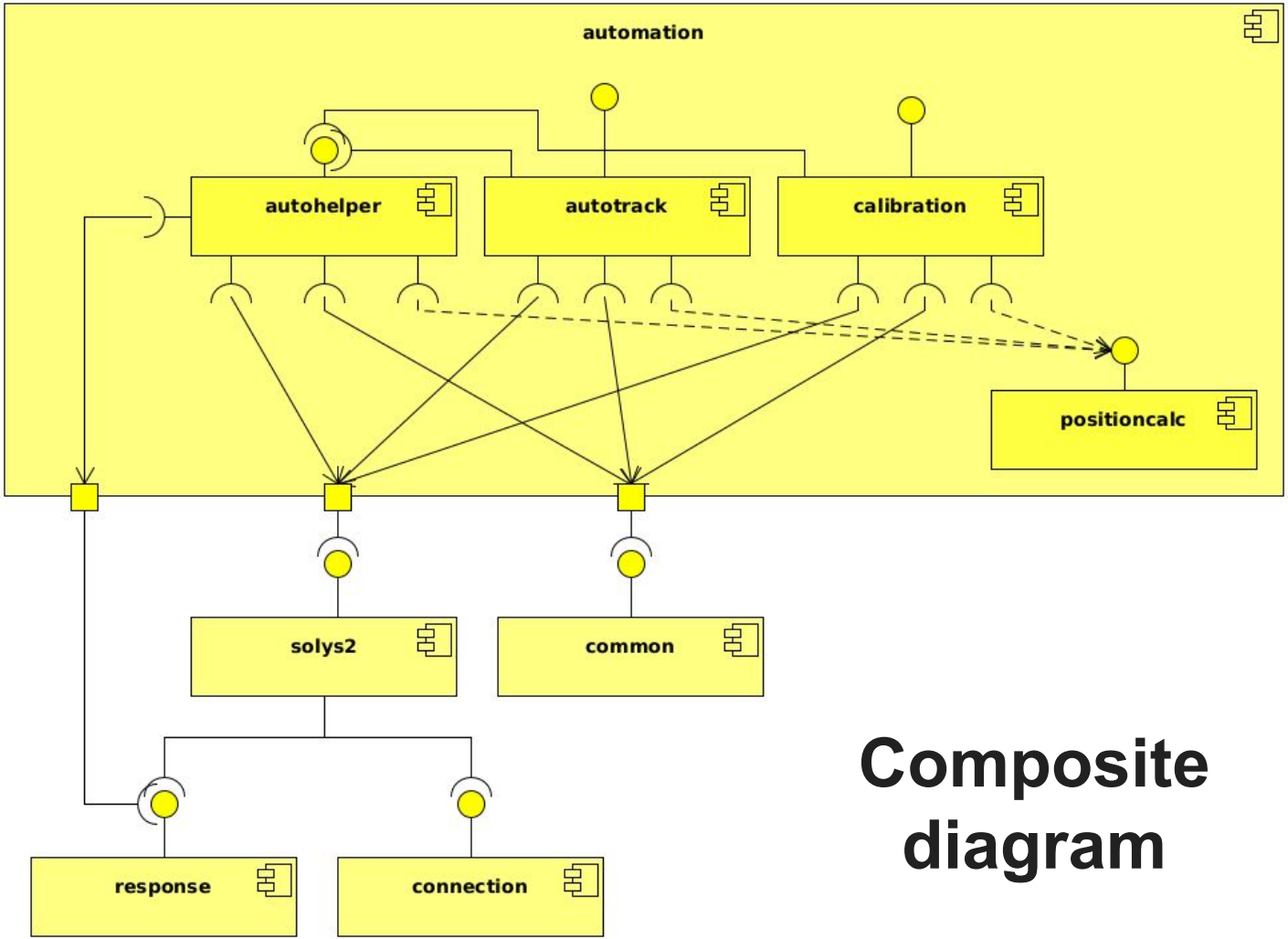
solys2

Python package for communicating with the Solys 2 via TCP/IP and tracking the Moon or the Sun.

v0.2.6

Scope

- Communicating with the Solys 2. → solys2.solys2
- Tracking the Moon (and the Sun). → solys2.automation.autotrack
- Perform calibration functions. → solys2.automation.calibration



Composite diagram

Communicating with the solys2 | Interface

- solys2.solys2.Solys2 object
- Call functions, or send the raw command with send_command()

```python
from solys2 import solys2

# Connect with the Solys2
solys = solys2.Solys2(ip, port, password)

# Tell the Solys2 to point at azimuth 30.
solys.set_azimuth(30)

# Get the current position at which the solys is pointing.
az, ze, _ = solys.get_current_position()

# Send the command "HO" to the Solys2
output = solys.send_command("HO")
# Another option would have been calling solys.home()

# Obtain the status of the Solys, the activated flags and the deactivated flags.
status, act_flags, deact_flags, _ = solys.get_status()
```

Communicating with the solys2 | How does it work?

- An attribute of type solys2.connection.SolysConnection handles the communication.
- The module solys2.response is used to understand the received raw data.
- Every response gets transformed to a solys2.solys2.CommandOutput dataclass instance.

Tracking the Moon (and the Sun) | Interface

- SunTracker and MoonTracker objects.
  - Once stopped they cannot be started again

```
from solys2.automation import autotrack
from solys2.automation import positioncalc as psc
from solys2 import common
import logging

logger = common.create_default_logger(logging.DEBUG)
# Track the sun, sending a new position each 15 seconds, and logging the
# information (movements, etc) to stdout.
st = aut.SunTracker(ip, 15, port, password, logger, psc.SunLibrary.PYSOLAR)

# Start tracking
st.start()

# Stop tracking the Sun
st.stop()
```

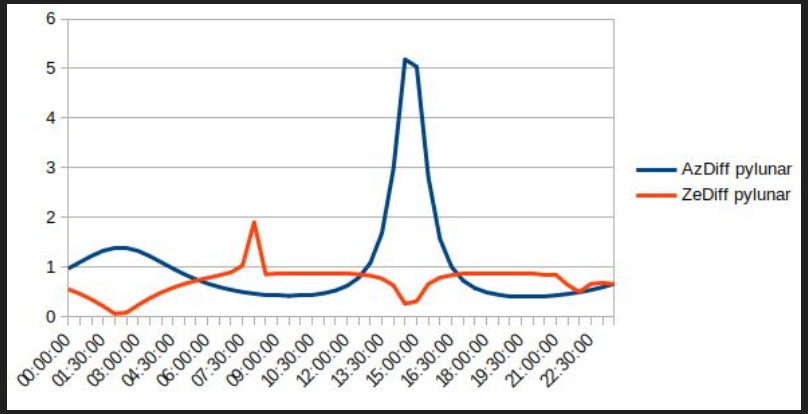
Tracking the Moon (and the Sun) | Coordinates

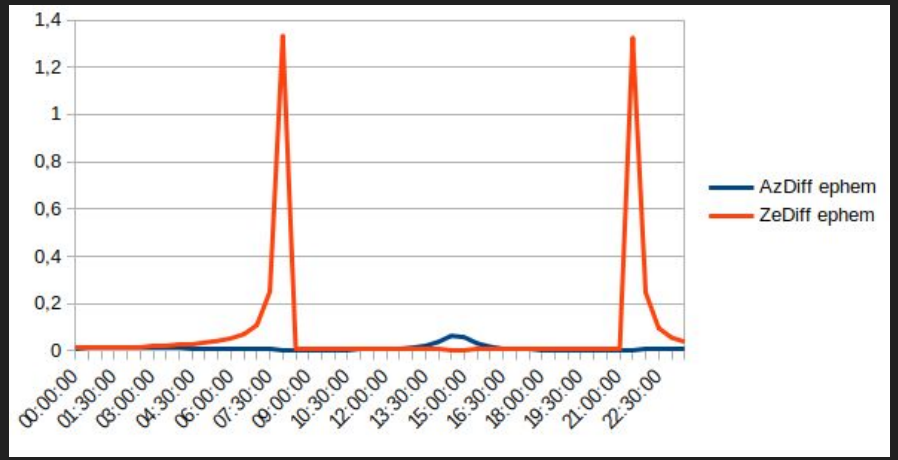
- The user is allowed to choose the library from which to obtain the coordinates:
  - SUN: ephem, pysolar or spicedmoon. Default: pysolar
  - MOON: pylunar, ephem or spicedsun. Default: ephem
- SPICE is a library created by NASA
  - Pros: Very precise
  - Cons: Somewhat slower, performs disk operations and sometimes fails (although very rarely)
  - In order to solve the rare errors, it’s possible to choose SPICE<SUN/MOON>SAFE which will use SPICE but in case it fails it will use the default library as a backup.

Tracking the Moon (and the Sun) | Coordinates

Comparing lunar libraries with SPICE:

- pylunar data is too different.
- ephem is different on moonset and moonrise. Aberrations.

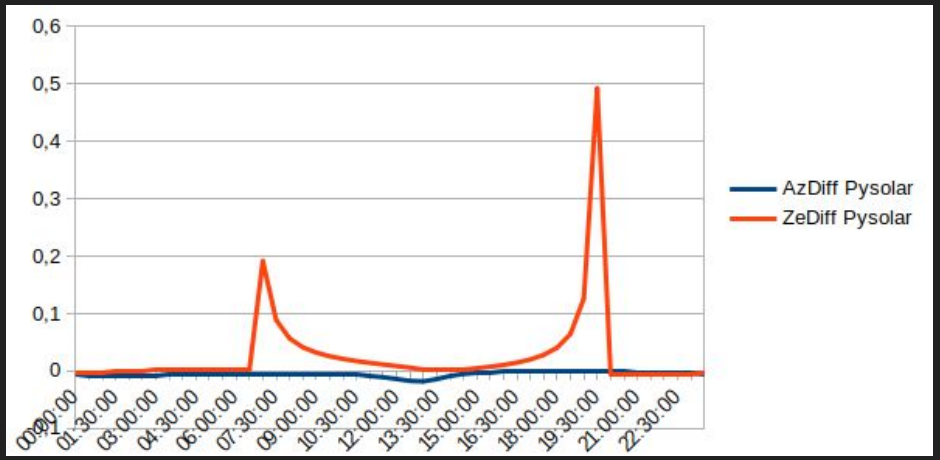


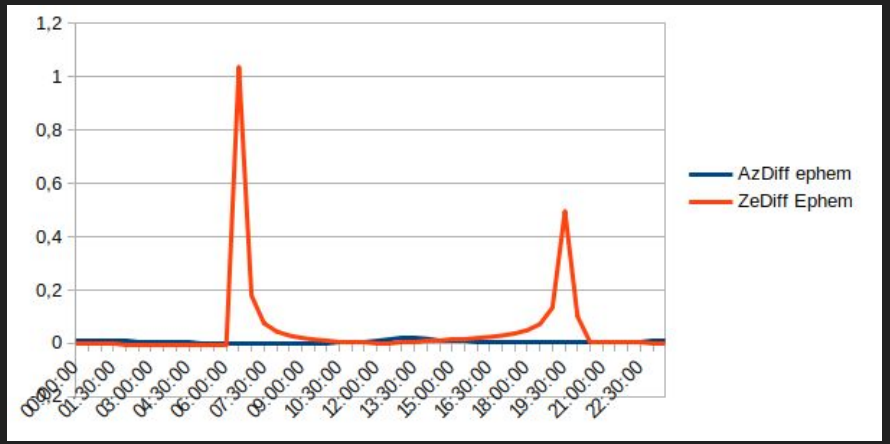


Tracking the Moon (and the Sun) | Coordinates

Comparing solar libraries with SPICE:

- ephem has approx 1º max. diff.
- pysolar has approx 0.5º max. diff.



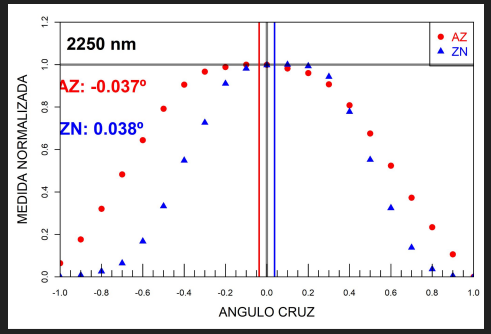


Difference on sunset and sunrise.

Aberrations.

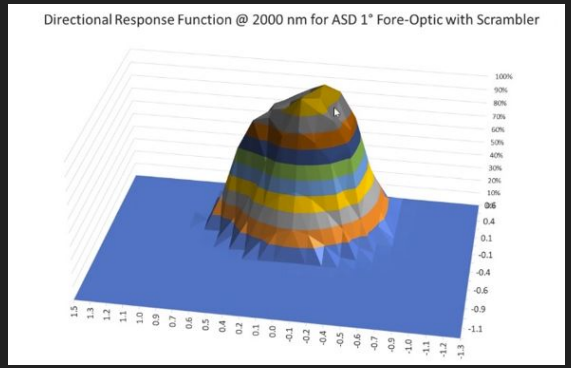
Calibration | Interface

- Cross and Mesh/Matrix




```
from solys2.automation import calibration as cali
from solys2.automation import positioncalc as psc
from solys2 import common

cp = cali.CalibrationParameters(-1, 1, 0.1, -1, 1, 0.1, 5, 1)
logger = common.create_default_logger(logging.DEBUG)
library = psc.MoonLibrary.EPHEM_MOON
lc = cali.LunarCross(ip, cp, library, logger)
lc.start()
```




```
from solys2.automation import calibration as cali
from solys2.automation import positioncalc as psc
from solys2 import common

cp = cali.CalibrationParameters(-1, 1, 0.1, -1, 1, 0.1, 5, 1)
logger = common.create_default_logger(logging.DEBUG)
library = psc.SunLibrary.PYSOLAR
sc = cali.SolarMesh(ip, cp, library, logger)
sc.start()
```

Calibration | Interface | Measure with instrument

- Countdown printed out on log, or will execute callback.

```
from solys2.automation import calibration as cali
from solys2.automation import positioncalc as psc
from solys2 import common

cp = cali.CalibrationParameters(-1, 1, 0.1, -1, 1, 0.1, 5, 1)
logger = common.create_default_logger(logging.DEBUG)
library = psc.MoonLibrary.EPHEM_MOON
lc = cali.LunarCross(ip, cp, library, logger)
lc.start()
```

```
from solys2.automation import calibration as cali
from solys2.automation import positioncalc as psc
from solys2 import common

cp = cali.CalibrationParameters(-1, 1, 0.1, -1, 1, 0.1, 2, 0)
logger = common.create_default_logger(logging.DEBUG)
library = psc.MoonLibrary.EPHEM_MOON
lc = cali.LunarCross(ip, cp, library, logger, inst_callback=measure)
lc.start()
```

License: GPLv3

- Out library is freely available for everyone
- If someone made an improvement, we’d want to be able to use it.

Constraints

- Only works over TCP/IP (ethernet cable), not with Serial port.
- It might not work on the Southern hemisphere.
  - Sun goes over 0º of Azimuth, which would cause the Solys2 to perform an almost 360º spin, right?

Useful links

- Full documentation: solys2.readthedocs.io
- GitHub repository: github.com/GOA-UVa/solys2
- My email: gaton@goa.uva.es


Thank you!
SOLYS2