INGOT Wavefront Sensor Simulations

Final Internship Report

Savina Tsichli
Supervisors: Elisa Portaluri, Demetrio Magrin, Mosè Mariotti
INAF – Osservatorio Astronomico di Padova

Feb - May 2025

1 The Ingot Project

The Ingot Wavefront Sensor (I-WFS) is a novel pupil-plane sensor developed to address the challenges posed by Laser Guide Stars (LGSs) in Adaptive Optics (AO) systems, especially for Extremely Large Telescopes (ELTs), as seen in [1], [2] LGSs are artificial sources introduced to replace Natural Guide Stars (NGSs) in star-poor regions, enabling AO correction across a wider portion of the sky. Unlike natural stars, however, LGSs are not true point sources—their emission originates from an elongated, cigar-shaped volume in the sodium layer at around 90 km altitude, illustrated in Fig. 1 by [3]. This 3-face configuration enables better sampling of the 3D LGS structure, provides the best compromise between the amount of light in the faces –improving signal stability– while making it particularly suitable for ELT-scale instruments. This extended structure leads to perspective-dependent variations across the telescope pupil, introducing optical effects such as spot elongation, truncation, and defocusing. These distortions can degrade the performance of conventional wavefront sensors, like the Shack-Hartmann, particularly for ELTs, where the size of the LGS becomes more significant, mentioned in [4].

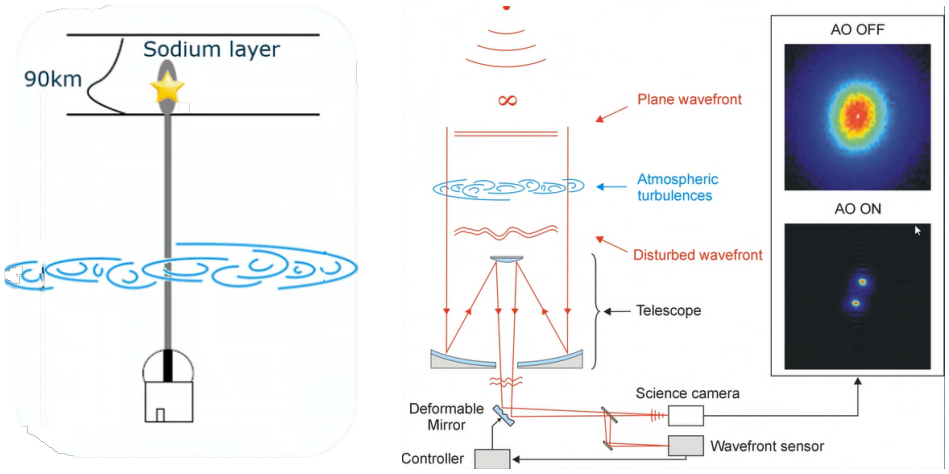


Figure 1: Adaptive Optics on LGSs (left) and NGSs (right).

The I-WFS was initially designed to exploit the elongated structure of the LGS by using a refractive geometry that better matches the light distribution observed from different sub-apertures. Originally proposed with six reflective surfaces, the design evolved into a more practical and compact configuration consisting of a three-face prism—two reflective and one transmitting surface—as described in [4] and illustrated in Fig. 2.

This geometry-aware approach aims to improve wavefront sensing performance under conditions where LGSs introduce non-negligible spatial distortions, offering a promising alternative to traditional SH-WFS systems, particularly for AO systems on next-generation telescopes like the ELT.

The I-WFS is being developed within the MORFEO project and tailored for the ELT, a 39-meter-class telescope that relies on multiple LGSs to perform high-resolution AO corrections. The ELT's

large aperture and use of multiple laser launchers heighten the relevance of geometrical effects, making accurate LGS modeling and tailored wavefront sensing essential. Ingot's unique design directly addresses this by factoring in the source geometry, atmospheric cone effect, and the different perspectives from each sub-aperture.

So far, simulations of the I-WFS show promising results in reconstructing atmospheric turbulence using modal (Zernike-based) approaches. The simulator also incorporates parameters such as sodium layer profiles, atmospheric turbulence models, and LGS sampling strategies to mimic realistic observing scenarios [2], [5].

To evaluate the performance of this design, a test setup is implemented at the optical bench in Padova, which mimics the ELT configuration. This allowed experimental validation of the simulated prism under realistic conditions and verification of its behavior under varying atmospheric parameters, as seen in [6], [7], [8].

Overall, the design's compatibility with the ELT and its capacity for high-performance wavefront reconstruction under complex conditions, make I-WFS a compelling solution for next-generation AO systems.

Figure 2: 3-faces Ingot prism.

2 Simulator Analysis Approach

The simulator foresees the following main components, also indicated in Fig. 3:

1. Simulation of LGS
2. Ingot prism design
3. Evaluation under atmospheric turbulence conditions
4. Reconstruction and Performance Analysis

As shown in Fig. 3, the process begins with an input dataset and progresses through stages that simulate the atmosphere, define the LGS geometry and sampling, model the propagation of the wavefront through the atmosphere, and finally reconstruct the wavefront using modal fitting techniques. Also, different flags are shown, that can be turned on depending on the aim of the simulation.

A detailed description of the parameters used in this figure can be found in the following sections.

2.1 Programming Tools

Over the past 20 years, astronomers have used various programming languages to simulate and analyze observational data; starting with SuperMongo, Fortran, then MATLAB and IDL. The code I have analyzed, testing the performance of the I-WFS, was written in IDL, first mentioned in [9]. However, due to IDL's limited visualization flexibility and performance constraints, much of the plotting and analysis has been translated to Python. As an open-source language, Python ensures better compatibility with modern tools, as well as allowing collaboration through shared libraries and community contributions.

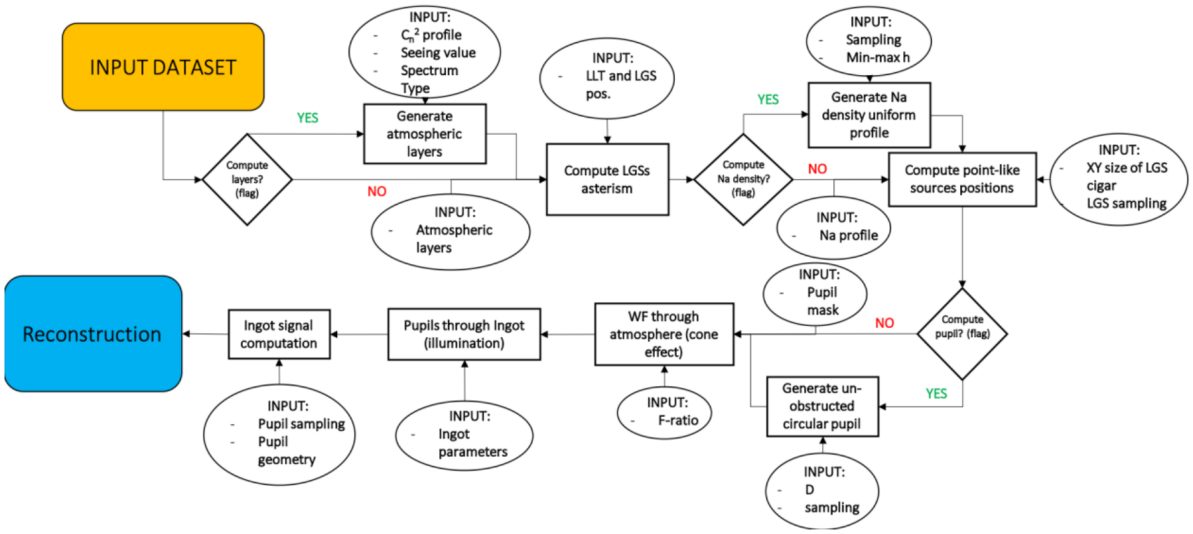


Figure 3: Logical flow of the simulation tool, including inputs and flags.

More specifically, at the beginning of the project, a significant amount of time was spent resolving installation issues and configuring the IDL license on a personal machine. Challenges such as license incompatibility with the operating system and the need to bypass server-based license access, slowed down early development. In addition, network-related access restrictions, including VPN requirements for the INAF environment, added further complexity.

Then, translating as much of the analysis as possible into Python, taking advantage of its scientific libraries and reproducibility was easier. This not only made visualization and debugging easier, but also made the code more accessible for future development.

Summarizing, this points to a wider issue; proprietary tools like IDL can limit accessibility and collaborative development. Open-source alternatives such as Python are easier to install, distribute, and adapt, making them more appropriate for scientific work that is meant to be shared and built upon.

2.2 Analysis Description

As mentioned earlier, the code is written in IDL version 8.7 and it is composed by a number of functions computing different steps required by the overall simulation. In order to asses the performance of the I-WFS, simulations were run varying parameters in different environments; specifically, the behavior of the LGSs interacting with the prism's geometry in simulated AO configurations.

Table 1 summarizes the main input parameters required by the simulation tool and the corresponding outputs it generates. These cover physical configurations (e.g., sodium layer and turbulence), instrumental setups (e.g., telescope pupil and Ingot geometry), and computational options (e.g., sampling style, reconstruction basis).

Input Parameter	Description
`D`	Diameter of the telescope.
`source_scale`	Scaling factor for the LGS cigar size.
`sampling_FP`	Number of sampling points in the focal plane.
`rot`	Rotation flag to apply geometric rotation to the LGS.
`grid_style`	Sampling strategy: `5points`, `regular`, `random`, etc.
`xy_sampling`	Resolution of sampling grid in X and Y directions.
`h_Na_slices`	Sodium layer altitudes sampled (e.g., 90000–90900 m).
`LGS_sampling_width`	width of sampled beam
`ingot_angle`, `base_width`	Geometry parameters defining the prism faces.
`LLT_position`	Location of the laser launcher on the telescope pupil.
`pupil_mask`	ELT aperture mask.
`h_Na`	Mean altitude of the sodium layer.
`h_Na_all`	Vertical extent of sodium layer.
`Na_profile`	Sodium density profile.
`Na_sampling`	Sampling rate along vertical axis.
`lambda_wfs`	Wavelength used for wavefront sensing (λ_{WFS}).
`pupil_sampling`	Number of actuators across pupil diameter.
Output	**Description**
`lgs_coord_slice.sav`	3D coordinates of sampled LGS points.
3D LGS plots (Python)	Visualizations of various sampling styles.
`punti`, `poli`	Arrays defining vertices and faces of the Ingot prism.
3D prism plot (Python)	Visual plot showing reflecting and transmitting surfaces.
Execution time (manual)	Performance timing of simulation runs for different grids.

Table 1: Subset of parameters and outputs used in this work.

In Fig. 4, the geometry of the LLT and LGS is shown for the ELT case. The sodium layer, at `h_Na` height from the `LLT_position`, is excited, creating the LGS which has a cylindrical form when observed in the sky, colloquially referred to as "cigar-shaped". The slices simulated are shown at top right as a magnified "point" of the LGS, composed of `h_Na_slice`. The width of the beam is `LGS_sampling_width`.

3 My contribution

3.1 Simulation of LGS and Sampling Strategy

3D plots of LGSs were produced to get familiar with the structure and geometry of the simulation code. The LGS must be accurately modeled to account for its finite 3D structure, caused by the excitation of the sodium layer at ∼90 km altitude. This structure is non-negligible for ELTs, especially when the LLT is launched off-axis, which introduces elongation effects across the pupil.

For the moment we are considering only 1 LGS with a given sodium profile and with a certain spatial sampling. For the simulation of the LGS; we can choose a random sample to fill the source, represent it like a cylinder uniformly sampled or with a number of disks with a certain intensity profile.

For this, the code provides several `grid_style` options for distributing sampling points:

1. `5Points`
2. `Random / Random Even / Random Even Gauss`

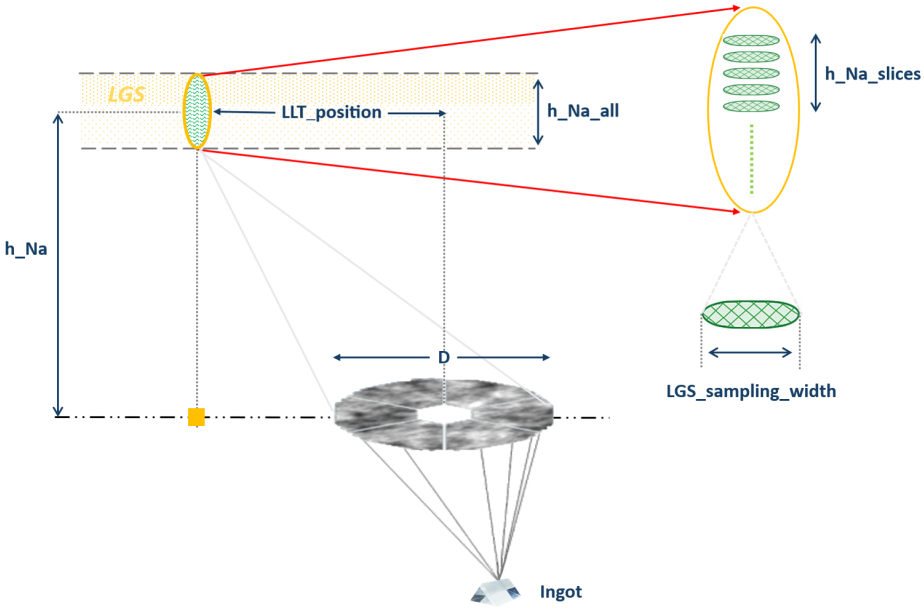


Figure 4: Geometry of the ELT.

3. `Regular / Regular Even / Regular Even Gauss`

4. `Modul`

The `5Points` mode uses a minimal representation of the LGS as an ellipse with five sample points. This approach is very lightweight computationally but does not adequately represent the spatial extent or density variations in the sodium layer. On the other hand, `Regular` and `Random` grids distribute points more realistically in 3D space, with Gaussian options adding a weighting that mimics atomic excitation profiles. While these modes offer improved spatial accuracy, they also demand greater computational resources in terms of memory usage and processing time.

3.2 Producing the plots

The steps followed were running the idl files (`disk.pro`, `dataset.dat`), exporting the coordinates of the LGSs in a .sav file, running a python script to 3D plot them.

1) the parameters .dat file is edited depending on the plot we want to export. Vary the parameters `source_scale`, `sampling_FP` and select a `grid_style`. Then, in disk.pro set the rotation `rot` to 0 or rot. The `h_Na_slices` values used for all the styles except for 5points were [`90000, 90100, 90200, 90300, 90400, 90500, 90600, 90700, 90800, 90900`]. In order to run only up to the part of disk.pro that calculates the coordinates, we type `stop` after the line that prints them.

2) in idl:

```
.r disk_sch_1000_V.pro
disk_sch_1000_V, 'input_test/disco_dataset_mac.dat'
save, lgs_coord_slice, filename="random.sav"
.reset
```

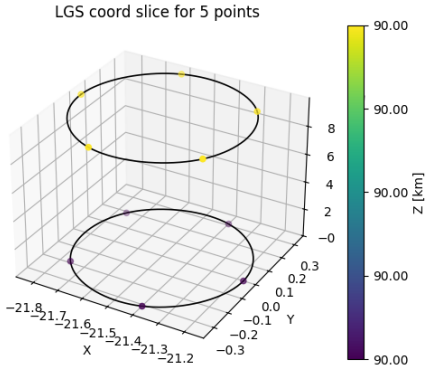
3) After the .sav file is saved, it is imported in a python script 3.2 that reads it, extracts the coordinates and plots them. The axes limits were set to [90000 - 90900] on all plots for comparison.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import readsav
from matplotlib.ticker import FuncFormatter
```

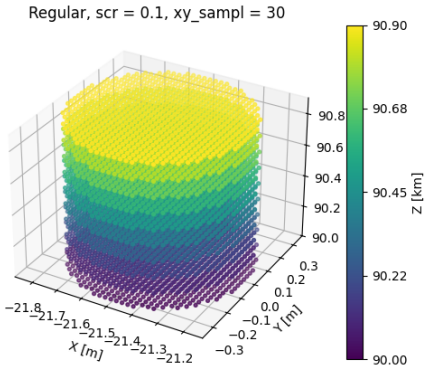
```
5
6  # read .sav file from idl
7  sav_data = readsav("pipa_ranevga_01_sampfp20_30.sav")
8
9  # extract coordinates from columns
10 coords = sav_data['lgs_coord_slice']
11 x, y, z = coords[:, 0], coords[:, 1], coords[:, 2] / 1000
12 print(x)
13
14 fig = plt.figure()
15 ax = fig.add_subplot(111, projection='3d')
16
17 sc = ax.scatter(x, y, z, c=z, cmap='viridis', s=8, vmin=np.min(z), vmax=np.max(z),
       linestyle='None')
18
19 ax.set_xlabel("X [m]")
20 ax.set_ylabel("Y [m]")
21 ax.set_zlabel("Z [km]")
22 ax.set_zlim(np.min(z), np.max(z))
23 plt.title("Random, rot = 1, scr = 0.1, sampling FP = 20, xy_sampl = 30")
24
25 cb = fig.colorbar(sc, ax=ax, label='Z [km]')
26 def fmt(x, pos): return f"{x:.2f}"
27 cb.formatter = FuncFormatter(fmt)
28 cb.set_ticks(np.linspace(np.min(z), np.max(z), 5))
29 cb.update_ticks()
30 plt.show()
```

Listing 1: Python snippet.

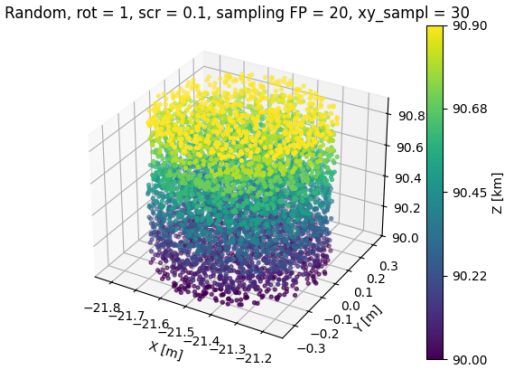
Some plots from all the grid styles are shown in Fig. 5.



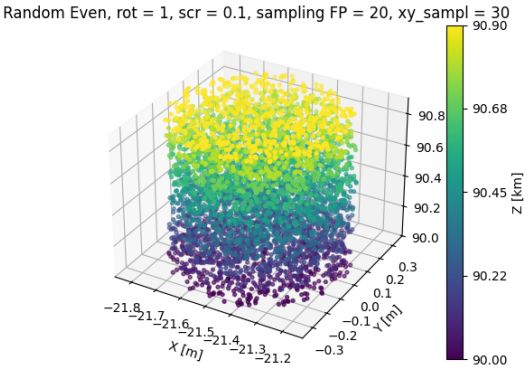
(a) 5 Points



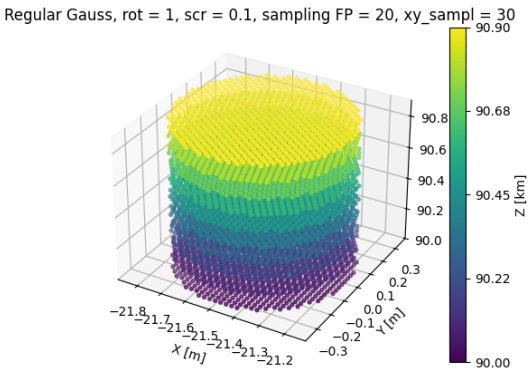
(b) Regular



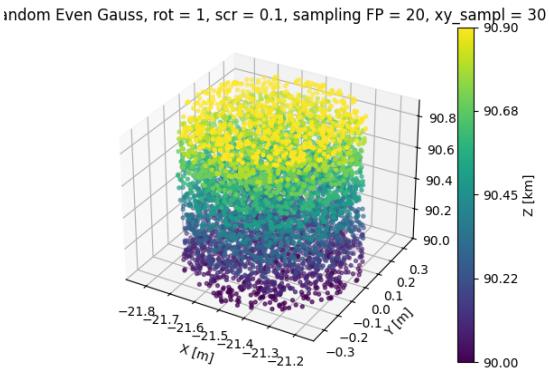
(c) Random



(d) Random Even



(e) Regular Gauss



(f) Random Even Gauss

Figure 5: Comparison of different grid styles for LGS sampling.

3.3 Ingot prism Design

After visualizing the elongated cigar-shaped LGS, the design of Ingot is plotted. First step is inspecting the `ingot_design_3.pro` script. This function calculates the design of Ingot based on the input parameters (angles, width, center). It pinpoints the intersecting points of the planes from the values stored in the `poli` variable and then connects them with lines obtained just using geometry.

As mentioned in the beginning, in this script, 3 faces are visualized, computed from vertices (`punti`) and faces (`poli`).

So, after setting `plot = 1` in the `disk.pro`, we run the script. The exported plot is Fig. 6. In order to make this design in 3D planes, we run a script in python. First, we export the values of the poli variable, as seen in Fig. 7.

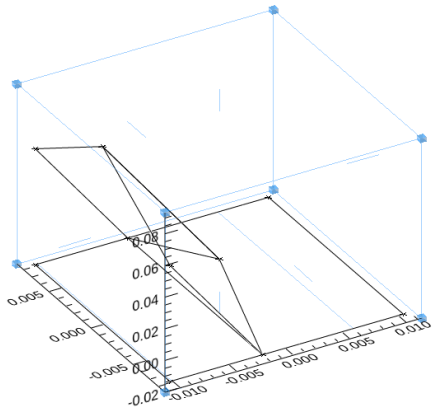


Figure 6: Ingot Design

```
Plane       1
  Vertex        1:    -0.0021400625,     0.0080000000,     -0.019161017
  Vertex        2:     -0.010325000,     0.0080000000,      0.054122726
  Vertex        3:     -0.010325000,        0.0000000,      0.092444767
  Vertex        4:        0.0000000,        0.0000000,        0.0000000
Plane       2
  Vertex        1:        0.0000000,        0.0000000,        0.0000000
  Vertex        2:     -0.010325000,        0.0000000,      0.092444767
  Vertex        3:     -0.010325000,    -0.0080000000,      0.054122726
  Vertex        4:    -0.0021400625,    -0.0080000000,     -0.019161017
Plane       3
  Vertex        1:      0.010325000,     0.0080000000,     -0.019161017
  Vertex        2:     -0.010325000,     0.0080000000,     -0.019161017
  Vertex        3:     -0.010325000,    -0.0080000000,     -0.019161017
  Vertex        4:      0.010325000,    -0.0080000000,     -0.019161017
```

Figure 7: Poli values from IDL

The following script in python 3.3 defines five polygonal faces based on these vertices: two inclined grey reflecting surfaces and a cyan vertical transmitting face, after reading the key vertices values from the IDL export.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
from matplotlib.patches import Patch

# Key vertices (from IDL export)
V0 = np.array([-0.0021400625,  0.008, -0.019161017])
V1 = np.array([-0.010325,      0.008,  0.054122726])
V2 = np.array([-0.010325,      0.0,    0.092444767])
V3 = np.array([ 0.0,           0.0,    0.0])
V4 = np.array([-0.010325,     -0.008,  0.054122726])
V5 = np.array([-0.0021400625, -0.008, -0.019161017])

reflecting_face_1 = [V0, V1, V2, V3]
reflecting_face_2 = [V3, V2, V4, V5]
transmitting_face = [V1, V2, V4, V5]
bottom_face       = [V3, V5, V4]
front_face        = [V0, V3, V5, V0]

faces  = [reflecting_face_1, reflecting_face_2, transmitting_face, bottom_face,
    front_face]
colors = ['grey', 'grey', 'cyan', 'lightgrey', 'lightgrey']

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

for face, color in zip(faces, colors):
    poly = Poly3DCollection([face], facecolors=color, edgecolors='black', linewidths
    =1, alpha=0.95)
    ax.add_collection3d(poly)

```

```
ax.set_xlabel("X [m]")
ax.set_ylabel("Y [m]")
ax.set_zlabel("Z [m]")
ax.set_xlim([-0.02, 0.02])
ax.set_ylim([-0.02, 0.02])
ax.set_zlim([-0.02, 0.1])
ax.view_init(elev=20, azim=45)
ax.set_box_aspect([1, 1, 1.5])

legend_elements = [
    Patch(facecolor='grey', edgecolor='black', label='Reflecting Mirrors'),
    Patch(facecolor='cyan', edgecolor='black', label='Transmitting Face (Back)'),
]
ax.legend(handles=legend_elements, loc='upper left', fontsize=9)

plt.tight_layout()
plt.show()
```

Listing 2: Python snippet.

The 3D plot produced shows the transmiting surface in cyan and the two reflecting surfaces in gray, as shown in Fig. 8.

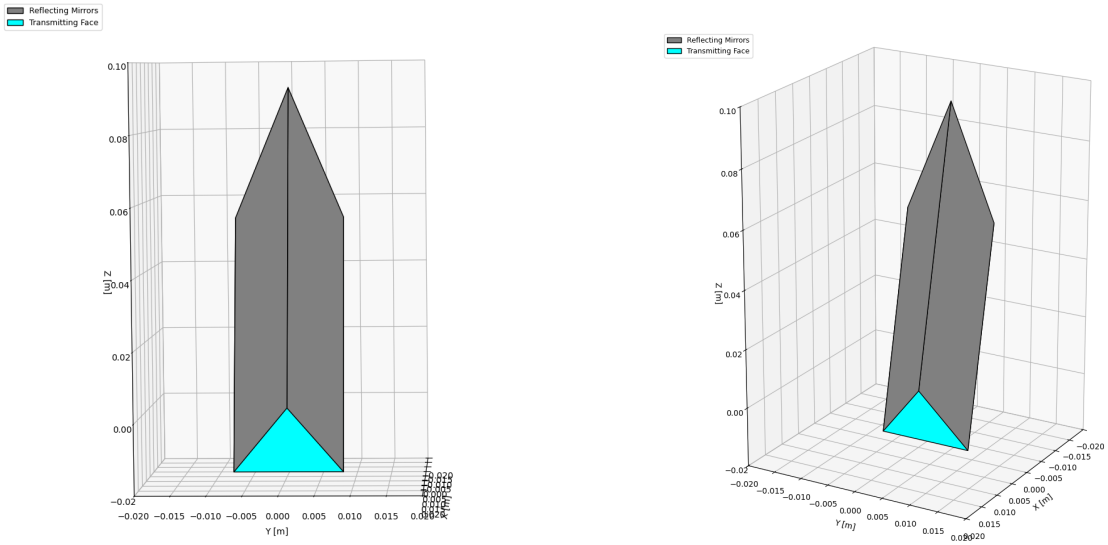


Figure 8: 3D plot of the Ingot prism.

4 Conclusions and Future Development

While simulating the LGS, we focused on balancing sampling resolution and code performance. Using fewer points speeds up the simulation but may reduce the realistic aspect. Therefore, an optimal balance was chosen for accurate yet memory-efficient results, keeping in mind the hardware constraints.

In addition to analyzing the LGS geometry, we also evaluated the computational performance of each grid style by timing the execution of the simulation. However, we found out that using fewer points (such as the Random grid), did not always result in faster runs. We can conclude that reducing the number of sampling points does not guarantee improved performance, but factors like code complexity, memory access and internal data handling may still dominate runtime.

Although the original IDL script defined six bounding planes for the Ingot geometry, only three were visualized and used in the simulation; two reflecting faces and one transmitting. This shape was chosen because it matches the dimensional parameters we wanted to represent the best while conserving its optical behavior of that geometry, as discussed in [4].

Our future work will be focused on validating the Ingot design even further.

More specifically, a first objective is to implement a complete end-to-end simulation to evaluate the Ingot WFS's capability to reconstruct aberrated wavefronts. This will involve introducing optical distortions into the simulation and analyzing how accurately the system can retrieve them under different atmospheric conditions.

Another important goal is to compare the simulation results with experimental data obtained from the laboratory setup in Padova. This comparison will provide a foundation for validating the simulation, preparing Ingot for a possible on-sky test, ultimately demonstrating its ability for integration into ELT-scale AO systems.

5 Bibliography

References

[1] R. Ragazzoni, E. Portaluri, V. Viotto, M. Dima, M. Bergomi, F. Biondi, J. Farinato, E. Carolo, S. Chinellato, D. Greggio, M. Gullieuszik, D. Magrin, L. Marafatto, and D. Vassallo. Ingot laser guide stars wavefront sensing. In *AO4ELT5*, 2017.

[2] E. Portaluri, V. Viotto, R. Ragazzoni, C. Arcidiacono, M. Bergomi, D. Greggio, K. K. Radhakrishnan Santhakumari, S. Di Filippo, L. Marafatto, M. Dima, F. Biondi, J. Farinato, and D. Magrin. Evaluating the performance of an ingot wavefront sensor for the elt: good news from simulations. In *Proceedings of the SPIE*, volume 11448, page 114483I, 2020.

[3] David Alaluf. Introduction to turbulence theory & adaptive optics, 2024.

[4] R. Ragazzoni, E. Portaluri, D. Greggio, M. Dima, C. Arcidiacono, M. Bergomi, S. Di Filippo, T. S. Gomes Machado, K. K. R. Santhakumari, V. Viotto, F. Battaini, E. Carolo, S. Chinellato, J. Farinato, D. Magrin, L. Marafatto, G. Umbriaco, and D. Vassallo. Ingot-like class of wavefront sensors for laser guide stars. *A & A*, 688:A21, 2024.

[5] V. Viotto, E. Portaluri, C. Arcidiacono, M. Bergomi, S. Di Filippo, D. Greggio, K. Radhakrishnan, M. Dima, J. Farinato, D. Magrin, L. Marafatto, and R. Ragazzoni. Ingot wavefront sensor: Simulation of pupil images. arXiv preprint arXiv:2012.06265, 2020.

[6] S. Di Filippo, D. Greggio, M. Bergomi, K. Radhakrishnan, E. Portaluri, V. Viotto, C. Arcidiacono, D. Magrin, L. Marafatto, M. Dima, R. Ragazzoni, P. Janin-Potiron, L. Schatz, B. Neichel, O. Fauvarque, and T. Fusco. Ingot wavefront sensor: from the optical design to a preliminary laboratory test. *A & A*, 647:A167, 2021.

[7] Kalyan Kumar Radhakrishnan Santhakumari, Davide Greggio, Maria Bergomi, Simone Di Filippo, Valentina Viotto, Elisa Portaluri, Carmelo Arcidiacono, Marco Dima, Luigi Lessio, Luca Marafatto, Tommaso Furieri, Stefano Bonora, and Roberto Ragazzoni. Aligning and testing the ingot wavefront sensor in the lab. In Laura Schreiber, Dirk Schmidt, and Elise Vernet, editors, *Adaptive Optics Systems VII*, volume 11448 of *Proceedings of the SPIE*, page 1144860, December 2020.

[8] T. S. Gomes Machado, S. Di Filippo, K. K. Radhakrishnan Santhakumari, M. Bergomi, D. Greggio, E. Portaluri, D. Malik, C. Nesme, C. Arcidiacono, A. Ballone, F. Battaini, V. Viotto, R. Ragazzoni, M. Dima, L. Marafatto, J. Farinato, D. Magrin, L. Lessio, and G. Umbriaco. New developments on the ingot wfs laboratory testing. In *Proceedings of the SPIE*, volume 12185, page 121852K, 2024.

[9] Valentina Viotto, Elisa Portaluri, Carmelo Arcidiacono, Roberto Ragazzoni, Maria Bergomi, Simone Di Filippo, Marco Dima, Jacopo Farinato, Davide Greggio, Demetrio Magrin, and Luca Marafatto. Dealing with the cigar: preliminary performance estimation of an INGOT WFS. In Laird M. Close, Laura Schreiber, and Dirk Schmidt, editors, *Adaptive Optics Systems VI*, volume 10703, page 107030V. Proceedings of the SPIE, 2018.