ROSACE

Version 0.0.1

Table of Contents

Contents:

Introduction 4

Get started 5

How to Contribute 7

Basic principles 8

Module 1: Dome / Shelter manager 9

Module 2: Scheduler 10

Module 3: Sequencer 11

Module 4: Data Reduction 12

Module 5: Display, share and archive 13

Observing file format 15

Welcome to the ROSACE project

ROSACE is the acronym for **Robotic Observations and Spectrocopic Astronomy, a Collaborative Experience**

This project aims to develop a network of amateur and professional observatories to run Robotic Spectroscopic astronomical observations. All together, we offer a new tool to the Science to better understand our Universe.

This project is based on software Open Source tools only.

Robotic observation means that **no human action** is required. From opening the dome (or shelter) to processing 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.

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.

We also consider that the system must be able to manage several science programs ; none can use a telescope at 100% of the time. A science program can have its own observing sequence and own data reduction process. in ROSACE, each observation must be part of a given scientific program.

One key of a robotic observatory is that it really runs fully automatically. If each observation requires 2 minutes of human brain, it will quickly be overloaded - because every night can produce dozens of observations. Then a full automatic process is required. The only human part can be the validation, to make sure that all went well - the system must give all the tools to do it very fast.

To make this system accessible for everyone, we've organized the project 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.

To help you starting with ROSACE, we propose this documentation, and a Get Started page . You can start with simulators, no need for a real instrument in a first step.

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 five independant modules. Indepedant means that each one can be used without the others. You can also use only some of them, and not all. Of course, the modules must exchange some data, and this is done through Rest API.

- 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, validate and share the results

We do think that the success of this project depends on few elements :

- The community (developers and users) must be large enough ; the system must not depend on one or few guys.
- A clear and up-to-date documentation is available.
- We propose a ‘get started’ path, to easy jump quiclky and easily into the project.

Get started

(this part is to be developed when the project will have enough maturity).

We propose here a step by step path to satrt with ROSACE.

We suggest to start by installing the module 2 (scheduler) and module 3 (observing sequencer). You'll use these modules in Simulators mode - no real instrument is needed.

(installation will be described)

Then, edit the configuration file, and check that the modules 2 and 3 have the right parameters.

Use your favorite web browser, and point it to module 3. You can start the observations.

... to be completed.

Once the Simulators mode is OK, you can switch to the real instrument. To do this, you must give the address of the INDI server of your observatory.

... check that you can control the science camera, then the telescope, and all the connected devices.

... to be completed

How to Contribute

Welcome to the ROSACE project .

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

Welcome to the ROSACE project .

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 [3] .
  - Python (3.10 or above) as much as possible.
  - Git for revision control.
  - Github repo to share the code.
  - Rest API interfaces (to split front-end and back-end).
  - User interfaces in (JavaScript / TypeScript). Works with any web browser.
  - 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 [4] .

- 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.

Footnotes

[3]
We know that most of people ar using Windows. The door is not closed. Since we use Python (which is not platform dependant) and ASCOM (under Windows) platform offers the same kind of feature than INDI (under Linux), this can probably be adapted in close future - despite Windows is not Open Source.

[4]
This is a simple and efficient way to split functions and protect the instrument. But we keep in mind that this condition is not made by lots of observatories (most of the rolling shelter ones), and there is no strong stopper to this.

Module 1: Dome / Shelter manager

The module is part of the ROSACE project

This module is made to manage the Observatory infrastructure: Dome or Shelter, and weather station.

It is able to decide wether we can start the observation, based on the weather station. When all is OK, it opens the dome / shelter and

gives the order to the Module 3 (sequencer) for starting the observation.

It also decide to stop observations if weather conditions are not met anymore (or simply if the night is over).

In few words, mission of the Module 1 is to Protect the Instrument permanently.

Module 2: Scheduler

The module is part of the ROSACE project

Getting an observatory that is able to work automatically is good. But then, as soon as it is operationnal, the key question is: ‘what is my next target’.

This is the mission of the Module 2 to reply this question, and define at any time what is the next target to observe.

The strategy can be very simple - for example, you can set a list of targets in a text file.

But is can be also very rich if you have many observatories working together, managing multiple scientific programs. In such cases, the question of the next target can be tricky. This module will adapt to any cases - strating by the simple one, of course.

Module 3: Sequencer

The module is part of the ROSACE project

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 simple.
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 logging system ; each step can log any data you want.
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 ROSACE project

Once the observing data is recorded, and the observation is completed, we must process it, to get the scientific result, corrected from any instrumental effect.

This is the job of this Module 4.

Of course, the data reduction process depends on each scientific program.

In a first time, we'll do this reduction step thanks to SpecINTI software. Then, we'll develop our own process - fully Open Source, in Python.

The data reduction process is a list of reduction operations (like master dark, spectrum extraction or wavelength calibration). ROSACE will propose a list of such operations, but you can write your owns, to adapt to your case.

Module 5: Display, share and archive

The module is part of the ROSACE project

At the end of the observing process, we must share our results and close the observation.

This module will do it, with some important steps:

1. Validate the data by the observer (we never know, something may have happened).
2. Send the data to the relevant repository. Depending on the science case, it can be a database, as simple mail, or any other way.
3. Archive the data (raw data and result), and fullfill the local database, to make any future search fast and easy.

In a first time, we can make it manually, but as soon as the production increases, it must become automatic.

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