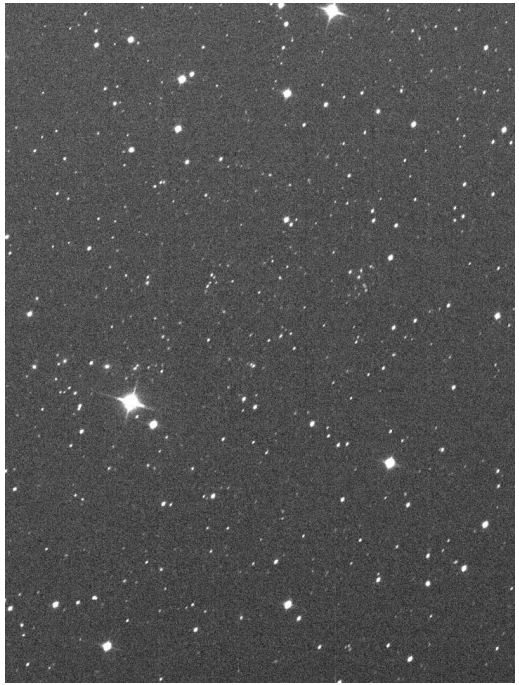
Image subtraction and transient detection

doing it the easy way

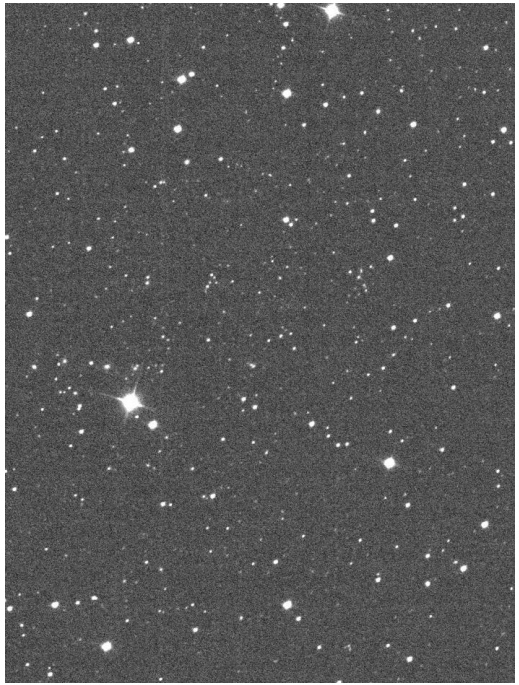
SPOT THE DIFFERENCE

Can you spot the 9 differences between these two pictures?

Transient detection

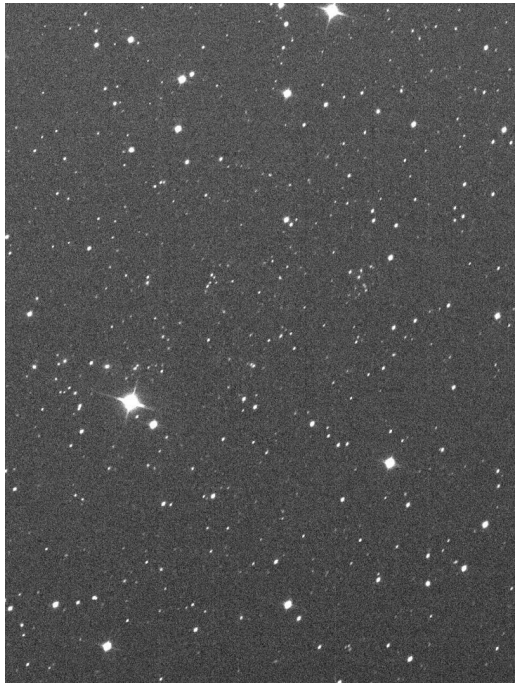


Science Image



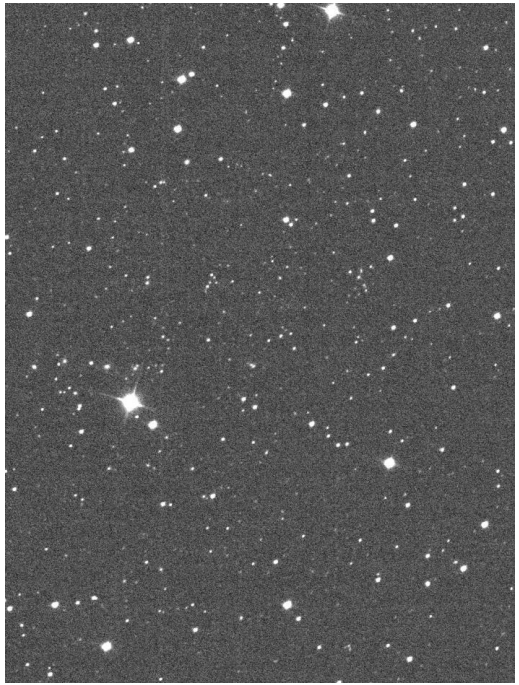
Reference Image

Transient detection by image subtraction



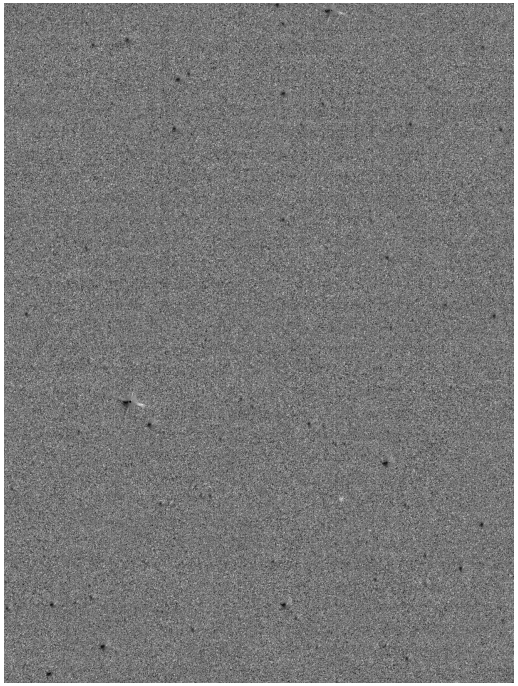
Science Image

-



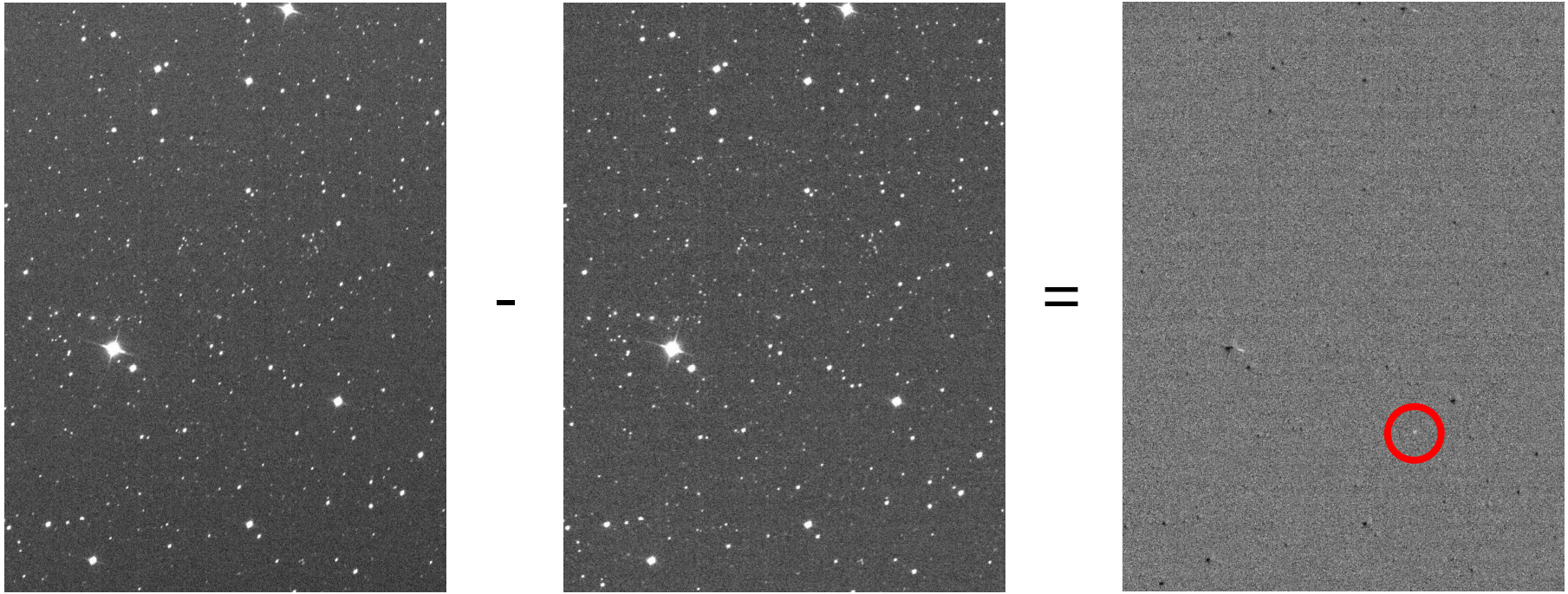
Reference Image

=



Difference Image

Transient detection by image subtraction



Science Image

Reference Image

Difference Image

Image subtraction

- **Alard & Lupton (1998)**
  - **HOTPANTS** (also **ISIS**, **LSST** stack, ...)
  - convolve one image to match another
  - kernel as a combination of basis functions
  - flux scaling is fitted in the process
  - does not need prior PSF info

- **Zackay, Ofek & Gal-Yam (2016)**
  - **ZOGY** (used by **ZTF**, **GOTO**, ...)
  - symmetric convolution
  - matched filter detection
  - more noise sources in the model
  - requires PSF and flux scaling as input

THE ASTROPHYSICAL JOURNAL, 503:325–331, 1998 August 10

A METHOD FOR OPTIMAL IMAGE SUBTRACTION

C. ALARD[1,2] AND ROBERT H. LUPTON[3]
Received 1997 December 17; accepted 1998 March 12

ABSTRACT

We present a new method designed for optimal subtraction of two images with different seeing. Using image subtraction appears to be essential for full analysis of microlensing survey images; however, a perfect subtraction of two images is not easy, as it requires the derivation of an extremely accurate convolution kernel. Some empirical attempts to find the kernel have used a Fourier transform of bright stars, but solving the statistical problem of finding the best kernel solution has never really been tackled. We demonstrate that it is possible to derive an optimal kernel solution from a simple least-squares analysis using all the pixels of both images, and we also show that it is possible to fit the differential background variation at the same time. We show that point-spread function (PSF) variations can be easily handled by the method. To demonstrate the practical efficiency of the method, we analyzed some images from a Galactic Bulge field monitored by the OGLE II project. We find that the residuals in the subtracted images are very close to the photon noise expectations. We also present some light curves of variable stars and show that despite high crowding levels, we get an error distribution close to that expected from photon noise alone. We thus demonstrate that nearly optimal differential photometry can be achieved even in very crowded fields. We suggest that this algorithm might be particularly important for microlensing surveys, where the photometric accuracy and completeness levels could be very significantly improved by using this method.

Subject headings: methods: data analysis — methods: statistical — techniques: image processing

THE ASTROPHYSICAL JOURNAL, 830:27 (23pp), 2016 October 10

PROPER IMAGE SUBTRACTION—OPTIMAL TRANSIENT DETECTION, PHOTOMETRY, AND HYPOTHESIS TESTING

BARAK ZACKAY, ERAN O. OFEK, AND AVISHAY GAL-YAM
Benoziyo Center for Astrophysics, Weizmann Institute of Science, 76100 Rehovot, Israel; bzackay@gmail.com, eran.ofek@weizmann.ac.il
Received 2016 January 11; revised 2016 May 23; accepted 2016 June 15; published 2016 October 4

ABSTRACT

Transient detection and flux measurement via image subtraction stand at the base of time domain astronomy. Due to the varying seeing conditions, the image subtraction process is non-trivial, and existing solutions suffer from a variety of problems. Starting from basic statistical principles, we develop the optimal statistic for transient detection, flux measurement, and any image-difference hypothesis testing. We derive a closed-form statistic that: (1) is mathematically proven to be the optimal transient detection statistic in the limit of background-dominated noise, (2) is numerically stable, (3) for accurately registered, adequately sampled images, does not leave subtraction or deconvolution artifacts, (4) allows automatic transient detection to the theoretical sensitivity limit by providing credible detection significance, (5) has uncorrelated white noise, (6) is a sufficient statistic for any further statistical test on the difference image, and, in particular, allows us to distinguish particle hits and other image artifacts from real transients, (7) is symmetric to the exchange of the new and reference images, (8) is at least an order of magnitude faster to compute than some popular methods, and (9) is straightforward to implement. Furthermore, we present extensions of this method that make it resilient to registration errors, color-refraction errors, and any noise source that can be modeled. In addition, we show that the optimal way to prepare a reference image is the proper image coaddition presented in Zackay & Ofek. We demonstrate this method on simulated data and real observations from the PTF data release 2. We provide an implementation of this algorithm in MATLAB and Python.

Image subtraction

- **Alard & Lupton (1998)**
  - **HOTPANTS** (also **ISIS**, **LSST** stack, ...)
  - convolve one image to match another
  - kernel as a combination of basis functions
  - flux scaling is fitted in the process
  - does not need prior PSF info

$$D_{AL} = N - K \otimes R$$

$$K_i(u, v) = \mathrm{e}^{-(u^2+v^2)/2\sigma_n^2} u^p v^q$$

- **Zackay, Ofek & Gal-Yam (2016)**
  - **ZOGY** (used by **ZTF**, **GOTO**, ...)
  - symmetric convolution
  - matched filter detection
  - more noise sources in the model
  - requires PSF and flux scaling as input

$$\widehat{D} = \frac{F_r \widehat{P_r} \widehat{N} - F_N \widehat{P}_N \widehat{R}}{\sqrt{\sigma_n^2 F_r^2 \left|\widehat{P_r}\right|^2 + \sigma_r^2 F_n^2 \left|\widehat{P_n}\right|^2}}$$

$$S = F_D D \otimes \overleftarrow{P_D}$$

$$S_{corr} = \frac{S}{\sqrt{V(S_N) + V(S_R) + V_{ast}(S_N) + V_{ast}(S_R) \ldots}}$$

HOTPANTS: High Order Transform of Psf ANd Template Subtraction code

```
Usage:  hotpants [options]
Version 5.1.11
Required options:
  [-inim fitsfile]  : comparison image to be differenced
  [-tmplim fitsfile]: template image
  [-outim fitsfile] : output difference image

Additional options:
  [-tu tuthresh]    : upper valid data count, template (25000)
  [-tl tlthresh]    : lower valid data count, template (0)

  [-tg tgain]       : gain in template (1)
  [-tr trdnoise]    : e- readnoise in template (0)
  [-tp tpedestal]   : ADU pedestal in template (0)
  [-tni fitsfile]   : input template noise array (undef)
  [-tmi fitsfile]   : input template mask image (undef)

  [-iu iuthresh]    : upper valid data count, image (25000)
  [-il ilthresh]    : lower valid data count, image (0)

  [-ig igain]       : gain in image (1)
  [-ir irdnoise]    : e- readnoise in image (0)
  [-ip ipedestal]   : ADU pedestal in image (0)
  [-ini fitsfile]   : input image noise array (undef)
  [-imi fitsfile]   : input image mask image (undef)

  [-r rkernel]      : convolution kernel half width (10)
  [-c  toconvolve]  : force convolution on (t)emplate or (i)mage (undef)
  [-n  normalize]   : normalize to (t)emplate, (i)mage, or (u)nconvolved (t)
  [-allm]           : output all possible image layers

  [-rss radius]     : half width substamp to extract around each centroid (15)
  [-ko kernelorder] : spatial order of kernel variation within region (2)
  [-bgo bgorder]    : spatial order of background variation within region (1)
  [-ng  ngauss degree0 sigma0 .. degreeN sigmaN]
                    : ngauss = number of gaussians which compose kernel
                    : degree = degree of polynomial associated with gaussian
                    : sigma  = width of gaussian
                    : (3 6 0.70 4 1.50 2 3.00
```

Data range

Noise model

Data range

Noise model

Convolution

Kernel fitting

$$D_{AL} = N - K \otimes R$$

$$K_i(u, v) = \mathrm{e}^{-(u^2+v^2)/2\sigma_n^2} u^p v^q$$

https://github.com/acbecker/hotpants

Rule of thumb for the **input parameters**:

- Error model = background RMS + Poissonian
- r = 3*FWHM, rss = 4*FWHM
- kernel model:
  Sigma_match = sqrt(Sigma_image^2 - Sigma_template^2)
  ng = 3 6 0.5*Sigma_match 4 Sigma_match 2 2*Sigma_match
- ko = 0, bgo = 0 if your images are small
- c = t – convolve template!
- n = i – normalize to original image

Results (difference image, its noise model, and noise-scaled difference) will be in the FITS image extensions of the **output image**

Image subtraction: complications

- Proper image alignment is important

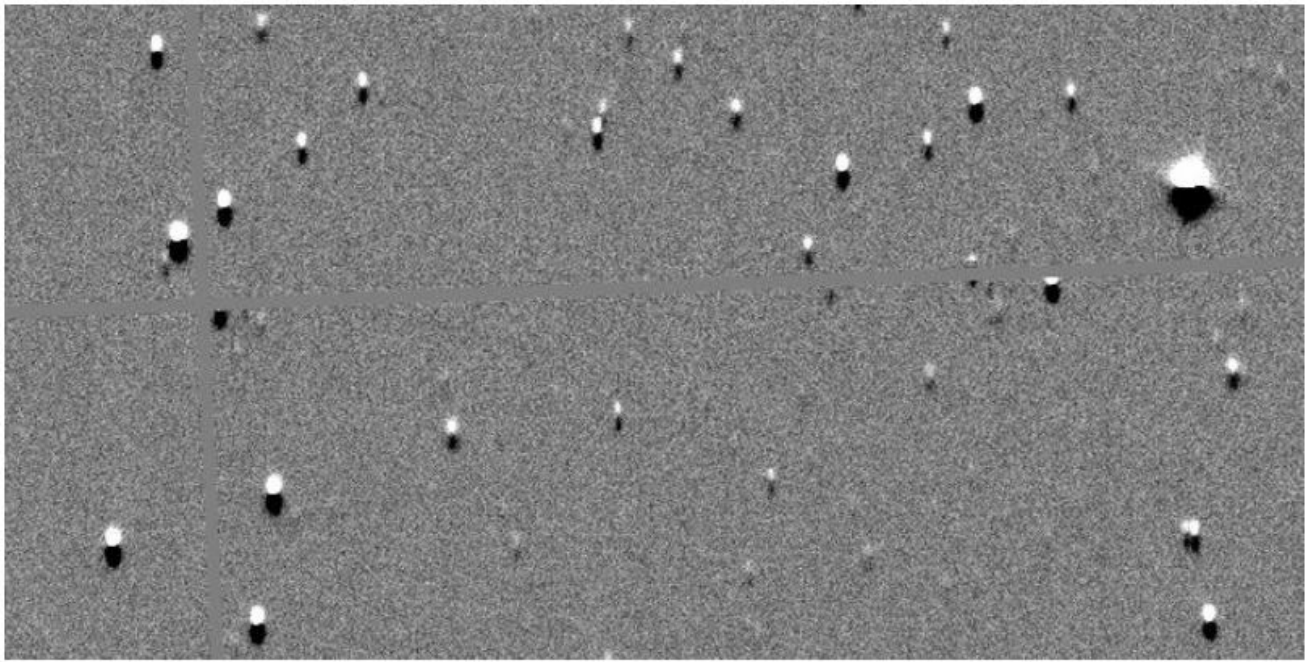
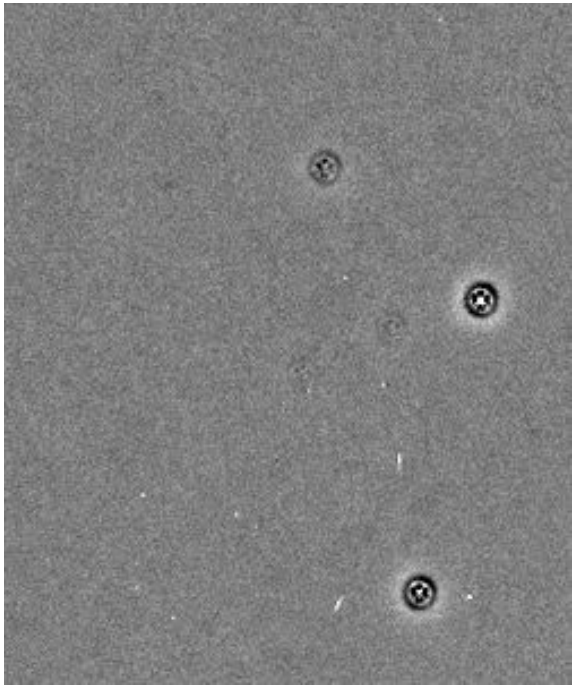


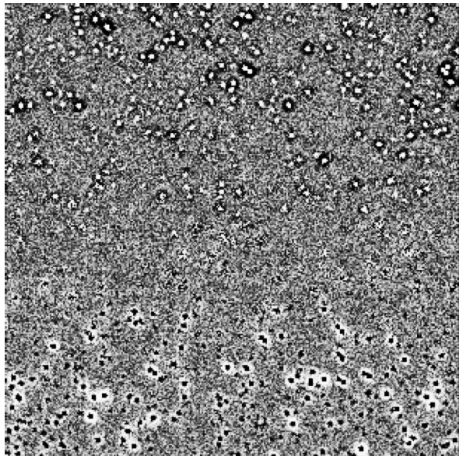
Image subtraction: artefacts

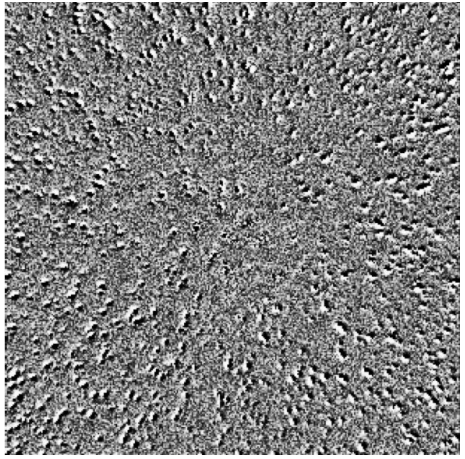
- Subtraction artefacts



deconvolution artefacts

variable flux scale

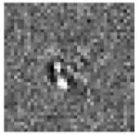




position-dependent PSF

perfect match

shifted too much



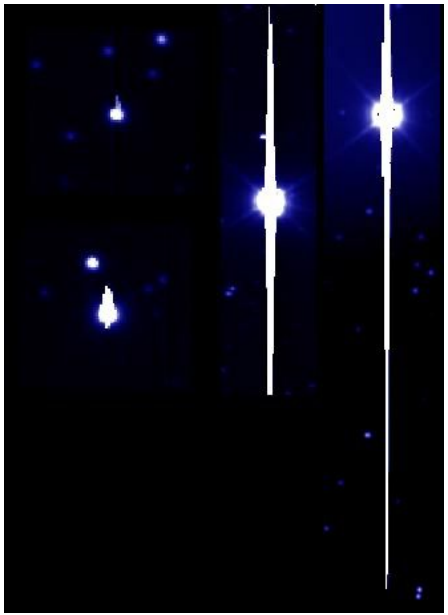
kernel is too large

polynomial order is too small

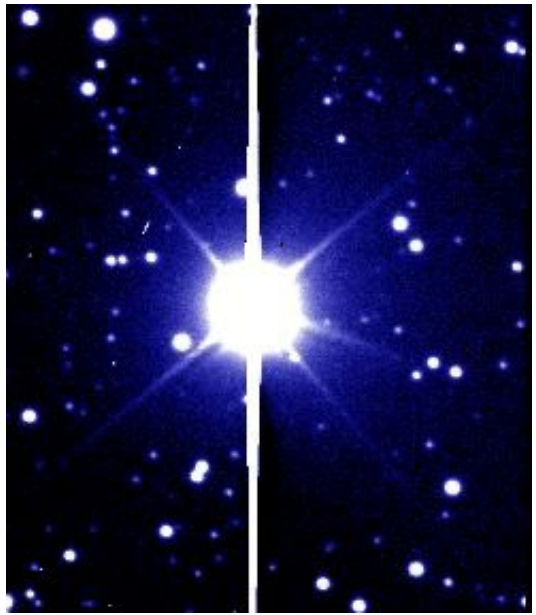
Image subtraction: artefacts

- CCD artefacts
  - have to be masked, e.g. with **LACosmic** / **astroscrappy**

cosmic rays and hot pixels

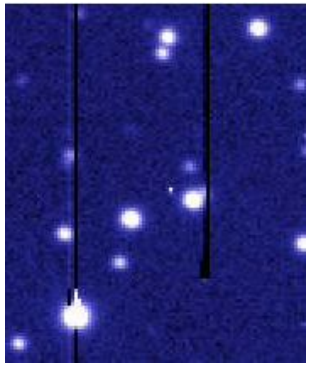


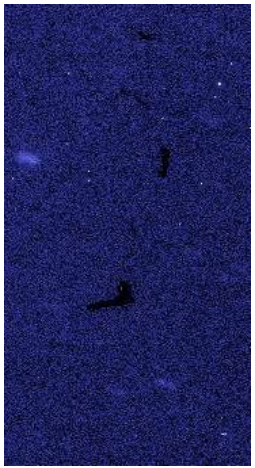
saturation trails

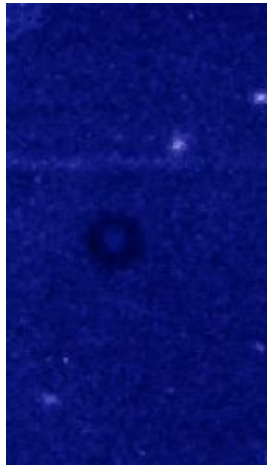


diffraction patterns

bad columns and traps





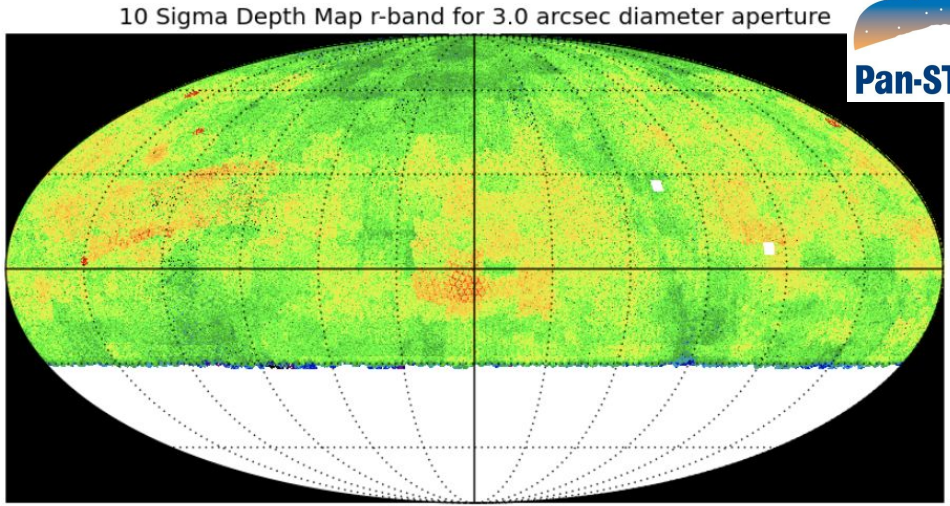


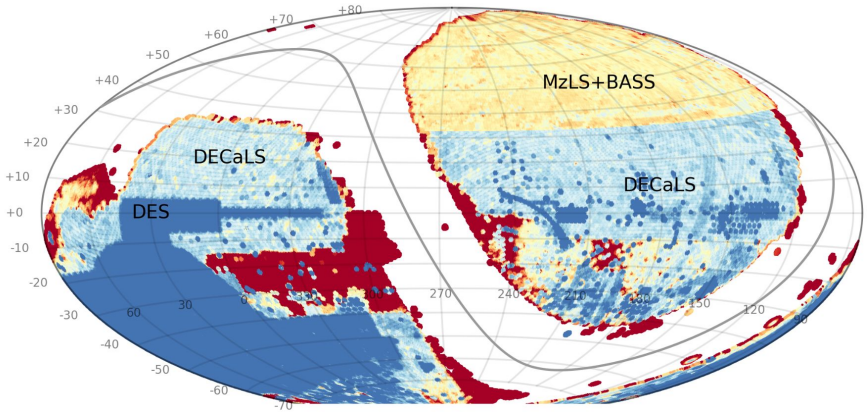
dust on CCD surface or filter

Template images

- Your own data
  - Pre-observed or post-observed

- Public data archives
  - Pan-STARRS DR1 stack images
    - 70% of the sky, Dec > -30 deg
    - g, r, i, z, y filters (quasi-Sloan)
    - down to ~23 mag
  - DESI Legacy Surveys
    - different surveys, North+South
    - most of |b| > 20 deg
    - deeper than Pan-STARRS
  - SkyMapper
    - Southern sky
    - down to ~21 mag
    - quality so-so

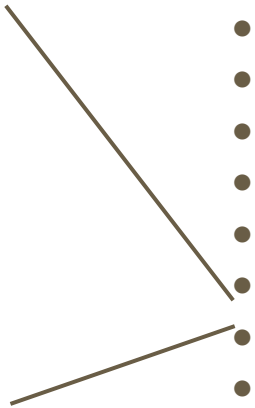




DESI Legacy Imaging Surveys

Image subtraction and Transient detection

- Acquire the template
- Align it with the image
- Estimate FWHM / PSF of both images
- Estimate the noise model
- Match the resolution
- Match the flux scale
- Subtract
- Assess pixel-level significances



- Pre-process and mask the image
- Detect sources in the image
- Get rough astrometric information
- Get reference catalogue
- Match the objects with catalogue
- Get photometric solution
- Select transient candidates
- Assess their significance

SWarp / **reproject**
SExtractor + **PSFEx**
HOTPANTS / **ZOGY**

astroscrappy
SExtractor / **photutils**
Astrometry.Net / **SCAMP** / **astropy**
astroquery
+ lots of boilerplate code

STDPipe - Simple Transient Detection Pipeline

STDPipe is a set of Python routines for astrometry, photometry and transient detection related tasks, intended for quick and easy implementation of custom pipelines, as well as for interactive data analysis.

Design principles:

- implemented as a library of routines covering most common tasks
- operates on standard Python objects: NumPy arrays for images, AstroPy Tables for catalogs and object lists, ...
- conveniently wraps external codes that do not have their own Python interfaces (SExtractor, SCAMP, PSFEx, HOTPANTS, Astrometry.Net, ...)

GitHub repository - https://github.com/karpov-sv/stdpipe

Documentation - https://stdpipe.readthedocs.io/

STDPipe - features

- ~~pre-processing~~ - should be handled before in an instrument-specific way
  - bias/dark subtraction, flatfielding, masking
- object detection and photometry
  - **SExtractor** or **SEP** for detection, **photutils** for forced photometry
- astrometric calibration
  - **Astrometry.Net** for blind WCS solving, both local and online
  - **SCAMP** or **Astropy**-based code for refinement
- photometric calibration
  - **Vizier** catalogues, passband conversion (PS1 to Johnson, Gaia to Johnson, ...)
  - spatial polynomial + color term + additive term + intrinsic scatter
- image subtraction
  - Pan-STARRS or DESI Legacy Survey images, or HiPS templates
  - **HOTPANTS** + custom noise model, or **ZOGY**
- transient detection and photometry
  - noise-weighted detection, cutout adjustment, ...
- auxiliary functions
  - PSF estimation with **PSFEx**, simulated stars, FITS header utilities, image splitting, plotting, ...
- light curve creation (in progress)
  - spatial clustering, color regression, variability analysis, ...

STDPipe - tutorial

We will use the following routines from STDPipe in our tutorial to build a complete pipeline:

- **plots.imshow** - drop-in replacement for Matplotlib imshow with better intensity scaling
- **photmetry.get_objects_sextractor** - wrapper for running **SExtractor** and getting its result as an Astropy Table object
- **photometry.measure_objects** - forced aperture photometry with optional sky annulus background subtraction
- **catalogs.get_cat_vizier** - for getting the catalogues from Vizier database
- **pipeline.refine_astrometry** - for refining existing astrometric solution using **SCAMP**
- **pipeline.calibrate_photometry** - for deriving the photometric solution
- **plots.plot_photometric_match** - for plotting the results of photometric calibration
- **templates.get_survey_image** - for downloading template images from Pan-STARRS, and aligning them with science image using **SWarp**
- **subtraction.run_hotpants** - for subtracting the images using **HOTPANTS**
- **pipeline.filter_transient_candidates** - for filtering the transient candidates based on simple criteria
- **cutouts.get_cutout** - for making the cutouts from the image around transient positions
- **plots.plot_cutout** - for displaying the cutouts