Visibility plot

I assignment of *Astrofisica Osservativa*

Bernardo Vettori

February 16, 2025

Abstract

The aim of the assignment is the implementation of a script to compute a visibility plot. A target's visibility is related to its apparent place in the sky and the relative position of the Sun and Moon. The Python function `visibility()` of the library `visibility` provides a routine to compute when (and for how long) a celestial object is visible from a chosen observatory on a selected date. Using this library, we implement a Python script that returns the visibility plots of selected targets. The results are discussed and compared with those from another software (*Stellarium*), showing a good accordance, although the work is just qualitative.

1 Introduction

The first assignment of *Astrofisica Osservativa* involves computing the visibility of a celestial object, that is, whether it is visible from a certain location on a given date and for how long.

The Python library `visibility` provides the required task. The description of the used procedure and the implemented modules is reported in Section 2. To test the correct functioning of the program we implement the Python script `script.py`, which computes the visibility plot for five different targets and three test cases. The results are shown in Section 3 and discussed in Section 4. Finally, Section 5 reports a review of the work.

The project directory also contains a collection of data, which are used in some routines, such as the conversion from *Universal Time* (UT) to *Dynamical Time* (TD) or the calculation of the ecliptical coordinates of the Moon; some comments about it are reported in Appendix A.

2 The Task

2.1 The visibility of a target

The question to be answered is: *when is an object visible, given the place and date of the observation?* Or better *when is an object not visible?* This is provided by the python function `visibility()`, which computes the visibility of a target during an entire night. Several factors influence the possibility of observing a celestial object, but only the positions of the target, Sun, and Moon in the sky are considered in this report.

The apparent place

The target's position on the celestial sphere is the first aspect to consider when studying visibility. The corrections to pass from the mean to the apparent place are all listed in [4–6], but for this work only the ones for the *proper motion*[1], the *precession*, the *nutation* and *annual aberration* are taken into account. The effect of *atmospheric refraction*[2] is considered too.

Then to establish for how long the target is visible the time of its rising and setting is computed: in brief, the routine estimates the time of the transit as a fraction of a day and computes the moment of the rise and set concerning it (see [4]). If the object never sets or never rises, the function returns a Boolean value (`True` for the first case and `False` for the latter) in addition to the time of the transit.

Sunrise and sunset

Observation during the daytime is impossible, so it is necessary to know when the Sun rises and sets. However, it is more important to estimate the twilight duration (after sunset or before sunrise) that causes residual sunlight due to atmospheric scattering [5, 6]. We chose to adopt the definition given in [1]: astronomical twilight lasts as long as the (apparent) altitude of the Sun is above $-18°$ (or the zenith distance is lower than $108°$).

The procedure for calculating sunrise and sunset is similar to that discussed previously, except that the Sun does not have a negligible apparent size, hence its angu-

[1]The correction for proper motion is calculated only if information about it is provided in input.

[2]See the discussion about the file `atm_data.csv` in Appendix A.

lar radius must be considered[3] [4]. After that, the duration of twilight is estimated and added (subtracted) to the time of sunset (sunrise) to obtain the temporal range of daylight [1]. If the Sun never sets or always remains above $-18°$ (in altitude) then the target can not be visible and the function for twilight estimation returns the Boolean value `True`.

The distance from the Moon

The result of the previous steps (if any) is a *time window* during the night when the object is above the horizon. However, in this time range, the Moon could be present too.

Since it is the proximity to the Moon that matters, the script computes the angular distance[4] between (the centers of) the satellite and the target, rather than the moonrise and moonset: if the distance is less than the angular semi-diameter of the Moon, the object is not visible. However, even if the target is not behind Moon, but it is close, the observation can be more complicated due to the brightness of the satellite. For this reason, the implemented routine for the plot of visibility also provides for the angular distance from Moon at different times and two values of the illuminated fraction of the Moon's disk $k = (1 + \cos i)/2$ (where i is the phase angle)[4], which are the maximum $k_{\max}$ and the mean $\bar{k}$ in the time window (of visibility).

Airmass

The lower the altitude, the more difficult the observation. The reason is the increase in air mass along the line of sight and consequently the increase in dispersion. [3, 5]. In the approximation of parallel planes, the *airmass* parameter can be defined as:

$$X \equiv \sec(z) = \frac{1}{\sin(alt)}$$

where alt is the altitude, while z is the zenith distance ($90° - alt$) [1, 3–6].

In this implementation, the effect on the target's apparent magnitude was not studied. Still, the value[5] of $X = 3$ ($alt \sim 20°$) is reported in the visibility plot (see Figure 1 and others).

2.2 The python library

The python library `visibility` is organized in 1 package and 4 modules:

- `Time`
  The package gathers the modules to handle times and dates (see Section 2.2.1). It also provides functions to compute the Julian Day, the Local Sidereal Time (LST), and the correction for nutation [4].

- `Angles.py`
  This module implements two classes for angle manipulations (see Section 2.2.1).

- `coor.py`
  This module contains the classes to handle the celestial coordinates (see Section 2.2.1). It also provides the functions[6] to pass from one frame to the other [1, 3–5]. The observatory's position on the Earth is set by giving its latitude, longitude, and altitude above sea level.

- `sky.py`
  This module collects the functions to fulfill the task. It provides methods to calculate the rising and setting time of the target and Sun, and the angular distance from the Moon. This module also implements the routine to compute both the time window and the visibility plot.

- `stuff.py`
  This module provides functions to import and interpolate data [4].

The library's documentation contains a detailed description for each function and module[7]. In this report, only the main steps of the implementation are discussed.

2.2.1 My classes

The necessary calculations for the assignment require handling angles in different units, dates, times, and coordinate systems. To simplify the routines, the library provides some implemented classes, described hereafter in brief:

- `Angles` and `HAngles` from module `Angles.py` are quite the same; these classes store the value of an angle in degrees, radians and, only for the latter, hours[8].

- `Time` from module `Tclasses.py` of package `Time` is used, as the name suggests, for the time: it collects the value of time in seconds and the information about the *"type"*, such as UT or TD. It also provides the conversion from UT to TD and vice versa.

- `Data` from module `Tclasses.py` of package `Time` is one of the most useful. It works with the previous class and permits manipulating dates and times.

[3] According to [1], to compute the size of the solar disk we have to take a reference value, which is the angular diameter θ_0 when the Sun-Earth distance R is equal to the semi-major axis of the orbit R_0 for the epoch $J2000.0$. Then the angular size θ can be expressed simply as $\theta = \theta_0\,(R_0/R)$.

[4] Typically the angular distance d is calculated from its cosine, but in [4] the author suggests to compute $\tan d = \sqrt{1 - z^2}/z$ (with $z \equiv \cos d$) to get a more accurate result.

[5] This value can be changed arbitrarily.

[6] These coordinate transformations can be obtained from appropriate rotations of the frame [3, 5].

[7] See https://github.com/00-berni/proj_0.

[8] $1\,\mathrm{h} = 15°$.

The class stores all information about a date and its corresponding value in Julian days, considering also time zone and daylight saving (only if the input time is local).

- `Equatorial`, `Ecliptical` and `AltAz` from module `coor.py` are the implementations of the corresponding celestial coordinates [1, 3–6]. `GeoPos` is used for the location of the observatory.

- `Target` from module `sky.py` collects the information about the target and provides the methods to compute its coordinates in date and the corresponding hour angle.

- `Sun` and `Moon` from module `sky.py` are used to compute the place of these two celestial objects. The former also calculates sunrise, sunset, and twilight and correction for the annual aberration [4, 6], and the latter gives the two values of the illuminated fraction k.

2.3 Plot of visibility

The `visibility()` function returns a text output that contains:

- the information about the observation (date and place)
- the data about the target and the times of transit, rising and setting
- the sunset and sunrise and the duration of twilight
- the minimum angular distance from Moon and the value k
- the time window of visibility (if any)

In addition to that, it also provides the plot of visibility, which is the aim of this assignment.

2.3.1 Trajectory

To track the trajectory of the target along the sky the plot shows the evolution of its apparent altitude with time (UT). This is obtained from[9] [1, 3, 4]:

$$\sin(alt) = \sin(\varphi)\sin(\delta) + \cos(\varphi)\cos(\delta)\cos(HA)$$
$$HA = GST - \lambda - \alpha$$

where α and δ are the target's equatorial coordinates (corrected for the selected date), λ and φ are respectively the longitude and the latitude of the observatory, HA is the local hour angle and GST is the corresponding sidereal time at Greenwich (computed for the selected date through the algorithm in [4]).

[9]This expression is from the transformation formulas to pass from Equatorial to Horizontal coordinates [1, 3, 4].

Then to obtain the apparent altitude the correction for atmospheric refraction[10] is added [4]:

$$alt_0 = alt + \mathcal{R}'(h)$$

where h is the altitude of the observatory above the sea level.

2.3.2 Plot description

Figure 1 shows an example of a visibility plot. The observation start time is the closest dusk to midnight of the selected date[11]. The trajectory tracks of the target, the Moon, and the Sun are shown in different colors. The long dashed vertical lines represent the times of dusk and dawn, while the dashed horizontal is the set value in *airmass*. Some other information is displayed in both the top and bottom parts of the figure: the former reports the name of the target, the date of observation, and the time window of visibility; while the latter shows on the left the observatory coordinates (latitude, longitude, and height a.s.l.) and on the right information about $\bar{k}$ and minimum distance from Moon[12].

The plots described in Section 3 are slightly different from the previous one: they display the trajectory only when the target is visible, instead of for the entire day. This feature can be triggered by the parameter `win` of the function `visibility_plot()`. Angular distances from the Moon at different times are also reported along the track of the trajectory.

3 Applications

The Python script `script.py` requires 11 inputs[13]:

1. the name of the target
2. α in hours
3. δ in degrees
4. μ^*_α and μ_δ in mas/y
5. the epoch of the coordinates data
6. the name of the observatory
7. φ in degrees (positive northward)
8. λ in degrees (positive westward)
9. h in meters
10. the date of observation

[10]See Appendix A

[11]For example: if dawn is at 02:00:00 UT on 28 July, the displayed time window will start from the end of the dusk on 27 October. See Figure 5 or Figure 7.

[12]This is the minimum distance from Moon in the time range of visibility.

[13]In reality, only 7 of these 11 inputs are necessary: information about names and proper motion can be neglected.

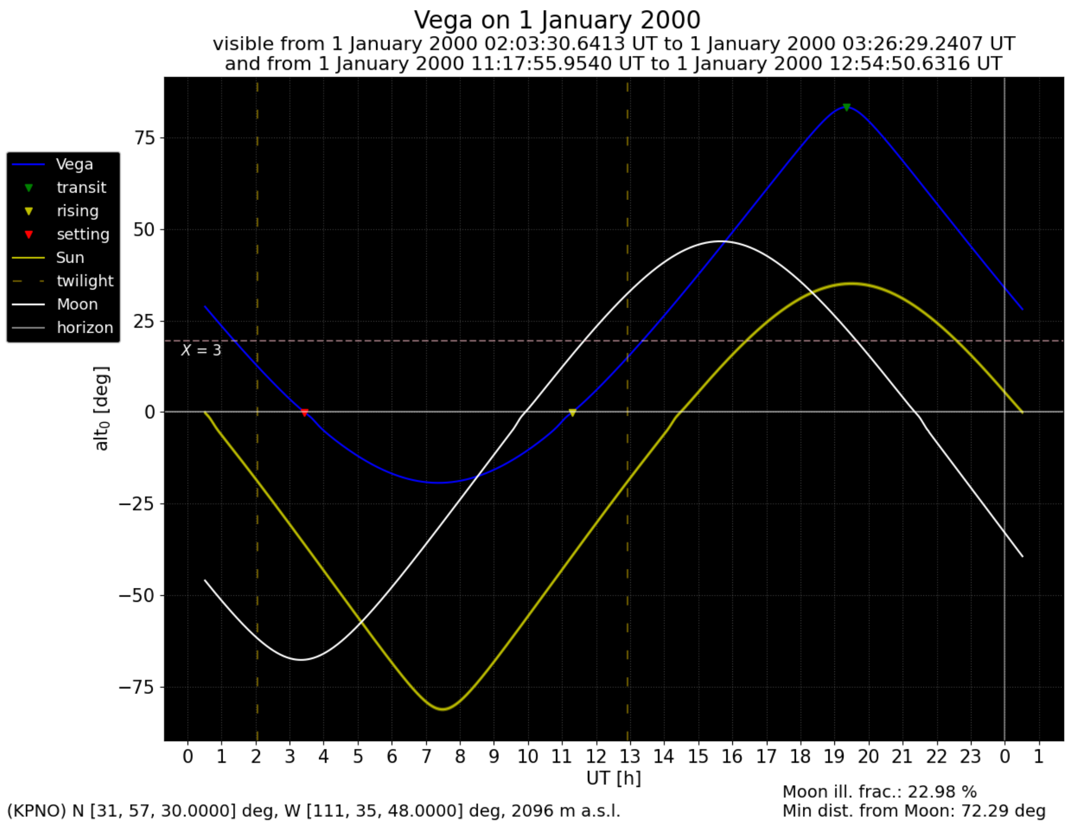

Figure 1: Trajectory of Vega along the sky from the Kitt Peak National Observatory on January 1, 2000.

μ_α^* and μ_δ are the information about the proper motion[14] of the target.

The following sections present several visibility plots to show how the script works. Stellar targets are selected first, then a run is done for celestial objects different from stars. For the sake of completeness, some test cases are reported at the end of this section to demonstrate the program's correct functioning. Data of targets are taken from SIMBAD and reported in Table 1. Each observation has a different date and/or place. The chosen observatories are listed in Table 2. The results are compared with ones from the software *Stellarium*[15]. Regarding k, only $k_{\max}$ is reported for the comparison.

[14]From SIMBAD database http://simbad.cds.unistra.fr/simbad/:

$$\begin{aligned}\mu_\alpha^* &\equiv \mu_\alpha \cos\delta = \\ &= \frac{d\alpha}{dt}\cos\delta \\ \mu_\delta &\equiv \frac{d\delta}{dt}\end{aligned}$$

[15]https://stellarium.org/it/

3.1 Stellar targets

Five plots of visibility are computed for Vega and Spica.

3.1.1 Vega

For Vega three computations are done, corresponding to observations on 01/01/2000, 02/21/2015, and 11/03/1990, shown in Figure 2, Figure 3 and Figure 4, respectively. The results are reported in Table 3.

3.1.2 Spica

For Spica two computations are done, corresponding to observations on 07/28/2024 and 05/20/2023, shown in Figure 5 and Figure 6, respectively. The results are reported in Table 4.

3.2 Other objects

Target does not have to be a star necessary, but any other celestial object[16] is allowed. For this reason, the visibility plots of M31 on 10/26/2023, the Large Magellanic Cloud on 02/02/2024, and Nova Cygni 1975 on

[16]This is not properly true: only *far* objects can be passed to the function. Planets, asteroids, and comets require additional corrections, which are not considered in this report.

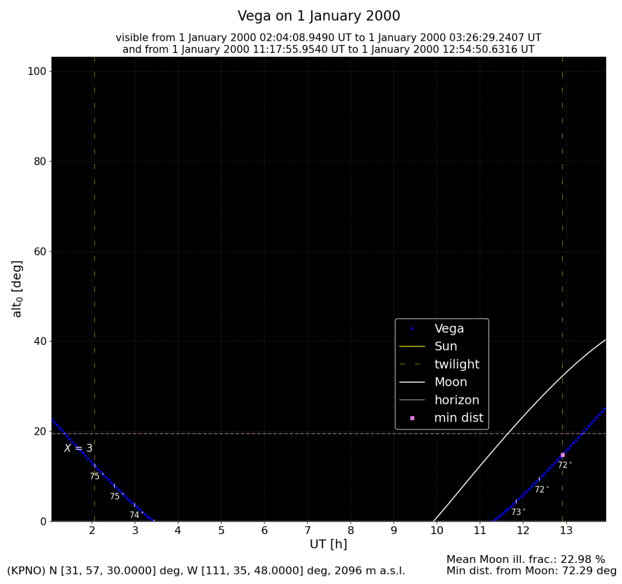

Figure 2: The values along the trajectory are the angular distances from the Moon. The pink cross is the time of minimum distance.

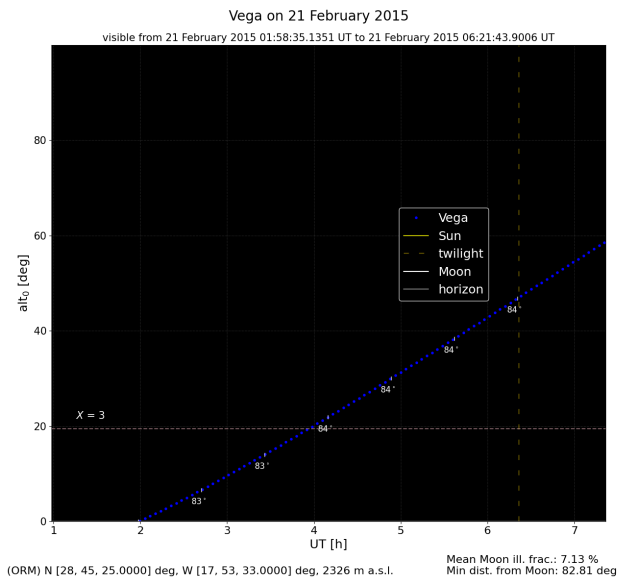

Figure 3: The values along the trajectory are the angular distances from the Moon.

07/27/2023 are reported in Figure 7, Figure 8 and Figure 9 respectively. The results are listed in Table 5, Table 6, and Table 7.

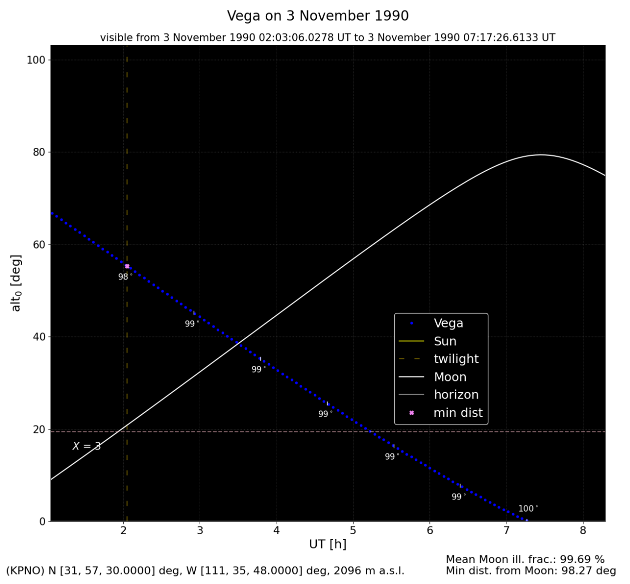

Figure 4: The values along the trajectory are the angular distances from the Moon. The pink cross is the time of minimum distance.

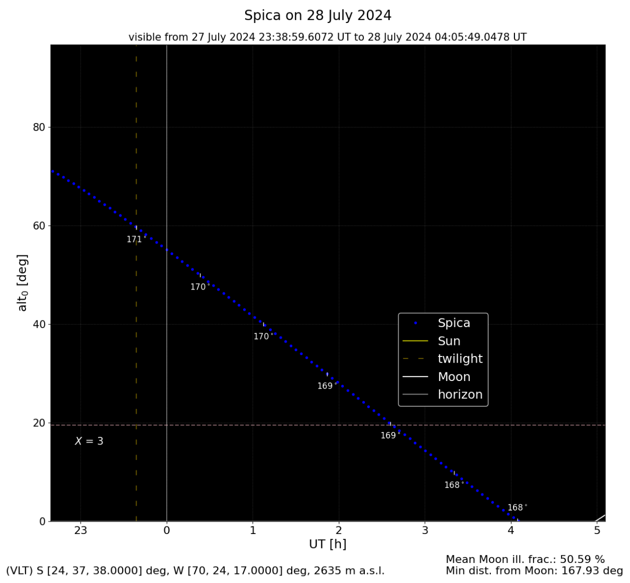

Figure 5: The values along the trajectory are the angular distances from the Moon.

3.3 Test cases

From the previous results, the correct functioning of the script is confirmed. However, there are still some particular cases to investigate.

First is the possibility that the object is not visible in the observation time window. This is shown in Figure 10. The program warns the user about the impossibility of observing the target (because it is below the horizon).

The second case is a circumpolar object, for which there are two possibilities: it is visible for the entire

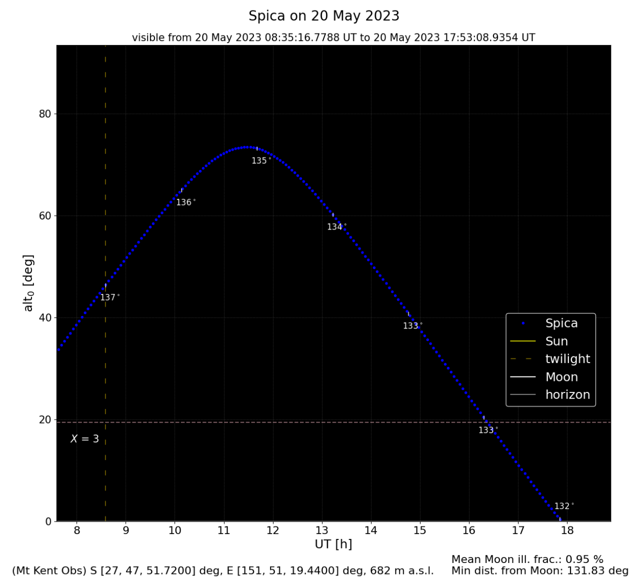

Figure 6: The values along the trajectory are the angular distances from the Moon.

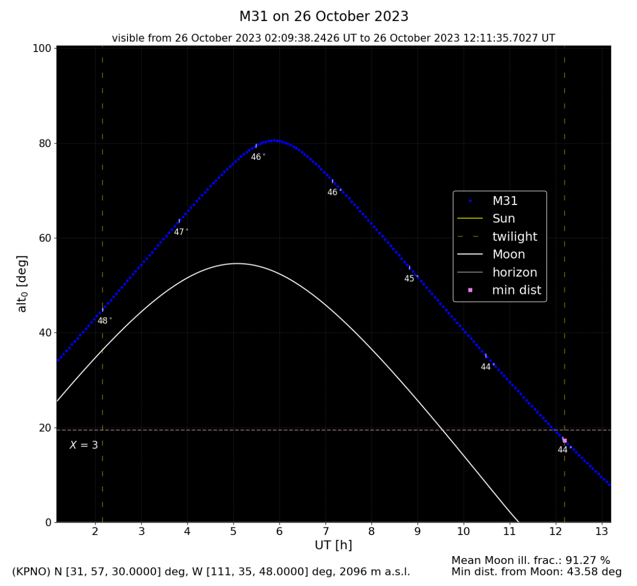

Figure 7: The values along the trajectory are the angular distances from the Moon. The pink cross is the time of minimum distance.

night (it never sets) or if the observatory is in the other hemisphere, it never rises. Both cases are shown in Figure 11.

The last particular situation provides for observation from an observatory near a pole. We choose the South Pole Observatory (see Table 2). The observation during the winter period is possible for the entire day, while during the summer Sun never sets. Both cases are shown in Figure 12.

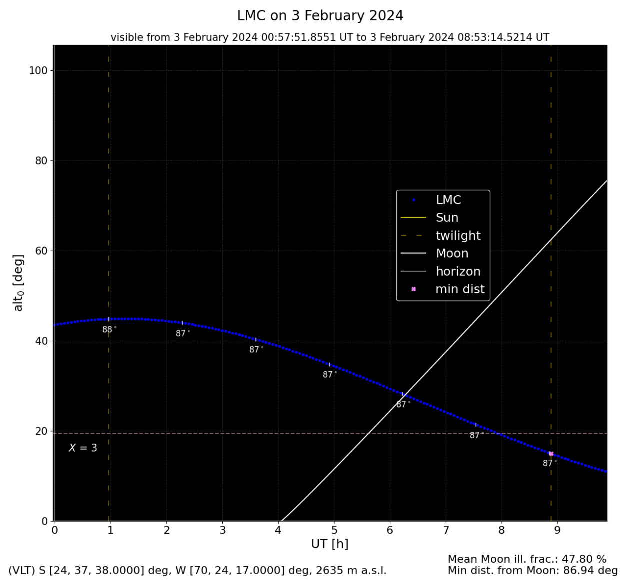

Figure 8: The values along the trajectory are the angular distances from the Moon. The pink cross is the time of minimum distance.

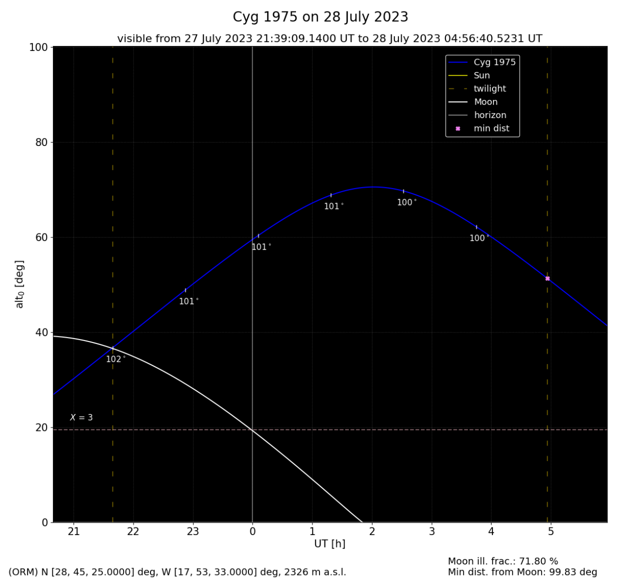

Figure 9: The values along the trajectory are the angular distances from the Moon. The pink cross is the time of minimum distance.

4 Results

From the comparison with the results of *Stellarium* (see Tables from 3 to 7) it appears that positions of the target and the Sun are in *"sufficient"*[17] accordance with the expected ones: the difference between the times of rising or setting (or transit) is about 1 minute at most.

[17]We did not perform a rigorous uncertainty treatment due to the complexity involved.

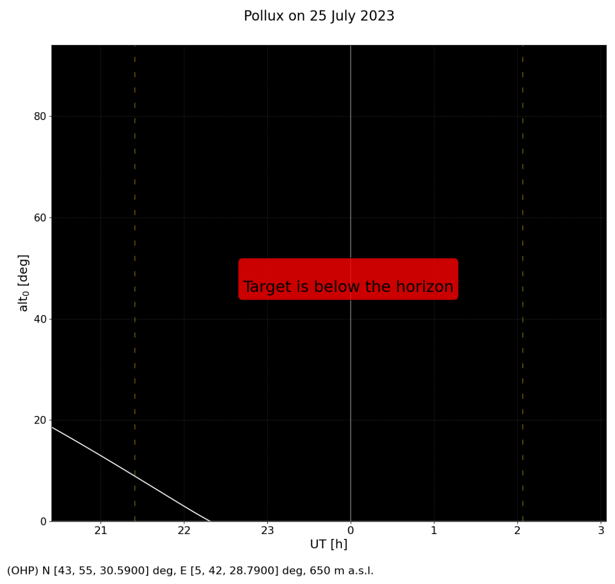

Figure 10: Visibility plot for a date on which the target does not rise during the night, so it is not visible.

It is not the same for the Moon: the discrepancy[18] in k is from 0.04% to 2.8% and reaches 54% for the observation on 20 May 2023 (Mt. Kent Obs.). However, this does not surprise us because the value of k can even increase (decrease) more than double in one day. The high variability together with the approximations used to compute k results in a reduction in accuracy, the effect of which is evident near the New Moon time ($k \sim 0$).

5 Conclusion

As discussed in the previous sections, the library `visibility` enables tracking and plotting a celestial object's trajectory along the sky by computing its visibility. The comparison with the results obtained by the software *Stellarium* shows a sufficient accordance (times differs by no more than 1 minute), except for the illuminated fraction of the Moon's disk k, because for this quantity the accuracy is sensitive to the approximation used to compute it: the lower the value of k, the lower the accuracy.

However, the considerations discussed in Section 4 and summarized here say nothing about the precision of the program and are only qualitative. The reason is that we did not treat the propagation of uncertainty in any approximated quantity because of the complexity involved. Hence, the library `visibility` is a simple and fast way to approximately know a target's visibility during the night, though it can not be used to compute accurate ephemerides.

[18]The discrepancy is computed from:

$$\delta k \equiv \frac{|k_{\max} - k_0|}{k_{\max}}$$

where k_0 is the expected k at the same time of $k_{\max}$.

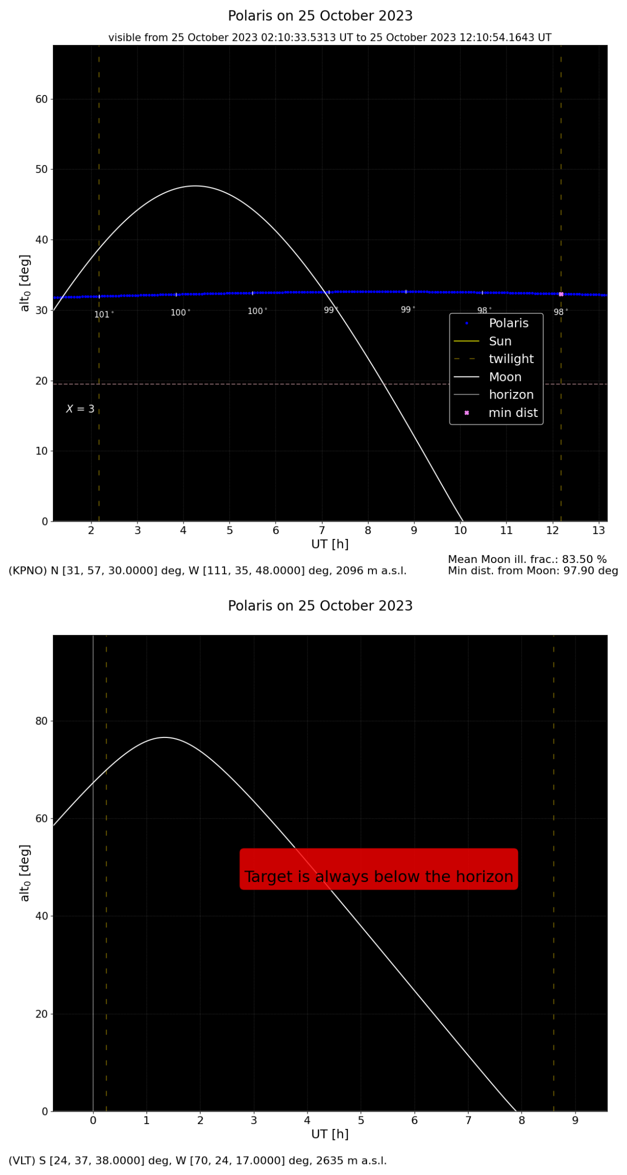

Figure 11: visibility plots of a circumpolar star (Polaris) from two different observatories, one in the northern hemisphere (top) and the other in the southern one (bottom).

Some additional implementations could be: (*i*) the study of the change of the target's apparent magnitude during the night, (*ii*) the estimation of eclipses, (*iii*) the computation of trajectory tracks of planets.

A Data

To calculate certain quantities, the program must interpolate from data tables, which are collected in the directory https://github.com/00-berni/proj_0/data.

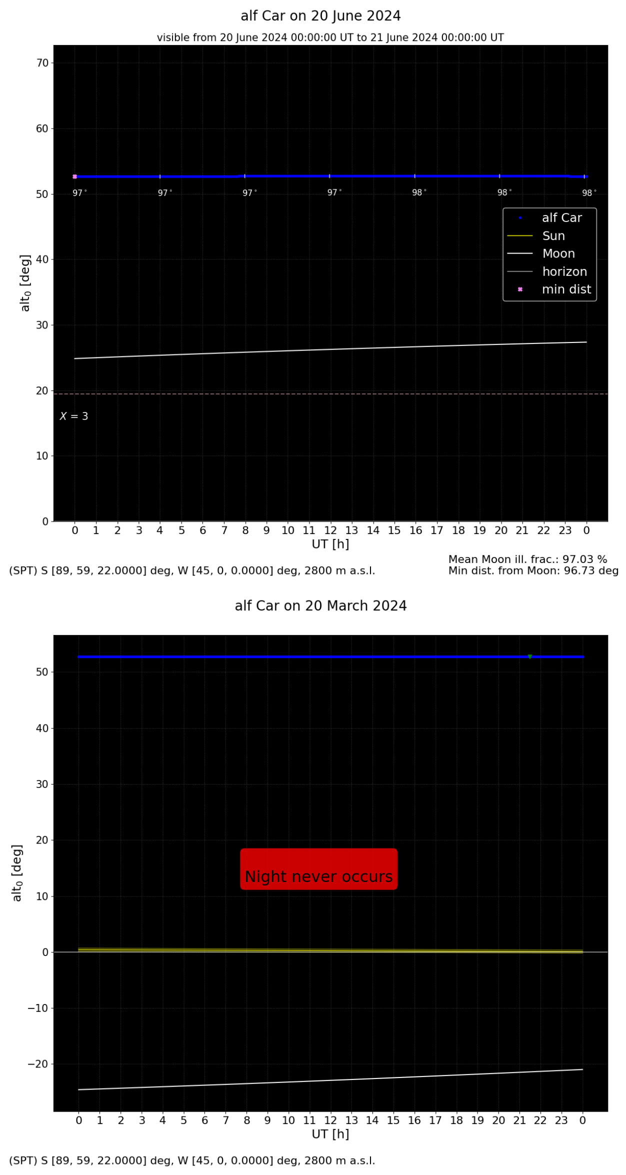

Figure 12: visibility plots of α Carinæ from the South Pole Observatory during winter (top) and summer (bottom).

- `atm_data.csv`
  This file collects data of the *ICAO Standard Atmosphere* from [2], particularly the atmospheric pressure and temperature trend by varying the height above the sea level. This model is not realistic, but used as a reference to compute the correction due to the atmospheric refraction, depending on the altitude h of the observatory [4]:

$$\mathcal{R}'(h) = \mathcal{R} \cdot \frac{P(h)}{P_0} \frac{T_0}{T(h)}$$

  where $\mathcal{R}$ is the refraction correction for an observatory at the sea level, P_0 and T_0 are respectively the pressure (in mbar) and temperature (in K) at $h = 0$. $P(h)$ and $T(h)$ are interpolated from the data table by the function `interpolate_three()` of the module `stuff.py`.

- `dT_data.csv`
  This table gathers the differences between the TD and the UT from 1620 to 1998, reported in [4]. The $TD - UT$ requires observations, so the script interpolates from the table. For the years that are not included in the range, we use an approximated formula [4].

- `moon1_data.csv` and `moon2_data.csv`
  According to the model in [4], the (geocentric) ecliptic latitude and longitude of the Moon are calculated through several harmonic terms, tabulated in these two files. They are used also to compute the Earth-Moon distance.

References

[1] P. Duffett-Smith and J. Zwart. *Practical Astronomy with your Calculator or Spreadsheet.* Cambridge University Press, 2011.

[2] *Manual of the ICAO Standard Atmosphere.* ICAO, 3rd edition, 1993.

[3] P. Léna, D. Rouan, F. Lebrun, F. Mignard, D. Pelat, L. Mugnier, and S.N. Lyle. *Observational Astrophysics.* Astronomy and Astrophysics Library. Springer Berlin, Heidelberg, third edition, 2012. ISBN 978-3-662-51733-8. doi: https://doi.org/10.1007/978-3-642-21815-6.

[4] J. Meeus. *Astronomical Algorithms.* Willmann-Bell Inc., 1998.

[5] Notes of *Astrofisica Osservativa* course, 2021-2022.

[6] P. K. Seidelmann. *Explanatory supplement to the Astronomical almanac.* University Science Books, 2006.

Targets					
Name	α	δ	μ_α^* [mas/y]	μ_δ [mas/y]	epoch
α Carinæ	$6^h\,23^m\,57.110^s$	$-52°\,41'\,44.381''$	261.212	-25.292	J2000
LMC	$5^h\,23^m\,34.6^s$	$-69°\,45'\,22''$	1.910	0.229	J2000
M31	$0^h\,42^m\,44.330^s$	$+41°\,16'\,07.50''$	/	/	J2000
V1975 Cyg	$21^h\,11^m\,36.581^s$	$+48°\,09'\,01.952''$	-6.449	-5.572	J2000
Polaris	$2^h\,31^m\,49.094^s$	$+89°\,15'\,50.792''$	44.48	-11.85	J2000
Pollux	$7^h\,45^m\,16.42^s$	$+28°\,01'\,34.5''$	-626.55	-45.80	J2000
Spica	$13^h\,25^m\,11.579^s$	$-11°\,09'\,40.750''$	-42.35	-30.67	J2000
Vega	$18^h\,36^m\,56.336^s$	$+38°\,47'\,01.280''$	200.94	286.23	J2000

Table 1: Data are taken from SIMBAD.

Observatories			
Name	lat	lon	h a.s.l. [m]
Kitt Peak National Observatory	$31°\,57'\,30''$ N	$111°\,35'\,48''$ W	2096
Mt Kent Observatory	$27°\,47'\,51.72''$ S	$151°\,51'\,19.44''$ E	682
Observatoire de Haute-Provence	$43°\,55'\,30.59''$ N	$5°\,42'\,28.79''$ E	650
Roque de los Muchachos Observatory	$28°\,45'\,25''$ N	$17°\,53'\,33''$ W	2326
South Pole Observatory	$89°\,59'\,22''$ S	$45°\,00'\,00''$ W	2800
Very Large Telescope array	$24°\,37'\,38''$ S	$70°\,24'\,17''$ W	2635

Table 2: Data are taken from the web.

Vega						
	01/01/2000		21/02/2015		03/11/1990	
	Script	*Stellarium*	*Script*	*Stellarium*	*Script*	*Stellarium*
rising	11 : 17 : 56	11 : 17 : 00	01 : 58 : 35	01 : 58 : 00	15 : 08 : 57	15 : 08 : 00
transit	19 : 20 : 15	19 : 20 : 00	09 : 45 : 01	09 : 45 : 00	23 : 11 : 14	23 : 11 : 00
setting	03 : 26 : 29	03 : 27 : 00	17 : 31 : 26	17 : 32 : 00	07 : 17 : 27	07 : 18 : 00
sunrise	14 : 27 : 17	14 : 27 : 00	07 : 45 : 29	07 : 45 : 00	13 : 45 : 04	13 : 45 : 00
sunset	00 : 31 : 01	00 : 32 : 00	19 : 04 : 34$^{(-1)}$	19 : 05 : 00$^{(-1)}$	00 : 36 : 11	00 : 36 : 00
$k_{\max}$	11 : 17 : 56		06 : 21 : 43		02 : 03 : 50	
	23.25 %	22.6 %	7.71 %	7.9 %	99.77 %	99.8 %

Table 3: Results for observations of Vega from KPNO (first and third columns) and ORM. Times are UT. The time, at which $k_{\max}$ is computed, is reported over the value. The superscript $^{(-1)}$ indicates that time refers to the previous day.

Spica				
	28/07/2024		20/05/2023	
	Script	*Stellarium*	*Script*	*Stellarium*
rising	15 : 17 : 59	15 : 17 : 00	05 : 01 : 58$^{(-1)}$	05 : 02 : 00$^{(-1)}$
transit	21 : 39 : 56	21 : 40 : 00	11 : 31 : 29	11 : 31 : 00
setting	04 : 05 : 49	04 : 06 : 00	17 : 57 : 05$^{(-1)}$	17 : 57 : 00$^{(-1)}$
sunrise	11 : 20 : 44	11 : 20 : 00	20 : 29 : 34$^{(-1)}$	20 : 29 : 00$^{(-1)}$
sunset	22 : 15 : 16$^{(-1)}$	22 : 16 : 00$^{(-1)}$	07 : 09 : 20$^{(-1)}$	07 : 09 : 00$^{(-1)}$
$k_{\max}$	23 : 39 : 00$^{(-1)}$		08 : 35 : 37$^{(-1)}$	
	51.64 %	51.3 %	0.13 %	0.2 %

Table 4: Results for observations of Spica from VLT (left) and Mt. Kent Obs. (right). Times are UT. The time, at which $k_{\max}$ is computed, is reported over the value. The superscript $^{(-1)}$ indicates that time refers to the previous day.

M31		
	Script	*Stellarium*
rising	21 : 33 : 26	21 : 33 : 00
transit	05 : 53 : 01	05 : 53 : 00
setting	14 : 08 : 41	14 : 09 : 00
sunrise	13 : 38 : 29	13 : 38 : 00
sunset	00 : 43 : 46	00 : 43 : 00
$k_{\max}$	12 : 11 : 36	
	92.64 %	92.6 %

Table 5: Results for observations of M31 from KPNO on 26 October 2023. The time, at which $k_{\max}$ is computed, is reported over the value. Times are UT.

LMC		
	Script	*Stellarium*
transit	01 : 14 : 08	01 : 14 : 00
sunrise	10 : 20 : 35	10 : 20 : 00
sunset	23 : 30 : 18$^{(-1)}$	23 : 31 : 00$^{(-1)}$
$k_{\max}$	00 : 57 : 52	
	49.45 %	49.2 %

Table 6: Results for observations of LMC from VLT on 3 February 2024. The time, at which $k_{\max}$ is computed, is reported over the value. Times are UT. The superscript $^{(-1)}$ indicates that time refers to the previous day.

V1975 Cyg		
	Script	*Stellarium*
transit	02 : 02 : 03	02 : 02 : 00
rising	17 : 23 : 48	17 : 23 : 00
setting	10 : 36 : 23	10 : 37 : 00
sunrise	06 : 30 : 54	06 : 30 : 00
sunset	20 : 05 : 31$^{(-1)}$	20 : 06 : 00$^{(-1)}$
$k_{\max}$	04 : 56 : 40	
	73.29 %	72.8 %

Table 7: Results for observations of V1975 Cyg from ORM on 28 July 2023. The time, at which $k_{\max}$ is computed, is reported over the value. Times are UT. The superscript $^{(-1)}$ indicates that time refers to the previous day.