Facultad de Ciencias Físicas
Máster en Astrofísica

Título

Pointing and tracking with a 39m-class telescope: how to keep your position when all about you are losing theirs

Por:
Gonzalo José Carracedo Carballal

Tutores:
Javier Piqueras López (CAB, CSIC/INTA)
Fraser Clarke (University of Oxford)

Trabajo de Fin de Máster

Madrid, Septiembre de 2021

Contents

Abstract

Este trabajo tiene como objetivo el desarrollo de un modelo de apuntado para HARMONI, uno de los instrumentos de primera luz del Telescopio Extremadamente Grande (ELT). Puesto que HARMONI está diseñado para trabajar al límite de difracción del telescopio ($\sim$ 10 mas, $\sim$ 30 µm), la caracterización de los distintos efectos capaces de afectar a la posición de los puntos de la imagen en el plano focal es crítica.

Esto se conseguirá mediante la definición de un modelo optomecánico de HARMONI. El modelo servirá para validar el diseño final del instrumento, la eficacia del modelo de apuntado a la hora de corregir la firma instrumental de HARMONI y el coste de la estrategia de calibración. Este último dato servirá para optimizar tanto el tiempo requerido para caracterizar los efectos sistemáticos del instrumento como el diseño de la máscara de calibración en sí.

El modelo optomecánico se implementó en un simulador de apuntado escrito en Python 3. Algunos de los resultados producidos por este simulador son mapas de calor del error de apuntado, parámetros del modelo de apuntado, residuos del modelo y gráficos temporales del proceso de calibración.

Debido a que el conocimiento que se tiene de las propiedades del instrumento es limitada en esta fase de diseño, el simulador se debe considerar un prototipo. Sin embargo, su arquitectura software permite incluir futuros refinamientos del modelo optomecánico de forma sencilla y sienta las bases para su empleo de forma regular en el observatorio. Además, algunos de sus resultados (como la determinación de la mejor estrategia de calibración) ya son útiles y se espera su inclusión en los documentos técnicos de HARMONI.

* * *

The goal of this master thesis is the development of a pointing model for HARMONI, one of the first-light instruments of the Extremely Large Telescope (ELT). As HARMONI was designed to work in the diffraction limit of the telescope ($\sim$ 10 mas, $\sim$ 30 µm), the characterisation of the different effects that may affect the position of the points of the image in the focal plane is critical.

This will be achieved by the definition of an opto-mechanical model of HARMONI. This model will enable validation of the final design of the instrument, the effectiveness of the pointing model over HARMONI's instrumental signature and the cost of the calibration strategy. This latter result can be used both to optimise the time required to characterise the systematic effects of the instrument and the design of the calibration mask itself.

The opto-mechanical model was implemented inside a simulator written in Python 3. Some of the results produced by the simulator are pointing error heatmaps, parameters of the pointing model, model residuals and time plots of the calibration process.

As the available knowledge of the instrument was limited in this stage of design, the simulator must be considered a prototype. Nonetheless, its architecture enables future refinements of the opto-mechanical model in a simple manner, and sets the foundation for its regular usage during observatory operations. Additionally, some of its results (like the determination of the optimal calibration strategy) are already useful and its inclusion in the technical documents of HARMONI is expected.

1. ELT-HARMONI

1.1. Introduction

The Extremely Large Telescope (ELT) is a 39-m class adaptive telescope led by the European Southern Observatory (ESO), currently under construction in Cerro Armazones (Chile). By the time of its completion (circa 2026) it will be the largest VIS-IR telescope ever built. As other contemporary telescopes, ELT will be able to perform diffraction-limited observations by means of adaptive optics (AO) techniques. However, while current 8-m class telescopes can achieve angular resolutions of up to $\sim$ 50 mas (where mas stands for milliarcseconds), ELT expands this limit up to $\sim$ 10 mas. These unprecedented figures will vastly advance our knowledge in many branches of Astrophysics, from the study of the earliest galaxies to exoplanets[4].

Figure 1: Side by side size comparison between ELT and Sagrada Familia (credit: ESO)

HARMONI is a slicer-based integral field spectrograph (IFU) designed to operate in the 0.47 µm − 2.45 µm range ($3500 < R < 18000$) in a broad variety of scientific programs[8]. Its different spectral set-ups will enable both VIS-only observations as well as observations in the NIR bands (including H, J, K and portions thereof). It will enable both AO (including laser-tomography adaptive optics –LTAO–, single-conjugate adaptive optics –SCAO–, and high-contrast adaptive optics –HCAO–) and non-AO observations, with fields of view (FoV) of $9" \times 6", 4" \times 3", 2" \times 1.5"$ and $0.8" \times 0.6"$. In its highest resolution configuration, it will be able to resolve objects at a pixel scale of 4 mas/px in a $0.8" \times 0.6"$ FoV with a plate scale of 3.3mm/".

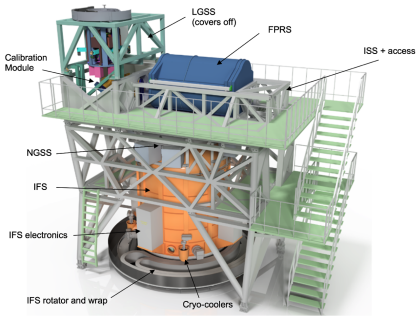


Figure 2: HARMONI: an integral field spectrograph for ELT.

In order to achieve this resolution level, both HARMONI and ELT must co-operate in closed loop mode, correcting its pointing continuously (order $\sim$ 1 Hz). This is done by measuring the position of natural guide stars (NGS) outside the science field (in the so-called technical field)

using a guiding probe in the form of a mobile mirror. The mobile mirror is installed at the end of a 200 mm Pick-Off Arm (POA) that can be placed anywhere in the technical field with a $\theta - \phi$ positioning stage. This positioning stage consists of two precision rotors with rotary encoders inserted in the Low-Order Wavefront Sensing subsystem (LOWFS). They provide the POA with the two necessary degrees of freedom that ensure complete coverage of the technical field, namely primary axis (θ) and secondary axis (ϕ).

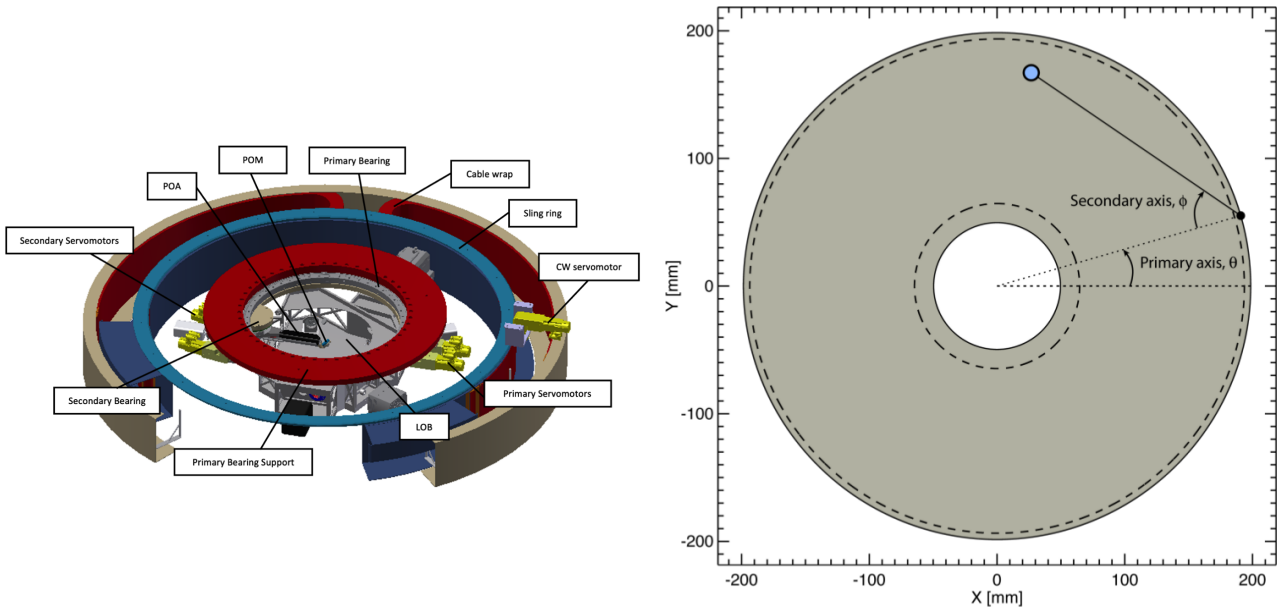


Figure 3: Left: low-order wavefront sensing subsystem (LOWFS). The black structure in the centre of the primary bearing is the POA. Right: degrees of freedom of the POA. Positive increments of ϕ represent clockwise motion of the arm.

LOWFS belongs to a higher-level system called Natural Guide Star Sensing System (NGSS). Its primary role is to track the position of natural stars both for pointing and wavefront sensing.

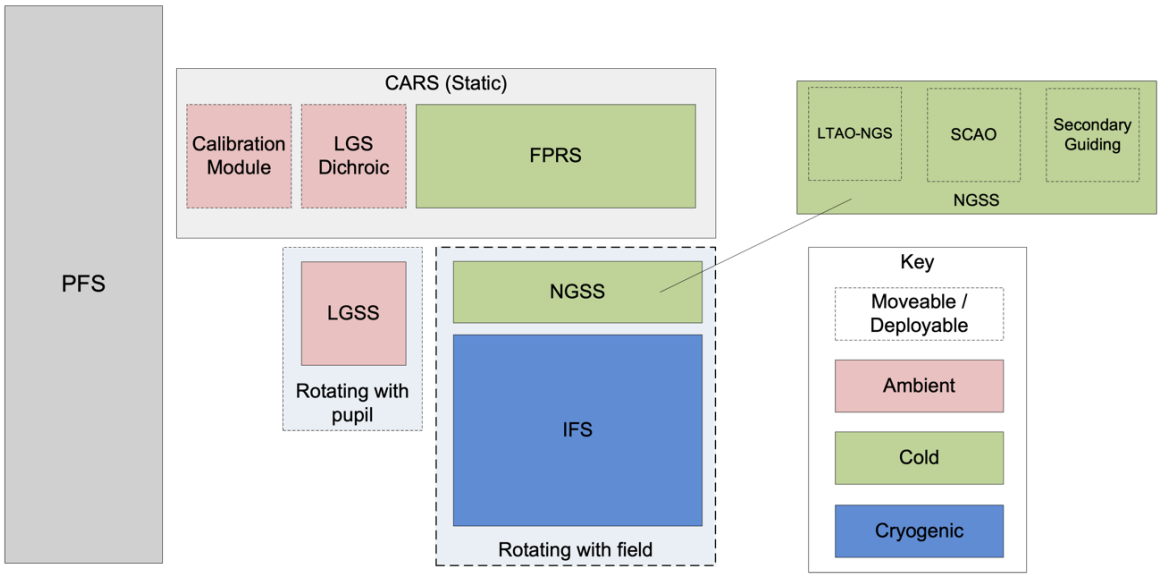


Figure 4: Top-level block diagram of the instrument [8].

Prior to its arrival to the NGSS, light entering the instrument (which includes science, laser

stars and natural guide stars) has traversed multiple stages. In a first stage, the prefocal station (PFS) installed in a Nasmyth platform routes the beam of light coming from the telescope to the different focal planes in which instruments are installed. Dichroic filters installed in the Calibration and Relay System (CARS) separate natural star light from laser star light. Finally, the optics of the Focal Plane Relay Subsystem (FPRS) re-focuses the natural star light from the telescope's focal plane to the NGSS in the so-called relayed focal plane. This plane holds a 1:1 magnification ratio with respect to the telescope's focal plane, and it is the plane upon which the POA senses guide stars.

1.2. Calibration

In order to exploit the diffraction-limit resolution of $\sim$ 10 mas, HARMONI's detector must Nyquist-sample the field of view. The current pixel scale (4 mas/px, 3.3mm/") ensures this requirement. Nonetheless, this also implies that the instrument must know the WCS coordinates of every spaxel with certain accuracy (R-HRM-153). In practice, this is not only determined by the current pointing of the instrument (determined by the reference pixel of the guide star), but also by the instrumental signature due to opto-mechanical aberrations in the optical path. This signature manifests itself as a certain distribution of systematic pointing errors along the relayed focal plane. A non-comprehensive list of contributions to this error could be the relay optics in the FPRS, wobblings of the derotator's bearings or mechanical tolerances of the structural components.

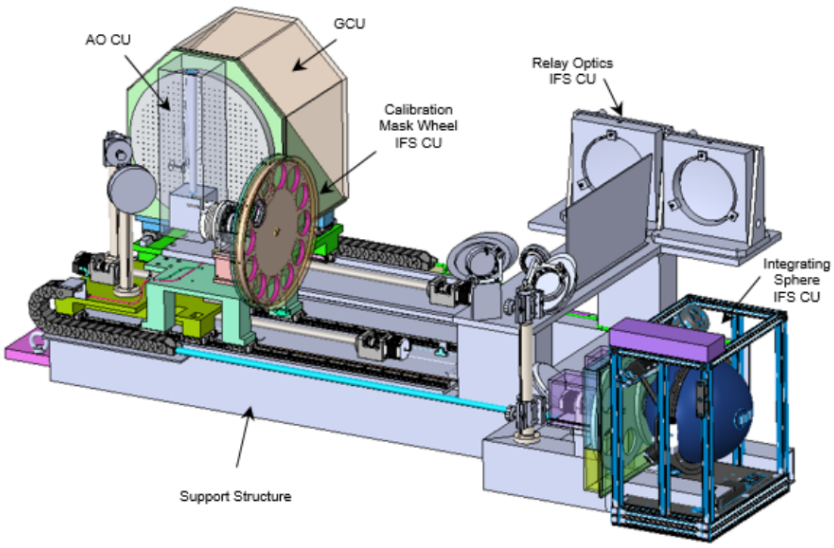


Figure 5: HARMONI's calibration module.

This error is calibrated by means of the the calibration module, in which the Geometrical Calibration Unit (GCU) is located. The GCU consists of a mask with a well-known pattern of barely resolved point sources that can be inserted across the telescope beam over the telescope focal plane. As the GCU mask pattern is known, the POA can be used to measure the apparent location of these point sources in the relayed focal plane and calculate the displacement with respect to their expected locations.

The result of the calibration is a corrective model (the pointing model) that compensates mechanical and optical displacements of the system up to the positioning accuracy of the POA ($\sim$ 10 µm). This model is then used to provide an absolute geometrical reference for the guiding probe over the whole technical field.

During operation, pointing error measurements will be fed to certain calibration software to obtain a pointing model. In order to optimise the number of free parameters of the model, the software must take into account not only the error measurements but also the nature of the processes causing them. Additionally, the pointing model may have less degrees of freedom than points in the GCU mask used for calibration. This means that the software may also provide hints to the control logic on how many and which calibration points are preferred.

1.3. Motivation for this work

The motivation of this work is the need for a characterisation of the performance of a pointing model under different sets of calibration points. This will be achieved by means of a software that will produce simulated measurements of the pointing error in different points. The simulator must take into account all effects that may affect the pointing, and therefore an opto-mechanical model of the instrument is required. The simulated measurements will then be used to fit a pointing model whose performance will be measured in terms of model complexity versus model residual.

The complexity of the instrument and the limited access to certain details of the instrument design prevent covering all the potential sources of pointing error in the time span of a master thesis. Instead, we will aim to provide a prototype simulator with a preliminary instrument model that attempts to account for the main contributions to the pointing error. Both the simulator and the instrument model are designed to ensure their extensibility in the form of additional error contributions.

2. Opto-mechanical model

The opto-mechanical model of the instrument must attain the following objectives:

1. Enable the characterisation of the accuracy of the measurement process of pointing errors by the POA.
2. Provide simulated measurements of the pointing error over the whole technical field, given the previous characterisation of the POA accuracy and
3. Evaluate different calibration strategies.

2.1. Working hypotheses

In this first iteration of the model, we agreed on the following set of assumptions. Many of them are based on technical documents of the instrument[10] describing the analysis of the WCS budgets in relation to top-level requirements of the instrument:

- The behavior of light is well described by geometric optics.
- Focal surfaces are flat (focal planes). Although this is not exactly true, we do not expect the pointing to be affected by the field curvature.
- Beams of light are described only by the location of their focal points. The description of the radiation field inside the telescope is reduced to 2D flux density fields (W/m^2). Diffraction effects are modelled as convolutions by the PSF in the flux fields.

- Pointing errors are observed in the focal planes and Z-axis errors (focus) are compensated by the wavefront sensor (WFS). This eliminates the necessity of tracing individual rays and reduces the description of light beams to 2D ($\mathbb{R}^2 \to \mathbb{R}^2$) transforms between focal points. These transforms map the $\boldsymbol{x}_1 = (x_1, y_1)$ coordinates of the point in the input focal plane to the $\boldsymbol{x}_2 = (x_2, y_2)$ coordinates of the output focal plane.
- Caustic-less regime. This implies that the reciprocal of each transform exists and is unique.

Hence the description of the behavior of light is reduced to providing the set of $T : \mathbb{R}^2 \to \mathbb{R}^2$ coordinate transforms such that $\boldsymbol{x}_2 = T[\boldsymbol{x}_1]$. This definition of the transforms –along with their invertibility– leads to some convenient properties. In particular, the total error transform can be split into individual transforms applied sequentially, each representing a conceptually different error contribution.

2.2. Parametrisation

The model accepts a set of parameters that may be either fixed values or random variables, being the latter the most numerous. For each random variable, not only the nominal value of the variable but also the specific probability distribution (along with other non-central parameters like its variance) must be provided. Additionally, parameters of the model are classified according to the time at which they are sampled:

- Fixed. The parameter describes a quantity that is known with absolute precision (e.g. bit resolution of an encoder).
- Instrument manufacturing. The variable describes a tolerance that is sampled upon manufacturing.
- Calibration session. The variable describes a quantity that remains constant during calibration (e.g. misalignment of the GCU after deployment).
- Arm positioning. The variable describes a quantity that remains constant after the arm is moved (e.g. true primary axis angle).

2.3. GCU mask

The fundamental reference for calibration is the GCU mask. The GCU mask consists of a circular plate with a pattern of barely resolved point sources whose geometry is assumed to be known beforehand with certain tolerance. When deployed, the GCU mask is inserted in the telescope's focal plane –right before the FPRS– and enables the characterisation of the opto-mechanical aberrations of the instrument.

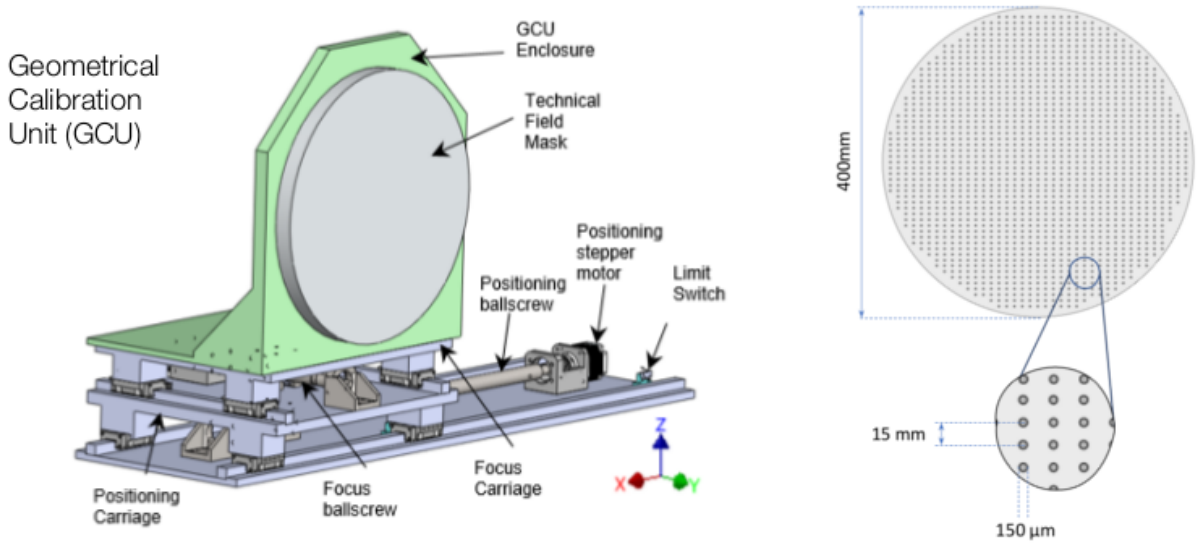


Figure 6: Detail of the GCU and the technical field mask (also known as GCU mask).

In the current iteration, the GCU mask is simply modelled as a square grid of homogeneously-illuminated circular points. The parameters defining the GCU mask model are:

Table 1: Parameters describing the GCU mask.

Symbol	Description	Sampling time
h_p	Point separation between consecutive centres	Fixed
D_p	Point diameter	Fixed
F_p	Point's spectral flux density (in dimensions of MT^{-2})	Fixed
x_0	X coordinate of the central point	Fixed
y_0	Y coordinate of the central point	Fixed
D_M	Mask diameter	Fixed

This model can be used to compute the locations of the mask dots, which are given by:

$$G = \{(x_0 + ih_p, y_0 + jh_p) | i, j \in \mathbb{Z}, \|(x_0 + ih_p, y_0 + jh_p)\| < D_M\} \subset \mathbb{R}^2 \tag{2.3.1}$$

Future iterations of the model may require a more realistic description of the GCU, with contributions like manufacturing tolerances acting individually on each mask dot.

2.4. Transform pipelines

The entry point of light of the instrument is always the telescope's focal plane, which may come either from the sky or the dot pattern of the GCU mask. Depending on whether we are characterising the behavior of the POA as a measurement device (objective 1) or simulating measurements of the pointing error (objective 2), we will use a different sequence of composed coordinate transforms (henceforth transform pipelines):

- The POA pipeline (T_{POA}), which translates coordinates in the telescope's focal plane to coordinates in the POA detector. It will be used to simulate the results of the pointing error measurement process, and
- The pointing error pipeline (T_ε), which translates coordinates in the telescope's focal plane to coordinates in the relayed focal plane *as measured by the POA*.

These pipelines, at the same time, can be further decomposed in two sets of transforms: a shared set (namely the common path) and a pipeline-specific set.

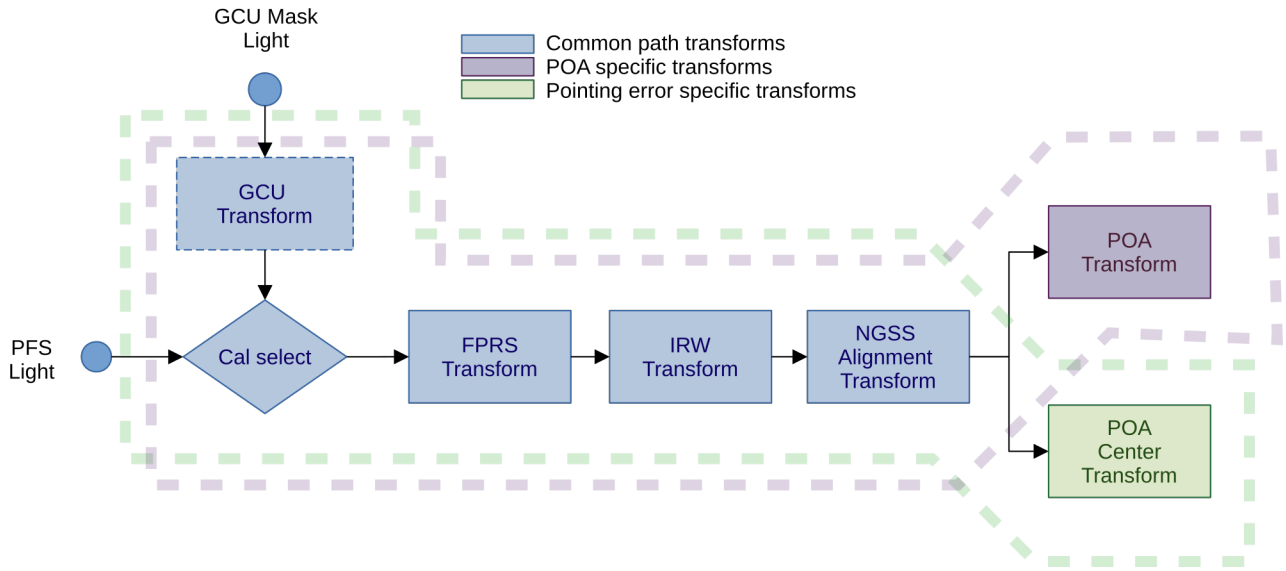


Figure 7: Block diagram of the transform pipelines. The POA pipeline (circled in purple dashed line) considers the light detected in the POA's focal plane (potentially the WFS detector plane). The pointing error pipeline (circled in green dashed line) considers light arriving at the relayed focal plane plus measurement uncertainties.

Both pipelines differ not only in the destination plane, but also in the interpretation of the coordinate change. While the POA pipeline models the physical behavior of light beams as they travel through the instrument, the pointing error pipeline also models the measurement uncertainty of the true location of bright objects in the relayed focal plane. This extra contribution is necessary as the measurement of the central location of a star (or a GCU mask dot) is performed by a fitting algorithm that can be affected by the resolution of the detector and the shape of the measured object.

2.4.1. Common path

In both cases, the optical path is shared up to the relayed focal plane in the Natural Guide Star Sensors System (NGSS). As previously mentioned, each pipeline can be broken down into a common transform and a pipeline-specific transform as:

$$T_{\mathrm{POA}} = T^{s}_{\mathrm{POA}}(\theta, \phi) \circ T_c \tag{2.4.1}$$

$$T_{\varepsilon} = T^{s}_{\varepsilon} \circ T_c \tag{2.4.2}$$

In this path, the light from the telescope's focal plane is relayed to the NGSS by the optics of the Focal Plane Relay Subsystem (FPRS) with an ideal magnification ratio of 1:1. Given the current knowledge of the instrument and its integration with ELT, the following contributions were considered:

Table 2: Random error contributions in T_c

Symbol	Description	Sampling time
Δx_G	GCU X axis misalignment (only if the GCU is inserted)	Manufacturing
Δy_G	GCU Y axis misalignment (only if the GCU is inserted)	Manufacturing
$\Delta\omega_{IRW}$	Angular error of the instrument's derotator	Calibration
Δx_N	X-axis misalignment of the NGSS structure	Manufacturing
Δy_N	Y-axis misalignment of the NGSS structure	Manufacturing

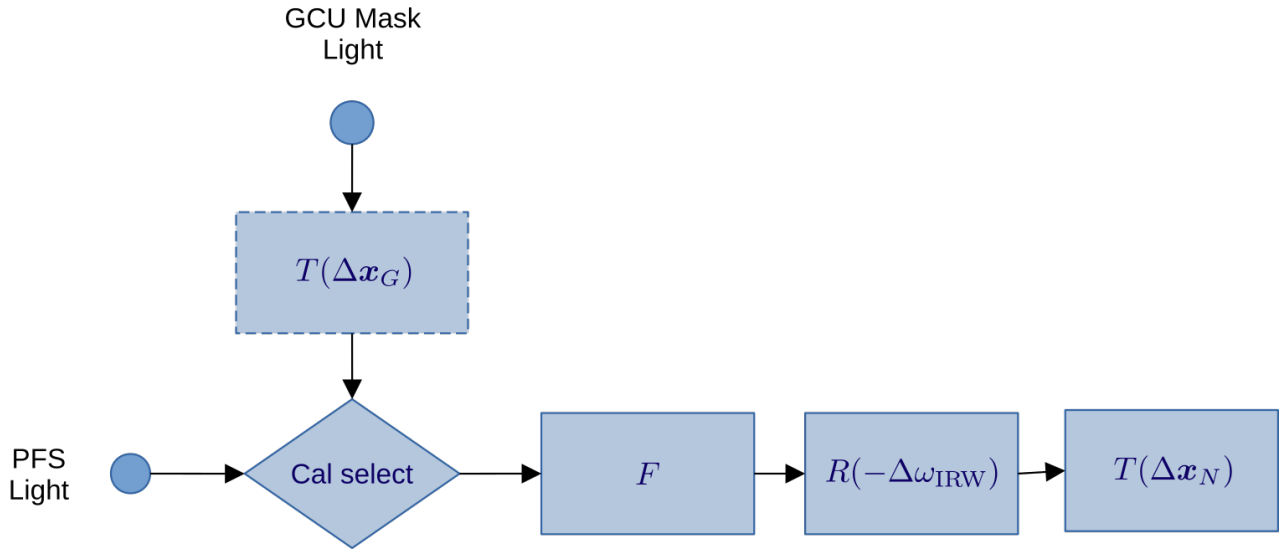


Figure 8: Modelisation of the common path.

The common transform T_c can be therefore expanded as follows:

$$T_c = T(\Delta \boldsymbol{x}_N) \circ R(-\Delta\omega_{\mathrm{IRW}}) \circ F \circ T(\Delta \boldsymbol{x}_G) \tag{2.4.3}$$

Where T is a translation transform by a given offset and R is a counter-clockwise 2D rotation transform perfectly described by a rotation matrix. F is the FPRS aberration transform which must be constructed by interpolation of the results of existing simulations of the FPRS optics. Details on the construction of the FPRS transform can be found in appendix B.

2.4.2. POA pipeline

The subpath specific to the POA pipeline relays a small section of the technical field in the surroundings of the POA's head of the technical field to the POA's detector plane, and is represented by $T^s_{\mathrm{POA}}(\theta, \phi)$. This subpath is dependent upon the specific POA configuration, which is completely determined by its axis angles θ, ϕ. An object in the relayed focal plane whose coordinates equals to those of the center of the POA's head at a given θ, ϕ configuration should appear (in the error-free case) exactly in the center of the detector plane ($x_d = 0, y_d = 0$). As the detector is expected to move together with the arm, the detector plane will experience a rotation of angle $\phi - \theta$ around its center with respect to the relayed focal plane.

The optics responsible for relaying the light between the POA's head and the POA detector belong to a component named Low-Order Optical Bench (LOB). For the sake of this study, it is assumed that the magnification ratio of the LOB optics is 1. This assumption could be revisited as the LOB design evolves in future versions of the model.

The random contributions to the pointing error considered in this subpath are:

Table 3: Fixed and random variables in $T^s_{\text{POA}}(\theta, \phi)$

Symbol	Description	Sampling event
B_θ	Bit resolution of the primary axis encoder	Fixed
B_ϕ	Bit resolution of the primary axis encoder	Fixed
q_θ	Quantisation error of the primary axis encoder	Positioning
q_ϕ	Quantisation error of the secondary axis encoder	Positioning
$\Delta\theta$	Angular error of the primary axis servomotor	Positioning
$\Delta\phi$	Angular error of the secondary axis servomotor	Positioning
e_R	Arm length tolerance	Manufacturing
ΔR	Arm length instability (integrated)	Positioning
$\Delta\rho$	Isotropic positioning errors (integrated)	Positioning

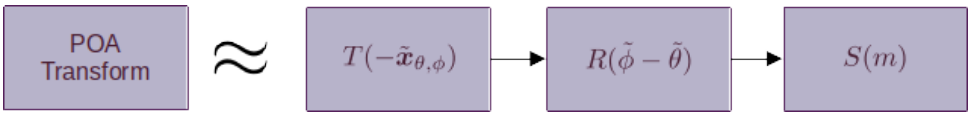


Figure 9: Modelisation of the POA path

$T^s_{\text{POA}}(\theta, \phi)$ can be further decomposed into the following transforms:

$$T^s_{\text{POA}}(\theta, \phi) = S(m) \circ R(\tilde{\phi} - \tilde{\theta}) \circ T(-\tilde{\boldsymbol{x}}_{\theta,\phi}) \tag{2.4.4}$$

Where R is the 2D counter-clockwise rotation transform and T the translation transform. S is a scale transform that depends on the magnification ratio m from the LOB optics. $\tilde{\theta}, \tilde{\phi}$ are the simulated angles of the POA axes, which are connected to the requested θ, ϕ angles by:

$$\tilde{\theta} = \frac{2\pi}{n_\theta}\left(\left\lfloor \frac{n_\theta}{2\pi}\theta \right\rfloor + q_\theta\right) + \Delta\theta \tag{2.4.5}$$

$$\tilde{\phi} = \frac{2\pi}{n_\phi}\left(\left\lfloor \frac{n_\phi}{2\pi}\phi \right\rfloor + q_\phi\right) + \Delta\phi \tag{2.4.6}$$

with n the number of discernible encoder intervals of each axis. This number is related to the bit resolution (B) of each encoder by $n = 2^B$.

On the other hand, the simulated POA's head center $\tilde{\boldsymbol{x}}_{\theta,\phi} = (\tilde{x}, \tilde{y})$ is obtained from $\tilde{\theta}, \tilde{\phi}$ as:

$$\tilde{x} = \tilde{R}\cos\,\tilde{\theta} - R\cos\,(\tilde{\phi} - \tilde{\theta}) + \Delta\rho\cos\,\alpha \tag{2.4.7}$$

$$\tilde{y} = \tilde{R}\sin\,\tilde{\theta} + R\sin\,(\tilde{\phi} - \tilde{\theta}) + \Delta\rho\sin\,\alpha \tag{2.4.8}$$

with α following a uniform distribution between 0 and 2π and $\tilde{R} = R + e_R + \Delta R$.

2.4.3. Pointing error pipeline

This pipeline is used to generate simulated measurements of the pointing error in the technical field. By taking advantage of the fact that the magnification ratio between the telescope's focal plane and the relayed focal plane is 1, the pointing error can be defined as:

$$\boldsymbol{\varepsilon} = T_\varepsilon[\boldsymbol{x}] - \boldsymbol{x} \tag{2.4.9}$$

Which can be used to train an appropriate corrective model for the systematic part of the pointing error.

The subpath specific to this pipeline is represented by the transform $T_\varepsilon^s[\boldsymbol{x}]$ and simulates the uncertainties of measuring the location of an object in the relayed focal plane. The calculation of $T_\varepsilon^s[\boldsymbol{x}]$ is a multi-step process that involves:

1. Calculating the nominal arm configuration θ, ϕ that centers the head around $\boldsymbol{x}$ as described in appendix A.
2. Simulating $\tilde{\boldsymbol{x}}_{\theta,\phi}$ from the requested θ, ϕ as described in equations 2.4.7 and 2.4.8, and
3. Simulating the measurement error by the fitting algorithm

Figure 10: Modelisation of the pointing error subpath.

The set of considered error contributions is the same as for the POA pipeline, plus the measurement error of the fitting algorithm. Nevertheless, as many details of the measurement process are yet to be decided, the exact error distribution of the fitting algorithm is currently unknown and therefore not included in the current iteration of the model. This reduces the definition of T_ε^s to:

$$T_\varepsilon^s[\boldsymbol{x}] = \tilde{\boldsymbol{x}}_{\theta(\boldsymbol{x}),\phi(\boldsymbol{x})} \tag{2.4.10}$$

And, in the case we are evaluating the performance of certain corrective model C such that $\boldsymbol{x}_c = C[\boldsymbol{x}]$:

$$T_\varepsilon^s[\boldsymbol{x}] = \tilde{\boldsymbol{x}}_{\theta(\boldsymbol{x}_c),\phi(\boldsymbol{x}_c)} \tag{2.4.11}$$

2.5. Kinematics of the POA

The third objective of the model was to enable comparisons between different calibration strategies. Since one of the figures of merit of a calibration strategy is how fast it completes the measurement of a set of points, we need to take the kinematics of the POA into account.

The estimation of the calibration time involves knowing the exact sequence of POA configuration requests and the behavior of the servomotor of each axis. The current iteration of the model assumes that the kinematics of the POA is purely governed by the two error contributions detailed in table 4.

Table 4: Random parameters of the POA kinematics

Symbol	Description	Sampling time
ω_θ	Maximum sweep speed of the primary axis servomotor	Positioning
ω_ϕ	Maximum sweep speed of the secondary axis servomotor	Positioning

As a first order approximation, the current model assumes that both servomotors can rotate immediately at their maximum sweep speed until they reach their desired angle. This implies that the completion time of a given $\Delta\theta, \Delta\phi$ sweep is given by:

$$t = \max\left(\frac{\Delta\theta}{\omega_\theta}, \frac{\Delta\phi}{\omega_\phi}\right) \tag{2.5.1}$$

A noteworthy consequence of this behaviour is that, since both servomotors run independently and in parallel, one axis may reach its desired angle way before the other. When this happens, the sweep transitions from two servomotors running to only one, causing a sudden variation both in sweep speed and direction. The transition point should be reflected as a small cusp in the POA path.

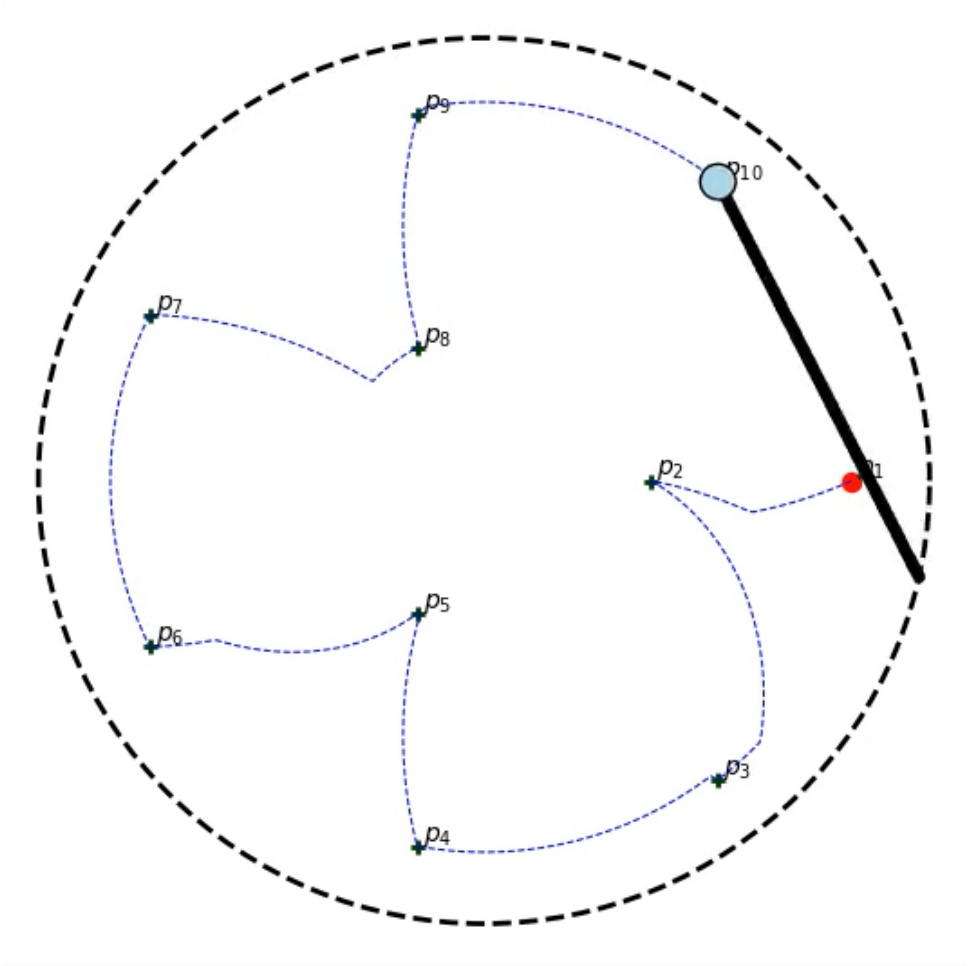


Figure 11: Traversal of 10 calibration points. The POA path from the first point (p_1, marked in red) to the last (p_{10}) exhibits cusps as one axis reaches its destination angle before the other. This effect is particularly noticeable in path sections $p_1 - p_2$, $p_5 - p_6$ and $p_7 - p_8$.

Although we do not expected that the true kinematics deviate much from this description, a complete model should include startup times, the effects of the POA's inertia, angle correction near the desired position and potential memory effects that may depend on past histories of arm displacements.

2.6. Pointing model

The goal of any calibration strategy is to gather the necessary data to fit a pointing model, whose performance analysis also falls within the scope of this study. This model consists of a certain 2D transform C that attempts to mimic the behavior of T_ε all over the technical field, minimising the residual $r = \|T_\varepsilon[\boldsymbol{x}] - C[\boldsymbol{x}]\|$. In a similar way as we did for $T_\varepsilon[\boldsymbol{x}]$ in equation 2.4.9, C can be expressed in terms of the pointing error model $\boldsymbol{\varepsilon}'(\boldsymbol{x})$ as:

$$C[\boldsymbol{x}] = \boldsymbol{x} + \boldsymbol{\varepsilon}'(\boldsymbol{x}) \tag{2.6.1}$$

and therefore the residual can also be written as:

$$r = \|\boldsymbol{\varepsilon}'(\boldsymbol{x}) - \boldsymbol{\varepsilon}(\boldsymbol{x})\| \tag{2.6.2}$$

In order to exploit the maximum resolution achievable by the telescope ($\sim$ 8 mas), the pointing model must ensure that $r < 10$ µm for all points in the relayed focal plane.

In real life operation, C is found during calibration by measuring the position of the GCU mask dots by means of the POA detector. As the time available for calibrations is limited, a good pointing model C should not only minimise the residual, but also the time required to gather the necessary data, which is directly connected to the number of points used for calibration. Consequently, we will favour models that require fewer points.

With these considerations in mind, we decided to express the pointing model C in terms of the complex Zernike expansion of the pointing error [9]. The rationale behind this approach is twofold:

- Zernike expansions are familiar to optical engineers when it comes to describing optical aberrations, and
- the first radial orders describe aberrations that can be connected to common opto-mechanical effects (misalignments, rotations...).

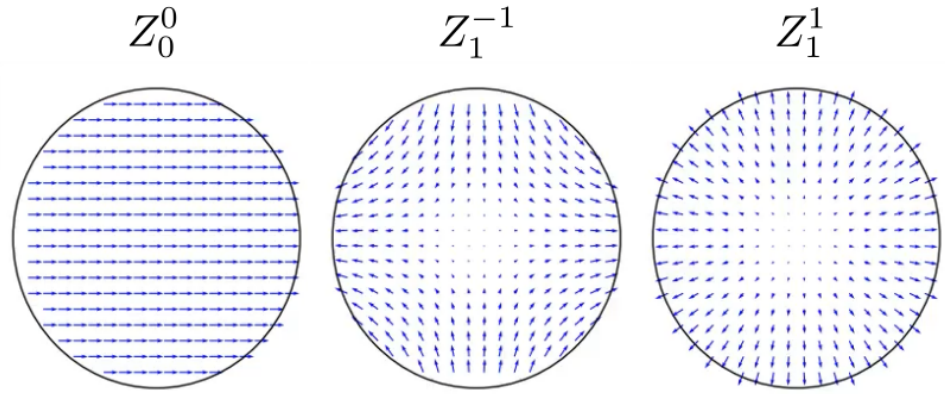


Figure 12: Complex Zernike polynomials up to radial order $n = 1$. Z_0^0 can be used to model field misalignments, Z_1^{-1} changes in the aspect ratio and Z_1^1 both changes in the global magnification ratio and field rotations (i.e. plate scale and field orientation).

There are some additional benefits derived from using Zernike polynomials: as they conform a complete orthogonal basis on the unit circle [5], coefficients of a Zernike expansion can describe properties of the aberration without redundancy or overlap of information between them (which provides insight on the nature of the aberrations). On the other hand, the algebra of a 2D Zernike expansion is greatly simplified when complex coefficients are used. In particular, if we identify 2D vectors with complex scalars as:

$$\boldsymbol{x} = \mathfrak{Re}[z]\hat{e}_x + \mathfrak{Im}[z]\hat{e}_y, z \in \mathbb{C} \tag{2.6.3}$$

Then, the pointing error model can be expanded as:

$$\varepsilon'(z) = \sum_{n=0}^{N} \sum_{i=0}^{n} a_n^{-n+2i} Z_n^{-n+2i} \left(\frac{z}{R}\right) \tag{2.6.4}$$

Where a_n^m is the complex coefficient multiplying the complex Zernike polynomial Z_n^m and N is the maximum radial order of the expansion. As Zernike polynomials are defined on the unit circle, z must be normalised prior to the evaluation by the technical field radius R. Note that for a maximum radial order N we need to fit $J = (N+1)(N+2)/2$ complex coefficients.

When Zernike polynomials are used to build certain vectors and matrices, it is customary to refer to the first J polynomials. In these cases, it is convenient to map the two indices m, n of the Zernike polynomials to a single integer index $j \geq 0$. For the sake of standarisation, we chose OSA/ANSI indices[2], which relate j and m, n by:

$$j = \frac{n(n+2)+m}{2} \tag{2.6.5}$$

2.6.1. Model fit

Fitting the aforementioned pointing model requires deciding both the maximum radial order of the model (N) and the calibration pattern $P = \{p_0, p_1, ...\} \subseteq G \subset \mathbb{C}$, with G the set of locations of the GCU mask dots in complex form. Ideally, we would need as many points as free parameters: $Q = J = (N+1)(N+2)/2$, with $Q = \text{Card}(P)$. However, since measurements are affected by noise, it could be convenient to introduce certain redundancy by choosing more than J calibration points.

Once the error measurement vector $\tilde{\boldsymbol{\varepsilon}} = (\tilde{\varepsilon}(p_0), \tilde{\varepsilon}(p_1), \cdots, \tilde{\varepsilon}(p_{Q-1}))^T \in \mathbb{C}^Q$ has been obtained, one can pose the model fitting problem as finding $\boldsymbol{a} = (a_0, a_1, \cdots, a_{J-1})^T \in \mathbb{C}^J$ such that:

$$\begin{pmatrix} Z_0\left(\frac{p_0}{R}\right) & Z_1\left(\frac{p_0}{R}\right) & \cdots & Z_{J-1}\left(\frac{p_0}{R}\right) \\ Z_0\left(\frac{p_1}{R}\right) & Z_1\left(\frac{p_1}{R}\right) & \cdots & Z_{J-1}\left(\frac{p_1}{R}\right) \\ \vdots & \vdots & \ddots & \vdots \\ Z_0\left(\frac{p_{Q-1}}{R}\right) & Z_1\left(\frac{p_{Q-1}}{R}\right) & \cdots & Z_{J-1}\left(\frac{p_{Q-1}}{R}\right) \end{pmatrix} \times \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{J-1} \end{pmatrix} = \begin{pmatrix} \tilde{\varepsilon}(p_0) \\ \tilde{\varepsilon}(p_1) \\ \vdots \\ \tilde{\varepsilon}(p_{Q-1}) \end{pmatrix} \tag{2.6.6}$$

The coefficient matrix of this system of equations is referred to as the collocation matrix[7], and it is denoted by $Z(P)$. When $Q > J$ the system is generally inconsistent due to the measurement noise. Nevertheless, a solution can still be found if we reformulate it as an optimisation problem:

$$\boldsymbol{a} = \underset{\boldsymbol{a}}{\text{argmin}} \|Z(P)\boldsymbol{a} - \tilde{\boldsymbol{\varepsilon}}\|^2 \tag{2.6.7}$$

The quality of the calibration pattern P is given not only by the number of points in it, but also by how evenly it samples the technical field. This is directly reflected in the reduction of the condition number of $Z(P)$, indicating higher robustness against measurement noise.

2.6.2. Calibration strategies

A calibration strategy is the choice of a calibration pattern P along with the order in which it should be traversed that ensures a calibration residual below 10 µm. A good calibration strategy is a trade-off between the quality of the calibration pattern and the time required to measure it. Since finding this trade-off is non-trivial, different calibration patterns and orderings should be tested. In the current iteration of the model, we propose three calibration patterns:

- Random (i.e. calibration points are chosen randomly from G).
- Spiral (closest GCU points to the spiral pattern described in [3]).
- Optimal Concentric Sampling (closest GCU points to the pattern described in [7]).

Finding the ordering that minimises the calibration time is a particular case of the Traveling Salesman Problem and can only be addressed via exhaustive search of all the $Q!$ possible paths. Instead, we propose two heuristics: as-is (traversing the points in a fixed arbitrary order), and closest neighbour (the next point to be traversed is the closest to the last one in infinity-norm distance).

3. Design and implementation

The resulting application must be understood as a prototype that will eventually be used in the observatory during operation. As such, it should not only address the functional requirements selected for this first iteration, but also foresee future extensions of the model by other contributors until then. The goal of the project is to provide the observatory with a useful tool to optimise the calibration strategies of the instrument. Its architecture should be easily upgraded to account for a more realistic description of the "as-built" instrument.

3.1. Software requirements

The simulator will consist of a series of programs which, upon successful execution, will produce different simulation products in the form of data files and graphs. From the perspective of functional requirements, the prototype must produce the following results:

1. Heatmaps of the pointing error in the relayed technical field.
2. Priors of the coefficients of the pointing model via Monte Carlo sampling.
3. Curve of residuals for different calibration strategies.
4. Heatmaps of the pointing error residual after applying the pointing model.
5. Time plots of POA's operation for different calibration strategies.

In addition to the goals described in section 2, the design must be aware of the incompleteness of the model. In particular, we must assume that other programmers with varying sets of skills will eventually be involved in the project. Also, as other use cases may be identified in the future (e.g. integration with the observatory's control software) proper component decoupling will be necessary. All this translates to the following set of non-functional requirements (NFR):

1. The programming language of the project must be the result of a trade-off between performance and popularity.
2. The architecture of the application must be decided beforehand.
3. The code of the entry point of the simulator (i.e. the executable programs) must be decoupled from the simulator's logic.
4. Implementation must prioritise maintainability before performance.
5. Simulator must work out of the box, even if no parameters are provided.
6. Exhaustive code and application documentation must be provided.

NFR 1 is satisfied by choosing Python 3 + NumPy. This is a popular combination among physicists and engineers that enables high-performance computing with tensors of arbitrary rank. Additionally, the functional requirements of the simulator can be fulfilled by a component-based architecture (NFR 2) which must be properly documented (NFR 6) prior to the coding phase.

3.2. Simulator architecture

NFRs 2 and 3 motivated a component-based architecture. The goal of this paradigm is to decompose the design of the application into individual functional or logical components that represent well-defined interfaces containing classes. It is the object-oriented equivalent of the well-known programming principle of divide-and-conquer.

In order to ease future code reuse, terminal components (i.e. those representing executable scripts) are separated from non-terminal components, with the latter being placed inside a Python package named `harmoni_pm`.

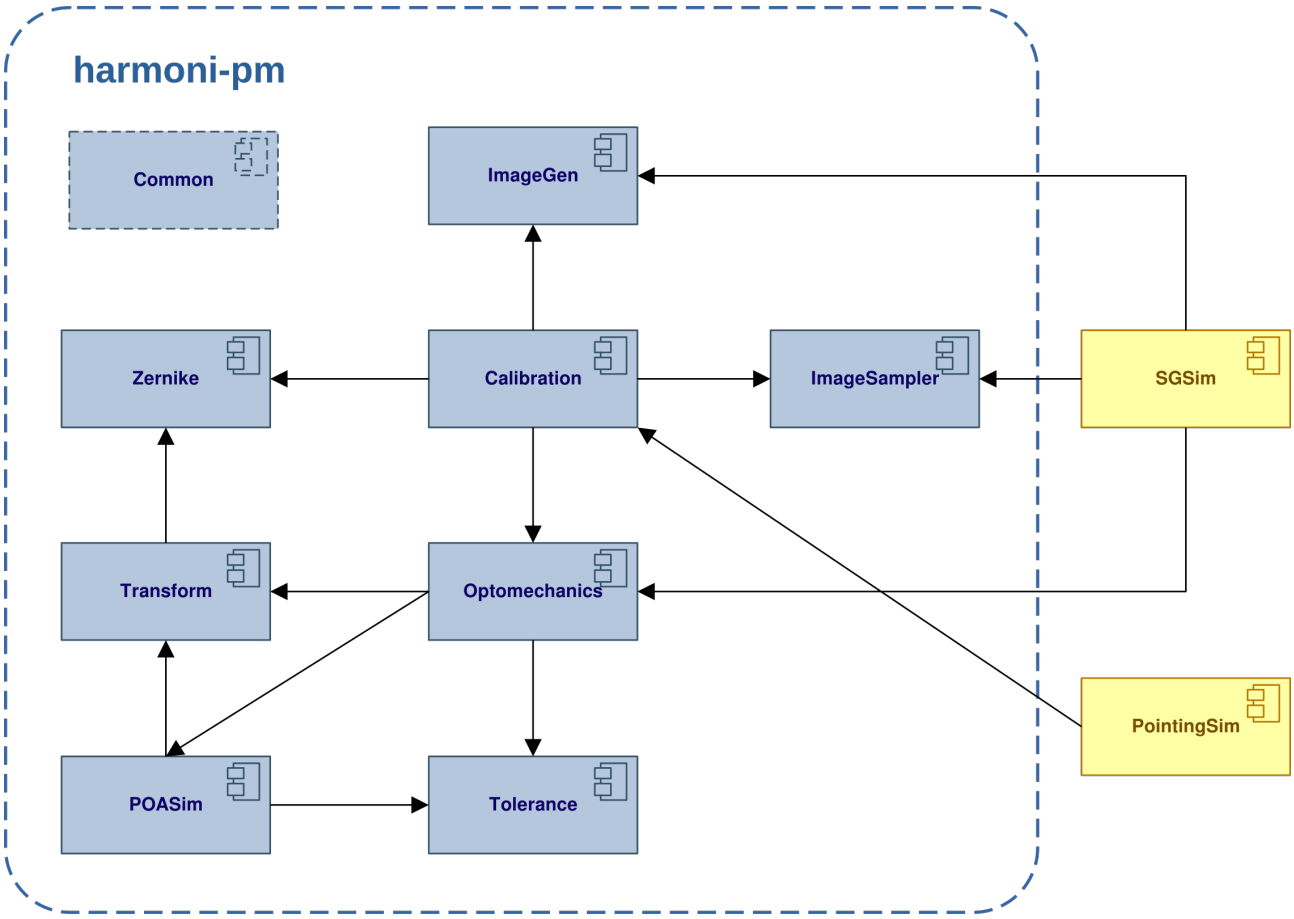


Figure 13: Component diagram of the project.

The responsibilities held by each component are summarised in the following table:

Component	Responsibilities
`Calibration`	Measurement and correction of pointing errors
`Common`	Utility module
`ImageGen`	Generation of light sources as flux fields
`ImageSampler`	Sampling functions defined in $\mathbb{R}^2$
`Optomechanics`	Instrument model
`POASim`	POA model
`Tolerance`	Random quantity handling
`Transform`	2D coordinate transforms
`Zernike`	Generation and evaluation of complex Zernike polynomials

Terminal components `PointingSim` and `SGSim` refer to the scripts that may be executed by the user. `PointingSim` will be in charge of producing the results enumerated in 3.1, while `SGSim` will be the prototype of detector simulator that will be used in the future to characterise the behavior of the POA. Although feature-limited and not used in this work, current progress on the latter is detailed in appendix D.

3.3. Source code

The source code of the simulator can be accessed from its public GitHub repository at `https://github.com/BatchDrake/harmoni-pm`, while the documentation of the internal API can be accessed at `https://actinid.org/harmoni-pm`. The usage of Python 3 as the project's programming language should ensure portability to other platforms. However, at the current stage of development, it is recommended to run the code in Unix-like operating systems.

Project files are structured so that every component of the simulator is kept in a separate directory under the `harmoni_pm` package, while executable scripts are kept at root level of the project directory.

In order to run the different executable scripts, the user must ensure the following `pip` dependencies are met: `chardet`, `matplotlib`, `numpy`, `pandas`, `Pillow`, `Pint`, `seaborn`, `scipy`, `skyfield`, `uncertainties`

4. Results and applications

This section details the results produced by simulations of different aspects of the instrument. Final values used for the parameters of the opto-mechanical model are as follows:

Table 5: Values of the opto-mechanical models used in the simulations, as specified in `harmoni.ini`. Conventions used for value representation are as in table 7.

Section	Key	Symbol	Type	Value	Reference
`fprs`	`desc_file`	N/A	S (file path)	`"FPRS_distortion_map.txt"` (default)	N/A
`gcu`	`point.separation`	h_p	R (metres)	15×10^{-3}	Table 1
`gcu`	`point.diameter`	D_p	R (metres)	150×10^{-6}	Table 1
`gcu`	`point.flux`	F_p	R (SIFU)	10^{-3}	Table 1
`gcu`	`mask.x0`	x_0	R (metres)	0	Table 1
`gcu`	`mask.y0`	y_0	R (metres)	0	Table 1
`gcu`	`mask.diameter`	D_M	R (metres)	0.39	Table 1
`gcu_alignment`	`x0`	Δx_G	D (length)	0 ± 5 µm (N)	Table 2
`gcu_alignment`	`y0`	Δy_G	D (length)	0 ± 5 µm (N)	Table 2
`irw`	`angle_bias`	$\Delta\omega_{IRW}$	D (angle)	0 ± 15" (N)	Table 2
`ngss_alignment`	`x0`	Δx_N	D (length)	0 m (δ)	Table 2
`ngss_alignment`	`y0`	Δy_N	D (length)	0 m (δ)	Table 2
`poa`	`encoder[theta].qerr`	q_θ	D (no dim.)	0.5 ± 0.5 (pp)	Table 3
`poa`	`encoder[theta].bits`	B_θ	I	21	Table 3
`poa`	`encoder[theta].err`	$\Delta\theta$	D (angle)	$0 \pm 1°$ (N)	Table 3
`poa`	`encoder[theta].speed`	ω_θ	D (ang. freq.)	$1°/s$ (pp)	Table 4
`poa`	`encoder[phi].qerr`	q_ϕ	D (no dim.)	0.5 ± 0.5 (pp)	Table 3
`poa`	`encoder[phi].bits`	B_ϕ	I	21	Table 3
`poa`	`encoder[phi].err`	$\Delta\phi$	D (angle)	$0 \pm 1°$ (N)	Table 3
`poa`	`encoder[phi].speed`	ω_ϕ	D (ang. freq.)	$1°/s$ (pp)	Table 4
`poa`	`radius`	e_R	D (length)	0.2 ± 10^{-6} m (pp)	Table 3
`poa`	`arm_instability`	ΔR	D (length)	0 m (δ)	Table 3
`poa`	`position_error`	$\Delta\rho$	D (length)	0 m (δ)	Table 3

Although the opto-mechanical model is expected to be refined prior to its final integration in the observatory, some of its simulation products have direct application in terms of calibration

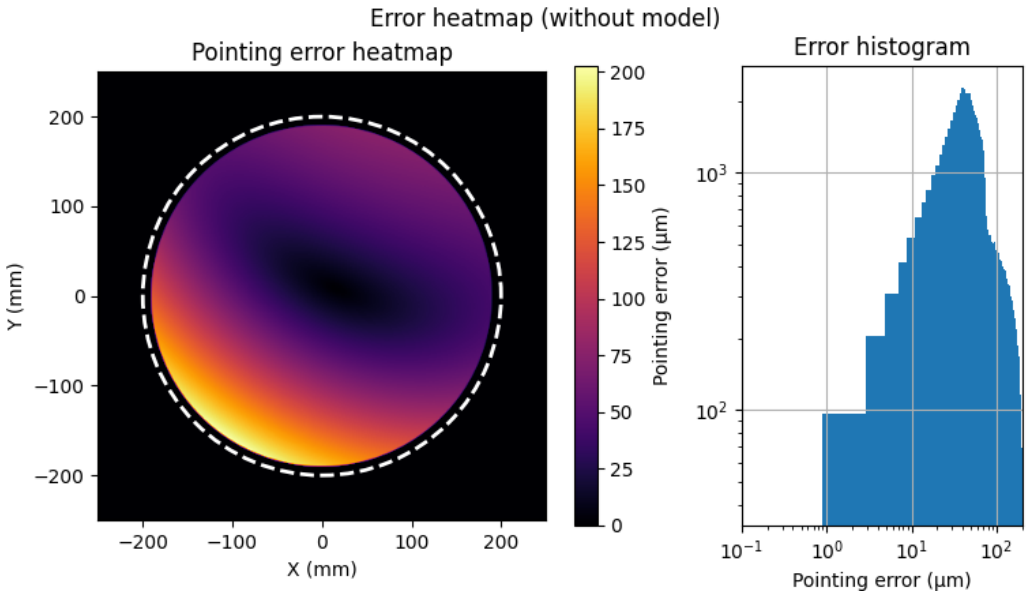
strategy. In particular, we provide a thorough comparison of the performance of different calibration strategies, providing strong evidence of the superiority of the OCS pattern with closest neighbour traversal. These results are expected to be included in the technical documentation of the instrument with the aim of guiding the design of the calibration control software.

Model-limited results are also included (like pointing error heatmaps before and after applying the pointing model). Although these results would be revisited in future phases of the instrument development, they already provide useful insight on the effects of using pointing models of different radial orders.

4.1. Study of the pointing model

4.1.1. Uncorrected error heatmap

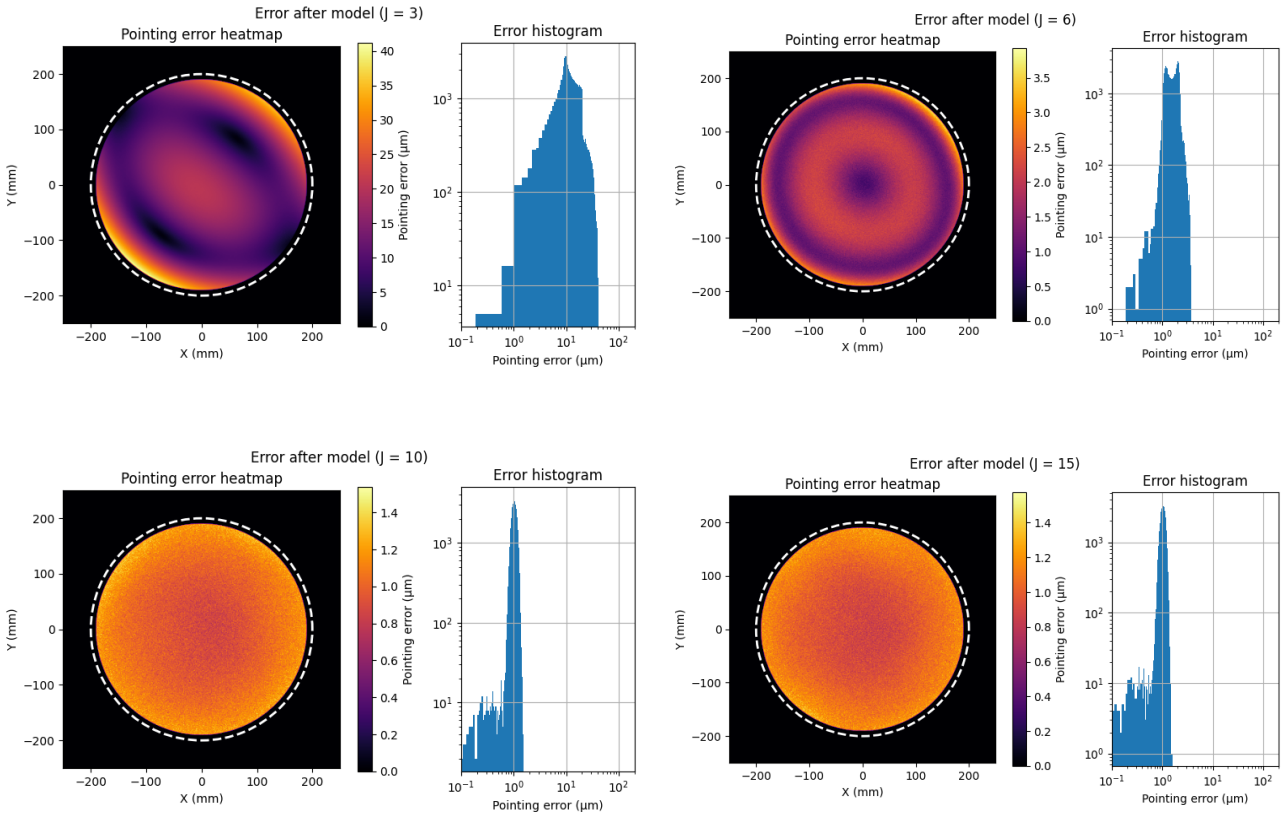
One of the questions we wanted to answer regarding the current model was which was the main contribution to the pointing error. In doing so, we evaluated $\tilde{\boldsymbol{\varepsilon}}$ in the whole relayed focal plane as seen by the POA using `PointingSim`'s subcommand `errormap`, obtaining a maximum error of around 200 µm (with typical values around 100 µm). This value surpasses the goal of 10 µm by more than one order of magnitude, and it has been tracked to the optical aberrations inside the FPRS.



4.1.2. Radial order of the pointing model

Following, we wanted to know whether complex Zernike polynomials were a good choice for both optical and mechanical aberrations, and what was the minimal radial order (N) required to describe them with an error below 10 µm. For this purpose, we simulated a full calibration (i.e. comprising all possible test points) fitting the coefficients of several complex Zernike expansions with N ranging from 1 to 4. The total number of coefficients (J) fitted in each case is related to the radial order by $J = (N + 1)(B + 2)/2$.

The heatmap represents the residual as defined by equation 2.6.2. Note that for the case $N = 1$, only displacements, rotations and changes in aspect ratio can be modelled:



We see that although the goal accuracy is achieved when $N = 2$ ($J = 6$), we still observe structure in the heatmap residual. This structure disappears at higher orders, and the residual is found to be lower-bounded by a noise floor at $r = 1.5$ µm. In the current iteration of the simulator, this floor is dominated by the angular error of the POA encoders ($\sigma_{\Delta\theta} = \sigma_{\Delta\phi} = 1$ as).

We therefore conclude that there are two candidate configurations of the pointing model, one with $N = 2$ and other with $N = 3$. Radial orders below 2 are too simple to meet the $r < 10$ µm goal, while radial orders above 3 do not provide significant improvements of the residual. This tie can be broken by making a trade-off between model accuracy and calibration time.

4.2. Priors of the aberration model

`PointingSim` is able to compute histograms of the coefficients of the pointing model by simulating multiple calibrations using the maximum number of calibration points. The resulting histograms of the coefficients of the aberration model can be used to model the priors of the bayesian formulation of the calibration problem:

$$p(a_j, ...|\tilde{\varepsilon}_i, ...) \propto p(\tilde{\varepsilon}_i, ...|a_j, ...)p(a_j, ...) \tag{4.2.1}$$

where $a_j, 0 \leq j < J$ are the model coefficients, $\tilde{\varepsilon}_i, 0 \leq i < Q$ the measured errors and $p(\tilde{\varepsilon}_i, ...|a_j, ...)$ the likelihood function, whose exact formulation requires a good understanding of the measurement process.

For illustrative purposes, we simulated a total of 2×10^4 calibrations over individual realisations of manufacture-time values, using all calibration points in the simulated GCU mask. The results of the simulation shows high statistical independence between coefficients, which suggests that the bayesian problem may be simplified down to J smaller problems:

$$p(a_j|\tilde{\varepsilon}_i, ...) \propto p(\tilde{\varepsilon}_i, ...|a_j)p(a_j), 0 \leq j < J \tag{4.2.2}$$

Full histograms and scatter plot matrices of the simulated coefficients can be found in appendix E

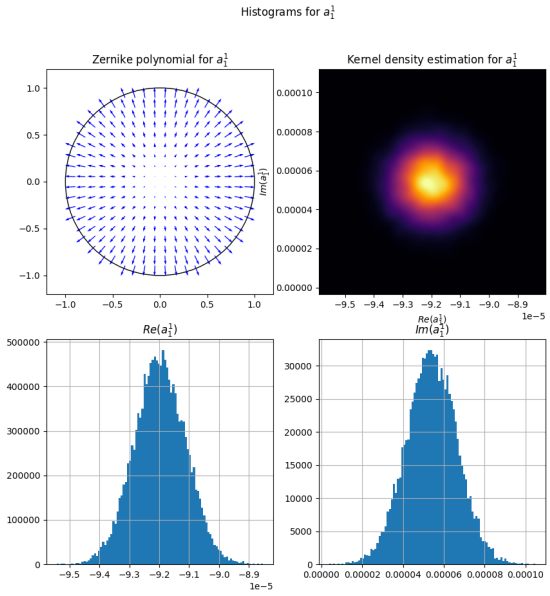


Figure 16: Typical histogram for a simulated complex pointing model coefficient (a_1^1 for Z_1^1). Top left: vector representation of the complex Zernike polynomial the coefficient refers to. Top right: kernel density estimation of the prior histogram in the complex plane. Bottom left: marginal histogram of the coefficient's real part. Bottom right: marginal histogram of the coefficient's imaginary part.

4.3. Calibration pattern

One of the results with immediate application comes from the performance analysis of all calibration patterns with different pointing model orders and numbers of points. This comparison was performed by fitting aberration models for each pattern (random pattern, spiral sampling and optimal concentric sampling - OCS) with radial orders 2 ($J = 6$), 3 ($J = 10$) and 4 ($J = 15$), different numbers of calibration points between 1 and 30 and 50 different realisations of each pattern. The quantity used for comparison was the median of the 50 mean residuals calculated for each pattern realisation. Error bars represent the maximum and minimum observed mean error observed in the 50 realisations of the pattern.

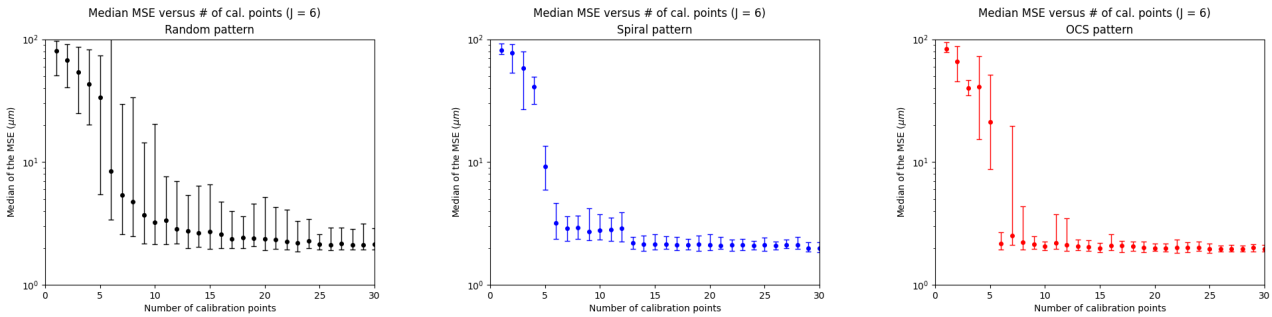


Figure 17: Median mean calibration error after fitting up to radial order $n = 2$.

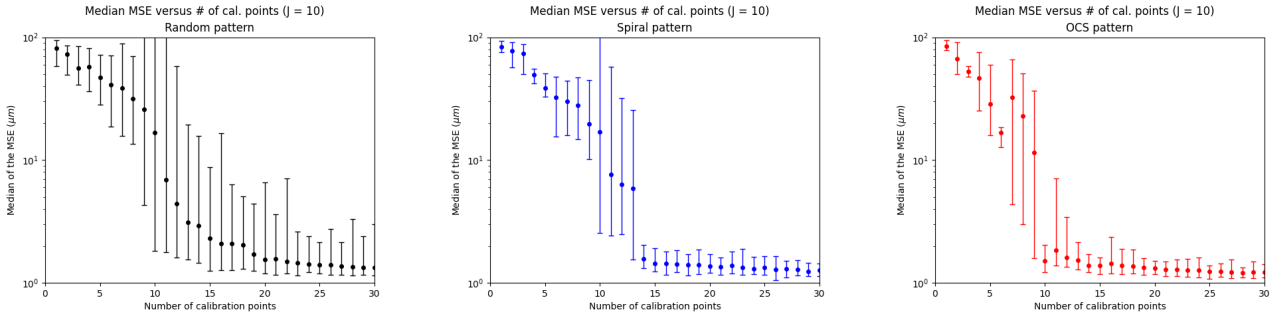


Figure 18: Median mean calibration error after fitting up to radial order $n = 3$.

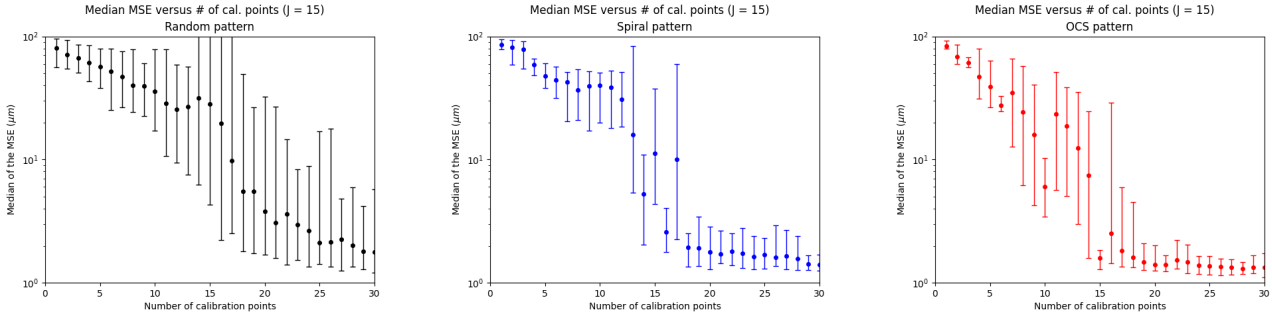


Figure 19: Median mean calibration error after fitting up to radial order $n = 4$.

From which it seems that that OCS exhibits a faster reduction of the mean error, closely followed by the spiral pattern (although with higher dispersion). However, there is still the possibility of tie for OCS and spiral patterns when the number of calibration points matches the number of complex coefficients of the pointing model.

One remarkable feature of OCS-based calibrations is the presence of valleys in the median error when the number of calibration points is triangular (3, 6, 10, 15...). This is expected: OCS patterns always have the number of points of a complete set of Zernike polynomials up to certain radial order (which is triangular), and the only way to obtain an intermediate number is by removing points of an existing complete pattern, affecting its symmetry and therefore its performance.

It is worth noting that the number of points required to perform a calibration that meets the $r < 10$ µm goal (order 10) is much smaller that the number of available dots in the GCU mask (order 10^2). Once a full opto-mechanical model is developed, this result may be used to optimise the design of the GCU mask in order to provide only those dots actually used for calibration.

4.3.1. Spatial distribution of the calibration residuals

In order to break the apparent tie between the OCS and spiral patterns, we carried a more detailed study of the residual. To that end, we obtained residual heatmaps for both the spiral and OCS pattern when $Q = J$ (i.e. when the number of calibration points matches the number of coefficients of the aberration model), with $J = 6$ and $J = 10$. In the $J = 6$ case, we observed that the lack of symmetry of the spiral pattern always causes one side of the relayed field to be undersampled, while OCS calibrations tend to sample the plane more homogeneously:

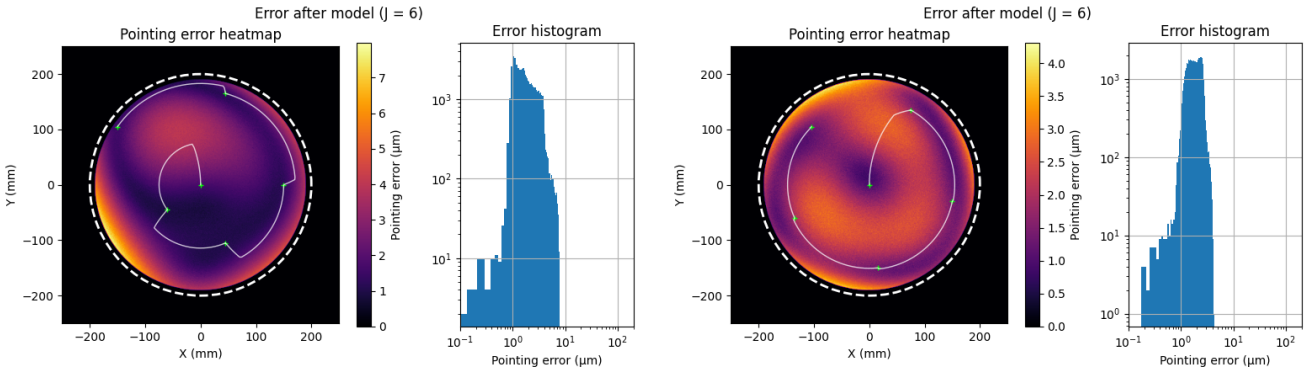


Figure 20: Calibration with spiral pattern (left) and OCS (right) up to radial order $N = 2$ ($J = 6$). The heatmap represents the Euclidean norm of the 2D residual after calibration.

This effect is exaggerated at higher orders, as the scarcity of sampling points in the edges of the field causes the aberration model to extrapolate in uninformed ways. In these regions, the pointing error reaches values comparable to those found in the uncorrected case:

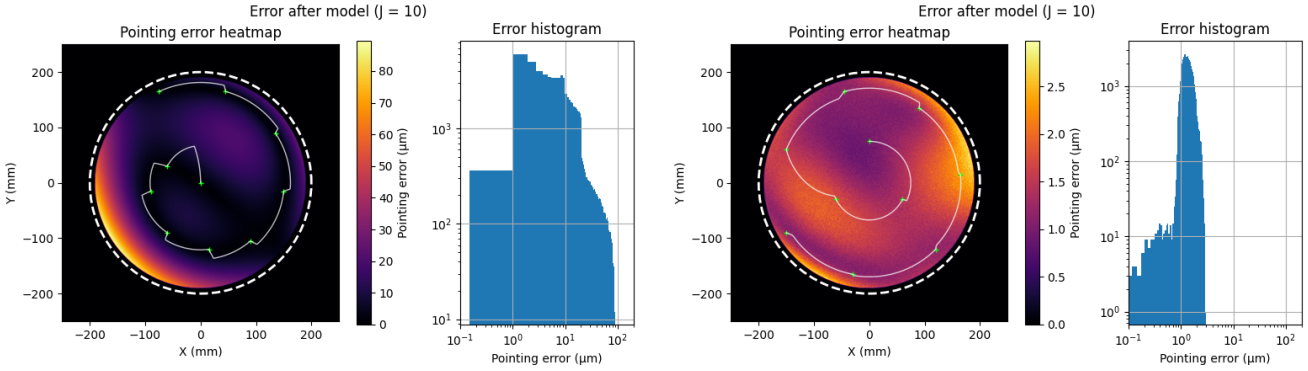


Figure 21: Calibration with spiral pattern (left) and OCS (right) up to radial order $N = 3$ ($J = 10$). The heatmap represents the Euclidean norm of the 2D residual after calibration. The spiral pattern reduces the error near the calibration points but increases it significantly near the undersampled region. On the other hand, the OCS pattern reduces the error almost everywhere, to the extent of making the non-systematic contributions of the error noticeable.

From these results, we conclude that OCS performs better than the spiral pattern when the number of calibration points equals to the number of coefficients of the pointing model. Additionally, when a model with radial order $N = 3$ is used, calibrations are far from the $r < 10$ µm goal due to the presence of highly undersampled regions. This renders OCS the only viable alternative. Nevertheless, the optimality of this result could still be challenged by error contributions not yet considered in the instrument model.

4.4. Sampling strategy

Next, we addressed the problem of finding the best calibration strategy, i.e. finding the path that traverses all calibration points in the shortest possible amount time. Since this problem is a particular instance of the Travelling Salesman Problem (TSP)[1] with a distance is given by the time the pick off arm takes to move from the departure configuration (θ_1, ϕ_1) to the destination configuration (θ_2, ϕ_2), it is NP-complete and the optimal solution implies testing all $Q!$ point permutations which, for the $Q = J = 10$ case, amount to more than 3.5×10^6.

Instead, we tested a heuristic based on finding the closest uncalibrated point (in L_∞ distance of its θ, ϕ coordinates) iteratively. In the $Q = 10$ case, we observed a speedup from 600 s to

350 s (which supposes a reduction of 40% in calibration time). The simplicity of the heuristic, along with its dramatic reduction of the calibration time justifies its usage when $Q \gg 10$.

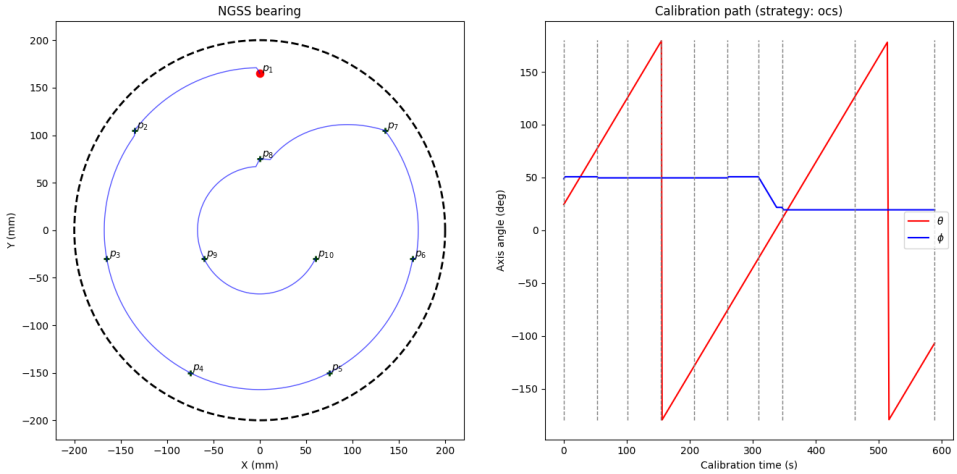


Figure 22: Time plot for a typical non-optimised OCS calibration with 10 calibration points.

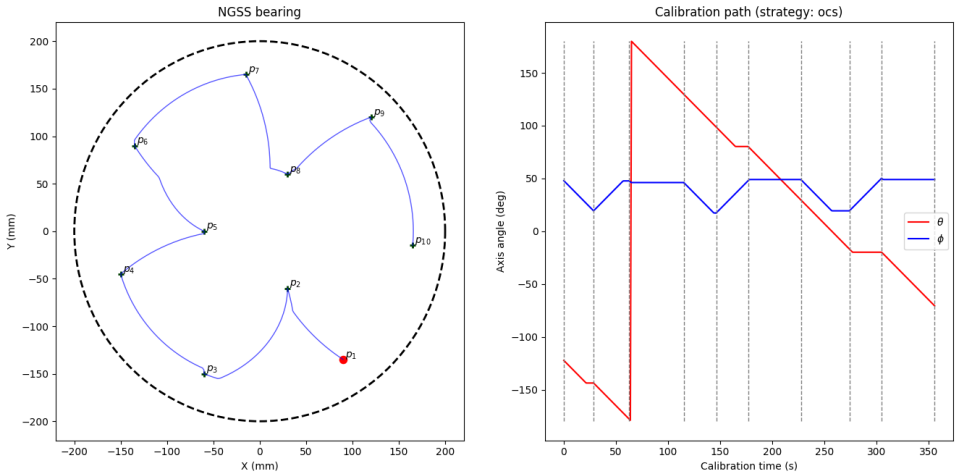


Figure 23: Time plot for a typical optimised OCS calibration with 10 calibration points.

5. Conclusions and future work

In this work, we addressed the problem of modelling the contributions to the pointing error as observed by HARMONI. Due to the inherent complexity of the instrument, we selected a subset of contributions that we considered to be the main drivers of this error. This resulted in a preliminary model whose formalisation foresaw future refinements. These refinements may take the form of better characterisation of the current contributions or the inclusion of contributions not yet considered in the model.

The model was put into practice by a set of Python packages and scripts that simulated different aspects of the instrument, including uncalibrated pointing errors, the performance of different calibration strategies and pointing model configurations and pointing model residuals.

These software components should be considered prototypes that aim to evolve to be eventually exploited by the observatory during operations. This evolution may be carried by other developers as well, and therefore the extensibility of the software was a key principle during its design phase.

Although many properties of the instrument components were not considered in this stage, the current software architecture facilitates the extension of the model as more details of the design are constrained. This did not prevented the simulator from producing results that can be applied in the future stages of development of the instrument:

- Assuming the FPRS model is realistic, we conclude that an aberration model with $N = 2$ (6 complex coefficients) fulfills the positioning accuracy of 10 µm. If an aberration model with $N = 3$ (10 complex coefficients) is fitted, the residual plummets below the amplitude of the mechanical instabilities. The choice between both alternatives should be the result of a trade-off between positioning accuracy and calibration time.

- We provided evidence for the superior performance of the OCS pattern over the traditional spiral pattern when the number of calibration points matches the number of complex coefficients of the aberration model, especially at higher radial orders. Nonetheless, this superiority is only significant when the number of coefficients of the pattern is a triangular number (i.e. when the pattern is complete).

- We also provided priors for the coefficients of the aberration model which, in a worst case scenario, will provide information on the orders of magnitude of the aberration as projected in each Zernike polynomial.

- Finally, we concluded that we can speed up the calibration process not only by reducing the number of calibration points to the number of coefficients of the aberration model, but also by reordering points in a way that does not involve testing all possible permutations of the calibration set.

Many simulations have been left for the future due to the lack of time and access to detailed specifications. In particular, the Secondary Guiding Simulator requires a realistic model of the LOB optics inside the POA, as well as the properties of the detector used for calibration. Once formalised and implemented, it will be used to characterise the POA as a measurement device and repeat the results of the Pointing Simulator.

Additionally, the current model of the GCU mask relies on fixed parameters and does not reflect is final design. The future refinements of the model must address this issue, as well as the possibility of performing simulations of the POA with other illumination sources than the GCU (i.e. natural guide stars).

Finally, as the number of points needed for calibration is much smaller than the number of dots in the GCU mask, optimisations of the latter should be explored. These optimisations may take the form of different GCU mask designs that reflect the distribution of points that are relevant from the perspective of the calibration process.

Bibliography

[1] Eugene L. Lawler, ed. *The Traveling salesman problem: a guided tour of combinatorial optimization*. Wiley-Interscience series in discrete mathematics. Chichester [West Sussex] ; New York: Wiley, 1985. ISBN: 9780471904137.

[2] Larry N. Thibos et al. "Standards for Reporting the Optical Aberrations of Eyes". en. In: *Vision Science and its Applications*. Santa Fe, New Mexico: OSA, 2000, S652–60. ISBN: 9781557526236. DOI: 10.1364/VSIA.2000.SuC1. URL: https://www.osapublishing.org/abstract.cfm?URI=VSIA-2000-SuC1 (visited on 08/26/2021).

[3] Rafael Navarro, Justo Arines, and Ricardo Rivera. "Direct and inverse discrete Zernike transform". en. In: *Optics Express* 17.26 (Dec. 2009), p. 24269. ISSN: 1094-4087. DOI: 10.1364/OE.17.024269. URL: https://www.osapublishing.org/oe/abstract.cfm?uri=oe-17-26-24269 (visited on 08/14/2021).

[4] ESO. *The E-ELT Construction Proposal*. https://www.eso.org/sci/facilities/eelt/docs/e-elt_executivesummary.pdf. Accessed: 2021-08-26. 2010.

[5] Rafael Navarro, Ricardo Rivera, and Justiniano Aporta. "Representation of wavefronts in free-form transmission pupils with Complex Zernike Polynomials". en. In: *Journal of Optometry* 4.2 (Apr. 2011), pp. 41–48. ISSN: 18884296. DOI: 10.1016/S1888-4296(11)70040-1. URL: https://linkinghub.elsevier.com/retrieve/pii/S1888429611700401 (visited on 08/14/2021).

[6] Princeton University Pres, ed. *Physics of the Interstellar and Intergalactic Medium*. Wiley, 2011, pp. 63–68. ISBN: 0691122148.

[7] D. Ramos-López et al. "Optimal sampling patterns for Zernike polynomials". en. In: *Applied Mathematics and Computation* 274 (Feb. 2016), pp. 247–257. ISSN: 00963003. DOI: 10.1016/j.amc.2015.11.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S009630031501468X (visited on 08/14/2021).

[8] Niranjan A. Thatte et al. "The E-ELT first light spectrograph HARMONI: capabilities and modes". In: *Ground-based and Airborne Instrumentation for Astronomy VI*. Ed. by Christopher J. Evans, Luc Simard, and Hideki Takami. Vol. 9908. International Society for Optics and Photonics. SPIE, 2016, pp. 595–605. URL: https://doi.org/10.1117/12.2230629.

[9] R. S. Biesheuvel et al. "Implementation and benchmarking of a crosstalk-free method for wavefront Zernike coefficients reconstruction using Shack-Hartmann sensor data". en. In: *OSA Continuum* 1.2 (Oct. 2018), p. 581. ISSN: 2578-7519. DOI: 10.1364/OSAC.1.000581. URL: https://www.osapublishing.org/abstract.cfm?URI=osac-1-2-581 (visited on 08/14/2021).

[10] Fraser Clarke. "HARMONI WCS Knowledge". Internal document.

[11] *Zemax website: OpticStudio*. https://www.zemax.com/pages/opticstudio. Accessed: 2021-08-26.

A. Determination of the POA configuration

In order to perform a measurement of a point in the relayed focal plane, the (x, y) coordinates of the point must be translated first into certain POA configuration (θ, ϕ). The problem of finding this configuration is degenerate: there are always at least two different POA configurations that theoretically fall into the same relayed focal plane point.

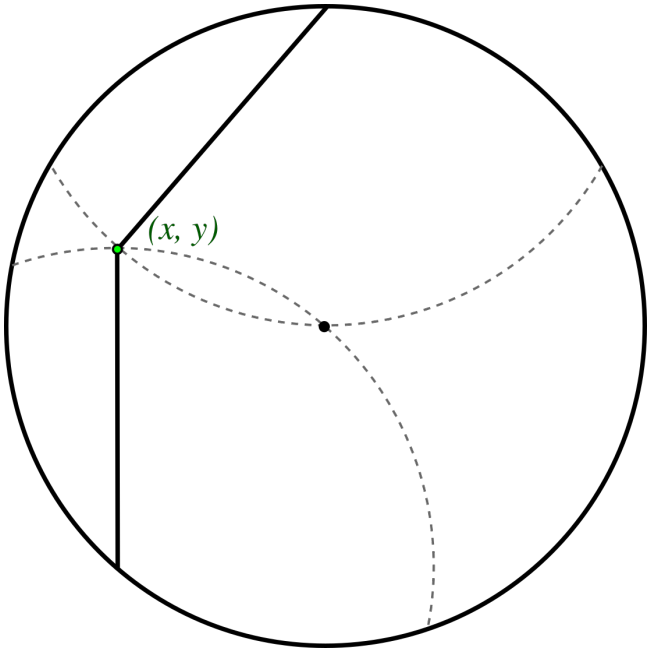


Figure 24: Degeneracy of the POA configuration problem. Two different configurations of the POA end up in the same point of the relayed focal plane.

While this degeneracy can be beneficial (it provides us with two alternatives to reach certain point in the relayed focal plane, potentially reducing the calibration time), the current iteration of the model breaks the degeneracy by accepting only positive secondary axis angles.

We find the POA angles (θ, ϕ) in two steps. In a first step, the desired (x, y) pair is converted to polar coordinates (ρ, α). We find ϕ by provisionally leaving θ set to 0, so that the head of the POA is at the same ρ as the desired point (figure 27).

In this configuration, the dependency of ρ^2 on ϕ can be written as:

$$\rho^2 = R^2 \sin^2\phi + R^2(1 - \cos\,\phi)^2$$

If we normalise ρ by R:

$$\frac{\rho^2}{R^2} = \sin^2\phi + 1 + \cos^2\phi - 2\cos\,\phi = 2(1 - \cos\,\phi)$$

we can find ϕ by solving the equation:

$$\cos\,\phi = 1 - \frac{\rho^2}{2R^2}$$

We can see that both ϕ and $-\phi$ are solutions to the equation. We break this degeneracy by arbitrarily choosing the positive solution.

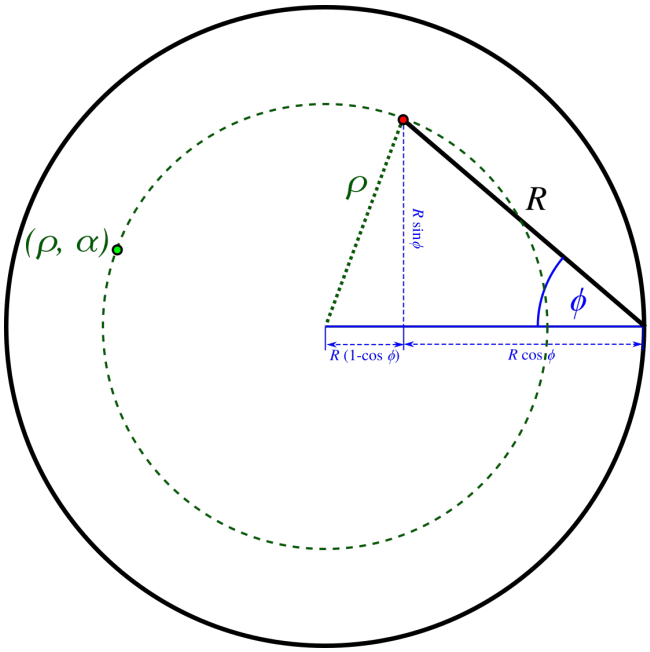


Figure 25: Calculation of ϕ.

In a second step, we find θ by taking into account that the previous adjustment of ϕ has placed the arm's head halfway the desired angle α by an amount β:

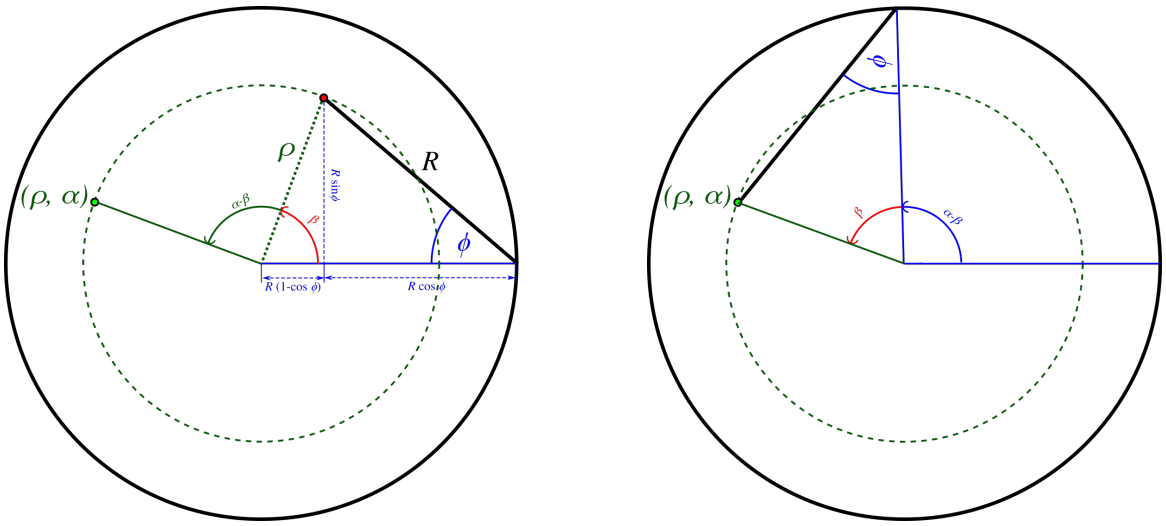


Figure 26: Calculation of θ from β.

where $\tan\,\beta$ is given by:

$$\tan\,\beta = \frac{\sin\,\phi}{1-\cos\,\phi}$$

The complete transform is therefore given by:

$$\rho = \sqrt{x^2+y^2} \qquad \phi = \arccos\left(1-\frac{\rho^2}{2R^2}\right)$$

$$\alpha = \arctan(y,x) \qquad \theta = \alpha - \arctan\left(\frac{\sin\,\phi}{1-\cos\,\phi}\right)$$

B. FPRS transform

The FPRS transform appears in the common part of the model coordinate transforms, and represents the aberrations introduced by the FPRS optics. This transform is constructed by interpolation from the results of a simulation performed by the lead optical engineer of the FPRS at UK ATC, using the commercial software OpticStudio[11]. These results take the form of a plain text file in which each line represents how much a ray is deviated from its departure position in the input focal plane as it traverses the optics of the FPRS.

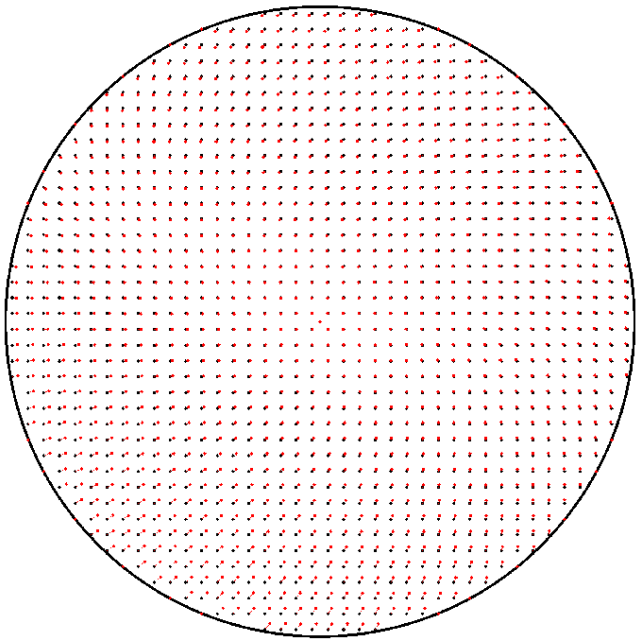


Figure 27: Aberration caused by the FPRS, exaggerated 25 times and simulated in a 400 × 400 mm grid of 10 mm-spaced points. True location of points are painted in black. Aberrated (displaced) points are painted in red.

B.1. Forward transform

The test points in this simulation were arranged in a fixed-step square grid ($\Delta x = \Delta y = h$) within the 200 mm radius of the technical field. In this grid, the $\boldsymbol{x} = (x, y)$ coordinates of each point could be calculated from two integer indices i, j as $x_i = x_0 + ih$ and $y_j = y_0 + jh$. This arrangement motivated the following bilinear interpolation of the pointing error between consecutive sampling points:

$$\boldsymbol{\varepsilon}(\boldsymbol{x}) \approx (1-\beta)\left[(1-\alpha)\boldsymbol{x}_{ij} + \alpha\boldsymbol{x}_{i+1,j}\right] + \beta\left[(1-\alpha)\boldsymbol{x}_{i,j+1} + \alpha\boldsymbol{x}_{i+1,j+1}\right] \tag{B.1.1}$$

where $\alpha = \frac{x-x_i}{\Delta x}$, $\beta = \frac{y-y_i}{\Delta y}$ and $\boldsymbol{x}_{ij}$ is generally easy to find. From this interpolation, one can construct the forward FPRS transform by addition:

$$F[\boldsymbol{x}] = \boldsymbol{x} + \boldsymbol{\varepsilon}(\boldsymbol{x}) \tag{B.1.2}$$

B.2. Backward transform

While the interpolated forward FPRS transform can be trivially evaluated, this is not true for the backward transform $F^{-1}[\boldsymbol{x}]$. In the forward transform, departure points are regularly spaced and it is easy to find the vertices of the square in which the 4 surrounding sampling

points are located. In the backward transform, this square is distorted into a trapezoid, and the location of its vertices cannot be known efficiently unless barely-vectorisable operations are used.

In order to prevent a performance bottleneck in the code, we will settle for an approximation based on the Taylor expansion of the backward transform. This approximation will let us obtain a measure of the backward pointing error in the departure points.

Let us assume that $F^{-1}[\boldsymbol{x}]$ can be written as:

$$F^{-1}[\boldsymbol{x}] = \boldsymbol{x} + \boldsymbol{\varepsilon}_b(\boldsymbol{x}) \tag{B.2.1}$$

where $\boldsymbol{\varepsilon}_b$ is the backward pointing error and it is expensive to calculate, and let $\boldsymbol{\varepsilon}_{ij} = \boldsymbol{\varepsilon}(\boldsymbol{x}_{ij})$. If we evaluate the backward pointing error in the displaced sampling point $\boldsymbol{x}_{ij} + \boldsymbol{\varepsilon}_{ij}$, we obtain:

$$\boldsymbol{\varepsilon}_b(\boldsymbol{x}_{ij} + \boldsymbol{\varepsilon}_{ij}) = \boldsymbol{\varepsilon}_b[F(\boldsymbol{x}_{ij})] = F^{-1} \circ F(\boldsymbol{x}_{ij}) - F(\boldsymbol{x}_{ij}) = \boldsymbol{x}_{ij} - F(\boldsymbol{x}_{ij}) = -\boldsymbol{\varepsilon}_{ij} \tag{B.2.2}$$

which tells us that the departure point is at a vector distance $\boldsymbol{\varepsilon}_{ij}$ in the opposite direction. Now, since $\boldsymbol{\varepsilon} \ll h$ (this was confirmed by inspection of the simulation results), we can approximate $\boldsymbol{\varepsilon}_b(\boldsymbol{x}_{ij})$ by its Taylor expansion around the displaced point $\tilde{\boldsymbol{x}}_{ij} = \boldsymbol{x}_{ij} + \boldsymbol{\varepsilon}_{ij}$ up to order 1:

$$\boldsymbol{\varepsilon}_b(\boldsymbol{x}) \approx \boldsymbol{\varepsilon}_b(\tilde{\boldsymbol{x}}_{ij}) + (J_{F^{-1}} - \mathbb{I})(\tilde{\boldsymbol{x}}_{ij})(\boldsymbol{x} - \tilde{\boldsymbol{x}}_{ij}) \tag{B.2.3}$$

with J_{T_b} the Jacobian matrix of the backward transform and $\mathbb{I}$ a 2×2 identity matrix. We showed in equation $B.2.2$ that $\boldsymbol{\varepsilon}_b(\boldsymbol{x}_{ij} + \boldsymbol{\varepsilon}_{ij}) = -\boldsymbol{\varepsilon}_{ij}$. If we evaluate $\boldsymbol{\varepsilon}_b(\boldsymbol{x})$ in the departure points instead of the displaced points, we get:

$$\boldsymbol{\varepsilon}_b(\boldsymbol{x}_{ij}) \approx -\boldsymbol{\varepsilon}_{ij} - (J_{F^{-1}} - \mathbb{I})(\tilde{\boldsymbol{x}}_{ij})\boldsymbol{\varepsilon}_{ij} \tag{B.2.4}$$

The order 1 term provides information on how the displacement evolves in each perpendicular direction. Since any 4 contiguous $\boldsymbol{\varepsilon}_{ij}$ are expected to be of the same order of magnitude (resulting in a Jacobian close to the identity) and much smaller than the sampling step, $(J_{F^{-1}} - \mathbb{I})(\tilde{\boldsymbol{x}}_{ij})\boldsymbol{\varepsilon}_{ij}$ is negligible compared to $\boldsymbol{\varepsilon}_{ij}$. We can therefore approximate $\boldsymbol{\varepsilon}_b$ in the departure sampling points safely as:

$$\boldsymbol{\varepsilon}_b(\boldsymbol{x}_{ij}) \approx -\boldsymbol{\varepsilon}_{ij} \tag{B.2.5}$$

which can be used to build another bilinear interpolator as illustrated in the forward case.

C. Software details

C.1. Configuration file

Model configuration is stored in a file named `harmoni.ini`, and the code will attempt to load it from the working directory in which it is executed. The configuration file contains model parameters and references to other files required by the simulation.

C.1.1. Format

`harmoni.ini` consists of lines of key-value pairs grouped into sections. Section names precede the set of key-value pairs inside the section and are enclosed in square brackets. Key names and values are separated by an equals sign (`=`), with the former admitting alphanumerical characters, dots, dashes and underscores. The file accepts single-line comments prefixed by a sharp symbol (`#`).

Values admitted by `harmoni.ini` are all Python 3 literals plus other key names (in that case, the value is taken as that of the referenced key). If the key refers to a random variable, a string (enclosed by double quotes) with the following format must be provided:

```
"value +/- dispersion units (distribution)"
```

In which `value` is the nominal value of the variable (usually the mean), `dispersion` is certain spread measure of the variable distribution, `units` is any unit string supported by the `Pint` library and `distribution` is a string that describes the statistical distribution followed by the variable.

The meaning of `dispersion` depends on the `distribution` it refers to. For instance, `"5 +/- 0.1 cm (normal)"` describes a Gaussian random variable with mean 5 and FWHM 0.2, expressed in centimetres. The following table summarises both the supported distributions and the interpretation of their dispersion measure:

Table 6: Statistical distributions accepted in `harmoni.ini`. In this table, μ refers to `value` and θ to `dispersion`.

Name	Dispersion measure	Equivalent distribution
`normal`, `gauss` or `gaussian`	1/2 of FWHM	$N(\mu, 32\theta^2 \ln 2)$
`uniform`, `flat` or `pp`	1/2 of peak-to-peak value	$U(\mu - \theta, \mu + \theta)$
`diracdelta`	N/A	$\delta(x - \mu)$

C.1.2. Configuration keys

The following table contains all the configuration keys, sections and data types that may appear in `harmoni.ini`. If any of these keys is not provided, it will be treated as it was declared with its default value.

Variable type may be R (real value with units), D (statistical distribution with dimensions), I (integer) or S (printable string). Random variables are indicated by their distribution between parentheses: N for normal distributions, pp for uniform distributions and δ for distributions with no uncertainty (fixed).

Table 7: Configuration keys accepted by `harmoni.ini`. SIFU stands for Système International Flux Units (W $Hz^{-1}m^{-2}$).

Section	Key	Symbol	Type	Default	Reference
`fprs`	`desc_file`	N/A	S (file path)	`"FPRS_distortion_map.txt"`	N/A
`gcu`	`point.separation`	h_p	R (metres)	15×10^{-3}	Table 1
`gcu`	`point.diameter`	D_p	R (metres)	150×10^{-6}	Table 1
`gcu`	`point.flux`	F_p	R (SIFU)	10^{-3}	Table 1
`gcu`	`mask.x0`	x_0	R (metres)	0	Table 1
`gcu`	`mask.y0`	y_0	R (metres)	0	Table 1
`gcu`	`mask.diameter`	D_M	R (metres)	0.4	Table 1
`gcu_alignment`	`x0`	Δx_G	D (length)	0 m (δ)	Table 2
`gcu_alignment`	`y0`	Δy_G	D (length)	0 m (δ)	Table 2
`irw`	`angle_bias`	$\Delta\omega_{IRW}$	D (angle)	0 rad (δ)	Table 2
`ngss_alignment`	`x0`	Δx_N	D (length)	0 m (δ)	Table 2
`ngss_alignment`	`y0`	Δy_N	D (length)	0 m (δ)	Table 2
`poa`	`encoder[theta].qerr`	q_θ	D (no dim.)	0.5 ± 0.5 (pp)	Table 3
`poa`	`encoder[theta].bits`	B_θ	I	11	Table 3
`poa`	`encoder[theta].err`	$\Delta\theta$	D (angle)	$0 \pm 1°$ (pp)	Table 3
`poa`	`encoder[theta].speed`	ω_θ	D (ang. freq.)	$1°/s$ (pp)	Table 4
`poa`	`encoder[phi].qerr`	q_ϕ	D (no dim.)	0.5 ± 0.5 (pp)	Table 3
`poa`	`encoder[phi].bits`	B_ϕ	I	11	Table 3
`poa`	`encoder[phi].err`	$\Delta\phi$	D (angle)	$0 \pm 1°$ (pp)	Table 3
`poa`	`encoder[phi].speed`	ω_ϕ	D (ang. freq.)	$1°/s$ (pp)	Table 4
`poa`	`radius`	e_R	D (length)	0.2 ± 10^{-6} m (pp)	Table 3
`poa`	`arm_instability`	ΔR	D (length)	0 ± 1 µm (N)	Table 3
`poa`	`position_error`	$\Delta\rho$	D (length)	0 m (δ)	Table 3

C.2. Usage

The entry point of this project consists in two executable command-line scripts that exploit the features implemented inside the `harmoni_pm` package. `pointingsim.py` performs different simulations related to the calibration of the pointing error. `sgsim.py` is the secondary guiding simulator, and attempts to characterise the behavior of the POA as a measurement device.

C.2.1. pointingsim.py

The simulations executed by `pointingsim.py` are organised in test types, and simulation products can be produced in form of console output, data files and graphs. The command syntax is as follows:

```
$ ./pointingsim.py -t TEST_TYPE [OPTIONS]
```

where `TEST_TYPE` specifies the simulation to run, which can be one of the following:

Table 8: Test types accepted by `pointingsim.py`.

Test type	Description
`errormap`	Simulates the uncorrected pointing error in the technical field in the form of a heatmap.
`prior`	Calculates histograms of the pointing model coefficients based on manufacturing tolerances.
`calplot`	Computes the median pointing error curves for different calibration strategies and point counts.
`calmap`	Computes the heatmap of the residual after applying a pointing model obtained from a simulated calibration process.
`caltime`	Computes the calibration time plot for a certain calibration strategy.
`caldist`	Computes histograms of the calibration time for different realisations of the same calibration strategy.

Tests types are exclusive, and `pointingsim.py` can only run one of the above at a time. On the other hand, `[OPTIONS]` is a list of switches that can be used to tune the execution of a particular test. From the simulation perspective, the most relevant options are:

Table 9: Optional arguments accepted by `pointingsim.py`.

Test type	Description
`-P`	Plot simulation results using Matplotlib.
`-m`	Overlay the GCU mask dots used for calibration in heatmap plots.
`-C Q`	Set the number of points used for calibration to Q (default: 553)
`-N N`	Sets the number of Monte Carlo simulations to N (default: 1000).
`-J J`	Set the number of coefficients of the pointing model to J. For realistic models, J should be a triangular number (default: 3).
`-S PAT`	Set the calibration pattern to `PAT` (`random`, `spiral` or `ocs`). Default is `random`.
`-s SECT.KEY=VAL`	Override the configuration key `KEY` in section `SECT` with the value `VAL`.

C.2.2. sgsim.py

The script `sgsim.py` is the prototype simulator for the POA detector, and should be understood as the foundation of a full-fledged POA simulator that will be used to obtain the likelihood function of the point measurement process. As the details of the detector were not included in the model, this script is a simple detector simulator with free parameters like pixel geometry and magnification ratio. Future iterations of the model will permit the augmentation of the features of this script, including the point fitting logic and Monte Carlo simulations of GCU mask dot measurements.

Figure 28: Simulation results of the POA detector with different exposition times. The Poisson nature of the detection process is evident at low exposition times.

The only data product of this script are the simulated images. The command syntax is as follows:

```
$ ./sgsim.py [OPTIONS]
```

where `[OPTIONS]` are used to set the parameters of the simulated CCD. The options accepted by `sgsim.py` are summarised in the following table:

Table 10: Optional arguments accepted by `sgsim.py`.

Test type	Description
`-W WIDTH`	Sets the detector width (in pixels) to `WIDTH`.
`-H HEIGHT`	Sets the detector height (in pixels) to `HEIGHT`.
`--px-width PXW`	Sets the pixel width to `PXW`. Default: 14 µm.
`--px-height PXH`	Sets the pixel width to `PXH`. Default: 14 µm.
`-f MAG`	Sets the magnification ratio to `MAG`. Default: 1.
`-S SRC`	Sets the image source to `SRC`. Default is `gcu`.
`-t THETA`	Sets the primary axis angle to `THETA`. Default is 0°.
`-p PHI`	Sets the secondary axis angle to `PHI`. Default is 0°.
`-O OVR`	Sets the oversampling value per pixel dimension. Default is 8.
`-e TIME`	Sets the exposition time. Default is 0, meaning that the image represents the spatial distribution of the source's flux density.

D. SGSim theory and implementation

The `SGSim` component refers to the executable script `sgsim.py`, which contains the prototype implementation of a POA detector simulator. As such, `SGSim` is not yet ready to produce usable results. Nonetheless, the core of the integration logic has already been implemented, and future developers can leverage the existing work to complete the tool. This section provides the implementation details that must be known prior to any modification of the code.

D.1. Integration theory

We refer by integration to the calculation of the number of photons that arrive to each pixel in a given period of time, assuming cartain radiative source at the other end of the optical system. This calculation is implemented by the `ImageSampler` class (which, at the same time, inherits from `PlaneSampler`) and is based on radiative transfer[6].

The original assumption considered a general radiative source at infinity, fully described by its radiative intensity $I(\alpha, \delta)$ (W m^{-2}sr^{-1}Hz^{-1} in SI units). Later iterations showed that this model was overly descriptive, and reduced the required knowledge of the source to its radiative flux field $F(x, y)$ over certain focal surface (W m^{-2}Hz^{-1}). Although I is still needed to relate the surface brightness in the sky with a flux in the focal plane, the following reasoning will assume that I can be converted to F by means of a change of coordinates.

D.1.1. Radiative transfer

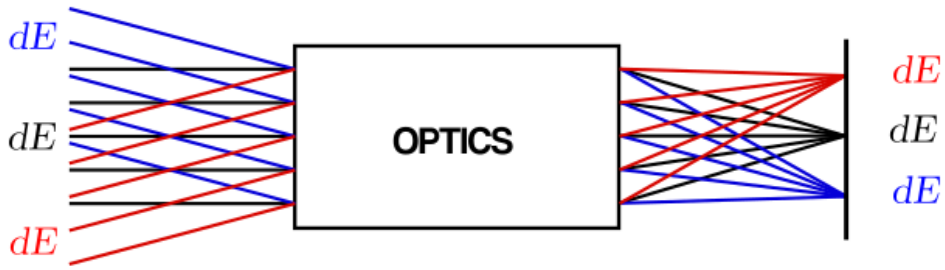


Figure 29: Modelisation of the instrument from the perspective of radiative transfer. Energy from radiative sources at infinity –described by their surface brightness I– is focused on points of the focal plane.

Let dE_t be the energy differential corresponding to the amount of radiative energy coming from a region in the sky of radiative intensity I and angular size $d\Omega$ entering an aperture of area A in a time dt and a small range of frequencues $d\nu$. This quantity can be written as:

$$dE_t = Id\Omega A\cos\,\theta d\nu dt \tag{D.1.1}$$

Since the field is relatively small, $\cos\,\theta \approx 1$ for all the rays that fall in the detector. On the other hand, the amount of energy dE_d arriving to a point in the detector at the instrument's focal plane will be a fraction of the total energy that entered the telescope due to opacities, obstructions and aperture stops. This dependence can be written in terms of a dimensionless efficiency factor β:

$$dE_d = \beta dE_t \tag{D.1.2}$$

Let F the flux corresponding to the region $d\Omega$ that is focused in a small area dS in the focal plane. The energy differential in the detector will be:

$$dE_d = FdSd\nu dt \tag{D.1.3}$$

which allows us to deduce the following expression for the flux in the focal plane:

$$F = \beta IA\frac{d\Omega}{dS} \tag{D.1.4}$$

from which we see that the flux is both proportional to the surface brightness and $\frac{d\Omega}{dS}$. The latter is actually the Jacobian determinant of the coordinate transform between angular coordinates in the sky and spatial coordinates in the focal plane. If we consider that this coordinate transform is performed in two steps (i.e. light from a small patch $d\Omega$ in the sky travels to a small area in

the input focal plane dS_{fp}, and from there to a small area in the detector plane dS_{dp}), equation D.1.4 can be written as:

$$F = \beta I A \frac{d\Omega}{dS_{dp}} = \beta_{tel}\beta_{opt} I A \frac{d\Omega}{dS_{fp}} \frac{dS_{fp}}{dS_{dp}} = \beta_{opt} F_{fp} \frac{dS_{fp}}{dS_{dp}} \tag{D.1.5}$$

in which we decomposed the efficiency factor in a telescope contribution β_{tel} and an instrument contribution β_{opt} such that $\beta = \beta_{opt}\beta_{tel}$. Under this formulation, $\beta_{tel} I A \frac{d\Omega}{dS_{fp}}$ can be identified with the radiative flux in the input focal plane (F_{fp}) after its processing by the telescope optics, and $\frac{dS_{fp}}{dS_{dp}}$ with the Jacobian determinant of the coordinate transform between the input and output focal planes. This determinant is dimensionless as it refers to a transform between spatial coordinates, and equals to the inverse of the square of the magnification ratio in the aberration-free case. In the particular case of the FPRS transform, it should remain close to unity.

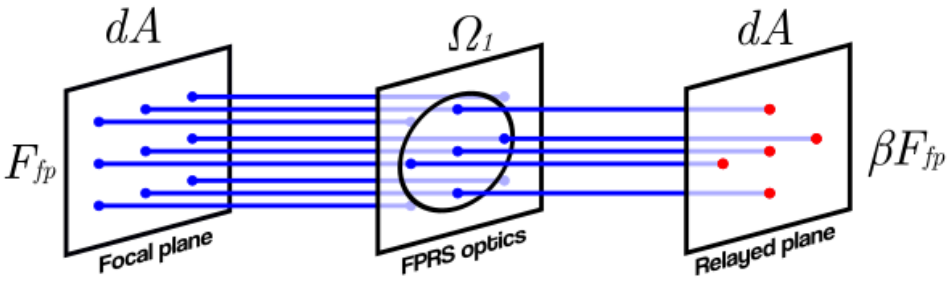


Figure 30: Simplified formulation applied to the 1:1 case of the FPRS optics. In this case, and destination surfaces have the same area, and only the efficiency factor affects the radiative flux.

F_{fp} may not only represent the radiative flux of the focused image in the input focal plane, but also the radiative flux field of the GCU mask during calibration. If an image of the sky is pre-processed in a way that only its flux is considered, we can disregard intensity fields completely: this is the approach adopted by the current implementation.

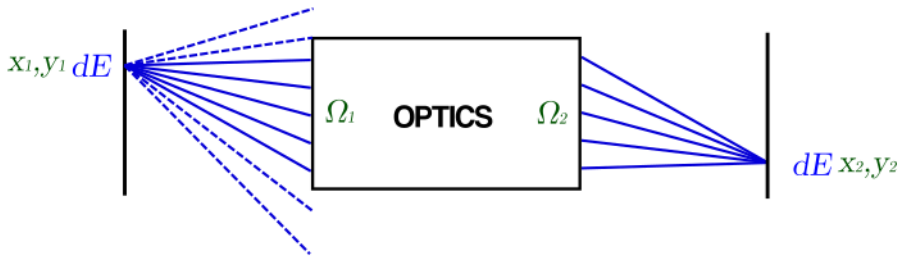


Figure 31: Radiative sources are characterised by their focused flux in the input focal plane.

Finally, if we know the quantum efficiency curve of the detector $Q(\nu)$ and the pixel area A_p, we can calculate the electron rate per pixel (in s^{-1}) as:

$$R = A_p \int F Q(\nu) \frac{d\nu}{h\nu} \tag{D.1.6}$$

from which we implicitly assumed that the detector Nyquist-samples the image plane. The true number of electrons generated in the integration time Δt can be modelled as a Poisson-distributed random variable with a rate λ given by:

$$\lambda = R\Delta t \tag{D.1.7}$$

D.1.2. Sky flux and telescope aberrations

In the previous point, we identified $\beta_{tel} I A \frac{d\Omega}{dS_{fp}}$ with the flux corresponding to sky sources. The factor $\frac{d\Omega}{dS_{fp}}$ is the Jacobian determinant of the telescope aberrations and, unlike $\frac{dS_{fp}}{dS_{dp}}$, has units of inverses of surface area. Since `SGSim` should also be able to perform simulations with sky sources, a modelisation of telescope aberrations is needed.

Let $T_b(\boldsymbol{x}) = [\phi(\boldsymbol{x}), \psi(\boldsymbol{x})]$ be the compound backward transform of all the optical elements in the optical path. In this expression $\boldsymbol{x} = x\hat{e}_x + y\hat{e}_y$ is the position vector of a point over the focal plane and $\phi(\boldsymbol{x})$ and $\psi(\boldsymbol{x})$ are two angular coordinates of the corresponding point in the sky with respect to the telescope pointing and centered in the equator. In this system ϕ is the longitude and ψ its latitude. The orthonormal basis $\hat{e}_x, \hat{e}_y$ can be assumed to be conveniently aligned to the horizontal and vertical directions of the input focal plane of the instrument.

The solid angle differential $d\Omega = \cos\psi d\phi d\psi$ is related to the area differential $dS = dxdy$ in the input focal plane as:

$$\cos\psi d\phi d\psi = \det J_b dxdy \tag{D.1.8}$$

Where J_b is the Jacobian matrix of the backward transform:

$$J_b = \begin{pmatrix} \frac{\partial\phi}{\partial x} & \frac{\partial\phi}{\partial y} \\ \frac{\partial\psi}{\partial x} & \frac{\partial\psi}{\partial y} \end{pmatrix} \tag{D.1.9}$$

In the paraxial approximation, $\cos\psi \approx 1$. J_b allows a polar decomposition of the form AR, where A is a symmetric matrix and R a rotation matrix that can be related to the field orientation.

In the aberration-free case, we can choose a system of sky coordinates such that ϕ depends only on x and ψ depends only on y. This implies that $\frac{\partial\phi}{\partial y} = \frac{\partial\psi}{\partial x} = 0$ and A will be simplified down to a diagonal matrix:

$$\frac{d\Omega}{dS_{fp}} = \det AR = \det \mathrm{A} = \frac{\partial\phi}{\partial x}\frac{\partial\psi}{\partial y} = \frac{1}{f^2} \tag{D.1.10}$$

where f is the focal distance of the telescope. As radial aberrations are introduced, A will lose diagonality and the determinant will change. This relationship suggests a way to estimate focal distances from coordinate transforms and measure the degree of aberration in the optics:

$$E(\boldsymbol{x}) = ||1 - f^2 \det \tilde{J}_b(\boldsymbol{x})|| \tag{D.1.11}$$

with E a dimensionless quality figure that measures how much the aberrations are affecting the image in a given point $\boldsymbol{x}$ in the focal plane, f the theoretical focal distance and $\tilde{J}_b(\boldsymbol{x})$ a numerical calculation of the Jacobian of the transform in the surroundings of $\boldsymbol{x}$.

D.1.3. Oversampling

Coordinate transforms must be calculated at least once for every pixel in the detector. If the detector Nyquist-samples the image plane, it should be enough to measure the flux at a fixed relative offset in every pixel. However, sub-Nyquist structure may exist in the image plane and oversampling would be needed to prevent aliasing / Moiré.

For a given oversampling value M, `ImageSampler` will measure the flux of the image plane in M^2 points inside the pixel surface and calculate the average flux of all of them. The offset of the sampling point $0 \leq i, j < M$ from the bottom left corner of a pixel of width Δx and height Δy will be:

$$\boldsymbol{p}(i,j) = \left(i + \frac{1}{2}\right)\frac{\Delta x}{M}\hat{e}_x + \left(j + \frac{1}{2}\right)\frac{\Delta y}{M}\hat{e}_y \tag{D.1.12}$$

D.2. Implementation

From the implementation perspective, flux calculation is decoupled from the generation of the set of pixels (subpixels) where the flux is evaluated. The former is implemented in the `ImageSampler` class (`harmoni_pm/imagesampler/image_sampler.py`) while the latter is implemented in the `PlaneSampler` class (`harmoni_pm/transform/plane_sampler.py`). `ImageSampler` also takes the sampled flux in each pixel and draws photon counts from a Poisson distribution (method `ImageSampler.save_to_file`). `sgsim.py` simply constructs an `ImageSampler` object according to parameters passed in the command line and spawns the simulation.

D.2.1. Coordinate generation

Generation of subpixel coordinates is vectorised by `PlaneSampler` and performed in several steps:

1. Generation of the matrix of row indices. A square matrix $r_{ij} = i$ is created as the tensor product of a `numpy.linspace(0, M - 1, M)` by a column vector of ones of the same size.
2. Generation of the x offsets. The vector $x_k = [\mathfrak{f}(r_{ij})_k + \frac{1}{2}]\delta x$ is created, with $\mathfrak{f}$ the `flatten` operator (which returns a vector of M^2 components such that $\mathfrak{f}(r_{ij})_k = r_{k \bmod M, \lfloor k/M \rfloor}$ and $\delta x = \Delta x / M$).
3. Generation of the y offsets. The vector $y_k = [\mathfrak{f}(r^T_{ij})_k + \frac{1}{2}]\delta y$ is created, with $\delta y = \Delta y / M$.
4. Generation of the coordinate offset vector. The matrix $\mathtt{xy}_{k2}$ is created, such that $\mathtt{xy}_{k0} = x_k$ and $\mathtt{xy}_{k1} = y_k$.

The resulting `xy` vector is then added to the coordinate of the bottom-left corner of each pixel to obtain the full list of oversampling points.

D.2.2. Slicing and parallelisation

The integration of the light arriving at $h \times w$ pixels with an oversampling of M requires the calculation of $h \times w \times M^2$ coordinate transforms. In addition to the computational cost of this operation, the memory allocation required by the vectorisation of these operations may be too big to fit in the computer's memory. These two problems are addressed by slicing and parallelisation.

Slicing refers to the process of selecting subsets of pixels in the detector from which integration operations are vectorised in `numpy`. The maximum size of a slice is that of a contiguous square of side `HARMONI_PLAME_SAMPLER_SLICE_SIZE` pixels, currently 128. The actual size of the slice may be smaller near the right and upper edges of the detector. The total number of coordinate transforms to perform can be calculated by multiplying the number of pixels in a slice K by the square of the oversampling M^2.

Parallelisation refers to the ability of the software to perform its calculations concurrently in different execution threads. This is achieved by vectorising the generation of coordinates in two stages. In a first stage (implemented in `PlaneSampler._process_slice`), matrices of row indices r_{ij} and column indices c_{ij} are created using the same strategy as described in the oversampling implementation, and flattened in order to obtain a $\mathtt{ij}_{K\times 2}$ matrix of K rows in which each row represents the row and column indices of the pixels to integrate.

In a second stage, the full coordinate list –including subsampling– is generated from an initial pixel list $\mathtt{ij}_{K\times 2}$. This stage is split again into the following steps:

1. Tiling of the oversampling offset matrix. A matrix `p_xy` consisting of concatenating `xy` K times is generated, producing an oversampling offset matrix of KM^2 rows.

2. Repetition of the pixel index matrix. Each individual row of the index matrix $\texttt{ij}_{K\times 2}$ is repeated contiguously M^2 times, growing the matrix up to KM^2 rows ($\texttt{ij}_{KM^2\times 2}$).

3. Generation of the sampling point matrix. The sampling coordinates are generated by adding the following matrix to `p_xy`:

$$\Delta\texttt{p\_xy}_{KM^2\times 2} = \texttt{ij}_{KM^2\times 2}\begin{pmatrix}\Delta x & 0\\ 0 & \Delta y\end{pmatrix} + \mathbb{1}_{KM^2\times 2}\boldsymbol{x}_0 \tag{D.2.1}$$

   where $\boldsymbol{x}_0$ is a position row vector that encodes the physical displacement of the detector.

The full coordinate list is then passed to the virtual method `PlaneSampler._process_region`, implemented by `ImageSampler`.

D.2.3. Evaluation

The evaluation of the sampled radiative flux field involves a data reduction due to oversampling, as the resulting flux vector must match the original pixel index matrix passed to `_process_region`. The reduced flux vector is evaluated by averaging the values of the flux in the subpixels belonging to the same pixel:

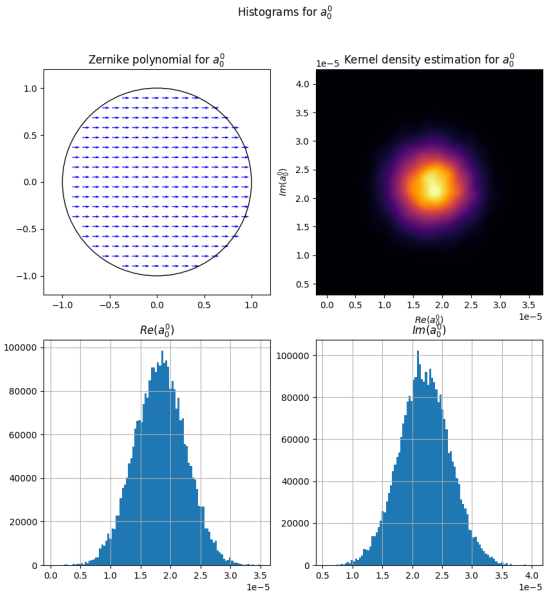
$$\bar{F}_k = \frac{1}{M^2}\sum_{i=0}^{M-1} F[T_b(\texttt{p\_xy}_{kM+i,0}, \texttt{p\_xy}_{kM+i,1})] \tag{D.2.2}$$

where T_b is the backward transform, $\bar{F}_k$ the reduced flux vector and F the flux function of the image plane. From this flux, one can obtain the Poisson rate λ (equations D.1.6 and D.1.7) and simulate per-pixel photon counts.

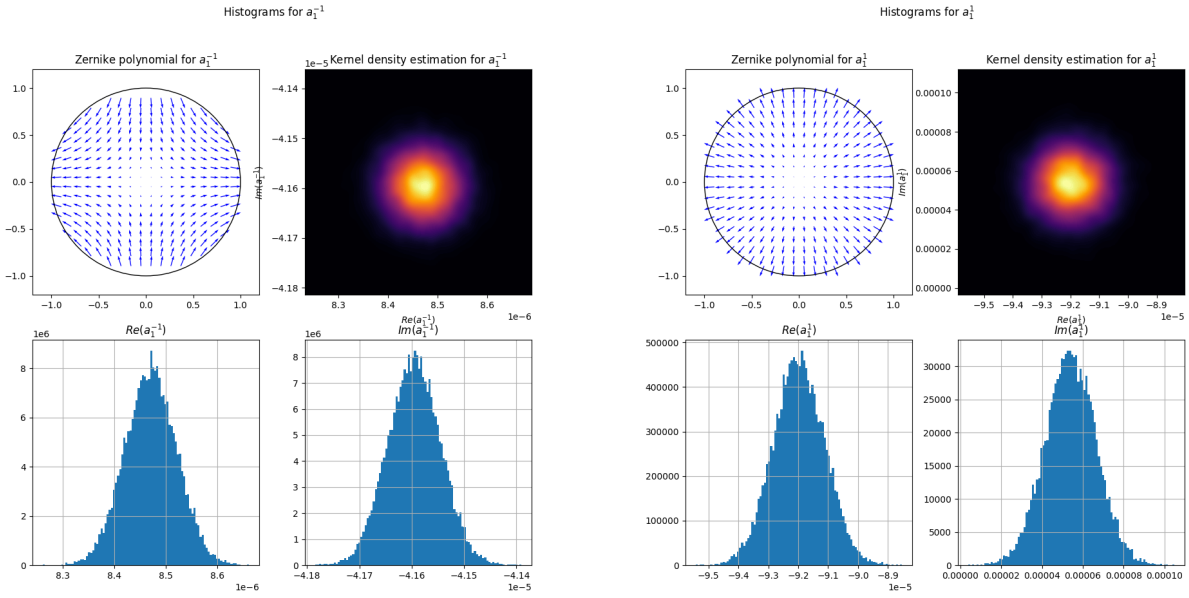
E. Priors of the aberration model

The following subsections detail the resulting prior histograms of the coefficients of the aberration model up to radial order $n = 3$. Scatter plot matrices are presented as well (E.5), providing evidence for the statistical independence between coefficients.

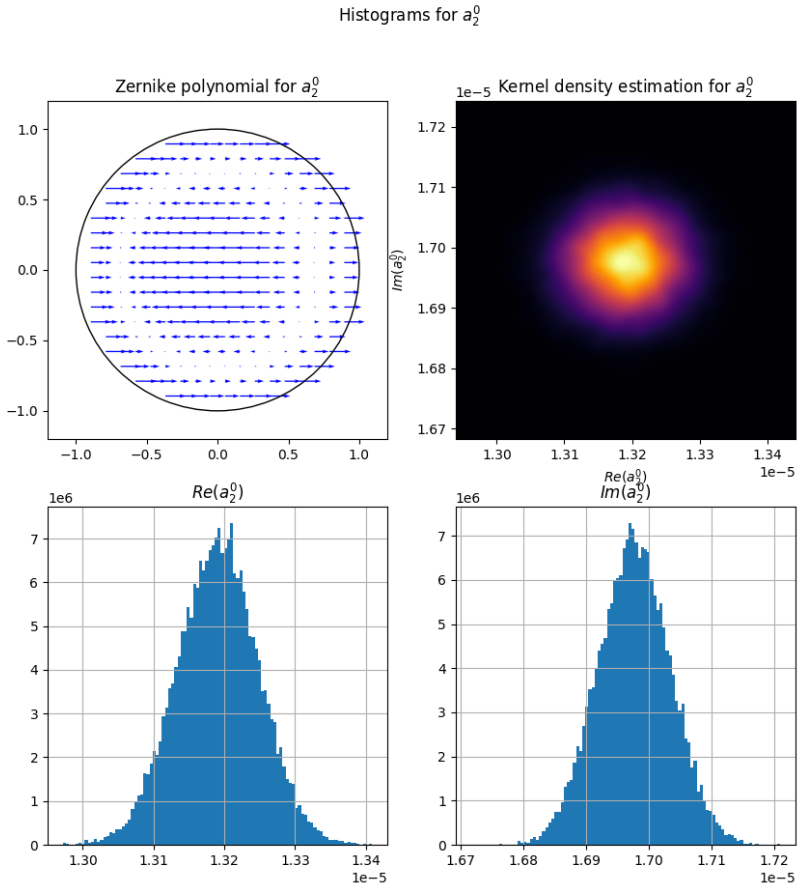
E.1. $n = 0$

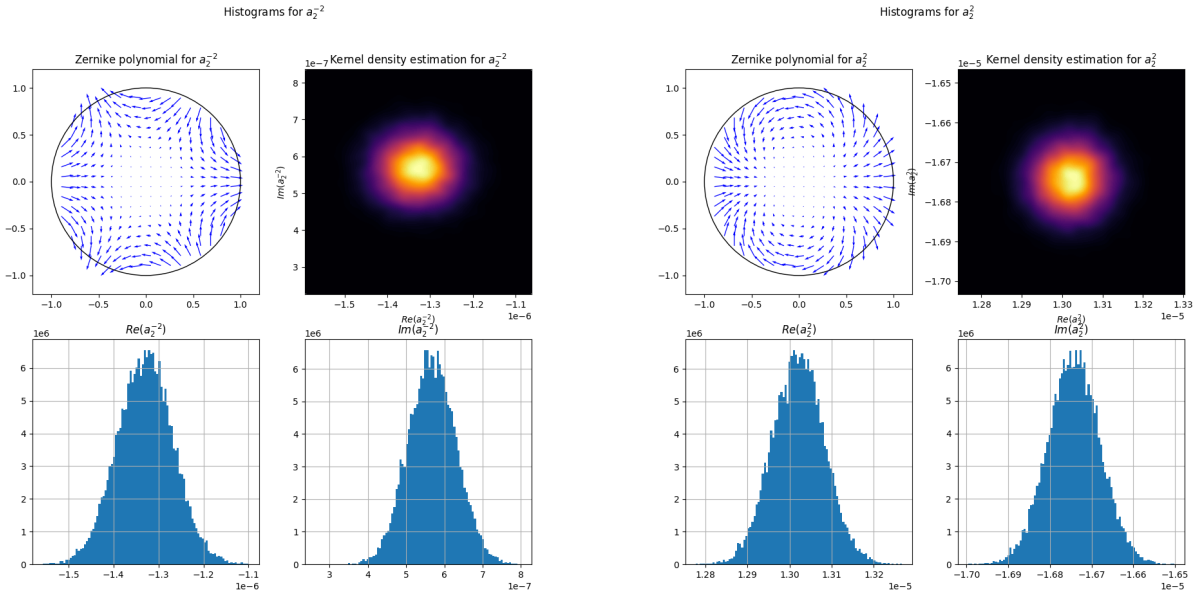

Histograms for a_0^0
Zernike polynomial for a_0^0
Kernel density estimation for a_0^0
$Re(a_0^0)$
$Im(a_0^0)$

E.2. $n = 1$

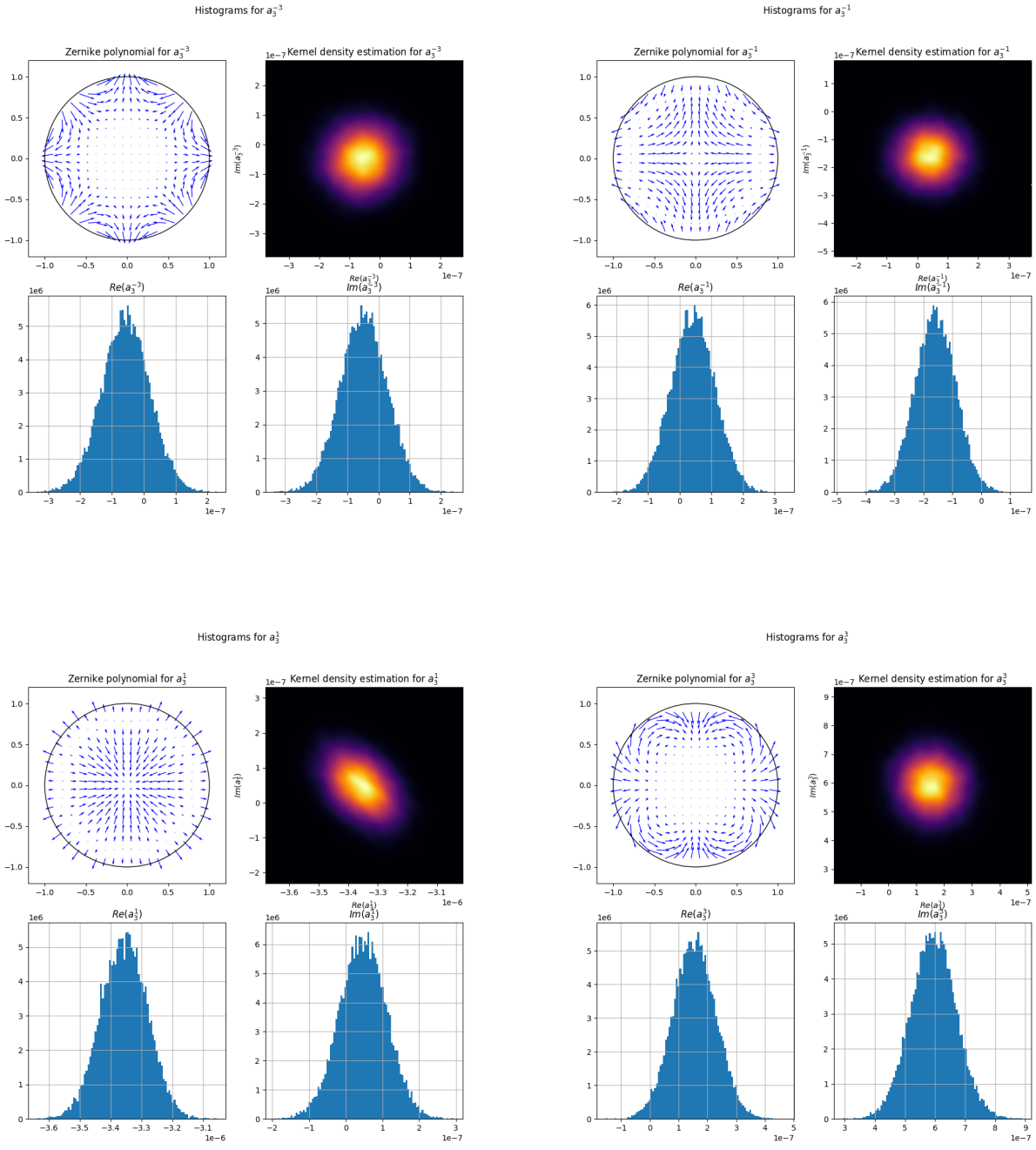

Histograms for a_1^{-1}
Zernike polynomial for a_1^{-1}
Kernel density estimation for a_1^{-1}
$Re(a_1^{-1})$
$Im(a_1^{-1})$
Histograms for a_1^1
Zernike polynomial for a_1^1
Kernel density estimation for a_1^1
$Re(a_1^1)$
$Im(a_1^1)$

E.3. $n = 2$


Histograms for a_2^0
Zernike polynomial for a_2^0
Kernel density estimation for a_2^0
$Re(a_2^0)$
$Im(a_2^0)$


Histograms for a_2^{-2}
Zernike polynomial for a_2^{-2}
Kernel density estimation for a_2^{-2}
$Re(a_2^{-2})$
$Im(a_2^{-2})$
Histograms for a_2^2
Zernike polynomial for a_2^2
Kernel density estimation for a_2^2
$Re(a_2^2)$
$Im(a_2^2)$

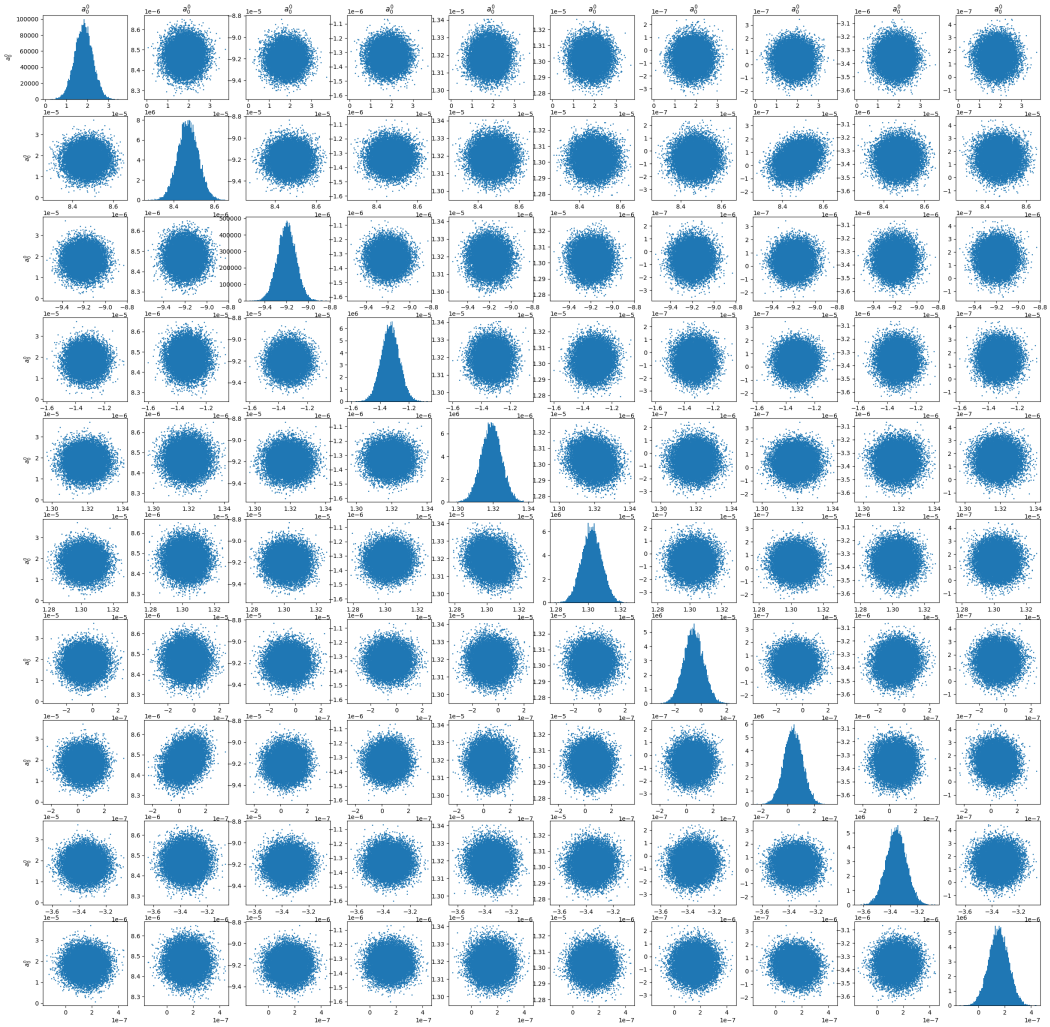
E.4. $n = 3$



E.5. Scatter plot matrices

The following scatter plot matrices detail potential correlations between the first 10 pointing model coefficients. Although a slight correlation between some coefficients can be identified, the absence of correlation is the general trend.

E.5.1. Real part



E.5.2. Imaginary part