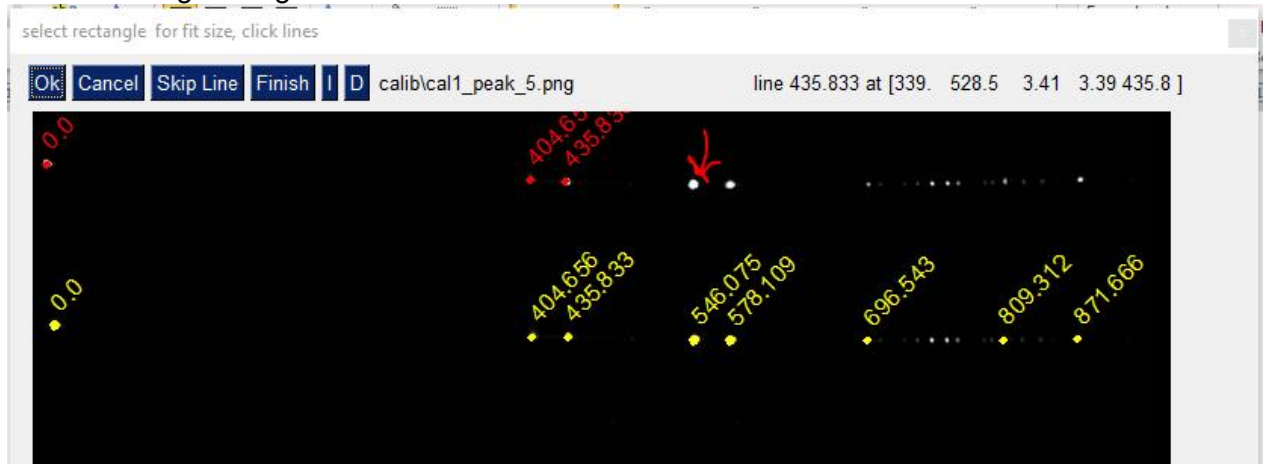
Run the script as before, with the configuration 'm\_set\_sipeng.ini' and if the linelist has not been entered in the configuration, select the correct linelist:



You can also change the linelist manually in the setup.

Finish the setup with 'Apply'.

Load the image and go to 'Select Lines'.

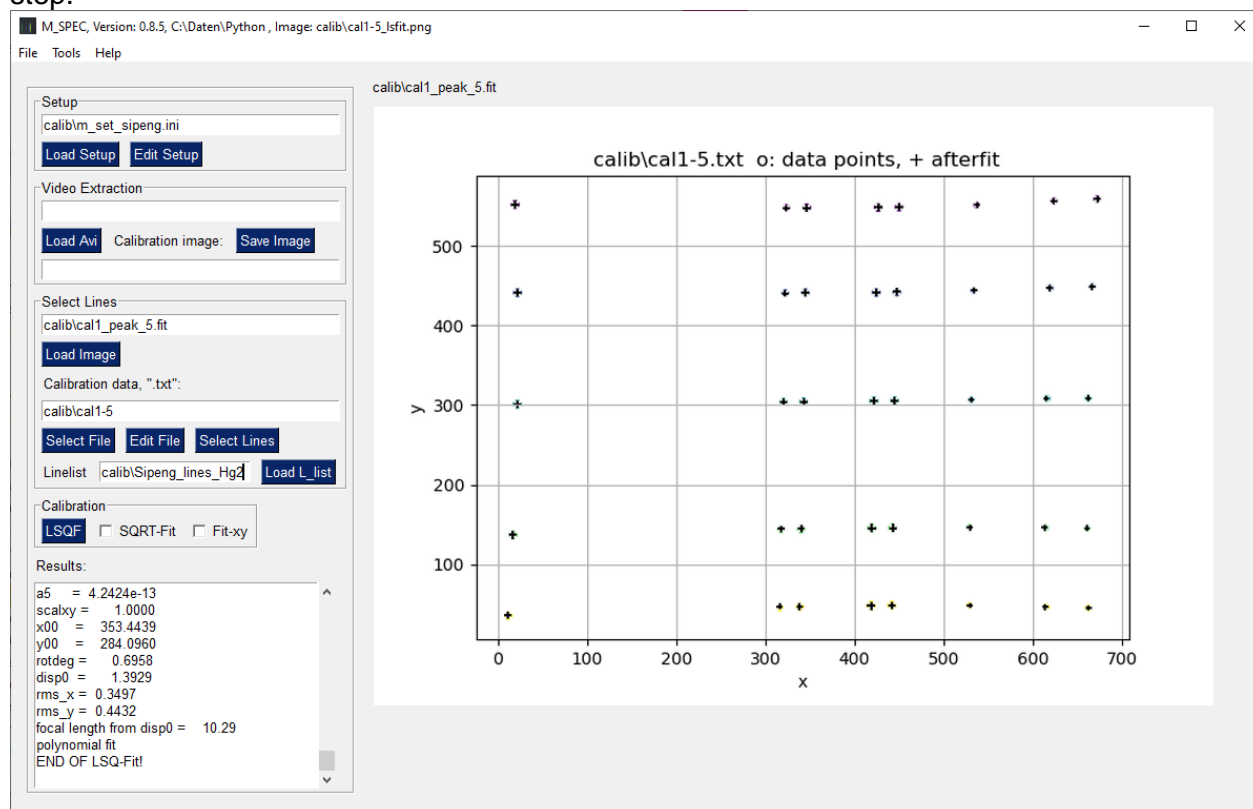


The selected lines are marked by a circle with wavelength\*order indicated. They have been selected to cover the full width of the sensor with regular spacings. Some weak lines have been omitted. Finish each spectrum with **OK**

Continue with the other spectra, Skip missing lines with 'Skip Line' (e.g. if the leftmost line is out of the image area or if a line is too weak). Select only the lines which you can assign correctly. Finish the list with **OK** again.

It is important that you choose the correct lines! If you select the wrong lines, the errors will be large. Start over with a new calibration file and select the correct lines!

Proceed the same way as for the laser calibration (edit the list if necessary, then go to the LSQF step).



If you would like to run the least square fit with other parameters (a sqrt fit would be suitable here for the fairly long focal length) go to the Setup and edit the corresponding field(s).

The results are from the logfile:

#### Result LSQF: **polynomial fit**

=====

Watec 902 H2 ultimate, f = 10mm

Sipeng\lamp\_video\cal1-5.txt

Parameters after fit:

a3 = 1.7052e-07

a5 = 6.5356e-13

scalxy = 0.9240

x00 = 352.6166

y00 = 283.4155

rotdeg = 0.6431

disp0 = 1.3906

rms\_x = 0.3387

rms\_y = 0.4269

focal length from disp0 = 10.31

polynomial fit

END OF LSQ-Fit!

#### Result LSQF: **square root fit**

=====

Watec 902 H2 ultimate, f = 8mm

Sipeng\lamp\_video\cal1-5.txt

Parameters after fit:

a3 = 2.7372e-07

a5 = 1.1239e-13

scalxy = 0.9671

x00 = 354.5030

y00 = 282.9148

rotdeg = 0.6752

disp0 = 1.3969

rms\_x = 0.3248

rms\_y = 0.4764

focal length from disp0 = 10.26

for sqrt fit: feff = 11.62

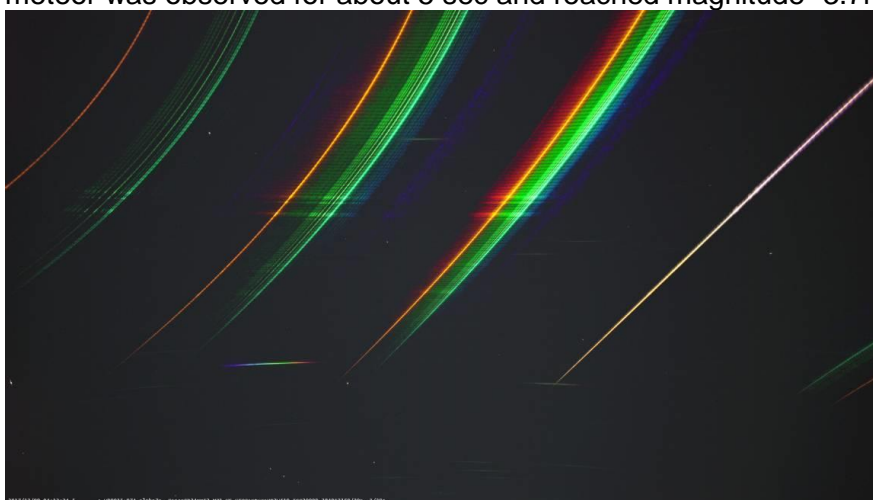
END OF LSQ-Fit!



Since the errors of the fit are similar, I would prefer the sqrt fit, but this is a matter of taste. Another possibility would be to constrain the value of  $\sigma_{\text{calxy}}$  to a more realistic value, as determined by UFO capture or calculated from the image size and pixel size and detector size. An rms error of  $< 1$  pixel is the accuracy which can be expected, subpixel accuracy is achieved with determining the line positions with a Gaussian fit.

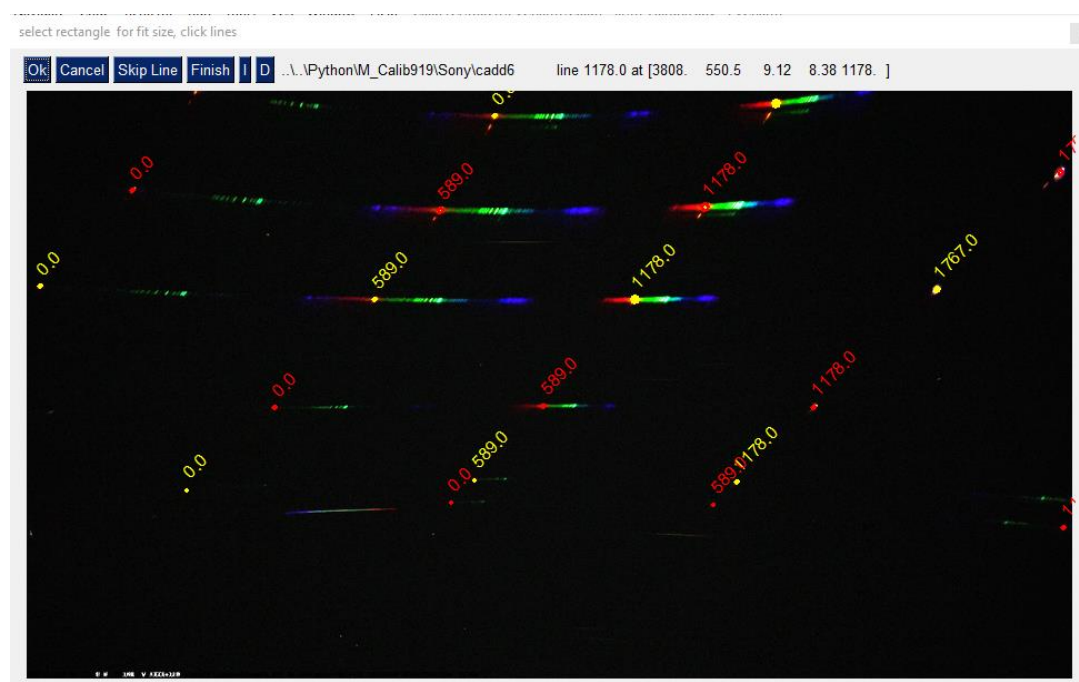
## Use of meteor spectra for calibration

When a meteor spectrum with identified spectral lines covers a large portion of the image area, it can also be used for the determination of the distortion parameters. This is useful if no laser or spectral lamp spectra have been recorded or if the camera setup has been changed in the meantime. I present an example which has previously been analysed with the old method. The spectrum was recorded by Koji Maeda with a Sony Alpha 7S equipped with a Canon 24 mm F/1.4 lens (used at F/2) and a 600 L/mm grating. The meteor spectrum was captured as a 4K video (image size 3840 x 2160 pixels) at 30 images/sec with SonotaCo UFOCaptureHD. The meteor was observed for about 5 sec and reached magnitude -3.7m.



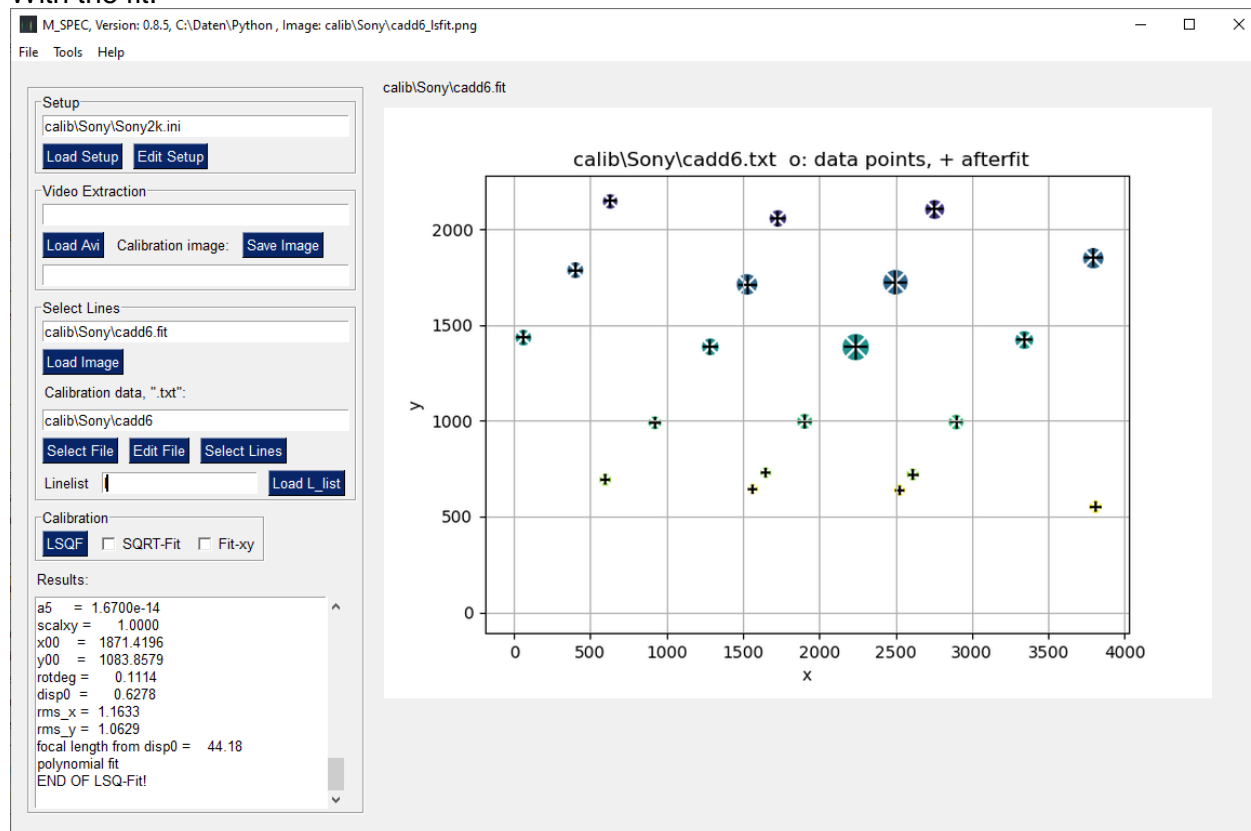
M20171209\_041224\_JPMZ1, Koji Maeda

The Na-lines are easily identified and can be used for calibration. The spectrum pre-processing was done with `m_pipe6` including the AVI conversion and background subtraction. Several colour fit-images were peak-added and saved as `cadd6.fit`.



For simplicity I only used the Na-lines, therefore no list of spectral lines was needed, I used the “laser” calibration. Notice that the orders are not correct, important is that the wavelength differences are constant between orders.

With the fit:



The size of the data points corresponds to the Gaussian width of the lines, increased by saturation, movement of the meteor and meteor train

The fit parameters agree with earlier calculations:

Result LSQF:

=====

Sony alpha, f 24 mm

sony/cadd6.txt

Parameters after fit:

a3 = 4.3292e-08

a5 = 1.6700e-14

scalxy = 1.0000

x00 = 1871.4196

y00 = 1083.8579

rotdeg = 0.1114

disp0 = 0.6278

rms\_x = 1.1633

rms\_y = 1.0629

focal length from disp0 = 22.09

polynomial fit

END OF LSQ-Fit!

With these parameters, also saved as m\_set.ini, the full spectrum can be analysed.

### Analysis of spectrum M20171209\_041224\_JPMZ1

(see: <http://sonotaco.jp/forum/viewtopic.php?p=47795#47795>)



The calibration with the Na-lines was used to calibrate the meteor spectrum, analysed with m\_pipe62.py and the distortion parameters from m\_set.ini.

After adding 23 registered files (about the 1<sup>st</sup> second of the meteor (near the bottom of the image9 and calibrating with Mg- and Na-lines the following result was obtained:

A linear fit was used:

polynom for fit lambda c: [ 1.259e+00 -1.503e+03]

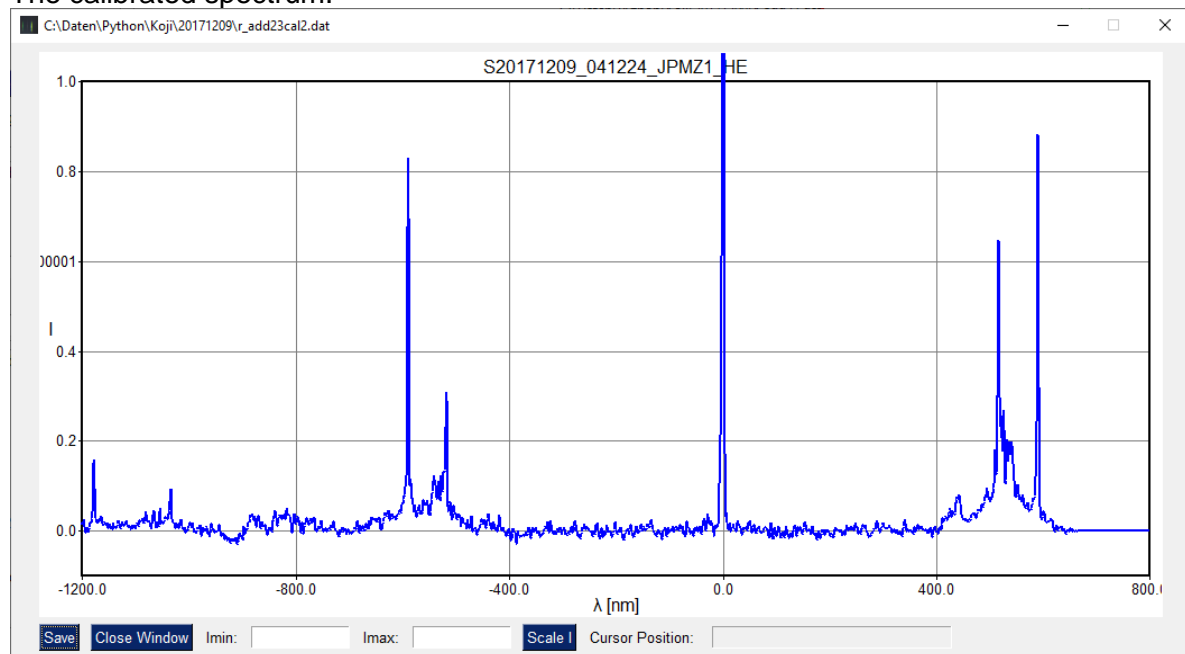
pixel	lambda	fit	error
257.95,	-1178.00,	-1177.83,	0.1668
725.21,	-589.00,	-589.48,	-0.4819
1193.86,	0.00,	0.62,	0.6196
1603.37,	517.50,	516.25,	-1.2451
1661.89,	589.00,	589.94,	0.9405

rms\_x = 0.7847

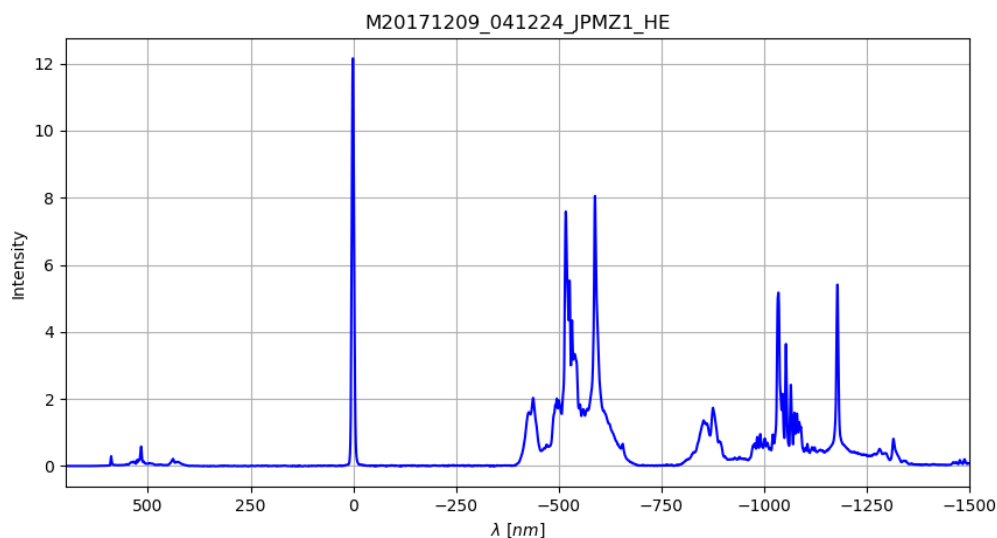
spectrum C:\Daten\Python\Koji\20171209\r\_add23cal.dat saved

Not surprisingly, a linear fit works well.

The calibrated spectrum:



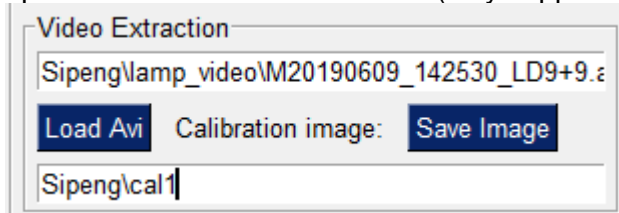
Below, the spectrum obtained from adding all image frames(123)



The spectrum was plotted with reverse wavelength scale, to show the strongest lines to the right of the zero order. The actual first order visible around 500 nm was very weak, because it was mostly outside the image area.

## Creating calibration images from video files

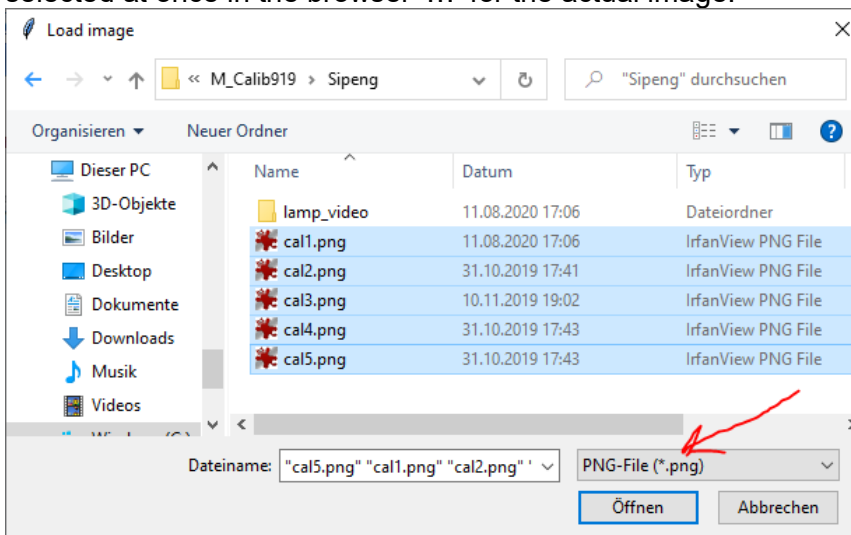
For the previous versions of this calibration script, the images used for calibration had to be created with different software (IRIS). In order to streamline the workflow, the creation of calibration images has been included in this script. For calibration images, short video clips of the laser or spectral lamp are recorded. These can be read with the button 'Load Avi', which opens a window for file selection (only supported format AVI, as produced by UFO capture):



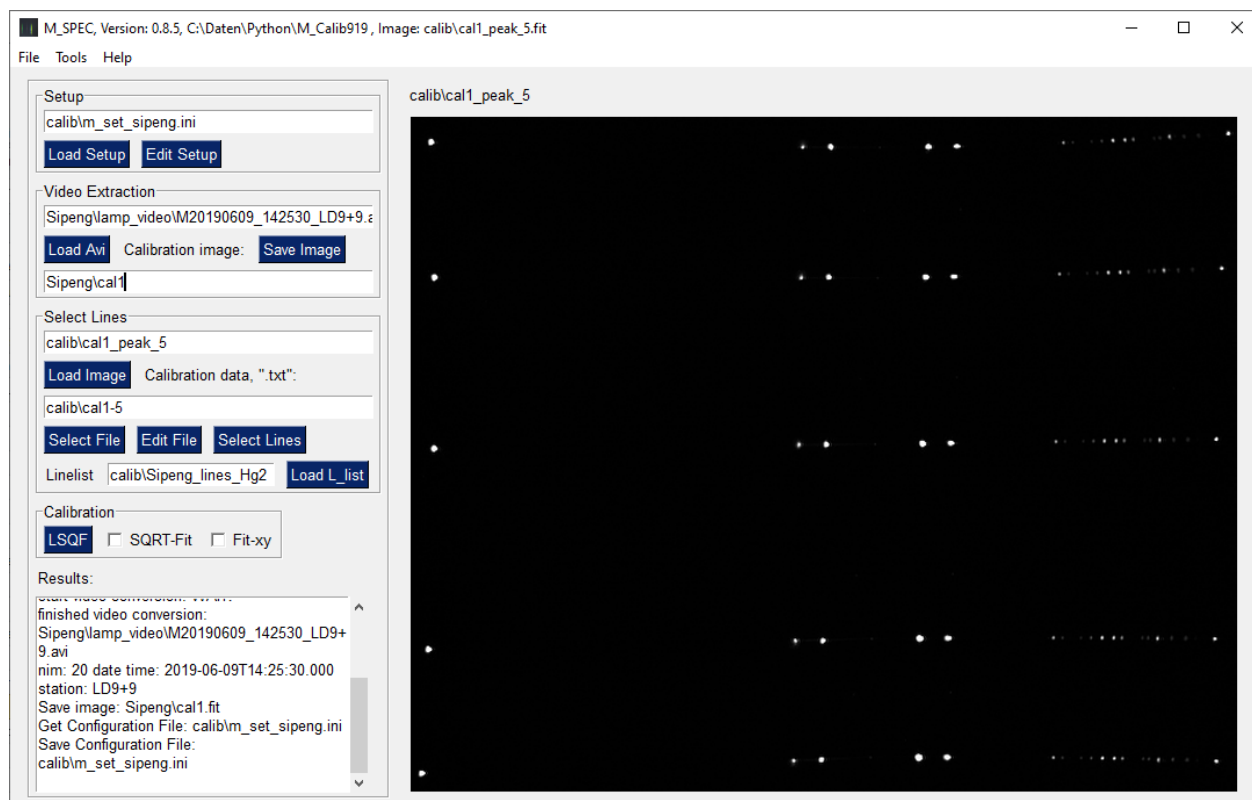
From the selected video file, a number of frames (20 at present) are converted to png images and from these a peak image and an average image are calculated with 'Load Video'. After a short delay the image can be saved with an appropriate filename by selecting the directory with the browser button '...' and adding the filename in the image window:  
If the videos are not in the demo files (for saving file size), try with your own meteor video file (a background image will be produced).

## Making peak images from single calibration images

You can load an image with a single spectrum, determine the line positions and then load another image, repeating these steps until all spectra are measured. However, it is more convenient to create a peak image from all the relevant calibration spectra. Previously this had to be done with IRIS or some other imaging software. In the present version of m\_calib, this is simplified considerably. Instead of loading the image files one by one, the whole series can be selected at once in the browser '...' for the actual image:



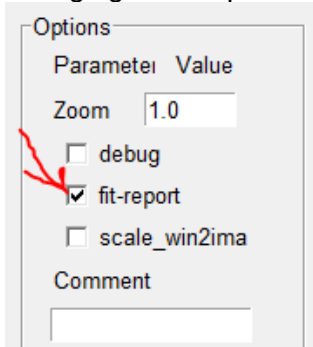
This creates two images (both in FIT and PNG format) named cal1\_peak\_5.png and cal1\_ave\_5.png (the filename is created from the name of the first image plus ', -peak-' plus number of images. Notice that the peak images can be produced from \*.fit, \*.png and \*.bmp images. The peak image is displayed:



## Appendix

### Detailed results of Least Square Fit

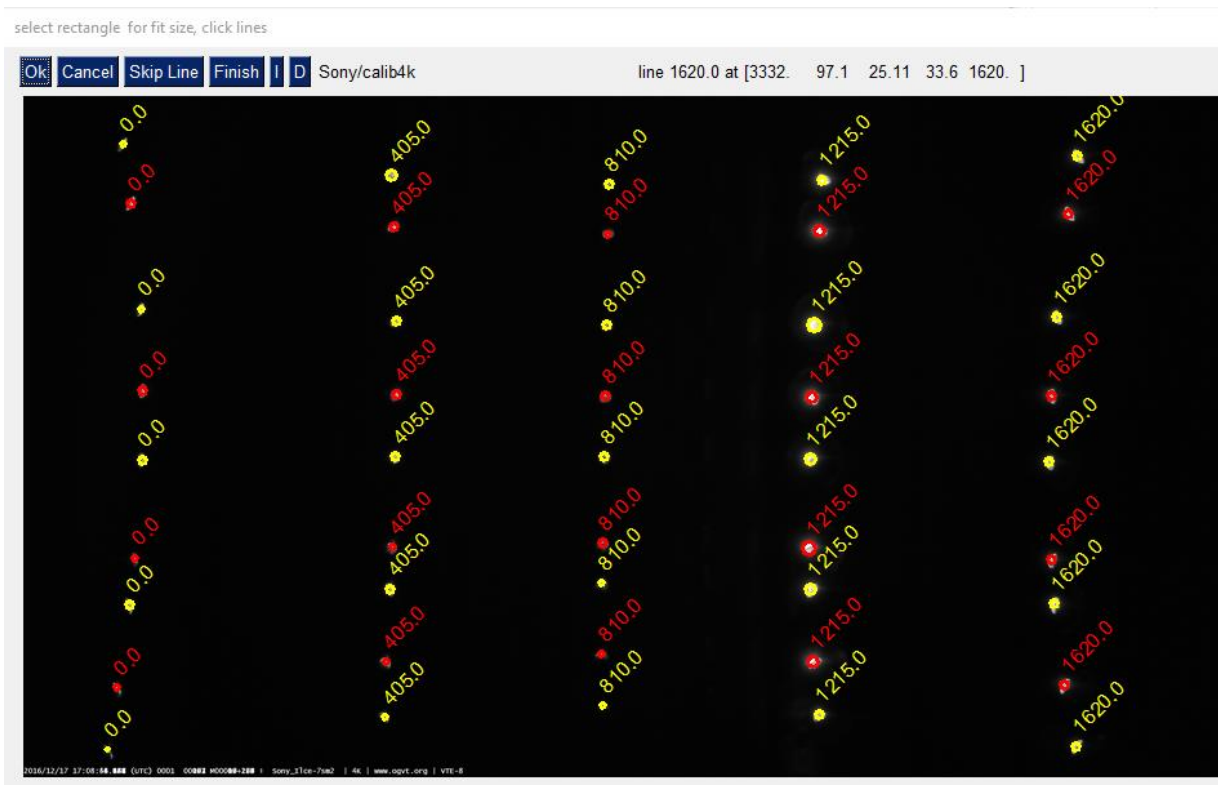
A detailed view of the results of the fit is given with the command “**report\_fit(out)**”, which shows statistical errors, correlations etc. in the shell window for the interested reader. Instead of changing the script as in a previous version, it can be switched on in the setup options:



For practical purposes the average rms error of the fit in x- and y-coordinates rms\_x and rms\_y are shown. With a well corrected and focused lens these values should be < 1 pixel. With noise free and aberration free synthetic data produced with ImageTools, rms-errors were < 0.05 pixel.

```
>>> report_fit(out)
[[Fit Statistics]]
# fitting method      = leastsq
# function evals      = 210
# data points         = 108
# variables            = 25
chi-square            = 10.1468068
reduced chi-square    = 0.12225068
Akaike info crit      = -205.416995
Bayesian info crit    = -138.363714
[[Variables]]
x0_0: 49.0181527 +/- 0.40095395 (0.82%) (init = 48.39159)
y0_0: 501.714124 +/- 6.76732160 (1.35%) (init = 515.1632)
...
scalxy: 0.85217823 +/- 0.03405730 (4.00%) (init = 0.92)
x00: 344.900732 +/- 1.73905549 (0.50%) (init = 345.386)
y00: 327.219815 +/- 3.81784835 (1.17%) (init = 322.3328)
rot: -0.00578993 +/- 3.1517e-04 (5.44%) (init = -0.006249105)
disp0: 1.99038480 +/- 0.00260282 (0.13%) (init = 1.987447)
a3: 2.9385e-07 +/- 3.1388e-08 (10.68%) (init = 2.402625e-07)
a5: -1.8900e-13 +/- 1.8562e-13 (98.21%) (init = 8.658912e-14)
[[Correlations]] (unreported correlations are < 0.100)
C(y0_7, y0_8) = 1.000
C(y0_0, y0_1) = 0.999
```

Notice, that the scalxy and the coefficient a5 are not well defined by the chosen data points (4 points in one spectrum cannot define the 5th power of a polynomial).



<pre> m_set.ini - Editor Datei Bearbeiten Format Ansicht ? [Lasercal] f_lam0 = 405.0 f_scalxy = 1.0 b_fitxy = 0 i_imx = 3840 i_imy = 2160 f_f0 = 24.0 f_pix = 0.00832 f_grat = 600.0 f_rotdeg = 0.0 i_binning = 2 s_comment = Sony alpha, f 24 mm s_infile = sony2k/calib2k s_outfil = sony2k/calib2k  [Calib] x00 = 953.3508684132236 y00 = 547.0873636640123 rot = -0.012352211369287523 disp0 = 1.2763356272969484 a3 = 1.8386668878523818e-07 a5 = 2.031462462324112e-13  Sony alpha, f 24 mm sony2k/calib2k.txt rms_x = 0.2861 rms_y = 0.4682 parameters after fit: ... \sony2k\m_set.ini saved </pre>	<pre> m_cal.ini - Editor Datei Bearbeiten Format Ansicht Hilfe [Lasercal] f_lam0 = 405.0 f_scalxy = 1.0 b_fitxy = 0 i_imx = 3840 i_imy = 2160 f_f0 = 24.0 f_pix = 0.00832 f_grat = 600.0 f_rotdeg = 0.0 i_binning = 1 s_comment = Sony alpha, f 24 mm s_infile = Sony/calib4k s_outfil = Sony/calib4k s_linelist = 1 b_sqrt = 0  [Calib] scalxy = 1.0 x00 = 1908.5919 y00 = 1099.8146 rot = -0.0051855845 disp0 = 0.6342405 a3 = 5.0031932e-08 a5 = 1.3980166e-14  Sony alpha, f 24 mm Sony/calib4k.txt rms_x = 1.5168 rms_y = 1.8173 parameters after fit: ... Save config in directory Sony\Sony4k.ini </pre>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

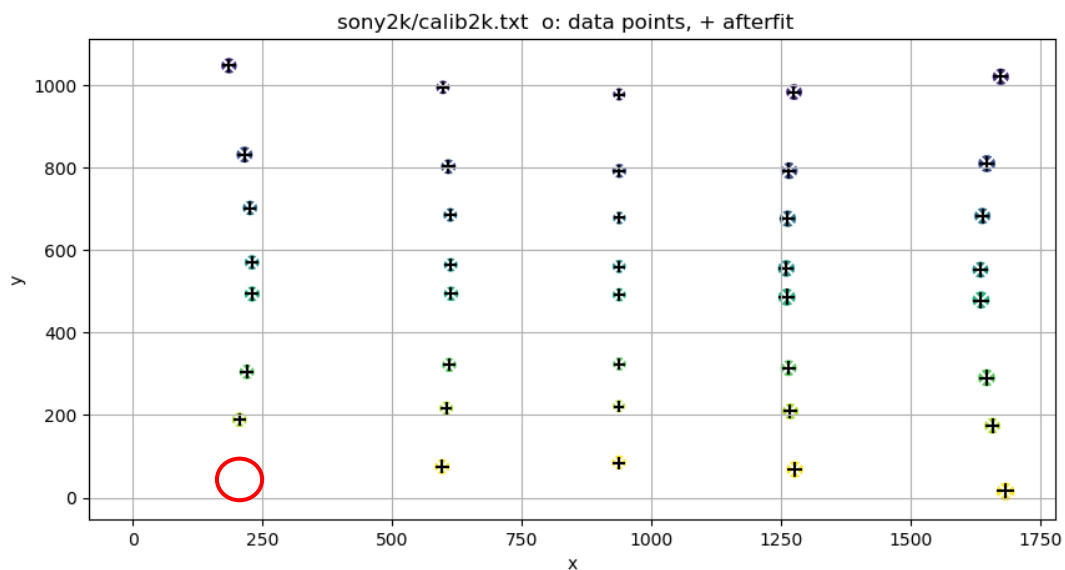
Note:

The parameter names in the configuration are preceded by a letter indicating the type of the variable. This simplifies the conversion. `f_` is for floating point parameters, `i_` for integers, `s_` for strings. `b_` is for boolean variables, **True** replaced by 1 and **False** by 0.

After finishing a calculation, the relevant parameters are stored in the file `m_cal.ini`. It corresponds to a Windows-style \*.INI and contains the input parameters under the heading [lasercal] and the results of the least square fit under the header [Calib]. The input and output directories you can change during the running of the program `l_calib.py`. Other changes have to be done manually by editing the file `m_set.ini` with a text editor such as Notepad. Next time you run `l_calib6.py` the new values will be used.

### Binning, directory structure

If you have different cameras, lenses, image formats you may need more than one configuration. In this case you can use different subdirectories or different ini-filenames for each case, in this example there is a directory “sony4k” for 4k images and a directory “sony2k” for binned 2k images. The only difference in the configuration is the `binning = 2` for the 2k images, as compared to `binning = 1` for the 4k images. Of course the results for [Calib] are different for the different binning factors; the values of `x00` and `y00` are 2 times larger for the unbinned images, the values of `a3` and `a5` 4x respectively 16x smaller for the unbinned images. Notice also that the errors are much smaller in the 2k images. This is because the first line in the lowest laser spectrum was omitted in the 2k analysis (this is permitted for the first or the last order, but you may not skip intermediate orders), since it overlapped partly with the image caption:



The results are included in the configuration file, because they will be used in the analysis of the meteor spectra, for the calculation of the distortion to transform to orthographic projection. This will be executed in the script `m_pipe`.

## Conclusion

The program `m_calib.py` allows calibration of meteor spectra with the same results as by the earlier EXCEL method. It has the following advantages:

- It runs on different platforms
- It is free (no license needed)
- It is much faster than EXCEL (about a factor 10 and less critical dependent on start values)
- It does not use language dependent macros, which made the EXCEL version unusable in different countries (Japan, Ukraine, China)
- It allows the treatment of meteor spectra without different software (IRIS, ImageTools, and ISIS), speeding up the treatment of meteor spectra.

## Important note

**The present versions are preliminary, only tested by myself. They probably contain many bugs. Comments about bugs or improvements are highly appreciated.**

## Content

Summary .....	1
History.....	1
Installation of Python.....	2
Laser calibration of meteor spectra .....	2
Run the test example .....	3
Setup.....	5
Select configuration.....	6
Select points of the different orders .....	7
Inspection of the calibration text file.....	9
Least square fit.....	9
Run the fit for the example <code>caladd9.txt</code> with <code>m_calib.py</code> .....	10
Use of spectrum lamp for calibration.....	12
Line list.....	12
Run the script for spectral lamp calibration .....	13
Use of meteor spectra for calibration .....	15
Analysis of spectrum <code>M20171209_041224_JPMZ1</code> .....	16
Creating calibration images from video files .....	18
Making peak images from single calibration images.....	18
Appendix.....	20
Detailed results of Least Square Fit.....	20
Binning, directory structure .....	22
Conclusion .....	22
Important note .....	23