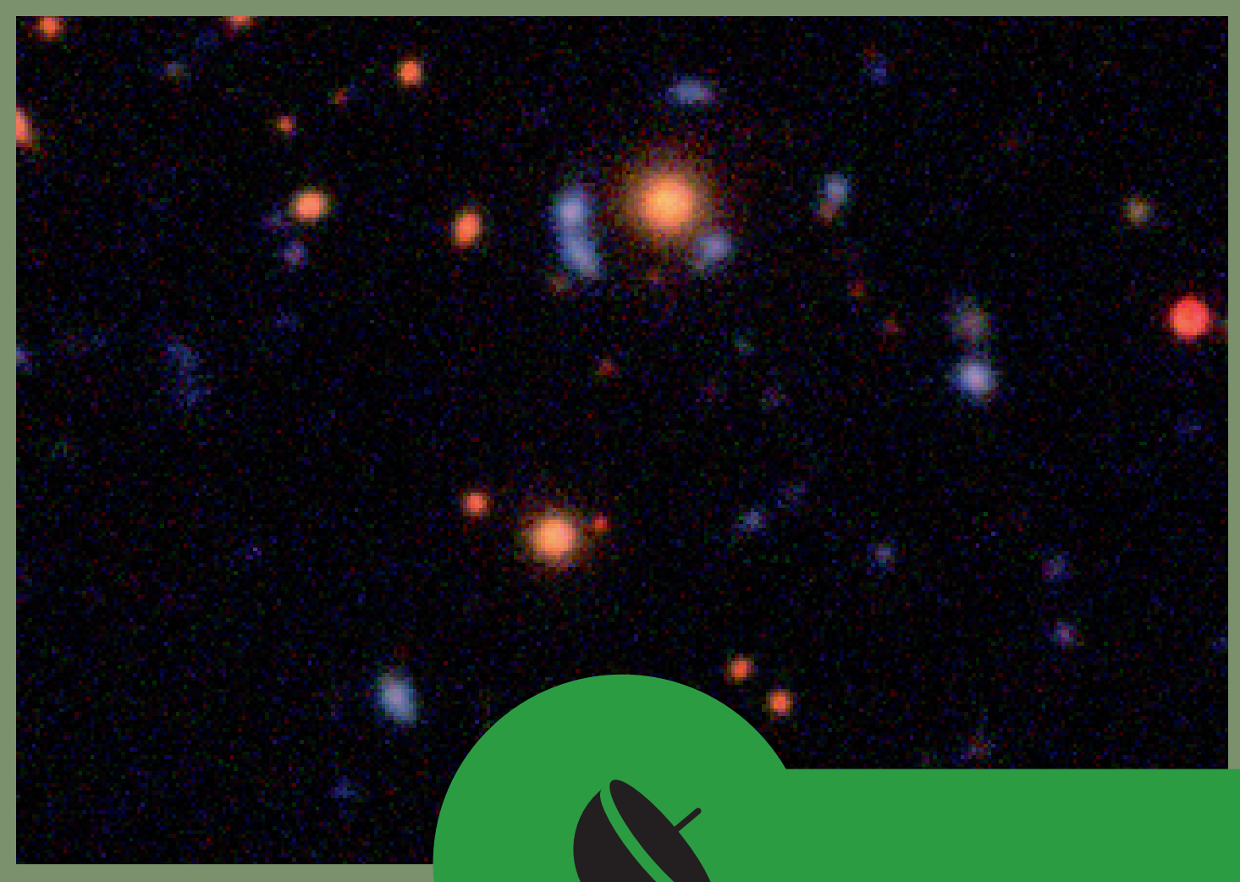
SpaghettiLens

A Citizen Science Powered Distributed Modeling Software Stack Written in Python

In the last three decades some 500 strong gravitational lenses have been discovered, and the number is set to increase 100-fold over the next decade. These new lenses will need mass models, and the traditional mode of professional astronomers modelling lenses individually will not be feasible. We have developed a new approach to lens modelling, which enables experienced but non-professional volunteers to make lens models collaboratively.

Citizen Scientists View

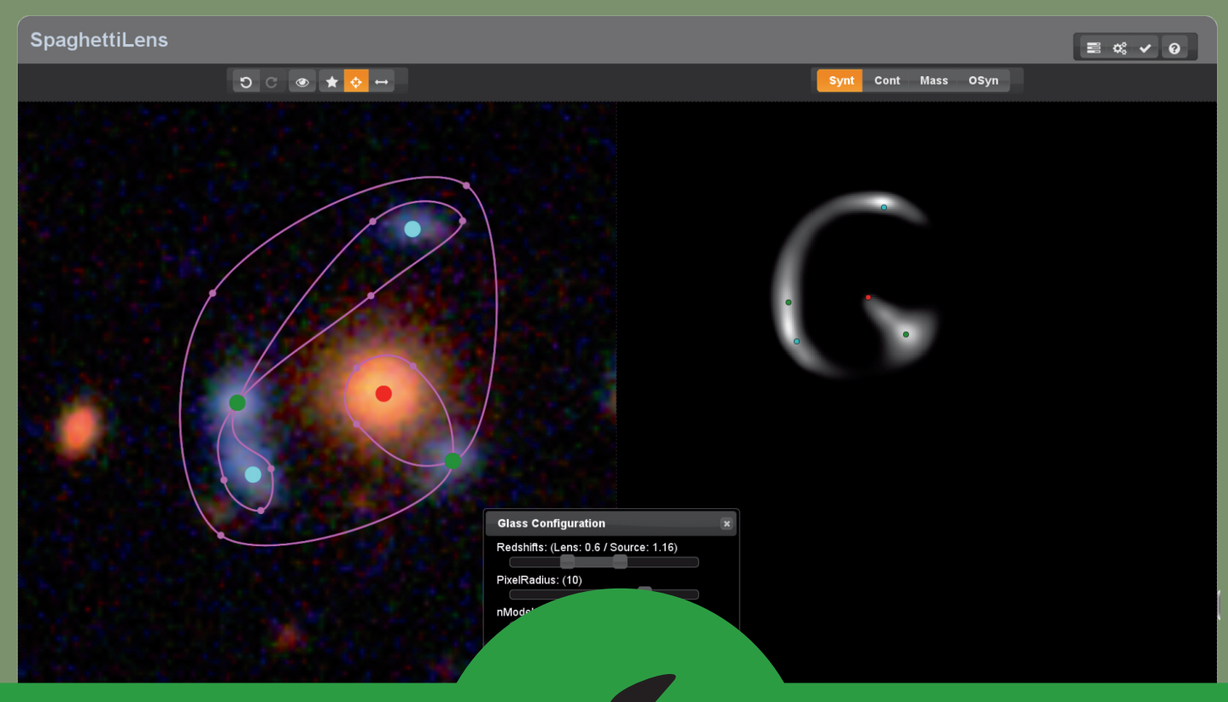


1 User chooses lens candidate and loads data

- Citizen Scientist identifies possible candidate lensing configuration
- Using the ID from the supported data repository, the user loads all available data into the UI
- Currently: mainly lensing candidates from Citizen science project SpaceWarps

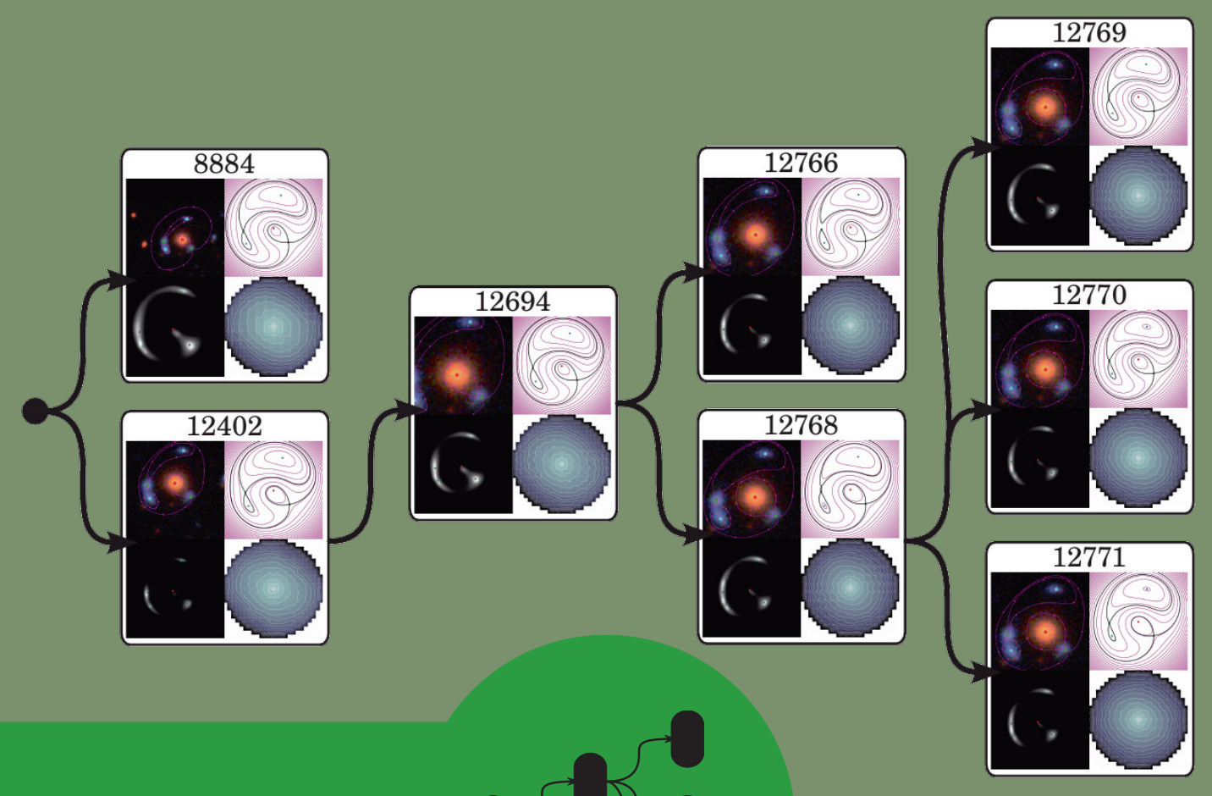
4 User chooses model configuration

- Volunteer adds markup to the survey image
- Parsing images according to Fermat's principle
- Based upon an educated guess and experience



6 Modelling and rendering

- Input model configuration is requested to be modelled according to model configuration
- Visual representation and further data plots from resulting model are requested



9 User receives model results in context

- User gets a link to their model that can be shared
- Those links can be used to revise models of others
- This lead to a tree of models
- Tree is pruned by discussion

2 Server requests data from remote data repository

- Server requests data from any available remote data repository
- SpagettiLens can be quickly extended with additional modules for future data repositories

Server
Django / Python

Broker
RabbitMQ

7 Broker distibutes workload

- To achieve a non blocking, asyncronous server, a task distribution system takes over computationally intensive modelling and rendering
- Central broker handels muliple worker nodes
- Easily scalable, "hot plug and play" additional nodes

Storage
CouchDB

Storage
File Cache

Worker
Celery / Python

Datasources

8 Worker starts GLASS and saves results

- Worker software runs on multiple nodes
- One modelling / rendering takes of order minutes
- Bottle neck of the application
- Generated data stored in database
- Created model cached
- Rendered images cached

3 Server caches and delivers data

- SpaghettiLens collects relevant data in its database
- Images get cached locally to prevent CORS issues and to enable the use of user uploaded files

5 Configuration stored in database

- Document-oriented NOSQL database

GLASS
Python / C
[JC et al., arXiv:1401.7990]

GLASS is a non-parametric (free form) gravitational modelling tool by JC.

It uses an under-constrained adaptive grid of mass pixels to model a gravitational lens, sampling ensembles of thousands of models to marginalise over model uncertainties.

Authors

Rafael Kueng
rafael.kueng@uzh.ch
Physik Institut
University of Zurich; Switzerland

Prasenjit Saha
Physik Institut
University of Zurich; Switzerland

Jonathan Coles
Exascale Research Computing Lab
Campus Teratec; France

Scientists View

10 Scientist analyses data

- Direct access to data in database and model files for post processing / analysis pipelines with Python through web API.
- Store results back in database
- Comparing recovered vs. actual Einstein radii of a set of simulated lenses to estimate voluneer performance [RK et al., 2015; arXiv:1502.00008]
- Follow up modelling on lens candidates of SpaceWarps project [RK et al., in prep.]

A special thank to all the SpaceWarps volunteers who helped finding lenses, creating the models, finding bugs and testing the software.

This work was supported by
the Candoc Forschungskredit
by the University of Zurich.

University of Zurich UZH