rrlyrae_metallicity Documentation

Eckhart Spalding, Ron Wilhelm, Nathan De Lee, Kenneth Carrell

Dec 10, 2019

Contents:

1 Introduction **1**
1.1 Attribution 1

2 Installation **3**

3 Getting started **5**

4 Dewarping **7**
4.1 Requirements 7
4.2 The idea 7

5 Sky orientation and plate scale **13**
5.1 Requirements 13
5.2 The idea 13
5.3 Procedure 13

6 Acknowledgements **17**

7 Indices and tables **19**

CHAPTER 1

Introduction

This software is for

1. determining and removing detector distortion
2. finding a correct detector orientation relative to true North
3. determining the detector plate scale

This was originally made for the LBTI LMIRcam detector, but is generalizable to any project with reference frames of ideal sources and/or an astrometric field paired with a .csv file of astrometric target locations.

The distortion removal is based on a mapping made from a distribution of empirical sources (like a pinhole grid) and an idealized distribution of point sources with a "perfect" distortion. This is them mapped to a perfect distribution without distortions.

The detector orientation and plate scale is determined by allowing the user to mark the identities of stars in a series of astrometric field frames. Following this, all lengths and angles of non-redundant baselines between pairs of stars is calculated to find both the plate scale and angle relative to true North.

1.1 Attribution

Please cite Dewarp *Spalding and Stone (2019)* if you used this code or distortion corrections generated from it. The BibTeX entry is:

```
@misc{dewarp,
  author = {{Spalding}, E. and J. {Stone}},
  title = {Dewarp},
  keywords = {Software},
  howpublished = {Astrophysics Source Code Library},
  year = 2019,
  archivePrefix = "ascl",
  eprint = {1907.008},
  adsurl = {https://ui.adsabs.harvard.edu/abs/2019ascl.soft07008S/abstract},
```

(continues on next page)

(continued from previous page)

```
  adsnote = {Provided by the SAO/NASA Astrophysics Data System}
}
```

CHAPTER 2

Installation

At the moment, if you have Python 3.5 and standard astropy packages, you can clone the code via

```
git clone https://github.com/mwanakijiji/dewarp.git
```

Options with pip install and container files will be available in the near future. I recommend creating a new environment on your system (with Conda, for example) so that you can seal off this software's dependencies from the rest of your system. Once you have done so, activate the new environment, download this `requirements.txt` and install the packages with

```
pip install -r requirements.txt
```

CHAPTER 3

Getting started

This repository provides importable functions that can be used to assemble a pipeline as needed. An example is in the script template_pipeline.py, which you can follow in the next pages.

In the config.ini file, replace the DIR_HOME string with the path to your copy of the repository. (The other directories will be made by the pipeline.)

Specify the other directories in the config.ini file (if you want to change them from the defaults), such as that which will contain the pinhole image in the config file.

Run

```
python template_pipeline.py
```

which will proceed to make directories with the function make_dirs() and then get stuck, because it cannot find the FITS file of the pinholes. Put the pinhole image into its directory, and rerun the above command.

CHAPTER 4

Dewarping

Okay, now we're going to move past the make_dirs() function in the template script and actually try to remap a FITS frame onto a smoothed basis.

4.1 Requirements

A FITS frame of sources which can be centroided like PSFs. Image should be dark-subtracted and bad-pixel corrected so that the centroiding will work right. It CANNOT include NaNs. A useable frame may look like this:

4.2 The idea

We want to find the polynomial coefficients that map between empirical pinhole locations and an idealized grid. We use a direct transliteration of IDL's `polywarp` procedure, which finds the coefficients $K_x^{(i,j)}$ and $K_y^{(i,j)}$ in the following polynomial mapping among (x, y) coordinates between the warped and ideal readouts:

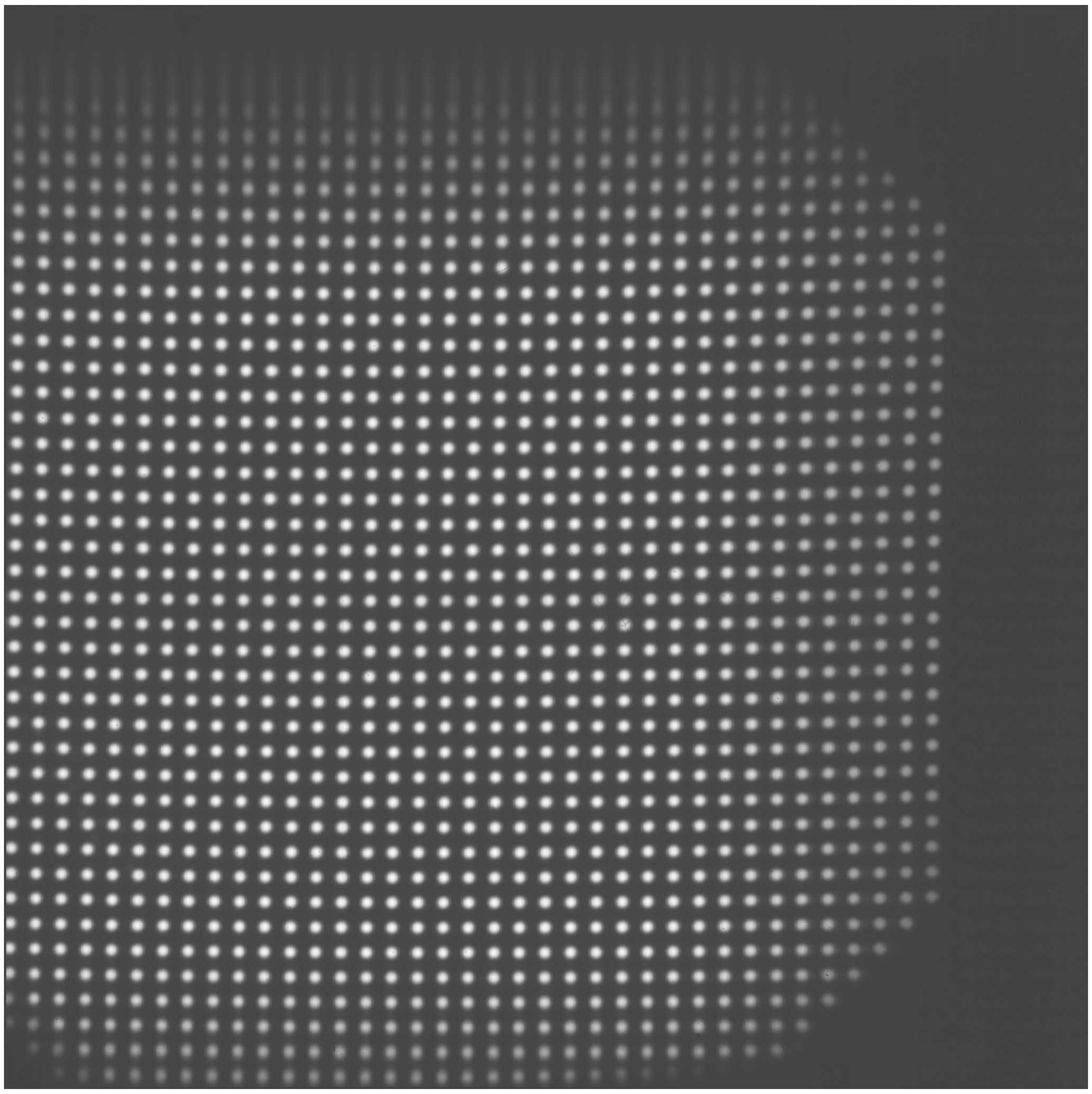
$$x_i = \sum_{i=0}^{N} \sum_{j=0}^{N} K_x^{(i,j)} x_o^{(j)} y_o^{(i)}$$

$$\underbrace{y_i}_{\text{warped}} = \sum_{i=0}^{N} \sum_{j=0}^{N} K_y^{(i,j)} \underbrace{x_o^{(j)} y_o^{(i)}}_{\text{dewarped}}$$

Note which sides of the mapping represent the 'warped' and 'dewarped' coordinates in this application, which may be opposite to what one may expect intuitively, or from the IDL documentation on polywarp. Let's see why we do it this way by plunging into the functions called within the `find_dewarp_solution.py` script.

4.2.1 Match empirical with model sources: `match_pinholes.match_pinholes()`

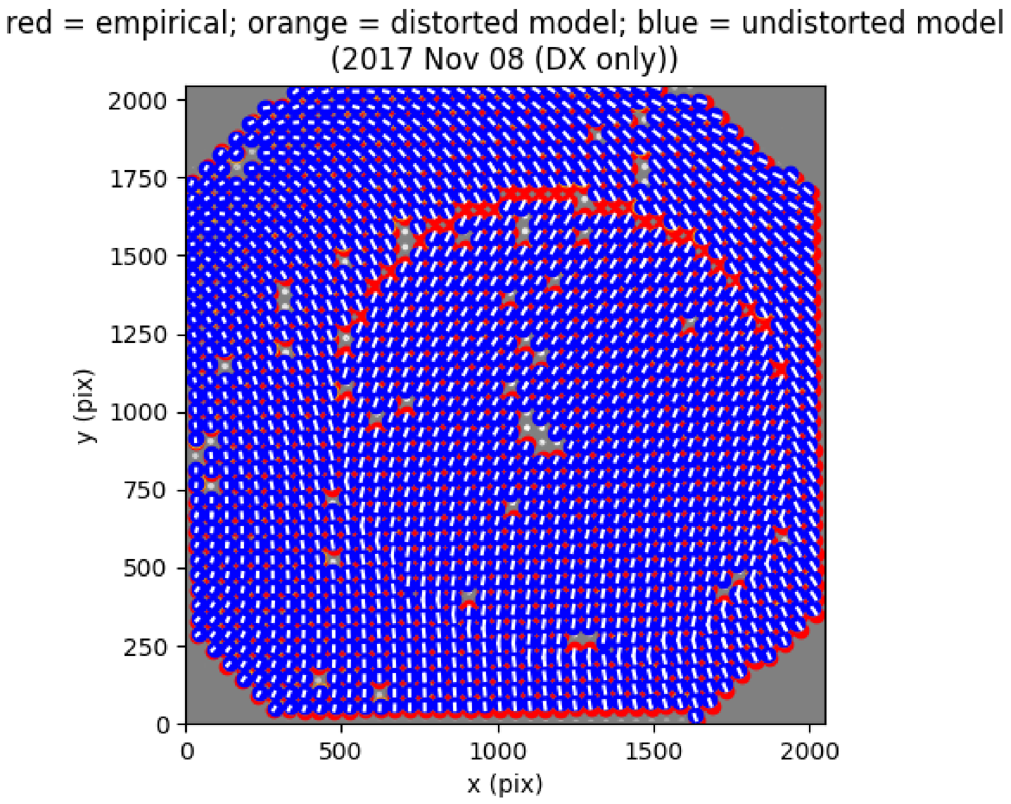
Within the template pipeline `template_pipeline.py`, you will see a call to `match_pinholes.match_pinholes()`. This overlays a model coordinate grid over the image that you need to try to match by tweaking the rotation, offset, and barrel settings. You may have to run through a couple times until you get a good



match. The match doesn't have to be exact; we're just making a grid so that the code recognizes which empirical point sources it detects can be matched with model points

Once that's done, run the script again so that it runs past the function `make_dewarp_coords()`. This finds the aforementioned coefficients by solving a least-squares problem via Moore-Penrose pseudoinverse matrices.

Here is an example of a matching that left some aliasing in the form of a crescent. If you see aliasing like this, tweak the parameters and try again:



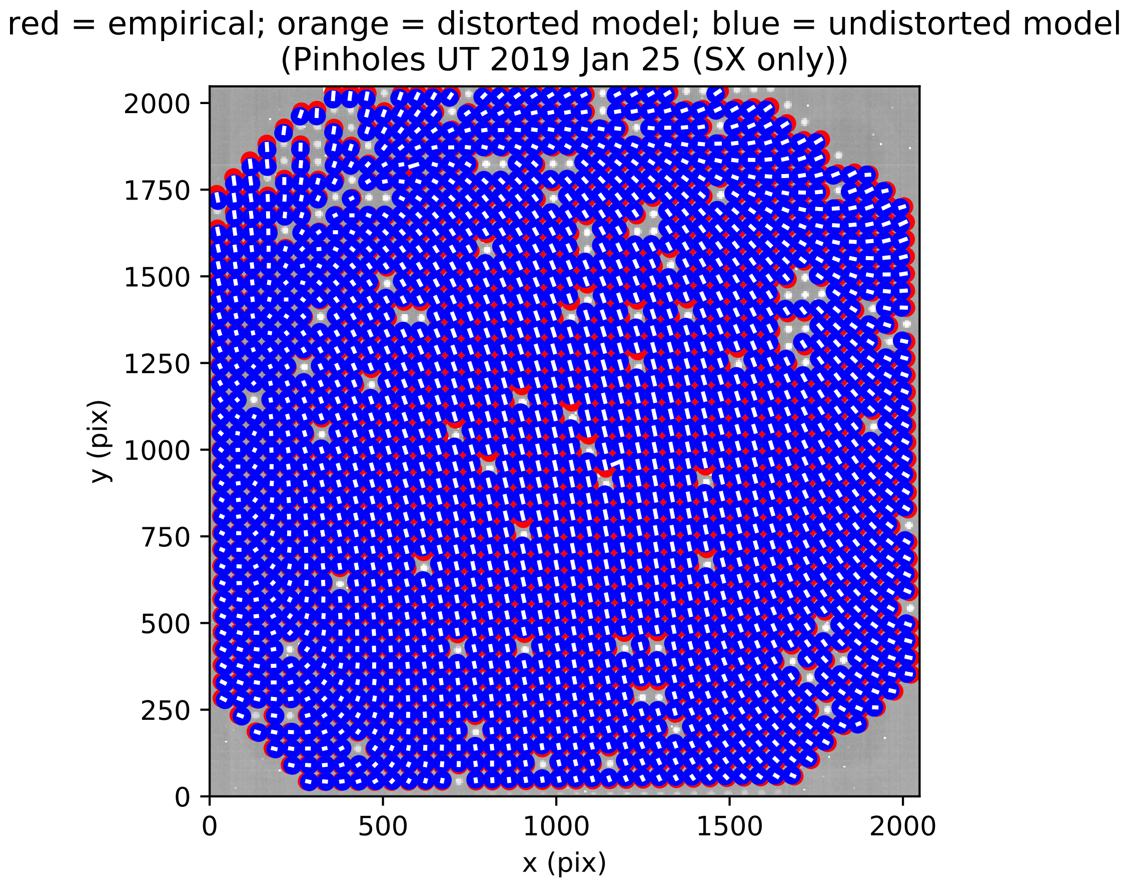
You should end up with something that looks roughly like this:

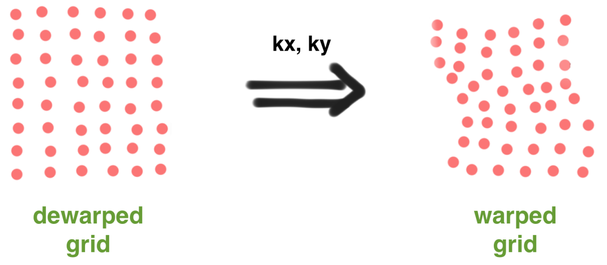
The match with the pinholes isn't perfect, but that's okay. We just need to avoid aliasing. Also note that some individual pinholes have been missed, but the code will just interpolate through them. If you're picky, try changing the parameters of the centroiding and try again.

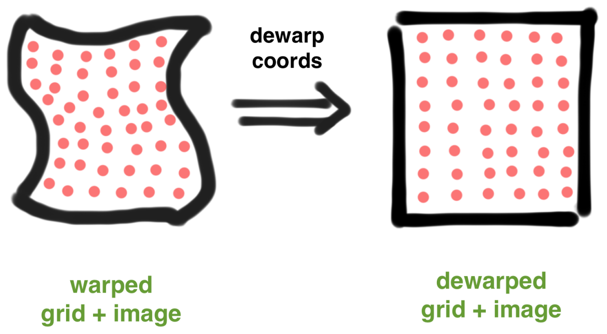
4.2.2 Make the mapping: `find_dewarp_solution.find_dewarp()`

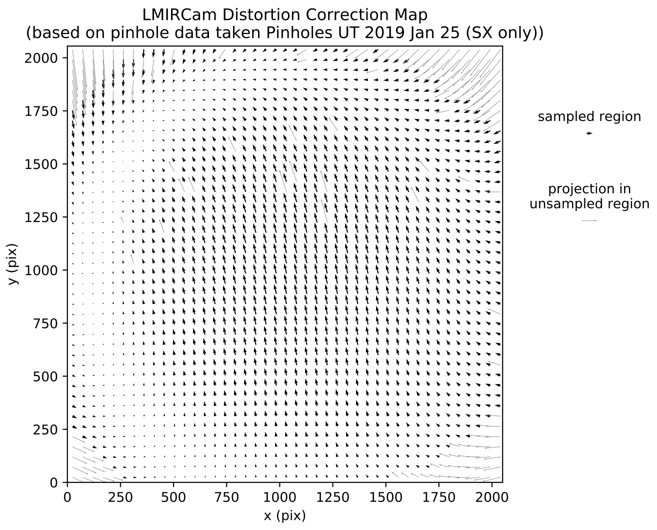
This next function takes the raw image, pastes the warped coordinates onto it, and then smooths everything out by resampling the image point-by-point over the entire image space, interpolating as needed when the coordinates are not at integer values (Fig. `warp_dewarp_grids_annotated`). As a check, closely compare the pinhole grid images before and after (Fig. `barb_plot_sx_2019jan`).

The last part of the script makes a barb plot, putting evenly-spaced vectors over the array to show the directions that points on the readouts have to be stretched in order to dewarp it:


red = empirical; orange = distorted model; blue = undistorted model
(Pinholes UT 2019 Jan 25 (SX only))
y (pix)
x (pix)
0
250
500
750
1000
1250
1500
1750
2000
0
500
1000
1500
2000


kx, ky
dewarped grid
warped grid


dewarp coords
warped grid + image
dewarped grid + image


LMIRCam Distortion Correction Map
(based on pinhole data taken Pinholes UT 2019 Jan 25 (SX only))
sampled region
projection in unsampled region
y (pix)
x (pix)

4.2.3 Apply the dewarp solution

There are two ways of doing this. One is to just let the pipeline run to the next function, `apply_dewarp_solution.apply_dewarp()`. The second way is to copy and the Kx and Ky matrices that were printed to screen in the last step, and paste them into the script `simple_dewarp.py`, and edit that script as necessary to dewarp your full dataset. (For a tarball of a self-contained dewarping package, try `this`.

CHAPTER 5

Sky orientation and plate scale

5.1 Requirements

1. Asterism images (preferably dewarped)
2. A csv of known stellar targets with RA and DEC

5.2 The idea

We flag star positions in pixel space, and compare their locations in RA, DEC. All possible baselines are used between identified stars to find their separations and angles relative to north at PA=0.

5.3 Procedure

5.3.1 Derotate the images based on meta-data (if applicable): derotation.derotate_image_forloop()

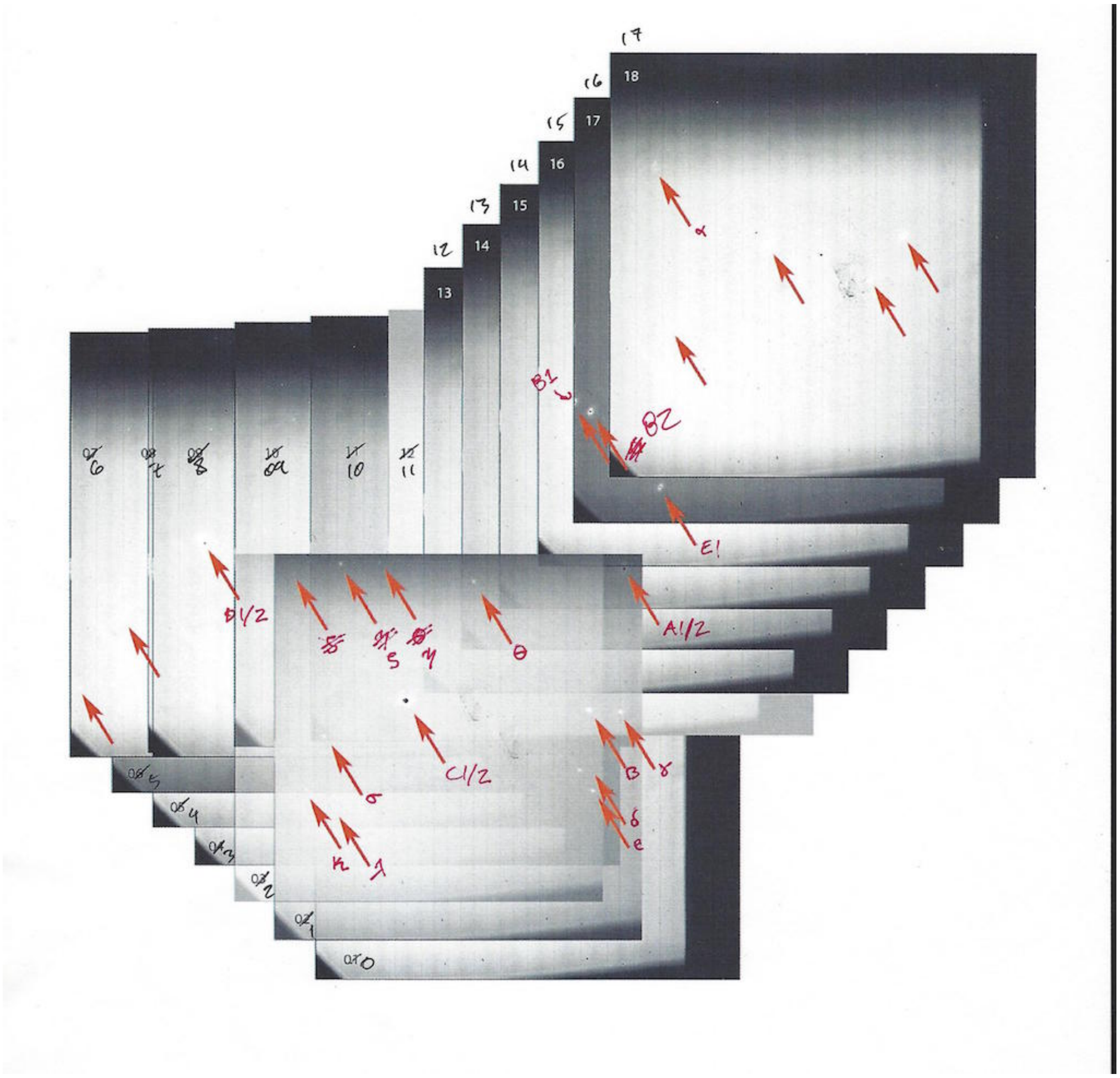
If there is parallactic angle information in the image meta-data (like in the FITS header), use this to derotate the asterism images. We'll find what residual there remains. If you have no parallactic angle meta-data, the following procedure should still work.

Note that in one frame of Ns stars, the total number of baselines among stellar pairs is "Ns choose 2":

$$N_b = \binom{N_s}{2} \equiv \frac{N_s!}{2!(N_s-2)!} = \frac{N_s(N_s-1)}{2}$$

5.3.2 Find stars in pixel space: find_asterism_star_locations.find_stars()

As an aid to identifying the stars, you might want to overlay the derotated asterism images and then cross-check them with a known astrometric source. In the case of the Trapezium Cluster, one can use the images and Table 1 in Close+ 2012 ApJ 749:180. In this figure, I labeled stars using the conventions in Close+ 2012, and used my own Greek lettering if they were without label:



The script find takes the intermediary step of determining star locations in pixel space, and printing locations in pixel space to the screen. Check each centroid manually in the plot, to see if it's a real star or not. Copy the true positive locations in pixel space that are returned in the Terminal.

5.3.3 Find plate scale, angle offset: comparison.angOffset_plateScale()

This function will make CDF plots with 1-sigma-equivalent boundaries, like these:

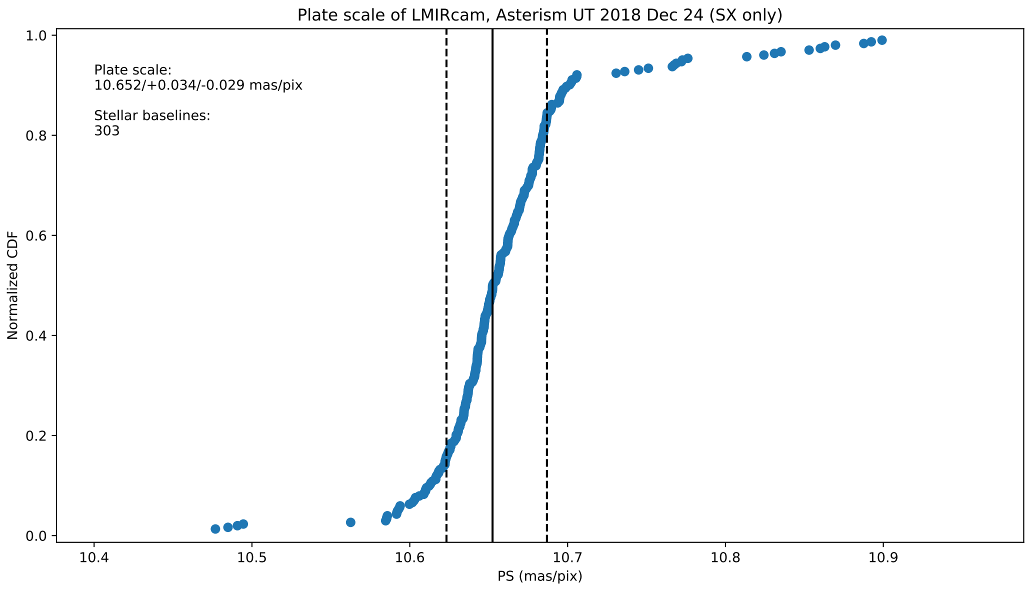
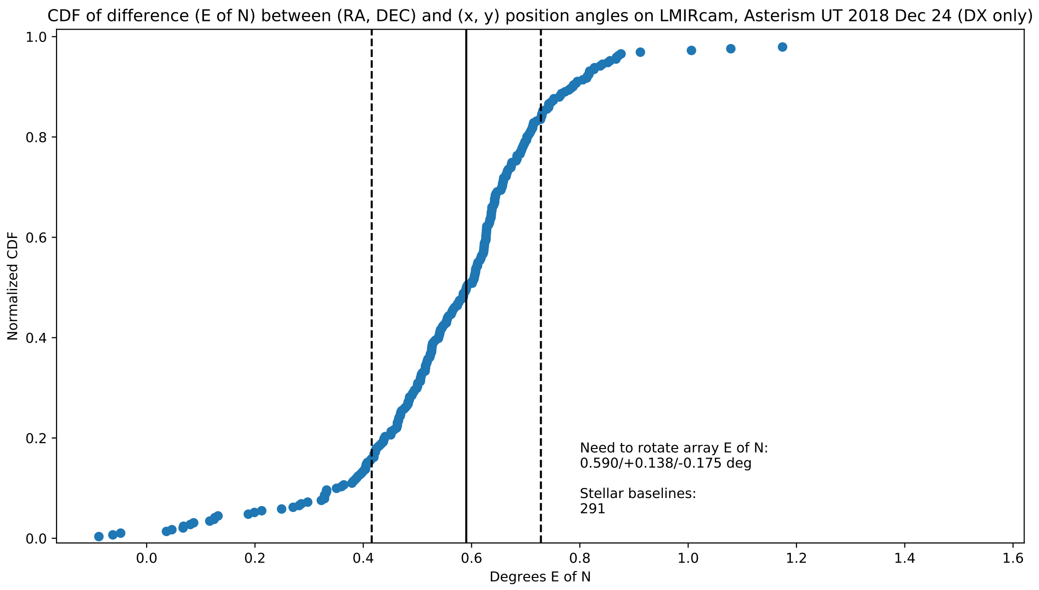


Plate scale of LMIRcam, Asterism UT 2018 Dec 24 (SX only)
Plate scale:
10.652/+0.034/-0.029 mas/pix
Stellar baselines:
303
Normalized CDF
PS (mas/pix)
0.0
0.2
0.4
0.6
0.8
1.0
10.4
10.5
10.6
10.7
10.8
10.9


CDF of difference (E of N) between (RA, DEC) and (x, y) position angles on LMIRcam, Asterism UT 2018 Dec 24 (DX only)
Need to rotate array E of N:
0.590/+0.138/-0.175 deg
Stellar baselines:
291
Normalized CDF
Degrees E of N
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6

CHAPTER 6

Acknowledgements

Jay Anderson of STScI provided assistance for using M92 astrometry. The original imaging data for this is from the ACS Globular Cluster Treasury survey (PI-Sarajedini), and is available from the MAST archive.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search