ROSACE

Version 0.0.1

Table of Contents

Contents:

Welcome to the ROSACE project

The goal of this project is to develop a network of amateur (and professional) observatories to run Robotic Spectroscopic astronomical observations. All together, we can offer a new tool to the Science to better understand our Universe.

This project is based on Open Source tools only.

Robotic observation means that **no human action** is required. From opening the dome (or shelter) to process the data and sharing the result, all is automated.

The project is developed by the community: any contribution from you is welcome. Our intention is to propose a system that can be adapted from smallest to biggest telescopes.

Introduction

This project proposes to setup your own robotic observatory for Astronomical Spectro observations.

A proposal : the name of the project is ROSACE, for **Robotic Observations and Spectrocopic Astronomy, a Collaborative Experience**

Our intention is that many observers, with different instruments and for various scientific programs can use this system. The final goal is that we can produce all together valuable data for the science, and at the end for better understand the Universe we're living in.

We do think that Spectroscopic observations **cannot be fully standard** . The observation sequence will depend on your own setup, and the data reduction will vary from one scientific goal to the other. You will probably have to 'put your hands on', and write some code to adapt the system to your own need. We want to make these adaptations as simple as possible.

To make this system accessible for everyone, we've organized with independant modules, with few general rules, and the code is intended to be easy to read and to maintain. Managing several small problems is better than a big one.

We work over the long term. People will change, instruments will change, new science opportunities will arise. The platform must be able to adapt to all these changes. To make this, we do our best to be clear on what is stable over the time (for instance the Observing File format), and what can evolve (the instruments, the observing sequences, and so on).

We think that continous improvement is the best way to move forward - one small step everyday.

The system is made of 5 independant modules:

- Module 1: The dome/shelter control
- Module 2: Shelduler (what is the next target to observe)
- Module 3: Sequencer (run the actual observation)
- Module 4: Data reduction (to get a scientific result)
- Module 5: Display and share the results

Get started

When the project will have enough maturity, we'll explain how to start from scratch.

How to Contribute

The project is developed by the community, and we need you!

Here are the ways to help developing this tool:

- Install your own observatory and give us any user feedback
- If you’re a developper, contact us and tell you where you can help (there is so much to do!)
- Write and/or translate the documentation (the plan is to get only an english version, but we never know)

Basic principles

The project is based on few principles:

- An observation is a consistent set of images and data, collected in an Observation File (yaml format)... or json? YAML pour les commentaires ?
- All files of one observing night are saved in a single folder
- All images and spectra are in FTIS format (which is THE format for astronomy files).
- We use only Open Source tools:
  - Linux.
  - Python (3.10 or above) as much as possible.
  - Git for revision control and sharing the code.
  - Rest API interfaces (to split front-end and back-end).
  - User interfaces in (JavaScript / TypeScript).
  - INDI server and devices (INDI is for Linux what ASCOM is for Windows).
  - Sphinx for the documentation.
  - Few standard Python libraries :
    - Astropy, for all Astronomy related stuff.
    - FastAPI for the Rest API interfaces.
    - MQTT and TIG (Telegraf / InfluxDB / Grafana) to reacord real time data.
    - Logging for the loggers (all actions are recorded).
    - PyTransitions (for Finite States Machine).
    - Cmd.cmd (for getting a command line interface (CLI), useful for debugging).
- The dome or shelter can open and close whatever is the telescope position.
- Each module can be installed locally or remotely (client-server architecture).
- We can use Kstars for monitoring the actions (very useful for debugging).

- We love the concept of continuous improvement ; a small step every day.
- There is a single configuration file (in yaml format) for each module.

The five modules

The project is made of 5 independant modules:

- Module 1: The dome/shelter control
- Module 2: Shelduler (what is the next target to observe)
- Module 3: Sequencer (run the actual observation)
- Module 4: Data reduction (to get a scientific result)
- Module 5: Display and share the results

Indepedant modules means that each one can be used independantly from the others. You can even use only some of them, and not all. Of course, the modules must exchange some data, and this is done through Rest API.

Here are more details for each module:

Module 1: The dome/shelter control

Module 2: Shelduler (what is the next target to observe)

Module 3: Sequencer (run the actual observation)

Module 4: Data reduction (to get a scientific result)

Module 5: Display and share the results

Module 1: Dome / Shelter manager

The module is part of the Obervatory

Module 2: Scheduler

The module is part of the Obervatory

Module 3: Sequencer

The module is part of the Obervatory

This module is the core of the Robotic observation. It runs all the steps of an observation sequence.

Each observation runs a given observing sequence. You can write your own sequence, to match your scientific need. The sequence is made of “bricks” ; you can use existing bricks or write your owns.

Here are the main (and required) principles for this module:

1. Each observation uses a specific sequence (mainly linked to the scientific goal of the observation).
2. A sequence is a single list of basic steps. No loops, no conditions, the system is very rustic.
3. A single structure is used to give each step all the data it needs (target star, exposure duration, and so on).
4. This structure can be append by each step.
5. This structure is also used at the end of the process to write the “Observing file”, needed for the data reduction.
6. We use a common logger system ; each step can log any data it wants.
7. Each step ends by a standard ‘return’, that allow the sequence to continue, or to stop if a problem occured.

Module 4: Data Reduction

The module is part of the Obervatory

Module 5: Display, share and archive

The module is part of the Obervatory

Observing file format

Each observation is fully defined by its Observing File. This file is in yaml format (human readable, easy to edit).

Here is the standard format for the Observing File:

— name: Vega RA: 18h36m56.33635s DEC: “+38d47m01.28021975s” nb: 3 exptime: 5 x1: 100 y1: 250 x2: 1500 y2: 1400 seq: BeUVEX

Indices and tables

- Index
- Module Index
- Search Page