Transient Detection using LSST Stack

By: Juan Pablo Reyes Gómez
Supervisors: Dominique FOUCHEZ and Marcela HERNANDEZ HOYOS

Outline

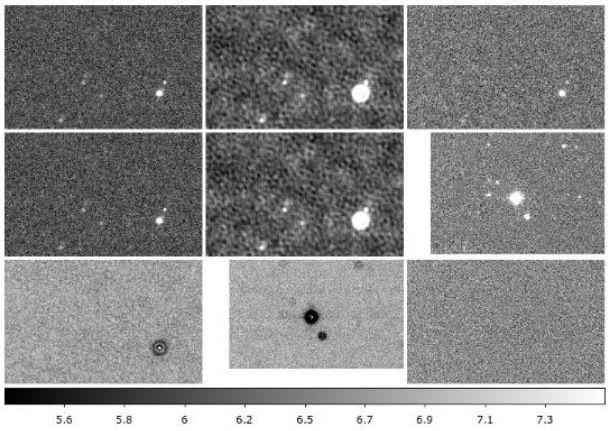
- Using the LSST DM Stack
  - Introduction
  - Understanding and modifying modules
  - Pipelines and Tasks
- Image Subtraction on the LSST DM Stack
  - Introduction
  - Pipelines and Tasks
- Tools and Utilities
  - Exposures and masks
  - Butler and byproducts
  - Visualization of results
  - Feedback, support and bug report
- Perspectives
- Conclusions

Using the LSST DM STACK: Basics

Using the LSST DM STACK: Introduction

In [18]:
```
subtractedExposure = afwImage.ExposureF(ScienceExposure, True)

if convolveTemplate:
    subtractedMaskedImage  = subtractedExposure.getMaskedImage()
    #subtractedMaskedImage -= results.matchedExposure.getMaskedImage()
    subtractedMaskedImage -= ReferenceExposure.getMaskedImage()

    print "ScienceExposure - MatchedExposure"
    displayImage(subtractedMaskedImage.getImage(), frame=frame, title="Science-Matched",path="figure4.pn
g");
    frame+=1

    subtractedMaskedImage -= results.backgroundModel

    print "ScienceExposure-background"
    displayImage(subtractedMaskedImage.getImage(), frame=frame, title="Science-Matched-background",path
="figure5.png")
    frame+=1

else:
    subtractedExposure.setMaskedImage(results.warpedExposure.getMaskedImage())
    subtractedMaskedImage  = subtractedExposure.getMaskedImage()
    subtractedMaskedImage -= results.matchedExposure.getMaskedImage()
    subtractedMaskedImage -= results.backgroundModel

    # Preserve polarity of differences
    subtractedMaskedImage *= -1
    #ds9.mtv(subtractedMaskedImage.getImage(), frame=frame, title="Polarity of differences preserved"); f
rame+=1
    #print "Polarity of differences preserved"
    #displayFigure("figure5.png")

    # Place back on native photometric scale
    subtractedMaskedImage /= results.psfMatchingKernel.computeImage(afwImage.ImageD(results.psfMatchingKe
rnel.getDimensions()), False)
    #ds9.mtv(subtractedMaskedImage.getImage(), frame=frame, title="Back to native Photometric scale"); fr
ame+=1
    #print "Back to native Photometric scale"
    #displayFigure("figure6.png")
```

ScienceExposure - MatchedExposure



- We have implemented many notebooks using libraries, tasks and utilities from stack.
- We have used v9.2, v11 and recently switched to lsstsw master, making it a "custom" version for our current work.
- We also have several stack modules modified for specifical needs on Image Subtraction.

Using the LSST DM STACK: Understanding and modifying modules

- Image subtraction uses tasks from several modules: **Obs_Cfht**, **Pipe_tasks**, **meas_algorithms** and **ip_diffim.**
- Most modifications and additions have to do with detection, measurement and input control.
- Some of these functions have been coded as Python functions to allow portability on different notebooks.

Using the LSST DM STACK: Pipelines and Tasks

- Image processing algorithms are implemented within tasks.
- The task inheritance from the class CmdLineTask allows usage as independent scripts or as instructions in Python.
- However, such tasks can be instantiated and run from Python with almost no effort.
- Tasks can be modularized for better comprehension, to experiment with results, or just to verify new steps.

Using the LSST DM STACK: Pipelines and Tasks

```
schema = afwTable.SourceTable.makeMinimalSchema()
table = afwTable.SourceTable.make(schema)
config = SourceDetectionTask.ConfigClass()

detectionTask = SourceDetectionTask(config=config, schema=schema)

table = afwTable.SourceTable.make(schema)
results = detectionTask.makeSourceCatalog(table=table,
              exposure=exposure , doSmooth=not False )
```

Example of task instantiation on Python

Image Subtraction on the LSST Stack

Image Subtraction on the LSST DM Stack: Introduction

- Image Subtraction on stack is implemented using the Alard-Lupton optimal subtraction (see SN presentation).
- The main principle is to project images onto patches, generate a coaddition as a template input and calculate the difference image.
- Transients, variable objects and other things are detected and classified using pre-existing tasks on Stack.

Image Subtraction on the LSST DM Stack: Pipeline and tasks

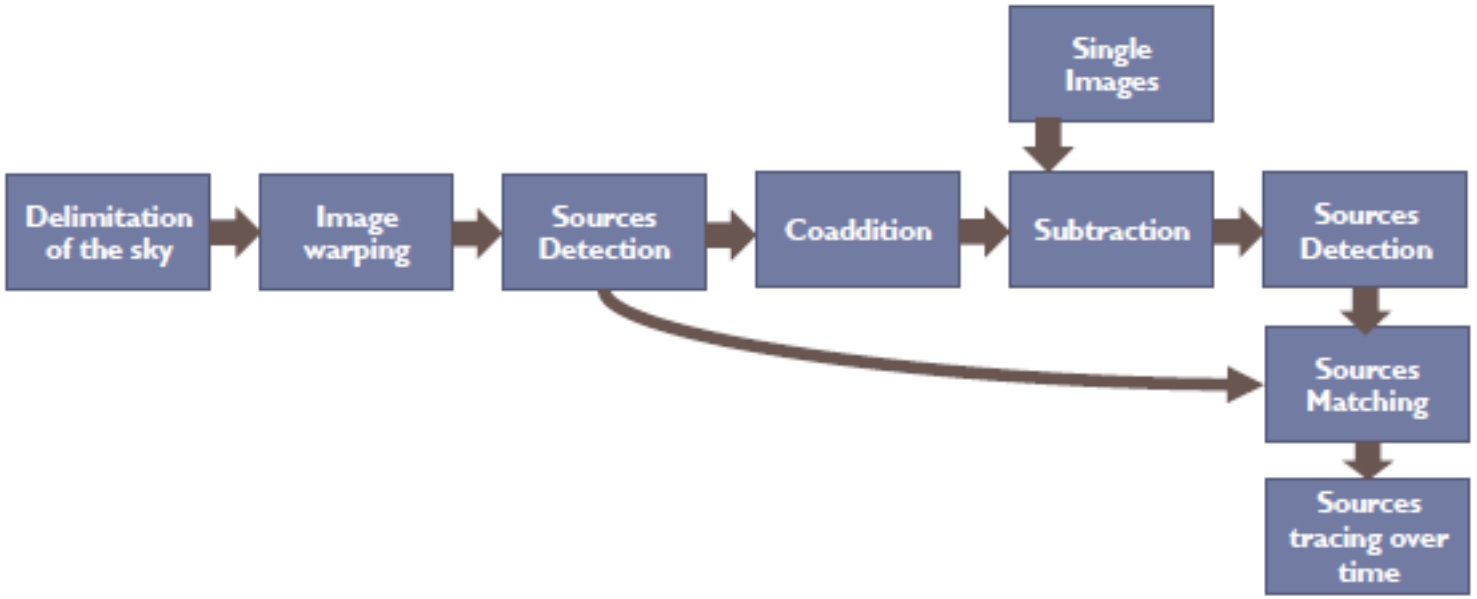


Image Subtraction on the LSST DM Stack: Pipeline and tasks

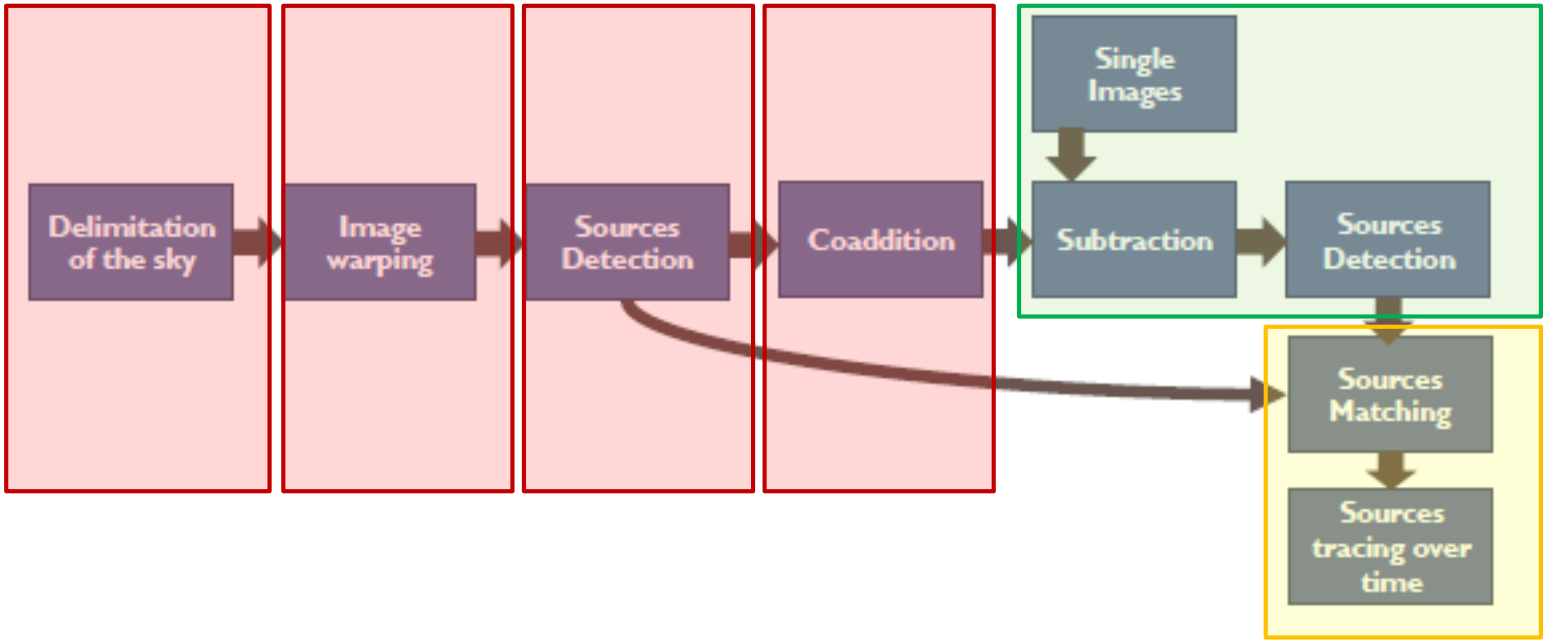


Image Subtraction on the LSST DM Stack: Pipeline and tasks

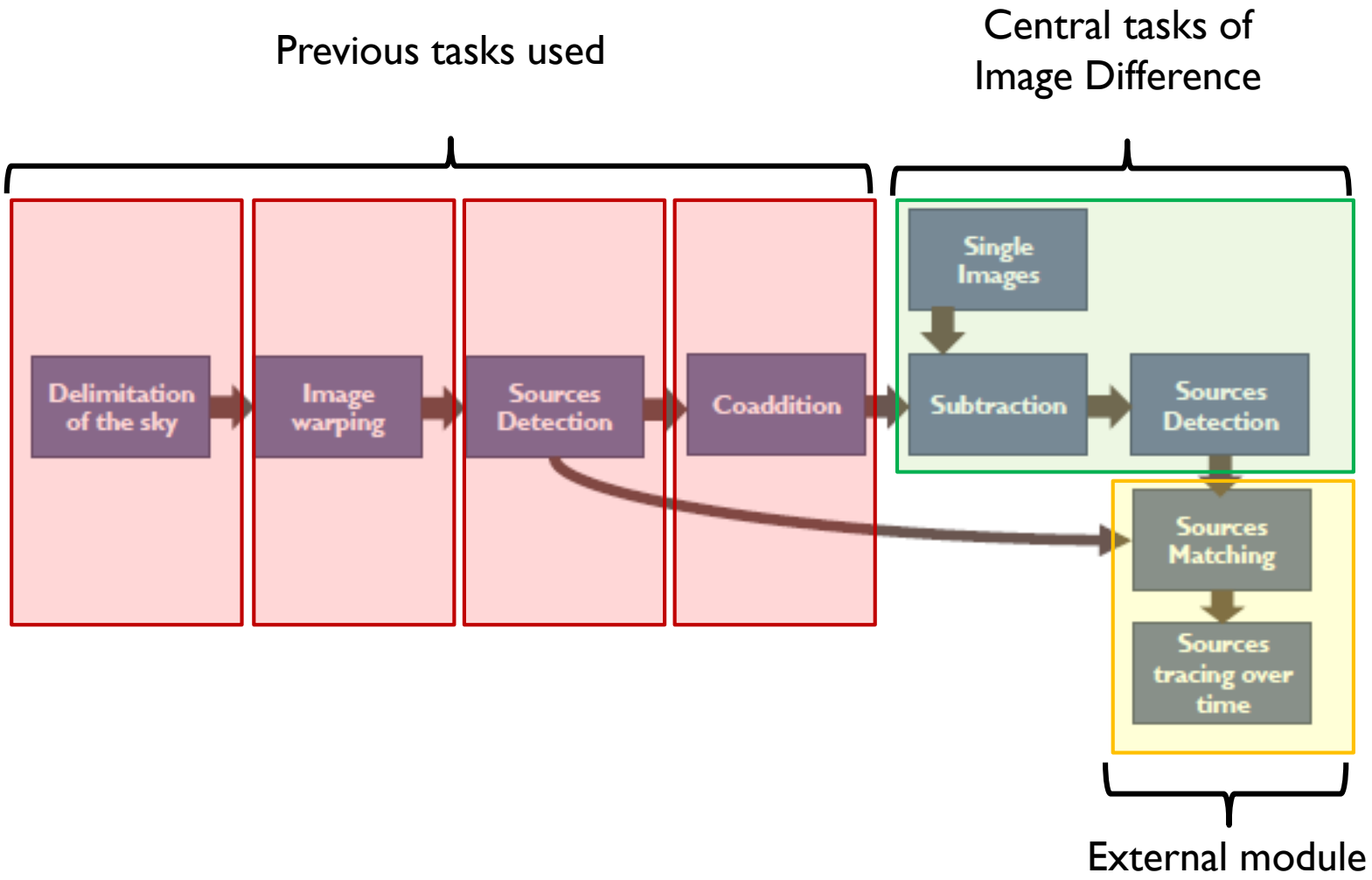


Image Subtraction on the LSST DM Stack: Pipeline and tasks

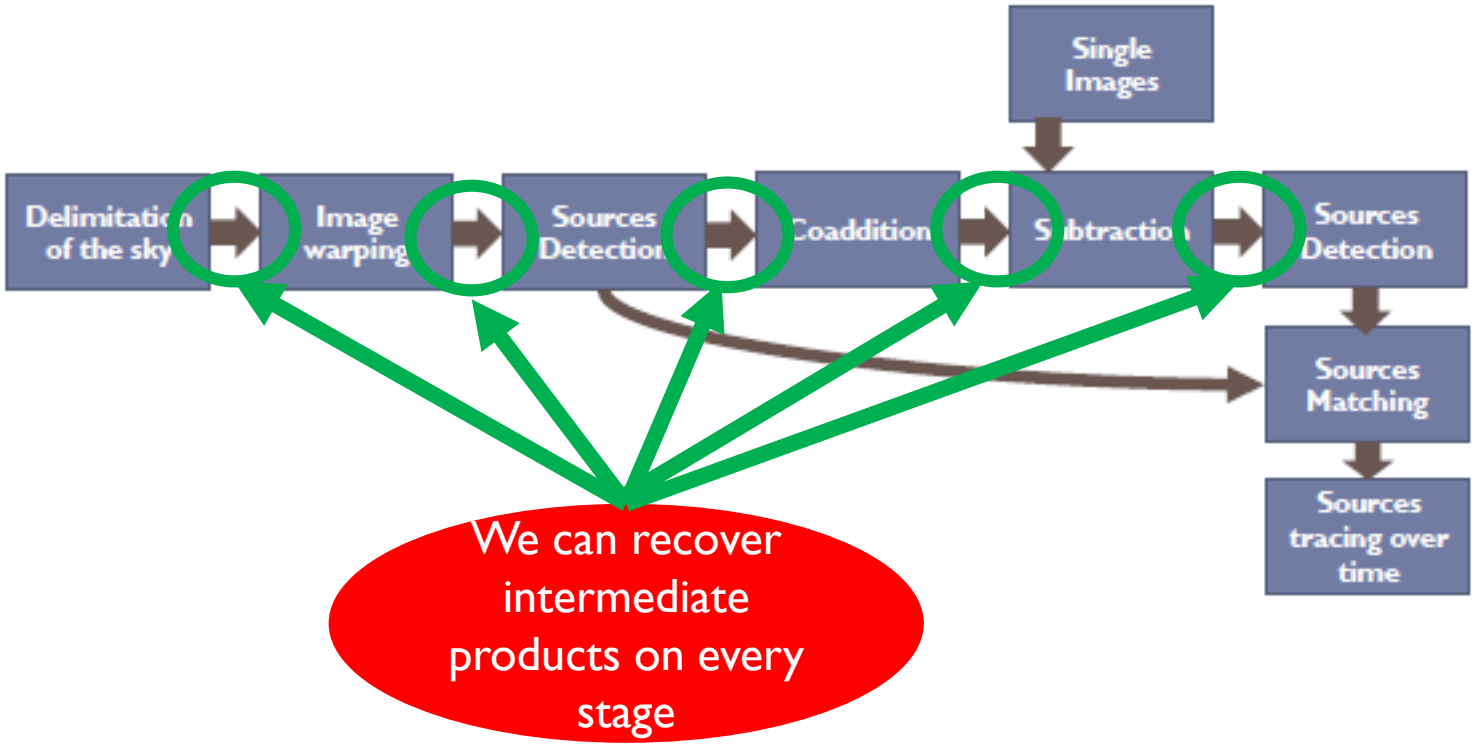


Image Subtraction on the LSST DM Stack: Pipeline and tasks

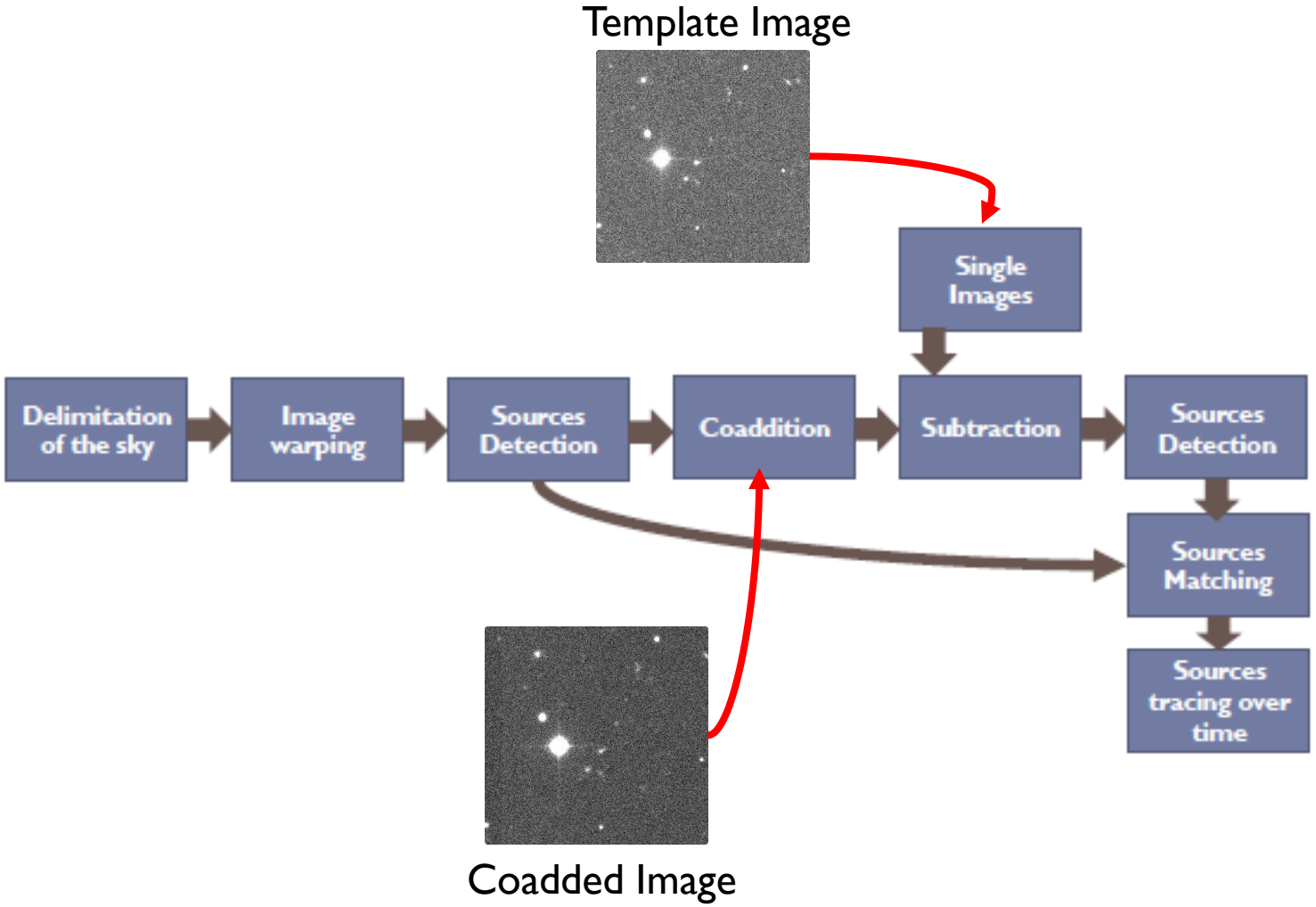


Image Subtraction on the LSST DM Stack: Pipeline and tasks

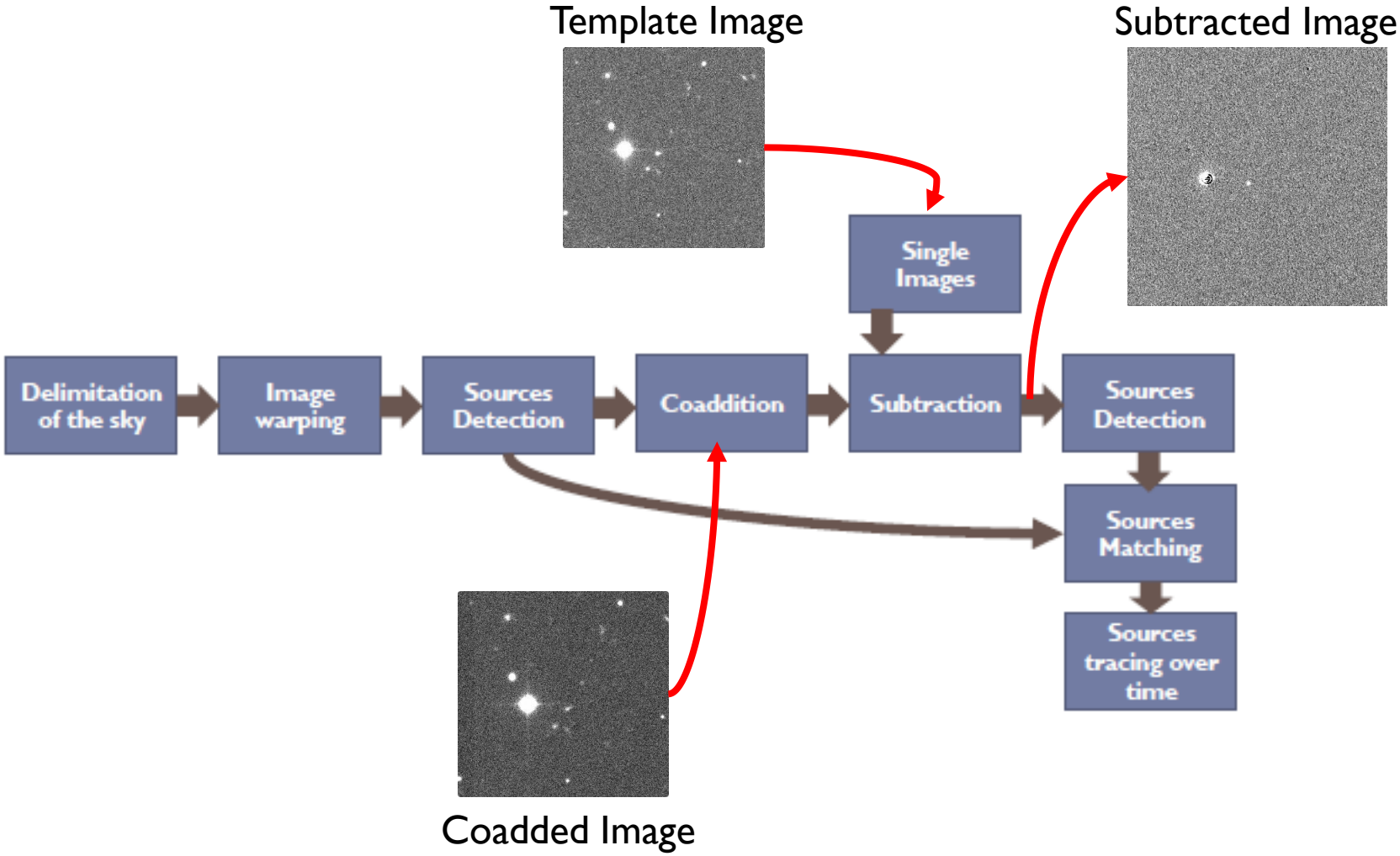


Image Subtraction on the LSST DM Stack: Pipeline and tasks

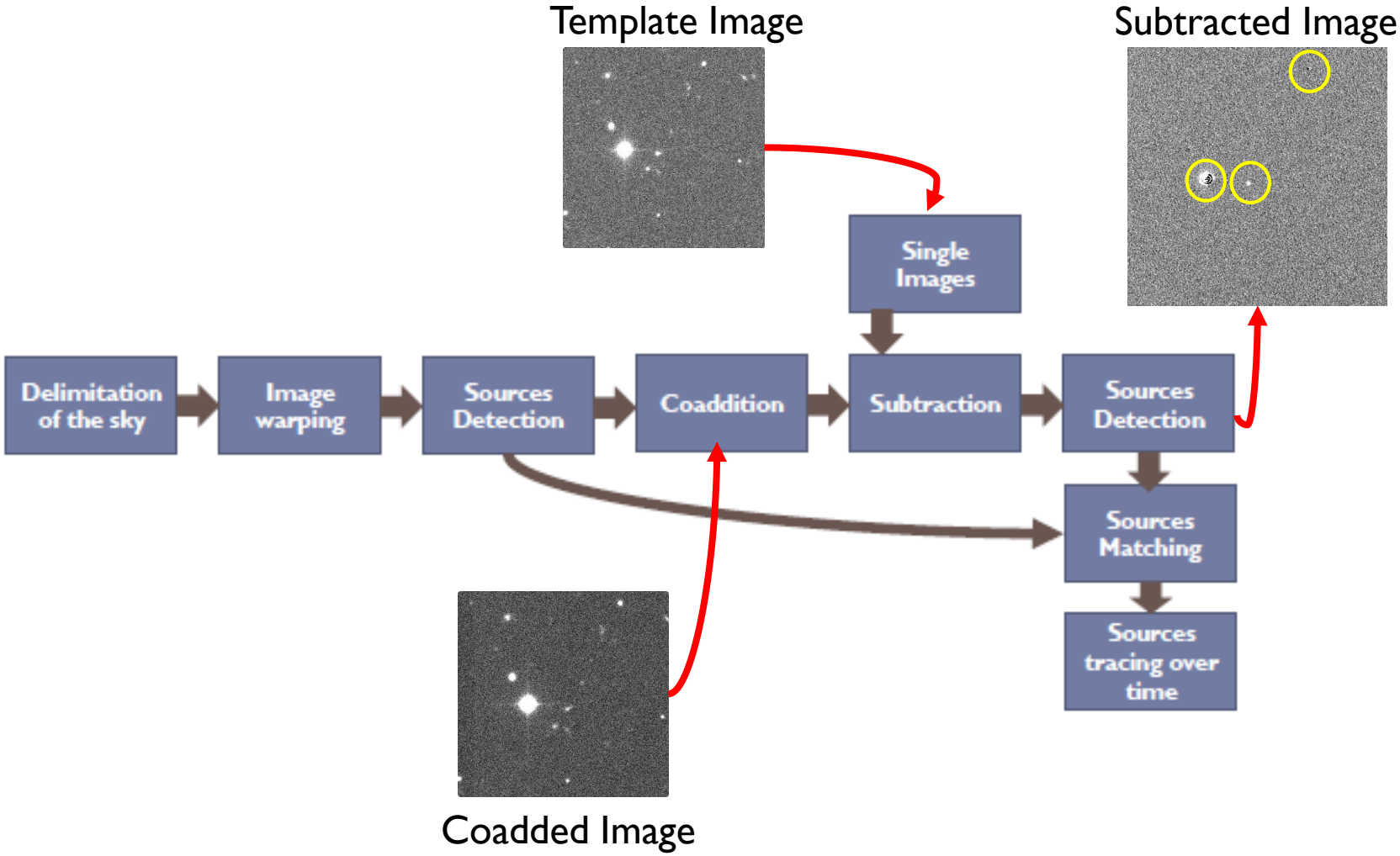


Image Subtraction on the LSST DM Stack: Pipeline and tasks

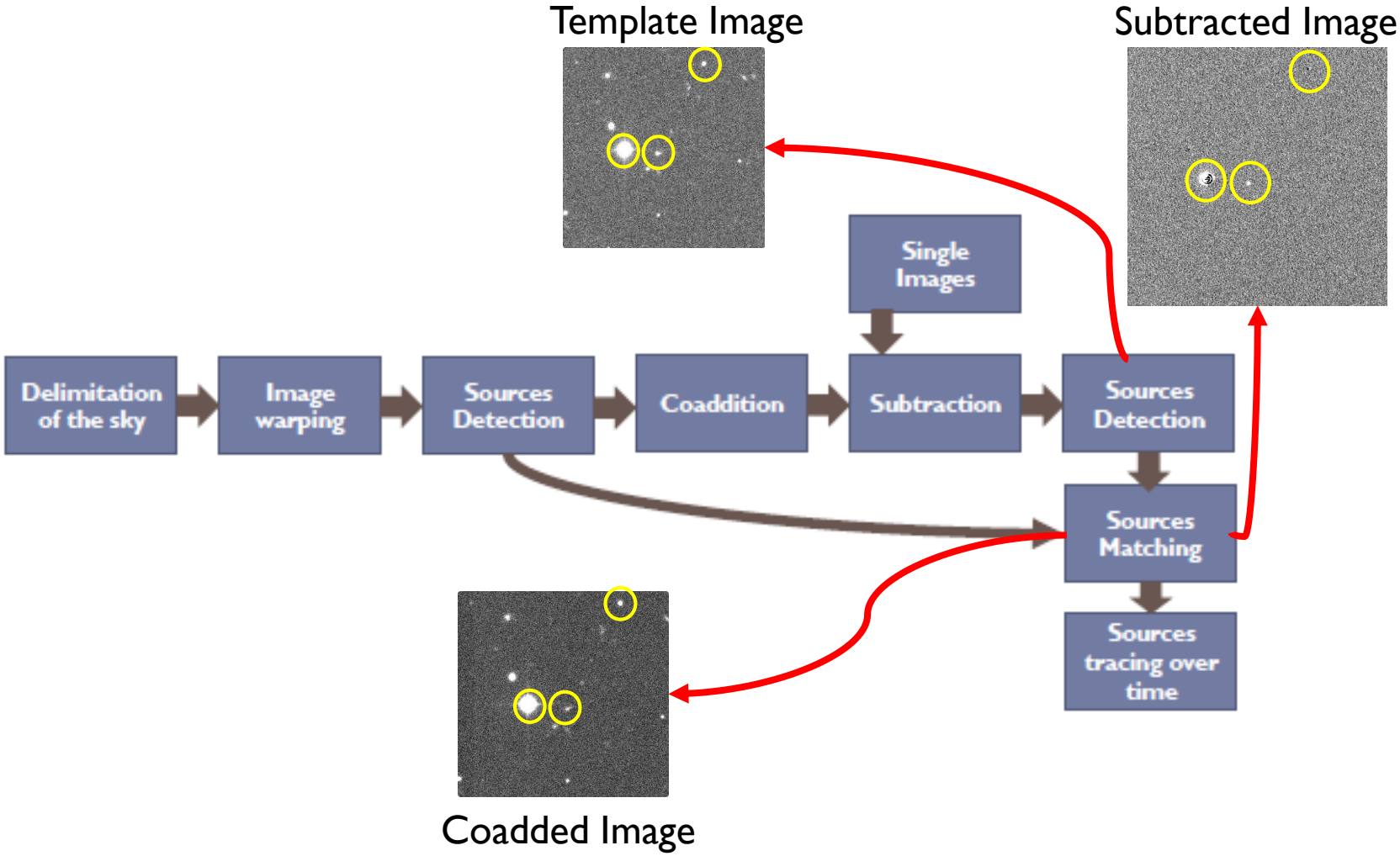


Image Subtraction on the LSST DM Stack: Pipeline and tasks

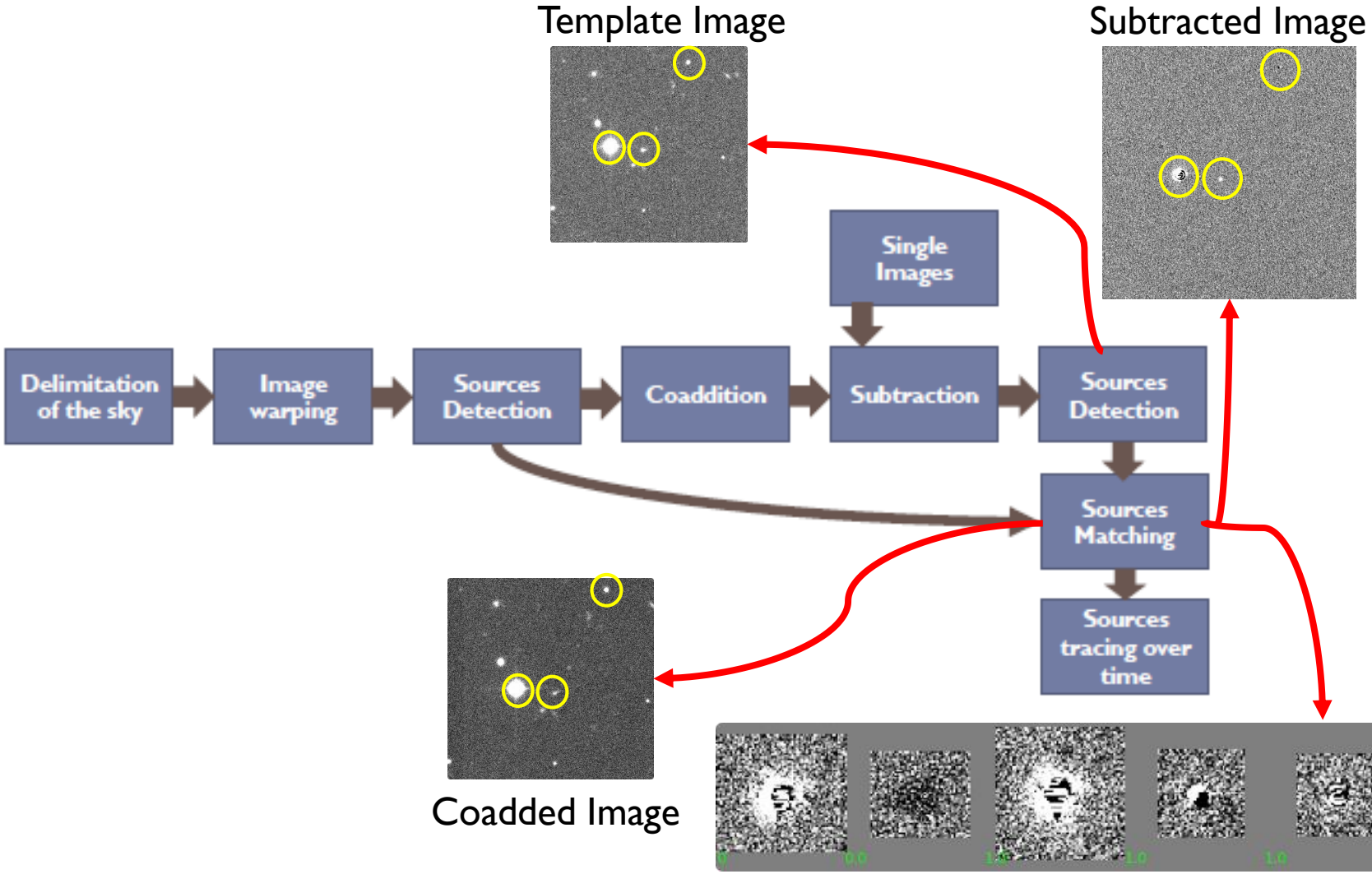
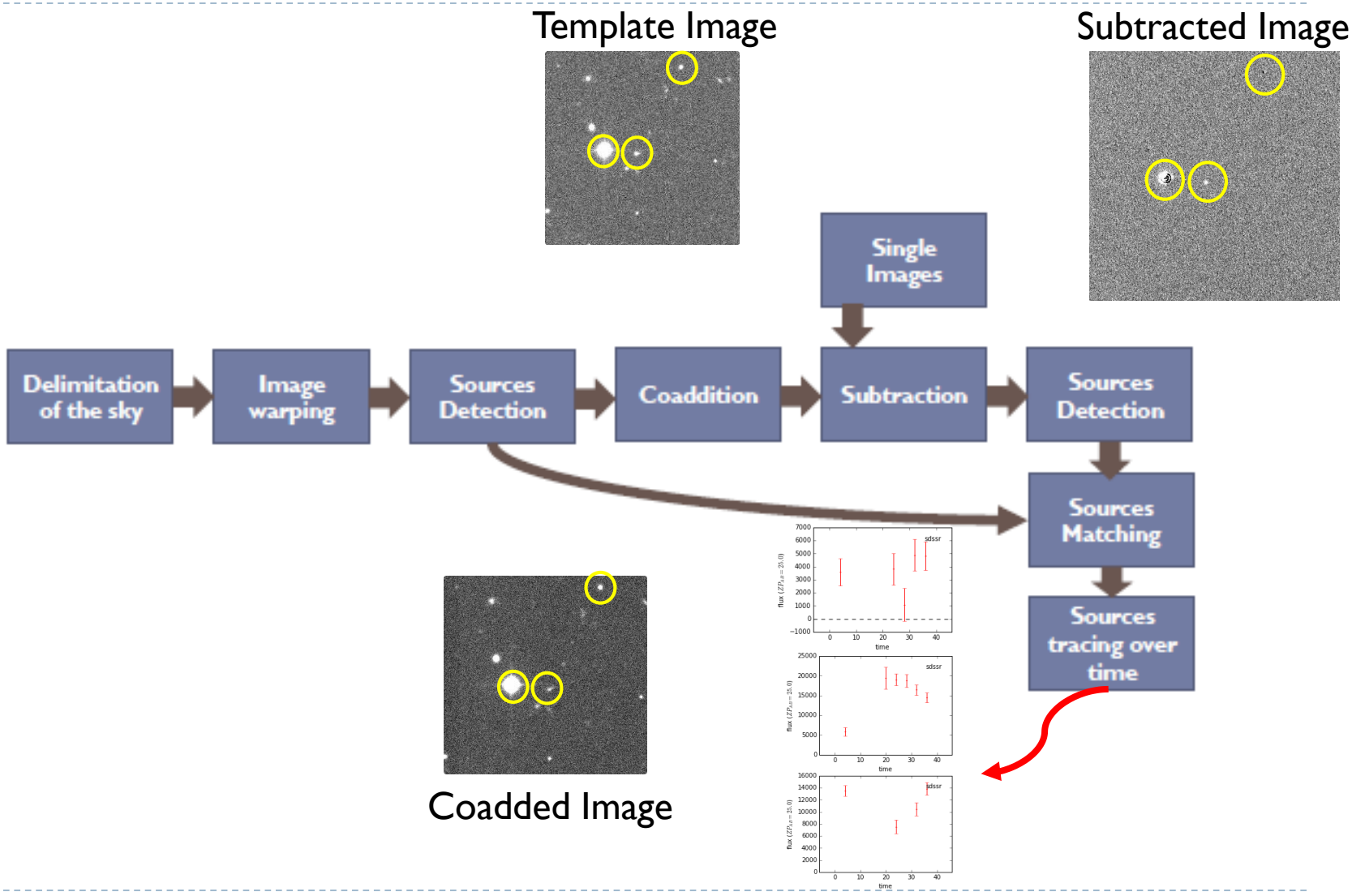


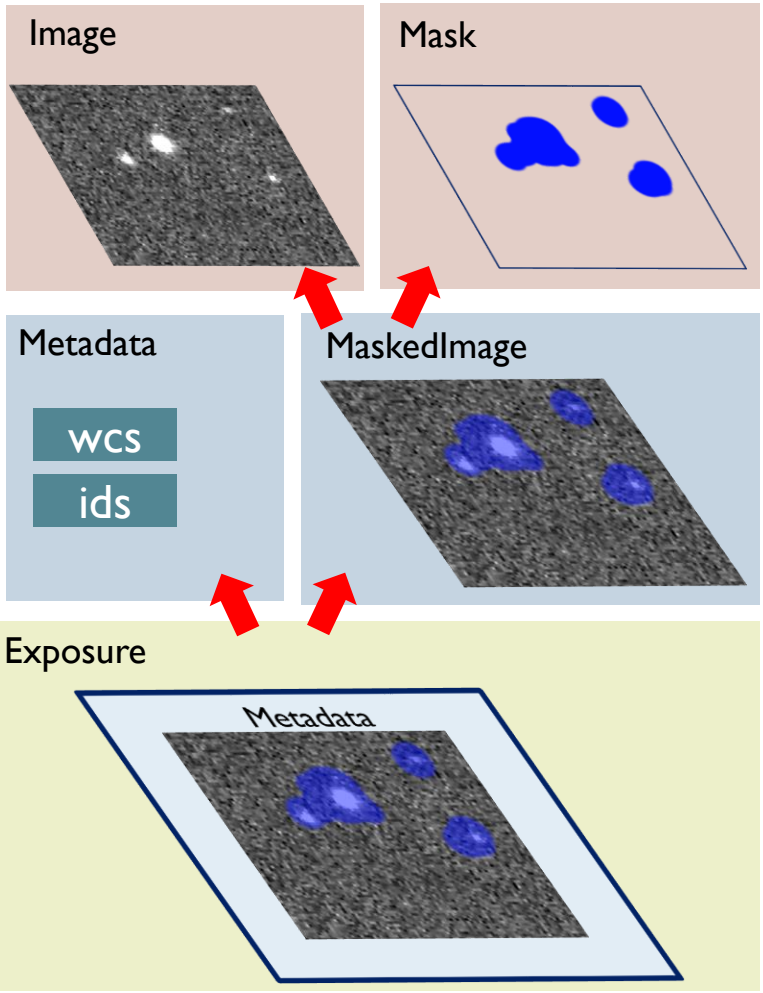
Image Subtraction on the LSST DM Stack: Pipeline and tasks



Tools and Utilities

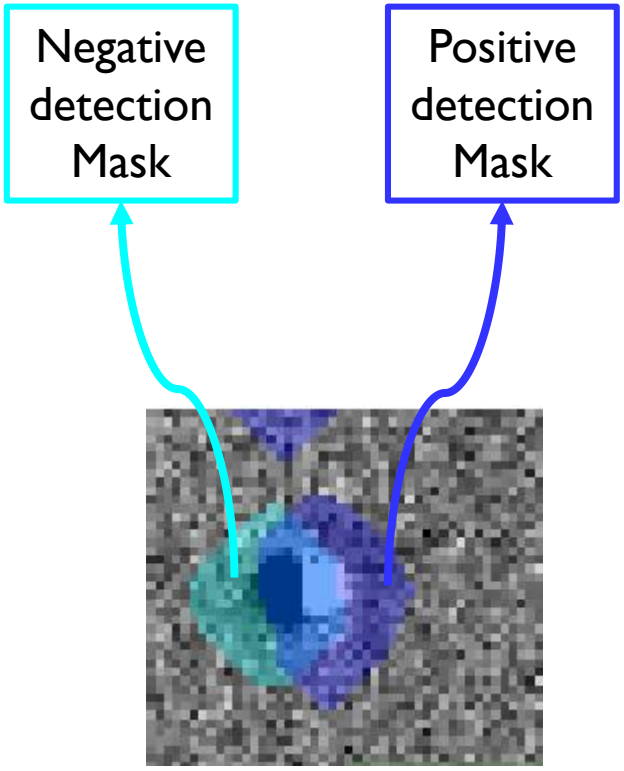
Tools and Utilities: Exposures and Masks

- Exposures contain Metadata and MaskedImages.
- MaskedImages contain Images and Masks.
- Masks are coded on a bit basis. They point to several elements on the image.



Tools and Utilities: Exposures and Masks

- Exposures contain Metadata and MaskedImages.
- MaskedImages contain Images and Masks.
- Masks are coded on a bit basis. They point to several elements on the image.
- For Image Difference, we used positive and negative detection masks (Footprints).



Tools and Utilities: Butler and byproducts

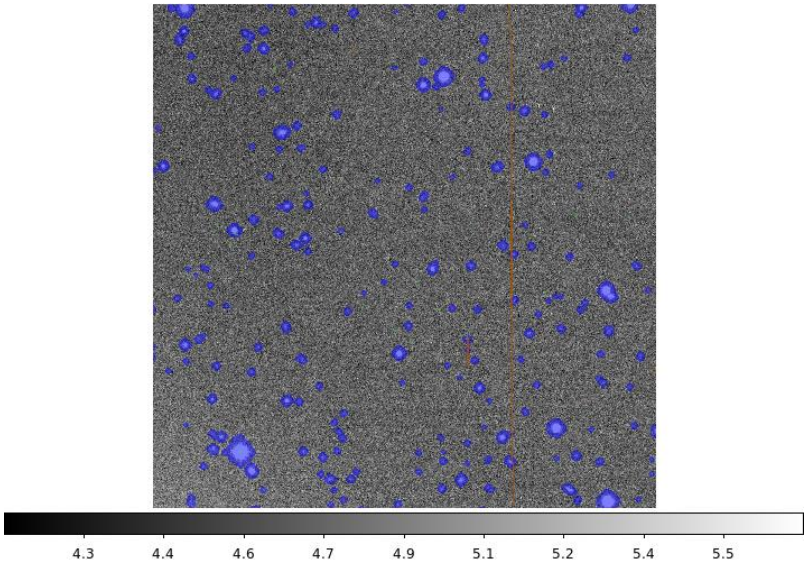
- Most products and images are handled on Python via the Butler.
- The Butler is capable of finding exposures, sources and catalogs using specifical "keys" in the form of a DataId.

```
dataId = {'visit': 845345 , 'filter':'r' , 'ccd':14}
diffExp =  butler.get("deepDiff_differenceExp", dataId}
```

```
deepDiff_differenceExp: {
      template:       "deepDiff/%(runId)s/%(object)s/%(date)s/
                        %(filter)s/diffexp-%(visit)d-%(ccd)02d.fits"
      python:         "lsst.afw.image.ExposureF"
      persistable:          "ExposureF"
      storage:     "FitsStorage"
      level:          "Ccd"
      tables:         "raw"
      columns:        "visit"
      columns:        "ccd"
  }
```

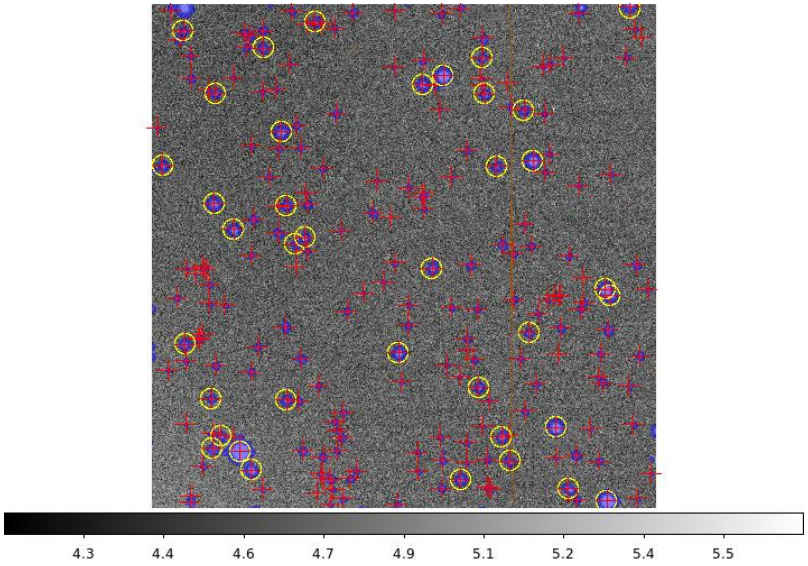
Tools and Utilities: Visualization of results

- DS9 instances can be opened from Python console or notebooks.
- Stack provides a whole library to interact with images opened on DS9. Labels and masks can be added.
- Custom images can be generated (stamps, source identification).
- Previous versions of stack had options to create image screenshots from DS9 to put on Notebooks.



Tools and Utilities: Visualization of results

- DS9 instances can be opened from Python console or notebooks.
- Stack provides a whole library to interact with images opened on DS9. Labels and masks can be added.
- Custom images can be generated (stamps, source identification).
- Previous versions of stack had options to create image screenshots from DS9 to put on Notebooks.



Tools and Utilities: Feedback, support and bug report

- There is basic documentation of all the modules. However such documentation is not exhaustive.
- Hipchat with DM team can help solve immediate and straightforward questions.
- Community.lsst.org can be used to discuss more complicated problems/questions.
- JIRA to report bugs / see patched bugs.
- Learn by doing.

Conclusions and Perspectives

Conclusions

- Stack can be flexible for certain needs, but it requires some tweaks to the base modules.
- Image Difference on Stack has given insights on the different elements. Including modularization and visualization.
- Lots of things on Stack can be learned by doing.

Perspectives

- Improvement of processes by taking into account time efficiency.
- Automatic classification algorithms on the Python layer, and eventually in C++.
- Map-Reduce applications on image subtraction.