

TUTORIAL

Jonathan St-Antoine

Through Focus CRED Script

| | |
|---|-----------|
| General | 2 |
| Scope | |
| Applicable document | |
| Acronyms | |
| Introduction | |
| Scripts | 3 |
| Running script | |
| Configuration file | 5 |
| Starting the CRED with Micro-Manager | 6 |
| Getting Started | |
| Testing connection | |
| cred_script.py | 8 |
| Manually Run the Analysis | 9 |
| CSV logs | 10 |

1 General

1.1 SCOPE

This document describe the steps to acquire images and analyse those images during NIRPS through-focus test. This document is does not cover the theory of the through focus test.

1.2 APPLICABLE DOCUMENT

| Doc no. | Document Name | date | version |
|---------|---------------|------|---------|
| | | | |
| | | | |
| | | | |

1.3 ACRONYMES

1.4 INTRODUCTION

A sets of python script have been written to perform the through focus of NIRPS. The measurement is best done with two person, but with the help of the scripts it can be done alone. The thought focus consists in sweeping multiple focus position with a camera, called CRED, an curve fit the result.

2 Scripts

The scripts can be downloaded here: <https://github.com/espressjo/cred-through-focus.git>. The repository should contains the following scripts:

1. **version.py** Small script that will check if all the required package are installed.
2. **cred.py** Can be used to display tmp image in DS9.
3. **im2csv.py** Use to create a log file in csv form.
4. **analyse_cred.py** Perform the through focus analysis.
5. **cred_script.py** Main script. Most probably the only one you will run.

2.1 RUNNING SCRIPT

Every script should be run in an anaconda python prompt. Figure 2.1 shows where to launch the python prompt. In the prompt, navigate to the directory where the python script are located. **Remember** this is window, to list file inside a directory user *dir* not *ls*. Once in the scripts folder, you can execute a script by typing this; `python <name-of-the-script>.py` .

For example;

```
python cred.py -help
```

will print the help menu of the cred.py script. Most of the scripts have an help argument.

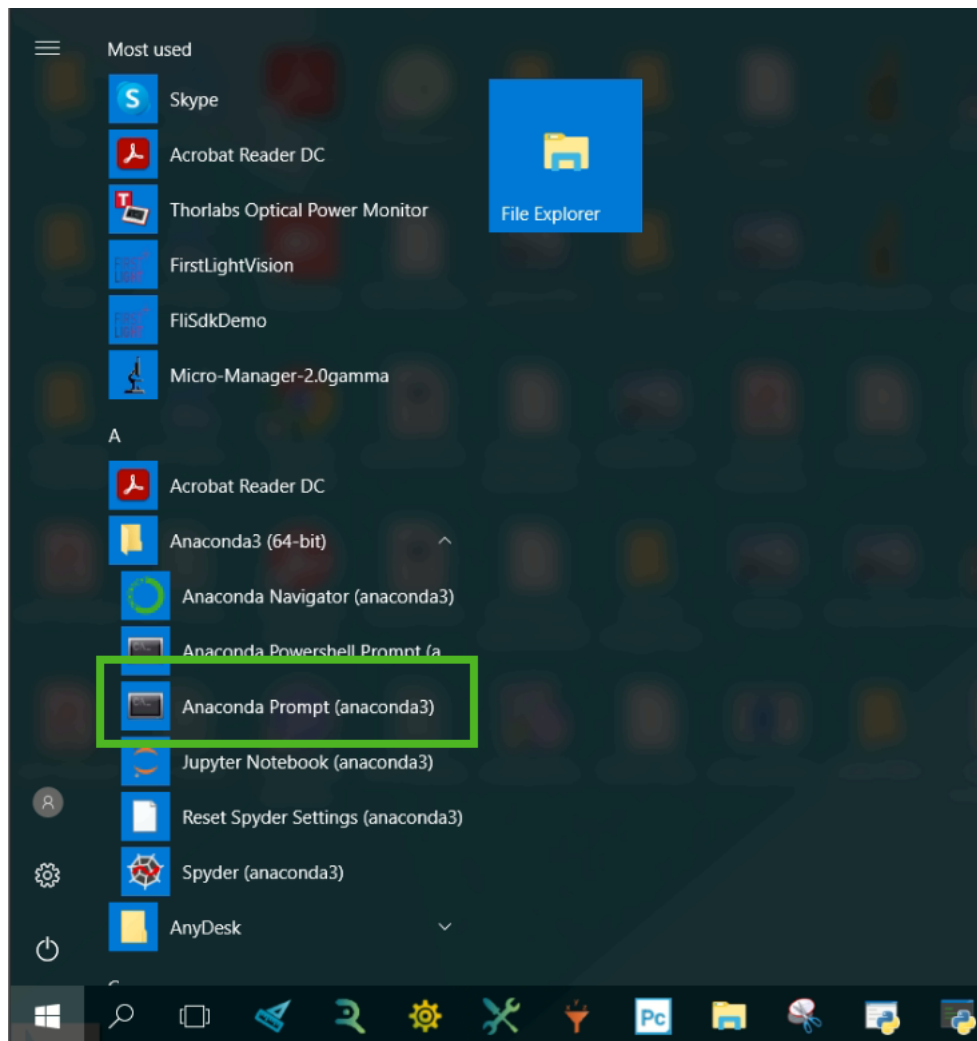


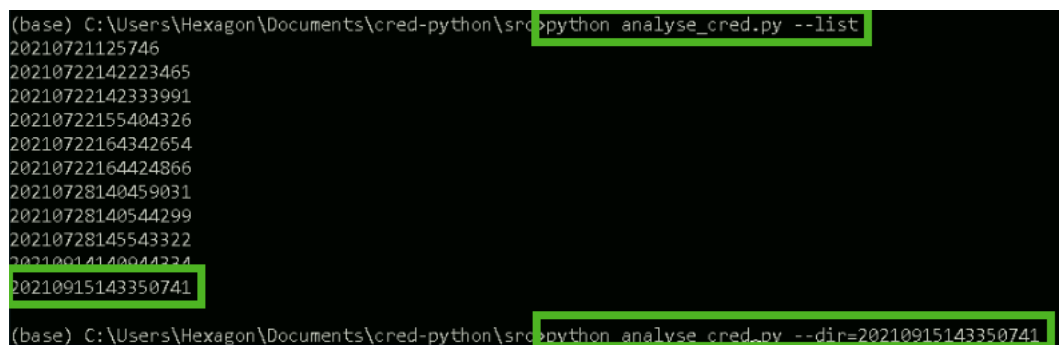
Figure 1: Anaconda prompt.

3 Configuration file

In the repository main folder there is a configuration file called cred.cfg. A set of argument can be predefined or changed by changing the values of each keywords. This is the list of all keywords;

1. **IMGNUM** 25 number of images taken at each focus position.
2. **DETTEMP** -15 Detector target temperature.
3. **TIMEOUT** 3 Delay before a set of images is taken.
4. **NDF** 0 ND filter value if used
5. **WDIR** C:/Users/Hexagon/Desktop/ base-directory where everything will be saved.
6. **COMMENT** my comment!
7. **COMMENT** my 2nd comment!!

The exposure time being very short, a few tens of millisecond, we can acquire a lot of images at each position. The IMGNUM keyword can be use to set the number of images taken. In case the test is performed alone, a delay (TIMEOUT keyword) can added before the script start data acquisition. The WDIR keyword is very useful. If sets, all the script will know where the data are saved. A special argument, `-list`, is available in `analyse_cred.py` and `img2csv.py`. This key argument list all the sequence folder in the working directory (WDIR). Another special argument, `-dir` is also usefull to launch the script in specific sequence folder. Figure 3 shows an example of the `-list` and `-dir` argument. A special keyword has been added, COMMENT, to include any additional information in **all** the images header. Each new COMMENT entry in the configuration file will add a new comment entry in the headers.



```
(base) C:\Users\Hexagon\Documents\cred-python\src>python analyse_cred.py --list
20210721125746
20210722142223465
20210722142333991
20210722155404326
20210722164342654
20210722164424866
20210728140459031
20210728140544299
20210728145543322
20210914140944274
20210915143350741
(base) C:\Users\Hexagon\Documents\cred-python\src>python analyse_cred.py --dir=20210915143350741
```

Figure 2: Example of how to use the `-list` and `-dir` argument once the WDIR keyword is sets.

4 Starting the CRED with Micro-Manager

Micro-manager is the software that directly communicate with the CRED camera. All of the python script use micro-manager server functionality to trigger control commands. Make sure Micro-manager is running before using any of the scripts.

4.1 GETTING STARTED

Launch Micro-manager (see figure 4.2). You will be welcome by the software startup configuration panel (see figure 4.2). If the hardware configuration file is not MMConfig_cred2.cfg use micro-manager file browser to select the file. The configuration file is located here;

C:\Program Files\Micro-Manager-2.0gamma\MMConfig_cred2.cfg

Press **ok**. Once the hardware configuration file has been uploaded you can start using python scripts.

4.2 TESTING CONNECTION

A test function has been added in the cred_script.py script. Launch the script with argument `-test`. Figure 4.2 shows an example of output when the camera is not connected. If this is the case, investigate the hardware.

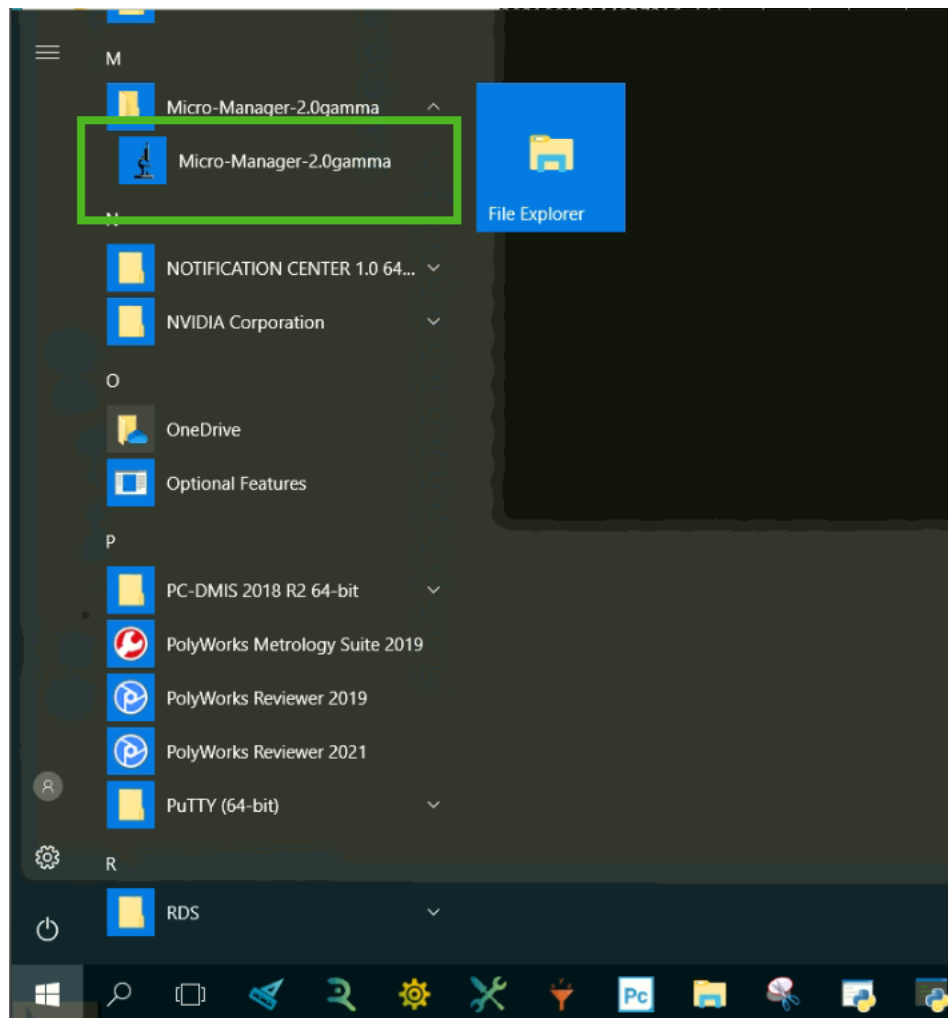


Figure 3: Micro-manager launcher.



Figure 4: Micro-manager startup configuration panel. Make sure the hardware configuration file is MMConfig_cred2.cfg.

```
(base) C:\Users\Hexagon\Documents\cred-python\src>python cred_script.py --test
::: Checking version :::
natsort: 7.1.1 [OK]
astropy: 4.2.1 [OK]
numpy: 1.20.1 [OK]
pycromanager: 0.13.1 [OK]
matplotlib: 3.3.4 [OK]
re: 2.2.1 [OK]
scipy: 1.6.2 [OK]
tqdm: 4.59.0 [OK]
Camera (CRED) [Not Connected]
Exiting...
```

Figure 5: output of python cred_script.py --test. In this case, we see that all python module satisfies the requirement, but the camera is disconnected.

5 cred_script.py

This is the main script. Follow instruction in section 2.1 to launch the script. The script will 1st test the camera connections. After, it will set the CCD target temperature and display a real time graph of the current CCD temperature. When the temperature reach the desired target, close the graph to go to the next step of the script. If you do not want to see the interactive temperature graph, use the `--no-ccdt` argument.

Next the script will ask a couple of questions.

1. **Name of the person using this script?** <string>
2. **Wavelength of the laser?** <float/int>
3. **Power of the laser source in mW?** <float>
4. **Name of the program ?** If you want to label this sequence with a specific name <string>

From now on, the script will ask in an infinite loop the following questions;

1. **Position in micron of the micrometer?** <float/int>
2. **Exposure time in milliseconds?** <int>

Before each iteration, the user will be ask to press anykey to continue. At this time the user can enter -1 or -2. -1 will abort the acquisition loop (his will launch the analysis automatically). -2 Will let the user enter a temporary comment in the headers of ALL the images for the next micrometer position.

6 Manually Run the Analysis

If for some reason the PDF report was not automatically created at the end of the `cred_script.py` routine, or if you want to re-run the analysis, you can use the `analyse_cred.py` script. Use the `-list`, `-dir` argument or the `-path` argument to specify which folder to run the script in.

The script will create a `results.pdf` file inside the specified directory. The graphic should automatically be displayed as well.

Figure 3 shows an example of how to use these arguments.

7 CSV logs

If for some reason the logs was not automatically created at the end of the `cred_script.py` routine, you can create the log with the `img2csv.py` script. Use the `-list`, `-dir` argument or the `-path` argument to specify which folder to run the script in.

The script will create a `log.csv` file inside the specified directory. You can use libre office calc spreadsheet software to examine the CSV file.