Astronomical Data Analysis Software and Systems XXVI
ASP Conference Series, Vol. 521
Marco Molinaro, Keith Shortridge, and Fabio Pasian, eds.
© 2017 Astronomical Society of the Pacific

Towards Robotic Operation With the First G-APD Cherenkov Telescope

Maximilian Noethe,[1] Dominik Neise,[2] and Sebastian A. Mueller[2], for the FACT Collaboration

[1]*TU Dortmund, Dortmund, Germany;* `maximilian.noethe@tu-dortmund.de`

[2]*ETH Zurich, Zurich, Switzerland*

Abstract. The First G-APD Cherenkov Telescope is an Imaging Air Cherenkov Telescope operating since 2011 at the Observatorio del Roque de los Muchachos. One of the major goals of the FACT collaboration is to achieve robotic operation of the telescope. Since 2011 FACT has been operated remotely. To reduce the necessity of human intervention, several programs were developed, most notably the `shifthelper` together with the `pycustos` library. This software monitors the telescope system and environmental conditions and calls the 'shifters' (remote operators) in case human intervention is required. This will lead to FACT being the first IACT with all shifters asleep during regular observations. The software presented here is open source and under MIT License.

1. The First G-APD Cherenkov Telecope

FACT, the First G-APD Cherenkov Telescope, is an Imaging Atmospheric Cherenkov Telescope (IACT) located at the Observatorio del Roque de los Muchachos on the Canary Island of La Palma (Anderhub et al. 2013; Biland et al. 2014).

After starting operation in October 2011, the FACT collaboration pursued three major goals: Continuous monitoring of the brightest gamma-ray sources, pioneering the Silicon Photomultiplier technology in gamma-ray astronomy and achieving not only remote, but robotic operation of the telescope.

2. Towards Robotic Operation

After a brief period of on-site operation, the FACT Collaboration achieved complete remote operation of the telescope. Two web interfaces are used to plan and execute data taking. First, a scheduler to plan the observations[1] and second, the smartfact interface[2], a status and control interface (Bretz et al. 2014).

The necessity for human interaction was reduced step by step. It was achieved that 'shifters' (the term used at FACT for remote operators, working shifts) only have to startup the telescope in the evening, monitor the telescope and the environmental conditions during observation and shut down the telescope in the morning. Human interaction during data taking is only needed in exceptional conditions like strong winds or heavy rain. This automation led to more than 2300 hours of data taken in the last twelve months.

---

[1] `https://www.fact-project.org/schedule`

[2] `http://fact-project.org/smartfact`

3. The pycustos library and the shifthelper program

To reach the target of robotic operation, the need for humans to monitor the telescope system and the environmental conditions needed to be reduced even further. For this purpose, the shifthelper program was created, written in python. The shifthelper performs regular checks of the system status and environmental conditions and calls the shifter in case human intervention is needed via a web service called twilio[3]. The program can also send texts and images, e. g. plots, via the Telegram[4] messenger.

For about a year, the shifthelper was used by the shifters on their computers. Currently, it is further developed to be a web service, automatically getting the information whom to notify in case human interaction is necessary from a database. A first version is being deployed and evaluated.

The infrastructure performing checks and notification was moved out into its own library called pycustos[5]. More possibilities to send notifications have been added, including sending emails via smtp, posting alerts to a REST-interface via http and basic logging to a file.

Here is a basic example how you could use the custos library to automatically call the fire department in case your house is on fire:

First a FireCheck class is implemented, inheriting from the pycustos abstract base class IntervalCheck. The only method we need to override is the check method. It creates a message via its critical method, in case our house is on fire.

```
from custos import Custos, IntervalCheck, TwilioNotifier
import my_house


class FireCheck(IntervalCheck):
    def check(self):
        if my_house.is_on_fire():
            self.critical('The house is on fire!')

if __name__ == '__main__':

    twilio_notifier = TwilioNotifier(
        sid='<your twilio sid>',
        auth_token='<your twilio token>',
        number='<your twilio number>',
        recipients=['112'],  # or '911'
    )

    fire_check = FireCheck(interval=30)

    with Custos(
            checks=[fire_check],
            notifiers=[twilio_notifier]
            ) as custos:
        custos.run()
```

To place calls, an instance of the TwilioNotifier is created, which is given the necessary credentials to use the twilio service and a list of recipients, for now just with one entry. In

[3] https://www.twilio.com

[4] https://telegram.org

[5] https://github.com/fact-project/pycustos

the main program, the `FireCheck` is instantiated and given an interval of 30 seconds between subsequent checks.

Both checks and notifiers are bound together by the `Custos` class, which runs as an event loop after calling `custos.run`. To allow more complex use cases, recipients of a notifier can either be a simple `list`, a `dictionary` mapping categories to lists of recipients or a function that takes the category and returns a list of recipients. Such a function is used to query the current shifter or the responsible expert in the `shifthelper` program. An image can be attached to a message and notifiers able to send images, e. g. the `SMTPNotifier` and the `TelegramNotifier` will relay these images to the recipients.

4. Additional tools

4.1. The `darkspot` program

FACT is performing measurements of trigger rate vs. trigger threshold each night to monitor atmospheric properties. These measurements are done pointing to a dark spot in the sky close to zenith. To find such a spot and schedule the measurement the shifter had to select a suited pointing position by hand, e. g. using software like `stellarium`[6]. To automatize this task, the `darkspot`[7] program was created. Using the Hipparcos star catalogue (Perryman et al. 1997), the area with the lowest light flux for the field of view of FACT is selected.

4.2. The `la_palma_overview` program

To get a quick overview about the conditions of a night, the `la_palma_overview`[8] program was developed. Continuously merging the streams of several webcams on the Roque de los Muchachos, weather information and system status reports, it creates a timelapse video.

Acknowledgement

This work has been supported by the DFG, Collaborative Research Center SFB 876, project C3 (`http://sfb876.tu-dortmund.de/`).

References

Anderhub, H., et al. 2013, JINST, 8, P06008
Biland, A., et al. 2014, JINST, 9, P10012
Bretz, T., et al. 2014, in Real Time Conference (RT), 2014 19th IEEE-NPSS
Perryman, M. A., et al. 1997, Astronomy and Astrophysics, 323

[6] `http://www.stellarium.org/`

[7] `https://github.com/fact-project/darkspot`

[8] `https://github.com/fact-project/la_palma_overview`