Sky Imager Aggregator V2 Documentation for Admin/User

## Short description of the system as a whole

The RaspberryPi is connected to the camera through an Ethernet cable. The purpose of this system is to accesses the camera to take a picture every 10 seconds. Once the photo is taken a mask is applied to cover the surrounding objects. When this process is finished the software accesses the server and sends the photo.

The system consists of 3 python3 scripts. Two executable scripts: SkyImagerV2.py (download images from camera and try to send to remote server) and SendStorageV2.py (send the images that for some reason were not sent on time) and the library LibrforPiV2.py written for the needs of the program. Systemd controls the scripts execution.

### Minimum HW requirement

* RaspberryPi 3 Model B+
* Noobs microSD
* Power supply 2.5A 5.1V
* Ethernet cable
* camera

### Library and SW

The RaspberryPi works on Debian operating system which is a Linux based system. For use of this code you will need python 3.5 or higher.

You will need the following libraries:

* cv2 - openCV
* numpy
* datetime
* base64
* json
* hashlib
* hmac
* requests
* astral – for compute sunrise and sunset - https://astral.readthedocs.io/en/latest/
* configparser
* logging
* apscheduler – for schedule job - https://apscheduler.readthedocs.io/en/latest/
* os

OpenCV library you can install by following this link:   
https://www.pyimagesearch.com/2018/09/26/install-opencv-4-on-your-raspberry-pi

To install most libraries use command: sudo pip3 install <name\_of\_library>

### Download and install scripts

Using the code is pretty simple and can be done in two steps:

1. You need to download the repository from github address https://github.com/UCEEB/Sky-Imager-Aggregator

### Configuration scripts

To set most general parameters a configuration file is available. The file is read at the very beginning at main program start up. You can find the file in same folder as python scripts. Parameters:

* *cap\_mod* = capture interval in seconds. Image is captured when following condition holds:
* default value is 10 s
* *cap\_url* = IP camera url
* path\_storage = path to temporary images storage when is unavailable internet connection to server
* upload\_server = url to upload server
* mask\_path = path to black and white mask file
* log\_path = path where scripts store log files
* log\_to\_console = set true if you wish display log message on console
* debug\_mode = true force script to save image to local storage
* upload\_format = not yet used
* crop = crop size relate to image from camera – four values left, top coordinates and width, height of cropped image
* camera\_latitude = camera position for calculate sunrise and sunset
* camera\_longitude =
* camera\_altitude =
* added\_time = computed sunrise shifts *added\_time* minutes early and sunset shifts *added\_time* minutes later
* filetime\_format = python datetime format for image filename

Example of configuration file:

[SETTING]

# path to temporary images storage when is unavailable internt connection to server

path\_storage = /home/pi/Sky-Imager-Aggregator/STORAGE

# url to IP camera

cap\_url = http://10.208.8.59/JpegStream.cgi?username=EA06B8A4521DF75A407CD98FA6693865B9DE198EE69AF60F913514B09B7573D7&password=4FA7493F1361E8C4DA95E4BBF52B0666752F59B59C4A77B1AFA7F43E10E10659&channel=1&secret=1&key=6B4jkkkc

# url to upload server

upload\_server = http://www.pvforecast.cz/api/aers/v1/upload.php?type=pic&signature=

mask\_path = /home/pi/Sky-Imager-Aggregator/config/bwmask.png

log\_path = /home/pi/Sky-Imager-Aggregator/log

cap\_mod = 10

log\_to\_console = false

debug\_mode = false

upload\_format = JSON

image\_quality = 70

#crop image from camera left,top width,height

crop = 331 , 45 , 1926 ,1926

# camera position for calculate sunrise and sunset

camera\_latitude = 50.1567017

camera\_longitude = 14.1694847

camera\_altitude = 360

#minutes added time between sunrise and sunset

added\_time = 10

filetime\_format = %%y-%%m-%%d\_%%H-%%M-%%S.jpg

### Installation and use scripts as daemon

Scripts are running as daemons executed by systemd.

To create systemd daemons do this:

First edit sky\_image\_aggr.service, sky\_image\_aggr-send\_storage.service located in systemd subdirectory and set right path to python scripts SkyImagerV2.py and SendStorageV2.py

Copy sky\_image\_aggr-send\_storage.service, sky\_image\_aggr-send\_storage.timer, sky\_image\_aggr.service from systemd directory to /lib/systemd/system/

sudo cp sky\_image\_aggr-send\_storage.service /lib/systemd/system/

sudo cp sky\_image\_aggr-send\_storage.timer /lib/systemd/system/

sudo cp sky\_image\_aggr.service /lib/systemd/system/

Then enable and start these daemons

sudo systemctl enable sky\_image\_aggr-send\_storage.timer

sudo systemctl start sky\_image\_aggr-send\_storage.timer

sudo systemctl enable sky\_image\_aggr.service

sudo systemctl start sky\_image\_aggr.service

And that is all

**Some useful commands:**

List of all timers:

sudo systemctl list-timers --all

Display daemon log:

sudo journalctl -u sky\_image\_aggr-send\_storage.timer

sudo journalctl -u sky\_image\_aggr-send\_storage.service

sudo journalctl -u sky\_image\_aggr.service

Daemon restart:

sudo systemctl restart sky\_image\_aggr-send\_storage.timer

sudo systemctl restart sky\_image\_aggr.service