# PlanetarySystemStacker for High-Resolution Imaging of Planetary System Objects through Turbulent Air



## 

## User Guide (Version 0.5.0, May 19, 2019)

Rolf Hempel

#### *Table of Contents*

[1. Introduction 2](#_Toc510165760)

[2. Changelog 3](#_Toc510165761)

[2.1 Changes since version 0.9.5 (October 2017) 3](#_Toc510165764)

[2.2 Changes since version 0.9.3 (October 2016) 4](#_Toc510165765)

[3. System Requirements and Software Installation 4](#_Toc510165766)

[3.1 Windows (7 / 8 / 10) 4](#_Toc510165770)

[3.2 Linux 6](#_Toc510165771)

[3.3 Connection to FireCapture 7](#_Toc510165772)

[4. Program Execution 8](#_Toc510165773)

[4.1 Program Start 8](#_Toc510165778)

[4.2 Layout of the Main Window 10](#_Toc510165779)

[4.3 Selecting a Landmark on the Moon for Mount Alignment 10](#_Toc510165780)

[4.4 Camera Rotation 12](#_Toc510165781)

[4.5 Selection and Positioning of the Focus Area 13](#_Toc510165782)

[4.6 Image Acquisition 13](#_Toc510165783)

[4.7 Determining and Correcting the Coordinate System Drift 16](#_Toc510165784)

[4.8 Automatic Alignment 18](#_Toc510165785)

[4.9 End of Program 20](#_Toc510165786)

[Appendix A: Parameters at the Configuration Dialog 21](#_Toc510165787)

[Appendix B: Algorithms Used by the Program 28](#_Toc510165788)

[Appendix C: Determination of the Focal Length of the System 35](#_Toc510165789)

## Introduction

PlanetarySystemStacker (PSS) processes a large number of video frames or still images of a planetary system object, taken in rapid succession, into a high-resolution image. It selects the sharpest sections of the best frames, removes any image warping caused by atmospheric turbulence, and combines the resulting snippets into a single image with maximum detail. In a second step, PSS offers the option to remove image blur by applying a multi-level sharpening filter.

With the introduction of digital image technology the photography of planetary system objects (the moon, sun and planets) has taken a great leap forward. The so-called [“Lucky Imaging”](https://en.wikipedia.org/wiki/Lucky_imaging) technique today allows amateur astronomers to record surface details which in the old days of analog photography were out of reach even to professional astronomers with access to the best earth-bound telescopes. For high-resolution work today the camera of choice is a video module which connects to the USB interface (preferably 3.0 or higher, allowing high data transfer speed) of a portable computer. With such a camera one can take many images in a short time and store them without compression artefacts in a [RAW image](https://en.wikipedia.org/wiki/Raw_image_format) sequence, using typically a SER or AVI video file container. On the negative side, however, the image sensors are quite small and exhibit relatively low pixel counts.

As an alternative, digital cameras (such as a DSLR) can be used to capture many images of the object in rapid succession. These images are stored as single files with identical pixel counts in a folder. Input to PSS can be either a video file or a directory containing still images of the same scene.

PlanetarySystemStacker is the first open-source software of its kind. The complete [source code](https://github.com/Rolf-Hempel/PlanetarySystemStacker/tree/prototype/Source) is published on Github, together with a detailed [documentation of the mathematical algorithms](https://github.com/Rolf-Hempel/PlanetarySystemStacker/blob/prototype/Documentation/algorithm_summary.pdf) used.

## Changelog



### Initial release 0.5.0 (May 2019)

* The Windows installer was built on a Windows 10 system. The software and the installation process were tested on Windows 10 only.

## System Requirements and Software Installation

PlanetarySystemStacker in principle runs on any computer where a Python 3.5 environment and the required Python libraries are available. For the time being, however, an automatic installer is available for Windows only. For a future release it is planned to provide a Linux installer as well.



### Windows (7 / 8 / 10)

The whole software was tested on an Acer laptop computer (type “Acer Aspire V5-573G”, Intel Core i5-4200U, 12 GBytes RAM), and on a PC (Intel Core i7-7700K, 64GBytes RAM), both running Windows 10 Professional, version 1803. Video stacking very much benefits from large RAM. A minimum of 16Gbytes of RAM is recommended. For large video files processed on a system with less RAM it is unlikely that all data can be held in memory, so re-reading and re-computing data will slow down execution.

The PlanetarySystemStacker software is distributed as a single file: the Windows installer [“PlanetarySystemStacker\_V0.5.0\_Windows-Installer.exe”](https://github.com/Rolf-Hempel/PlanetarySystemStacker/releases). Before installation, any earlier PlanetarySystemStacker version should be de-installed first using the “Uninstall” entry in the program menu. When the installer is started, a wizard guides the user through the installation process. Apart from the program start entries, PlanetarySystemStacker does not write any parameters into the Windows Registry and can be installed at any file system location. An uninstaller is provided with the software. It removes all installed files. The following temporary files which are created at execution time in the user’s home directory are not removed:

* .PlanetarySystemStacker.ini (configuration during the latest PlanetarySystemStacker run)
* PlanetarySystemStacker.log (logfile)

Those files can be deleted manually at any time. Please note that all configuration parameters are lost if the .PlanetarySystemStacker.ini file is deleted.

The Windows installation wizard offers to place a program starter on the desktop. It is decorated with the PlanetarySystemStacker icon automatically, which can be found under the filename “PlanetarySystemStacker.ico” in the installation folder.

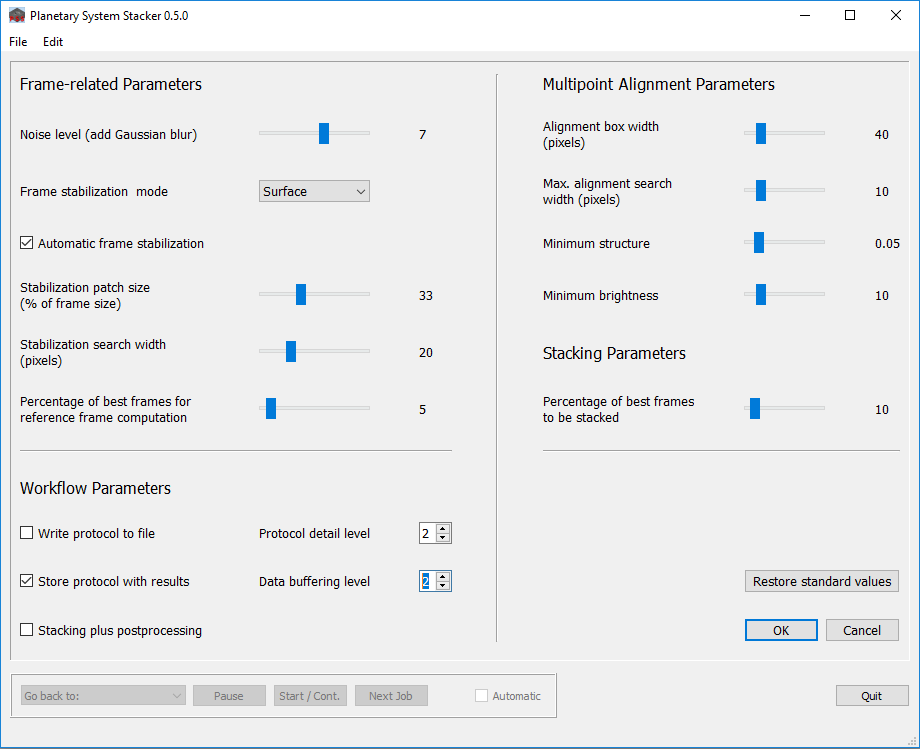
## Program Execution

PlanetarySystemStacker communicates with the user via a graphical user interface (GUI). Good usability was a high-priority design criterion. In particular, at any time the GUI presents to the user only the information which is of current relevance. The user interface was developed using the QT5 toolkit.



### Program Start / Setting Parameters

When the program is started for the first time, a view opens automatically for the input of individual parameters. The parameters are arranged in groups relating to frame stabilization, multi-point alignment and workflow control.



Predefined parameter values should work okay in most cases, but the user is encouraged to experiment with different settings. If the mouse pointer hovers over a parameter name, a tooltip appears describing its meaning. A detailed explanation of all parameters can be found in “Appendix A: Parameters at the Configuration Dialog”.

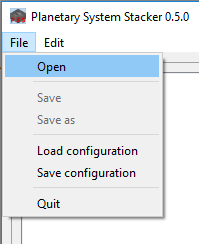
The pre-defined parameter values can be restored at any time by pressing the button “Restore standard values”.

On leaving the parameter dialog, and on termination, PSS writes the configuration file “PlanetarySystemStacker.ini” into the user’s home directory. At later program invocations the parameters are loaded from this file silently, the input dialog does not open, and PSS is ready for job input. Parameters can still be changed via the “Edit / Edit configuration” menu.

Additionally to the standard configuration file in the home directory, the current parameter configuration can be saved to any file system location via the “File / Save configuration” menu entry. Later on, this configuration can be loaded via “File / Load configuration” to replace the current parameter configuration. Please note that the configuration not only comprises the parameters shown at the dialog above, but also all active postprocessing variants.

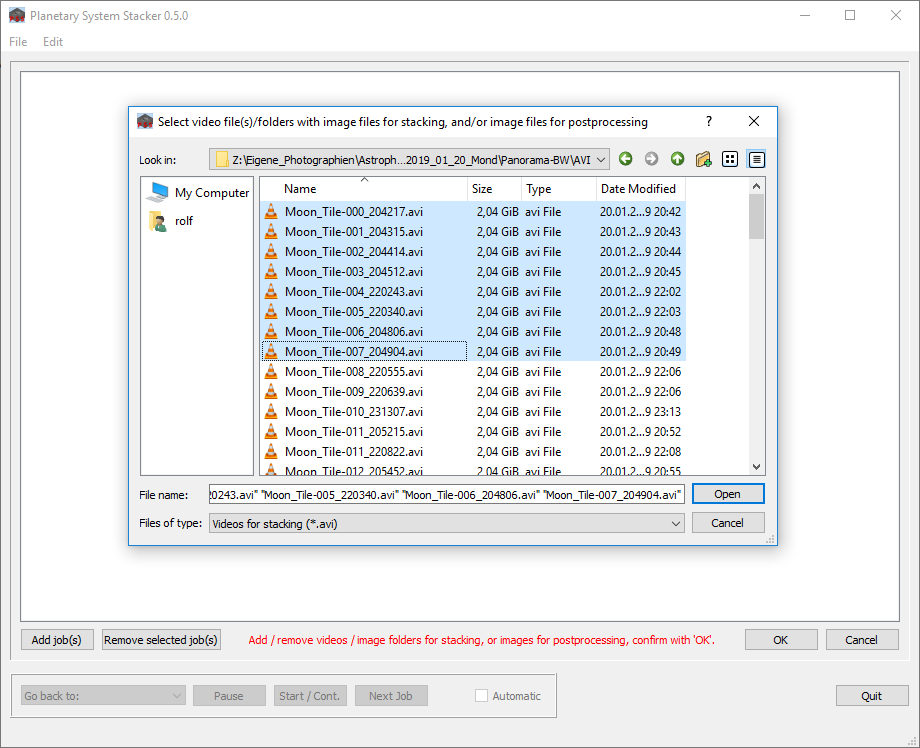
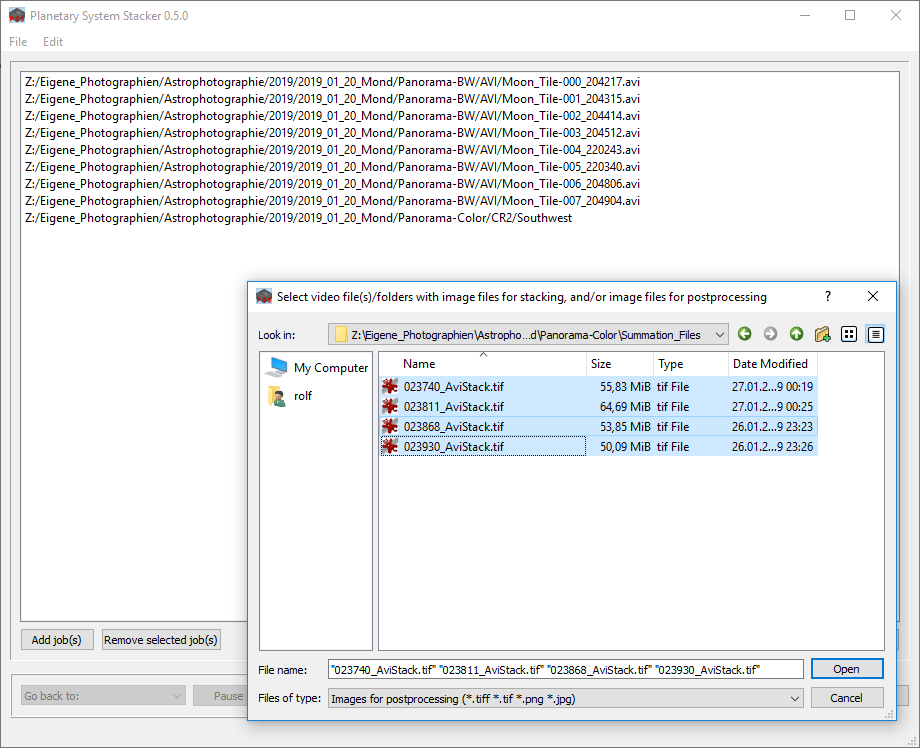
### Job Specification

The user is instructed to open the “File / Open” dialog, and to define the jobs to be processed. Throughout the program, instructions like this are displayed in red print in the status line at the bottom of the main GUI, or within a dialog window.

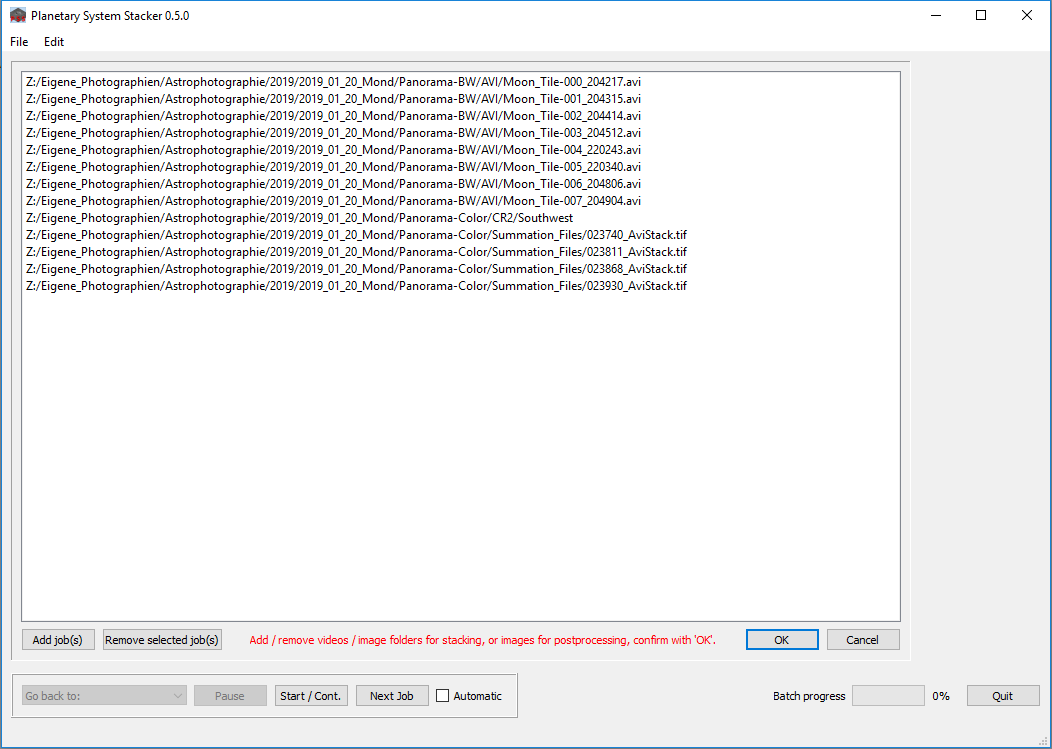


Upon choosing “File / Open”, a window appears where jobs can be specified. At this point it is important to understand how PSS processes jobs.

Basically, the user specifies a list of jobs which PSS then processes one after the other. There are two different job types which PSS recognizes by the kind of input specified:

* To define a stacking job, the user selects a video file (extension “.avi”) or a directory containing still image files with identical pixel dimensions. At the bottom of the file dialog window “Files of type” must be set to “Still image folders / video files for stacking (\*.avi)”. Selecting a directory with image files which do not match will result in an error message later on.
* To define a postprocessing job, the user selects a single image file (with the extension .tiff, .tif, .png, or .jpg). At the bottom of the file dialog window “Files of type” must be set to “Images for postprocessing (\*.tiff \*.tif \*.png \*.jpg)”

When the user closes the dialog window by pressing “Open”, the selected jobs are added to the list of jobs presented in the main window. This list may contain stacking and postprocessing jobs in any order. Additional jobs can be defined by pressing “Add job(s)” to re-open the job selection dialog again. Jobs can be removed from the list by clicking on them and pressing “Remove selected job(s)”.



When all jobs are defined, press “OK” to finalize the job specification.

### Starting and Controlling the Workflow

Pressing “Start / Continue” triggers PlanetarySysstemStacker to start processing the jobs in sequential order, either interactively or in batch mode (fully automatic). By checking / unchecking the “Automatic” checkbox at the bottom of the main GUI, the user can switch back and forth between the two modes at any time during execution.

If a large number of similar video files are to be processed, or many images of the same moon panorama are to be postprocessed, the recommended procedure is to process the first job in interactive mode (“Automatic” unchecked). PSS then stops at every processing step and prompts the user to adapt parameters as he or she likes. After the first job the user checks the “Automatic” box, and PSS continues processing all the remaining jobs automatically using the adapted set of parameters.

When batch processing is active, the user at any time can seize control by unchecking the “Automatic” box. PSS just finishes the current processing step and switches back to interactive mode.



During computational intensive processing phases PSS displays a progress bar at the bottom of the GUI. If more than one job is being processed, a second progress bar shows the fraction of finished jobs. The status line displays information about the files being processed and the current processing step.

### Execution Protocol

Being able to look at the execution details of a job can be very useful, especially if something went wrong in batch mode, of if the resulting image is of lower quality than expected. With PSS the user can choose among various protocol variants.

First of all, the parameter “Protocol detail level” selects the amount of information provided. Level “0” means no output at all. At level “2” detailed info is provided, e.g. on the number of alignment points and the warp distribution. If parameter “Write protocol to file” is checked, all protocol data is appended to the standard file “PlanetarySytsemStacker.log” in the user’s home directory.

In particular if many jobs are processed, it is recommended to set the option “Store protocol with results”. In this case, additional to the sequential protocol file the part pertaining to a given job is written as a separate file. Its name is derived from the name of the job result, with the stacking suffix “\_pss.tiff” being replaced with “\_stacking-log.txt”. This way it is easy to associate the log file with the corresponding job output. In the case of postprocessing jobs, the ending “\_postproc-log.txt” corresponds to the postprocessing result “\_gpp.tiff”.

### Reading Input Data and Buffering

Input data for stacking can be large. In the worst case, the input video file itself is larger than the available RAM space. To make things worse, additional to the original video frames PSS uses several image variants which result from applying certain filters (e.g. Gaussian or Laplacian filters). Since those variants in general are used more than once, processing is fastest if the data can be kept in memory. If the RAM is too small, however, this is not an option. Unfortunately, relying on the paging mechanism of the operating system isn’t an option, either, since this tends to slow down execution way too much.

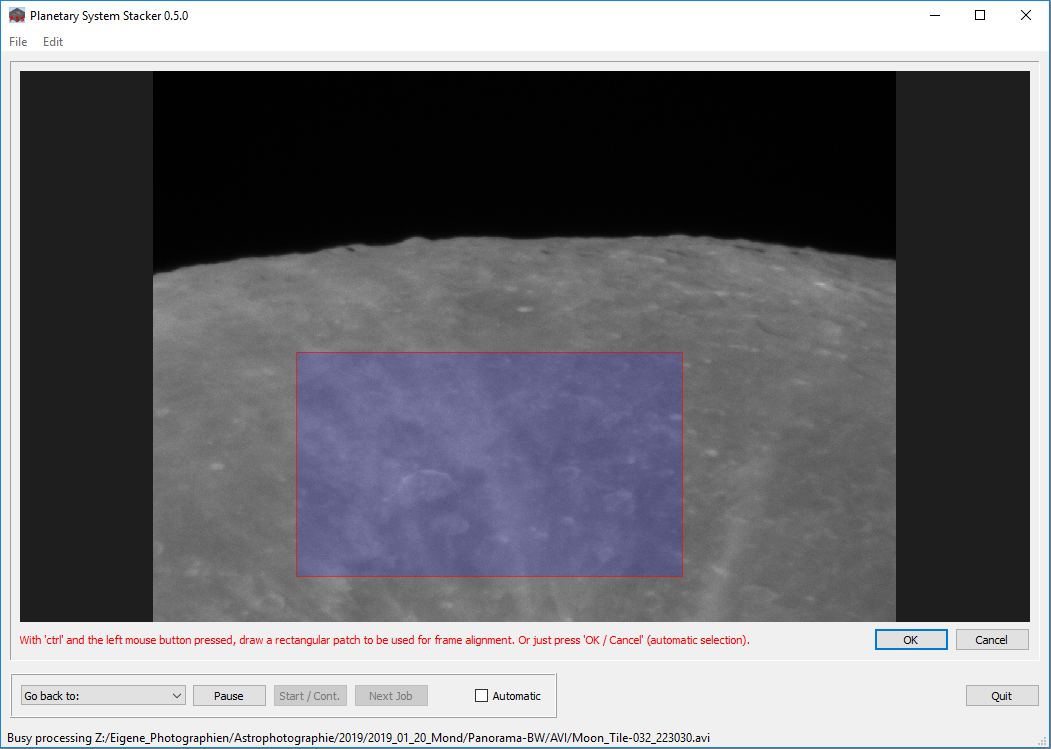
PSS, therefore, offers a range of buffering levels. Which one is best depends on the size of input data and the size of the available RAM. The user can choose the most appropriate scheme in the configuration dialog (parameter “Data buffering level”). At level “0”, data is not buffered at all. The original frames are read three times during job execution, and derived images are recomputed when needed. Obviously, this mode leads to maximum computational load. At the other end of the scale, at buffering level “4” all data is kept in memory until a job is finished. On computers with enough RAM this is the best choice. In general, the default setting “2” is a good compromise between speed and RAM usage.

### Frame stabilization

Before frames are stacked they first have to be roughly aligned with each other. This way, drift effects of poor mount alignment or guiding inaccuracies are removed. PSS offers two stabilization modes:

* In “Planetary” mode, it is assumed that a planet is located somewhere in the frame, completely surrounded by dark sky. In this case, frame stabilization is very simple and reliable. Therefore, it is strongly recommended to use this option whenever appropriate.
* In all other cases, “Surface” must be used. This is usually the case for moon or sun imaging. In Surface mode, a so-called “stabilization anchor” must be selected in the image. By comparing its position in all frames, PSS determines the drift between them. Obviously, it is crucial to choose the anchor appropriately.

PSS offers to identify the stabilization anchor automatically (check “Automatic frame stabilization”). If this option is chosen, additional parameter s (“patch size” and “search width”) can be modified to control the automatic algorithm. Obviously, in batch mode automatic stabilization is the only choice, so the configuration parameter “Automatic frame stabilization” is ignored.



If the user chooses to define the stabilization anchor manually in interactive mode, first the parameter “Automatic frame stabilization” needs to be unchecked. PSS interrupts the workflow, shows the full scene covered by the video to the user and asks to draw the stabilization anchor as a rectangular patch. Since this is the first time in the workflow where the frame viewer appears, this is the time to explain its general features:

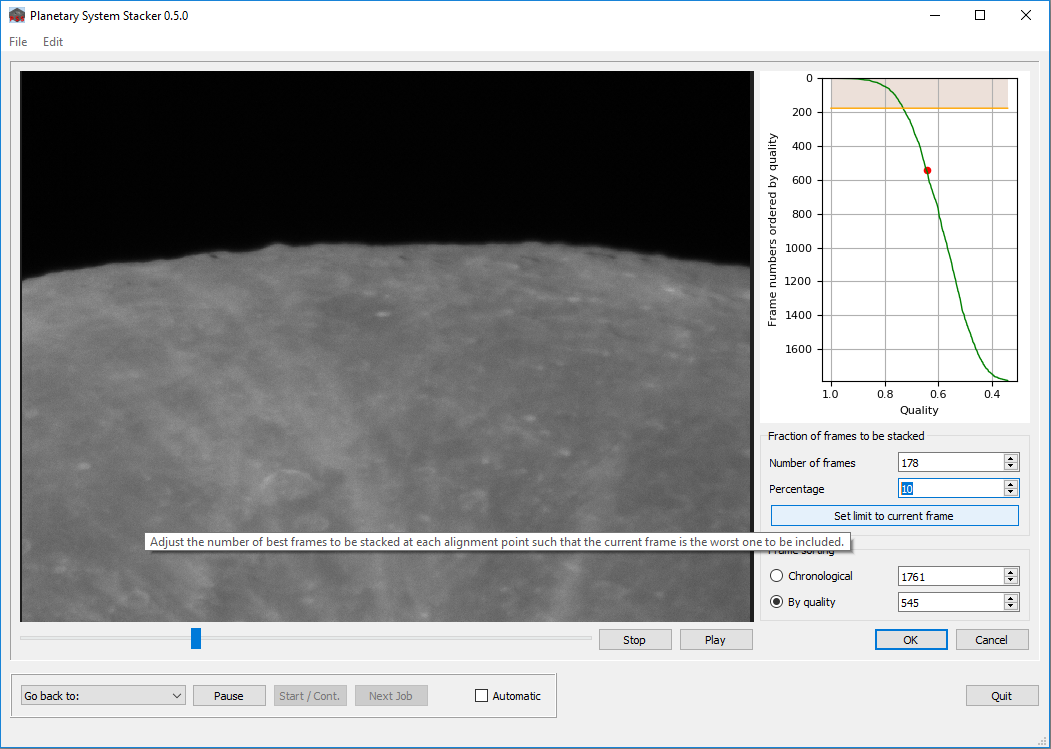
* At the beginning the viewer shows the full frame. Using the scroll wheel (or the keys “+” and “-“), the user can zoom in and out. Moving (panning) the scene laterally is accomplished by moving the mouse while keeping the left mouse button pressed.
* If the contents of the view are to be manipulated, like in this case by drawing the rectangular patch, the “ctrl” key has to be kept pressed during the manipulation. As long as the key is pressed, the hand symbol at the mouse pointer location changes into an arrow.
* Drawing a rectangular patch is done by pressing a mouse button (in this case the left one) at one corner of the patch and moving the mouse to the opposite corner.
* Other manipulation options depend on the use case and are described there.

For the stabilization anchor an area with as much small-scale contrast (both vertically and horizontally) is to be preferred. It should be large enough. Usually some 30% of the frame size (per coordinate direction) is a good choice.

Once the patch is selected, the user acknowledges the choice by pressing “OK”. By pressing “Cancel” instead, PSS is instructed to define the stabilization anchor automatically. This also happens if the patch selected by the user is too small.

When the frame stabilization process is completed, PSS computes the image area common to all frames. This area is the basis for all processing steps from this point on.

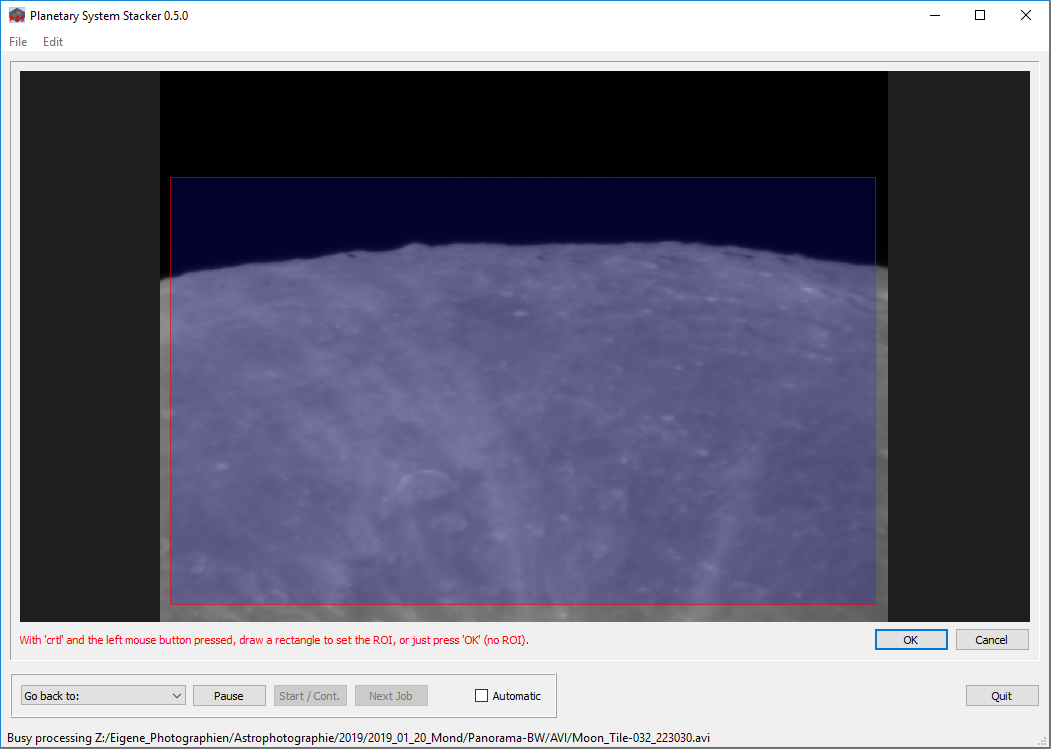
### Setting the Stacking Fraction

In the next step the number of frames to be used for stacking is to be set. The GUI opens a view on the scene where frames can be displayed either ordered by decreasing overall quality, or chronologically. A horizontal slider can be used to scroll through all frames. As an alternative, consecutive frames can be displayed as a video by means or the buttons “Play” and “Stop”.

On the right-hand side, a graph shows the position of the current frame in the video (vertical axis) and its quality (horizontal axis). The shaded area shows the fraction of frames to be stacked. The stacking limit can be moved by changing the input editor fields for “Number of frames” or “Percentage”. As an alternative, pressing the key “Set limit to current frame” moves the border of the shaded area to the position of the frame being displayed.

The selection of the stacking limit is acknowledged by pressing “OK”.

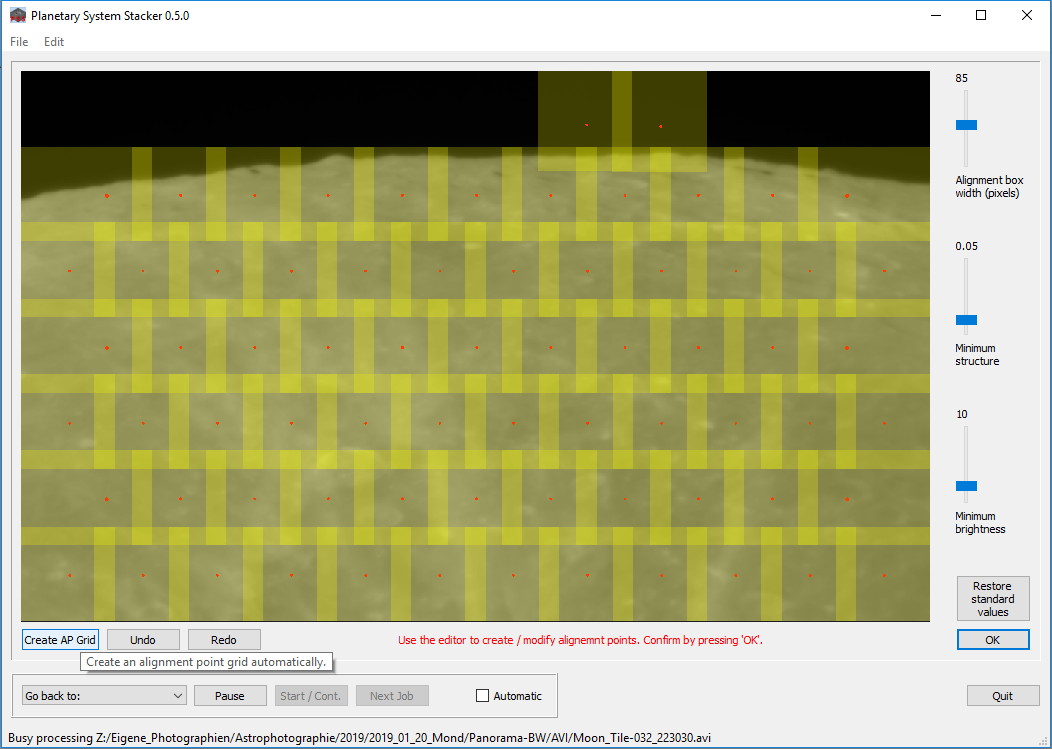
### Setting a Region of Interest (ROI)



Optionally, a so-called region of interest smaller than the intersection of all frames can be specified. To this end a viewer opens and prompts the user to select the ROI as a rectangular patch (as described in Section 4.6 above). If the full view is to be stacked (no ROI selected), just press “OK”.

### Selecting Alignment Points

In the next view the alignment points (APs) used to de-warp the individual frames are selected. Again, a viewer opens, along with controls to the right and below.

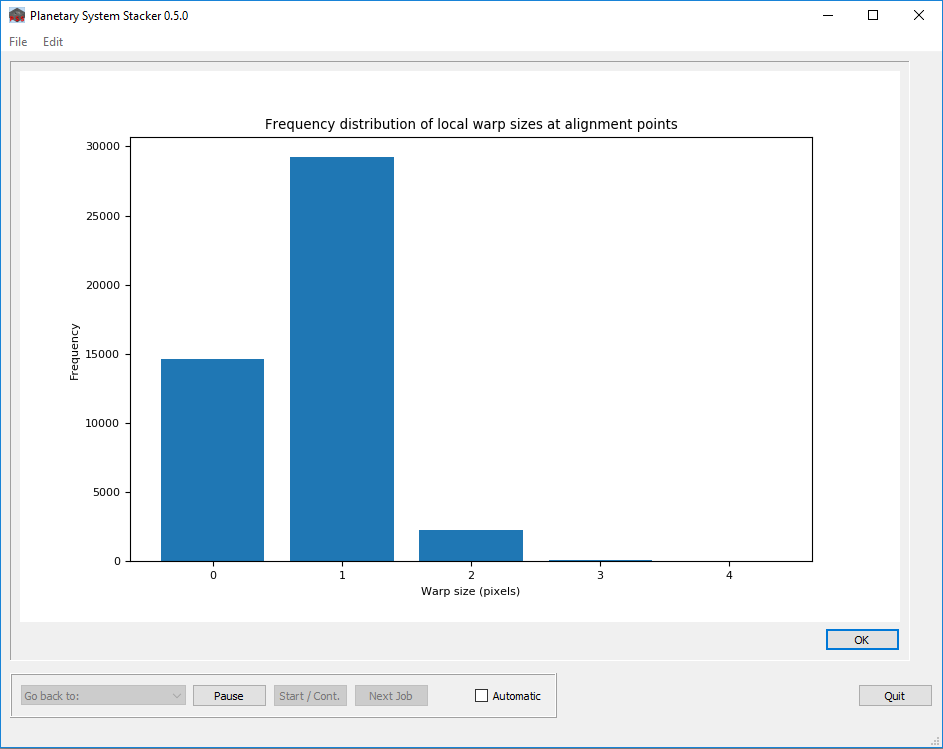


Initially, the view just shows the complete scene (if a ROI was selected, the view is restricted to this area). Alignment points can be generated automatically or set individually by the user:

* To create APs automatically, just press the button “Create AP Grid”. The size of individual AP patches can be controlled by changing the “Alignment box width” slider on the right. Thresholds for excluding areas which are too dim (e.g. sky background) or contain too little structure can be changed by moving the two other sliders.
* After changing the slider settings, press “Create AP Grid” again to compute a new AP grid.
* APs can be deleted, added and modified manually. This way the user can adapt the grid as required. Remember that all these manipulations are done while keeping the “ctrl” key pressed.
  + To delete an AP, place the mouse pointer close to the AP center (red dot), and press the right mouse button.
  + To delete a whole AP region, open a rectangular patch by moving the mouse while keeping the right mouse button pressed. When the button is released, all APs in the patch are removed.
  + To move an AP, press the left mouse button at an AP and drag the AP with the mouse to the desired location.
  + To change the size of an AP patch, move the mouse close to the AP center and use the scroll wheel to change its size.
  + To add an AP, left click on the desired location. An AP with (initially) the standard size (see slider on the right) is created there.
* All those AP operations can be undone / redone by pressing the buttons “Undo” or “Redo”. The size of the undo stack is unlimited.
* If the sliders have been set to different values, they can be reset to default values by pressing the button “Restore standard values”.

### Frame Stacking

When the AP selection is completed, PSS has gathered all the information it needs to stack the frames. First, at every AP it identifies the sharpest frames to be used for stacking. Since the seeing is a very local phenomenon, frame sets will be different for different APs. Then, for every AP and every contributing frame the local displacement relative to a reference frame is measured, and the shifted AP patch added to the stacking buffer. Progress bars are updated regularly throughout the process.



Once the stacking is finished, a graph shows the frequency distribution of local displacements at all APs and all contributing frames. Usually, small displacements (a few pixels) occur most often. If the distribution extends too much towards larger numbers, this could be due to a low stacking quality. In this case it is recommended to experiment with different parameters, for example:

* Increase the “Noise level (Gaussian blur)” parameter.
* Increase the AP size.
* Eliminate APs in areas with too little structure.

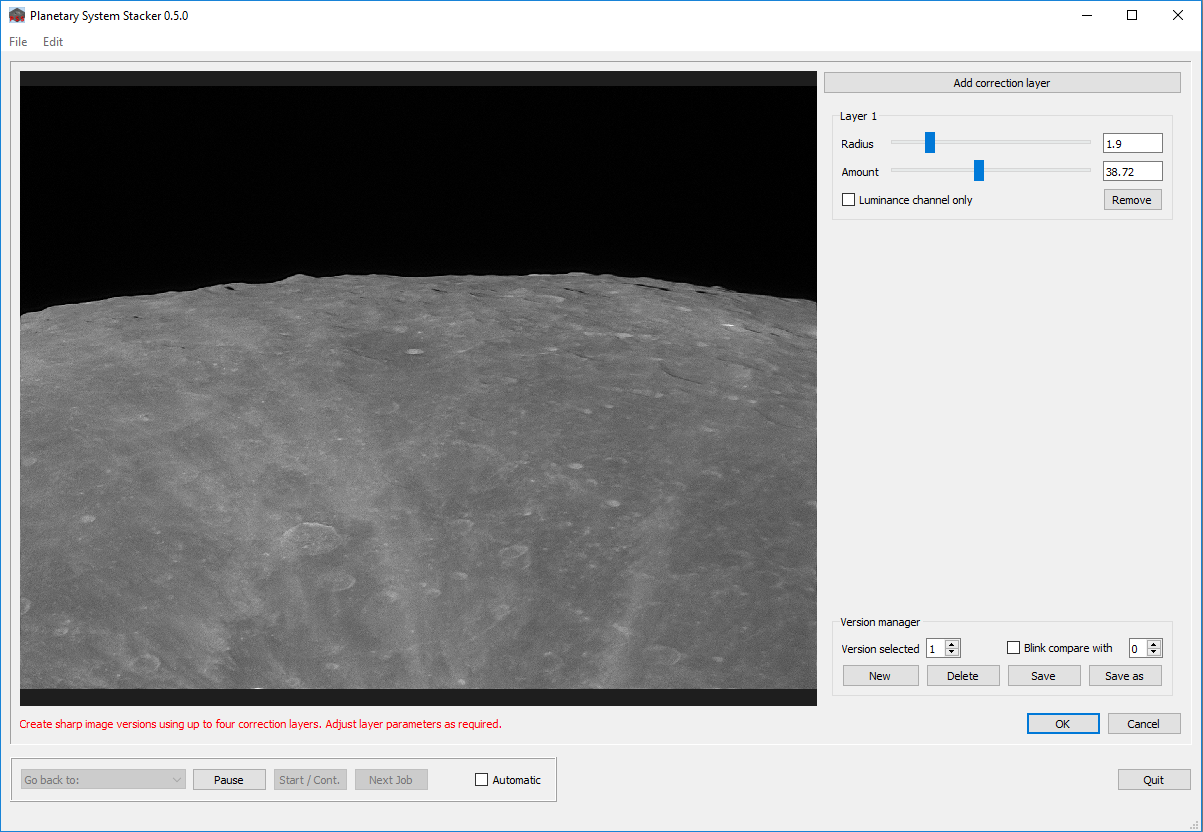
After inspecting the warp distribution, press “OK” to finish the current job. The stacked summation image is written in 16bit Tiff format to the file system into the same folder where the input is located. The name is copied from the input video or image directory, extended by the suffix “\_pss.tiff”.

### Postprocessing (Image Sharpening)

As explained in Section 4.2, PSS can sharpen a still image in a postprocessing job. Input to this type of job is a single image file, and tailoring and applying the sharpening filter is the only job activity.

As an alternative, postprocessing of a stacked summation image can be included as the last step in a stacking job. In this case, the configuration parameter “Stacking plus postprocessing” must be checked before the end of the stacking job.

This section describes how postprocessing works in PSS. As explained above for stacking, postprocessing can be performed either interactively, or in fully automatic batch mode. In the case of postprocessing a bunch of similar images, executing the first job interactively and then continuing in batch mode (by setting the “Automatic” checkbox) is recommended as best practice. The reason is that for sharpening no reasonable default values can be specified, so finding a good set of sharpening parameters is possible only in an interactive trial and error loop. PSS was designed to support this interactive process as effectively as possible.



To support experimentation with different sharpening configurations, PSS offers to define and compare an arbitrary number of model versions. With the “version manager” at lower right new versions can be created. With the “Version selected” spin box another version can be selected. Other GUI elements in this view, such as “Delete”, “Save”, “Save as”, and the model details in the panel at upper right, refer to the version currently selected. Version number “0” is reserved for the original image, with no correction being applied.

Sharpening in PSS is implemented using a multi-layer unsharp masking algorithm. A hierarchy of up to four layers can be defined. Each layer is defined by

* its spatial frequency (in pixels),
* the strength (or Amount) with which the layer is applied, and
* whether the sharpening should only affect the luminance channel or all color channels.

When the postprocessing view opens in the GUI, the viewer to the left shows the input image, with the sharpening model selected in the version manager being applied. If PSS is called for the first time, additional to the original image (Version “0”) it starts with a version 1, consisting of a trivial single correction layer (Radius = 1.0, Amount = 0., i.e. no correction is being applied, sharpening in all color channels) being defined and selected.

Layers are added / removed by pressing the buttons “Add correction layer” and “Remove”, respectively. When layer parameters are changed, the image viewer is updated immediately. In the case of large images, a “busy” message appears in the status line until the update is finished.

Different versions can be compared with each other using the “blink comparator”. To this end, additionally to the selected version, another version is chosen in the spin box to the right. When both versions are set, checking the “Blink compare with” box causes the viewer to alternate between them. While the blink comparator is running, the layer parameters of the selected version can be changed. Layers can be added or removed. Even the numbers of the two versions can be changed. Last but not least, as with all PSS viewers, zooming and panning can be used to inspect the effect of the sharpening model in detail.

Once the user is satisfied with the selected version, pressing the “OK” button triggers PSS to save this image version along with the input image (as a 16bit Tiff image). The file name is the one of the input image, extended by the suffix “\_gpp.tiff”. The parameters of all postprocessing versions and the number of the selected version are saved in the configuration file. When the postprocessing view opens next time, all versions and the selection index are restored.

Different postprocessing configurations can be saved and restored at any time using the menu entries “File / Save configuration” and “File / Load configuration” (see above in Section 4.1)..

### End of Program

When a job is finished in interactive mode, the user can instruct PSS to repeat execution starting with a selected job phase. Pressing the combo box “Go back to” opens a panel with the available phases. This can be useful if one wants to change parameters affecting later job phases only, without repeating the whole job.

When all jobs are done, PSS returns to its initial state. New jobs can be selected in the “File / Open” dialog. If no more jobs are left, the user can quit the program by pressing the “Quit” button. As a last activity PSS saves the current configuration in the standard configuration file and closes all protocol files.

## Appendix A: Parameters at the Configuration Dialog

The main window of the configuration dialog allows entering user-specific values for various parameters. They are arranged in the following groups:

* Geographic Position
* Camera
* Telescope
* Workflow
* Tile Visualization
* Alignment



In the following, definitions for all parameters are given. Please note that for fractional values a decimal point (no comma) is used.

#### Geographical Position

|  |  |
| --- | --- |
| **Longitude:** | Geographic longitude (in degrees and fractional part), counted from the Greenwich meridian, positive to the East. |
| **Latitude:** | Geographic latitude (in degrees and fractional part). Northern latitude positive. |
| **Elevation:** | Elevation above sea level (in meters, no decimal point). |
| **Timezone:** | Drop-down list. Choose the time zone corresponding to the geographical location. |

#### Camera

|  |  |
| --- | --- |
| **Brand / Name:** | Drop-down list of all camera models for which specifications have been entered. With the buttons “Edit”, “New” and “Del” the user can modify parameters for existing models, add specifications for new models, or delete models from the list. |
| **IP address to access FireCapture:** | If camera automation is switched on, MoonPanoramaMaker triggers the FireCapture program for video recording. The two programs communicate via TCP/IP and may run on different computers. The IP address of the system where FireCapture is located can either be “localhost”, i.e. both programs run on the same system, or a fully qualified IP address, such as “192.168.0.1”. Make sure that port 9820 is not blocked by a firewall. |
|  | The sub-dialog for entering new camera models or for editing existing ones contains the following additional parameters:  D:\Python\MoonPanoramaMaker\Documentation\Screenshots_1-0-0\1-0-0_Configuration-dialog-camera.PNG |
| **Pixel size:** | Horizontal or vertical distance (“pitch”) of two adjacent pixels of the camera sensor. In general this is not the size of the light-sensitive cell itself. It is assumed that the distance is the same in both directions. Please note that the value is given in millimeters, not in microns. |
| **Pixel count horizontal / vertical:** | Number of sensor pixels in horizontal and vertical direction, respectively. |
| **Repetition count** | Number of videos to be taken in succession at each tile location. This parameter can be used, for example, to take three videos of the same area through RGB filters. FireCapture can be configured such that it cycles through the different filters. Please note that setting the repetition count in MoonPanoramaMaker does not change the FireCapture settings. This is left to the user. |
| **External margin pixels:** | Minimal width of the area surrounding the Moon to be recorded (in pixels). If this value is chosen too small, sunlit mountain peaks beyond the terminator could lie outside the recorded area. Also, less than perfect mount alignment could cause marginal areas of the Moon to be cut off. Too large a value unnecessarily increases the number of tiles to be recorded. |
| **Tile overlap pixels:** | Minimal overlap between neighboring tiles in pixels. Again, experience is needed to strike the right balance between safety and too much overhead. |

#### Telescope

|  |  |
| --- | --- |
| **Focal length:** | Focal length of the complete optical system in millimeters, including any projection system between objective lens and sensor. If such a projection lens is used, the resulting focal length of the overall system is difficult to predict. In “Appendix C: Determination of the Focal Length of the System” it is shown how the focal length can be determined experimentally.  Please note that the precision of the focal length entered is critical if auto-alignment is used. If the error is too large, auto-alignment initialization will fail. Also, the translation of measured image shifts into equatorial coordinate corrections will be less accurate. |
| **Telescope mount interface:** | Drop-down list. Choose the appropriate interface available on your computer. ASCOM must be selected if MoonPanoramaMaker runs on a Windows computer, INDI if it runs on Linux. Make sure that the required ASCOM or INDI software is installed (see Section 3.  Press the “Configure” button to enter specific parameters for the mount driver used. The sub-dialog which opens is different for ASCOM and INDI: |
|  | **Case “ASCOM”:**  The “Configure” button opens the following sub-dialog:  D:\Python\MoonPanoramaMaker\Documentation\Screenshots_1-0-0\1-0-0_Configuration-dialog-ASCOM.PNG |
| **Select and configure the ASCOM telescope driver:** | Press this button to open the (external) ASCOM chooser. It offers a selection of all available drivers (or hubs). Please do not forget to configure the selected driver by pressing the “Properties” button.  D:\Python\MoonPanoramaMaker\Documentation\Screenshots_1-0-0\1-0-0_Configuration-dialog-ASCOM-chooser.PNG |
| **Guide pulse duration:** | Duration (in seconds) for tracking corrections issued during video recording. This prevents the Moon from drifting away for videos longer than a few seconds. The longer the pulses, the less frequent they are inserted. Pulses which are too long result in a jerky motion. |
| **Pulse guide speed RA:** | Speed of guiding corrections in right ascension. The range of values allowed depends on the driver software. The unit is “degrees per second”. |
| **Pulse guide speed DE:** | Speed of guiding corrections in declination (see above). |
| **Wait interval:** | Looking up the current coordinates where the telescope mount points at is implemented as an iterative loop. The loop finishes when the readouts stop changing. This way it is avoided that a moving mount “on its way” returns some intermediate readouts. The parameter sets the length of one iteration step (in seconds). |
| **Telescope position lookup precision:** | As explained for the “Wait interval” parameter, the mount position lookup is implemented iteratively. Two consecutive readouts are interpreted as being “equal” if their difference both in right ascension and declination is not greater than specified by this parameter (in arc seconds). For some telescope mounts (e.g. Vixen Sphinx NexSXD) the readout keeps on changing in very small erratic steps even if it is running in constant siderial tracking mode. In this case, too small a value can cause an infinite loop. |
|  | **Case INDI:**  It is assumed that the INDI Web Manager is installed on the same system where the INDI server is running. |
| **Standard web browser** | Absolute path to the executable of the standard web browser. |
| **Start / configure INDI server and select telescope driver:** | Press this button to open the INDI Web Manager in the standard web browser. There you can choose the drivers to be connected to the INDI server. Please note that if more than one driver of type “Telescope” is connected, MoonPanoramaMaker will use the first one it finds. Usually it is a good idea not to connect more than one telescope driver at the same time. |
|  | D:\Python\MoonPanoramaMaker\Documentation\Screenshots_1-0-0\1-0-0_Configuration-dialog-INDI-webserver.PNG |
| **IP address of the INDI server:** | The IP address of the system where the INDI server and the INDI Web manager are located can either be “localhost”, i.e. both programs run on the same system as MoonPanoramaMaker, or a fully qualified IP address, such as “192.168.0.1”. Make sure that ports 7624 and 8624 are not blocked by a firewall. |
| **Pulse guide speed:** | The speed of pulse guide operations used by MoonPanoramaMaker for tracking the Moon. “SLEW\_GUIDE”, “SLEW\_CENTERING”, “SLEW\_FIND” or “SLEW\_MAX” can be selected from the drop-down list. |
| **Guide pulse duration:** | Same parameter as in the ASCOM case. It is referred to the explanation given there. |
| **Wait interval:** | Same parameter as in the ASCOM case. It is referred to the explanation given there. |
| **Telescope position lookup precision:** | Same parameter as in the ASCOM case. It is referred to the explanation given there. |

#### Workflow

|  |  |
| --- | --- |
| **Session protocol level:** | The drop-down list offers four levels of detail for session logging. If set to 0, no log info is printed. If set to 1, 2 or 3, the session is documented with increasing level of detail. Wall clock time is printed along with every log record. |
| **Write protocol to file or StdOut:** | Drop-down list with two choices:   * Write the protocol to a file: This is the standard case. The protocol is appended to the file “MoonPanoramaMaker.log” in the user’s home directory. If this option is chosen, additionally an image of the tile arrangement is written in a file named “MoonPanoramaMaker\_xxx\_Tile-Layout.png”, with xxx denoting the time stamp when the layout was created, e.g. 2018-03-30\_21-30 for March 30, 2018, 21:30. * Write the protocol to standard output: The protocol appears in the window where MoonPanoramaMaker was started. Usually this option is used for debugging only. |
| **Focus on a star or lunar feature:** | Drop-down list. For a detailed explanation please refer to Section 4.5. Please note that the labeling of the GUI buttons used for focusing changes, according to the choice of this parameter. |
| **Begin at moon limb or terminator:** | Drop-down list. For a detailed explanation please refer to Section 4.6 |
| **Camera automation:** | Drop-down list with two choices:   * Trigger FireCapture automatically. In this case make sure that the FireCapture program is running, and that the option “MoonPanoramaMaker” is selected in its “PreProcessing” menu. * Trigger the camera manually.   For a detailed explanation please refer to the discussion in Section 4.6 |
| **Camera trigger delay:** | Time in seconds between moving the telescope to a new tile and triggering the camera (has no effect if “Camera automation” is set to manual mode). The optimal value is found experimentally. The goal is to trigger the camera not before the view has become steady. |

#### Tile Visualization

|  |  |
| --- | --- |
| **Figure size horizontal / vertical:** | Width and height (in inches) of the “Tile Arrangement” window, respectively. |
| **Font size for labels:** | Size (in points) of the tile numbers in the “Tile Arrangement” window. If the tessellation is rather fine, a large value can make labels overlap. |
| **Label shift parameter:** | This parameter causes the tile numbers to be printed at the center of the tile (value = 0.) or shifted horizontally (maximal shift for value = 1.). In the case of large tile overlaps shifting the labels can avoid that labels overlap in print. |

#### Alignment

|  |  |
| --- | --- |
| **Minimum auto-alignment interval:** | If auto-alignment is active, MoonPanoramaMaker automatically inserts new alignments between video recordings as necessary. The time between alignments is adapted dynamically. It will not drop below the value given by this parameter (in seconds). |
| **Maximum auto-alignment interval:** | The time between automatic alignments will not exceed the value given by this parameter (see above). |
| **Maximum alignment error:** | Criterion if an image shift detected during an auto-alignment is accepted or not. It is expressed as the fraction of the overlap width between tiles. (Example: “40” means that the alignment error must not exceed 40% of the overlap width.) The rationale behind this is that the panorama creation will fail if the overlap between adjacent tiles is too small. This is likely to happen if the detected pointing error is above a certain threshold. If auto-alignment detects that the current alignment error is too large, all videos taken since the last successful auto-alignment are repeated. (See the discussion in Section 4.8) |

## Appendix B: Algorithms Used by the Program

#### Shape and Orientation of the Sunlit Part of the Moon

For the construction of an optimal tessellation of the sunlit part of the Moon, the precise shape and orientation of the Moon phase must be known. To this end, the following quantities have to be determined:

|  |  |
| --- | --- |
|  | Elongation of the Moon from the Sun ( |
| : | Distance between the Moon and the Sun |
| : | Phase angle of the Moon (. A value of corresponds to new Moon, to half Moon und to full Moon. |
| : | Position angle of the great circle through the Moon and the Sun, as measured at the Moon’s position in topocentric equatorial coordinates, counted from North counterclockwise. |
| : | Position angle of the “upper” pole of the sunlit Moon phase, counted from equatorial North counterclockwise. The upper pole (“N” in the illustration below) is defined as follows: If it is pointing up, the sunlit Moon limb is pointing to the right. The terminator runs from top to bottom somewhere between the right and left Moon limbs. The construction of the optimal tessellation is based on this normalized orientation of the Moon. |



In the spherical triangle Moon-Sun-Celestial North (left illustration) the following equation holds:

= (1)

And from the plain triangle Sun-Moon-Earth (right illustration):

= (2)

From the spherical triangle Moon-Sun-Celestial North it also follows:

= (3)

= (4)

= (5)

= (6)

= (7)

with the following definitions:

|  |  |
| --- | --- |
|  | Topocentric right ascension and declination of the Moon (true coordinates referring to the current epoch and equinoctium) |
|  | Topocentric right ascension and declination of the Sun (true coordinates referring to the current epoch and equinoctium) |
| : | Topocentric distance between Earth and Sun |
| : | Topocentric distance between Earth and Moon ( have to be defined using the same units) |

The input values for are computed for the observing site by using the astronomy program package “PyEphem” (<http://rhodesmill.org/pyephem/index.html>).

As explained above in the definition of the angle , the tile layout is constructed for a normalized orientation of the Moon phase. The center coordinates of all tiles are computed in the corresponding plane first. Only at the end they are converted into offsets from the Moon center in right ascension and declination . Please note that in the normalized orientation the sunlit limb is on the right side, even for waning Moon phases. denotes the angle between the two coordinate systems. This situation is depicted in the illustration below.



Apart from the ephemeris data mentioned above, the construction of the tile layout is based on the following variables which are either provided by the user directly, or computed using simple transformations from user-defined parameters:

|  |  |
| --- | --- |
|  | Height and Width of the camera’s field of view (angle). These variables are computed from the linear pixel size, the horizontal and vertical pixel counts, and the focal length of the telescope system. |
|  | Minimal width of the space around the Moon to be covered by the tiles, given as an angle (same in vertical and horizontal directions). The user specifies the minimal number of pixels. From this is computed using the linear pixel size and focal length of the telescope system. |
| : | Minimal overlap between neighboring tiles (angle, both in vertical and horizontal direction). Again, this value is computed from the corresponding pixel count as specified by the user. |

The definition of these parameters is illustrated in the following picture. Tiles are constructed row-wise, starting with the upper boundary in the plane. When the user requirements given above lead to non-integer numbers of tiles in x and y direction, they are rounded up to the next higher integer value by slightly enlarging the tile overlaps. Larger overlaps will help later when the panorama is produced from the tiles. The external margin width is not increased.



The following algorithm computes for all tiles the center coordinates and the corresponding angular offsets relative to the Moon center in right ascension and declination . Here denotes the row index and the column index, both counted from zero. is the topocentric radius of the Moon (angle). It is provided by PyEphem as part of the Moon ephemeris.

= ⌈ ⌉ (8)

= (9a)

= (9b)

= (10)

= (11)

= (12)

= (13)

= (14)

(15)

(16a)

(16b)

= ⌈ ⌉ (17)

= (18a)

= (18b)

(19)

(20)

(21)

= (22)

#### Coordinate Offset of the Landmark Relative to the Moon Center

In the mount alignment process MoonPanoramaMaker needs to know the coordinate differences in right ascension and declination between the landmark and the Moon center. The computation of these quantities depends on the selenographic coordinates of the landmark and the orientation of the Moon as seen from the observer. The Moon’s orientation is quantified by its topocentric (observer-centered) libration angles in longitude and latitude , and the position angle of its rotational axis in the Earth’s equatorial coordinate system. The following algorithm provides these three quantities with sufficient accuracy to determine the coordinate offsets of the landmark to about 3” in right ascension and declination. This accuracy is achieved by, on the one hand, including the topocentric correction of libration angles, while on the other hand neglecting the effect of nutation.

First, using the algorithm in (Simon, Bretagnon, Chapront, Chapront-Touzé, Francou, & Laskar, 1994), the obliquity of the ecliptic and the mean elements and of the Moon’s orbit referred to the mean ecliptic and equinox of date are computed:

(23)

(24)

(25)

(26)

(27)

Now, using the algorithm presented in (Taylor, Bell, Hilton, & Sinclair, 2010), and omitting the higher-order corrections due to nutation, the position angle of the rotational axis and the libration angles are computed:

(28)

(29)

(30)

(31)

(32)

(33)

(34)

(35)

with the following definitions:

|  |  |
| --- | --- |
|  | Current obliquity of the ecliptic |
|  | Inclination of the ecliptic to the mean lunar equator |
| : | Longitude of the ascending node of the ecliptic w.r.t. the lunar equator |
|  | Mean longitude of the Moon |
|  | Inclination of the lunar equator to the equator of the Earth |
|  | Topocentric longitude and latitude of the Moon (true coordinates referring to the current epoch and equinox). These quantities are computed using the PyEphem software. |
|  | The arc of the true equator of the Earth from the true equinox of date to the ascending node of the mean equator of the Moon on the true equator of the Earth |
|  | Position angle of the axis of rotation of the Moon in the equatorial coordinate system of the Earth |
|  | Topocentric libration angles in longitude and latitude, respectively, in the equatorial coordinate system of the Moon |

The computation of the equatorial coordinate offsets of the lunar landmark starts from the landmark’s coordinates in a rectangular coordinate system with its origin at the Moon’s center, the positive axis pointing east, the positive axis lying in the equatorial plane and pointing away from the Earth, and the positive axis pointing North. First, the position vector of the landmark is rotated by , then by , and finally by :

(36)

(37)

(38)

(39)

(40)

(41)

(42)

with the following variable definitions:

|  |  |
| --- | --- |
|  | Selenographic longitude and latitude of the landmark on the Moon |
| : | Topocentric radius of the Moon (angle) |
|  | Offsets in right ascension and declination of the landmark from the Moon center, as measured in the true topocentric equatorial coordinate system of the Earth |

#### References:

Simon, J., Bretagnon, P., Chapront, J., Chapront-Touzé, M., Francou, G., & Laskar, J. (1994). Numerical expressions for precession formulae and mean elements for the Moon and the planets. *Astronomy and Astrophysics, 282*, pp. 663-683.

Taylor, D.B., Bell, S., Hilton, J., & Sinclair, A. (2010). *Computation of the Quantities Describing the Lunar Librations in the Astronomical Almanac.* United Kingdom Hydrographic Office, Taunton, Great Britain.

## Appendix C: Determination of the Focal Length of the System

The precise focal length of the complete telescope system (from the objective lens to the sensor) determines the field of view of the camera and is thus an important input parameter for the construction of the optimal tile layout. Unfortunately, this figure usually is not known a priori if some kind of projection system (e.g. a Barlow lens) is inserted into the light path.

The easiest way to determine this figure is by experiment. In the following three possible techniques based on test exposures of the Moon, are presented.

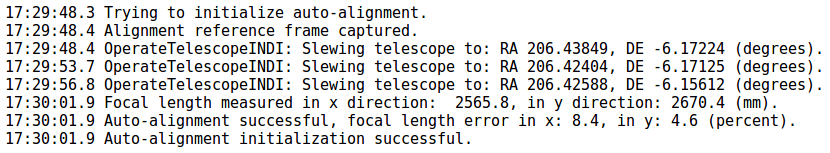
#### Let MoonPanoramaMaker do the work (limited accuracy):

As described in Section 4.8, when MoonPanoramaMaker initializes automatic alignment, it measures the focal length of the telescope system, separately in x and y sensor coordinate directions. If these measurements differ too much from the focal length the user entered in the configuration dialog, auto-alignment is deactivated.

This comparison can be used to measure the focal length experimentally:

* Enter an approximate value for the focal length (parameter “Focal length” in the “Telescope” section, see Appendix B).
* At the “Configure camera” dialog for your camera model, set the value for parameter “Tile overlap pixels” larger than usual. Some 30% of the number of pixels of the short side of the sensor is a good choice. (Do not forget to change the value back to its original value after the experiment.)
* Set the parameter “Session protocol level” to “2”.
* Start MoonPanoramaMaker and let it run through the initialization phase. When the program is ready to begin the recording workflow, press “Switch on auto-alignment”.
* Follow the instruction to center the landmark and press “Enter”.

MoonPanoramaMaker then moves the telescope to three different locations in the neighborhood of the landmark and takes still images. From the measured image shifts and the RA/DE differences of the corresponding locations in the sky, the program computes approximate values of the focal length (in millimeters), separately in x and y direction. The results are documented in the protocol like in this example:



If you take the average of the two focal length values you get a good approximation of the focal length of the complete optical system. In this example the result is 2618.1mm, which is not too far away from the nominal focal length of the Celestron 11 used for the experiment (2794mm).

#### Absolute Method:

This method starts from an exposure of some area on the Moon, using the complete optical system. Two well-defined landmarks are identified in this image, e.g. small and bright craterlets on the mare floor. In some image processing program, e.g. Photoshop, rectangular coordinates und of both landmarks are measured. Using Pythagoras’ theorem their distance is computed as

(43)

In the same image processing program one measures the width of the camera’s field of view (in the same measurement unit as the (x,y) coordinates).



In a planetarium program (e.g. Guide 9.0) the Moon’s appearance is simulated for the time when the exposure was taken. Here the angular distance of the two landmarks is measured. In the example below the distance is found to be arc seconds.



With the known camera parameters (see Appendix A: Parameters at the Configuration Dialog)

: “Pixel count horizontal” and

: “Pixel size” (pitch)

the sought focal length of the overall system is computed as:

(44)

Please note that published values for may be unreliable. As explained in “Appendix A: Parameters at the Configuration Dialog”, this parameter defines the horizontal or vertical distance of two adjacent sensor pixels, and not the size of the light-sensitive cell itself.

If no reliable value for is available, it can be found experimentally if a telescope with well-known focal length is available. For this purpose a refractor or Newtonian reflector is to be preferred. (Using a Schmidt-Cassegrain telescope is not recommended because its effective focal length depends on the location of the sensor in the light path behind the primary mirror cell.) As explained above, from an image taken with such a telescope is derived as:

(44a)

Obviously it is impossible to measure both the focal length and pixel distance at the same time. Fortunately, for the purpose of MoonPanoramaMaker the absolute values of both parameters do not matter, as long as they are consistent according to equation 44. A wrong value for leads to a wrong value for . If used as input for MoonPanoramaMaker, however, both errors cancel out each other.

#### Relative Method:

There is an easier way if the primary focal length of the telescope (i.e. without projection system) is known precisely enough. In this case two exposures are taken, one at the prime focus, the other one with the projection system installed.

As described above for the full optical system, the linear distance of the two landmarks is measured in the prime focus exposure as well. The resulting distance is denoted .

With these measurements, the focal length of the overall system is computed as:

(45)