A Step-By-Step Guide for Reducing Time-Series Photometry with the HiPERCAM Pipeline for the Boston University White Dwarf Group

By Joseph Guidry[*]

v0.1: last updated: March 27, 2025

Contents

[*] Please don't hesitate to email Joseph at jaguidry@.bu.edu for questions and assistance.

1 Conventions for this document

1.1 Fonts

Text written in `typewriter font` and enclosed in thistle-colored boxes is generally meant to be copy and pasted into the command line, with individual commands being separated from one line to the next. For example:

```
mkdir test_directory
cd test_testdirectory
```

1.2 Disclaimer on Hardware and Software

For posterity, I developed this document and the instructions herein working from my MacBook (MacOS=v15) whose Terminal operates using Zsh, not bash. While the necessary adjustments from Zsh to bash should be trivial, it is unclear how installation will be affected from MacOS to Windows, for example.

2 Installation and Configuration

2.1 Installing `hipercam`

You must first install and configure the HiPERCAM pipeline on your machine before you can crank out any light curves.

This is best done by creating a new `conda` environment[1]:

```
conda create -n hipercam python=3.9.19
```

You can then install the pipeline as directed on `https://github.com/HiPERCAM/hipercam`:

```
mkdir hipercam
cd hipercam
conda activate hipercam
git clone https://github.com/HiPERCAM/hipercam.git
pip install .  --user
```

2.2 Installing `pgplot`

You must next install the dependencies for `hipercam`. These include `pgplot`. For Mac users, this is best achieved through `homebrew`: follow these instructions.

Next you must install Tom Marsh's Cython wrapper to `pgplot`: follow these instructions.

2.3 Installing `phot2lc`

All still within your `hipercamconda` environment, you should next install `phot2lc`, Zach Vanderbosch's light curve extraction tool designed to replace WQED:

[1]The selection of this Python version is arbitrary. I have not tested this on more recent version releases (v≥3.10). At the time of developing this pipeline, v3.9.19 was the most up-to-date version of Python 3.9.

```
pip install phot2lc
```

You next need to configure phot2lc to be compatible with our hipercam-reduced photometry:

```
photconfig
```

If you happen to be lucky enough to be Joseph Guidry and reducing PTO+PRISM photometry on his machine, your inputted parameters would be:

```
author             = Joseph Guidry
image_list_name    = hcm.lis
pixloc_name        = aperture.ape
photbase_name      = output
stardat_location   = /Users/astrojoe/Research/Variability/phot2lc/stars.dat
default_telescope = pto
default_source     = hcm
default_image      = None
default_object     = None
```

See https://phot2lc.readthedocs.io/en/latest/configuration.html for more information. stars.dat is simply a text file with three columns: the object's name, and then its RA and dec in HH MM SS.SS +/−DD MM SS.SS formatting.

Finally, for users of PTO+PRISM and LDT+LMI, see Appendix **??** for the necessary hard-code to pasted into to the teledat.py file within phot2lc.

2.4 Install Remaining Dependencies

It is likely you will not have all the requisite Python packages installed to run phot2lc, and perhaps even hipercam. You likely are lacking–based on my personal experience–pandas:

```
conda install -c conda-forge pandas
```

And lmfit:

```
pip install lmfit
```

And possibly even PyQt5:

```
pip install PyQt5
```

Do check the requisite dependencies for hipercam and phot2lc and compare against what you currently have installed (run conda list in Terminal) to be most prudent.

You probably also need to have XQuartzinstalled to be able to use PGPLOT.

The final Python package to install is ccdproc, which is used in the Python image reduction routine calibrate_science_images that I custom-built into hipercam:

```
conda install -c conda-forge ccdproc
```

2.5 Configuring `hipercam`

2.5.1 Updating your Zsh Path

After every **`hipercam`** installation[2], make sure "/usr/home/.local/bin" is in path so you can execute the **`hipercam`** command line commands. Again, to use my machine as for illustration, this check would look like:

```
echo $PATH
path+=(/Users/astrojoe/.local/bin) to append OR
path=(/Users/astrojoe/.local/bin $path) to prepend
export PATH
source ~/.zshrc
```

`hipercam` will inform you of the exact directory you need to append into your path, which will vary from user-to-user. See this stackoverflow post for more information on this process of updating your Z shell path. I've had issues with keeping this path amendment permanent, so you may need to re-append after a reboot.

2.5.2 Updates to `hipercam` Hard Code

To be able to run this pipeline as instructed, I had to make various amendments to the **`hipercam`** source code. From this GitHub repo, download the following edited files I prepared to replace the originals within in your **`hipercam`** directory:

- `fits2hcm.py`
- `setaper.py`
- `reduce.py`
- `calibrate_science_images.py` – this is a novel routine I wrote
- `setup.py`

Each file should be uploaded to `'~/hipercam/hipercam/scripts/'`, except `setup.py`, which belongs in the root directory, `'~/hipercam/'`.

You must re-install **`hipercam`** so these edits to the source will take effect (**again, still with your `hipercam` conda environment activated AND within your `hipercam` directory**):

```
pip install .  --install
```

You also need to replace a few **`phot2lc`** files. These should be uploaded in, using my machine again as an example: '/Users/astrojoe/Python/anaconda3/envs/hipercam/lib/python3.9/site-packages/phot2lc/'. The files to download are:

- `photfunc.py`
- `teledat.py`

3 Calibrate your Science Images

You now have all the necessary software installed and configured on your machine. It's now time to calibrate your raw photometry. We achieve this by running: `calibrate_science_images --i <instrument>`

[2]You must re-run `pip install . --user` within your **`hipercam`** directory after making any edits to the source code for the updates to take effect.

The instrument choices are `ProEM`, `PRISM`, and `LMI` (or another instrument that `hipercam` is compatible with).

This routine works form the assumption that you have organized your nightly data in a directory structure such as the following:

- `bias`
- `dark` (if applicable for your instrument!)
- `dome_flat` or `sky_flat`
- `WDJHHMM+/−DDMM` – it's not necessary to name your folder after the white dwarf's coordinates, but you science images should be in some unique folder separate from your calibration images. You should also have a unique folder for every target observed.

These all would live in a directory named after the UTC date of the night you observed: 'YYYYMMDD/', which again is not essential, but is a wise convention to adopt.

This structure is necessary because you must run `calibrate_science_images` (and all the remaining `hipercam` routines) within the folder of the target whose photometry you are attempting to reduce. Once commanded, `calibrate_science_images` will prompt you to edit the header information of your science images and check the timestamps. It will then produce master calibration images (if they have not yet been made from reducing another object of this night), and then bias subtract, dark subtract (if applicable), and flat field your raw science images, finally saving them as new copies, now with 'c.fits' extensions.

`calibrate_science_images` also writes out two files: aperture.ape and reduce.red, which will be used below, along wtih creating a new directory for you .hcm files that you will soon write out. Before reducing and extracting your photometry, you should be choose how you wish to extract your photometry. By default the file is created to do PSF photometry using variable apertures in the "optimal" mode. You can instead do fixed aperture photometry, the "normal" mode. You can change between these modes in a text editor. A meticulous astronomer would reduce their photometry using both methods, along with tinkering with other settings, to achieve the highest fidelity photoemtry. Refer to Tom Marsh's notes for more information:
https://cygnus.astro.warwick.ac.uk/phsaap/hipercam/docs/html/photometry.html.

4 Reduce your Photometry with `hipercam`

With our calibrated science images, we can finally perform our photometry and extract light curves.

But first, we must convert our 'c.fits' images into file formats that `hipercam` can read. We do this by running `fits2hcm`:

```
fits2hcm hcm.lis [origin] True
```

Here, `origin` is either "PRISM", "ProEM", or "LMI".

UPDATE ADDITIONS TO FNAME, BNAME, ONAME AND PROEM AND LMI COMPATIBILITy

Next, we have to set the aperture positions of our target and comparison stars using `setaper`:

```
setaper hcm_files/[first image] aperture.ape 10 20 30 xlo xhi ylo yhi True p 1 99
```

In the pop-up window, mouse over your target and type the 'a' key. Repeat for each comparison star.

It can be prudent to trim the image to exclude the margins from consideration. For example, a 410×380 PRISM image could be trimmed to xlo=10, xhi=400, ylo=10, yhi=370. ProEM images should use the native dimensions: 0 256 0 256. If your target is too faint for the PSF to be identified, try reducing the range percentile range to 5 95.

Finally, run `reduce` to perform the photometry extraction:

NEED TO make COPIES OF REDUCE.RED for each instrument

```
reduce source=hf hcm.lis False reduce.red output.log 0 True False
```

The final 'False' parameter can be changed to 'True' to prompt a window to pop up that projects each image and the apertures as you are performing the photometry from one image to the next. It is best practice to monitor this in case a aperture begins to wander away from a target or comparison star and begin to randomly walk around the images. This will only occur if the source is extraordinarily faint. You can guard against this by reducing the box size to search for Gaussian sources within when the variable aperture positions are being searched for: reduce the 'search_half_width' value in reduce.red. If some comparison stars prove to be problematic, either because they are so bright they go non-linear, or wander too near the edge of the chips, for example, re-run `setaper` and then re-run `reduce`.

5 Extract your Light Curve with `phot2lc` and Post-Processing

Ta-da! You reduced your photometry. Now we need to extract our light curve, clean it up with some processing, barycentric-correct the timestamps, and write it out, all using the `phot2lc` suite.

To extract your light curve with `phot2lc` simply call (still in that `hipercam` conda environment!):

```
phot2lc [-t --telecode] [-o --object]
```

For PTO photometry, you may have to include the object name: WDJHHMM±DDMM, where HHMM and DDMM are the sexagesimal RA and dec coordinates in hours and minutes and degrees and minutes, respectively.

Relevant telecodes to BUWD observers:

- `pto` – Perkins Telescope Observatory, set as default (§ 2.3)
- `mcd` – McDonald Observatory 2.1-m Telescope
- `ldt` – Lowell Discovery Telescope

When using `phot2lc`, consult Zach's documentation for all the commands and best practices:
https://phot2lc.readthedocs.io/en/latest/quickstart.html#basic-usage

You can save a quicklook plot of the light curve you extracted using the `phot2lc` script `quicklook`:

```
quicklook [-f --.lc filename] [-s --save plot] [-p --prewhiten periodogram]
```

Another useful post-processing script is `weldlc`. Say you had two runs within the same night of the same target, but had to change exposure times because of clouds. You can combine these into a single light curve file using `weldlc`, which will correctly order the timestamps from the barycentric corrections that `phot2lc` applies. You can weld together as many light curves as you wish, aggregating years-worth of photometry of a given star if you wish.

```
weldlc [-f --.lc filenames] [-o --output filename]
```