



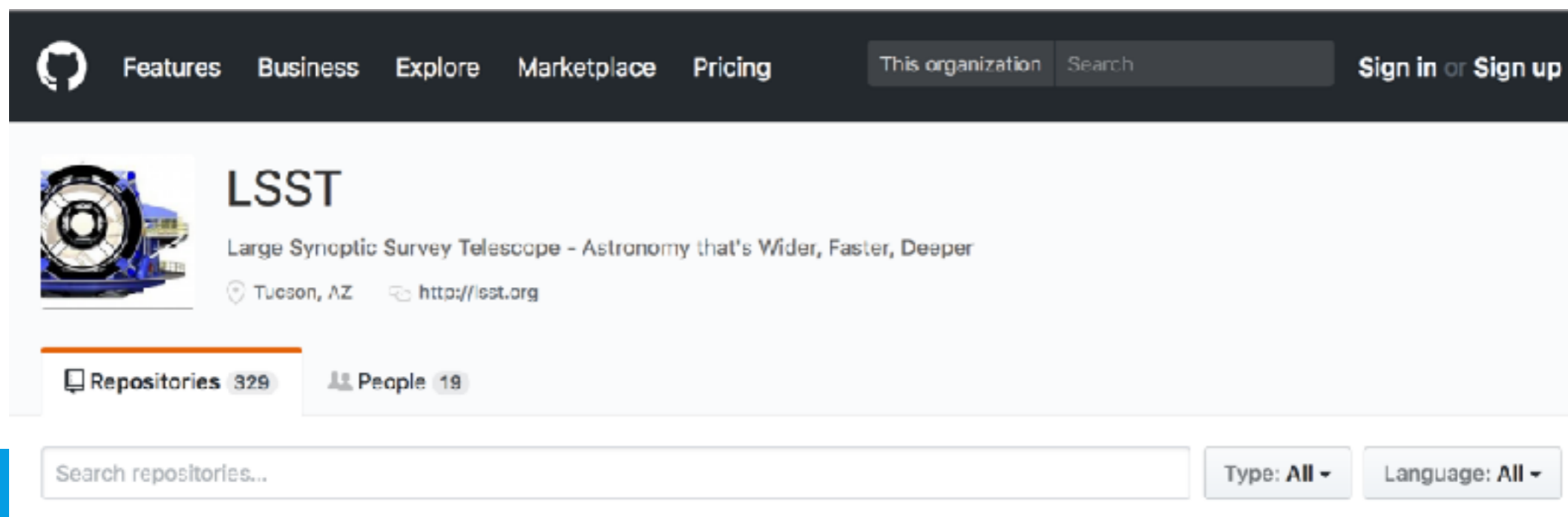
The
University
Of
Sheffield.

Using and adapting the LSST processing pipeline

Lydia Makrygianni & James Mullaney
+ The GOTO Collaboration

What is the LSST processing pipeline (the Stack)?

- It will deliver LSST's data products:
 - **Prompt:** Nightly processing; sources that have changed in brightness/position; catalogues from difference imaging.
 - **Data Release:** Annual release of coherent processing of entire dataset to date; fluxes, shapes, variability, light curve description.
- Written in:
 - Python (high level “calling” scripts)
 - C++ (lower level calculations)
- Designed to be a standard processing pipeline for other wide-field surveys.



The image shows a screenshot of the LSST organization page on GitHub. The page features a dark navigation bar with links for Features, Business, Explore, Marketplace, and Pricing. A search bar and a 'Sign in or Sign up' button are also present. The main content area displays the LSST organization profile, including a profile picture of a telescope, the name 'LSST', and the tagline 'Large Synoptic Survey Telescope - Astronomy that's Wider, Faster, Deeper'. Below this, there are statistics for 'Repositories 329' and 'People 19'. At the bottom, there is a search bar for repositories and filters for 'Type: All' and 'Language: All'.

Our target audiences

- The Typical User:
 - is interested in the standard outputs of the LSST Stack.
- The Advanced User:
 - The standard outputs do not meet your scientific requirements; you want to reconfigure the Stack to process LSST data to meet your needs.
- The Super Advanced User:
 - wants to use the Stack to process data from other facilities for, e.g., consistent incorporation into LSST database.

Our experience with the LSST stack

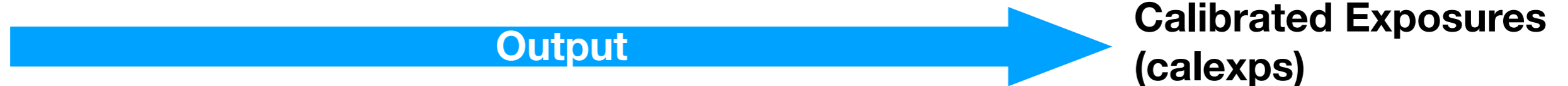
- GOTO:
Gravitational-wave Optical Transient Observatory*
- Currently 4 x 40cm, 5 sq. deg FOV `scopes on common mount.
- Conduct high-cadence survey to 20th mag & follow-up LIGO triggers.
- Total FOV, cadence and desired outputs similar to LSST.
- In addition to in-house pipeline, we're also using the Stack to process GOTO data:
 - created static coadded images;
 - coadds used as reference for nightly forced photometry.



Processes and outputs; Standard Use

Processes and outputs; Standard Use

- Organises raw input frames (database of image type, date etc);
- Instrument signature removal (i.e., bias, dark, flat correct);
- Background subtraction;
- PSF modelling;
- Astrometric and photometric calibration.



Processes and outputs; Standard Use

- Organises raw input frames (database of image type, date etc);
- Instrument signature removal (i.e., bias, dark, flat correct);
- Background subtraction;
- PSF modelling;
- Astrometric and photometric calibration.

Output



**Calibrated Exposures
(calexps)**

- Source detection;
- Photometry (aperture, PSF, Gaussian, Kron, CModel, deVauc., etc.);
- Shape measurement;

Output



**Source catalogues
(src)**

Processes and outputs; Standard Use

- Organises raw input frames (database of image type, date etc);
- Instrument signature removal (i.e., bias, dark, flat correct);
- Background subtraction;
- PSF modelling;
- Astrometric and photometric calibration.

Output

**Calibrated Exposures
(calexps)**

- Source detection;
- Photometry (aperture, PSF, Guassian, Kron, CModel, deVauc., etc.);
- Shape measurement;

Output

**Source catalogues
(src)**

- Image alignment, warping, coaddition;
- Deep detection, photometry and catalogue merging;

Outputs

**Coadded exposures
and merged catalogues**

Processes and outputs; Standard Use

- Organises raw input frames (database of image type, date etc);
- Instrument signature removal (i.e., bias, dark, flat correct);
- Background subtraction;
- PSF modelling;
- Astrometric and photometric calibration.

Output

Calibrated Exposures
(calexps)

- Source detection;
- Photometry (aperture, PSF, Guassian, Kron, CModel, deVauc., etc.);
- Shape measurement;

Output

Source catalogues
(src)

- Image alignment, warping, coaddition;
- Deep detection, photometry and catalogue merging;

Outputs

Coadded exposures
and merged catalogues

- Forced photometry, difference imaging.

Outputs

Nightly forced
photometry and diffims.

PSF modelling

- After ISR, the Stack performs a high-level source detection on each input image and attempts to identify point sources;
- It then uses these point sources to model the PSF across each input image and records summary statistics of the PSF:
 - default model uses **Principal Component Analysis** (PCA), but other modules can be implemented (e.g., **PSFEx**).
- This gives an indication of the image quality in terms of seeing.

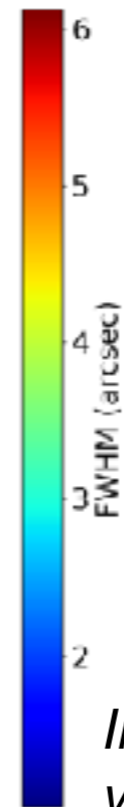
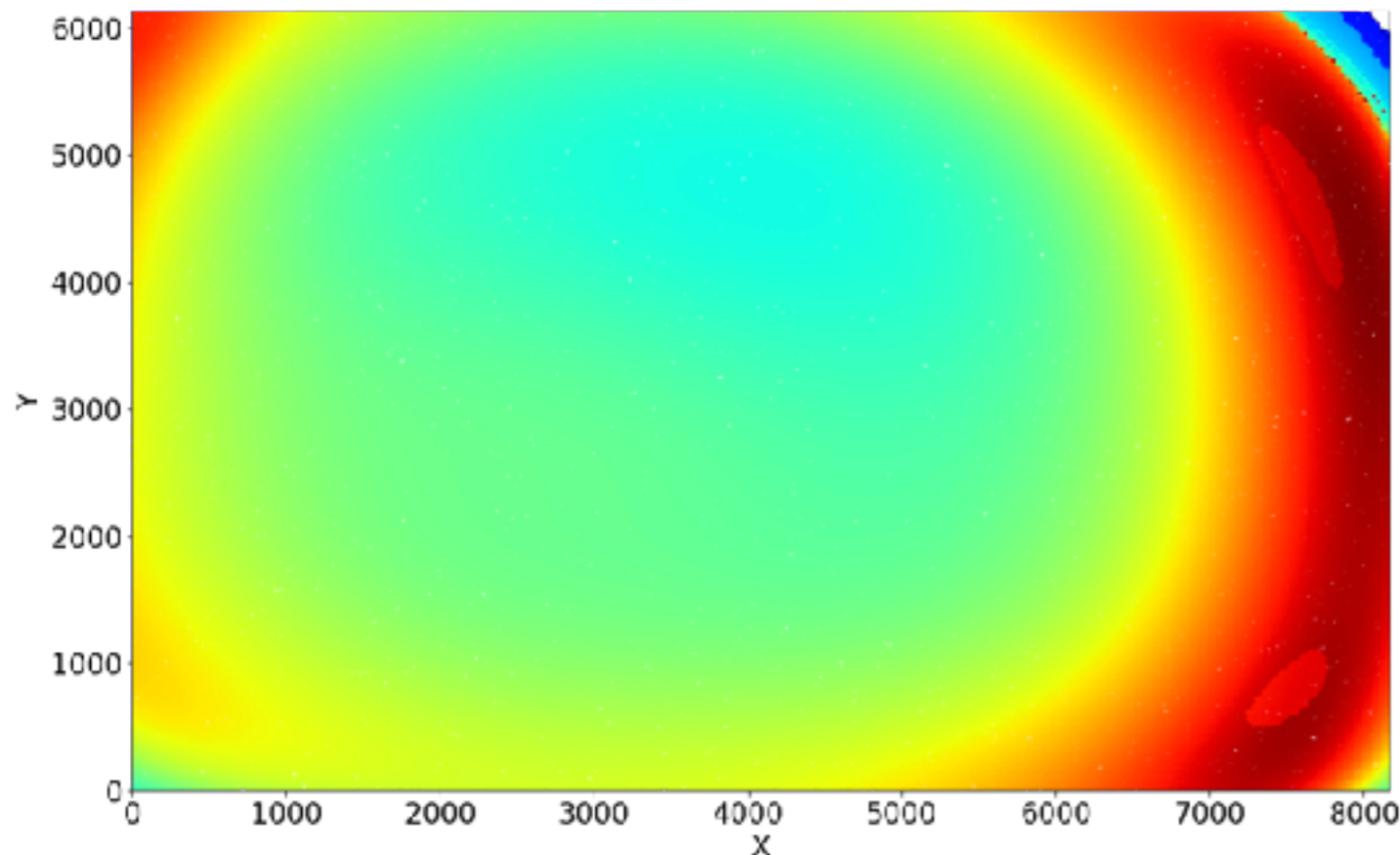


Image showing the PSF FWHM variation over a GOTO exposure.

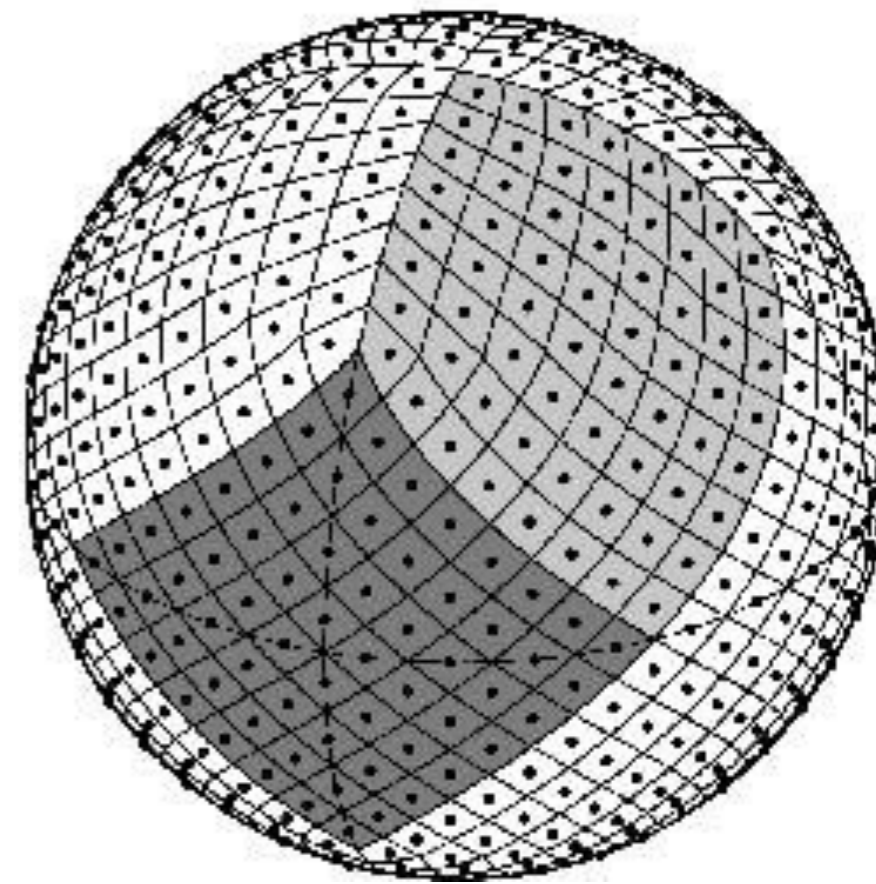
Coaddition

The depth of the LSST survey is achieved via the coaddition of multiple epochs of data.

Prior to coaddition, the Stack re-projects caexp images onto a single sky map (e.g., HEALPix).

Involves warping each caexp onto the sky map (CPU-intensive & slow).

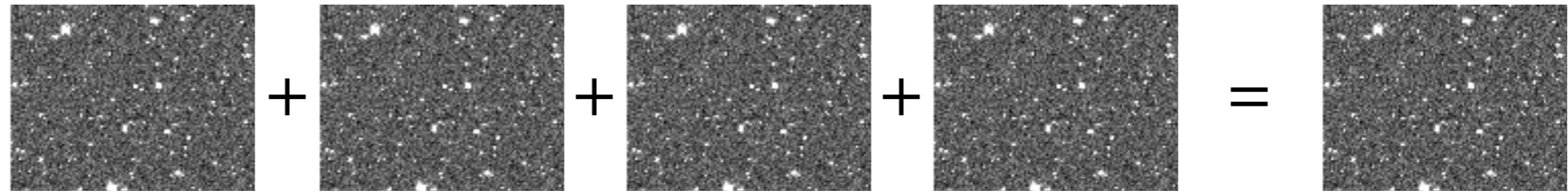
However, once warped, any combination of exposures can be combined to produce a coadd (fast).



HEALPix projection

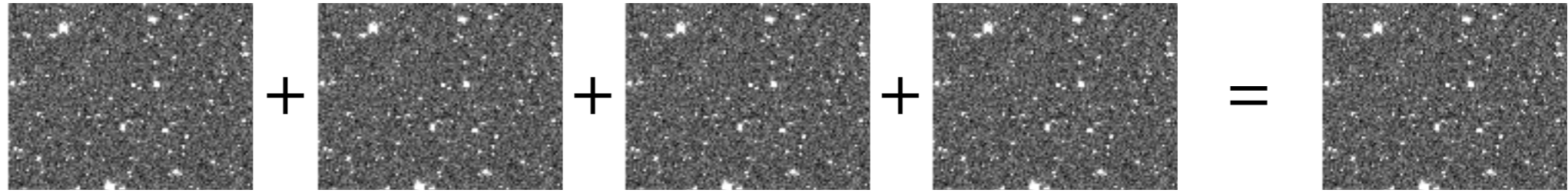
Quick, flexible coadds

For deep coadd:
Add everything
(e.g., low surface
brightness, high-z)

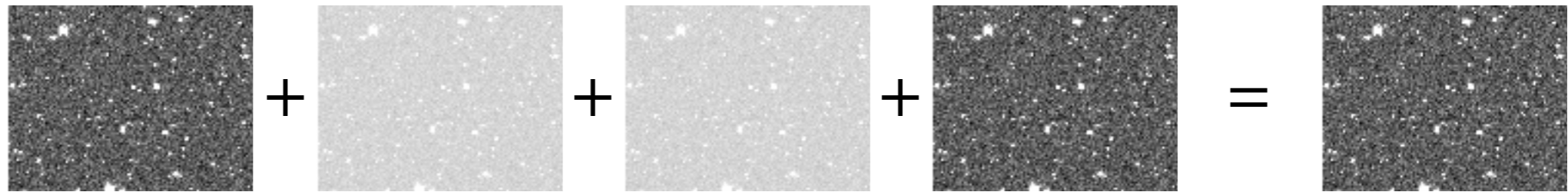


Quick, flexible coadds

For deep coadd:
Add everything
(e.g., low surface
brightness, high-z)

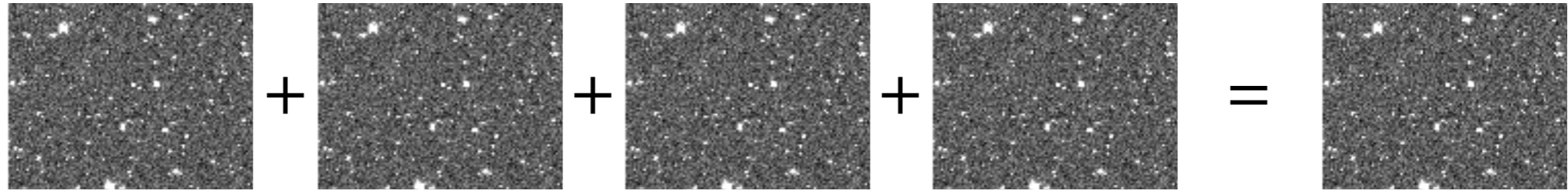


For high resolution:
Select best PSF
(e.g., lensing,
crowded fields)

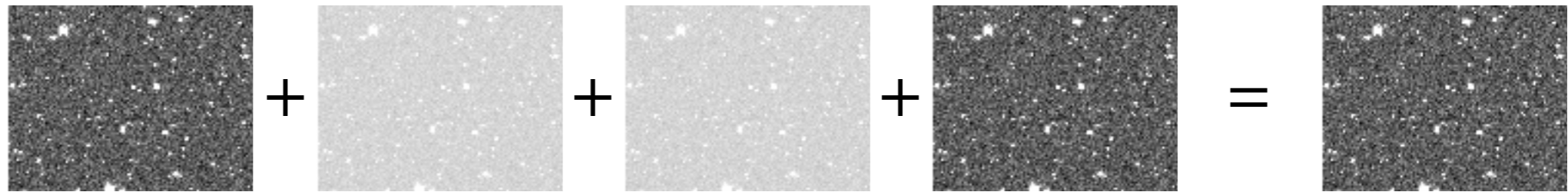


Quick, flexible coadds

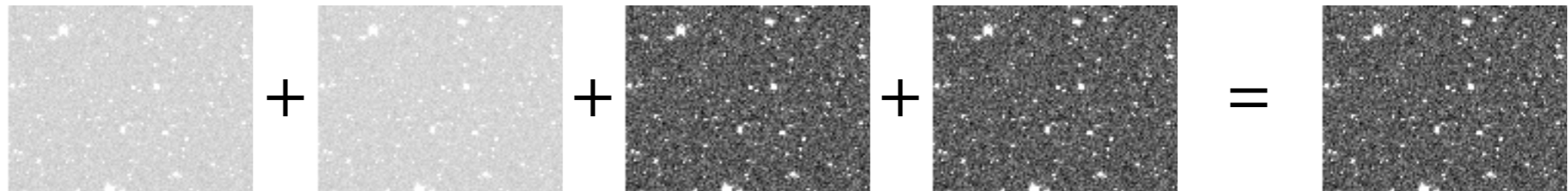
For deep coadd:
Add everything
(e.g., low surface
brightness, high-z)



For high resolution:
Select best PSF
(e.g., lensing,
crowded fields)

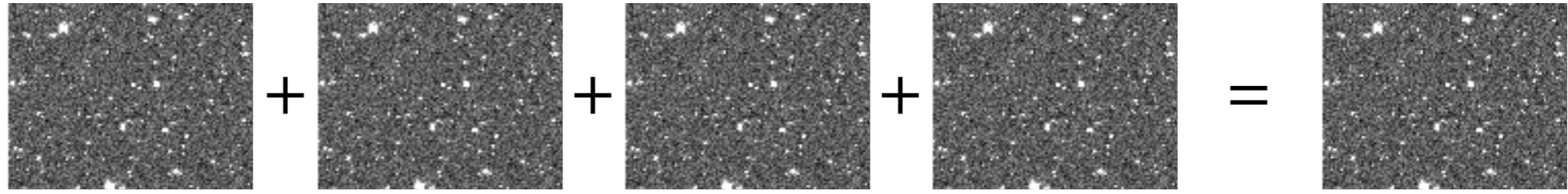


To search for faint
changes
Select a date range

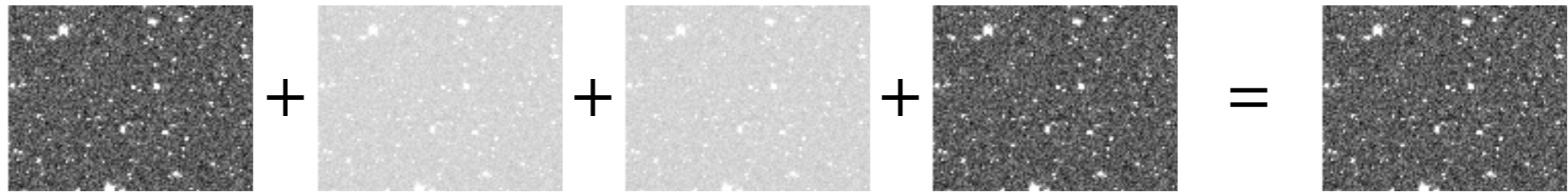


Quick, flexible coadds

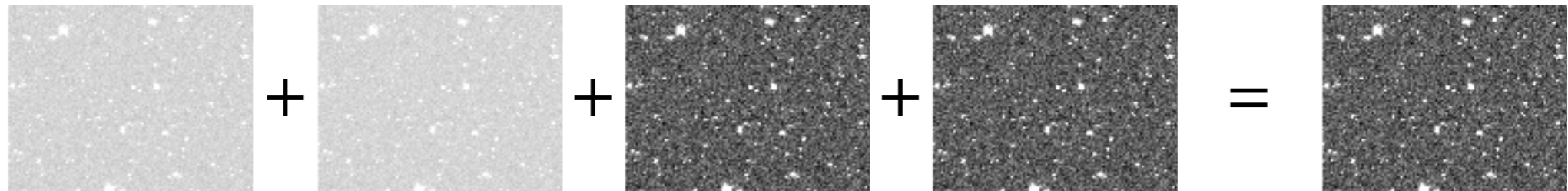
For deep coadd:
Add everything
(e.g., low surface
brightness, high-z)



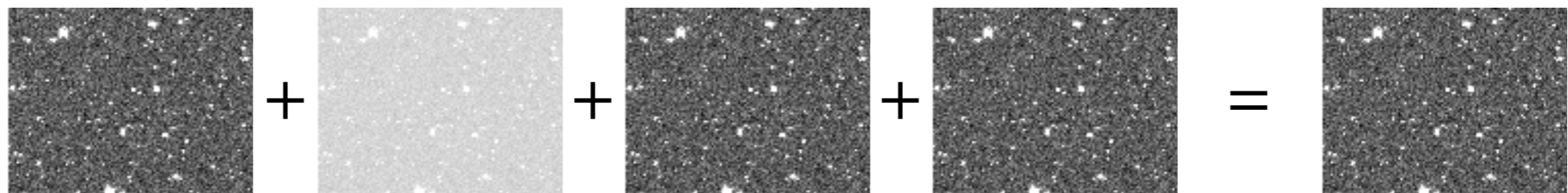
For high resolution:
Select best PSF
(e.g., lensing,
crowded fields)



To search for faint
changes
Select a date range

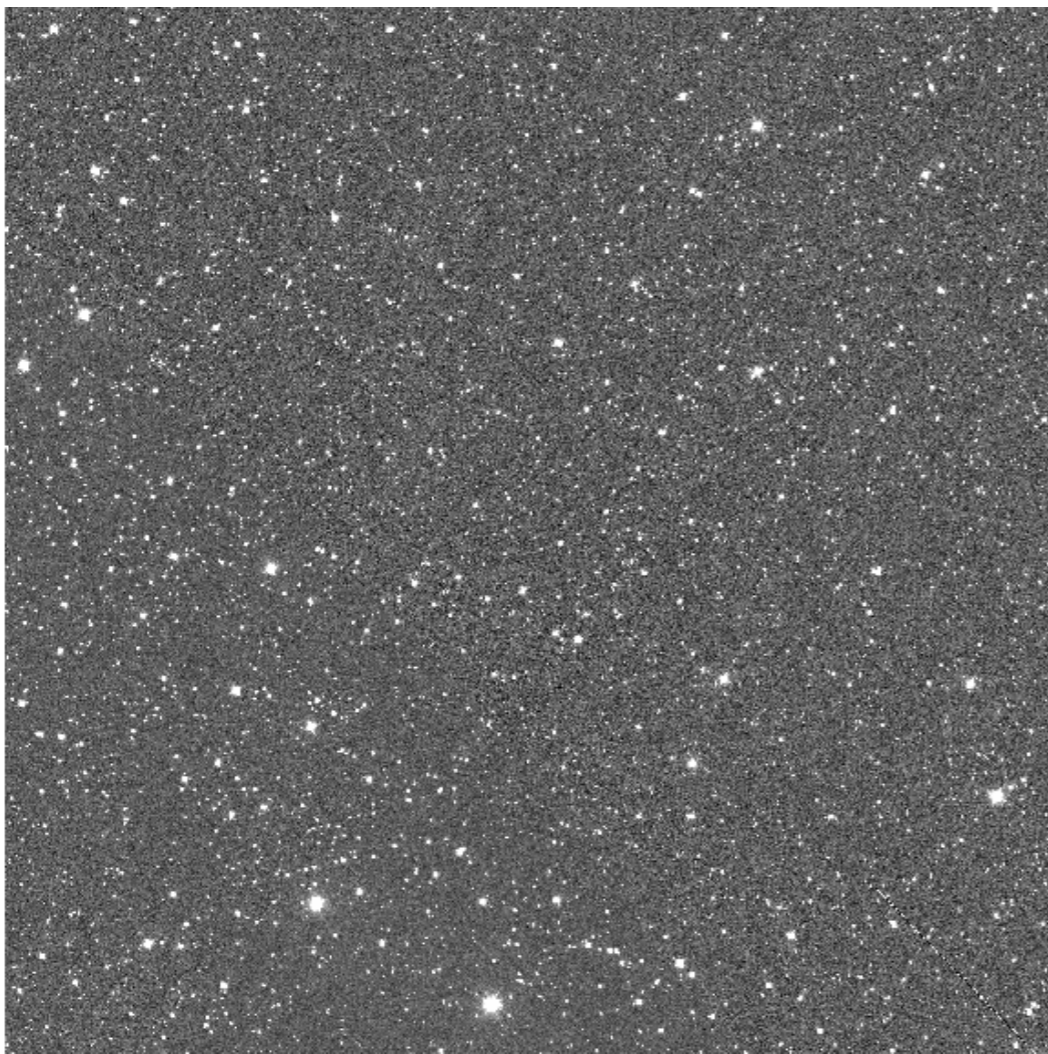


“Insert science here”:

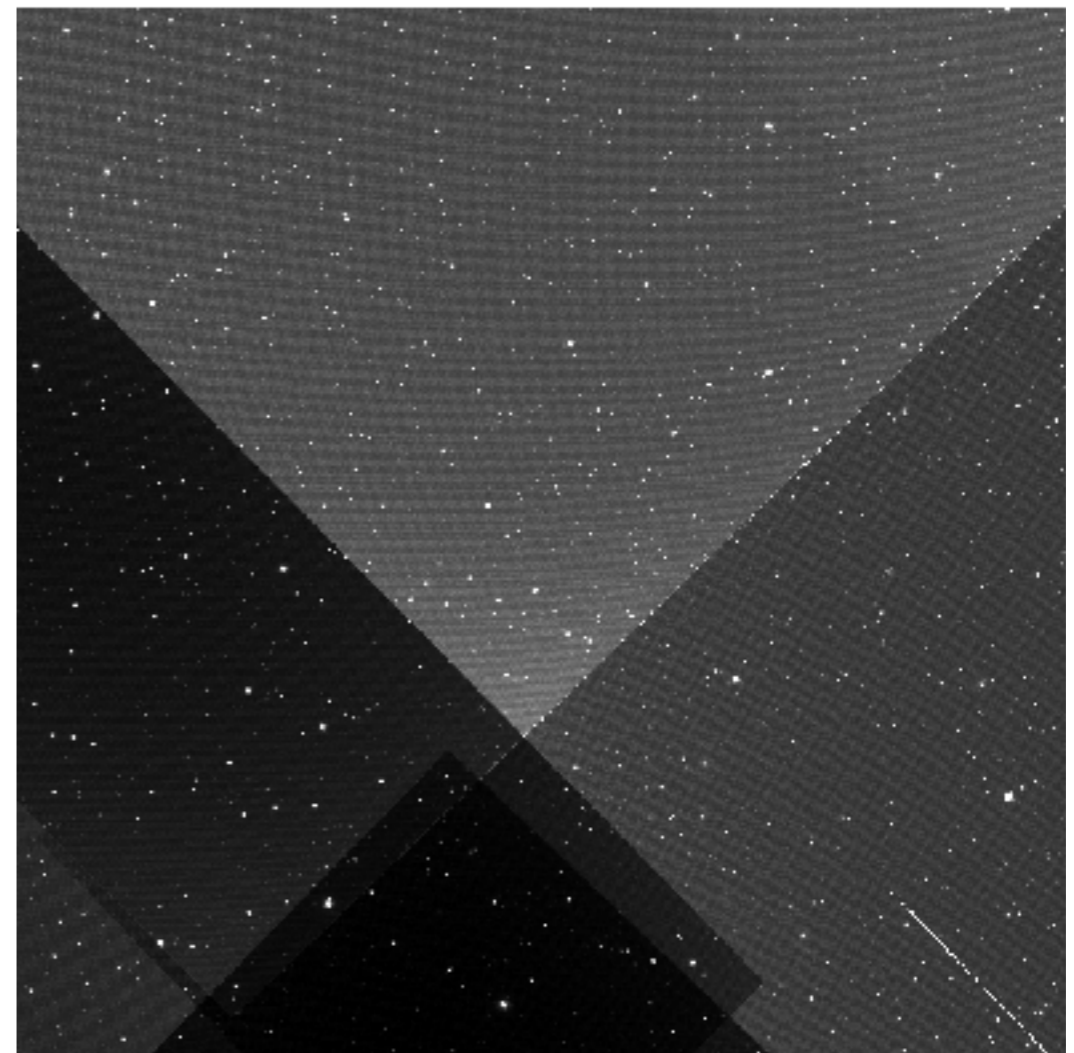


Coadd bookkeeping

...and each warped science image and coadd comes with its own variance image to ensure errors are propagated correctly...



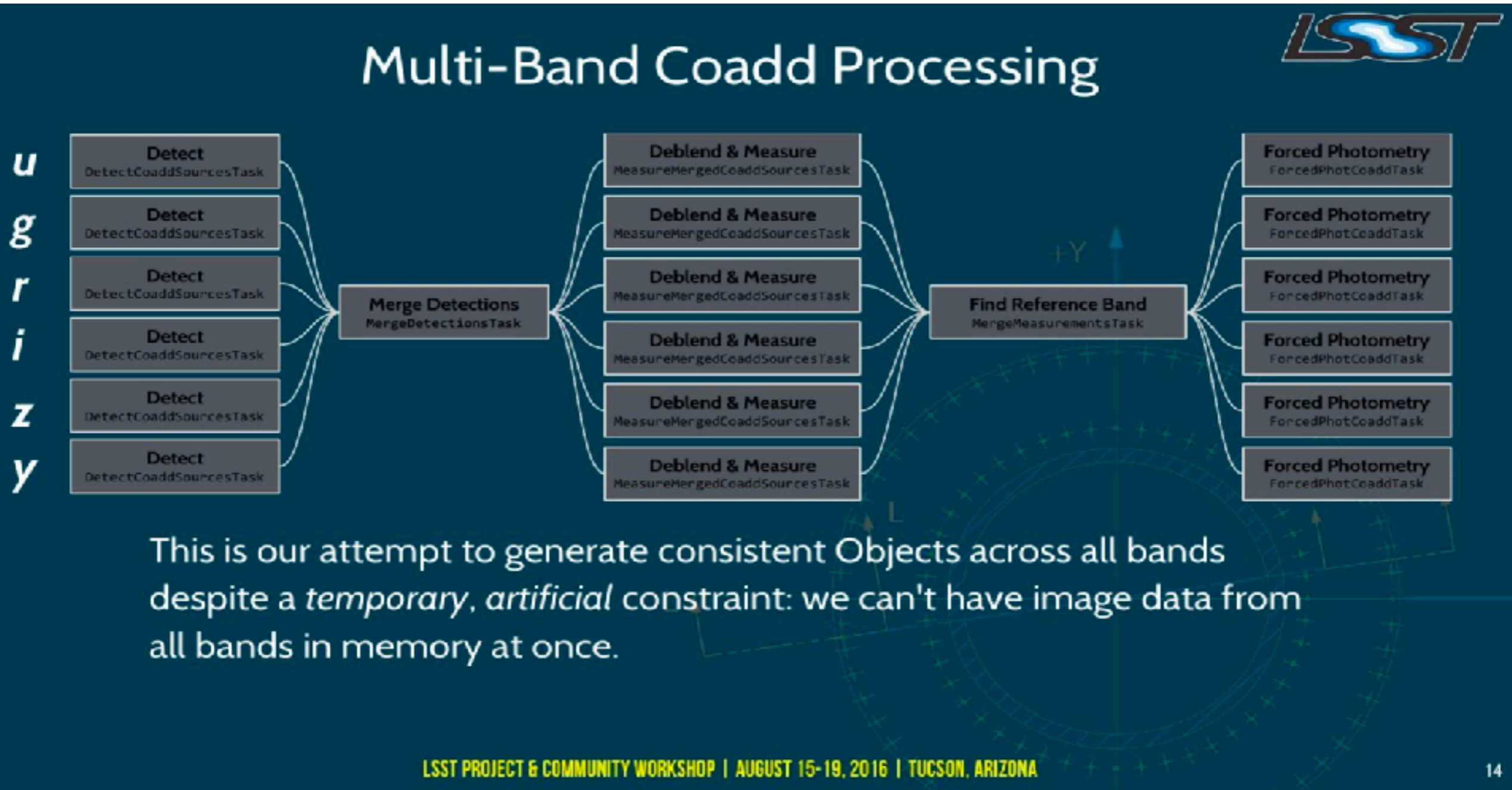
Warped and coadded GOTO science image



Accompanying variance image

Merged catalogues

Stack runs source detection on each coadd in each band, merges those detections, attempts to deblend and measure based on all detections in all bands, then selects the optimum band for each source from which to build a reference for forced photometry.



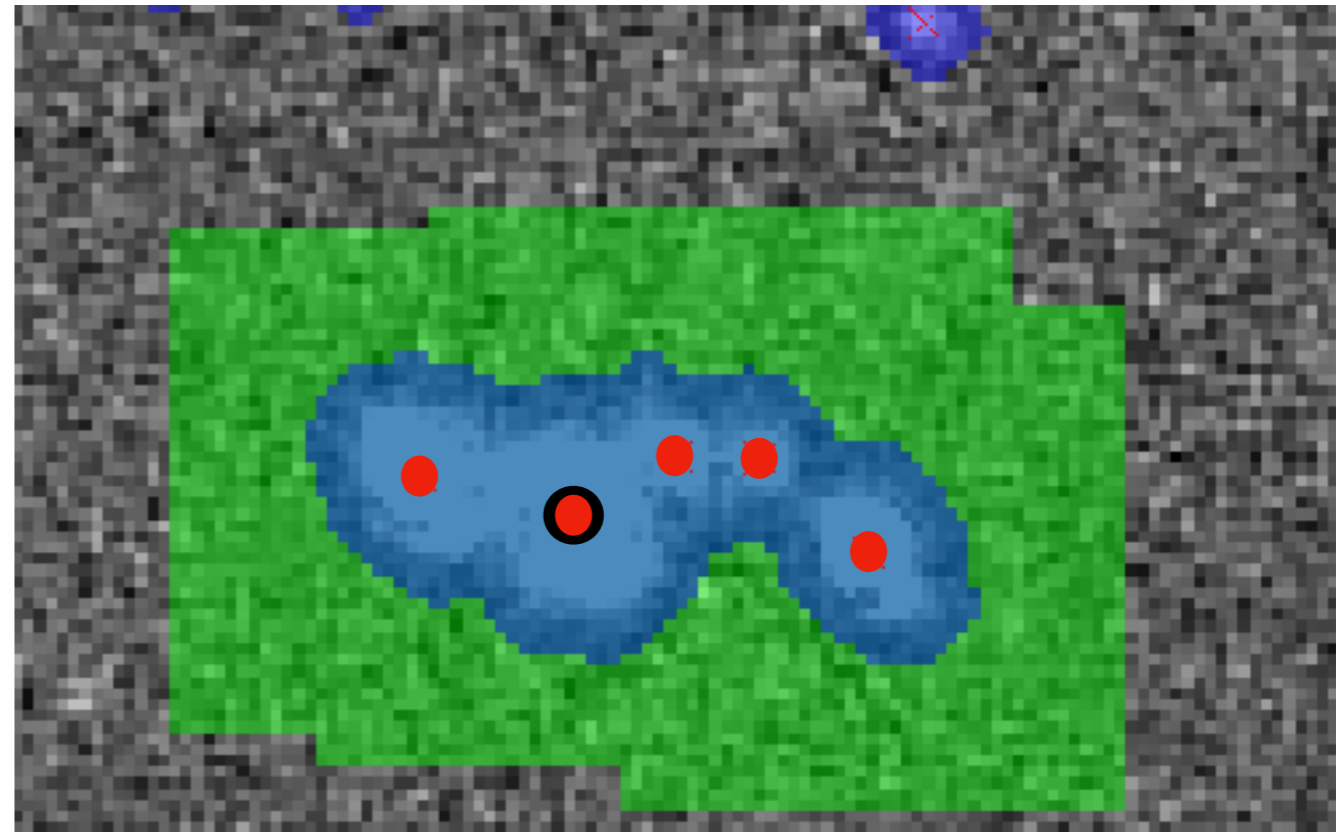
Reconfiguring the LSST stack: A case study

Deblending

Due to GOTO's increased pixel size relative to the LSST, the Stack's default deblending routine was sub-optimal.

Every module in the Stack comes with its own set of config parameters, which can be overridden...

...but some modules contain thousands of parameters!



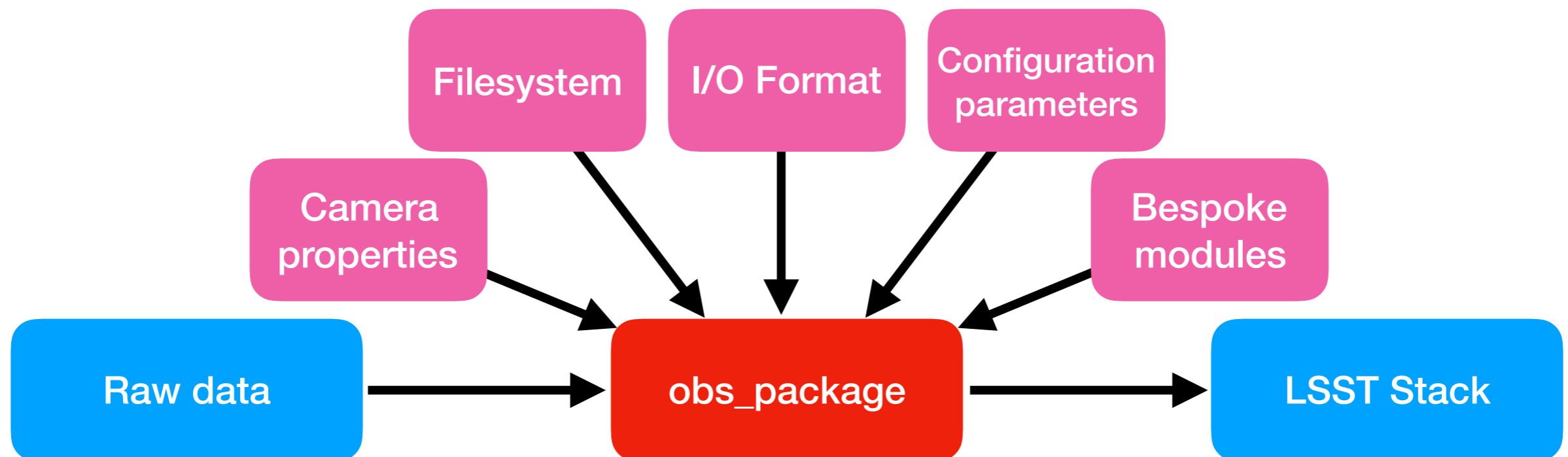
```
#This is needed as we use 'run' instead of 'visit':  
from lsst.obs.goto.makeGotoCoaddTempExp import GotoMakeCoaddTempExpTask  
config.makeCoaddTempExp.retarget(GotoMakeCoaddTempExpTask)  
  
#Uncommentn this if we want to select frames based on PSF:  
#from lsst.pipe.tasks.selectImages import PsfWcsSelectImagesTask  
#config.assembleCoadd.select.retarget(PsfWcsSelectImagesTask)  
  
#Detection can be slow, especially if measuring a lot of parameters.  
#Can turn on when creating deep catalogues.  
config.doDetection = False  
config.detectCoaddSources.detection.minPixels = 10  
config.detectCoaddSources.detection.background.binSize=8192  
config.detectCoaddSources.detection.tempLocalBackground.binSize=32  
config.detectCoaddSources.detection.tempWideBackground.binSize=1024
```

Adapting the LSST stack to other surveys

Why bother?

- Consistent data products for ingestion into the LSST database.
- At present, the Stack isn't set up to take unformatted catalogues and use them for forced photometry, for example.

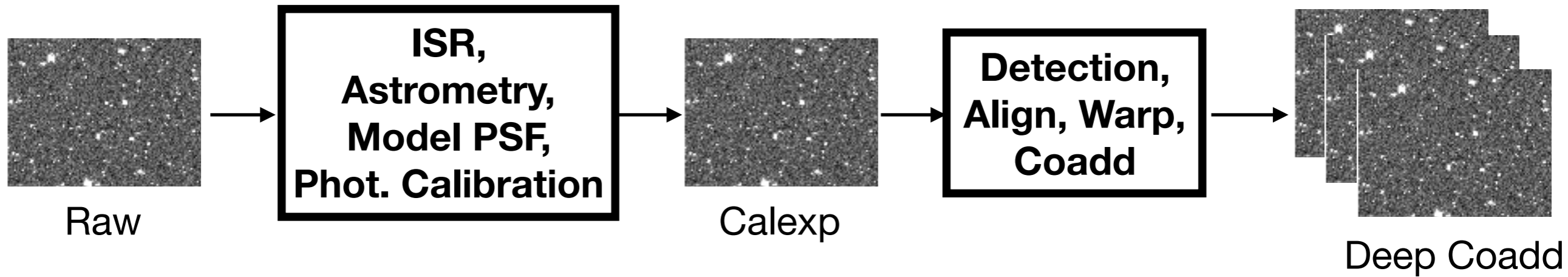
If you want to take your NIR survey, for example, and use it as a basis for forced photometry on LSST data, it'll need to be re-formatted or re-processed using the Stack.



Bespoke modules: A case study

GOTO's `singleVisitDriver.py`

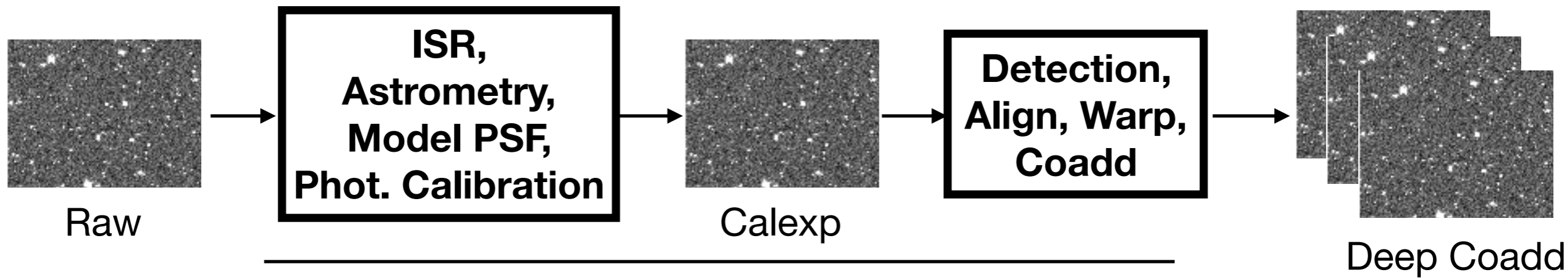
Default module
(`singleFrameDriver`)



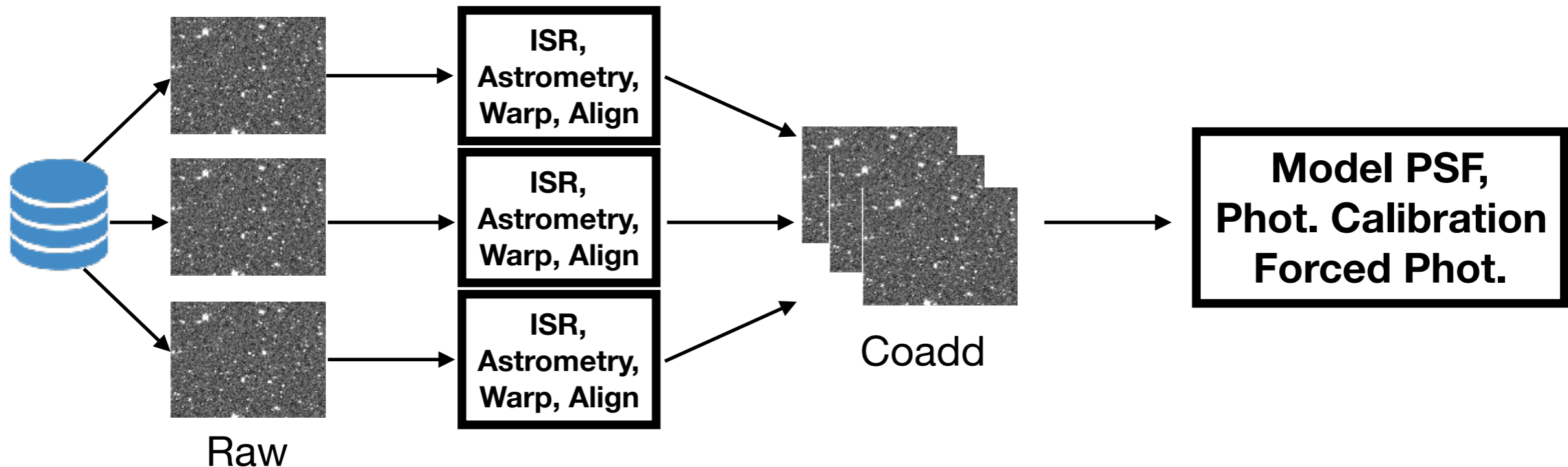
Bespoke modules: A case study

GOTO's `singleVisitDriver.py`

Default module
(`singleFrameDriver`)



Bespoke module
(`singleVisitDriver`)



Summary

- As one would expect, the LSST Stack will provide a huge variety of images and measured parameters.
- However, some questions may benefit from an adaptation of the stack.
- With well-catalogued image metadata (PSF, date), it will be possible to produce bespoke coadded data.
- Most aspects of the Stack can be easily reconfigured.
- And, with some effort, new modules can be written to perform bespoke tasks, including processing non-LSST data.