Automated ULTRACAM pipeline

User manual

Richard Ashley

April 9, 2014

1 Description of the pipeline

2 Preparation

2.1 Config file

2.2 Location of the raw ultracam data

2.3 objectdbcreator.py

This is the most important script in the automated pipeline. It takes the raw image data in the ULTRACAM archive and sends it to SExtractor for processing. Based on the SExtractor output, it compiles and maintains a list of objects across all of the frames and each of the channels. These are given as three output catalogs when the script finishes running.

2.3.1 Command line parameters

objectdbcreator.py takes the following command line parameters:

- `runName` This is a path to the `.xml` and `.dat` for a specific run and is specified in the format `YYYY-MM-DD/runXXX` (for example `2013-07-21/run011`).
- `-d[n] --debug [n]` Use this parameter to determine how much output you would like to see while the program is running. There are 3 debug levels, *1* is silent (except for errors) and is the default debug level; *2* shows general progress of the pipeline; *3* shows detailed info to help with debugging. Note that the default is 'silent' and therefore, unless there are errors, you will not see anything on the command line and a long run through the data could last an hour or more. It is recommended that you use `-d2` in most cases.
- `-n[n] --numframes [n]` Specifies the number of frames you would like the script to process. The default is all of the frames in the run. Making this number smaller is useful for running a quick test. For example, `-n100` will run through 100 frames only.
- `-s[n] --startframe [n]` Specifies which frame to start at. The default is frame 1 (the first frame in the run).
- `-c[filename] --configfile [filename]` Allows you to specify an alternative configuration file. By default, the script will look for a file called "ultracam.conf" in the local directory.
- `-C[r,g,b] --channels [r,g,b]` Which channels to operate the pipeline over. By default, the script will process all three channels, namely, r, g, and b. This parameter allows you to specify a subset of these channels. For example, you could omit the processing of the 'green channel' by passing in `-Crb`.
- `-p --preview` Specifying this parameter enables a preview window for each frame and each channel using *Matplotlib*. This allows you to see each frame as it is being processed. The colour palettes match the channel, red for r, green for g and blue for b. The preview window also draws a green circle around each object that SExtractor has identified on that particular frame. Warning: This preview slows down the pipeline significantly so should only be used for information and debugging purposes.

- `-t[n] --sleep [n]` Time to pause (in seconds) between the processing of each frame. Useful for debugging in 'preview' mode.
- `-r --crop` For 'preview' mode, crop the windows to show only the areas that were not masked in the original data. Useful for runs where the windows are fairly small.

2.3.2 Output while running

If the `--debug` option is left to the default value of `1` then the output will be mostly *silent* with only errors appearing in `stdout`. This mode is designed for use during the running of the pipeline across a complete night where we want to suppress a lot of the output. If you are running *objectdbcreator.py* in standalone mode, then `-d2` is recommended.

The output of the script with `-d2` set looks like this:

```
[10:31:25] 00:10:22 Frame: [1681,1681 87%] MJD:56495.1395466 r:1899 g:1030 b:551
[10:31:27] 00:10:19 Frame: [1682,1682 87%] MJD:56495.1396132 r:1900 g:1030 b:551
[10:31:29] 00:10:17 Frame: [1683,1683 87%] MJD:56495.1396799 r:1900 g:1030 b:551
[10:31:32] 00:10:14 Frame: [1684,1684 87%] MJD:56495.1397465 r:1900 g:1030 b:552
```

Where,

- `[10:31:25]` is the current time, in HH-MM-SS format;
- `[00:10:22]` is the estimated time remaining until the run has finish being processed in HH-MM-SS format;
- `Frame:[1681, 1681 87%]` The first number is the absoluted frame number being processed (starts at first frame of the run = 1), the second number is the relative frame being processed (different if the start frame was not = 1), and the percentage completed;
- `MJD:56495.1395466` is the MJD for this frame;
- `r:1899 g:  1030 b:551` shows the number of objects being tracked in each of the r, g, b channels.