ScopeSim-Targets

A unified, ESO-compatible target description framework

Why?

The ELT is a G€-project, **each night is ~300 k€**. The ability to clearly communicate *what* is being observed with *which parameters* using simulations is crucial to avoid wasting observing time and thus money. The lack of a **standardized target description syntax** across observation simulations, exposure time calculations and observation planning is a major limitation for the ELT.

What and how?

Provide a unified, consistent and easy-to-use framework to describe astronomical targets, in the context of both the **ScopeSim** observation simulator and ESO's exposure time calculator and proposal tool infrastructure. Take full advantage of the whole-scene simulation capabilities provided by **ScopeSim** and subsequently directly reuse their target descriptions in ESO's proposal tools, creating a streamlined end-to-end workflow.

The backend is given by the needs of **ScopeSim** and the ESO target description., the frontend should be simple, **straightforward, self-documenting and consistent**. Instantiation from YAML and Python should be mostly identical, serialization into both YAML and FITS headers must provide round-trip capability. From this follows a "Readme Driven Design" to connect the target definition to a **ScopeSim.Source** object.

Why now?

Science team members of first-light ELT instruments are currently thinking about the objects they want to observe and the GTO time that they need to allocate for them. Providing a common language to describe these objects that can be **directly reused once ESO starts to plan actual observations** will streamline the whole process enormously and avoid friction between scientists and committees. The ability to connect observation proposals to detailed simulations will **minimize the failure rate** of such an expensive machine as the ELT.

Targets

Stellar

- **Star**
- **Star field**
- **Binary**
- Globular cluster
- Young cluster
- IMF sampler

Exoplanetary

- **Exoplanet**
- Disk
- Planetary system

Extragalactic

- **Sérsic profile**
- Mock spiral galaxy
- Galaxy cluster
- AGN host
- Multiple component galaxy

Star field

```
!StarField
band: R
positions:
  - [0, 1]
  - [2, -3]
  - [-2, 1]
  - [3, 0]
  - [1, -2]
spectra:
  - G2V
  - A0V
  - M2III
  - K5V
  - K2IV
brightness:
  - ["V", 12 mag]
  - 15 mag
  - 8 mag
  - 10 mag
  - 9 mag
```

Simple single star

```
!Star
position:
  x: 2 arcmin
  y: 1 arcmin
spectrum: A0V
brightness: ["R", 15 mag]
```

Binary physical separation

```
!Binary
position:
  distance: 100 pc
offset:
  separation: 0.1 AU
  position_angle: 50 deg
spectra: [F0V, M5V]
brightness:
  - ["R", 15 mag]
  - ["R", 20 mag]
```

System with two exoplanets and host

```
!PlanetarySystem
position:
  x: 2 arcmin
  y: 4 arcmin
  distance: 20 pc
primary:
  !Star
  spectrum: G2V
  brightness: ["V", 12 mag]
components:
  - !Exoplanet
    offset:
      separation: .3 AU
      position_angle: 50 deg
    contrast: 1e4
  - !Exoplanet
    offset:
      separation: 2 AU
    contrast: 5e5
    spectrum: "file://myspectrum.fits"
```

Single Sérsic profile mock elliptical galaxy

```
!Sersic
position:
  ra: 190.5 deg
  dec: 11.6 deg
spectrum: "spex:brown/NGC4621"
brightness: ["V", 9.6 mag]  # total brightness
params:
  r_eff: 46 arcsec  # effective (half-light) radius.
  n: 5.98  # sersic index
  ellip: 0.5  # E5
  theta: 163.3 deg  # position angle
  c: -0.35  # disky/boxy shape
```

Compatibility with the **ESO "Astronomical Targets at LPO"** standard for use in **Exposure Time Calculators**, the proposal tool ecosystem and data flow systems in general is a key design goal and will be implemented in a future version.

Defining target positions

All positional information is processed using astropy's excellent **SkyCoord**, fully harnessing the different formats and coordinate systems supported by it.

Defining target spectra

ScopeSim-Targets will make use of the extended spectral library of the **SpeXtra** Python package (also from A*Vienna), which contains templates of stellar and galactic spectra of various types. Simply specifying a spectral type for a star will lookup the next-closest spectrum in the template library. Spectra from local files are also supported.

Defining target brightness

Currently, brightness is specified as apparent magnitude in a given band (filter). Future versions will also support physical flux in a band as well as flux over an emission line profile and absolute magnitude plus distance.

Feature Roadmap:

Documentation:

ScopeSim-Targets will be integrated as an essential part into the ScopeSim Ecosystem as outlined in Haberhauer+ (2025).

It will replace the currently used ScopeSim_Templates package for all target definitions.

Source definitions for pipeline test data will also benefit from the standardized target description syntax.

Fabian Haberhauer, Hugo Buddelmeijer, Kieran Leschinski

References: Leschinski+ 2020, Haberhauer+ 2025

astropy-powered
astropy.org

universität wien