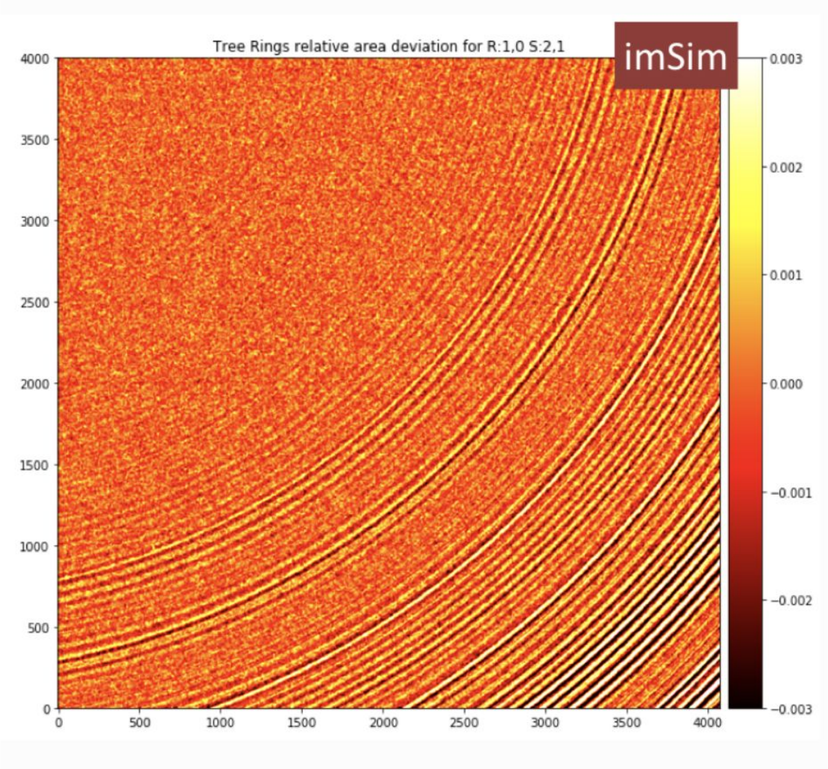
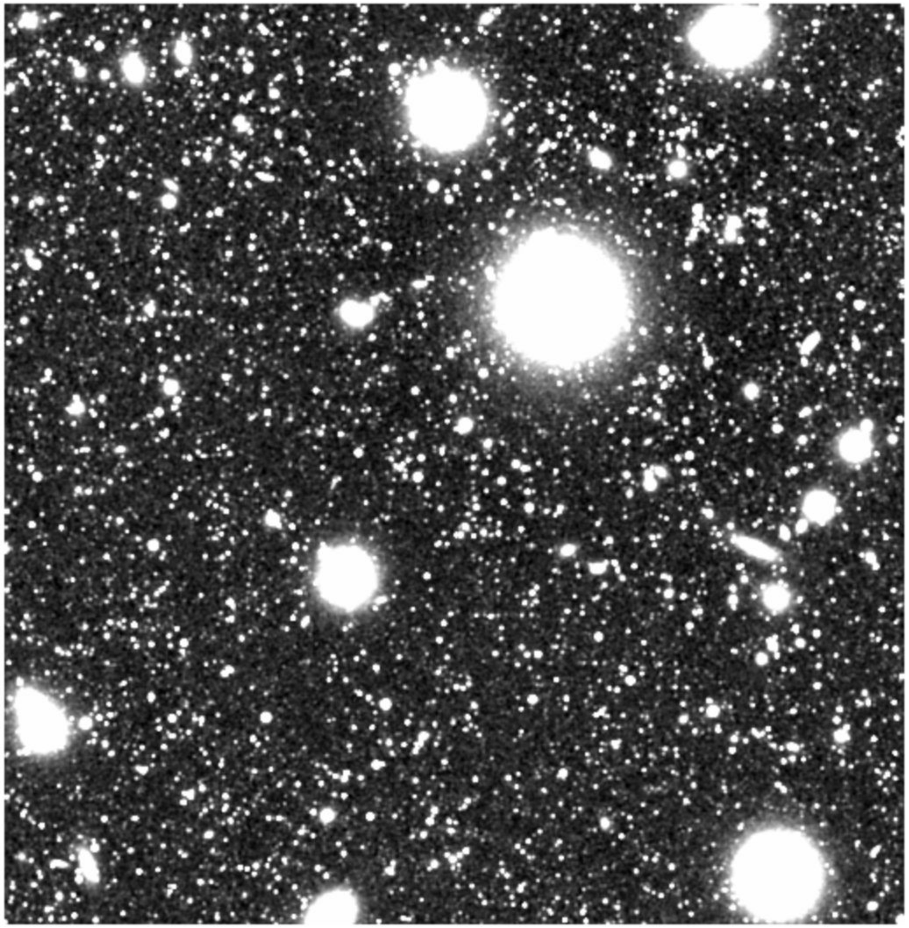
imSim: Introduction and Opportunities





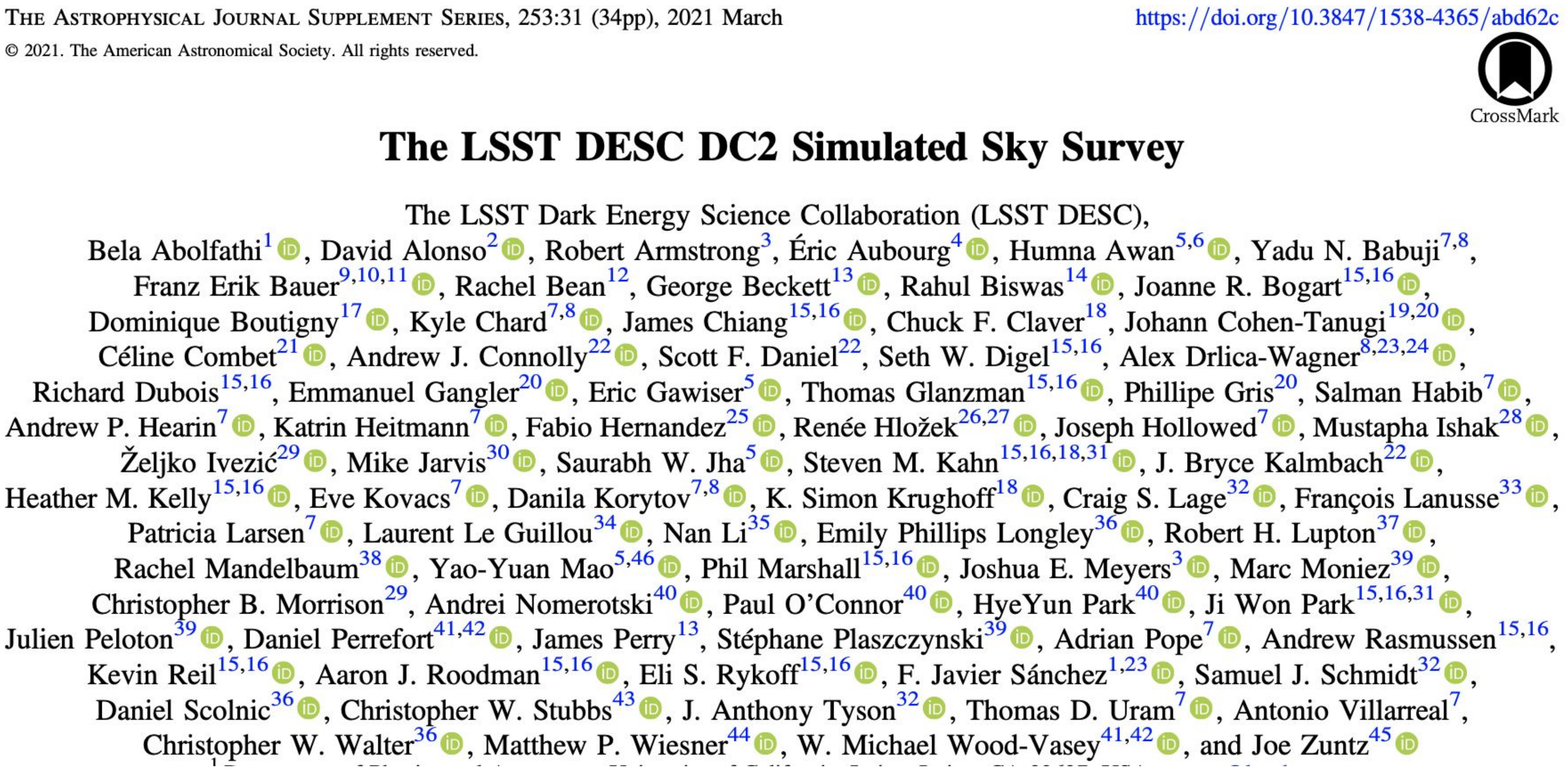
Chris Walter for the *imSim development team*
Duke University
Rubin Project and Community Workshop
August of 2023

Mastodon:
chris.walter@fediscience.org

Goal of this session:

- Introduce imSim to the wider Rubin community
- Tell you about recent developments
- Discuss how you might get involved.

Many of you have already used imSim



The Astrophysical Journal Supplement Series, 253:31 (34pp), 2021 March
https://doi.org/10.3847/1538-4365/abd62c

CrossMark

The LSST DESC DC2 Simulated Sky Survey

The LSST Dark Energy Science Collaboration (LSST DESC),
Bela Abolfathi[1], David Alonso[2], Robert Armstrong[3], Éric Aubourg[4], Humna Awan[5,6], Yadu N. Babuji[7,8], Franz Erik Bauer[9,10,11], Rachel Bean[12], George Beckett[13], Rahul Biswas[14], Joanne R. Bogart[15,16], Dominique Boutigny[17], Kyle Chard[7,8], James Chiang[15,16], Chuck F. Claver[18], Johann Cohen-Tanugi[19,20], Céline Combet[21], Andrew J. Connolly[22], Scott F. Daniel[22], Seth W. Digel[15,16], Alex Drlica-Wagner[8,23,24], Richard Dubois[15,16], Emmanuel Gangler[20], Eric Gawiser[5], Thomas Glanzman[15,16], Phillipe Gris[20], Salman Habib[7], Andrew P. Hearin[7], Katrin Heitmann[7], Fabio Hernandez[25], Renée Hložek[26,27], Joseph Hollowed[7], Mustapha Ishak[28], Željko Ivezić[29], Mike Jarvis[30], Saurabh W. Jha[5], Steven M. Kahn[15,16,18,31], J. Bryce Kalmbach[22], Heather M. Kelly[15,16], Eve Kovacs[7], Danila Korytov[7,8], K. Simon Krughoff[18], Craig S. Lage[32], François Lanusse[33], Patricia Larsen[7], Laurent Le Guillou[34], Nan Li[35], Emily Phillips Longley[36], Robert H. Lupton[37], Rachel Mandelbaum[38], Yao-Yuan Mao[5,46], Phil Marshall[15,16], Joshua E. Meyers[3], Marc Moniez[39], Christopher B. Morrison[29], Andrei Nomerotski[40], Paul O'Connor[40], HyeYun Park[40], Ji Won Park[15,16,31], Julien Peloton[39], Daniel Perrefort[41,42], James Perry[13], Stéphane Plaszczynski[39], Adrian Pope[7], Andrew Rasmussen[15,16], Kevin Reil[15,16], Aaron J. Roodman[15,16], Eli S. Rykoff[15,16], F. Javier Sánchez[1,23], Samuel J. Schmidt[32], Daniel Scolnic[36], Christopher W. Stubbs[43], J. Anthony Tyson[32], Thomas D. Uram[7], Antonio Villarreal[7], Christopher W. Walter[36], Matthew P. Wiesner[44], W. Michael Wood-Vasey[41,42], and Joe Zuntz[45]

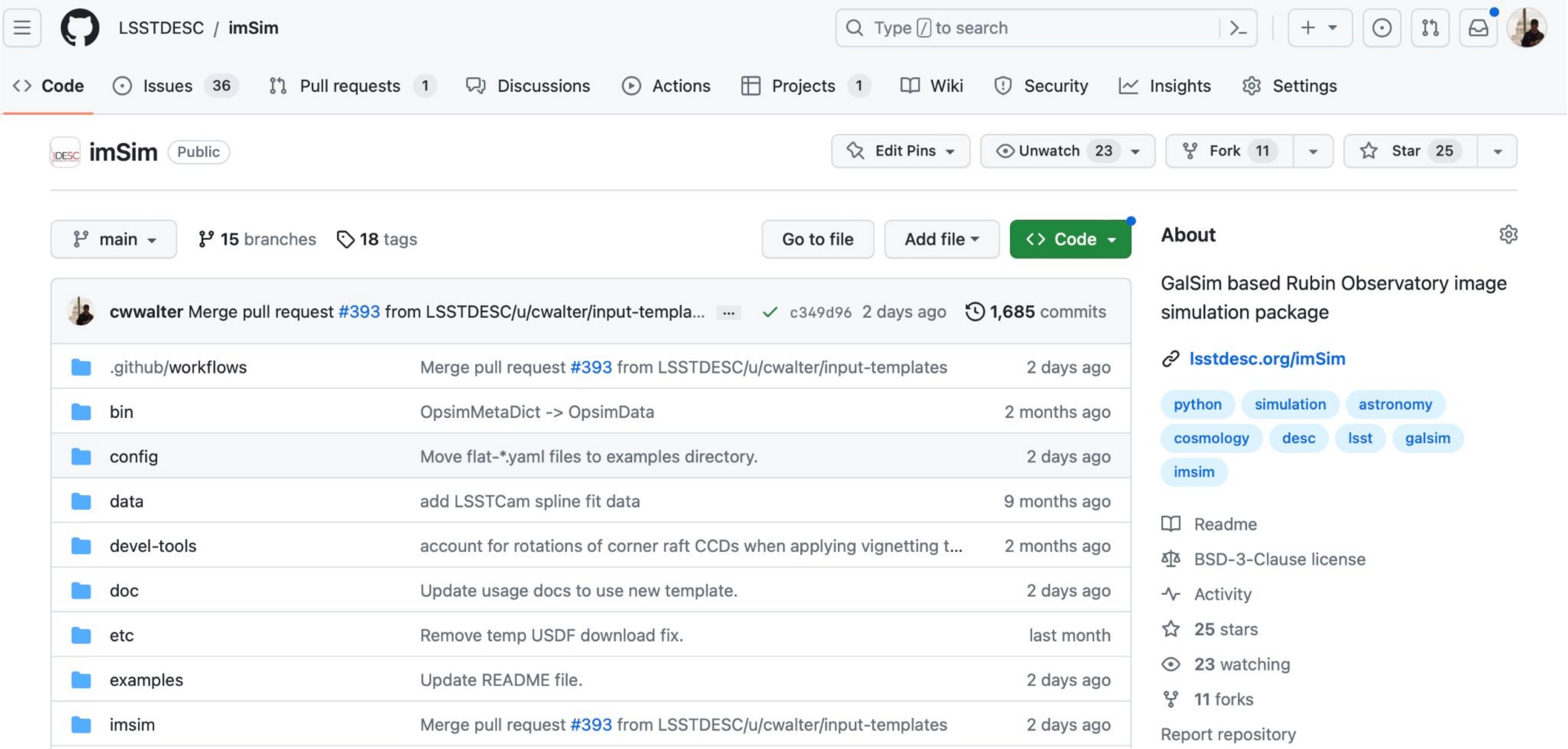
An imSim introduction

- imSim began in the fall of 2016
- The goal was to have a program embedded in the Rubin software environment that would use GalSim for its image rendering engine with a truly open and collaborative development model.
- imSim is developed on GitHub and hosted by the DESC but we welcome collaboration from members across the Rubin community.

- imSim has been a big success!
- It was the image simulation used in DC2. **That data set is now being used not only by the DESC but by delegates on the RSP and by the public via available download.**

Where can you find imSim?

Code: https://github.com/LSSTDESC/imSim
Documentation: lsstdesc.org/imSim



cwwalter Merge pull request #393 from LSSTDESC/u/cwalter/input-templa... ✓ c349d96 2 days ago 1,685 commits

.github/workflows	Merge pull request #393 from LSSTDESC/u/cwalter/input-templates	2 days ago
bin	OpsimMetaDict -> OpsimData	2 months ago
config	Move flat-*.yaml files to examples directory.	2 days ago
data	add LSSTCam spline fit data	9 months ago
devel-tools	account for rotations of corner raft CCDs when applying vignetting t...	2 months ago
doc	Update usage docs to use new template.	2 days ago
etc	Remove temp USDF download fix.	last month
examples	Update README file.	2 days ago
imsim	Merge pull request #393 from LSSTDESC/u/cwalter/input-templates	2 days ago

About

GalSim based Rubin Observatory image simulation package

lsstdesc.org/imSim

python simulation astronomy cosmology desc lsst galsim imsim

Readme
BSD-3-Clause license
Activity
25 stars
23 watching
11 forks
Report repository

How is the work organized?

- imSim is developed in public repositories in an open process.
- The development is overseen by the DESC science collaboration (imSim is not developed by the Rubin project although many project members contribute).
- In the DESC the imSim effort is a topical team under the computing group.
- **We welcome other contributors**
- The technical work is organized on LSSTC slack #desc-imsim-technical and there are developer meetings every two weeks at 8:00 US Pacific / 11:00 US Eastern / 17:00 CET.

Who are the developers?

- The core group of developers includes about 10 people. We are mostly individual scientists. There are contributors from the UK, Switzerland, and the US.
- There is some in-kind software engineering contributions and pipeline scientists support via Rubin/DESC.
- -> A larger group of interested people also attend the meetings for consultation or specific contributions. We host contributions from **students(!)** and others at various universities who add needed realism features or enhancements. These are good stand alone projects.
  - Good opportunity to really understand a feature of Rubin, contribute, and learn modern software development techniques.

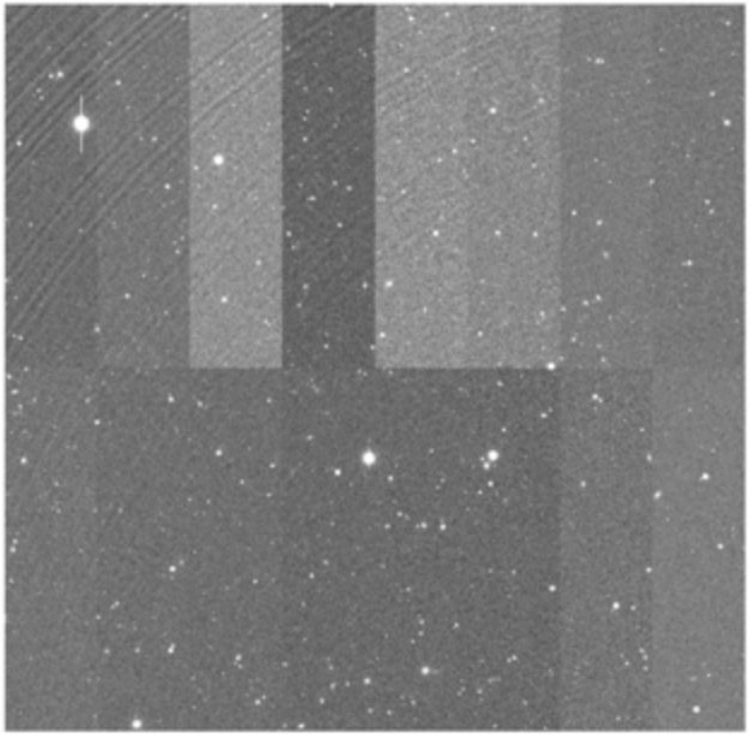
Because this, the implemented features are oriented towards the needs of Dark Energy science and commissioning. We aren't funded and don't have the expertise to implement some of the needs of the other science collaborations.

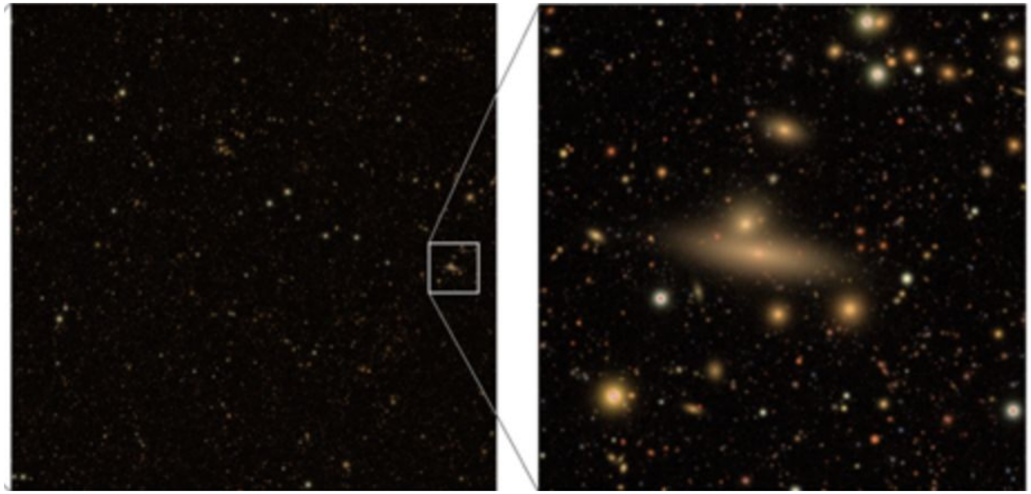
We welcome those with expertise to add new features!

Some Primary Past Features

- Usable at **scale** (like in DC2) but also for smaller scale studies (like stellar streams) via handcrafted input files including on your **laptop**.
- Can read instance catalogs derived from the opSim database, cosmoDC2 galaxy models, and Rubin SED and stellar databases.
- Uses the Rubin sky model 🌈.
- Has multi-layer atmospheric PSF model with ray-tracing through six optical screens.
- CCD sensor effects such as tree-rings and the brighter-fatter effect are included.
- Has full camera electronics readout.
- Produces FITS files that can be processed with the Rubin Science Pipelines.
- Also, produces truth images and text files with per source information
- Much effort went into optimizing the image production for speed.

Example DC2 images





Now: a major new release!

We have been working hard for the last 1.5-2 years on a redesign of imSim.

- We learned a lot during DC2 and found several points to address.
- We want to leverage new commonly available hardware configurations (especially GPUs).
- We want to make the system helpful during **commissioning** for the project, useful for testing DESC **pipelines**, and a tool we will continue to use into **operations for analysis**.
- We want to build an **ecosystem** where pieces of imSim can can be mixed-and matched when needed with other simulation and analysis packages based on GalSim. So, for example, blending or source injection programs could use the imSim source input, or electronics readout modules.

Some new features.

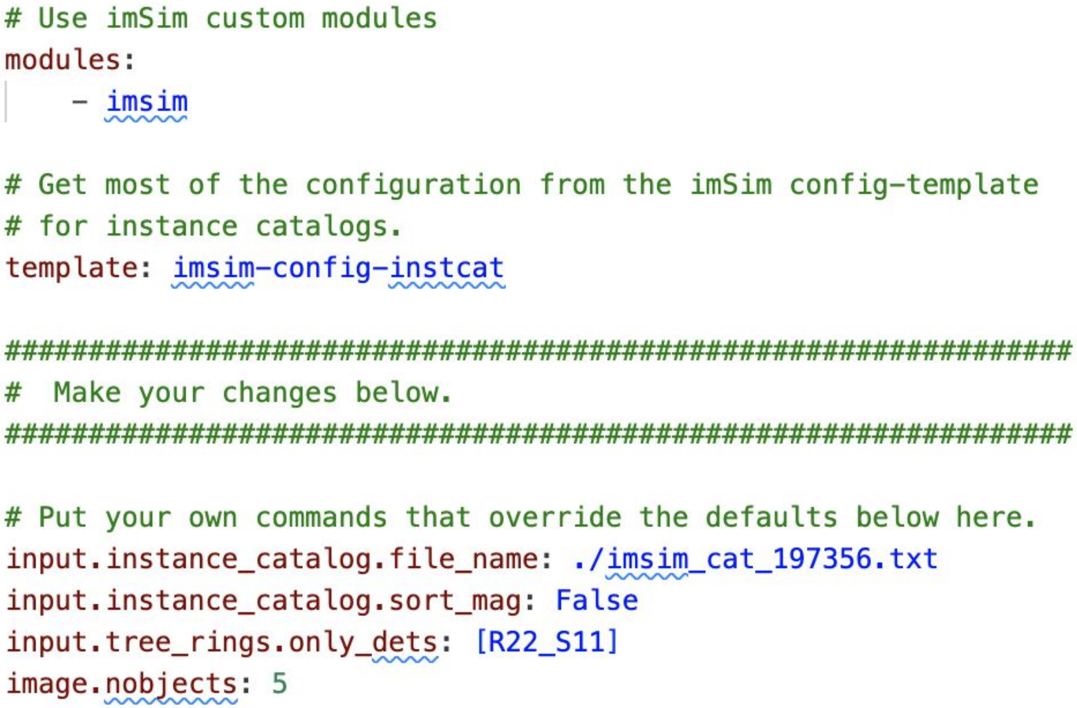
- **Real documentation** 😀
- A new self contained **SkyCatalog** input format
- Using the GalSim YAML config system for control and modularization
- True optical ray tracing through the telescope with **Batoid** and production of associated WCSs.
- Improved wavelength-dependence of PSFs
- Leveraging of GPUs for speed.
- Simulation of telescope AOS system and associated degrees of freedom.
- New realism including:
    - **Diffraction spikes** including camera rotation
    - Vignetting
    - Fringing in y-band
    - Smoother and varying sky levels
    - ….

New feature highlights

Some of the major changes are highlighted in the following slides.

GalSim Configuration System

- We now control the system via the GalSim YAML files config system. This is key for allowing imSim modules to mix-and-match with other GalSim based packages. Users need only list the imSim module in their config file.
- We have a hierarchical system where a base imSim config can be overridden by users.



```
# Use imSim custom modules
modules:
    - imsim

# Get most of the configuration from the imSim config-template
# for instance catalogs.
template: imsim-config-instcat

################################################################
#  Make your changes below.
################################################################

# Put your own commands that override the defaults below here.
input.instance_catalog.file_name: ./imsim_cat_197356.txt
input.instance_catalog.sort_mag: False
input.tree_rings.only_dets: [R22_S11]
image.nobjects: 5
```

Mike Jarvis

SkyCatalogs

- In DC2 we generated a new instance Catalog for each visit. We keep this functionality for small handcrafted tests.
- Now we also have a “SkyCatalog” which is a database describing a piece of our synthetic sky.
- Driven by an OpSim run, we can query the SkyCatalog for each visit to return all objects to simulate via an API.
- The SkyCatalog also will serve as the future source of truth information. It is divided into healpixels and can be downloaded for local analysis.
- Transient object fluxes can be cached and pre-calculated for performance.

Joanne Bogart
Jim Chiang

skyCatalogs and time domain objects

- **skyCatalogs** solve the problem of representing the sky in a database where objects can be retrieved via a consistent API so that we do not need to repeat the same information over-and-over again when we simulate visits in the same part of the sky.
- This is relatively straightforward for static sources and we are already doing this for galaxies and Gaia stars (for commissioning).

- Time domain objects (SN) are more complicated because the sky is not the same for all times (**example of SN**).
- We will store time dependent SEDs light curves at high cadence for transient objects.

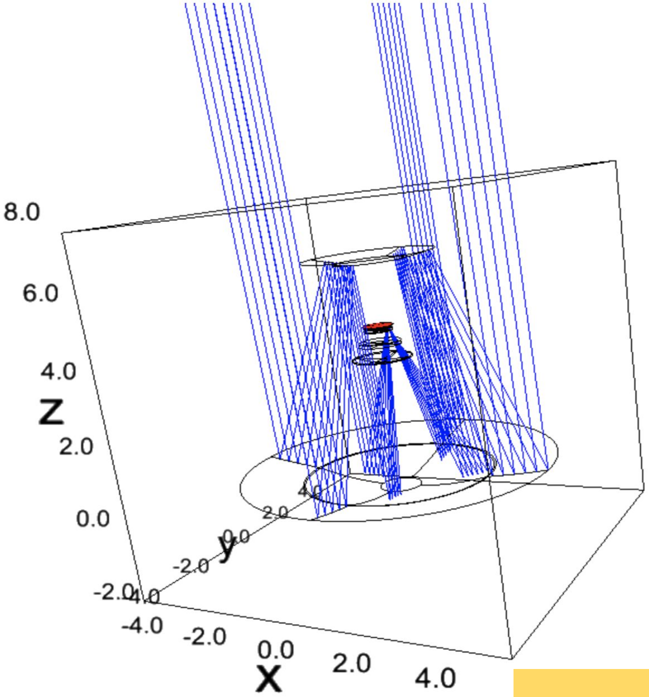
We already, store DC2 galaxies, Gaia Stars, and will soon be storing SN etc.   Soon we will also store new more realistic synthetic galaxies.

This code could be extended for things like moving objects.

True optical ray-tracing with batoid

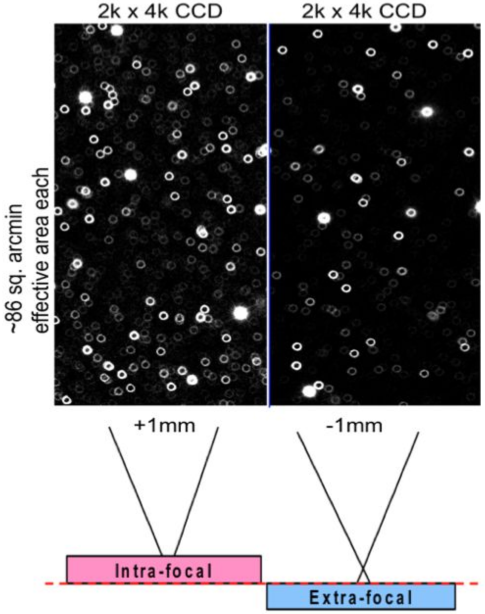
https://github.com/jmeyers314/batoid

- “Batoid” = superorder of cartilaginous fish, consisting of skates, ***rays***, and other fish
- Python-based c++-backed optical raytracer
- Validated by comparing the Rubin optical model against *Zemax. (*~1e-5 of a pixel)
- We have full control over the telescope optical degrees of freedom.

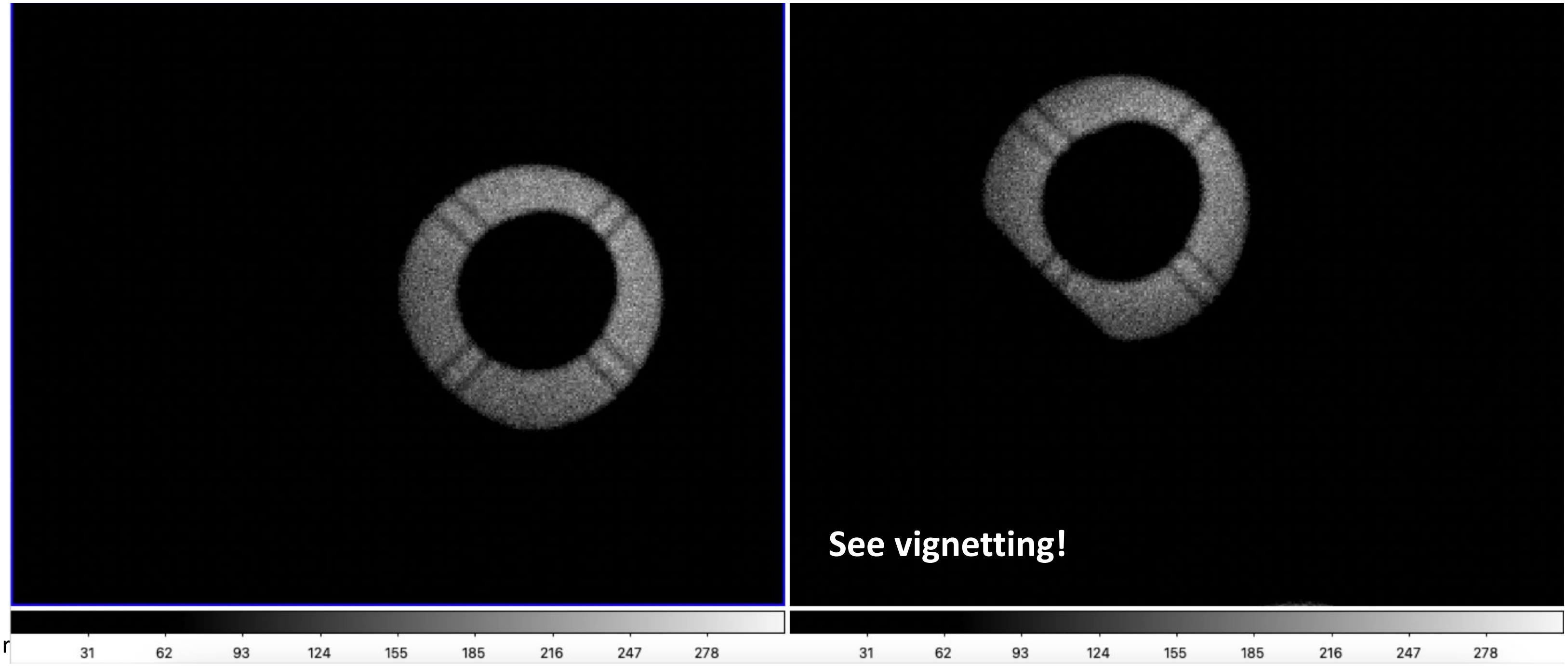


Josh Meyers

imSim uses the project camera and telescope descriptions.



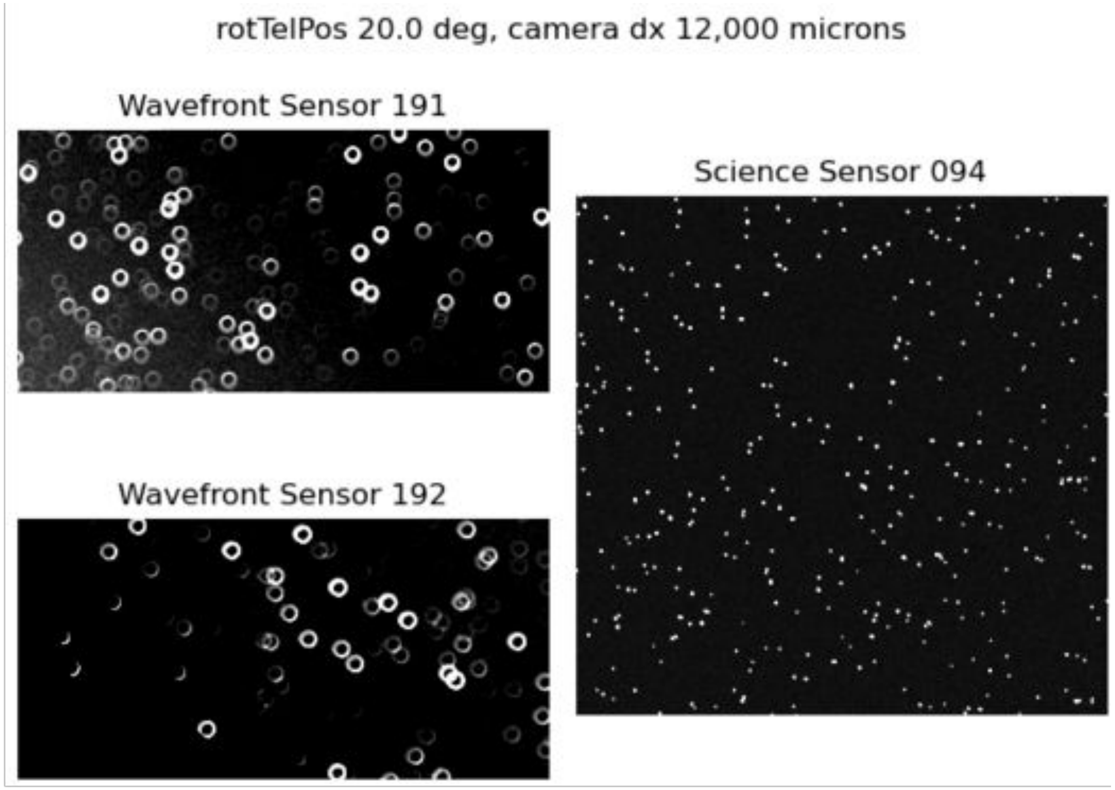
- This allows very important/interesting tests of the system.
- We are now using imSim to prepare for optical system commissioning.


See vignetting!

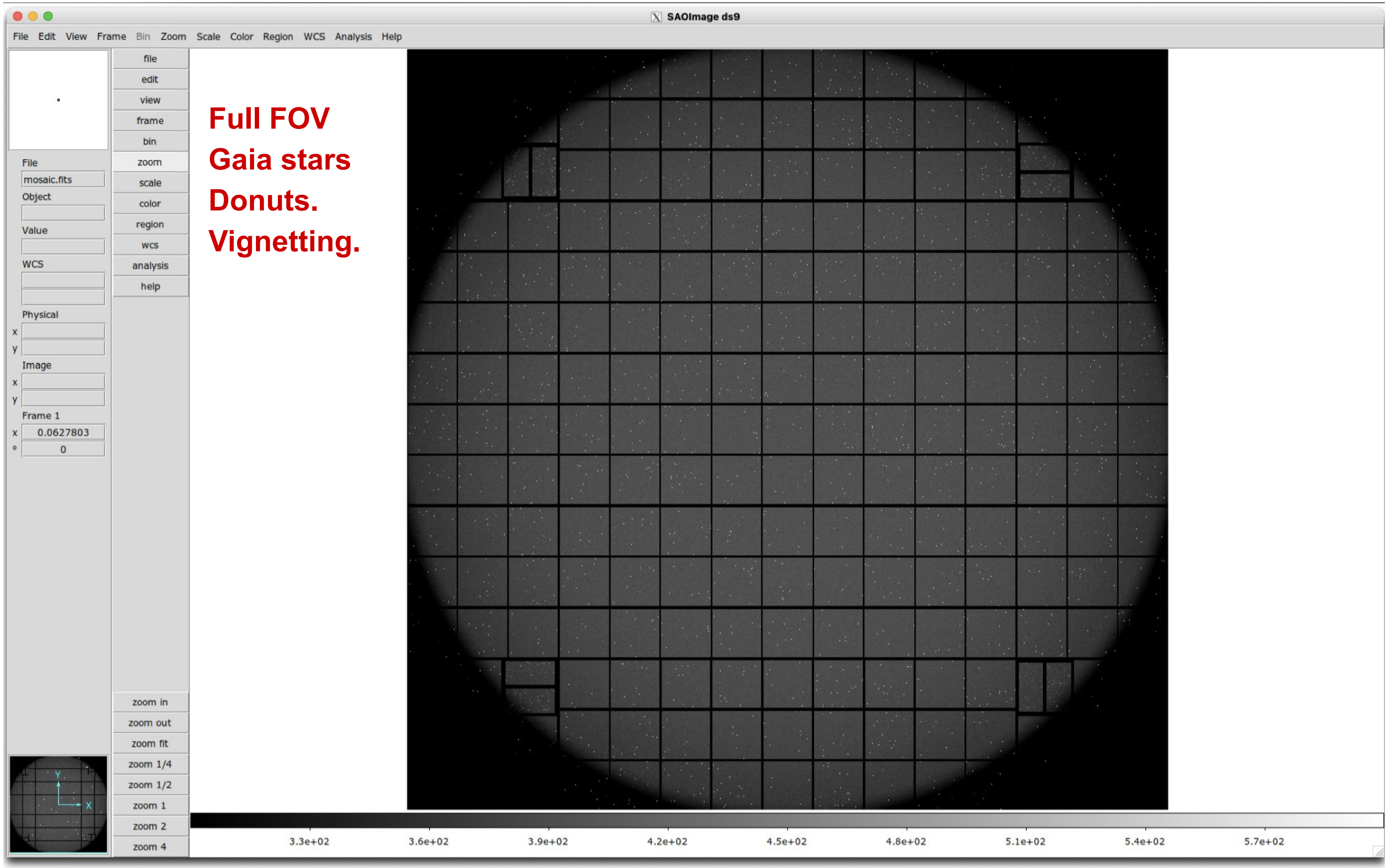
Chr

Now using to test the AOS commissioning pipelines.

Simulate and analyze out of focus "donuts" in the wave front sensors. Extract optical aberrations from the wavefronts, and determine how to move the optical degrees of freedom to correct for them. Use Gaia stars from skyCatalogs for realism.



Put all the pieces together!



File: mosaic.fits
Object:
Value:
WCS:
Physical x: y:
Image x: y:
Frame 1 x 0.0627803 ° 0

Full FOV

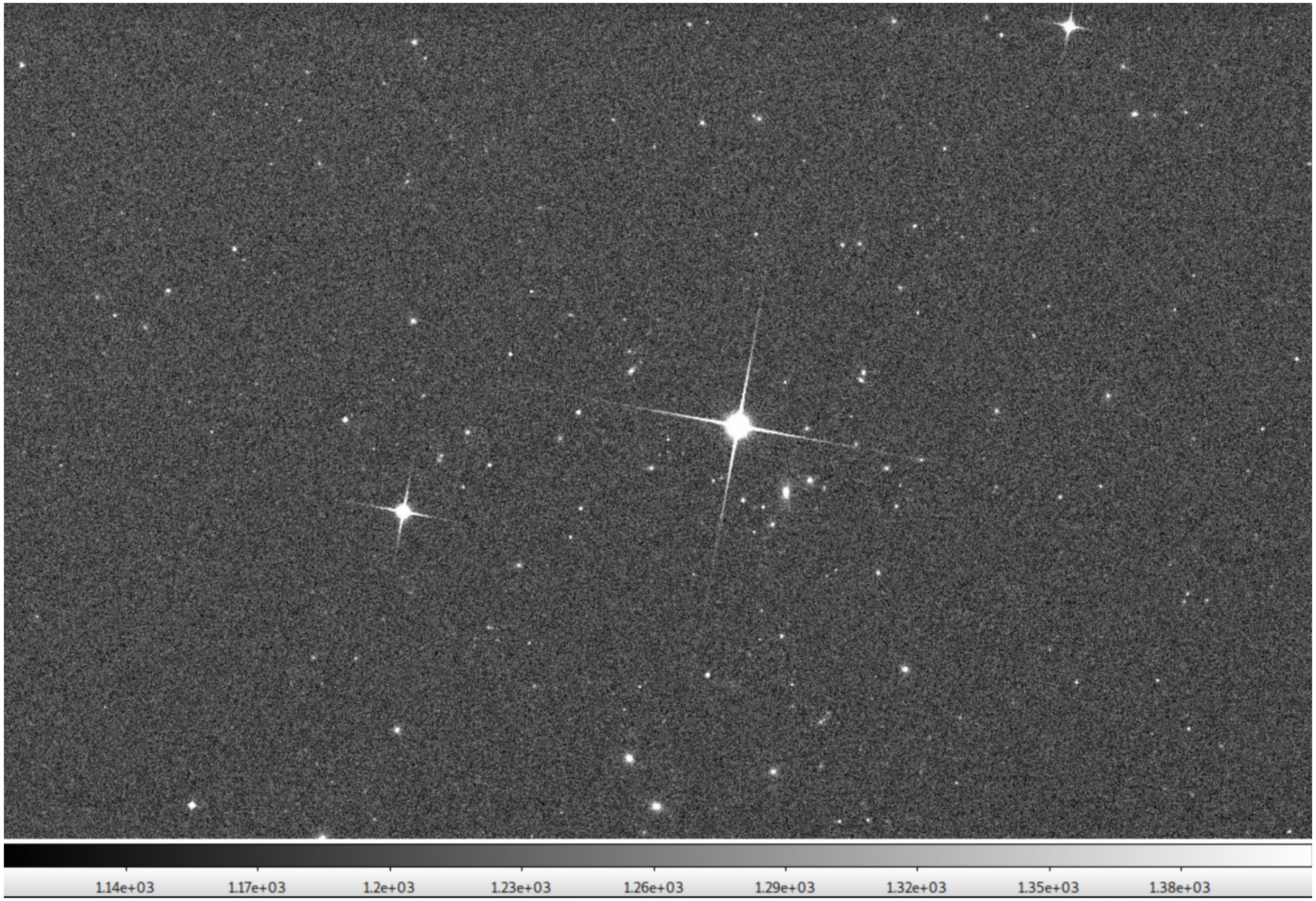
Gaia stars

Donuts.

Vignetting.

Realism example: Diffraction Spikes

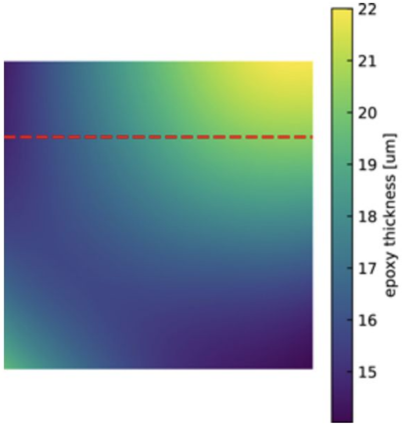
We even implement effects like the spiders rotating relative to the sky.

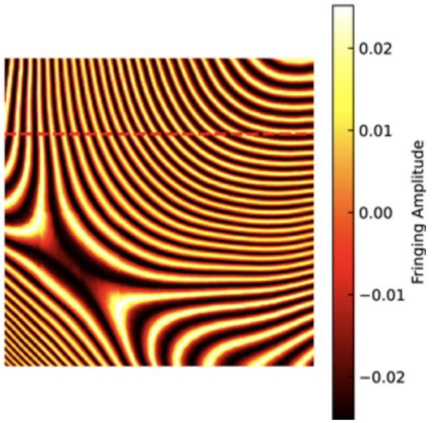


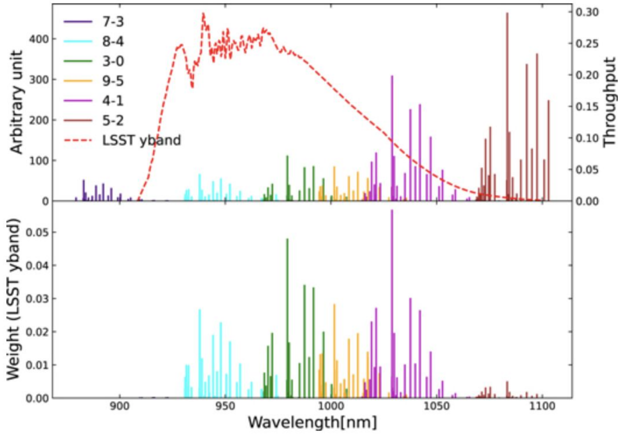
Josh Meyers
Geri Bräunlich

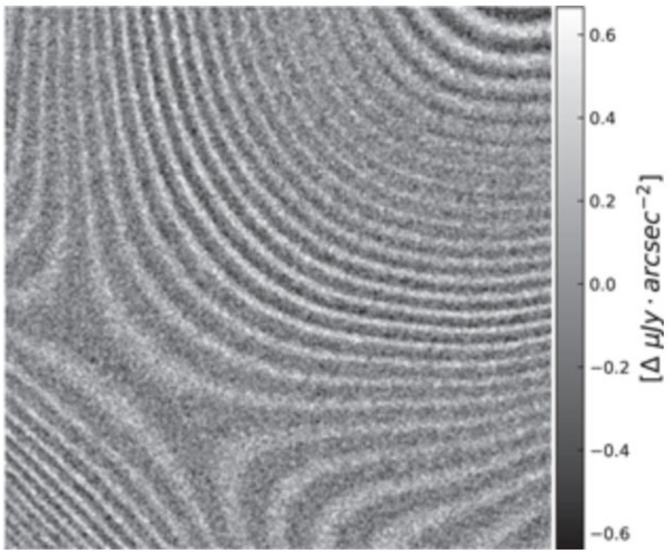
Another realism example: y-band Fringing

Sky background subtraction with fringing: important for LSB objects.





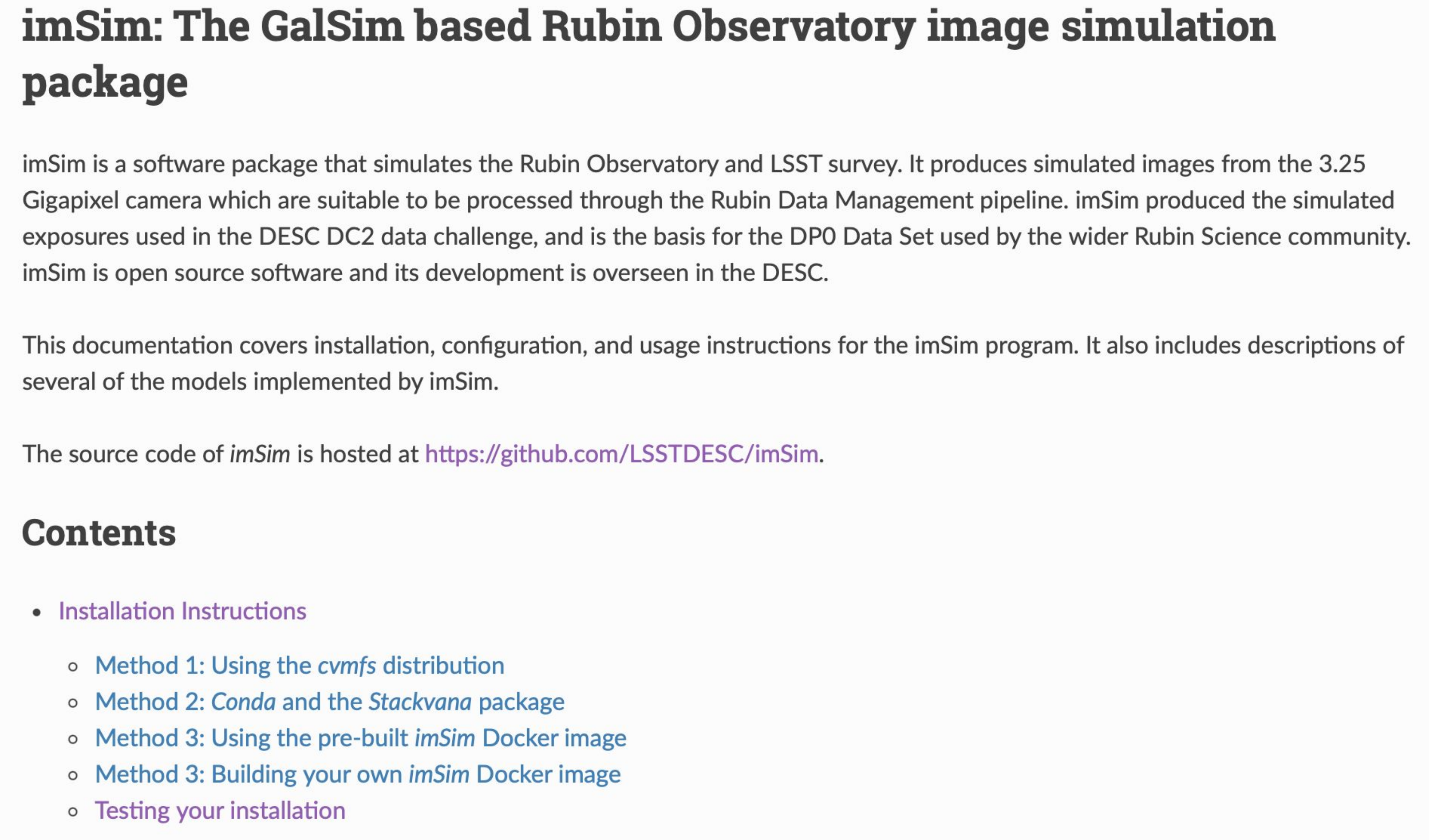




Student project: DOI 10.1088/1538-3873/acbe67

Zhiyuan Guo

Documentation



imSim: The GalSim based Rubin Observatory image simulation package

imSim is a software package that simulates the Rubin Observatory and LSST survey. It produces simulated images from the 3.25 Gigapixel camera which are suitable to be processed through the Rubin Data Management pipeline. imSim produced the simulated exposures used in the DESC DC2 data challenge, and is the basis for the DP0 Data Set used by the wider Rubin Science community. imSim is open source software and its development is overseen in the DESC.

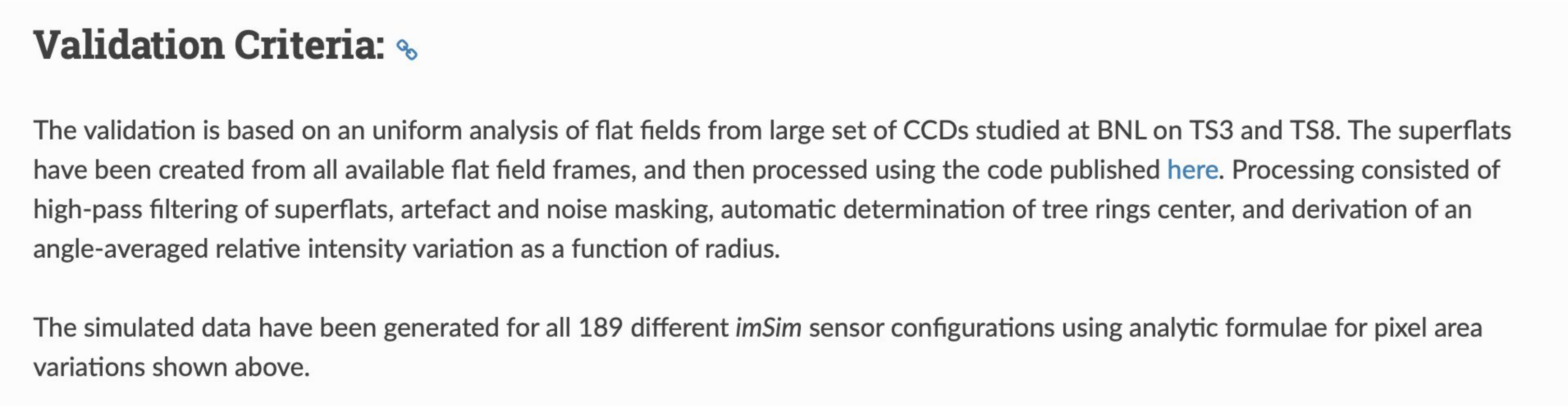
This documentation covers installation, configuration, and usage instructions for the imSim program. It also includes descriptions of several of the models implemented by imSim.

The source code of *imSim* is hosted at https://github.com/LSSTDESC/imSim.

Contents

Documentation: model details

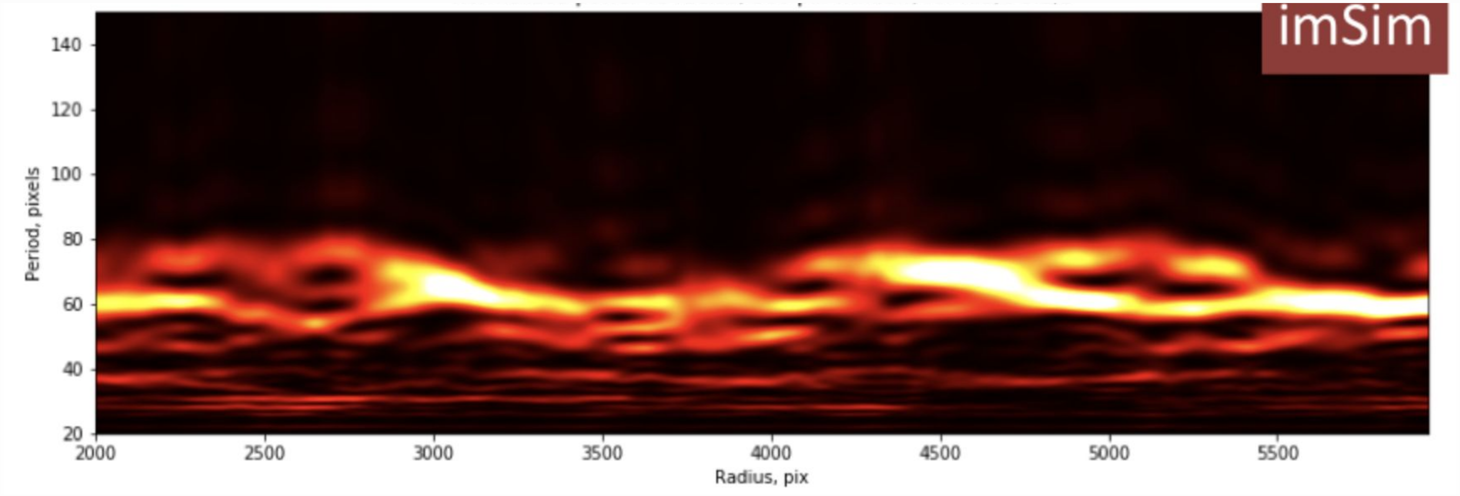
From Tree Ring model page.



Validation Criteria:

The validation is based on an uniform analysis of flat fields from large set of CCDs studied at BNL on TS3 and TS8. The superflats have been created from all available flat field frames, and then processed using the code published here. Processing consisted of high-pass filtering of superflats, artefact and noise masking, automatic determination of tree rings center, and derivation of an angle-averaged relative intensity variation as a function of radius.

The simulated data have been generated for all 189 different *imSim* sensor configurations using analytic formulae for pixel area variations shown above.



Comparison of radius-resolved periodograms of simulated and actual data

Some current realism work on our radar

https://github.com/orgs/LSSTDESC/projects/1/views/3

- Possible new features (**good student projects!**):
  - ELG galaxies
  - Satellites and other sky artifacts
  - Intra-cluster light
  - Clouds
  - Galactic cirrus
  - Sensor to sensor variations in the focal plane
  - Sensor inhomogeneity
  - Ghosts / Ghouls
  - Generative model galaxies
  - Varying sky model on a sensor size scale
  - Zenith dependent Atmospheric transmission curves
- Possible “applications”
  - Simulating ComCam
  - Simulating Collimated Beam Projector (CBP)
- **YOUR** science!

How can you use imSim?

Go to lsstdesc.org/imSim and follow the installation and usage instructions!

If you find issues etc, please file them at GitHub.

What are the opportunities?

- We are open to suggestions about basic features. Please let us know if something is not working or missing and we will consider it. **Please tell us!**
- There are many missing features that would be useful to the wider Rubin community that we don’t have the expertise or support to add (an obvious example is moving objects). **Please join us!**

Conclusions

- There is a **new release of imSim** with many new exciting features.
- imSim will be an exciting tool for **commissioning**, **planning** and **analysis** across Rubin!
- We welcome input and new developers who are interested in extending imSim.

Let’s discuss ideas and your questions in this session!