UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Fisica e Astronomia "Galileo Galilei"

Corso di Laurea in Fisica

Tesi di Laurea

Characterization of the Visible Camera System for RFX-mod2

Caratterizzazione del Sistema di telecamere visibili per RFX-mod2

Relatore

Dr. Matteo Agostini

Correlatore

Dr. Andrea Belpane

Laureando

Jacopo Carotenuto

Anno Accademico 2021/2022

Contents

Abstract

RFX-mod2 is an experiment that aims to study the physics of fusion plasma and magnetic confinement in a Reversed Field Pinch (RFP) configuration. This thesis aims to characterize the diagnostic cameras of the Optical Camera System (OCS) of the new RFX-mod2. These cameras will observe the internal wall of the RFX-mod2 experiment to measure the visible emission due to the interaction between the plasma and the carbon first wall. This characterization will be accomplished using the Python library OpenCV, in particular leveraging his camera calibration algorithms to extract the internal parameters of the cameras from images and the relative position between the camera and the object observed. These parameters will then be used to reconstruct the camera's Field of View (FOV) in a 3D render of the RFX-mod2 machine in CAD software CATIA, and to calculate the final total surface coverage of all OCS cameras.

In chapter 1 the RFX-mod2 experiment is introduced and the role of the OCS is explained. In chapter 2 the theory behind camera calibration is presented. In section 3.1 the setup used is described, then from section 3.3 the program used to analyze the data and calibrate the camera is explained. In section 3.6 the 3D reconstruction of the camera and his field of view is produced.

Chapter 1

RFX-mod2

RFX-mod2 (Reversed Field eXperiment) is a toroidal machine that utilizes magnetic fields to confine plasma (hot ionized gas) exploiting the "Reversed Field Pinch" configuration, in alternative to the "Tokamak" configuration, based on less intense magnetic fields and only on the ohmic effect for heating the plasma. The experiment aims to study the physics of fusion plasmas and magnetic confinement in a Reversed Field Pinch (RFP) configuration.

The new machine (RFX-mod2) has a greater volume of plasma and the boundary of the plasma will be nearer to the magnetic coils that control the plasma position and instabilities. The state-of-the-art control system of magnetic instabilities is one of the best in the world. Thanks to the newest improvement it is capable of more efficiently controlling the energy leakage caused by the plasma-wall interaction with the graphite tiles of the first wall, obtaining "Single Elicity States" more stationary and hotter, and better-confined plasma. The challenge of this new machine is to be able to produce a plasma with much-improved parameters compared to those of the RFX-mod and to clarify whether the RFP configuration can be a viable alternative to the tokamak line in the perspective of a fusion reactor.

Some parameters of RFX-mod2 [6][7] are presented in Table 1.1.

Maximum Plasma Current	$2\,MA$
Maximum Plasma Temperature	$1.5 \cdot 10^6\ eV$
Impulse Time	$0.5s$
Average Electron Density	$\sim 5 \cdot 10^{19}$
Minor Radius	$0.459m$
Major Radius	$2.0m$

Table 1.1: RFX-mod2 future parameters

Figure 1.1 shows a comparison between RFX-mod and RFX-mod2, in particular the difference between the plasma minor radius.

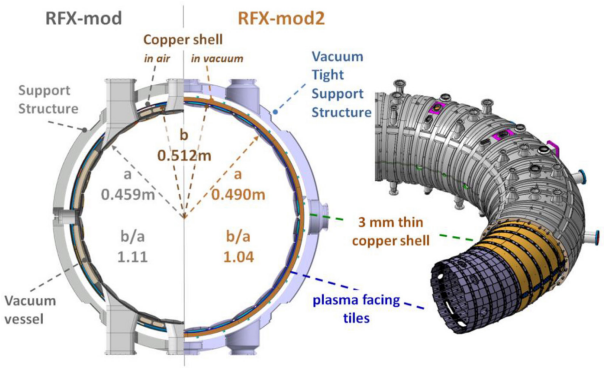

Figure 1.1: RFX Structure Overview [6]

1.1 Optical Camera System

The Optical Camera System of RFX-mod2 will be comprised of 7 visible light cameras observing the inside of the fusion chamber placed like in the Figure 1.2

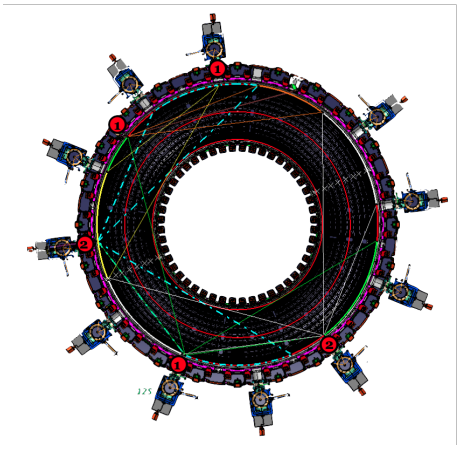

Figure 1.2: The schematic of RFX-mod2, with red circle where the camera will be positioned, and their field of view

Figure 1.2 describes the position of the cameras: there are 2 locations with 2 cameras, and 3 with 1 camera. This arrangement is able to cover almost all of the volume of the chamber. Every camera will be placed in cylindrical ports, with every compartment housing two or one camera. In figure 1.3 the diagnostic structure (two camera configuration) is reported.

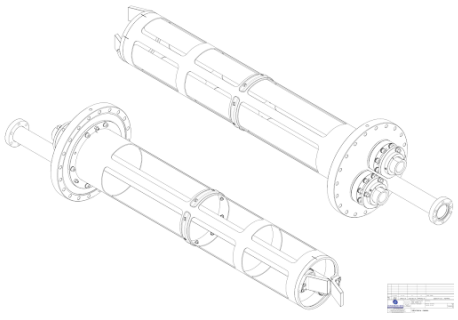

Figure 1.3: Isometric View of the ports

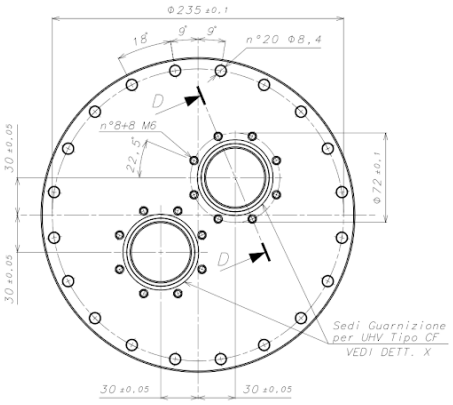

Figure 1.4: Front View of a port

The system is composed of a flange on which two ducts, that host the cameras, are installed. At the

end of the duct, two flat mirrors are used to obtain a toroidal view.

There will be 5 of these tubes, for a total of 7 cameras, installed at periodic intervals in the vacuum chamber. The cameras will provide new insights into the behavior of the plasma inside the chamber and more visual information on the plasma-wall interaction.

Data from the previous Camera Systems

The Optical Camera System (OCS) is designed for studying the plasma-wall interaction (PWI) inside the chamber. In RFX-mod an older version of the OCS was present and it provided insights into the configuration of the instabilities and deformations of the plasma. For example, the OCS was used to study the behavior of the PWI due to the locking of the magnetic modes[11]. An example of the images acquired in RFX-mod is shown in Figure 1.5 and 1.6: the difference in the magnitude of the plasma wall interaction with and without the locking of the magnetic modes can be clearly seen.

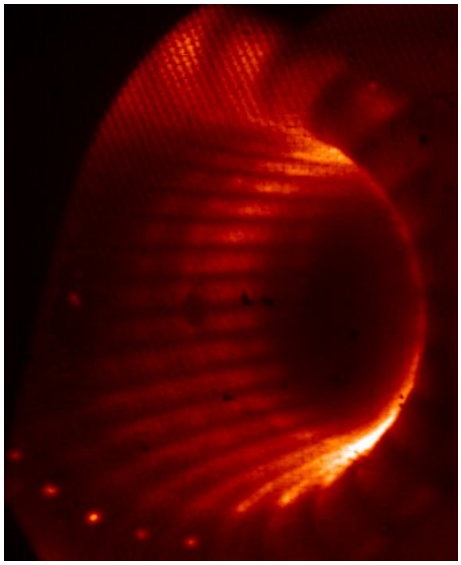

Figure 1.5: Without Phase Locking, PWI are weak and diffused

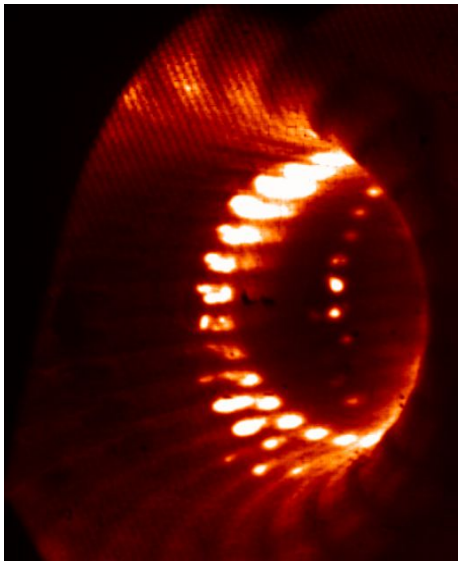

Figure 1.6: With Phase Locking, PWI are very localized and strong

The camera used in this example in RFX-mod had $2ms$ of integration time, and observed the CI line at 970 nm in a toroidal region centered at $\varphi = 0°$ with a toroidal field of view (FOV) of about 40°. Typically, the cameras were used with an interferential filter for the CI line, for looking at the neutral carbon emission. Figure 1.6 shows two stripes corresponding to distortion in the helical plasma, and from this type of images, their location and behavior can be studied to better understand the instabilities of the plasma configuration.

The images provided by the OCS are used also to confirm the magnetic reconstruction reliability the other diagnostics[11]. Another example of an image taken by the OCS of RFX-mod with a 3D reconstruction of the magnetic boundary is shown in Figure 1.7 [9]:

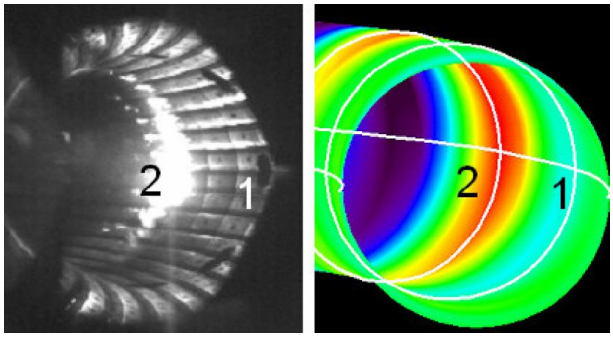

Figure 1.7: OCS image and 3D reconstruction [9]

Chapter 2

Camera Calibration

A camera is a tool to project objects in 3D space to a 2D surface. To study this transformation, the process that associates a 3D point in real space to a 2D point in pixel space must be understood.

First, a world coordinates system, $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, with a arbitrary origin is established. With this system, a point "A" with coordinates (X, Y, Z) is chosen. Now a camera is placed in the world coordinate system and pointed at our point "A". The coordinates of point "A" relative to the position of the camera need to be known. Let the camera position be (t_x, t_y, t_z). So, to get a coordinates system with the origin on the camera, the world coordinate system is *translated* of (t_x, t_y, t_z). But the camera has a specific orientation, so a *rotation* must be applied to the camera coordinate system relative to the world coordinate system. A 3D rotation is represented using three parameters in a 3×3 matrix. Let's call this matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$.

So, to obtain point "A" within the camera coordinates system it is simply rotated and translated in the world coordinates system.

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \mathbf{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{t} \tag{2.1}$$

The translation vector can be appended as a column at the end of the 3x3 rotation matrix, obtaining a 3×4 matrix. The fourth coordinate of the real world point is set to 1. So the equation is:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = [\mathbf{R}|\mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.2}$$

Now the 3D image is projected into the 2D coordinate system of the camera. Let's call the optical center (the pinhole of the camera) O_c. The image plane is placed at a distance f (called *focal length*) from the optical center. It can be shown that the coordinates of the projection are

$$x = f\frac{X_C}{Z_C} \qquad y = f\frac{Y_C}{Z_C} \tag{2.3}$$

Equation 2.3 is then expressed in matrix form in equation 2.4, where the coordinate z it retained after the projection because it will be necessary to normalize the image coordinates and obtain the right pixel coordinates.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \tag{2.4}$$

Matrix 2.4 is for a perfect camera, but in the real world, there are a number of possible distortions. For our model, two different focal lengths will be implemented, together with the coordinates of the optical center c_z, c_y, which may not be at the center of the image, and a skew factor γ to account for the possibility of skew between the x and y axes of the camera sensor. Taking into account all of these the resulting equation becomes Equation 2.5.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \tag{2.5}$$

The intrinsic camera matrix is then defined in Equation 2.6

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

Finally, the equation must be written in homogeneous coordinates, constraining $z = 1$ and adding a scaling parameter s so that the resulting x, y coordinates are then the real pixel coordinate of the objects in the images. The entire projection process from 3D world coordinates to 2D pixel coordinates is summarized in Equation 2.7.

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.7}$$

The process is visually described in Figure 2.1

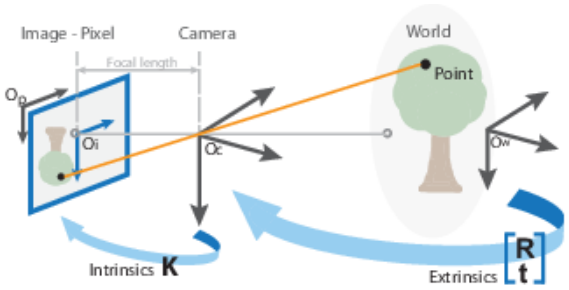

Figure 2.1: Projection Process [10]

However, to fully model a real-life camera with a real lens, the distortion effects need to be accounted for, both radial and tangential. Radial distortion is described in equations 2.8 and 2.9 and tangential distortion in equations 2.10 and 2.11:

$$x_{distorted} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.8}$$
$$y_{distorted} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2.9}$$

$$x_{distorted} = x + \left[2p_1 xy + p_2(r^2 + 2x^2)\right] \tag{2.10}$$
$$y_{distorted} = y + \left[p_1(r^2 + 2y^2) + 2p_2 xy\right] \tag{2.11}$$

With $(k_1, k_2, k_3, p_1, p_2)$ distortion parameters.

This is the complete camera model that will be used throughout this thesis.

Chapter 3

OCS Characterization

3.1 Test Camera

In this section, the camera used throughout the thesis is described. It is used to take various photos of the interior of RFX-mod2 with multiple checkerboards in view to perform a rough calibration and obtain a first estimate of the intrinsic parameters.

As all the cameras will be identical, it is necessary to only characterize one camera. So a camera identical to that of the optical system has been used for all the tests described here. It was placed inside a support structure equipped with a tilted mirror, to be able to deviate the camera's optical rays inside the vacuum chamber of RFX-mod.

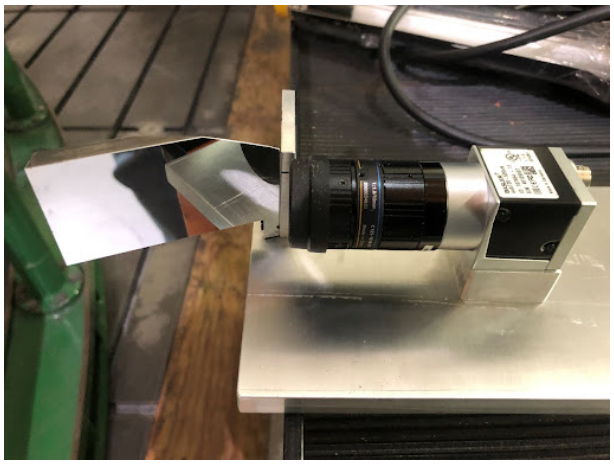

Figure 3.1: Test Camera with Mirror

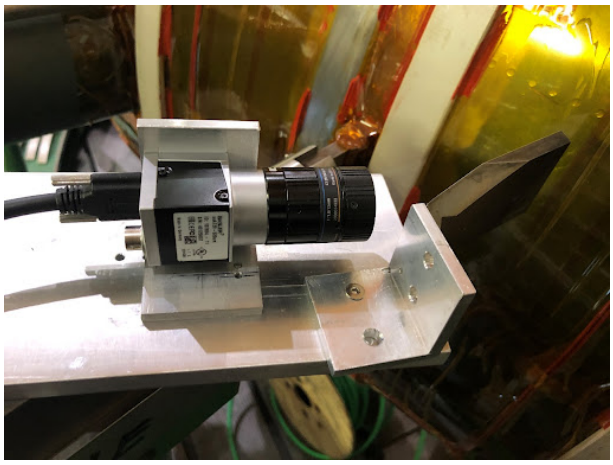

Figure 3.2: Test camera with mirror support visible

The camera was then positioned in one of the portholes for the diagnostics tubes, similar to the designed position. The mirror is positioned as in Figure 3.1 and in 3.2 the mirror support can be seen. In Figure 3.3 the draft of the old mirror, used for the test, is presented and in Figure 3.4 the new one can be seen.

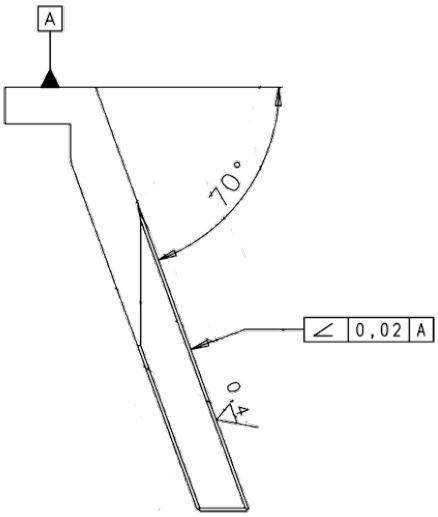

Figure 3.3: Schematic of old mirror

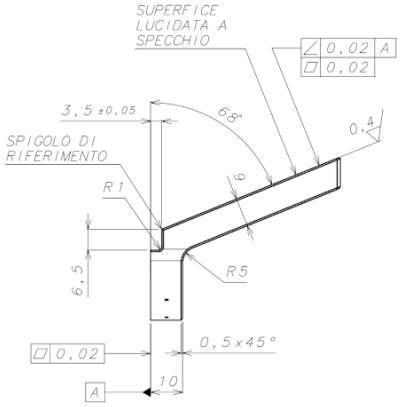

Figure 3.4: Schematic of new mirror

For the new mirror was calculated an angle of 68° like shown in Fig. 3.4. However, during the experimental phase it was adjusted to optimize the field of view obtaining an angle of 63.4°.

The old mirror had a tilt of 70°; for the new mirror an angle of 68° was calculated like shown in Figure 3.4. However, during the experimental phase it was adjusted to optimize the field of view obtaining an angle of 63.4° [1]. The camera is positioned at $(-320.157, -9.497, -2469.338)mm$, relative to the center of the torus. The camera used here, identical to the camera that will be used in RFX-mod2, is a "Basler acA720-520um". Its main parameters are summarized in Table 3.1

Resolution ($H \times V$ Pixels)	728×544 (full resolution) 720×540 (default resolution)
Sensor Type	Sony IMX287LLR-C Progressive scan CMOS Global shutter
Sensor Format	1/2.9”
Effective Sensor Diagonal	$6.3mm$
Pixel Size ($H \times V$)	$6.90\mu m \times 6.90\mu m$
Frame Rate (at Default Settings)	525 fps (at default resolution) 328 fps (at default settings)
Mono / Color	Mono

Table 3.1: Camera Parameters

3.2 Checkerboard Example

The simplest setup that can be used to calibrate a camera is to use a checkerboard pattern. Photographs from multiple different angles and orientations of some checkerboards are taken and then the built-in OpenCV function to detect checkerboards patterns and calculate the intrinsic matrix of the camera is used. This thesis uses the OpenCV[2] Python package to handle camera calibration. OpenCV is a library for Python[3] that gives the tools necessary to perform a variety of tasks in computer vision. It is the most well-known and well-developed package for this type of computation, and that’s why it’s used in this work.

For this example, multiple checkerboard patterns of various sizes were put inside the torus of RFX-mod

[1] All angles relative to radial direction
[2] OpenCV, version "4.5.5.64": https://opencv.org/,[2]
[3] Python 3.7, https://www.python.org/

and photographed them with the camera. Figure 3.5 shows some images of the checkerboard placed inside the RFX-mod vessel.

All the photos shown in this these were taken with an objective with focal length of $12mm$ mounted on the camera.

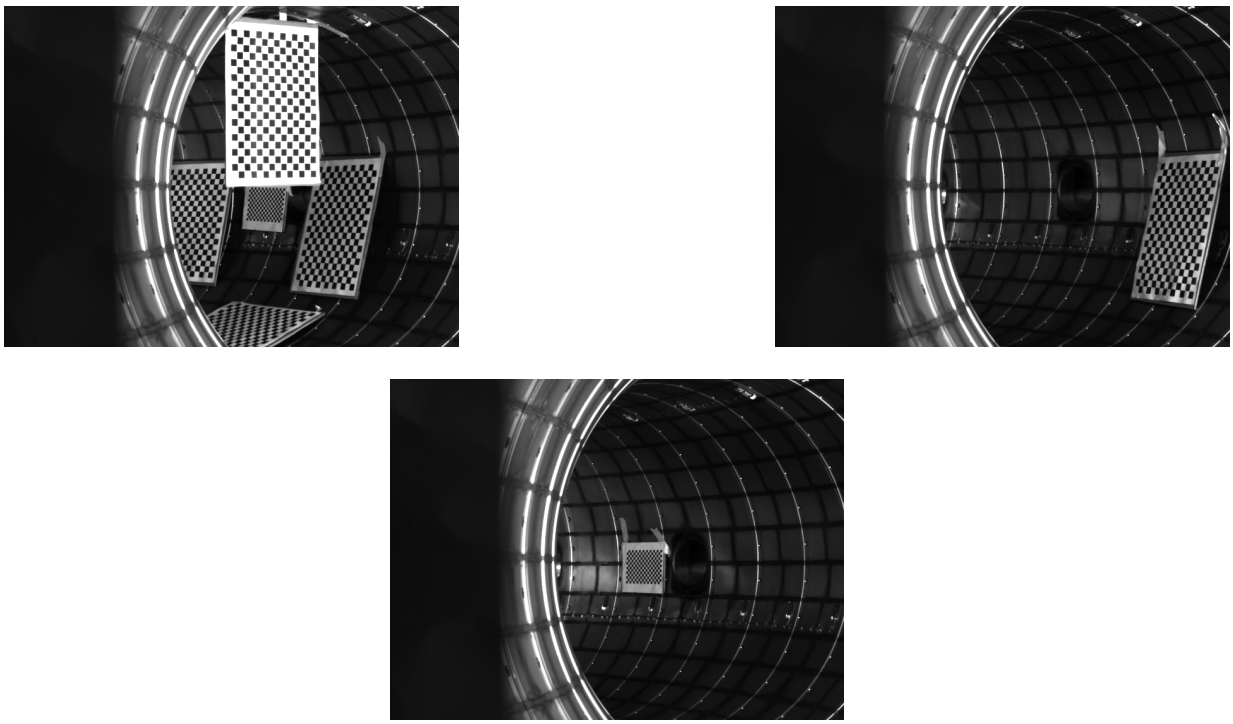

Figure 3.5: Examples of pictures taken

The built-in function "findChessboardCorners" is used to find the corners of all the squares, returned as an array of arrays.

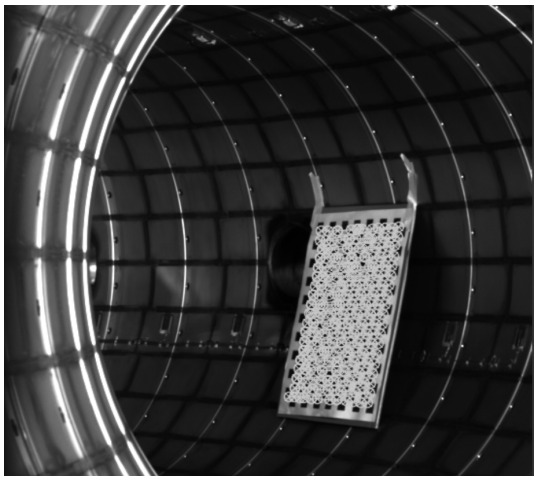

Figure 3.6: Corners detected by the algorithm

We use this information to run `calibrateCamera` to obtain a first guess of the intrinsic camera matrix

$$\mathbf{K} = \begin{bmatrix} 1140 & 0 & 266 \\ 0 & 1256 & 286 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Note that this matrix is expressed in pixel units. This is a very rough guess because the points from

the checkerboard are not very precise. As a matter of fact, if the focal length is converted from pixel to millimeters, $f_x = 7.87mm, f_y = 8.67mm$ are obtained, not at all the $12mm$ we expected. This is due to multiple factors such as the low quality of the images that result in poor pixel coordinates for the corners of the checkerboard, the checkerboard pattern not being completely parallel to the image plane, lack of light, and lack of camera focus. The method described is not sufficiently accurate to use in real experiments, so another method is required.

3.3 Code Overview

In this section a more accurate camera calibration is achieved by using only a single image, exploiting the high degree of accuracy of some construction parameters of RFX-mod2. This analysis uses the position of the screws that are visible inside the experiment both in 3D and 2D, as in both cases their coordinates can be obtained with a high degree of accuracy and in large numbers, resulting in more accurate extrinsic and intrinsic matrices.

This is done by leveraging the tools made available by the Python package "OpenCV" (version "4.5.5.64", [2]) to automatically calibrate the camera using construction information and photos taken inside the chamber. Two simple metrics are provided to determine the quality of the estimates of the program. The information is then used to study the positions of the cameras inside the chamber and provide a more analytical method to determine the area covered by the cameras.

The camera used in this analysis is identical to that of section section 3.1, so the parameters are the same.

3.4 Gathering the Data

With the camera in the position and rotation previously specified, multiple photos of the same section of the RFX-mod2 machine were taken. The picture below was selected to be used in the camera calibration process for the clearness of the features of the machine.

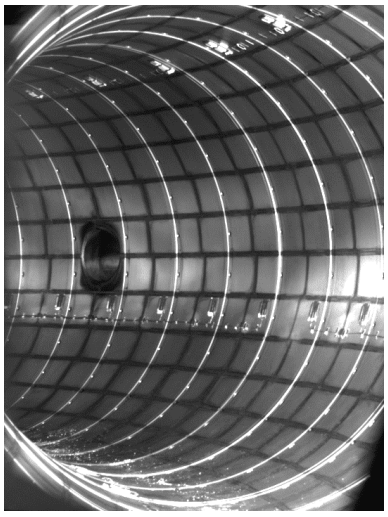

Figure 3.7: Image used for calibration, the contrast was artificially augmented to better distinguish features

As control points for the calibration, the screws, have been selected for their high number and ease of identification. As the position and orientation of the camera were known, it was easy to associate every screw on the image with its real counterpart.

Figure 3.8: 3D render of the inside of the experiment visible in Figure 3.7. The orange points indicate the screws

To acquire the pixel coordinates of the screws the contrast-augmented image was enlarged and each of the screws was identified manually. The screws were selected as control points for the ease of identification: each screw appears as a round coherent group of white pixels, most groups from 6 to 10 pixels each. The coordinates of the center of such groups were used as the position of the screws. 72 screws were identified with enough precision to be used in the calibration, with an error on both axes of approximately 0.1 pixels.

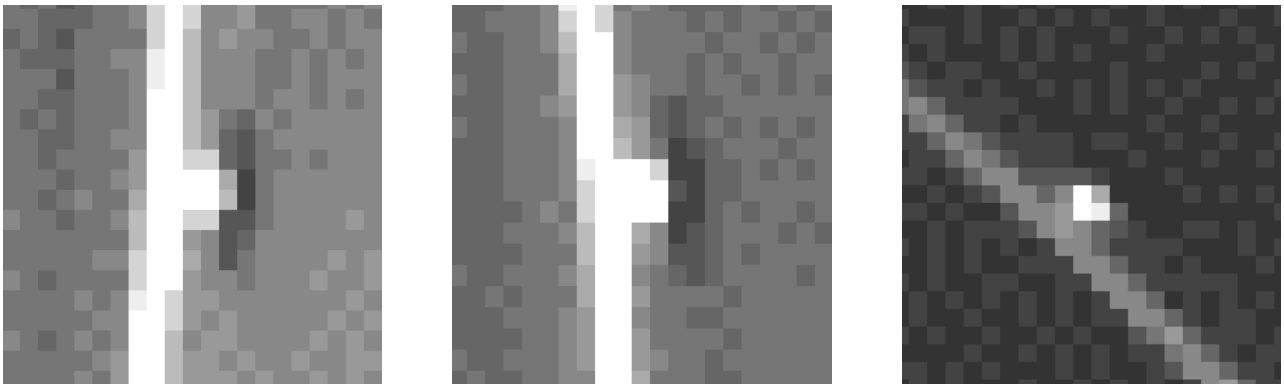

Figure 3.9: Some examples of the screws as they appear in the enlarged image

At the end of the analysis, 72 screws were selected and their coordinates were acquired. In Figure 3.10 the red points highlight the position of the screws in the acquired image.

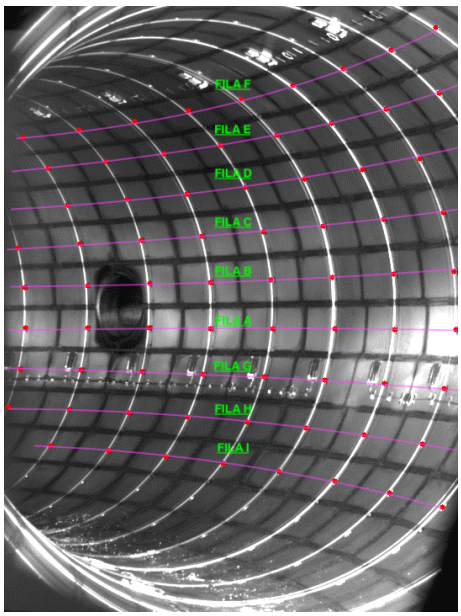

Figure 3.10: All the selected screws

The real-world coordinates of the screws were extracted from the CAD file of the RFX-mod2 experiment. The screws are each tagged with a letter (A-I, as in figure 3.10) and a number, assigned from left to right. In table 3.2 some examples of real-world and image coordinates are provided.

Screw Designation	Image Coordinates [pixel]	Real World Coordinates [mm]
A1	(537.250 ; 384.875)	(2021.82 ; -53.63 ; -1420.58)
A2	(464.938 ; 384.625)	(2137.94 ; -53.63 ; -1238.96)
A3	(393.125 ; 383.875)	(2237.79 ; -53.63 ; -1047.91)
A4	(319.417 ; 383.417)	(2320.60 ; -53.63 ; -848.89)
A5	(245.375 ; 383.000)	(2385.76 ; -53.63 ; -643.40)

Table 3.2: Example of 2D-3D coordinates pair

3.5 Camera Calibration

In this section the program written to use the 2D and 3D coordinates of the screws to perform the camera calibration and to obtain the intrinsic and extrinsic matrices is explained.

3.5.1 Preparing the Data for Input

The data of the screw positions inside the vacuum vessel are saved in CSV format, and it needs to be loaded in a format that can be accepted by OpenCV. The data is loaded as a Numpy[4] array directly from the file, then it is transformed into an array of arrays (the format needed for the OpenCV function) and then every element is cast as a `float32`.

This process is done for the 3D and the 2D coordinates.

```
temp = np.genfromtxt("./Programs/DATA/COOR_VITI_REAL_A-I.csv",delimiter=",")
temp = temp[1:,1:] # Get only the coordinates
REAL = np.array([[temp[0,:].tolist()]])
temp = temp.tolist()
for i in range(1,len(temp)):
  REAL = np.append(REAL,np.array([[temp[i][:]]]),axis=0)
REAL = REAL.astype("float32")
Points3D = [np.array(REAL)]
```

[4]Numpy Python Library for scientific computing [3], website

3.5.2 calibrateCamera Function

To use the camera calibration function, some more steps are needed. As the screws are not in a planar configuration in 3D, the `calibrateCamera` function need an initial estimate for the camera matrix $\mathbf{K}'$ (this is imposed by the OpenCV package). The estimate is calculated by the process described in section 3.2 using the checkerboards but with a large number of photos. As stated before, it is not a very good estimate for $\mathbf{K}'$ but it provides a good starting point for the function used. The estimate obtained is:

$$\mathbf{K}' = \begin{bmatrix} 1140 & 0 & 266 \\ 0 & 1256 & 286 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

With this estimate, the function is run with the appropriate flags, and it returns a camera Matrix $\mathbf{K}$, a rotation vector $\mathbf{R}$, a translation vector $\mathbf{t}$ and the distortion coefficients d.

```
InitialCameraMatrix = np.zeros((3,3))
InitialCameraMatrix[0,0] = 1.14009736e+03 # f_x
InitialCameraMatrix[0,2] = 2.66761656e+02 # c_x
InitialCameraMatrix[1,1] = 1.25646506e+03 # f_y
InitialCameraMatrix[1,2] = 2.86126004e+02 # c_y
InitialCameraMatrix[2,2] = 1 # Must be one for the algorithm to work

# Calibrate Camera Function with relevant flags
flags = cv2.CALIB_USE_INTRINSIC_GUESS
ret,mtx,dist,rvecs,tvecs = cv2.calibrateCamera(Points3D,Points2D,IMG_SIZE,
    InitialCameraMatrix,None,None,None,flags=flags)
```

The function returns:

$$\mathbf{K} = \begin{bmatrix} 1744 & 0 & 98 \\ 0 & 1749 & 387 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 0.01185 \\ 2.2423 \\ 0.0006328 \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} 1701 \\ 35.97 \\ -1910 \end{bmatrix} \tag{3.3}$$

3.5.3 Extracting the Parameters

The matrix $\mathbf{K}$ is expressed in pixel unit. To be able to check the focal length of the camera it is needed to convert to millimeters. As the camera is a "Basler acA720-520um" the pixel dimension is $6.9\mu m \times 6.9\mu m$ [1]. So the estimate for the focal length of the camera is $f_x = 12.03mm, f_y = 12.06mm$. The two focal lengths are the same, and they correspond to the nominal focal length of $12mm$ as expected.

It is worth noting that the parameters $c_x = 98, c_y = 387$, that denote the estimated optical center of the camera, are not in the center of the sensors, that is $(270, 360)$. This is due to a non-perfect alignment between the camera lens and the sensors, and it is expected for real-world commercial cameras, as the sensor is usually not aligned with the lens.

Extracting Camera Position

To obtain the camera position in 3D coordinates, the origin and rotation of the camera coordinate system need to be calculated. In computer vision this type of problem is called the "Perspective n-Point" problem: in this problem, the pose, that is the orientation and position of an object, is unknown and needs to be calculated from a set of points, the distortion coefficients of the camera and the camera matrix.

As all the necessary data to solve this type of problem has been already calculated and is also available an initial guess of the translation and rotation vector of the object, the function `solvePnP`, provided by OpenCV [2], can be used to calculate the exact pose of the camera.

This process is necessary because the translation and rotation vector given by the `calibrateCamera` function may not be the most accurate estimates of the real translation and rotation vectors, so another optimization algorithm must be run in order to obtain the most accurate results possible [4] [5].

The function returns:

$$\mathbf{R} = \begin{bmatrix} 0.01185 \\ 2.2423 \\ 0.0006328 \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} 1701 \\ 35.97 \\ -1910 \end{bmatrix} mm \tag{3.4}$$

The initial estimates were accurate, as there are no changes to the rotation and translation vectors from solving the PnP problem.

These vectors can be used to calculate the position of the camera. Let $\mathbf{C}$ be the position of the camera in the real-world coordinate system. In the camera coordinate system, the camera position is $\mathbf{0}$, as it is the origin of the coordinate system. As the camera coordinate system is simply the real-world coordinate system translated and rotated, Equation 3.5 can be written and solved for $\mathbf{C}$.

$$\mathbf{0} = \mathbf{C}\mathbf{R}_m + \mathbf{t} \Rightarrow \mathbf{C} = -\mathbf{R}^{\mathbf{T}}{}_m\mathbf{t} \tag{3.5}$$

where $\mathbf{R}_m$ is the rotation matrix obtained from the rotation vector R. To obtain $\mathbf{R}_m$ Rodrigues Formulas must be applied to convert the vector $\mathbf{R}$ to a matrix [8].

```
_, rot, trasl = cv2.solvePnP(REAL,FAKE,mtx,dist,rvecs[0],tvecs[0],True,0) # Find
  better estimates
rotM = cv2.Rodrigues(rot)[0] # Convert from rotation *vector* to rotation *matrix*
cameraPosition = -np.matrix(rotM).T * np.matrix(trasl) # Do the geometric
  transformation to find camera position
```

The function gives us:

$$\mathbf{C} = \begin{bmatrix} -437.5 \\ -30 \\ -2520 \end{bmatrix} mm \tag{3.6}$$

This is the position of the camera in the coordinate system centered in the center of the torus, and it is shown with respect to the RFX-mod2 experiment in Figure 3.11. The estimated position of the camera is not exactly where the camera was placed in reality, but due to the construction of the test apparatus and the use of mirrors to deflect the optical rays of the camera, the estimated position is the point behind the mirror where the optical rays would have converged without the mirror.

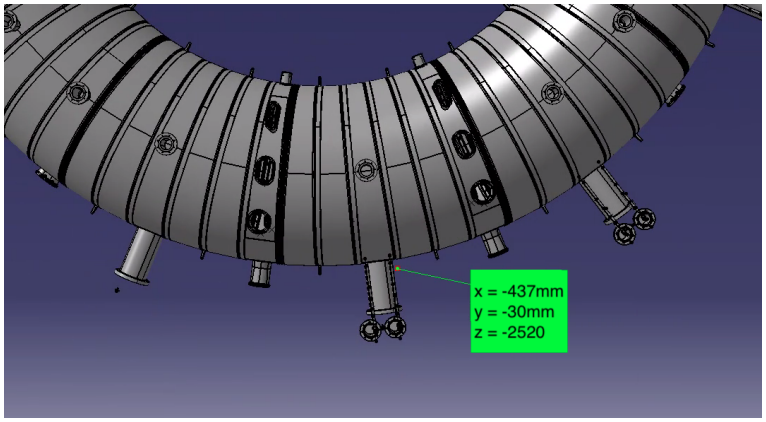

Figure 3.11: Position of the Camera in the Real World

The goodness of the estimates can be discerned by two simple metrics: the mean reconstruction error and the mean reprojection error. That is the mean error from the transformation of the coordinates from 2D to 3D and vice versa, and they will be discussed in the next section.

Reprojection

Using the `projectPoints` function all the 3D coordinates of the screws used for the image calibration can be projected in 2D coordinates p' and consequently the norm between them and the real 2D coordinates p can be calculated. In matrix form, this translates to:

$$s \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = K \left(R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t \right) \tag{3.7}$$

With s a scale factor these can be calculated, as X, Y, Z are the known position of the screws in the real world. The coordinates x', y' are transformed with these equations to account for the radial and tangential distortion effect of the camera lens. Radial distortion can be represented as follows:

$$x_{distorted} = x \left(1 + k_1 r^2 + k_2 r^4 + k_4 r^6\right) \tag{3.8}$$

$$y_{distorted} = y \left(1 + k_1 r^2 + k_2 r^4 + k_4 r^6\right) \tag{3.9}$$

Similarly, tangential distortion occurs because the image-taking lens is not aligned perfectly parallel to the imaging plane. So, some areas in the image may look nearer than expected. The amount of tangential distortion can be represented as below:

$$x_{distorted} = x + \left(2p_1 xy + p_2(r^2 + 2x^2)\right) \tag{3.10}$$

$$y_{distorted} = y + \left(p_1(r^2 + 2y^2) + 2p_2 xy\right) \tag{3.11}$$

All of this is done internally in the `projectPoints` function.

```
  mean_error = 0
for i in range(len(Points3D)):
    imgpoints2, _ = cv2.projectPoints(Points3D[i], rvecs[i], tvecs[i], mtx, dist)
    error = cv2.norm(Points2D[i], imgpoints2, cv2.NORM_L2)/len(imgpoints2)
    mean_error += error
```

The mean reprojection error is 0.07 *pixels*. It means that by projecting the 3D coordinates of the screws into the 2D sensor, the average distance between them and the real image is 0.07 pixels. This result confirms the goodness of the camera calibration obtained.

Reconstruction

Reconstruction is the reverse of reprojection. Given the pixel coordinates of the screws, the estimated matrix is used to obtain their 3D coordinates and then these are confronted with the real 3D coordinates of the screws.

OpenCV does not provide a function that calculates reconstruction, so it is been written.

```
  def reconstructPoints(Coor2D,Z, mtx,dist,rotM,trasl):
    # First we undistort the points
    X_u, Y_u = cv2.undistortPoints(Coor2D, mtx, dist, P = mtx)[0][0]
    s = ( (np.matrix(rotM).T*(np.matrix(trasl)))[2,0] + Z)/(np.matrix(rotM).T * np.
    linalg.inv(mtx) * (np.matrix([X_u,Y_u,1]).T))[2,0]
    pos_world = np.matrix(rotM).T * (-np.matrix(trasl) + s*np.linalg.inv(mtx)*np.
    matrix([X_u,Y_u,1]).T) # 2D to 3D relative to origin
    return pos_world
```

The $x_{distorted}$ and $y_{distorted}$ are known. The "undistortPoints" function is used to calculate the undistorted 2D coordinates of the point $x_{undistorted}$ and $y_{undistorted}$. Then the projection equation 3.7 is reversed to find the 3D coordinates.

$$R^{-1}\left(sK^{-1}\begin{bmatrix} x_{undistorted} \\ y_{undistorted} \\ 1 \end{bmatrix} - t\right) = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.12}$$

As can be seen, the reconstruction equation needs the 3D coordinate Z to be known to give the correct coordinates. For this benchmark, we provide the real Z coordinate of the points, as it is known. First it is necessary to calculate s: to do that first the equation can be rewritten, definining $\Gamma = R^{-1}t$ and $\Lambda = R^{-1}M^{-1}\vec{u}$ and $\vec{u} = \begin{bmatrix} X_{undistorted} \\ Y_{undistorted} \\ 1 \end{bmatrix}$ and $\vec{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$ as

$$\Lambda s - \Gamma = \vec{X} \tag{3.13}$$

Solving for s:

$$s = \frac{Z + \Gamma_3}{\Lambda_z} \tag{3.14}$$

With s calculated, the real-world position is then obtained with the equation above. The distance between the estimated and the measured position is calculated for every point and the mean taken.

```
recon_error = 0
for i in range(len(Points3D[0])):
    p = Points2D[0][i][0]
    reconstructed_p = reconstructPoints(p, Points3D[0][i][0][2], mtx, dist,rotM,trasl
    )
    recon_error += cv2.norm(Points3D[0][i][0] - reconstructed_p.T, cv2.NORM_L2)/len(
    Points3D[0])
```

The mean reconstruction error is $1.44mm$, so our estimates are deemed sufficiently accurate to model the camera.

3.6 CAD Analysis

3.6.1 3D

With all the geometric information about the camera, it is possible to incorporate a model of the camera inside the CAD model of RFX-mod2 to determine its Field of View and its coverage of the inside of the experiment. For this CAD analysis the CAD Software "CATIA"[5], an industry leader mechanical CAD manipulation software is used. First, the point corresponding to the estimated camera position is created, as in Figure 3.12.

[5]Dassoult Catia V5, website

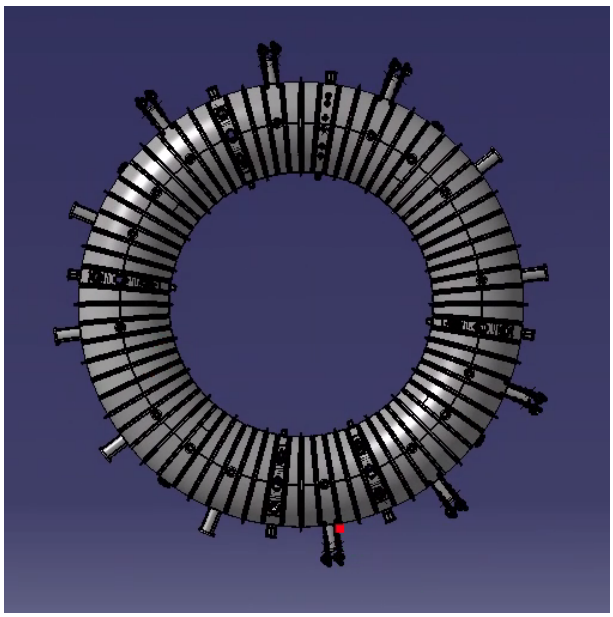

Figure 3.12: The red point indicates the estimated position of the camera relative to the whole RFX-mod2

As can be seen, the point is not exactly in the real position of the camera (as in figure 3.12), but is outside of the torus. This was expected as the camera's optical rays were deviated with a mirror, and the reconstruction did not account for this. As deviating the rays to the real position is a matter of simple geometry, from now on the camera is supposed to be in the estimated position (RED). To calculate the field of view (FOV) of the camera, the reconstruction process described in section 3.5.3 is applied to the corners of the image, so to points $UR = (540, 720), UL = (540, 0), DR = (0, 720), DL = (0, 0)$. As the process requires also a Z coordinates it is set to a nominal $1000mm$. This is not important as only the direction identified by the line passing through these points and the camera position are of interest to calculate the FOV of the camera.

The process returns these coordinates:

$$UR = (7490.0947, -1739.6365, 1000)mm \tag{3.15}$$
$$UL = (7584.4313, 1812.3586, 1000)mm \tag{3.16}$$
$$DR = (3511.4535, -1073.7309, 1000)mm \tag{3.17}$$
$$DL = (3533.1443, 1099.3940, 1000)mm \tag{3.18}$$

Drawing these points in the CAD software then drawing the lines that connect them to the camera, the FOV pyramid can be seen, as reported in Figure 3.13. The obtained field of view of the camera is shown as blue lines.

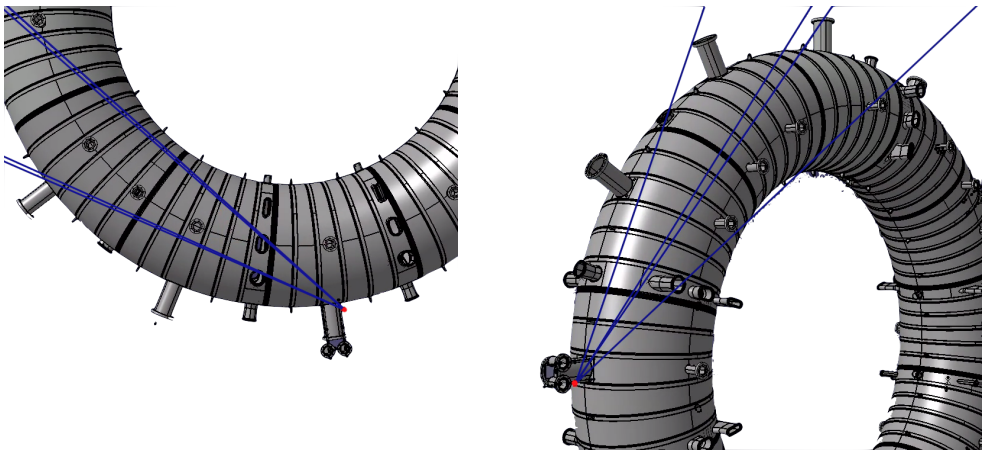

Figure 3.13: Camera FOV in CATIA

The FOV aperture can be measured: 17.5° horizontal and 23.0° vertical. If the scene is seen through the camera perspective, it can be confirmed that the picture used is exactly what can be seen inside the FOV of the camera, with millimetric precision. Comparing Figure 3.15 and 3.14 it can be seen the exact match between the two. Figure 3.15 can be seen exactly inside the green rectangle of Figure 3.14. For example, the left side of Figure 3.15 is tangent with one of the sector delimiters, and in 3.14 the situation is exactly presented.

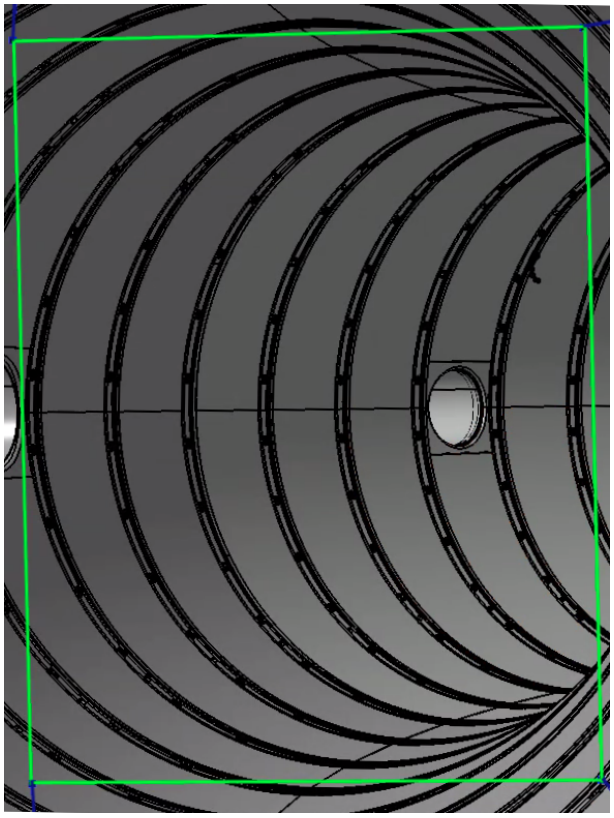

Figure 3.14: Green rectangle indicates the FOV of the camera

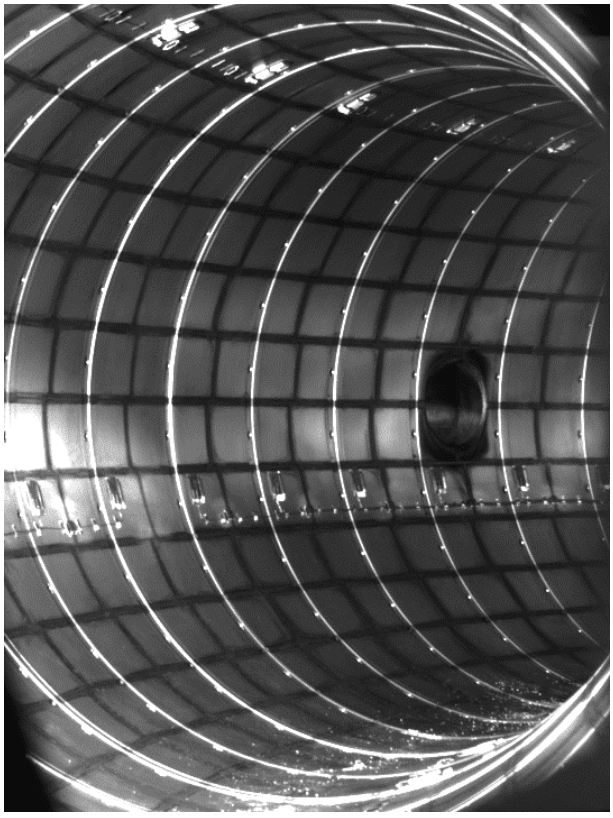

Figure 3.15: Real Picture for Comparison (mirrored)

3.6.2 2D

With all the necessary information, 2D drawings were created using CATIA. Knowing the FOV angle, the camera's field of view can be highlighted by drawing the boundaries of the camera FOV, as shown in figure 3.16.

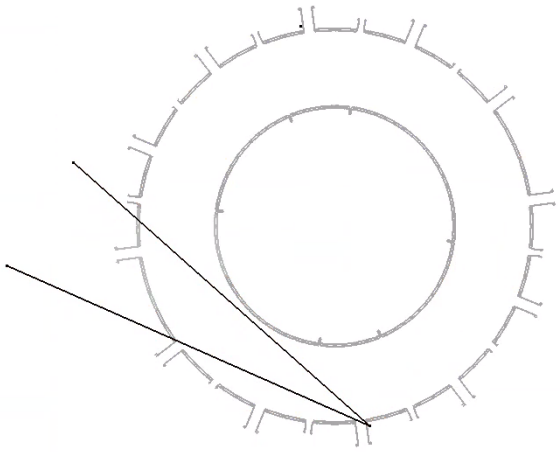

Figure 3.16: Test Camera Coverage

Then, as all the cameras will be identical, the construct can be copied and rotated accordingly to position all the cameras in the schematic. For better precision, the placement of the cameras in the double configuration is completely symmetrical with respect to the center of the porthole.

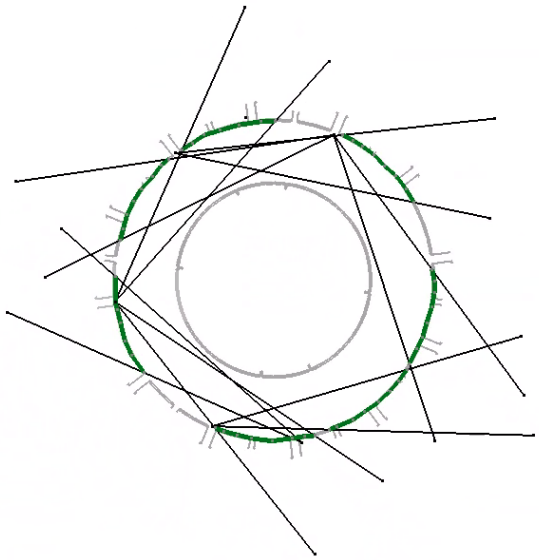

Figure 3.17: Green indicates the final coverage of the OCS

In Figure 3.17 the green curves indicate the area of the walls inside a camera Field of View. The system was designed to cover all of the insides of the experiment, but this analysis estimates a coverage of 70%, or approximately 260°.

Chapter 4

Conclusion

In this thesis, the Optical Camera System (OCS) of RFX-mod2 has been characterized. To achieve this, multiple images of the inside of the RFX-mod2 experiment were taken, and one was selected for the clarity of its features. From this image, 72 screws were identified, their pixel coordinates taken, and using the CAD model the respective 3D coordinates were identified. A Python program, leveraging the library OpenCV, was written to analyze these coordinates and extract the intrinsic matrix of the camera and its real-world coordinates. The intrinsic matrix is essential to build a model of the camera in 3D to calculate the total coverage of the OCS and to use only the 2D image to reconstruct the 3D model of the phenomenon observed, in this case, the Plasma Wall Interactions.

Using the coordinates pairs the newly created Python program was used to obtain the internal parameters of the camera, which achieved sub-millimetric precision measured by two different metrics. These parameters were then used to reconstruct the field of view (FOV) of the camera in a 3D reconstruction using CAD software CATIA and to position all the cameras in a 2D schematic of RFX-mod2 to calculate the total coverage of the new OCS, which is approximately 70% of the total chamber surface.

The next step is to use the reconstruction method with all 7 cameras of the OCS to recreate a complete image of the Plasma Wall Interaction inside RFX-mod2.

Another step is to use all the above information to precisely position all the cameras and to study the mechanical component that will house these cameras, which will involve inserting the cameras in ducts radially attached to the chamber and manipulating their chief rays with a mirror to better control their FOV. This information will be used to build an accurate mechanical cradle for each camera and to verify the compatibility of the test setup with the mechanical build. This will permit a new source of data regarding the plasma-wall interaction that occurs inside the RFX-mod2 machine and to further advance the understanding of the instabilities inside a plasma confined in a Reversed Field Pinch configuration.

Bibliography

[1] *acA720-520um | Basler*. URL: https://docs.baslerweb.com/aca720-520um (visited on 08/11/2022).

[2] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[3] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[4] Long Quan and Zhongdan Lan. "Linear N-point camera pose determination". en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.8 (Aug. 1999), pp. 774–780. ISSN: 01628828. DOI: 10.1109/34.784291. URL: http://ieeexplore.ieee.org/document/784291/ (visited on 09/07/2022).

[5] Xiao Xin Lu. "A Review of Solutions for Perspective-n-Point Problem in Camera Pose Estimation". en. In: *Journal of Physics* (), p. 9.

[6] L. Marrelli et al. "Upgrades of the RFX-mod reversed field pinch and expected scenario improvements". en. In: *Nuclear Fusion* 59.7 (July 2019), p. 076027. ISSN: 0029-5515, 1741-4326. DOI: 10.1088/1741-4326/ab1c6a. URL: https://iopscience.iop.org/article/10.1088/1741-4326/ab1c6a (visited on 08/10/2022).

[7] *RFX-mod2*. it-IT. URL: https://www.igi.cnr.it/ricerca/magnetic-confinement-research-in-padova/rfx-mod2/ (visited on 08/01/2022).

[8] Rodrigues. "Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire." fre. In: *Journal de Mathématiques Pures et Appliquées* (1840), pp. 380–440. URL: http://eudml.org/doc/234443.

[9] P. Scarin et al. "Helical plasma-wall interaction in the RFX-mod: effects of high- n mode locking". en. In: *Nuclear Fusion* 59.8 (Aug. 2019), p. 086008. ISSN: 0029-5515, 1741-4326. DOI: 10.1088/1741-4326/ab2071. URL: https://iopscience.iop.org/article/10.1088/1741-4326/ab2071 (visited on 08/10/2022).

[10] *What Is Camera Calibration? - MATLAB & Simulink - MathWorks Italia*. URL: https://it.mathworks.com/help/vision/ug/camera-calibration.html (visited on 08/01/2022).

[11] P. Zanca et al. "Plasma wall interactions in RFX-mod with virtual magnetic boundary". en. In: *Journal of Nuclear Materials* 363-365 (June 2007), pp. 733–737. ISSN: 00223115. DOI: 10.1016/j.jnucmat.2007.01.072. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022311507001286 (visited on 08/10/2022).