POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ability of a Camera to Define a Line of Sight Through a Ball Center for Machine Tool and Industrial Robot Calibration

CHRISTIAN JIRE KANDOLO

Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise des sciences appliquées*

Génie mécanique

Juin 2025

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Ability of a Camera to Define a Line of Sight Through a Ball Center for Machine Tool and Industrial Robot Calibration

présenté par **Christian Jire KANDOLO**

en vue de l'obtention du diplôme de *Maîtrise des sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

DEDICATION

To my dear family, who have been my pillar of support and encouragement during this journey, thank you.

To my parents, thank you for instilling in me the values of integrity, tenacity, and hard work. Your affection and selflessness have served as my beacon.

To my friends and coworkers: thank you for the support, motivation, and shared passion for engineering. My life and career have been made richer by your friendship and insights.

To my instructors and mentors: thank you for all your help, insight, and patience. Your commitment to learning and creativity has had a considerable influence on my career.

Finally, to everyone who believes that engineering can change the world. May our combined efforts keep expanding the realm of the possible.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to everyone who contributed to the success of this research endeavor and offered their unwavering support.

First and foremost, I am deeply indebted to my advisor, Prof. Rene Mayer, whose invaluable advice, patience, and insightful critique were instrumental throughout this study. His expertise, understanding, and guidance greatly enriched this work.

I am also particularly grateful to Vincent Mayer, Guy Gironne, and the technician team at the dimensional metrology laboratory of the Virtual Manufacturing Research Laboratory at Polytechnique Montréal for their technical support and for fostering a collaborative environment that made my research journey smoother.

On a personal note, I would like to extend my heartfelt thanks to my family and friends for their continuous encouragement and inspiration.

Finally, I wish to thank my peers and colleagues for their friendship and for the many thought-provoking discussions that greatly enhanced my research experience.

RÉSUMÉ

L'étalonnage des machines-outils est un processus essentiel, réalisé régulièrement tout au long de la durée de vie d'une machine afin de soutenir la planification de la maintenance, d'améliorer les performances par des ajustements mécaniques ou des compensations d'erreurs, et d'assurer une précision globale optimale. La fiabilité et la précision de l'étalonnage dépendent fortement du système de mesure utilisé. Les techniques conventionnelles font généralement appel à des dispositifs de métrologie de haute précision, tels que les interféromètres laser, les sondes capacitives ou les capteurs à courants de Foucault, qui sont certes très performants, mais souvent onéreux. Cette étude explore le potentiel d'utilisation de caméras compactes, économiques et disponibles dans le commerce comme solution de mesure optique sans contact pour l'étalonnage des machines-outils. L'objectif principal est de quantifier la capacité du capteur d'image à définir avec précision une ligne de visée passant par le centre d'une sphère de précision, une cible standard utilisée dans les procédures de type R-test. Le R-test est couramment employé pour évaluer la précision volumétrique des machines-outils multiaxes. Son efficacité, dans le cas d'une approche par caméra, repose sur une localisation rigoureuse du centre de la sphère par rapport à la ligne de visée de la caméra. Pour valider cette approche, un algorithme a été développé afin d'extraire des ellipses à partir d'images 2D de la sphère, de rétroprojeter leurs extrémités dans l'espace 3D à l'aide des paramètres intrinsèques de la caméra, et de calculer l'erreur géométrique entre les lignes de visée obtenues et le centre estimé de la sphère. En utilisant une caméra de 12.3 mégapixels, le système a atteint une précision de l'ordre de 2 µm. Ces résultats montrent que, dans des conditions contrôlées, des caméras compactes peuvent définir avec précision des rayons visuels traversant un point cible, offrant une alternative accessible, économique et sans contact aux dispositifs classiques d'étalonnage pour certaines applications industrielles.

ABSTRACT

Machine tool calibration is a critical process carried out regularly throughout a machine tool's operational lifespan to support maintenance planning, improve performance through mechanical adjustments or error compensation, and ensure overall accuracy. The reliability and precision of the calibration depend heavily on the measurement system used. Conventional techniques typically involve high-end metrology devices such as laser interferometers, capacitive probes, or eddy current sensors, which, although highly accurate, are often expensive. This study investigates the potential of using low-cost, off-the-shelf compact cameras as a non-contact optical measurement tool for machine tool calibration. The main objective is to quantify the camera sensor's ability to accurately define a line of sight passing through the center of a precision sphere, a standard target used in R-test procedures. The R-test is widely applied to assess the volumetric accuracy of multi-axis machine tools, and its effectiveness, when using cameras, depends on precise localization of the sphere center with respect to the camera's line of sight. To validate this approach, an algorithm was developed to extract ellipses from 2D images of the sphere, back-project their endpoints into 3D space using the camera's intrinsic parameters, and compute the geometric error between the resulting lines of sight and the estimated center of the sphere. Using a 12.3-megapixel camera, the system achieved a measurement precision of approximately 2 µm. These results demonstrate that under controlled conditions, compact cameras are capable of accurately defining visual rays through a target point in space, offering a viable, accessible, and low-cost alternative to traditional calibration equipment for specific industrial applications.

Key words: camera, calibration, center of sphere, industrial robot, machine tool, triangulation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

CAD	Computer Aided Design
CCD	Charged Coupled Device
CCS	Camera Coordinates System
CMM	Coordinate Measurement Machine
CNC	Computer Numerical Control
$\vec{D}$	Translation vector (Rigid body adjustment)
e_c	Ellipse center
f	Focal length
FOV	Field of View
IGF	Ideal Geometric Feature
IoT	Internet of Things
$\boldsymbol{K}$	Intrinsic matrix
LOS	Line of Sight
MP	Mega Pixel
o	Origin
P	Projection matrix
R	Rotation matrix
$\vec{R}$	Rotation vector
RMSE	Root Mean Square Error
s_c	Sphere (ball) center
SNR	Signal to Noise Ratio

TCP	Tool Center Point
x_c	Center coordinates in the x direction
y_c	Center coordinates in the y direction

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Overview

Over the past few decades, machine tool calibration has advanced significantly. Early methods were simple in principle but relied heavily on manual readings and adjustments. Since the late 20th century, technological progress in sensors, miniaturization, and wireless communication has transformed calibration practices. Modern tools including ball bars, electronic levels, and laser interferometers have largely supplanted traditional techniques, offering enhanced accuracy, reliability, and ease of use. Among the advanced methods developed in this context is the R-test, a diagnostic technique widely used for the volumetric calibration of CNC (Computer Numerical Control) machines. The R-test evaluates a machine's volumetric accuracy by detecting deviations from a programmed path as the machine moves through a series of predefined positions. In this procedure, a precision sphere is mounted on the machine spindle, while at least three linear displacement sensors either contact or non-contact are fixed to a fixture attached to the machine table. As one or more rotary axes move, the sphere shifts relative to the linear axes, generating translational displacements that are nominally replicated by the machine. The sensors then measure the position of the sphere's surface relative to themselves, from which the coordinates of the sphere's center can be calculated as described in [1]. Systems using touch-trigger probes to measure the sphere statically such as those provided by Renishaw [2] follow a similar approach. Common sensor types used in such calibration processes include capacitive sensors [3] and eddy current sensors [4]. Eddy current sensors are well-suited for non-contact measurement of conductive materials in harsh environments, while capacitive sensors provide extremely high resolution, ideal for applications requiring sub-nanometer precision [5].

Despite their benefits, touch-based methods have limitations, including material compatibility constraints, limited measurement ranges, and mechanical friction. These challenges have fueled interest in alternative approaches, such as calibration systems based on 3D optical camera sensors. However, the accuracy of these optical methods depends critically on precisely determining the line of sight to the sphere's center from a single camera viewpoint; a parameter that must be carefully quantified.

1.1.1 Research objective

- Quantify the ability of a low-cost compact camera to define a line of sight passing through the center of a precision sphere, for application in machine tool calibration.

1.1.2 Research sub-objectives

- Select an image-based method to detect the elliptical contour of a precision sphere:
    - ➢ Use existing computer vision techniques to extract ellipse parameters from 2D images of the sphere.
- Back-project image features into 3D rays using known camera intrinsics parameters
    - ➢ Use the pinhole camera model and intrinsic calibration to compute the 3D line of sight corresponding to the sphere center.
- Quantify the geometric error between the back-projected line(s) of sight and the known 3D sphere center.

- Report the root mean square error (RMSE) or minimum distance error achieved as a performance metric.
- Interpret the result.
- Discuss the strengths and limitations.

1.1.3 Organization of the thesis

The thesis is structured as follows:

- Chapter 1 gives the background related to the camera model in image processing and how to perform camera calibration to remove distortion related systematic errors;
- Chapter 2 is the literature review;
- Chapter 3 discusses the main camera parameters and camera calibration;
- Chapter 4 discusses the method used to perform feature extraction which will lead to finding the center of the sphere;
- Chapter 5 covers the development of an algorithm that will quantify the error between the sphere center point and the corresponding line of sight. This error gives us the accuracy of the sensor;
- The conclusion resumes the main outcome, and the future work proposes avenues to improve the system.

1.1.4 Methodology

The methodology used to achieve the desired results is:

1. **Review of State-of-the-Art:** Conduct a comprehensive literature review on the use of cameras in precision metrology, focusing on techniques relevant to machine tool calibration and the measurement of sphere centers.
2. **Selection of measurement technique:** Identify and select the most suitable image-based measurement technique for compact and low-cost cameras, considering factors such as accuracy, ease of implementation, and applicability to the task.
3. **Conceptualization of device operation:** Define the operating principles of the measurement system, including image acquisition, ellipse detection, and the mathematical framework for projecting image features into 3D space.
4. **Device design: Functions and constraints:**
   - **Functions:**
     - Capture high-resolution images of the precision sphere to detect elliptical contours accurately.
     - Enable precise alignment and stable mounting to minimize vibrations and positional errors during measurement.

- **Constraints:**
    - Use low-cost, off-the-shelf components to ensure affordability and accessibility.
    - Maintain compactness and portability for ease of use in industrial environments.
    - Manage lighting conditions to avoid reflections and shadows that could degrade image quality.
    - Account for lens distortion and sensor noise, which can affect measurement accuracy.
    - Limitations in processing speed or computational resources due to hardware constraints.

5. **Device assembly:** Procure all necessary components and assemble the measurement device according to the design specifications, ensuring proper alignment and calibration.
6. **Performance validation:** Develop an experimental setup capable of generating known displacements of a precision sphere within a 5 mm side cube. Use this setup to validate the critical performance attribute which the accuracy of the camera-based system in reconstructing the line of sight passing through the sphere, by quantifying the geometric error between the estimated line of sight and the known center of the sphere.

CHAPTER 2 LITERATURE REVIEW

2.1 Introduction

Modern production processes depend heavily on the accuracy and precision of machine tools. Machine tool calibration has become crucial to maintaining optimal performance as industries demand ever tighter tolerances and higher quality. To test, correct, and validate the performance of their machine tools, manufacturers rely on calibration instruments. This literature review of machine tool calibration devices' current state-of-the-art.

2.2 The R-test

The R-test device is a measuring tool that uses at least three displacement sensors to detect the three-dimensional translational deviations of a precision sphere mounted to the spindle with respect to the device mounted on the machine workpiece table or vice versa. It is used for error calibration of five-axis machine tools in both academia and industry since it is simple to calculate the sphere center displacement. First introduced in [1], the static R-test process detects rotary axis location error relative to the linear axes. The performance of a conventional contact-type R-test in dynamic measurement may be affected by the dynamics of the supporting spring or friction in the displacement sensors [2]. Some early conventional R-test devices, including commercially available ones use a contact-type linear displacement sensor with a flat-ended probe [3]. The impact of friction or supporting springs in conventional contact-type R-test devices can differ greatly depending on the measurement parameters (such as feed rate) and the composition of the sphere or probe surface [2]. Measurement uncertainty associated with friction dynamics or spring dynamics is generally difficult to quantify. When paired with the R-test, non-contact displacement sensors offer a few built-in advantages such as: (1) no measurement distortion caused by friction on the sphere surface and probes; (2) no measurement distortion caused by the dynamics of a supporting spring in contact-type displacement sensors; and (3) measurement safety resulting from a longer working distance between the sphere and sensors [2]. Laser displacement sensor are non-contact presenting the advantages of longer measuring range and reference distance than other non-contact sensors like capacitive and inductive displacement sensors [2].

2.2.1 Non-contact R-test devices

IBS Precision Engineering commercializes an R-test device using a capacitive probe [3] to detect the volumetric error of five-axis machine tool sensor as described in Figure 2.1. Because of the capacitor's limited sensing range and short standoff distance, caution must be used when installing and configuring the device to prevent unintentional collisions.

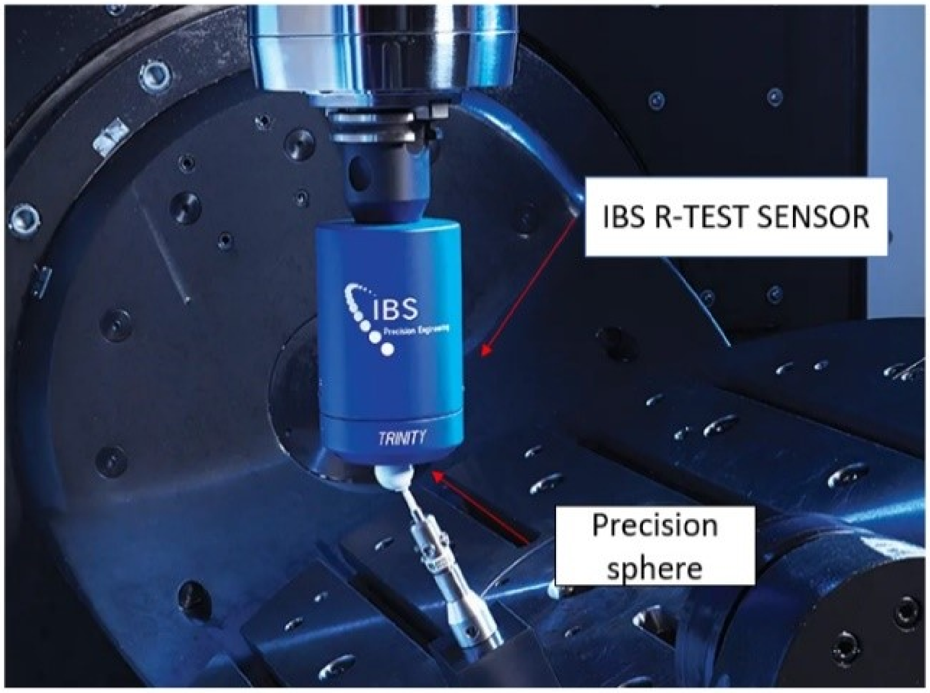


Figure 2.1 Illustration of the IBS R-test sensor with a precision sphere performing the commonly used R-test on a machine tool [3]

A non-contact R-test device called the "Capball" with capacitance displacement sensors was proposed by Zargarbashi and Mayer [4]. With the help of the suggested single setup test, the approach enables the estimation of all eight link errors, or the three-squareness error of linear axes, the four orientations errors, and center line offset between rotational axes. The system is illustrated in Figure 2.2.

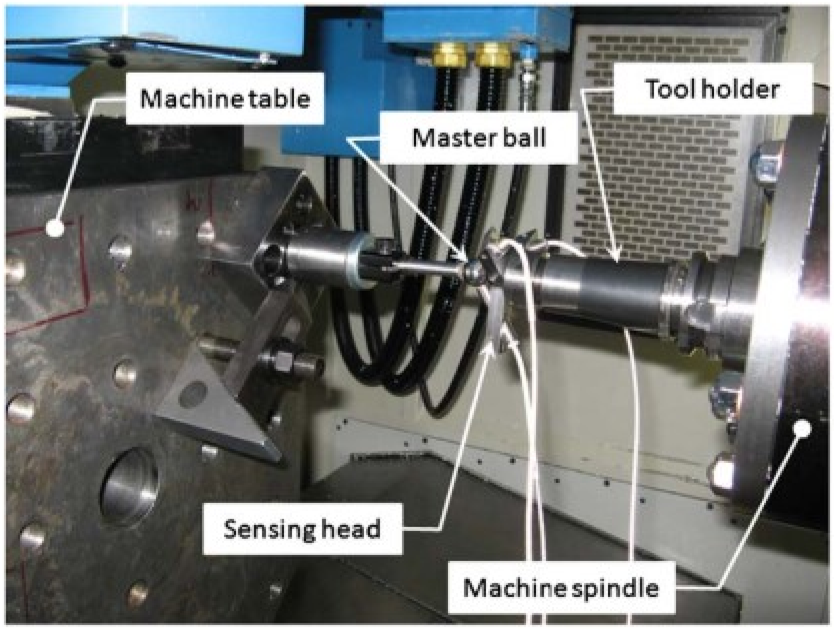


Figure 2.2 The Cap-ball measuring device: The master ball is mounted in the tool holder attached to the machine spindle, while the sensing head is fixed on the machine table. As the machine executes programmed movements, the sensing head detects the displacement of the master ball's surface [4]

2.3 Optical Methods

The geometry and motions of a machine tool can be captured in detail in space using 3D optical sensors [5]. These sensors can produce precise three-dimensional (3D) representations of machine tools by using light such laser triangulation [6] , which typically offers accuracy in the range up to 30 nanometers [7]. A division of current optical method used is illustrated in Figure 2.3 below:

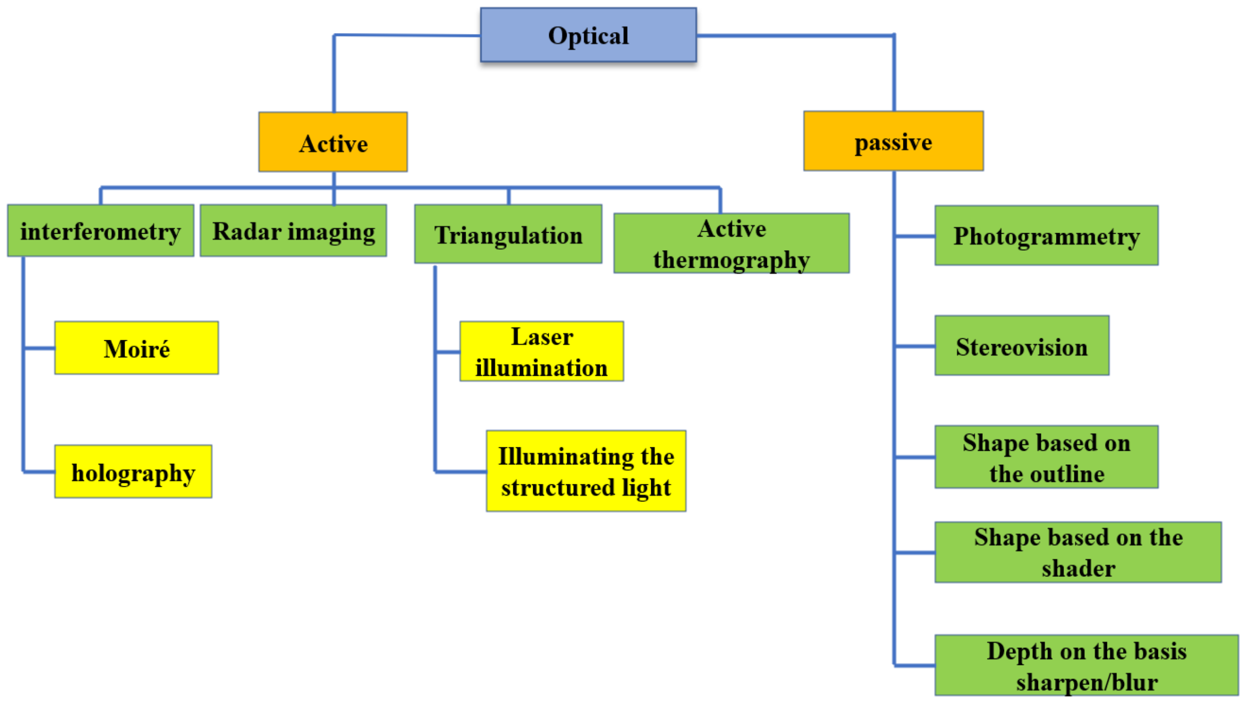


Figure 2.3 Division of different optical methods [5]

Laser based technologies enabled the development of novel measurement techniques for assessing various physical attributes such as distance. There are three ways to measure something accurately using a laser beam, depending on how large and how far away is the object: interferometer method, telemetric with modulated beams and optical radar [8].

Etalon [9] commercializes the Laser TRACER high-speed automatic tracking laser length finder to automatically track the length data obtained by the laser to detect the three-dimensional position of a target by combining 3 to 4 of such measurements. In [10] they developed an R-test device using optical noncontact detection composed of four-quadrant photosensors, lasers, and lens modules as, shown in Figure 2.4, based on the ISO-10791–6 testing method [11]. This system can be used to detect errors in high-end and five-axis machine tools, such as eccentricity errors and five-axis motion trajectory translational errors (volumetric error). Therefore, it only needs to be set up once, and during a single measurement process, the X, Y, and Z trajectory errors during five-axis movement can be obtained simultaneously. This greatly reduces setup and measurement time.

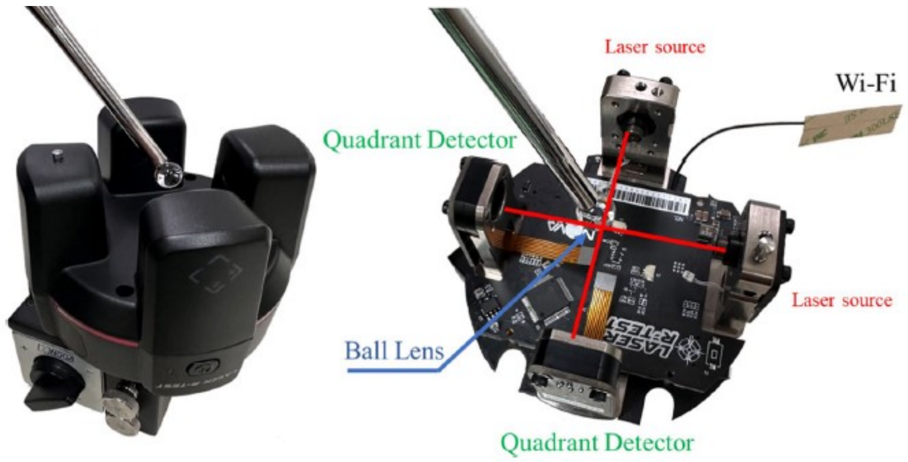


Figure 2.4 The laser R-test made of four quadrant photosensors : two laser sources and two quadrant detector and a ball lens [12]

Hong and Ibaraki introduced an enhanced non-contact R-test by using three laser triangulation displacement sensors [2].

Photogrammetry (e.g., stereovision system) can measure the 3D position of features by employing two or more cameras and triangulating the position of the same feature in each image. Calibration of the cameras is the initial step towards adopting camera triangulation. This entails estimating the extrinsic parameters, which specify each camera's location and orientation in a common coordinate system, as well as the intrinsic properties of each camera, like focus length and distortion coefficients. Extracting and matching features from the photos is the next stage. Any distinguishing points or patterns, such as corners, or edges, that are easily recognized throughout a series of photos can be considered features. Once the characteristics coincide, the triangulation procedure can start. By finding the intersection of the lines of sight from each camera that passes through a feature, one can determine the coordinates of the feature in three dimensions. Either the Iterative Linear Triangulation (ILT) approach or Direct Linear Transformation (DLT) can be used for this. The technique can be used for several tasks like object tracking, 3D reconstruction, and augmented reality.

2.4 Some limitations of optical systems

Certain camera-based triangulation methods offer a smaller measurement volume compared to other systems. Additionally, the material characteristics of the object being measured can limit the suitability of specific techniques. Moreover, surface properties such as high reflectivity can interfere with non-contact systems like laser radar. As mentioned earlier, restricted access to measurement points can also present significant challenges [13].

Air turbulence significantly affects the propagation of light through air, imposing fundamental limitations on the achievable measurement accuracy of optical systems. In addition to turbulence, ambient lighting can introduce noise into the detected signal, further degrading precision. Moreover, contamination of optical components such as lenses and sensors can substantially reduce measurement accuracy by scattering or attenuating the optical signal. These factors collectively present critical challenges for maintaining the reliability and repeatability of optical-based calibration systems in industrial environments [14]. Another limitation of optical systems, such as laser interferometry, is that setup-induced uncertainties can arise if the laser beam is not perfectly aligned with the direction of motion. Moreover, for accurate measurement of linear positioning errors, the interferometer optics must be mounted on the stationary structure, while the reflector moves with the machine axis or component under test; conditions that may not always be feasible in complex or spatially constrained setups [15].

For stereo systems using cameras it is important that the two cameras are synchronized, in case of target motion relative to the measuring head to avoid mismatches from the images taken from two different viewpoints that leads to incorrect point correspondences and errors in depth estimation.

2.5 Optical calibration using cameras

A noncontact method [16] to measure the calibration points in space using a stereovision system mounted on the robot arm was proposed. Regardless of the viewing angle, points are represented as spheres that project an ellipse in two image capture planes. Each sphere center's spatial coordinates are obtained for various robot setups. Robotic absolute positioning errors are measured based on these readings. Improved accuracy results are seen from calibration trials conducted on an industrial robot, a KUKA KR 6 R900. A camera pixel is said to map to 0.05 mm. The greatest positioning error around calibration points dropped from 3.63 mm before calibration to 1.29 mm after the operation.

A novel approach was proposed in [17] making use of a vision-based closed-loop calibration technique with point and distance constraints to detect a spherical ceramic ball with a diameter of 25.4 mm (approximately 1 in). The partial robot's pose was measured using a single high-resolution camera. To minimize lens distortion, the ball was positioned in the middle of the camera's field of view. Using the ball's center as a constraint point, the camera location was adjusted so that the ball's center was on its optical axis. Their approach required a high computation time and resulted in a significant drop in processing speed, even if it was simple to use and increased the Stäubli TX-60 robot's positioning accuracy. Improving the positional accuracy was also challenging due to the difficulty of maintaining the ball's center and the camera's optical axis coincident.

In [18] a 3D reconstruction-based measurement technique is suggested. To increase the accuracy of position identification in 2D images, a newly created self-illuminating target with a high signal-to-noise (S/N) ratio is combined with the extended intensity fitting method.

In [19] to improve the milling industrial robots position and orientation (pose) of the robot tool center point (TCP), a stereo vision system was employed. The results of the studies showed that the stereo vision system's pose information was more accurate than the robot's measurement. The results show that the robot's absolute positioning error can be lowered to about 0.1 mm. The system accuracy is up to 50 µm per m^3 measuring volume and the volume range is 1-2 m^3.

In [20] an identification algorithm and a binocular-vision based error detection system are proposed. First, the 3D error detection system is examined. High-precision set up and high-signal-to-noise picture acquisition are made possible by the creation of a novel self-luminous cooperative target, which is intended to characterize the movement information of a rotary table. Furthermore, a sphere center localization approach based on reconciled conjugate constraints is used to increase the rotary table's position precision to further ensure the correctness of the 3D vision measurement.

In [21] Several precision tests are performed to assess the accuracy of various CCD camera calibration and measuring techniques used in 3D stereo vision, identifying their error limitations under predetermined operating conditions. The tools used include a CMM and a few calibration objects like plates, spheres, grids, etc. Two useful applications, a low-cost free-form surface measurement device that can produce CAD models and CMM measuring programs, are explained. The goal of technology is to expedite the conventional digitization process while mitigating some of the challenges related to stereo vision. The other application is for measuring automotive frames. Using two rotating CCD cameras, a new automatic measuring system was created that enables contactless automotive frame measurement.

Traslosheros et al. [22] suggested a new, low-cost, external calibration technique for adjusting the three degrees of freedom (DOF) of a parallel robot. It uses positions (not absolute but incremental positions) from the motor resolvers and visual information from a spherical element to determine the joint and 3D incremental locations. It then solves the constraint equations numerically to get the optimal set of geometrical parameters. To compare optimal behavior with calibrated parameters to nominal parameters, several tests are finally run. This approach's primary contribution is the low-cost sensor used to calibrate the robot, and the methodology applied with the obtained results. The visual measuring system consists of a calibrated camera at the robot's end effector and a fixed ball supported by a rod. However, it is not possible to achieve accuracy better than 0.02 mm.

Recent advances in learning-based pose estimation have explored the use of event cameras, which record pixel-level changes asynchronously and offer high temporal resolution. Tabia et al.[23] introduced a deep neural network for regressing the 6-DoF pose of an event camera by first converting raw events into dense image-like representations and then using a dual-branch convolutional network to extract spatial features. Their approach avoids recurrent architectures, relying instead on pixelwise outer product encoding and global pooling, enabling faster convergence and reducing overfitting. Compared to traditional geometric approaches, such as those based on aligning lines of sight (LOS) to known sphere center points, this method removes the need for explicit calibration or physical modeling and demonstrates robustness under fast motion and variable lighting.

Haralick et al. [24] presented a comprehensive and rigorous treatment of the pose estimation problem from corresponding point data. Their work addresses several configurations, including 2D–2D, 3D–3D, and most importantly, the 2D–3D perspective pose estimation, which is highly relevant in computer vision and metrology. They introduced a globally convergent iterative algorithm for estimating the rigid-body transformation (rotation and translation) that aligns a set of

3D world points with their 2D image projections, even under significant noise. The authors proved that their method avoids local minima; a common issue in nonlinear optimization by leveraging a gradient-based descent scheme on a specially constructed cost surface. Additionally, they performed extensive Monte Carlo simulations to analyze the behavior of their algorithm under various levels of measurement noise, demonstrating that achieving high accuracy in low signal-to-noise conditions requires a large number of point correspondences. The paper also includes a robust variant of the algorithm designed to handle outliers and mismatches in the correspondence set. Compared to modern learning-based methods or geometry-specific approaches like line-of-sight (LOS) alignment to sphere centers, Haralick et al.'s algorithm offers strong theoretical guarantees and remains a benchmark for accuracy, convergence reliability, and analytical transparency in pose estimation.

This evolution in calibration technologies highlights a growing demand for systems that are not only precise but also more accessible, flexible, and adaptable to diverse industrial contexts. While traditional sensors such as capacitive and eddy current probes continue to deliver high accuracy, their high cost, limited measurement range, and need for careful handling restrict their widespread adoption, especially in environments requiring rapid setup, portability, or automation. Furthermore, the mechanical complexity and reliance on contact-based measurement in many existing systems pose limitations in dynamic applications or with sensitive materials. To move toward an ideal calibration system one that is accurate, low-cost, compact, easy to deploy, and compatible with smart manufacturing platforms there is a need to explore alternative technologies.

This project proposes to investigate the feasibility of using a single compact camera to determine a line of sight passing through the center of a precision sphere. Instead of relying on physical contact or expensive probes, the system leverages image-based processing techniques to extract the apparent ellipse of the sphere and reconstruct the optical ray corresponding to each camera viewpoint. An algorithm is developed to quantify the deviation between this ray and the actual sphere center, allowing for a direct assessment of measurement precision.

Ultimately, this work contributes to the ongoing transition from complex, sensor-heavy calibration systems to lightweight, digital, and scalable solutions. It illustrates a path forward toward integrating optical metrology with IoT-based infrastructure, enabling wireless, automated, and remote calibration procedures. As such, this research serves as an essential step toward realizing a practical, camera-based alternative to traditional machine tool calibration methods bridging the gap between current practice and an ideal Industry 4.0 compatible metrology solution.

CHAPTER 3 CAMERA PROPERTIES AND CAMERA CALIBRATION

3.1 General camera pinhole model

Let us consider the projection of points in space onto a plane. Let the center of projection be the origin of an Euclidean coordinate system and consider the plane $Z = f$ which is called the image plane or focal plane [25]. Using the pinhole camera model, a 3D point in space with coordinates $\boldsymbol{M} = [X, Y, Z]^{\mathrm{T}}$with its projection on the image plane as $\boldsymbol{m} = [u, v]^{\mathrm{T}}$ is illustrated in Figure 3.1.

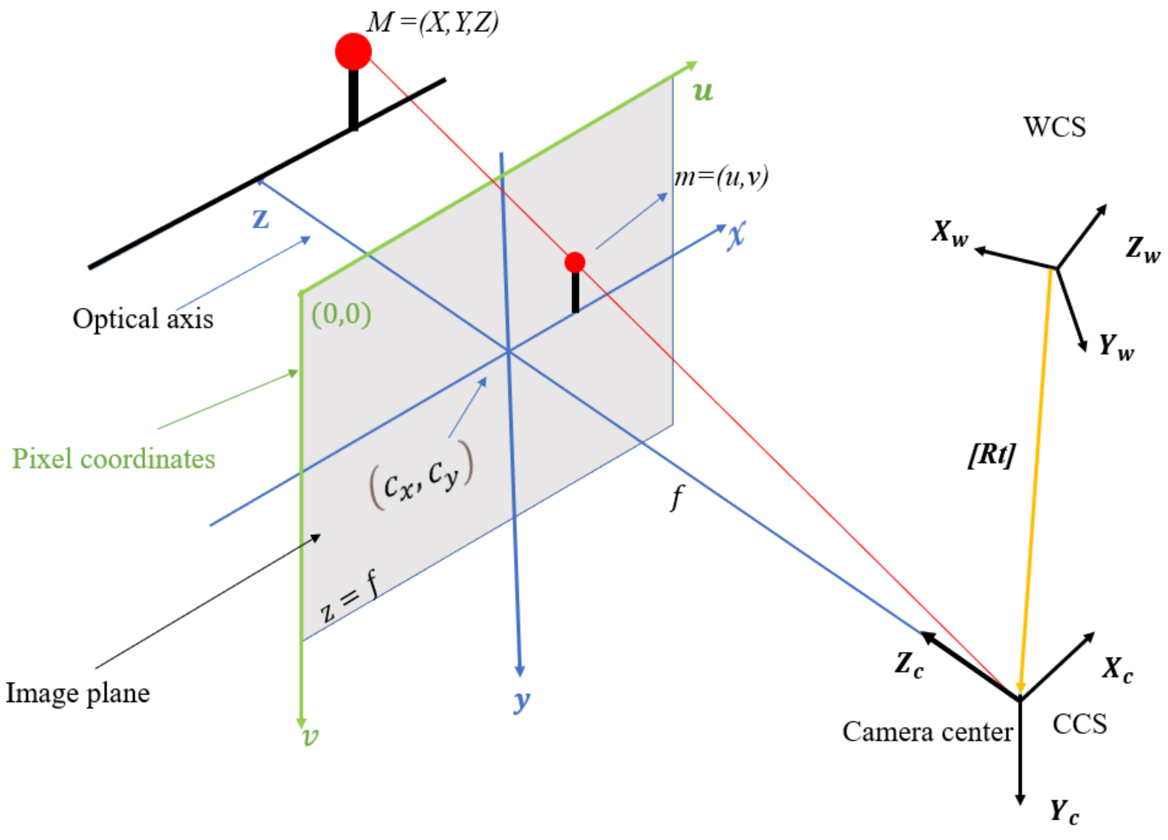


Figure 3.1 General pinhole model: A 3D point $\boldsymbol{M} = [X, Y, Z]^{\mathrm{T}}$ in world coordinates (WCS) is projected through the camera center onto the image plane at pixel coordinates $\boldsymbol{m} = [u, v]^{\mathrm{T}}$, the principal point of the image plane is at (c_x, c_y) while the focal length is represented by f [25]

Two parameter groups can be described, the intrinsic parameters and the extrinsic parameters:

3.1.1 Intrinsic parameters:

- Focal length (f) is the distance between the optical center and the image plane, and usually expressed in millimeters or pixels.
- Principal point (c_x, c_y) refers to the intersection of the image plane and the optical axis. The ideal center of the camera's field of view is represented by this point .

Together they form the intrinsic matrix ***K*** of a pinhole model used in the projection matrix, to map 3D point into 2D pixel coordinates point.

$$\boldsymbol{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

A distortion factor in the lens of a camera, is usually represented by a distortion vector, as in the equation below:

$$D = (k_1, k_2, p_1, p_2, k_3) \tag{3.2}$$

1. **Radial distortion coefficients** (k_1, k_2 , k_3), indicate how much the camera lens' radial distortion causes straight lines to appear curved (fisheye effect).

   1. Primary correction (k_1) manages the largest distortions called barrel or pincushion distortion.
   2. Secondary correction,or (k_2) applyies additional fine-tuning improves the accuracy of the correction across the entire image, with the most noticeable improvements occurring in the areas away from the center.
   3. The tertiary correction,or (k_3) is mostly required for lenses with more intricate distortion patterns and is used for extremely precise adjustments.

2. **Tangential distortion coefficients** (p_1, p_2) quantify the misalignment between the lens and the image plane, giving the impression that images are distorted, slanted, closer, or farther away than they are. The main types of distortions experienced by a camera are illustrated in Figure 3.2.

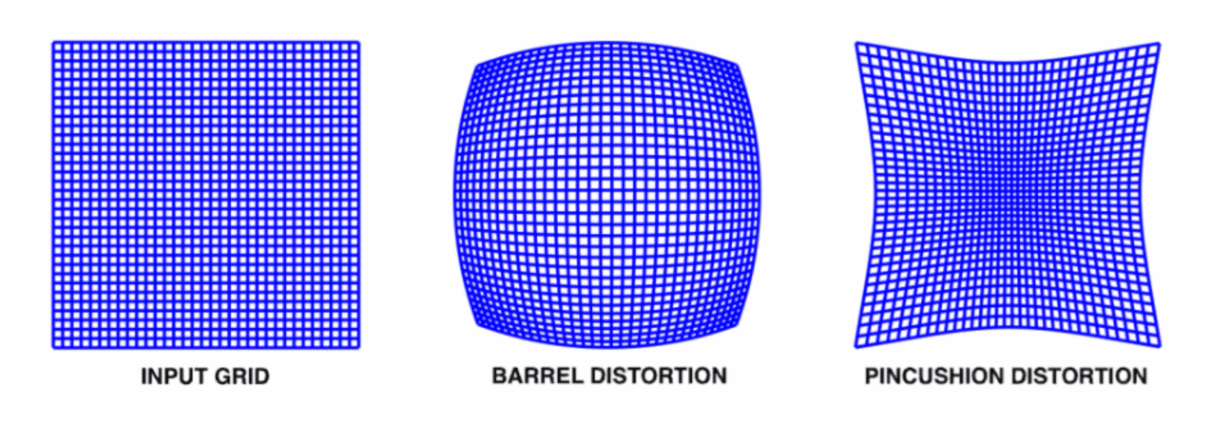


Figure 3.2 Types of distortion managed by k_1 in a camera

3.1.2 Extrinsic parameters:

A distinct Euclidean coordinate frame, referred to as the world coordinate frame, is typically used to express points in space. Rotation and translation are used to link the two coordinate frames, camera, and world.

Extrinsic parameters are illustrated in Figure 3.3.

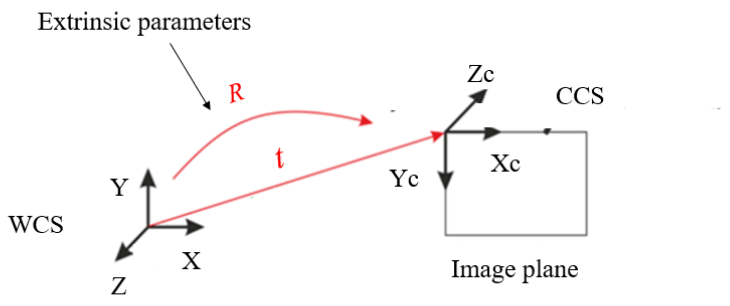


Figure 3.3 The transformation from the world coordinate system (WCS) to the camera coordinate system (CCS) is defined by the extrinsic parameters transformation $[\boldsymbol{Rt}]$ which aligns the world frame $(\mathrm{X}, \mathrm{Y}, \mathrm{Z})$ with the camera frame $(\mathrm{X_c}, \mathrm{Y_c}, \mathrm{Z_c})$, allowing the projection of 3D points onto the image plane [23]

The extrinsic matrix as illustrated in Figure 3.3 is composed of rotation and translation component.

$$[\boldsymbol{Rt}] = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \tag{3.3}$$

The combination of the intrinsic and extrinsic matrix is called the Projection matrix denoted as P is used to map a 3D point $\boldsymbol{M} = [X, Y, Z, 1]^{\mathrm{T}}$to its corresponding 2D pixel or image coordinates $\boldsymbol{m} = [u, v, 1]^{\mathrm{T}}$ both expressed in homogenous coordinates.

$$\mathrm{P} = \boldsymbol{K}[\boldsymbol{Rt}] \tag{3.4}$$

$$\boldsymbol{m} = \mathrm{P}\boldsymbol{M} \tag{3.5}$$

$$\boldsymbol{m} = \boldsymbol{K}[\boldsymbol{Rt}]\boldsymbol{M} \tag{3.6}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.7}$$

Figure 3.4 illustrates the process of converting a 3D world point into the 2D image pixel coordinates system.

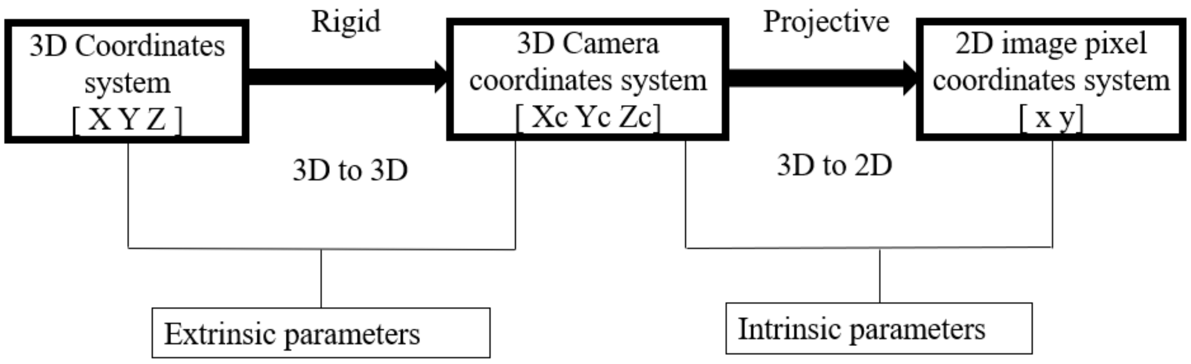

3D Coordinates system [X Y Z]
Rigid
3D Camera coordinates system [Xc Yc Zc]
Projective
2D image pixel coordinates system [x y]
3D to 3D
3D to 2D
Extrinsic parameters
Intrinsic parameters

Figure 3.4 The process conversion of 3D world point in 2D images coordinates [27]

The process of finding the intrinsic and extrinsic parameters, and removing distortion is called camera calibration, which will be discussed in the next section.

3.2 Camera calibration

3.2.1 Theory

Camera calibration is a necessary step in 3D computer vision to extract metric information from 2D images [28]. These camera parameters are used to remove lens distortion effects from an image, measure planar objects, reconstruct 3-D scenes from multiple cameras, and perform many other computers vision tasks.

The goal of the calibration process is to find the 3×3 intrinsic matrix, $\boldsymbol{K}$, the 3×3 rotation matrix, and the 3×1 translation vector using a set of known 3D points $\boldsymbol{M} = [X, Y, Z]^{\mathrm{T}}$and corresponding image coordinates $\boldsymbol{m} = [u, v]^{\mathrm{T}}$.

3.2.2 Types

Photogrammetric calibration: This calibration is performed by observing a calibration object whose geometry in 3-D space is known with sufficient precision [29]. The calibration object usually consists of two or three planes orthogonal to each other. Sometimes, a plane undergoing a precisely known translation is also used as described in [30]. This type of calibration requires an expensive calibration device and a complex setup.

Self-calibration techniques: In this category we do not use any calibration object. Just by moving a camera in a static scene, the rigidity of the scene provides in general two constraints on the camera's internal parameters from one camera displacement by using image information alone [31].

Other methods for camera calibration can be used, such as vanishing points for orthogonal directions [30] and calibration from pure rotation [32].

3.2.3 Zhang calibration method

The calibration technique developed by Zhang [28, 33, 34] will be used because it is simple and reliable since after its development it was implemented as a MATLAB toolbox.

The camera takes images of a planar pattern in a few distinct orientations. If the pixels are square, the minimum number of orientations is two; nevertheless, for optimal quality, it is suggested to use four or five alternative orientations. Either the planar pattern or the camera can be moved by hand. The technique lies between photogrammetric calibration and self-calibration, while it is not necessary to know the motion. The planar pattern should not be positioned parallel to the image plane when there are just two orientations. For instance, we could use a laser printer to print a pattern and then affix the paper to a suitably flat surface, like a hard cover book envelope. The printed pattern presents the disadvantage of having a dimensional error between the actual size and the printed size of the square of 0.1 mm (measured during a laboratory experiment with a traveling microscope on a printed pattern). Thus, for more accuracy it is recommended to buy from a specialized manufacturer like Calib.io [35], specialized in producing camera calibration patterns which can achieve patterns with dimensional accuracy around 2 μm.

3.2.4 Considering distortions coefficient

In a camera model, the distortion function is likely dominated by the radial components which is partly due to lens distortion [36]. The strategy used by Zhang is to estimate k_1 and k_2 after having estimated the other parameters such as the rotation and translation components. Once k_1 and k_2 (coefficients of radial distortion) are obtained the camera calibration including lens distortion can be performed by minimizing the reprojection error, which is the distance between the observed image points and their predicted positions based on the estimated parameters [33].

In summary the steps used for the calibration are highlighted below:

- Create and print a pattern that will be applied on a level surface. This may be done simply with the built-in capabilities in MATLAB such as opencheckerboardPattern.pdf.
- Move the camera or the patterned surface to take many pictures of it from 10 to 20 viewpoints or locations This usually entails snapping ten to twenty pictures. And find or identify feature points in every picture that has been taken.
- Estimate the camera's five intrinsic characteristics and all extrinsic parameters that specify its location in relation to the patterned surface.
- Determine the radial distortion coefficients and adjust each parameter until convergence is achieved, guaranteeing precise camera system calibration. This is all done using the camera calibrator app.

3.2.5 The mean reprojection error

The mean reprojection error is a geometric error corresponding to the image distance between a projected point and a measured one. It is used to quantify how closely an estimate of a 3D point recreates the point's true projection as illustrated in Figure 3.5.

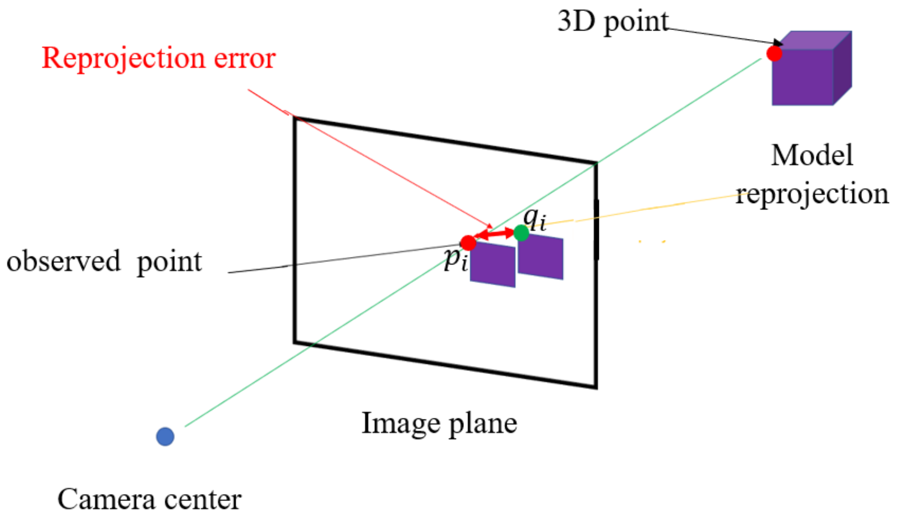


Figure 3.5 Schematic representation of the reprojection error

The calibration algorithm's performance is gauged by the reprojection error, an error metric that is independent of the camera and setup.

The 3D locations of all detected feature points are determined after having modelled (calibrated) the imaging equipment. We take the internal 3D model of all feature points and project them back into the scene using the camera and distortion models, which results in a collection of feature point model predictions. The reprojection error decreases with increasing model prediction accuracy between the feature points and the related data.

We calculate the reprojection error as the mean L2 norm of point correspondence errors where the error denoted by e is given by:

$$e = \frac{1}{N}\sum_{i=0}^{N-1} |p_i - q_i|_2 \tag{3.8}$$

The observed feature points on the image plane are denoted by p_i and the anticipated image plane positions by q_i.

While using the MATLAB toolbox, this error is given already as an output without the need to perform any calculations.

CHAPTER 4 FEATURE EXTRACTION AND ESTIMATION OF SPHERE CENTER LINE OF SIGHT

4.1 Edge detection

By joining the groups of pixels on the boundary between two distinct regions in an image, the edges can be generated if there are noticeable local variations in intensity at a particular spot in the image. A local maximum at an edge point is taken into consideration using the image's first derivative. The intensity change at a specific edge point is measured using the gradient magnitude [37]. Edge detection methods are divided into two categories: Laplacian and gradient [38].

4.1.1 Used filter to perform edge detection

Two main filters are used in images processing: Gaussian and Laplacian.

The Gaussian smoothing operator, or filter, removes Gaussian noise by performing a weighted average of surrounding pixels based on the Gaussian distribution.

The Laplacian filter belongs to the derivative filter class. In image processing, this second-order filter is used for feature extraction and edge detection. First-order derivative filters combine both vertical and horizontal edges by applying distinct filters for each. The Laplacian filter, in contrast, can identify any edge, regardless of its orientation.

4.1.2 Types of edge detector

The main types of edge detection techniques used in image processing are grouped into two categories highlighted in Figure 4.1 :

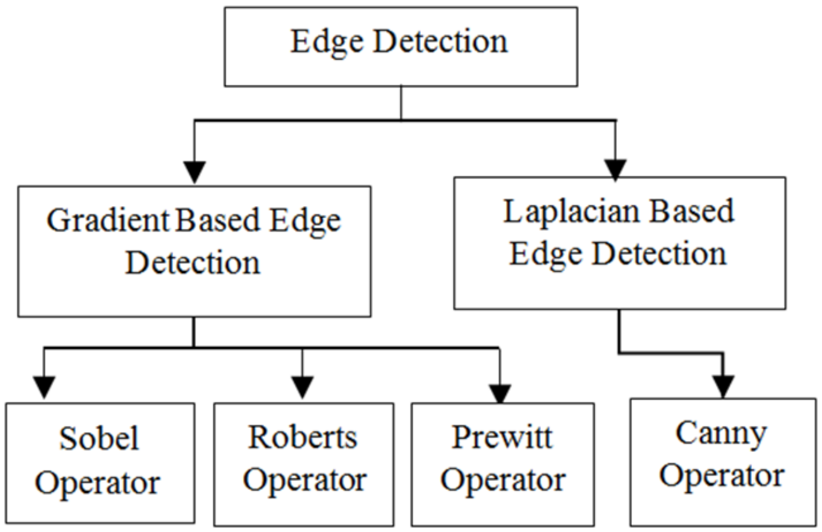


Figure 4.1 Types of edge detectors [39]

4.1.3 Gradient based

1. Sobel operator

Sobel edge detection is a traditional method in image processing, particularly well-suited for detecting edges along the vertical and horizontal axes [38].

2. Robert’s operator

Robert’s operator approximates the gradient of an image through discrete differentiation. The Roberts operator is renowned for having extremely high edge detection precision, but it is also highly noise sensitive. As such, it works well for image segmentation in which noise is low and edges are clearly defined [40].

3. Prewitt operator

The Prewitt operator is a discrete differentiation operator used to approximate derivative values in both the horizontal and vertical directions [41]. It possesses some degree of noise suppression, achieved through pixel averaging. However, the Prewitt operator may produce edges that are several pixels wide and is generally less accurate in edge detection compared to the Roberts operator.

4.1.4 Laplacian based

1. **Laplacian of Gaussian (LOG)**

In this method, Gaussian filtering is combined with a Laplacian filter to break down the image where the intensity varies to detect the edges effectively [42].

2. **Canny Edge detector**

An intricate technique called Canny Edge detection uses multiple stages to identify various edges present in an image. This detector locates edges using the derivative of a Gaussian filter. The method uses two thresholds to detect strong and weak edges and includes the weak edges in the output only if they are connected to strong edges [37].

4.1.5 Resume on edge detectors techniques

One of the main problems of gradient-based algorithms, like the Prewitt filter, is that they are highly susceptible to noise. The coefficients and kernel filter sizes are fixed and cannot be changed to fit a specific image. An adaptive edge-detection algorithm is necessary to provide a robust solution that is adaptable to the varying noise levels of these images to help distinguish valid image contents from visual artifacts introduced by noise [42].

Canny’s edge detection algorithm is computationally more expensive compared to Sobel, Prewitt, and Robert’s operator. However, Canny’s edge detection algorithm performs better than all these operators under almost all scenarios [40]. A summary on edge detector techniques [40] is presented in Table 4.1.

Table 4.1 Edge detectors summarized

Operator	Advantages	Disadvantages
Sobel, Prewitt	- Simple to implement. - Recognize edges and orientation effectively.	- Inaccurate under certain conditions. - Highly sensitive to noise.
Laplacian second directional derivative	- Provides consistent properties in all directions. - Effectives on identifying edges.	- Very sensitive to noise, which may affect accuracy.
Laplacian of Gaussian	- Accurately determines edges locations.	- Performs poorly on curves, corners, or areas with fluctuating gray levels.
Canny	- Excellent edge detection, even in noisy environments. - Superior SNR ratio.	- Computationally complex and time-consuming.

4.1.6 Experimental test of different edge detectors

Figure 4.2 details the comparative evaluation of various edge detection algorithms applied to a 1-inch (25.4 mm) ceramic precision sphere.

Under standard lighting conditions, the Canny edge detector exhibits superior contour definition relative to other methods. The results and image were obtained through experiments conducted by the author.

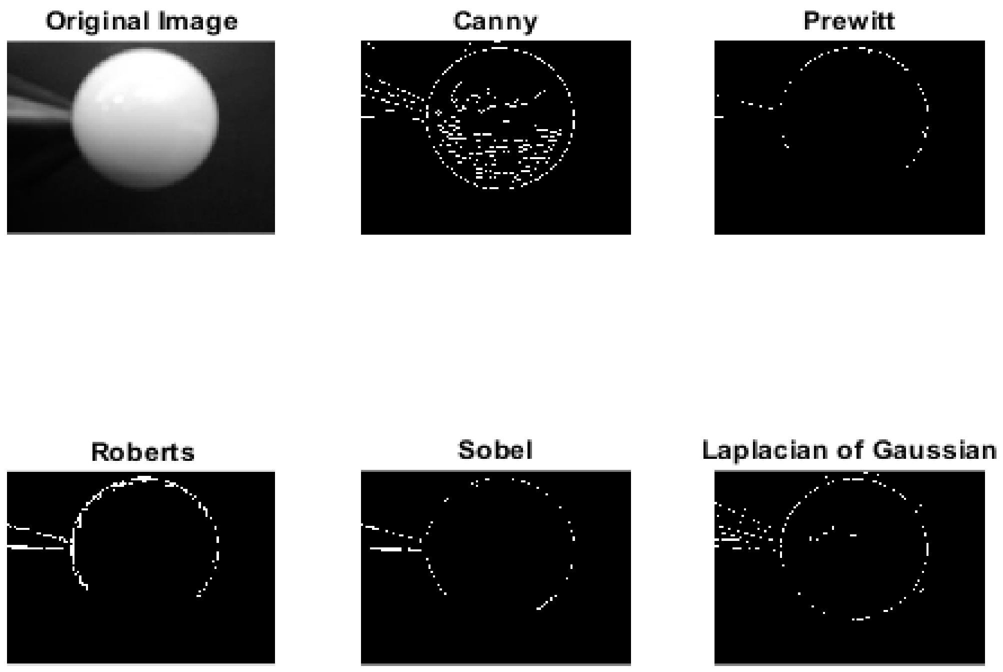


Figure 4.2 Performance results of different edge detectors

4.2 Ellipse fit

4.2.1 Overview

Ellipse fitting is a mathematical process used to find the best-fitting ellipse to a given set of data points, typically in a two-dimensional plane. This technique is commonly applied in computer vision, image processing, and pattern recognition to model shapes that resemble ellipses, such as orbits, cell contours, or object boundaries [43].

The goal of ellipse fitting is to determine the parameters of the ellipse (center coordinates, major and minor axes lengths, and orientation) that minimize the difference between the data points and the ellipse curve. There are two main types of ellipse fitting methods: direct fit and iterative fit. Direct fitting uses algebraic solutions for quick approximations, while iterative fitting refines the parameters by minimizing the geometric distance for higher accuracy [44]

4.2.2 Ellipse fitting methods

Several articles [45], [46], [47], [48] have focused on the issue of recovering ellipses. Although Bookstein's method has been frequently used in the past ten years, it does not specifically enforce the fitting to be an ellipse, meaning the algorithm can output a hyperbola or a parabola, even from an elliptical input. Also, Porrill [46] used Bookstein's approach to initialize a Kalman filter whose role is to effectively estimate a dynamic system's state from noisy observations. By checking the discriminant $b^2 - 4ac < 0$ at each iteration, the Kalman filter iteratively minimizes the gradient distance to obtain fresh picture evidence and reject non-ellipse fits. In addition to using a Kalman filter, Rosin [49] reiterates that ellipse-specific fitting is a non-linear problem requiring the use of iterative techniques. Additionally, in [49] the advantages and disadvantages of two widely used normalizations, f=1 and a+c =1, are examined showing that the first method tends to produce ellipses that are less elongated, which makes it harder to keep the same shape when the data is changed (like being rotated or resized). As a result, it makes it more likely to get an ellipse as the final shape.

- **Direct least squares fit**

A direct least squares method for ellipse fitting based on the algebraic distance metric was proposed by A. Fitzgibbon [45], who also developed a MATLAB implementation later enhanced in [50] to improve numerical stability and robustness. This method guarantees an elliptical fit even with a limited number of noisy data points and it is non-iterative, avoiding issues like local minima and numerical instability.

However, because the method minimizes algebraic distances rather than geometric distances, the solutions tend to be biased toward smaller ellipses. The algebraic distance "prefers" the points lying inside an ellipse; thus, the algorithm tends to produce ellipses smaller than they should be.

- **Iterative least squares fit**

Accurately fitting an ellipse to a set of data points is required in many applications that demand exact geometric measurements, including calibration, object tracking, and machine vision. When dealing with noisy or imperfect data, the iterative ellipse fitting method is a reliable way to get high-precision ellipse parameters. This technique minimizes the difference between the computed elliptical curve and the observed locations by gradually adjusting the ellipse parameters through an iterative optimization procedure.

- **Overview of the iterative fitting method**

An initial estimate of the ellipse parameters, such as the center coordinates, semi-major and semi-minor axis lengths, and orientation, is the first step in the iterative process. The residuals, or the distance between the observed data points and their closest spots on the candidate ellipse, are then calculated. The ellipse parameters are improved in each step to converge towards an ideal fit by iteratively minimizing these residuals [46].

Nonlinear least-squares minimization is a popular method in iterative elliptical fitting, where the sum of squared residuals is the objective function to be reduced. Using optimization techniques like gradient descent or the Levenberg-Marquardt algorithm [48,49], the method gradually improves the fit by lowering the overall residual error after adjusting parameters based on an initial guess.

- **Advantages of the iterative fitting**

Iterative fitting techniques are especially useful when there are outliers or high noise levels because they can more effectively account for data points that deviate from ideal values. Compared to direct methods, these techniques produce more accurate and consistent results by improving the fit over multiple iterations, which reduces the likelihood of noise-induced mistakes. Therefore, iterative fitting is frequently used in applications like metrology, medical imaging, and machine tool calibration where high precision is essential.

Thus, for optimum results the iterative fitting method is the best choice over the direct. The algorithm used for this project was developed by Hu ma [51]. A simple ellipse fit was performed on the selected precision sphere as illustrated in Figure 4.3. The method used is the iterative least square technique, which allows an accurate fit even in noisy conditions.

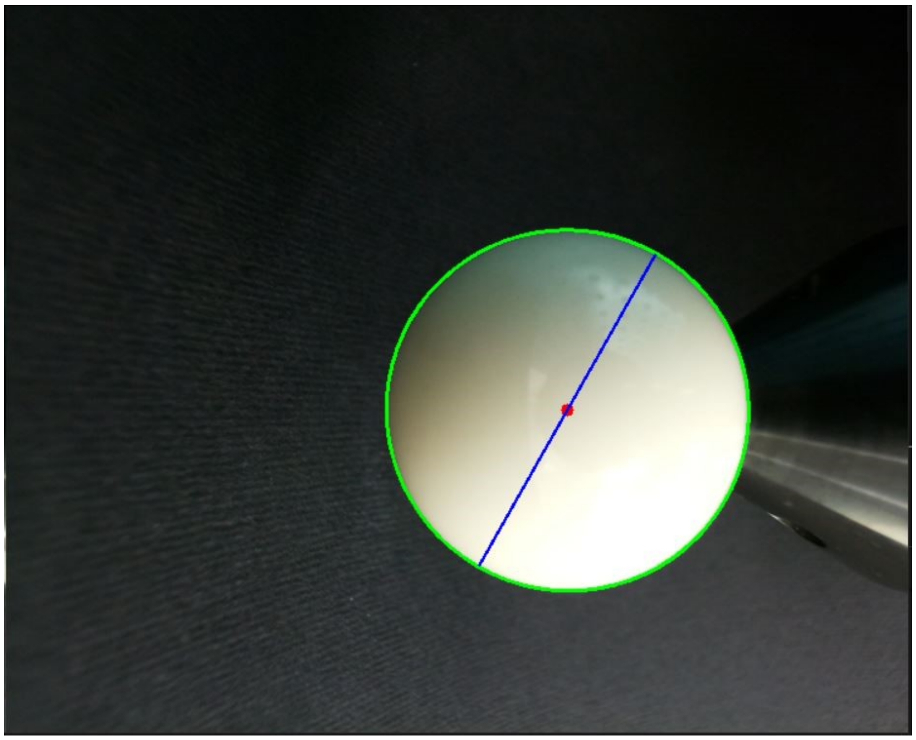


Figure 4.3 Image of the reference sphere captured by the vision system. The green contour represents the fitted ellipse using an iterative method, the blue line indicates the estimated major axis direction, and the red dot marks the computed center of the ellipse

4.3 Back projection of 2D pixel points to 3D rays

To compute the line of sight through the center of the precision sphere it is important to understand the image processing concept called, the back projection of pixel point to 3D rays. The 3D ray computed is a representation of the line that goes through the 3D point and its corresponding 2D images coordinates.

Let’s recall that Zisserman [25] modelled the pinhole camera as illustrated in Figure 4.4.

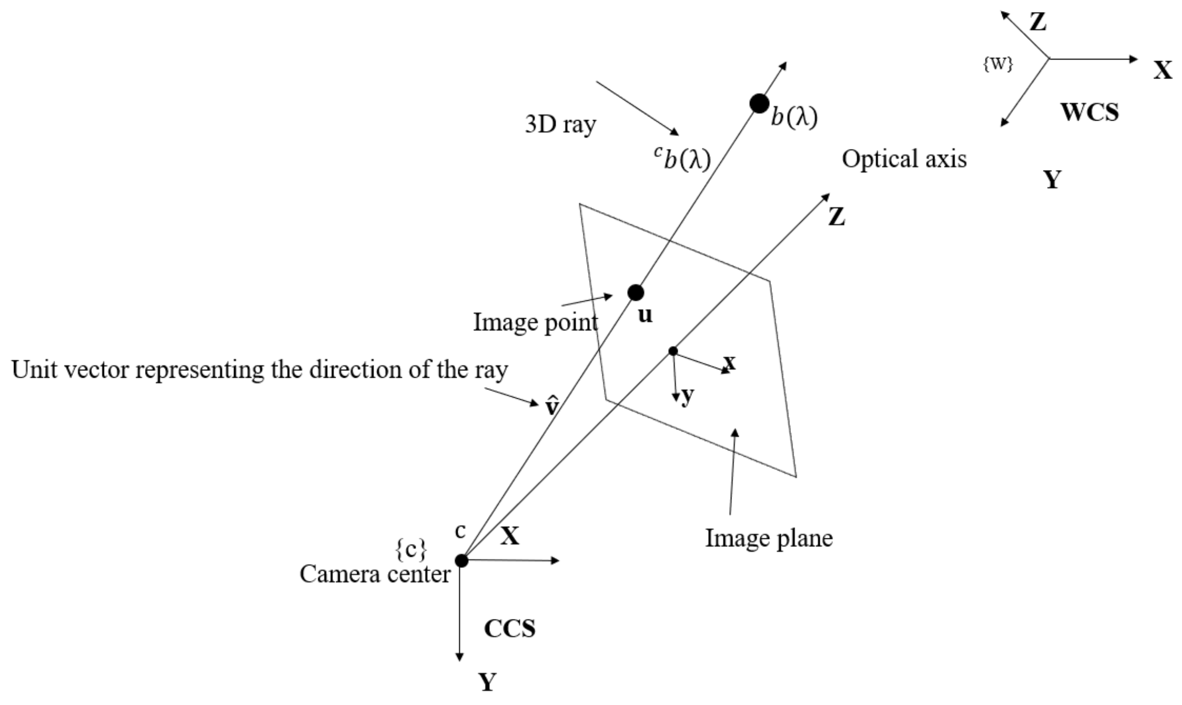


Figure 4.4 Back-projection of a 2D image point onto a 3D ray in the camera coordinate system (CCS). The image point **u** lies on the image plane, and its corresponding 3D ray originates from the camera center c and follows the direction of unit vector $\hat{\mathbf{v}}$. The ray is parameterized by $\boldsymbol{\lambda}$, resulting in 3D points $b(\lambda)$ along its path. The world coordinate system (WCS) is shown for reference using the simple camera pinhole model [25]

A ray equation describes how a 3D point in space is related to the camera center and the corresponding image point. From Figure 4.4, the equation of a ray in the camera coordinate system can be computed as:

$$ {}^{\{c\}}b(\lambda) = \lambda\,{}^{\{c\}}\mathbf{v}\,,\ \lambda \in [0\,,\infty) \qquad (4.1) $$

$b(\lambda)$: represents a set of points along a ray in 3D space.

λ : is an unknown scaling factor that determines how far we are along the ray.

${}^{\{c\}}\mathbf{v}$: is the unit direction vector of the ray in the camera frame.

The value of λ determines the depth of a 3D point along the ray.

Thus, an equation to model ${}^{\{c\}}\mathbf{v}$ is:

$$ {}^{\{c\}}\mathbf{v} = \frac{\boldsymbol{K}^{-1}\mathbf{u}}{\|\boldsymbol{K}^{-1}\mathbf{u}\|} \tag{4.2} $$

where;

$\boldsymbol{K}$ is the intrinsic matrix of the camera.

$\mathbf{u} = (u_1, u_2, u_3)^{\mathrm{T}}$ is the image point in homogenous coordinates.

$\boldsymbol{K}^{-1}\mathbf{u}$ converts the 2D point $\mathbf{u}$ into a **3**D direction vector in the camera frame.

The division by the norm ensures ${}^{\{c\}}\mathbf{v}$ is a unit vector.

4.4 Determination of the Line of Sight

The projection of a sphere in the image plane is an ellipse, with its special case being a circle [52]. A sphere's projection on the image plane is a circle only at the center of the image plane and is otherwise typically an ellipse due to perspective distortion [53], this entails that the center of the fitted ellipse is not coincident with the sphere center. The approach used in the thesis to determine the cone axis from a spherical projection using the fitted ellipse on the image plane was developed by T. Tekla and H. Levente [54]. The cone axis recovered passes through the center of the observed spherical object.

A simple projection of a sphere unto the image plane is illustrated in Figure 4.5.

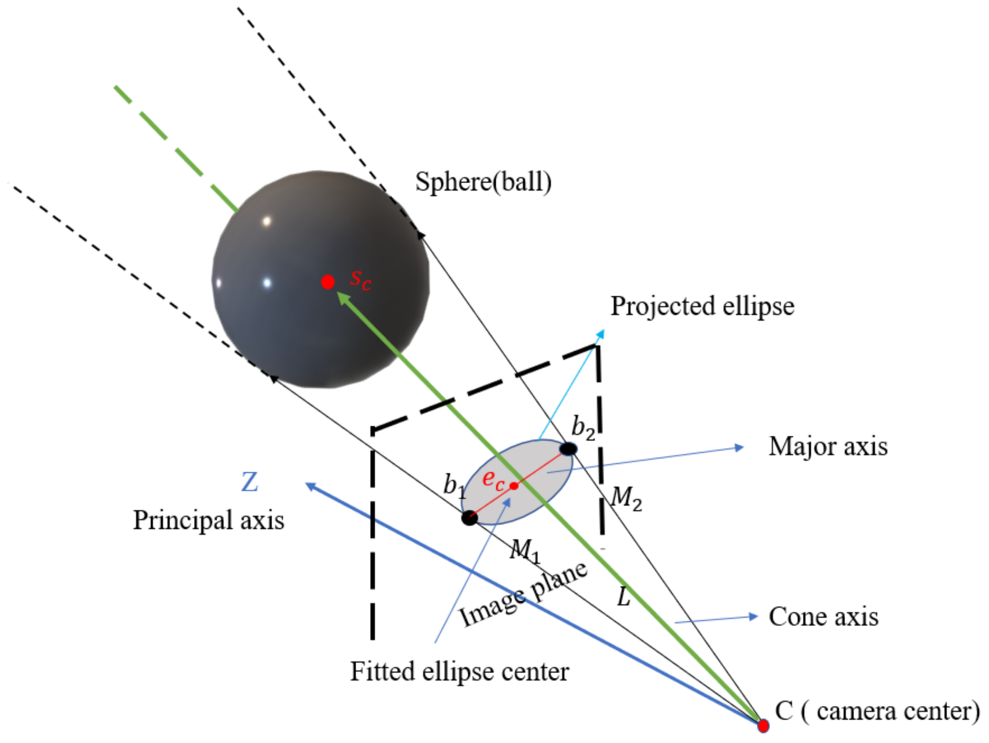


Figure 4.5 Visualization of the sphere back-projection geometry. The cone axis from the camera center through the ellipse major axis helps recover the sphere's center in 3D space [54]

To recover the cone axis, first we estimate the endpoints of the major axis $b_1$and b_2.

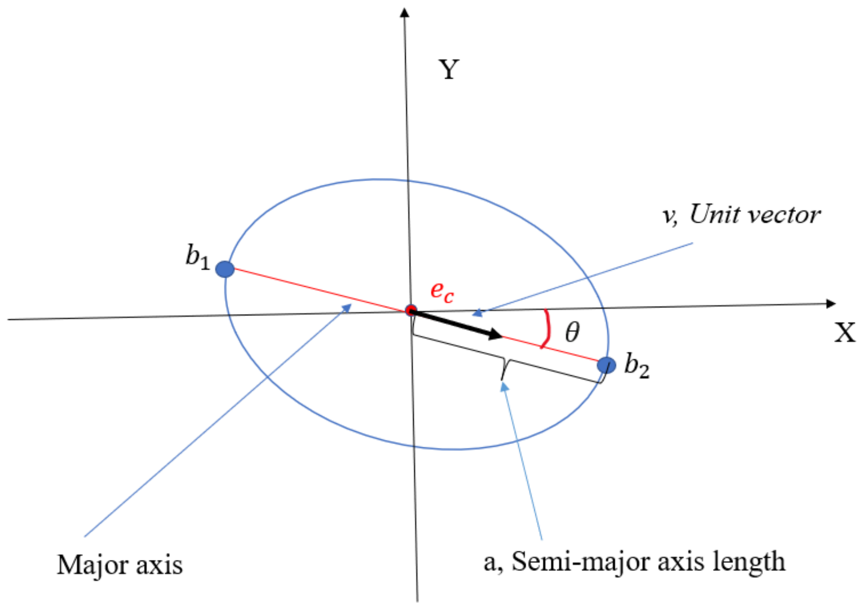


Figure 4.6 2D representation of the projected ellipse showcasing the major ellipse parameters

Given the ellipse parameters in Figure 4.6**:**

- Center: $e_c = \begin{bmatrix} x_c \\ y_c \end{bmatrix}$
- Semi-major axis length: a
- Orientation angle: θ

The major axis direction is given by a unit vector v pointing in the direction of the major axis of the ellipse. Here is why this is the case:

- θ: The orientation angle θ is the angle between the positive x-axis and the major axis of the ellipse.
- v: A unit vector in the direction of an angle θ is given by v= $\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix}$
- a: The major axis endpoints are located at a distance a from the center along the major axis direction. Let's denote the major axis endpoints as b_1 and b_2. These endpoints are given by:

$$b_1 = e_c + a\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix} \tag{4.3}$$

$$b_2 = e_c - a\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix} \tag{4.4}$$

The next step is to find the direction of the cone axis. We back-project the points b_1 and b_2 into the 3D ray in space, these rays are denoted M_1 and M_2 and their unit vector in the camera coordinates can be obtained as:

$$M_1 = \frac{\boldsymbol{K}^{-1}b_1}{\|\boldsymbol{K}^{-1}b_1\|} \tag{4.5}$$

$$M_2 = \frac{\boldsymbol{K}^{-1}b_2}{\|\boldsymbol{K}^{-1}b_2\|} \tag{4.6}$$

The angle between M_1 and M_2 is the angle of the cone and the angle bisector of these vectors give the direction as shown in Figure 4.7:

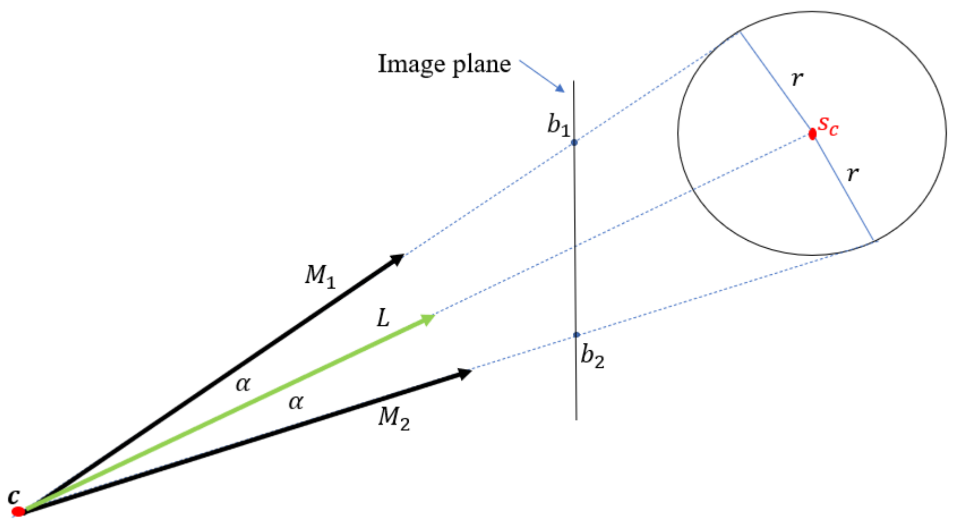


Figure 4.7 Vector projection of the fitted ellipse

Thus *L,* representing the cone axis can be found as:

$$L = \frac{M_1 + M_2}{2} \tag{4.7}$$

4.4.1 Impact of Major/Minor Axis orientation accuracy on global results

1. Propagation of angula error

If the measured major axis is off by a small angle $\Delta\theta$, this translates into a directional error in the estimated 3D ray. The larger the sphere-camera distance, the greater the 3D positional deviation becomes.

The effect is nonlinear: small angular errors can lead to millimeter-scale or worse positional errors, depending on system geometry and baseline.

2. Impact on global results

The overall global results likely involve multiple views or sphere positions. If each cone axis is slightly misoriented this affects the final residuals / RMSE reconstructed LOS and measured 3D sphere center points as shown in Figure 4.8.

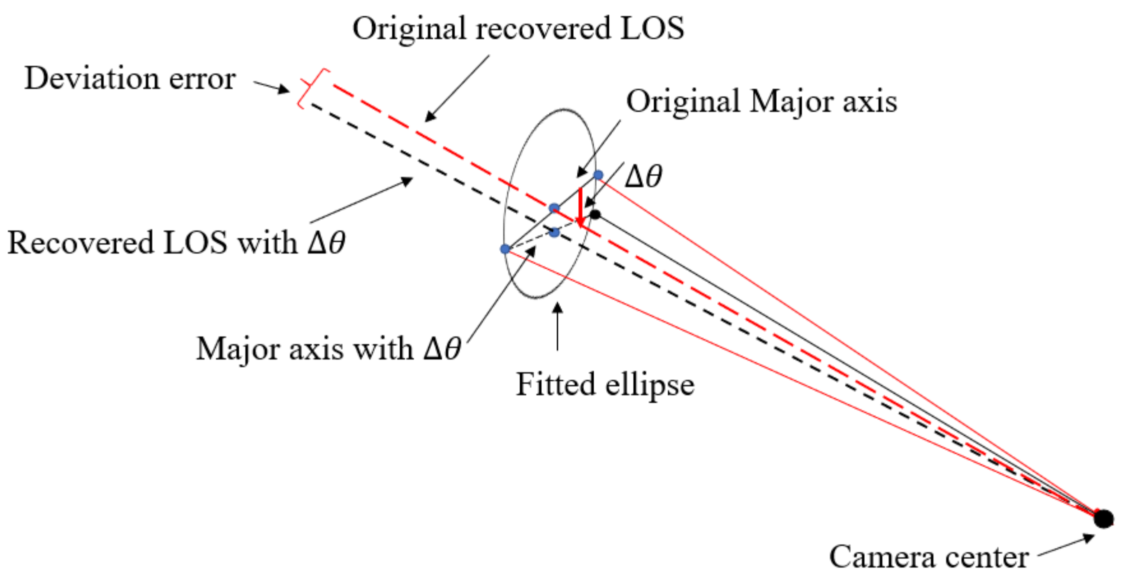


Figure 4.8 Illustration of how a measurement error in the orientation angle of the ellipse's major axis affects the direction of the cone axis formed by the corresponding vectors

4.4.2 Process pseudo-code

A simple pseucode illustrating all various steps invole in computing the cone axes in the camera coordinates are explained in the following lines :

Algorithm Process Ellipses and compute cone axes

```
1  **Input**: Set of images, camera intrinsic matrix $\boldsymbol{K}$
2  **Output**: 3D cone axes and saved ellipses data
3  **Procedure**: ProcessAndBackProjectEndPoints (input: images, intrinsic parameters)
4   Compute the intrinsic matrix $\boldsymbol{K}$ , use camera focal lengths and principal points. Refers to
    equation (3.1)
5   Compute the inverse of the intrinsic matrix $\boldsymbol{K}^{-1}$
6  **Initialize** data storage for ellipses and cone axes
7  **for** each image in $i$ **do**
8          FitEllipseFromFile: Fit ellipse to the images
9                  **if** ellipse fitting is successful, then
10                   Compute semi-major axis endpoints $b_1 = (x_1, x_2)$ ; $b_2$=$(x_{1,} x_2)$
11                   $b_1$= $((x_0 + a\ \cos(\theta), y_0 + a\sin(\theta))$
12                   $b_2$= $((x_0 - a\ \cos(\theta), y_0 - a\sin(\theta))$
13                   Back project endpoints to 3D rays using $\boldsymbol{K}^{-1}$
14                   Refer to equation (4.8) and (4.9)
15                   Compute the cone axis as the average of the two 3D rays
16                   Refer to equation (4.10)
17                   Plot cone axes and store data
18                 **else**
19                  **Skip** to next iteration
20                 **end if**
21 **end for**
   **end Procedure 1**
22 Save ellipses data and cone axes vectors into an Excel file
23 **Procedure 2** FitEllipseFromFile (input: imageFile)
24                 Read images and detect edges
25                 Fit an ellipse to detected points
26                 Return ellipse parameters $(x_0, y_{0.}, a, \theta)$
   **end Procedure 2**
```

CHAPTER 5 QUANTITATIVE ERROR ANALYSIS ALGORITHM FOR SPHERE CENTERS RELATIVE TO LINES OF SIGHT

An algorithm is developed to quantify the ability of a single camera to measure the center of the reference sphere as described in Figure 5.1.

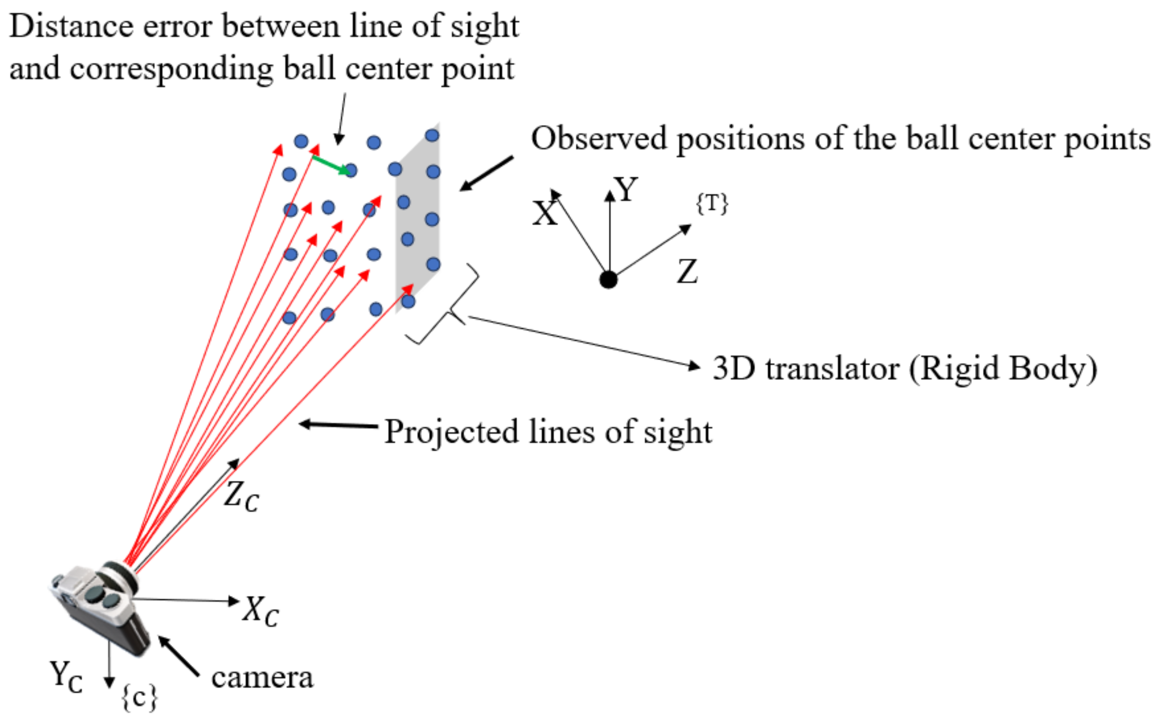

Distance error between line of sight and corresponding ball center point
Observed positions of the ball center points
X
Y
{T}
Z
3D translator (Rigid Body)
Projected lines of sight
Z_C
X_C
Y_C
{c}
camera

Figure 5.1 Illustration of the distance error between a line of sight and the known positions of ball center points and the two-frame camera as c and translator T

A more detailed description of the 3D space relationship between line of sight and the distance error measured is shown in Figure 5.2.

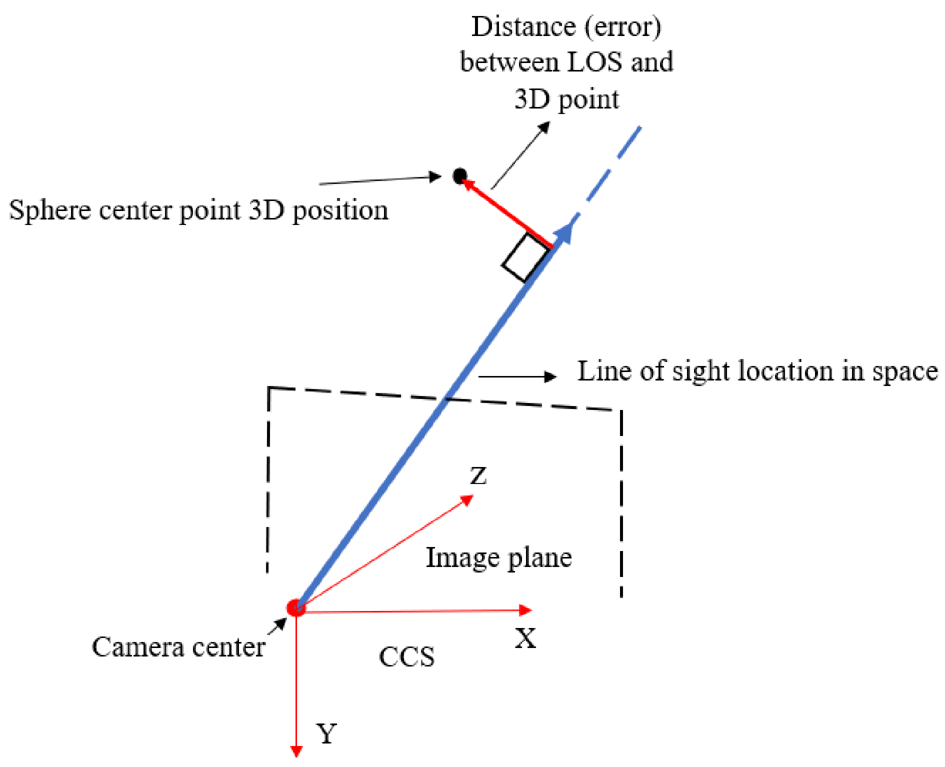

Distance (error) between LOS and 3D point
Sphere center point 3D position
Line of sight location in space
Z
Image plane
Camera center
CCS
X
Y

Figure 5.2 Description of the 3D space relationship between line of sight and the distance error measured

5.1 Rigid body adjustment fundamentals

In this study, rigid body adjustment principles are applied to align the lines of sight with their corresponding sphere center points by minimizing the initial measured distance. The key principles of rigid body transformation are as follows:

• A **rotation** applied to any point on a rigid body result in the same rotation at all other points of the body.

• A **translation** applied to a point on a rigid body leads to:

1. An equal translation at all points of the body.
2. No induced rotation anywhere on the body.

• A **rotation** at one point of a rigid body:

1. Can induce a translation at another point. For small rotations, this translation is given by the cross product of the rotation vector and the relative position vector from the rotation point to the point of interest.
2. Produces the same rotational effect at all points of the rigid body.

In this context, the rigid body transformation is used to adjust the ideal geometric feature (IGF) representing the sphere center points positions, ensuring its alignment with the corresponding line of sight (LOS) as represented in Figure 5.3. This transformation consists of an optimal rotation and translation that minimizes the distance between the measured LOS and their corresponding sphere center points.

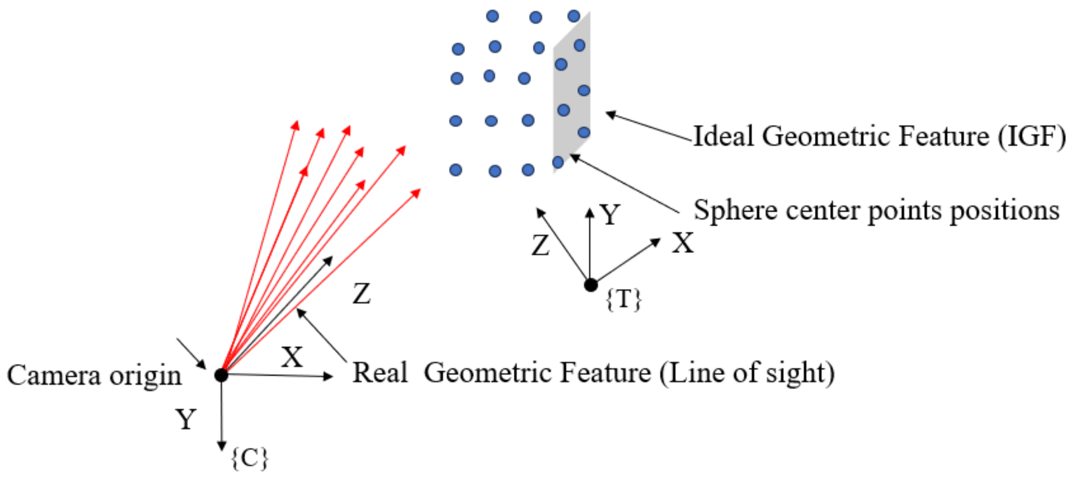

Ideal Geometric Feature (IGF)
Sphere center points positions
Y
Z
X
{T}
Z
Camera origin
X
Real Geometric Feature (Line of sight)
Y
{C}

Figure 5.3 Illustration of the ideal geometric feature (IGF) which represents the 3D translator and the real geometric feature which are the LOS

A rigid body can be represented as an association between an ideal geometric feature and a set of points. This approach, initially developed by Pierre Bourdet for part metrology, is adapted here as shown in Figure 5.4, for the calibration of machine tools using a 3D optical sensor. Instead of evaluating the geometric conformity of a manufactured part, we apply the same principles to refine the spatial alignment of the LOS relative to the known reference sphere positions.

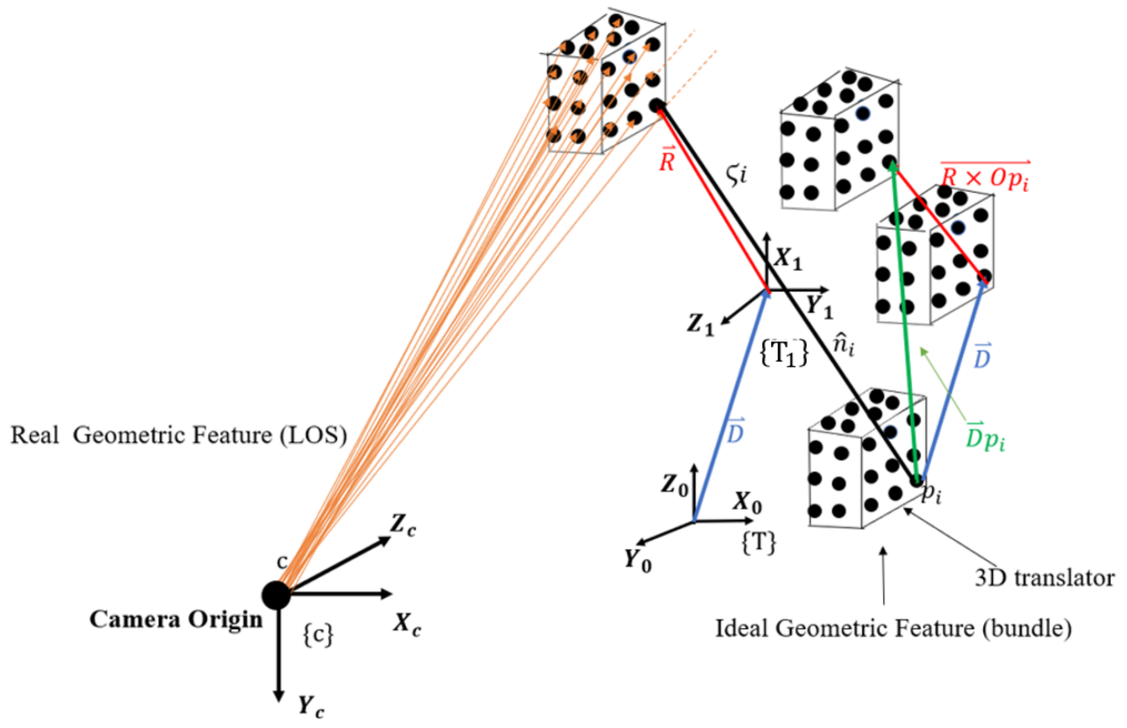


Figure 5.4 Illustration of the rotation and translation of the IGF (translator frame) to which are attached the ball centers, and the LOS attached to the camera frame showcasing how they relate to each other.

where;

with a general equation for fit adjustment given by:

$$e_i = \varsigma i - {}^{\{T\}}\vec{D}p_i \bullet {}^{\{T\}}\hat{n}_i \tag{5.1}$$

where;

${}^{\{T\}}\vec{D}p_i$: ${}^{\{T\}}\vec{D} + {}^{\{T\}}\vec{R} \times {}^{\{T\},\{T\}}p_i$

ςi : perpendicular distance between the LOS and the ball center ${}^{\{T\},T}p_i$ to minimize.

e_i : distance after micro-adjustment.

${}^{\{T\}}\vec{D}$: translation of the IGF.

${}^{\{T\}}\vec{R}$: small rotation of the IGF.

and

$$ {}^{\{T\}}\vec{D} = \begin{bmatrix} U \\ V \\ W \end{bmatrix}, {}^{\{T\}}\vec{R}=\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}, {}^{\{T\},T}p_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}, {}^{\{T\}}\hat{n}_i = \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} $$

${}^{\{T\}}\hat{n}_i$: the unit vector of the local direction, closest to the given point, in which the adjustment is the most effective in reducing the distance between the LOS, and the given ball center position.

The fit equation can be re-formulated to adopt a matrix form as follows:

$$ e_i = \varsigma_i - \left({}^{\{T\}}\vec{D} + {}^{\{T\}}\vec{R} \times {}^{\{T\},T}p_i \right) \bullet {}^{\{T\}}\hat{n}_i \tag{5.2} $$

$$ e_i = \varsigma_i - {}^{\{T\}}\vec{D} \bullet {}^{\{T\}}\hat{n}_i - {}^{\{T\}}\vec{R} \times {}^{\{T\},T}p_i \bullet {}^{\{T\}}\hat{n}_i \tag{5.3} $$

$$ e_i = \varsigma_i - {}^{\{T\}}\vec{D} \bullet {}^{\{T\}}\hat{n}_i - {}^{\{T\},T}p_i \times {}^{\{T\}}\hat{n}_i \bullet {}^{\{T\}}\vec{R} \tag{5.4} $$

By further developing Equation (5.4), we obtain:

$$ e_i = \varsigma_i - \begin{bmatrix} U \\ V \\ W \end{bmatrix} \bullet \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} - \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \times \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} \bullet \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} $$

When we isolate $\begin{bmatrix} U \\ V \\ W \end{bmatrix} \bullet \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix}$ and $\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \times \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} \bullet \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}$ they become respectively:

$$ \begin{bmatrix} U \\ V \\ W \end{bmatrix} \bullet \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} = [\, U * n_{ix} + V * n_{iy} + W * n_{iz}] $$

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \times \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \end{bmatrix} \bullet \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} i & j & k \\ x_i & y_i & z_i \\ n_{ix} & n_{iy} & n_{iz} \end{bmatrix} \bullet \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} (y_i * n_{iz} - z_i \ * n_{iy}) \\ -(x_i * n_{iz} - z_i * n_{ix}) \\ (x_i * n_{iy} - y_i * n_{ix}) \end{bmatrix} \bullet \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix}$$

$$= [(y_i * n_{iz} - z_i \ * n_{iy}) * \ \alpha - \ (x_i * n_{iz} - z_i * n_{ix}) \ * \ \beta + (x_i * n_{iy} - y_i * n_{ix}) * \Upsilon]$$

Combining both:

$$e_i = \ \varsigma_i - [\, U * n_{ix} \ + V * n_{iy} + W * \ n_{iz}] - [(y_i * n_{iz} - z_i \ * n_{iy}) * \ \alpha - \ (x_i * n_{iz} - z_i * n_{ix}) \ * \beta + (x_i * n_{iy} - y_i * n_{ix}) * \gamma]$$

A direct solution can be achieved by imposing $e_i = 0$;

Thus, the orthogonal distance ς_i becomes :

$$\varsigma_i = \ U * n_{ix} \ + V * n_{iy} + W * n_{iz} + \ (y_i * n_{iz} - z_i \ * n_{iy}) * \ \alpha - (x_i * n_{iz} - z_i * n_{ix}) \ * \ \beta + (x_i * n_{iy} - y_i * n_{ix}) * \gamma$$

In matrix form expressed as:

$$\begin{bmatrix} n_{1x} & n_{1y} & n_{1z} & (y_1 * n_{1z} - z_1 \ * n_{1y}) & (x_1 * n_{1z} - z_1 * n_{1x}) & (x_1 * n_{1y} - y_1 * n_{1x}) \\ n_{2x} & n_{2y} & n_{2z} & (y_2 * n_{2z} - z_2 \ * n_{2y}) & (x_2 * n_{2z} - z_2 * n_{2x}) & (x_2 * n_{2y} - y_2 * n_{2x}) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (n_{ix}) & (n_{iy}) & (n_{iz}) & (y_i * n_{iz} - z_i \ * n_{iy}) & (x_i * n_{iz} - z_i * n_{ix}) & (x_i * n_{iy} - y_i * n_{ix}) \end{bmatrix} \begin{bmatrix} U \\ V \\ W \\ \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} \varsigma_1 \\ \varsigma_2 \\ \vdots \\ \vdots \\ \varsigma_i \end{bmatrix}$$

(5.5)

This can be substituted as:

$$\mathrm{A}\mathbf{x} \ = \ \mathrm{b}$$

where the least square solution is a Moore **Penrose inverse:**

$$\mathbf{x} = (\mathrm{A}^{\mathrm{T}}\mathrm{A})^{-1}\mathrm{A}^{\mathrm{T}}\mathrm{b} \qquad (5.6)$$

Equation (5.6) represents the overall least squares solution to the linear system A**x** = b.

where A is a known matrix constructed from the problem constraints, b is the vector of observed measurements or residuals, **x** is the unknown parameter vector to be estimated.

In the context of rotation estimation, **x** corresponds to a small rotation vector representing an incremental change in rotation. Because rotations are inherently nonlinear, the problem is approximated by assuming that these incremental rotations are small enough to be treated as linear. This assumption enables the nonlinear rotation estimation problem to be expressed as a linear system solvable by least squares. This approximation holds true locally when the rotation increments are sufficiently small, typically within a few degrees. It allows the use of iterative optimization methods that successively solve linear least squares problems to refine the rotation estimate until convergence.

Therefore, the equation above provides an effective and efficient solution to estimate rotation and translation parameters by minimizing residual errors in the linearized system at each iteration.

5.1.1 Rigid body adjustment

- **Objective**

The aim of this process is to determine a sequence of rigid body transformations that can be applied to a 3D translator. The translator is responsible for moving a spherical target in 3D space. For each position, the objective is to move the translator frame such that the sphere center aligns with a corresponding Line of Sight (LOS) obtained from the camera observations.

- **Context and Assumptions**

• The camera system captures a series of images, each image corresponding to a position of the ball obtained by moving the ball with the translator.

• At each translator position, a line of sight (LOS) through the center of the sphere is defined by :

$^{\{c\},c}o_c$: The origin of the camera is a point on the LOS in camera coordinates (usually the camera center),

$^{\{c\}}\hat{b}$: the direction vector of the LOS (unit vector).

• The sphere center points in the translator frame is denoted by $^{\{T\},T}p_i$ are fixed with respect to the translator.

• The LOS vectors remain fixed, and only the translator frame is moved (in the model) using rigid body transformations.

- **Alignment criterion**

To ensure that the updated ball center points lies on the corresponding LOS, the orthogonal distance between ball center and LOS denoted by ς_i illustrated in Figure 5.5 and given by Equation (5.7) must be minimized.

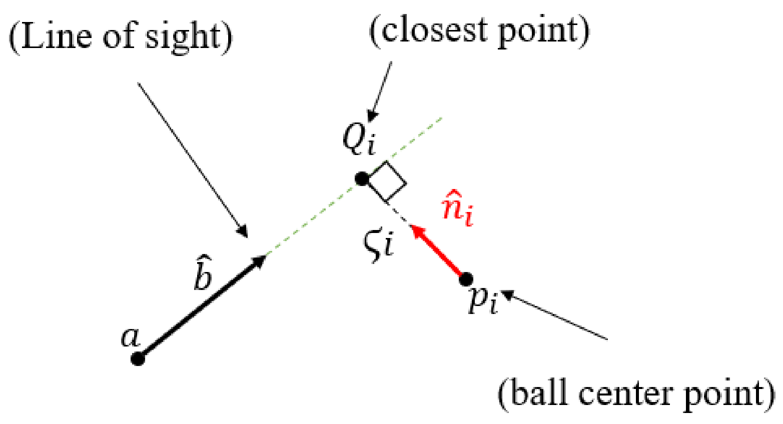


Figure 5.5 Distance from a point to a line in 3D space

$$\varsigma_i = \|(p_i - a) \times \hat{b}\| \tag{5.7}$$

where:

p_i : the ball center.

a : a point on the line of sight.

$\hat{b}$: the unit vector representing the direction of the line of sight.

- **Main procedures**

A. The first step is to define the normal vector ${}^{\{T\}}\hat{n}_i$. Figure 5.6 illustrates the various parameters involved in the formulation of this normal vector.

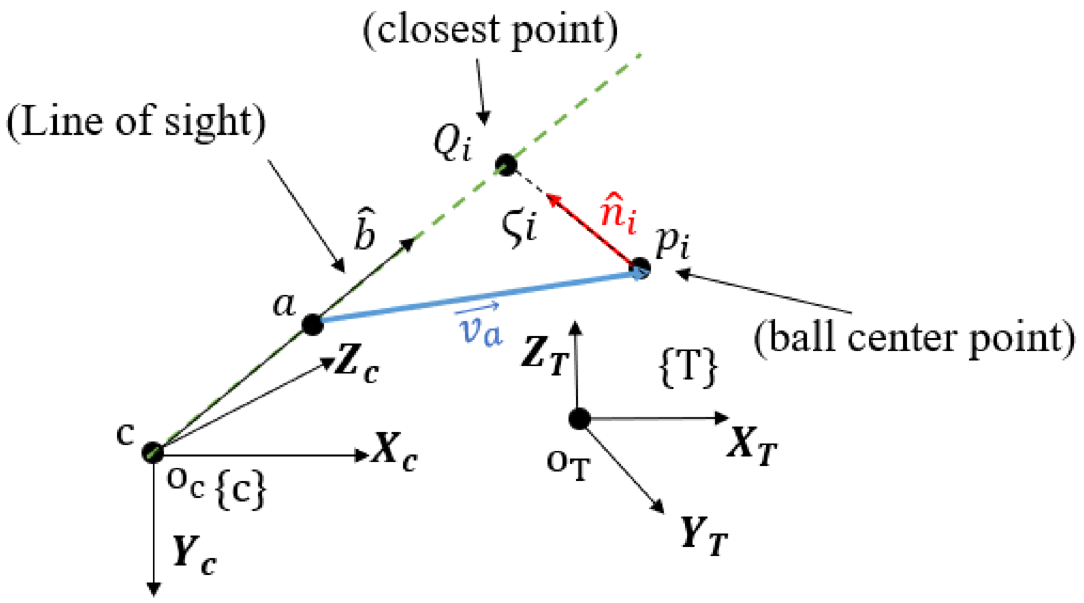


Figure 5.6 3D space relationship between the ball center and the LOS

Thus, from Figure 5.6, the steps involved in finding the closest point on the line of sight ${}^{\{T\},T}Q_i$ in order to derive the vector ${}^{\{T\}}\hat{n}_i$ are described in the following lines:

1. Transform the ball center ${}^{\{T\},T}p_i$ to the camera frame c.

For this process the known positions of the sphere center points in the translator frame denoted by ${}^{\{T\},T}p_i$ are transformed to the camera frame using the estimated initial position and rotation of the translator relative to the camera as shown in Figure 5.7:

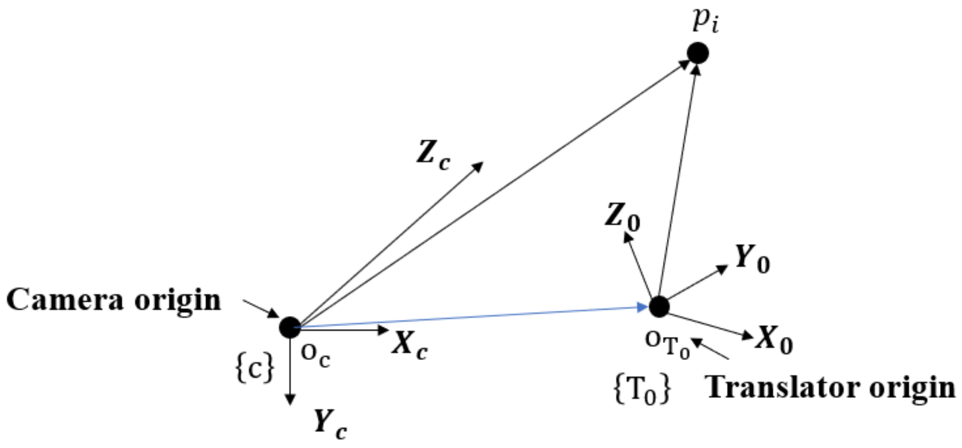


Figure 5.7 Visualization of the ball center position p_i from two different coordinates frames

Thus ${}^{\{c\},c}p_i$ becomes:

$$ {}^{\{c\},c}p_i = {}^{c}_{T_0}T \, {}^{\{T\},T}p_i \tag{5.8}$$

with:

$${}^{c}_{T_0}T : \begin{bmatrix} {}^{c}_{T_0}R & {}^{\{c\},c}o_{T_0} \\ 0_{1\times3} & 1 \end{bmatrix}$$

$${}^{c}_{T_0}R = R(\gamma, z)R(\beta, y)R(\alpha, x)$$

Υ, β, α : These serves as initial estimates of the orientation of the translator frame. They are the successive rotation in moving frame to align frame c with frame T. They can be nominal like 90 degrees (1.5 rad).

${}^{\{c\},c}o_{T_0}$: origin vector point of the translator frame at its initial position expressed in the camera frame.

where $R(\gamma, z)$; $R(\beta, y)$; $R(\alpha , x)$ matrices are obtained as in Equations (5.9); (5.10); (5.11); below.

$$R(\gamma, z) \text{ around z} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.9}$$

$$\mathrm{R}(\beta, y) \text{ around y} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{5.10}$$

$$\mathrm{R}(\alpha, x) \text{ around x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{5.11}$$

The sequence $\mathrm{R}_z\mathrm{R}_y\mathrm{R}_x$ is used because it represents intrinsic rotations, meaning each rotation is applied about the local (moving) axes of the object being aligned, not about the fixed axes of the camera frame. This convention is appropriate when the goal is to rotate a moving frame (i.e., object or IGF) into alignment with a fixed target frame (i.e., the camera frame).

2. We can derive the equation for any point B lying on the LOS as:

$$^{\{c\},c}\mathrm{B}(t) = {}^{\{c\},c}a + t\, {}^{\{c\}}\hat{b} \tag{5.12}$$

where;

$^{\{c\},c}a$ is point on the line.

$^{\{c\}}\hat{b}$ is the direction vector of the line of sight.

t is a scalar parameter.

3. We can define a vector $^{\{c\}}v_a$, from the point $^{\{c\},c}a$ on the LOS to the sphere center point position $^{\{c\},c}p_i$ as illustrated in Figure 5.6 using Equation (5.13).

$$^{\{c\}}v_a = {}^{\{c\},c}p_i - {}^{\{c\},c}a \tag{5.13}$$

- In in our camera model the LOS passes through the camera frame origin and so this origin can be selected as $^{\{c\},c}a$.

4. To find the projection of the point ${}^{\{c\},c}p_i$, we calculate the scalar t, which defines how far along the line the projection lies. This is given by :

$$t = {}^{\{c\}}v_a \bullet {}^{\{c\}}\hat{b} \tag{5.14}$$

This formula provides the projected length of ${}^{\{c\}}v_a$ in the direction of ${}^{\{c\}}\hat{b}$.

5. Derive the closest point on the line : Once t is known, the closest point ${}^{\{c\},c}Q_i$ is defined by the equation below :

$${}^{\{c\},c}Q_i(t) = {}^{\{c\},c}a + t\,{}^{\{c\}}\hat{b} \tag{5.15}$$

6. We then transform back ${}^{\{c\},c}Q_i$ to the translator frame T using Equation (5.16):

$${}^{\{T\},T}Q_i = {}^{c}_{T_0}T^{-1}\, {}^{\{c\},c}Q_i \tag{5.16}$$

Thus ${}^{\{T\}}\hat{n}_i$ becomes:

$${}^{\{T\}}\hat{n}_i = \frac{\vec{d}}{|\vec{d}|}; \vec{d} = {}^{\{T\}}\vec{d} = {}^{\{T\},T}Q_i - {}^{\{T\},T}p_i \tag{5.17}$$

B. Derive the rotation vector and translator vector in order to move the translator.

The derivation of the components of the rotation vector ${}^{\{T\}}\vec{R}$ and translation vector ${}^{\{T\}}\vec{D}$ is performed using Equation (5.6) and this move the translator from T_0 (initial position) to its new position T_1. This yields a homogenous transformation matrix ${}^{T_0}_{T_1}T$ that will be applied to ${}^{\{T\},T}p_i$.

Thus;

$$\text{new Translator pose} = {}^{T_0}_{T_1}T\, {}^{\{T_1\},T_1}p_i$$

where $${}^{T_0}_{T_1}T\left({}^{\{T_0\}}\vec{D}, {}^{\{T_0\}}\vec{R} \to {}^{T_0}_{T_1}R\right) \to \begin{bmatrix} {}^{T_0}_{T_1}R & {}^{\{T_0\}}\vec{D} \\ 0_{1\times 3} & 1 \end{bmatrix}$$

with;

${}^{\{\mathrm{T}_0\}}\vec{R} = \mathrm{R}(\gamma, z)\mathrm{R}(\beta, y)\mathrm{R}(\alpha, x)$;

${}^{\{\mathrm{T}_0\}}\vec{R}$ produces a rotation matrix ${}^{\mathrm{T}_0}_{\mathrm{T}_1}\mathrm{R}$ that projects a vector expressed in T_1 into how its components in T_0.

The final pose of the translator can be represented as a series of homogeneous transformations applied sequentially through intermediate coordinate frames as expressed in Equation (5.18).

Thus, for n pose;

$$\text{n Translator pose} = \left(\prod_{j=0}^{n-1} {}^{\mathrm{T}_j}_{\mathrm{T}_{j+1}}\mathrm{T}\right) {}^{\{\mathrm{T}_{n-1}\},\mathrm{T}_{n-1}}p_i \tag{5.18}$$

C. Iterative registration update

The iterative part consists of progressively updating the translator frame's pose by applying small incremental transformations at each step based on the current alignment. This implies finding the rotation vector $\vec{R}$ and the translation vector $\overrightarrow{D}$ and combining them into a homogenous matrix at each pose. But at each step the transformation from the current pose of the translator frame to the camera frame is computed in order to derive ${}^{\{\mathrm{T}_n\}}\hat{n}_i$, with n representing the current translator's pose. The underlaying concept principle is illustrated in Figure 5.8.

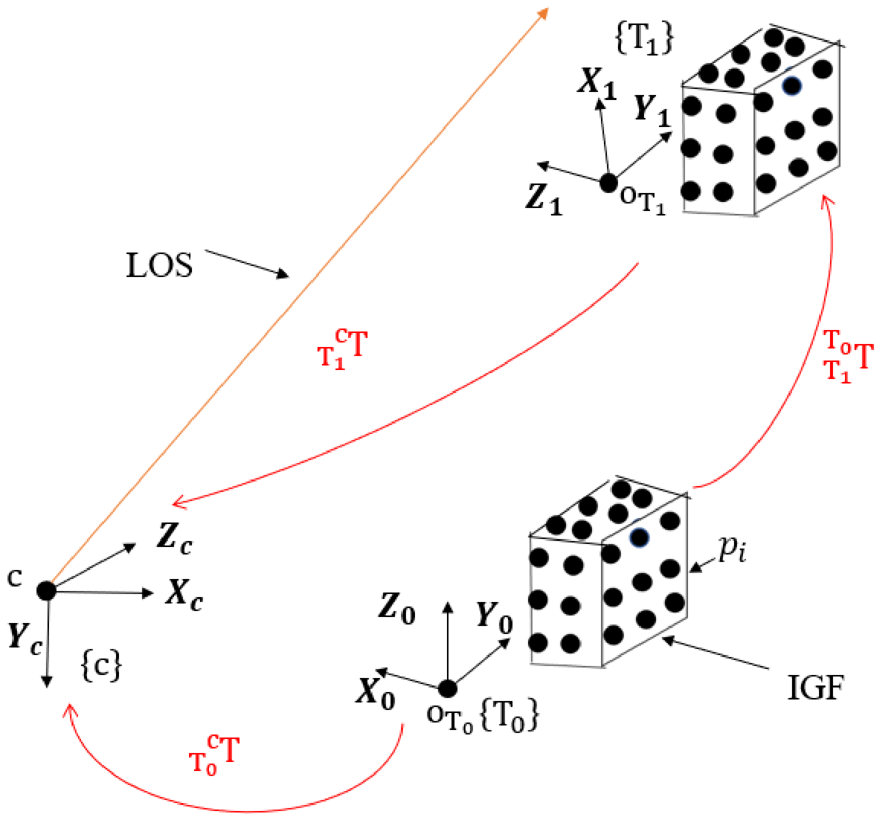


Figure 5.8 Illustration of the movement of the translator from T_0 to T_1

Let's say we have moved the translator from T_0 to T_1 as done in Figure 5.8. We transform ${}^{\{T_1\},T_1}p_i$ into the camera frame base in order to calculate a new ${}^{\{c\},c}Q_i$ which is then re-expressed in frame $\{T_1\}$ as ${}^{\{T_1\},T_1}Q_i$ so that a new ${}^{\{T_1\}}\hat{n}_i$ is calculated. We define a transformation matrix ${}^{c}_{T_1}T$ which is expressed by:

$${}^{c}_{T_1}T = {}^{c}_{T_0}T \; {}^{T_0}_{T_1}T$$

- Thus the new ${}^{\{c\},c}p_i$ becomes;

$${}^{\{c\},c}p_i = {}^{c}_{T_1}T \; {}^{\{T\},T}p_i \tag{5.19}$$

for successive transformation we have:

$$ {}_{\mathrm{T_n}}^{\mathrm{c}}\mathrm{T} = {}_{\mathrm{T_0}}^{\mathrm{c}}\mathrm{T}\, {}_{\mathrm{T_1}}^{\mathrm{T_0}}\mathrm{T}\, {}_{\mathrm{T_2}}^{\mathrm{T_1}}\mathrm{T} \dots {}_{\mathrm{T_n}}^{\mathrm{T_{n-1}}}\mathrm{T} \tag{5.20}$$

Thus ${}^{\{\mathrm{c}\},\mathrm{c}}p_i$ at the current pose becomes:

$$ {}^{\{\mathrm{c}\},\mathrm{c}}p_i = {}_{\mathrm{T_n}}^{\mathrm{c}}\mathrm{T}\, {}^{\{\mathrm{T}\},\mathrm{T}}p_i \tag{5.21}$$

5.1.2 Bundle adjustment principle

In this application, bundle adjustment refers to the iterative refinement of the pose of the translator frame (ball center positions) to bring it as close as possible to their corresponding LOS.

The detailed steps are:

1. For each position of the translator frame relative to the camera frame calculate the closest point on the LOS to ball centers positions to find the normal vector .
2. Calculate the distance between the LOS and corresponding ball center positions.
3. Find the rotation and translation parameters and move the translator frame accordingly in the model.
4. In the next iterative frame check for residuals (which is the distance after the micro-adjustement between LOS and ball centers).
5. Check if the model meets the convergence criteria:

- In synthetic simulations, a convergence RMSE of residual vectors of 10^{-12} mm is employed as the convergence criteria because this level of precision is expected.
- For real data we monitor the change in residuals or the change in parameters (rotation or translation) between iterations. If the change becomes too small, this indicates that further iterations are unlikely to yield significant improvements. To avoid the loop to run for ever we set a maximum iterations target like 15.

6. If the real or synthetic model does not meet their required convergence criteria listed above, we repeat step 1-6 until convergence.

The flow chart depicted in Figure 5.9 and Figure 5.10 describe the process of deriving the LOS from the ellipse fit on the precision sphere and adjustment of the translator to the LOS.

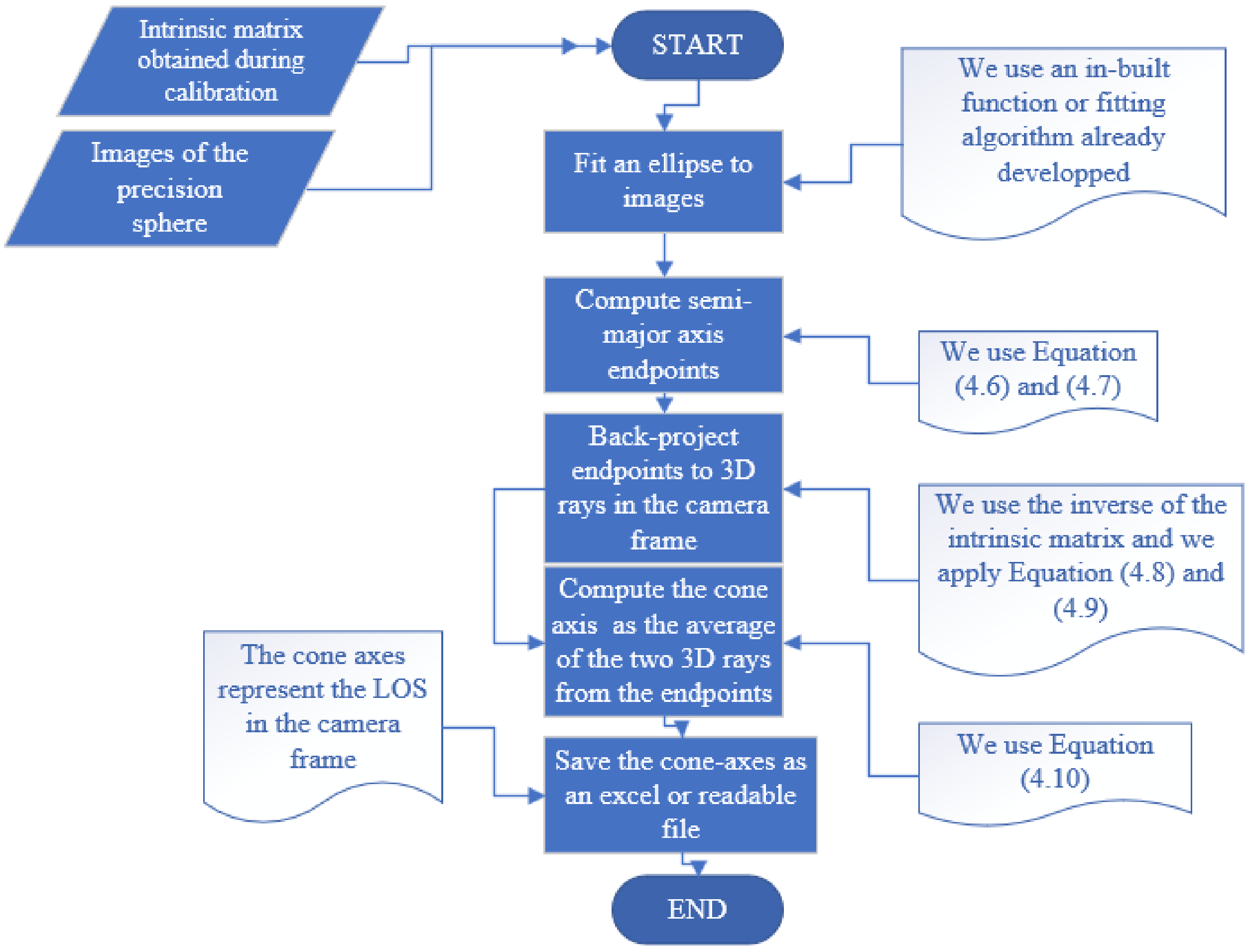

Intrinsic matrix obtained during calibration
Images of the precision sphere
START
Fit an ellipse to images
We use an in-built function or fitting algorithm already developped
Compute semi-major axis endpoints
We use Equation (4.6) and (4.7)
Back-project endpoints to 3D rays in the camera frame
We use the inverse of the intrinsic matrix and we apply Equation (4.8) and (4.9)
Compute the cone axis as the average of the two 3D rays from the endpoints
The cone axes represent the LOS in the camera frame
We use Equation (4.10)
Save the cone-axes as an excel or readable file
END

Figure 5.9 Flow chart describing the derivation of the LOS from sphere images

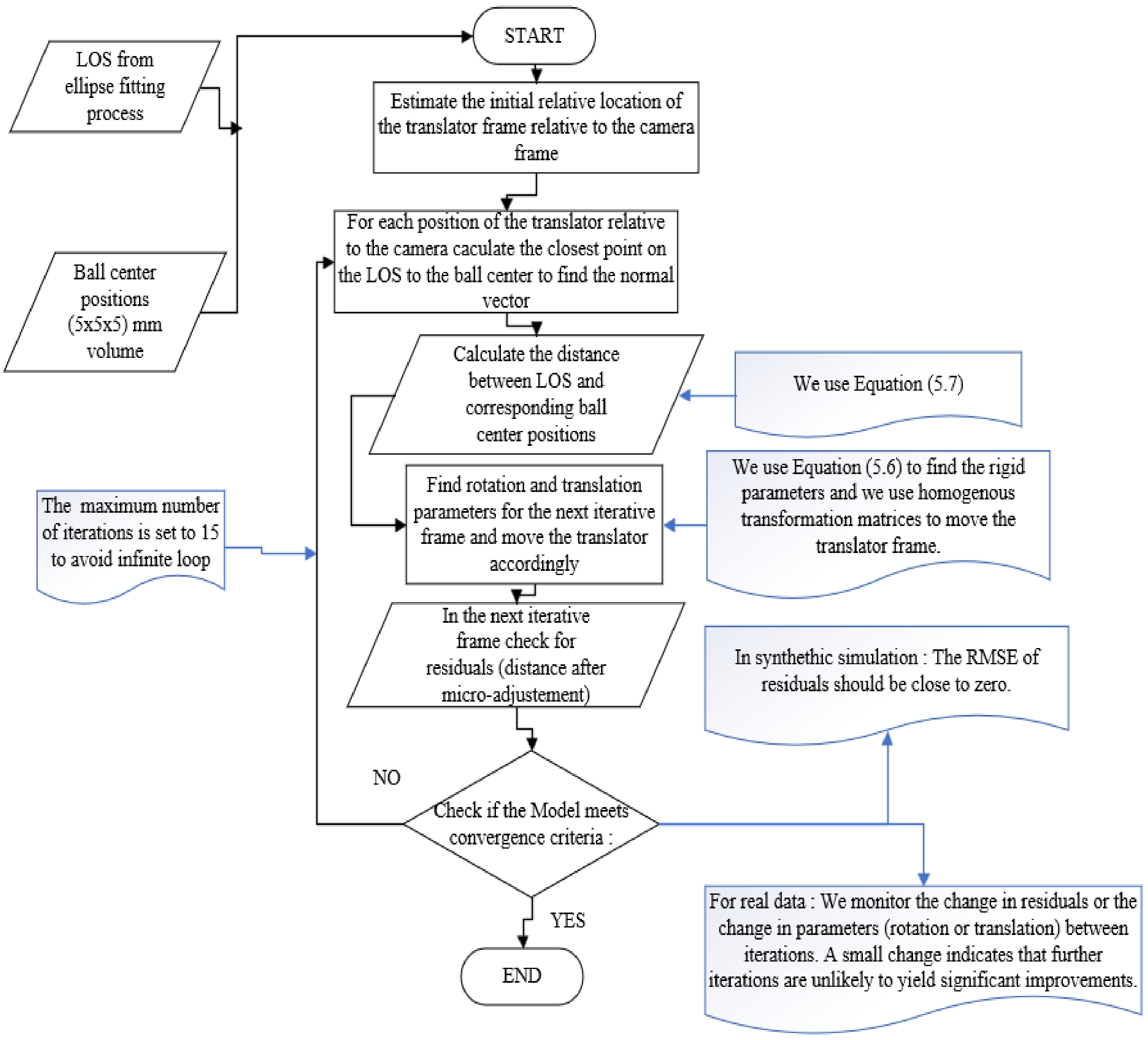

LOS from ellipse fitting process
Ball center positions (5x5x5) mm volume
START
Estimate the initial relative location of the translator frame relative to the camera frame
For each position of the translator relative to the camera caculate the closest point on the LOS to the ball center to find the normal vector
Calculate the distance between LOS and corresponding ball center positions
We use Equation (5.7)
Find rotation and translation parameters for the next iterative frame and move the translator accordingly
We use Equation (5.6) to find the rigid parameters and we use homogenous transformation matrices to move the translator frame.
The maximum number of iterations is set to 15 to avoid infinite loop
In the next iterative frame check for residuals (distance after micro-adjustement)
In synthethic simulation : The RMSE of residuals should be close to zero.
NO
Check if the Model meets convergence criteria :
YES
END
For real data : We monitor the change in residuals or the change in parameters (rotation or translation) between iterations. A small change indicates that further iterations are unlikely to yield significant improvements.

Figure 5.10 Flow chart describing the fitting process of the 3D translator to the LOS

5.2 Validation

5.2.1 Validation using synthetic data

In the simulated experiments, the conditions are as follows:

- ➢ A grid of dimensions 5×5×5 (1 mm increment) 125 points.
- ➢ A grid located at [0 70 0]
- ➢ The camera center located at [0 0 0]

The simulated experiment performed using MATLAB is illustrated in Figure 5.11.

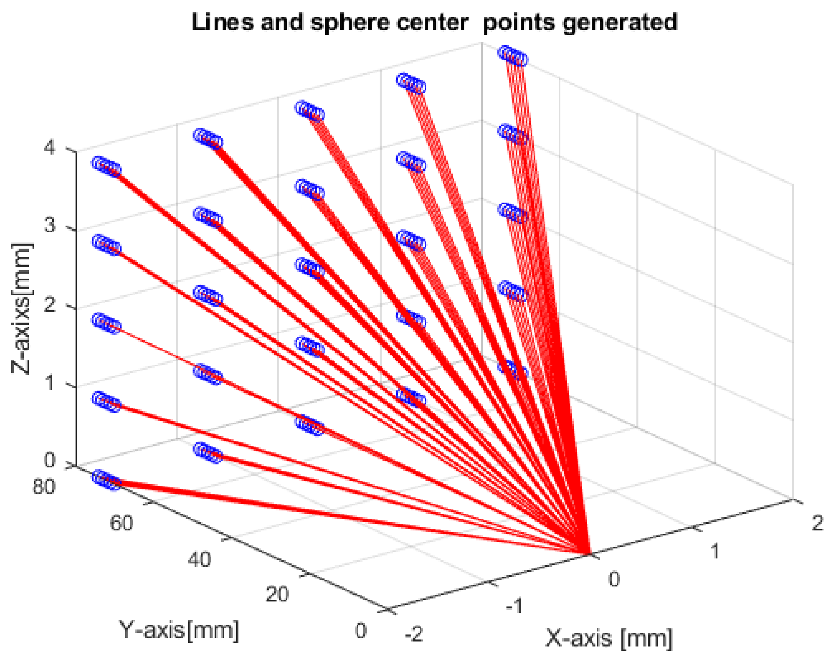


Figure 5.11 Simulated experiment performed using MATLAB code

An RMSE of the residual's distances between the LOS and the ball centres of 3.9×10^{-12} mm is obtained which indicates that the lines of sight and sphere centers points are perfectly aligned.

Number of iterations:1

a) With set up errors

Translation parameters: t_x = 0.1 mm; t_y= - 0.2 mm; t_z = 0.3 mm

Rotation angles: α = 0.5 radians; β = 0.15 radians; γ = 0.1 radians

A simple graph of RMSE versus Number of iterations is illustrated in Figure 5.12. The RMSE values obtained in Table 5.1 indicate that the algorithm accurately recovered the initial positions of the bundle (or sphere center points) with a precision up to 10^{-12} mm in 5 iterations.

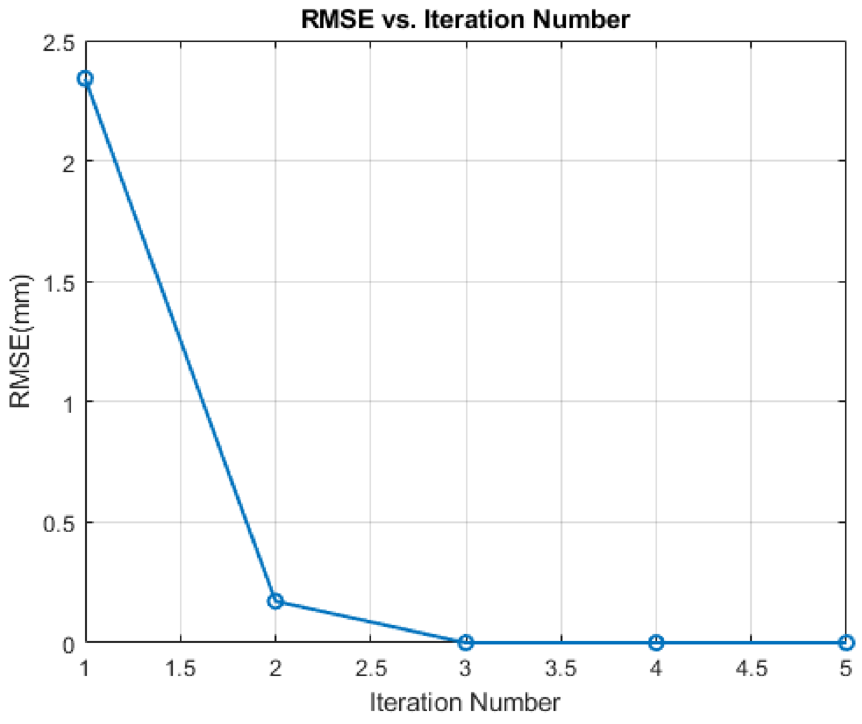


Figure 5.12 RMSE graph vs synthetic data (when introducing set up errors)

Table 5.1 describes the associated value of the RMSE at each iteration.

Table 5.1 Iterations vs RMSE values (synthetic data with set up errors)

Iterations	RMSE (mm)
1	2.342
2	0.172
3	2.8178×10^{-5}
4	1.5407×10^{-11}
5	4.5343×10^{-12}

5.2.2 Experimental validation

The experimental set up used to validate our results is shown in Figure 5.13.

- The ceramic sphere diameter is 1 inch or 25.4 mm.
- The camera has a resolution of 12.3 MP (4056×3040) for a maximum FOV of 75°.
- The working distance from camera to target is approximately 60 mm.
- The sphere is mounted on a 3D translator that allows movement in all 3 directions (X, Y, Z).
- 125 pictures were taken following 1 mm increment for a maximum displacement of 4 mm in X, Y and Z yielding a 5×5×5 = 125 points grid.

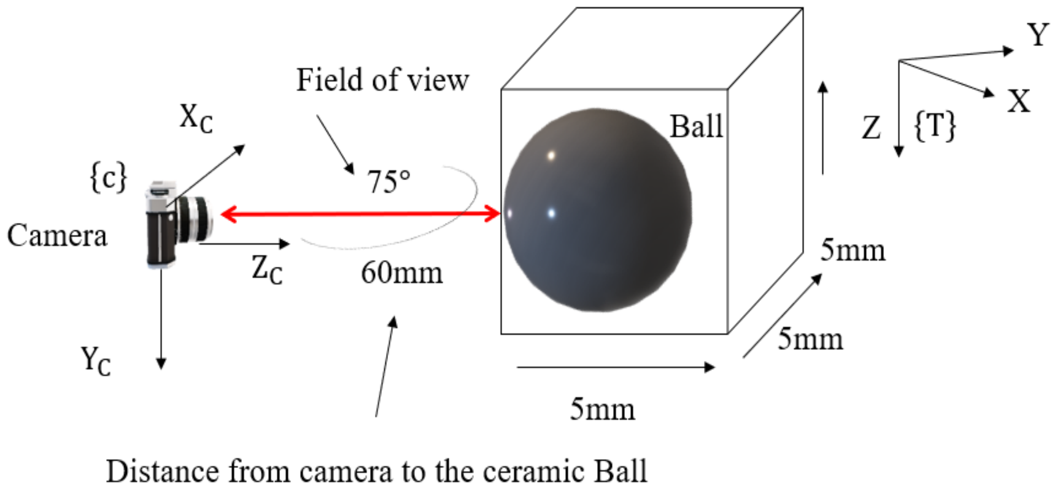


Figure 5.13 Experimental set up

A lab version of the illustrated experiment is shown in Figure 5.14.

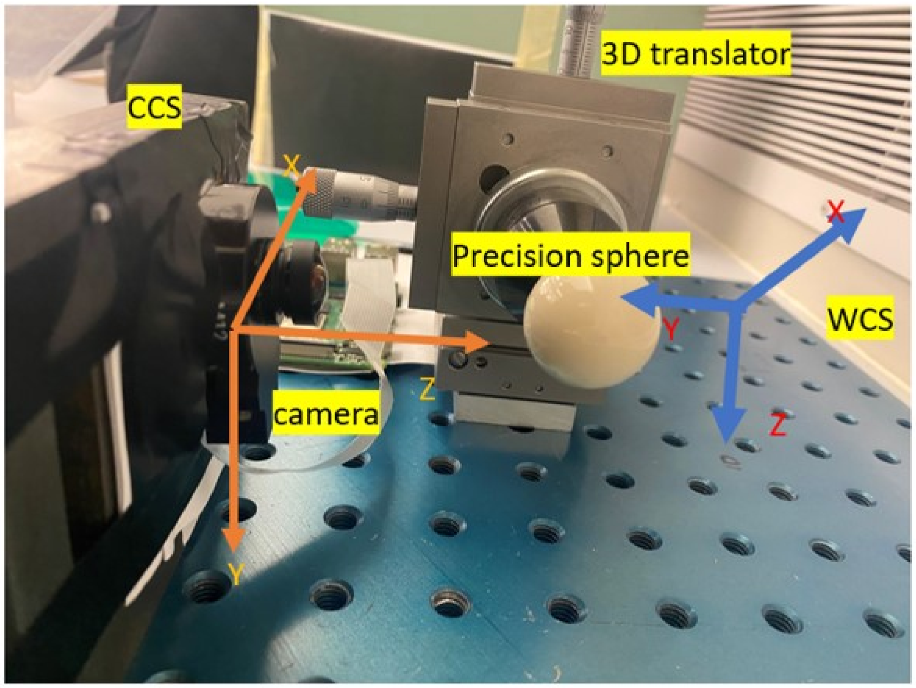


Figure 5.14 Lab experimental set up

1. **Calibration results**

The calibration pattern (checkboard pattern) used for the camera intrinsic calibration is illustrated in Figure 5.15. The dimensional accuracy of the pattern is 10 µm with a square size of 5 mm.

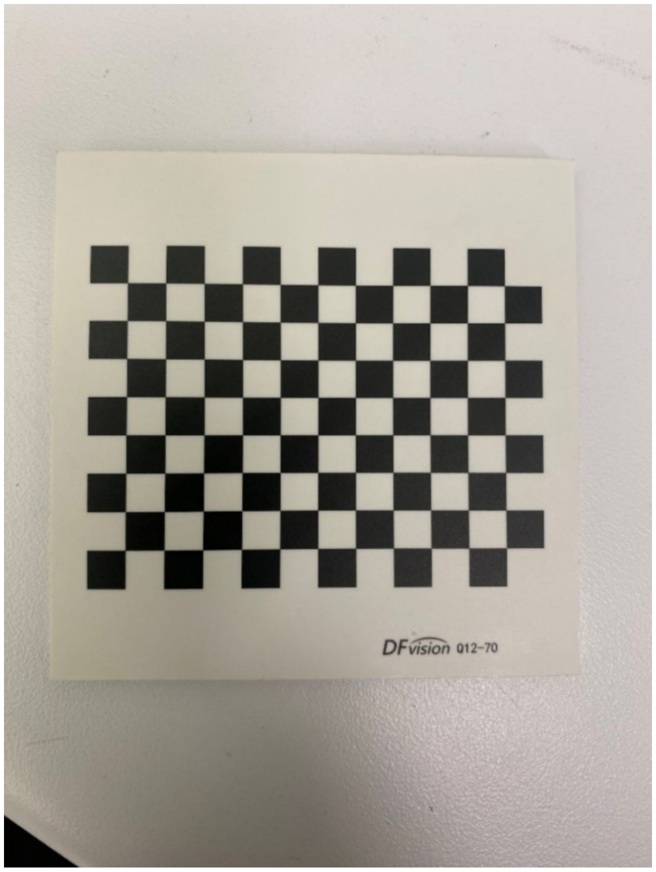


Figure 5.15 Calibration pattern used during the experiment

a) Camera parameters

Table A.1 in the Appendix A lists the intrinsic and extrinsic parameters obtained for calibration including the coefficients of distortion, and the values of the standard deviations related to each parameter directly computed by the MATLAB camera calibration toolbox used.

b) Mean reprojection error

The mean reprojection error found after calibration is illustrated in Figure 5.16.

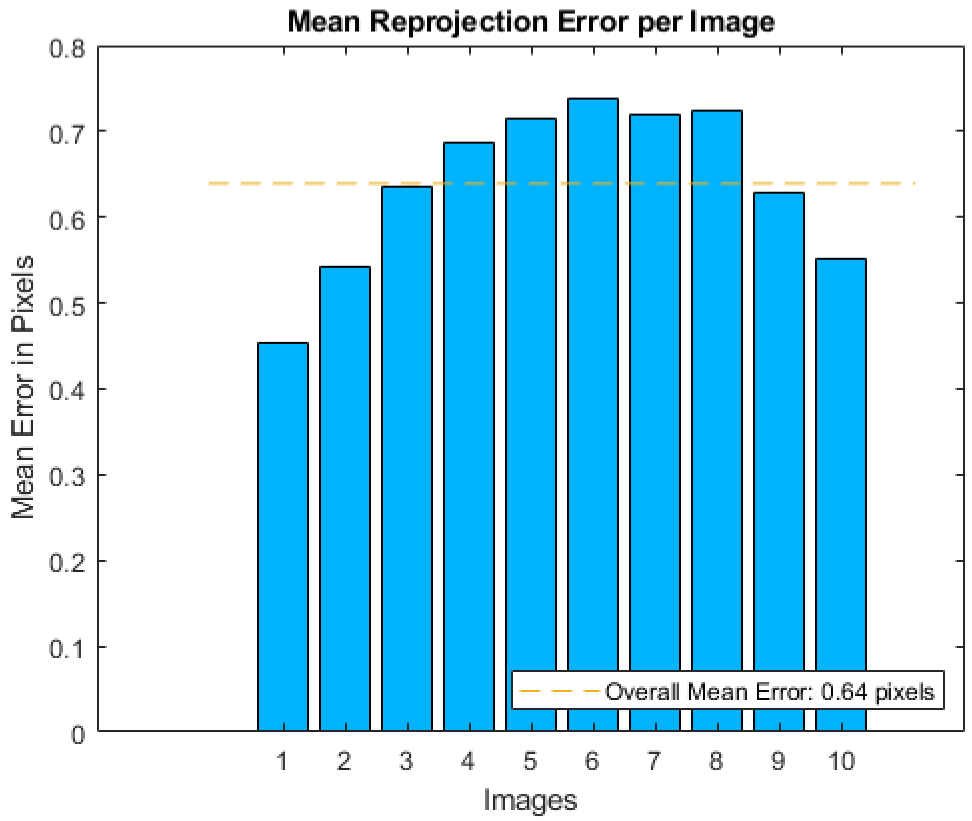


Figure 5.16 Mean reprojection error graph for 10 positions

The extrinsic visualization graph in Figure 5.17 shows the 10 different orientations of the pattern relative to the camera during the calibration.

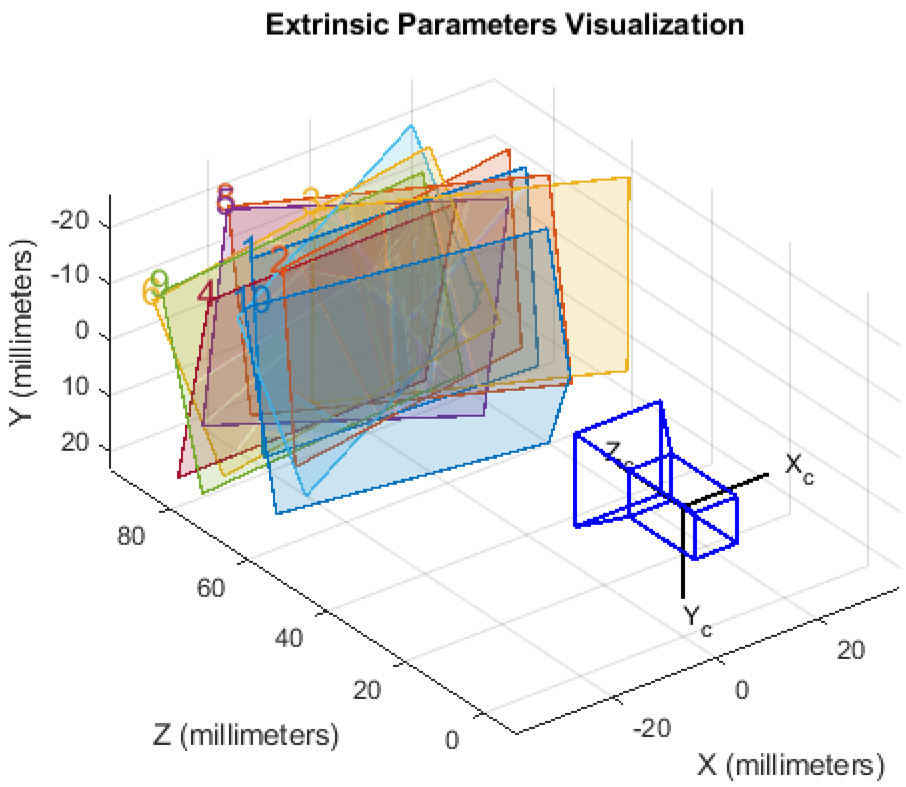


Figure 5.17 Extrinsic parameters visualization

2. **Evaluating the camera calibration accuracy**

The reprojection error obtained from camera calibration reflects the difference between observed image points and their projected counterparts from estimated 3D models. Although this error is typically expressed in pixels, it can be converted to a metric spatial error given knowledge of the camera's geometry and the experimental working distance. This approach was developed by cam.calib [55] and it is described using Figure 5.18.

Let us consider a pixel's angular extent, denoted α_{pixel}(deg) which is defined by the camera's horizontal field of view (FOV_H) and the camera horizontal pixel count (w_{px}) as illustrated in Figure 5.18.

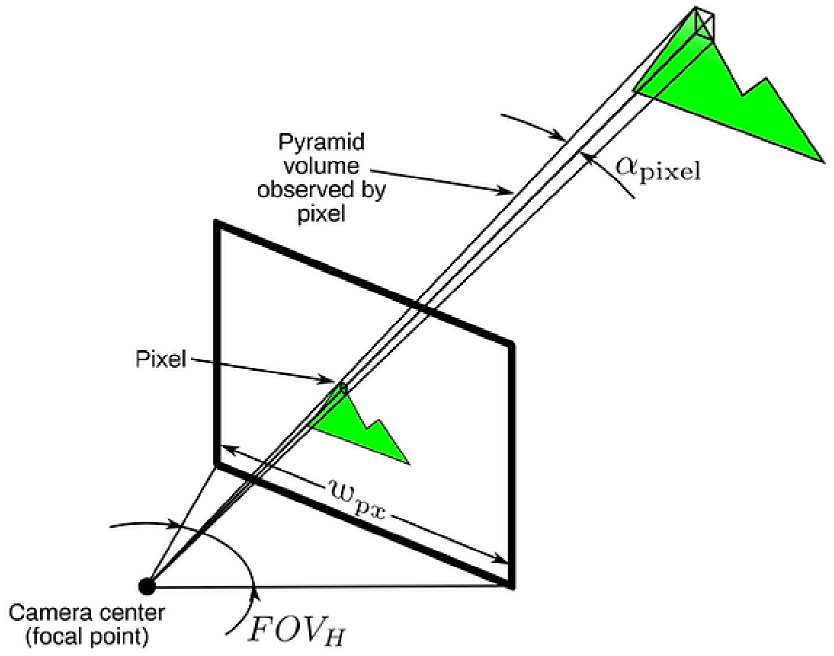


Figure 5.18 Pyramid formed by pixel grid [55]

Every square pixel in the scene is a miniature pyramid, and all pyramids start at the camera's focal point thus;

$$\alpha_{\text{pixel}}(\text{deg}) = \frac{FOV_H}{w_{px}} = \frac{75}{3040} = 0.024°$$

$w_{px} =$ Number of pixels in the horizontal direction

FOV_H= Horizontal field of view

$$\alpha_{pixel} = 0.024°$$

The angular error $e_{arc(degree)}$ corresponding to a reprojection error ($e_{reproject}$) = 0.64 pixel is then given by:

$$e_{arc}(\text{deg}) = \alpha_{pixel} \times e_{reproject} = 0.024 * 0.64 = 0.015 \text{ deg}$$

Converting it to radians gives:

$$e_{arr}(\text{rad}) = 0.015° \times \frac{\pi}{180} = 2.618 \times 10^{-4} \text{ rad}$$

Assuming a working distance of 60 mm; the corresponding length-error is:

$$e_{len}(\text{mm}) = d_{work} \times e_{arc}(\text{rad}) = 0.06 * 2.618 \times 10^{-4} = 1.566 \times 10^{-5}\ \text{m} = 15.7\mu\text{m}.$$

This approximation demonstrates that a seemingly small reprojection error in pixel units may translate to a measurable physical error in the tens of micrometers range, depending on the optical configuration and distance. While this model assumes a pinhole projection geometry and uniform pixel angular coverage, it offers a reasonable first order estimation suitable for assessing calibration precision.

3. Experimental Results

Figure 5.19 shows the RMSE of the 3D distance at each iteration between sphere center points and the corresponding lines of sight (from the camera).

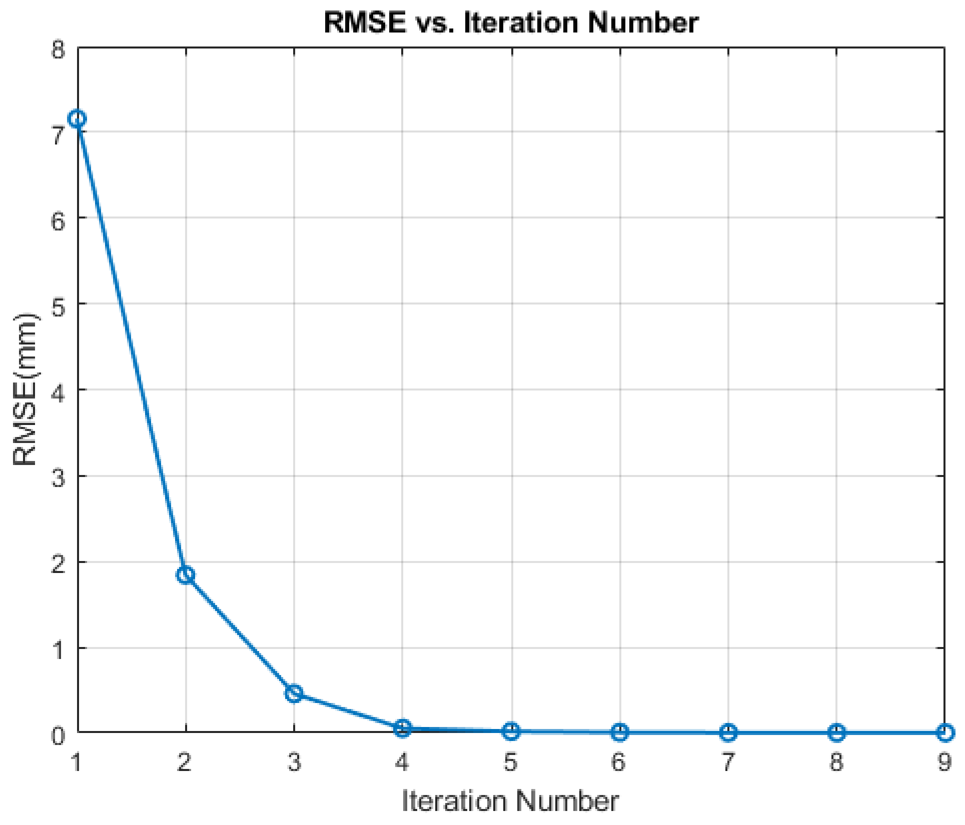


Figure 5.19 RMSE vs iteration (experiment results)

Table 5.2 shows the values of RMSE obtained at different iterations. The convergence or the best fit value obtained so that all lines of sight pas as close as possible to each ball center is RMSE=2 μm after 9 iterations.

Table 5.2 Lab experiment RMSE values

Iterations	RMSE (mm)
1	7.1567
2	1.8426
3	0.4582
4	0.0542
5	0.0182
6	0.0073
7	0.0036
8	0.0024
9	0.0020

5.3 Discussion of results

5.3.1 Calibration results

1. Reprojection error analysis

The reprojection error, which quantifies the discrepancy between observed image points and their corresponding projections based on the estimated camera parameters, provides a critical measure of calibration accuracy. In this study, the overall mean reprojection error was found to be 0.64 pixel (sub pixel accuracy), with individual image errors ranging from approximately 0.45 to 0.74 pixel. This level of accuracy is well within acceptable limits (less than 1 pixel is considered to be a good calibration) [56], particularly considering the high resolution of the image sensor and the focal length exceeding 2500 pixels. A high focal length in pixels improves image resolution at the center of the image and allows for more precise localization of features, especially when the camera is used at short to medium distances from the calibration target. The relatively uniform error distribution across the dataset, with no significant outliers, indicates a robust calibration. The slightly higher errors observed in a few images may be attributed to suboptimal conditions such as image blur, reduced contrast, or oblique viewing angles. Nevertheless, the overall low and consistent reprojection error confirms that the estimated intrinsic and distortion parameters accurately model the camera's geometry, making the calibration suitable for high-precision applications such as 3D reconstruction and machine tool alignment.

2. Analyzing camera parameters

The intrinsic parameters of the calibrated camera system offer critical insight into the optical characteristics and stability of the imaging configuration. The focal lengths, estimated at 2573.9990 ± 2.4102 pixels in the horizontal direction and 2572.6654 ± 2.3419 pixels in the vertical direction, demonstrate a high degree of isotropy in the sensor's resolution. The negligible difference between the two values confirms that the sensor pixels are effectively square, a desirable property for accurate image-based measurement and reconstruction tasks. Moreover, the very low uncertainty (< ± 3 pixels) associated with these values is indicative of a strong optimization process during calibration and sufficient image coverage of the calibration pattern.

The principal point estimated at (2034.8963 ± 0.9878, 1536.5415 ± 0.7988) pixels are closely aligned with the geometric center of the image sensor (2028,1520) pixels, suggesting proper sensor alignment within the camera housing. This parameter defines the point where the optical axis intersects the image plane and is crucial for accurate projection and back-projection in 3D computer vision algorithms. The proximity of this point to the image center is expected in a well-manufactured optical system, and the small uncertainties reflect confidence in this estimation.

The radial distortion coefficients, k_1= 0.0255 ± 0.0012 and, k_2 = − 0.094 ± 0.003 characterize the nature of radial displacement of image points caused by lens imperfections. The positive value of k_1 and the negative value of k_2 together indicate a modest barrel distortion near the center, transitioning toward slight pincushion distortion in the periphery, an expected pattern in high-quality wide or standard lenses. These values are relatively small; given that typical values for mild barrel distortion range between 0 and 0.1, confirming that the lens exhibits minimal distortion, further supported by the accurate reprojection error results.

Tangential distortion arises due to misalignment of the lens with the image sensor, such as lens decentering or tilt. The near-zero values obtained in this calibration imply that the lens is well-centered and aligned, and that any residual tangential effects are negligible. This is important for high precision applications where off-axis distortion could lead to significant positional errors in 3D reconstruction.

In summary, the parameters imply that the camera has been properly designed and calibrated. It possesses a high degree of accuracy with respect to the intrinsic parameters due to the small residuals in the optimization process indicating compatibility across defined parameters. Such cameras are suitable for use in areas requiring precision such as industrial metrology, robot calibration, or photogrammetric measurement.

During the calibration process, the rotation vectors convey key details concerning the pattern alignment in 3D space. The differences in the magnitude of the X, Y and Z components indicates the need to capture different orientations during the calibration, while the small uncertainties result in more precise parameter retrieval. The presence of small rotations in all views indicates that the calibration dataset has been captured from multiple angles, which helps improve parameter estimation.

Looking at the distribution of the translation vectors, it is evident it varies across the X, Y, and Z axes. The Z component shows more significant variation, with values ranging from approximately 68.87 mm to 95.27 mm, indicating that the pattern was moved along the Z-axis over the course of the calibration. This range of translations ensures that the calibration captures depth variations, which is critical for accurately modeling camera parameters

5.3.2 Experimental results

In the initial phase of verification by simulations, it was verified tha the algorithm could recover known positions after the intentional introduction of error. The initial sphere position within camera coordinates was successfully recovered with an accuracy greater than 10^{-12} mm in 4 iterations. However, in the laboratory experiments, the final RMSE observed between the lines of sight and the corresponding sphere center is around 0.0020 mm with vectors norms not exceeding 0.0059 mm. This deviation is attributed to noise present between the actual points (sphere contour points, and pattern corner points) and their projections on the image plane. Potential sources of this noise include calibration inaccuracies and variables in image acquisition. The noise levels are further influenced by the following factors such as lighting conditions and image processing techniques used during the experiment, which directly impact the quality of acquired data.

The residual vectors shown in Figure 5.20, magnified by a factor of 1000 for better visibility, represent the misalignment between sphere centers and their corresponding lines of sight (LOS). These residuals are smaller and more uniform near the center of the grid, indicating better alignment accuracy in that region. Toward the edges and corners, the residuals increase in magnitude.

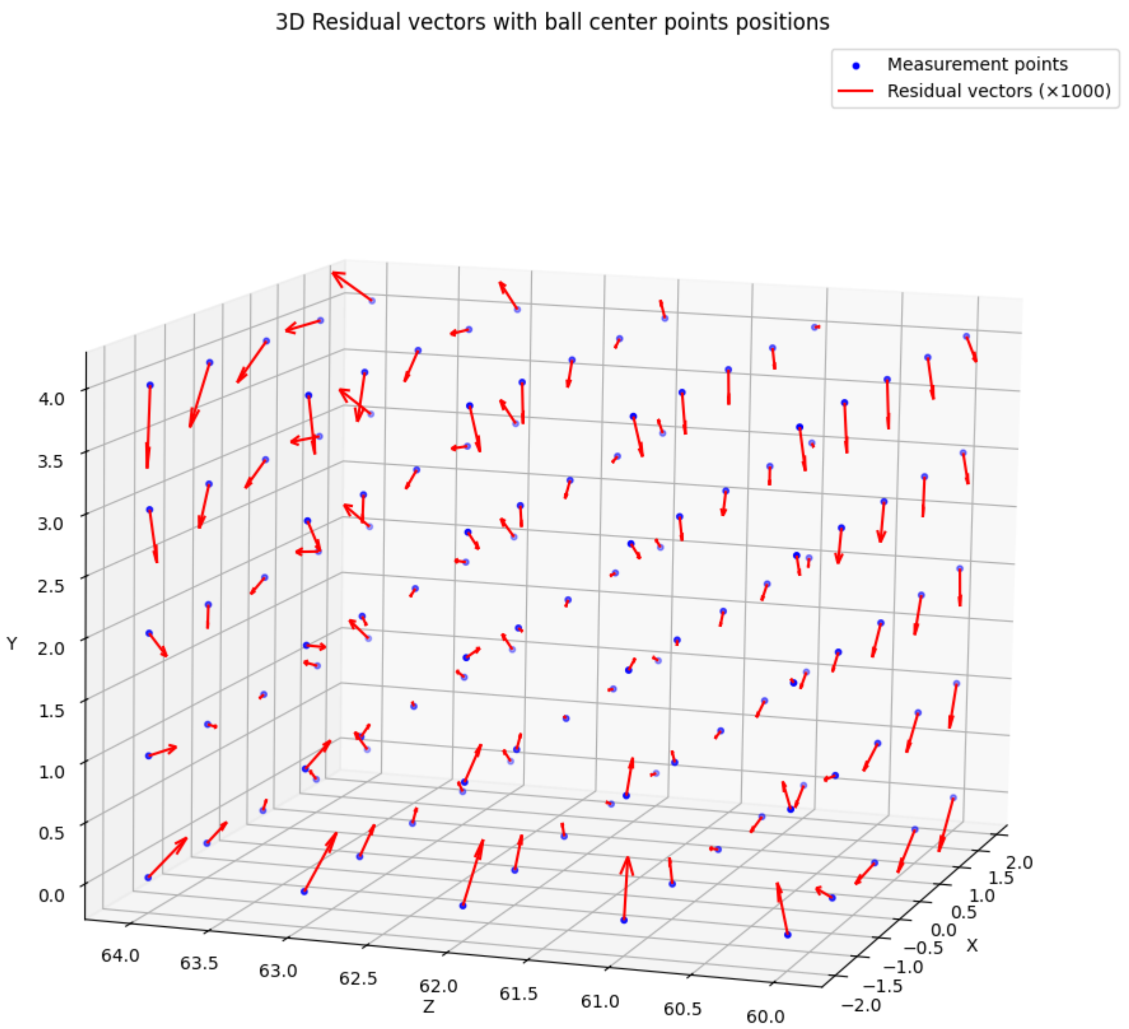


Figure 5.20 XYZ plot displaying residual vectors (red), scaled by a factor of 1000, alongside observed displacements of ball center points (blue). The visualization shows that residuals are smaller near the central region and tend to increase toward the periphery

This behavior suggests that the alignment accuracy varies depending on where it occurs in the field of view; a phenomenon referred to as spatially dependent alignment accuracy. In other words, the system performs more accurately near the optical center and less accurately toward the periphery. Contributing factors may include residual lens distortion, reduced calibration precision at the edges, and uneven lighting, which can affect ellipse detection quality and lead to errors in reconstructing the LOS through back-projections.

These results highlight the importance of good calibration coverage and consistent illumination across the entire scene to ensure reliable performance throughout the measurement volume.

To confirm the validity of the results, The residual vector map depicted Figure 5.21; Figure 5.22; Figure 5.23 provides a detailed visualization of residual vector distributions associated with each data point in different plane configuration.

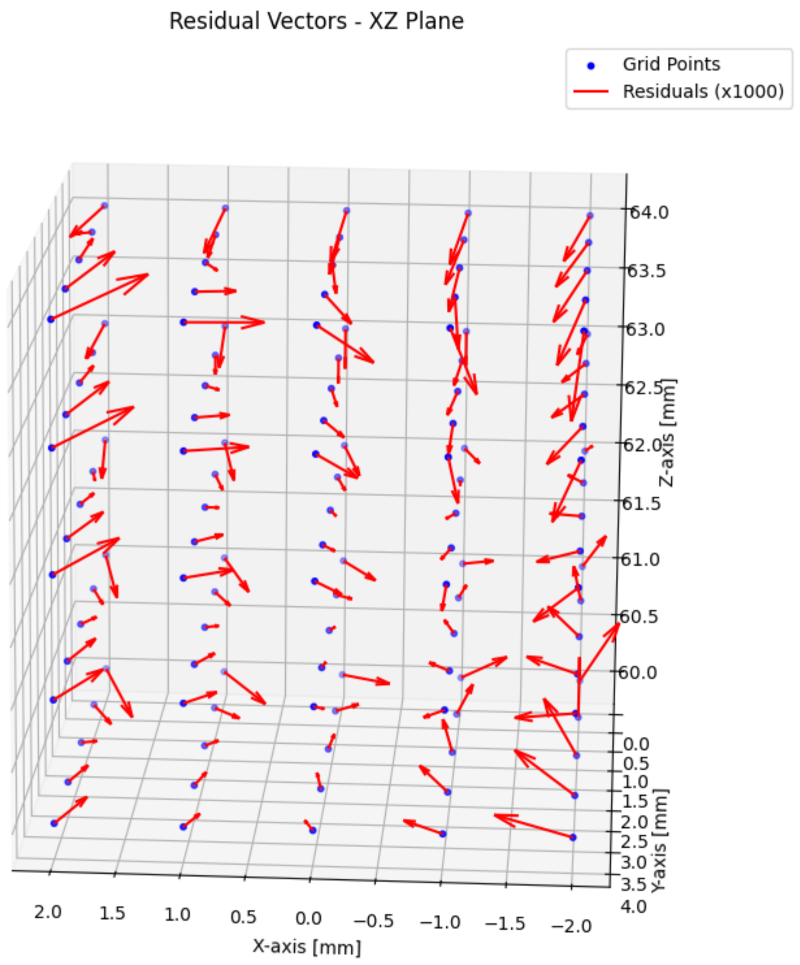


Figure 5.21 A map of residual vectors is plotted in the XZ plane

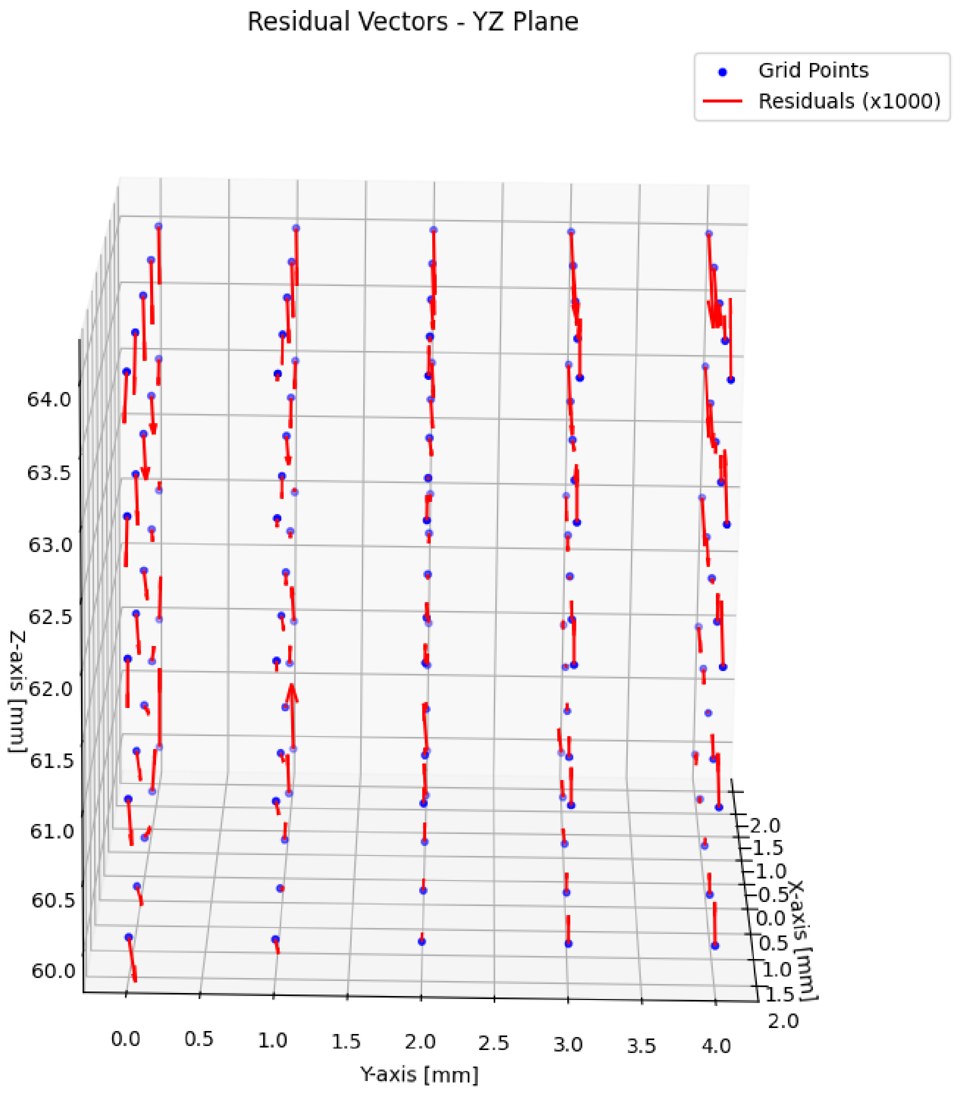


Figure 5.22 A map of residual vectors is plotted in the YZ plane

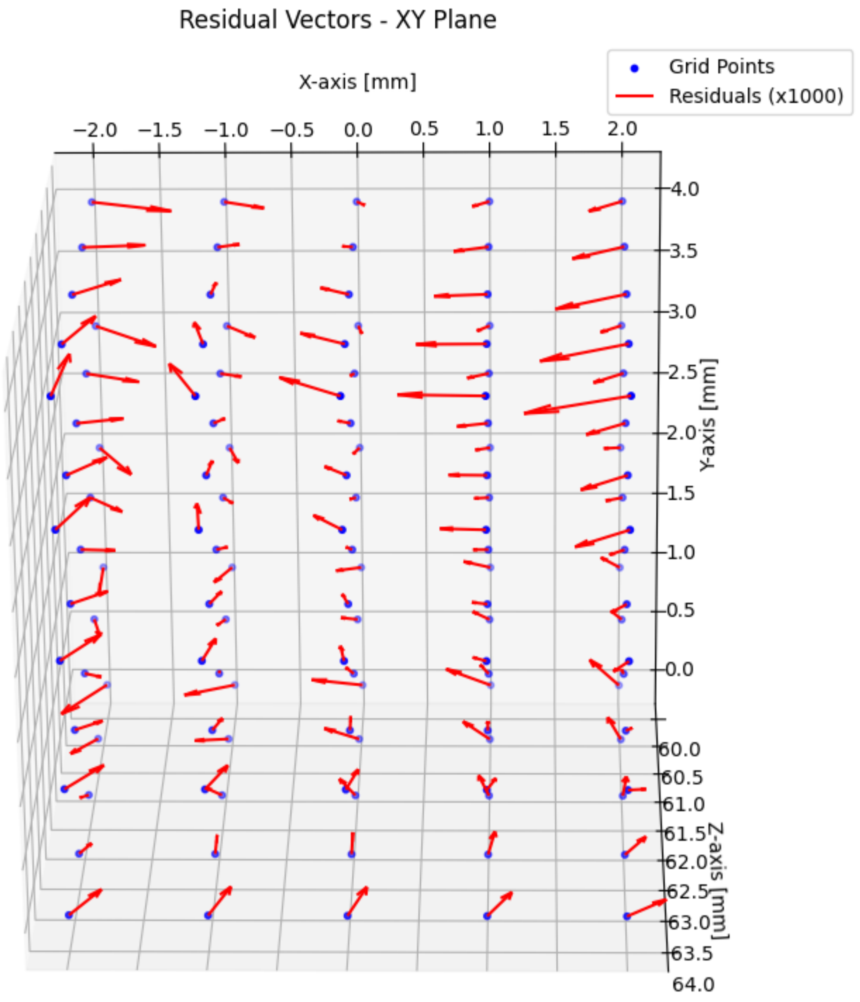


Figure 5.23 A map of residual vectors is plotted in the XY plane

5.3.3 Convergence behavior of the algorithm

The alignment of lines of sight (LOS) from the camera to known sphere center positions, expressed in the machine tool's reference frame, is formulated as a nonlinear least-squares optimization problem. Due to its non-convex nature [25], the optimization process is susceptible to convergence toward local minima, particularly in the presence of calibration inaccuracies, image noise, or geometric ambiguities.

To improve convergence toward the global minimum, the algorithm is initialized with a well-informed estimate of the camera pose, derived from known constraints of the setup. In this study, a 3D translator was used to mimic the motion of a calibration sphere, producing controlled and known displacements along predefined axes. This enabled accurate estimation of the sphere's center positions in the reference frame of the translator, which serves as a which act as a stand-in for the machine tool reference frame. The initial guess for the camera pose is constructed using prior information such as the nominal camera mounting orientation and the expected displacement path of the sphere.

Providing an estimate close to the true pose helps avoid convergence to incorrect local minima. Furthermore, structural constraints; such as the known, fixed radius of the calibration sphere and the linear motion imposed by the translator reduce ambiguity and increase the robustness of the optimization process.

5.3.4 Impact of camera lens choice on accuracy on the overall set-up

From an experimental perspective, the choice of camera lens significantly impacts the accuracy and practicality. Wide-angle lenses provide a broad field of view, enabling the capture of larger scenes or multiple calibration targets simultaneously, which is advantageous in space-constrained environments. However, these lenses typically introduce greater distortion, such as barrel distortion, which complicates calibration and can reduce measurement precision if not properly corrected. Regular, or standard, objective lenses offer a balanced field of view with lower distortion levels, making them well-suited for setups where the camera can be positioned at a moderate distance from the calibration target. This balance often results in more reliable and straightforward calibration processes. Telephoto lenses, on the other hand, provide a narrower field of view with higher effective resolution on distant objects, enhancing precision for detailed measurements. Nevertheless, their limited field of view requires careful positioning and can increase sensitivity to vibrations and small movements. Considering these factors, a standard objective lens is generally preferred for its optimal compromise between distortion, field coverage, and resolution, unless specific spatial constraints or precision requirements dictate the use of wide-angle or telephoto lenses.

CHAPTER 6 CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

This study has demonstrated the feasibility and limitations of using compact, cost-effective cameras for the calibration of machine tools and industrial robots. While the selected image sensor offers a theoretical spatial resolution (pixel size) of 1.5 µm, the system achieved an overall RMSE of 2 µm, with residual vector norms not exceeding 5.9 µm. These results highlight both the capabilities and constraints of the proposed vision-based approach under realistic experimental conditions.

A detailed analysis of the residual vectors representing the misalignment between reconstructed lines of sight and the actual 3D positions of target sphere centers revealed a radial distribution pattern. Residuals were generally smaller near the center of the field of view and increased toward the edges, suggesting spatially dependent alignment accuracy. This behavior is attributed to a combination of factors, including minor lens distortions, reduced calibration precision at the image periphery, slight variations in focal length, mechanical instabilities, and uneven illumination across the scene. Despite these influences, the residuals remained within a tight bound, confirming that system inaccuracies are not dominated by sensor resolution alone but by cumulative optical and environmental effects.

Calibration quality was also quantitatively verified. A sub-pixel mean reprojection error of 0.64 pixel was achieved denoting an acceptable calibration accuracy. In the other hand intrinsic parameters were also estimated with low standard errors of ± 2.41 pixels for the focal length and less than ± 1 pixel for the principal point. Low uncertainty was observed in the radial distortion parameters (e.g., ± 0.0012), and extrinsic parameters across different calibration poses showed standard errors typically below 0.1 mm. These values confirm strong internal consistency and reliable camera pose estimation.

Ultimately, this work emphasizes that reaching sub-5 µm accuracy in practical applications depends not only on sensor resolution but also on the overall quality of the system design. Future improvements should prioritize enhanced mechanical rigidity, more robust calibration procedures, improved lighting uniformity, and advanced image processing algorithms. By addressing these factors, the performance of low-cost optical systems can be further optimized for high-precision industrial calibration tasks.

By addressing each component of the system, from optics to setup stability to computational methods. It becomes possible to narrow the gap between theoretical resolution and operational accuracy, paving the way for reliable, sub-5 µm vision-based calibration systems for advanced manufacturing and robotic applications.

6.2 Recommendations

- Develop a sensing head that employs the triangulation principle to determine the 3D location of a precision sphere's center for calibrating a machine or industrial robot. This head could be stereoscopic or use three cameras to enhance accuracy and provide redundancy.
- Explore the use of a self-illuminating target, as discussed in various research papers. The aim of the self-illuminating target is to reduce image noise caused by lighting conditions.
- Seek a more compact image sensor or camera that maintains the same performance to reduce the packaging size.

REFERENCES

[1] S. Weikert, "R-test, a new device for accuracy measurements on five axis machine tools," *CIRP annals,* vol. 53, no. 1, pp. 429-432, 2004.

[2] C. Hong and S. Ibaraki, "Non-contact R-test with laser displacement sensors for error calibration of five-axis machine tools," *Precision Engineering,* vol. 37, no. 1, pp. 159-171, 2013.

[3] I. P. Engineering. "R-test instrument." IBS. http://www.ibspe.com (accessed 10/06/2024.

[4] S. Zargarbashi and J. Mayer, "Single setup estimation of a five-axis machine tool eight link errors by programmed end point constraint and on the fly measurement with Capball sensor," *International Journal of Machine Tools and Manufacture,* vol. 49, no. 10, pp. 759-766, 2009.

[5] T. Engel, "3D optical measurement techniques," *Measurement Science and Technology,* vol. 34, no. 3, p. 032002, 2022.

[6] R. Chen, J. Xu, and S. Zhang, "Comparative study on 3D optical sensors for short range applications," *Optics and lasers in engineering,* vol. 149, p. 106763, 2022.

[7] M.-E. USA, *Laser triangulation sensors*. 2015.

[8] S. Ekinovic, H. Prcanovic, and E. Begovic, "Calibration of machine tools by means of laser measuring systems," *Asian Trans. Eng,* vol. 2, no. 06, pp. 17-21, 2013.

[9] Etalon. "Etalon part of hexagon." Etalon https://hexagon.com/products/etalon (accessed 13/06/2024.

[10] W. Jywe, T.-H. Hsu, and C.-H. Liu, "Non-bar, an optical calibration system for five-axis CNC machine tools," *International Journal of Machine Tools and Manufacture,* vol. 59, pp. 16-23, 2012.

[11] I. 10791-6:2014. "ISO 10791-6:2014 Test conditions for machining centres Part 6: Accuracy of speeds and interpolations." ISO https://www.iso.org/standard/46440.html (accessed 13/06/2024.

[12] T.-H. Hsieh, W.-Y. Jywe, J.-J. Zeng, C.-M. Hsu, and Y.-W. Chang, "Geometric error compensation method using the Laser R-test," *The International Journal of Advanced Manufacturing Technology,* pp. 1-19, 2024.

[13] W. Cuypers, N. Van Gestel, A. Voet, J.-P. Kruth, J. Mingneau, and P. Bleys, "Optical measurement techniques for mobile and large-scale dimensional metrology," *Optics and Lasers in Engineering,* vol. 47, no. 3-4, pp. 292-300, 2009.

[14] M. Puttock, "Large-scale metrology," *Ann. CIRP,* vol. 21, no. 1, pp. 351-356, 1978.

[15] W. Gao *et al.*, "Machine tool calibration: Measurement, modeling, and compensation of machine tool errors," *International Journal of Machine Tools and Manufacture,* vol. 187, p. 104017, 2023.

[16] M. Švaco, B. Šekoranja, F. Šuligoj, and B. Jerbić, "Calibration of an industrial robot using

a stereo vision system," *Procedia Engineering,* vol. 69, pp. 459-463, 2014.

[17] R. Wang, A. Wu, X. Chen, and J. Wang, "A point and distance constraint based 6R robot calibration method through machine vision," *Robotics and Computer-Integrated Manufacturing,* vol. 65, p. 101959, 2020.

[18] K. Mori, D. Kono, and A. Matsubara, "Vision-based volumetric displacement measurement with a self-illuminating target," *CIRP Annals,* vol. 72, no. 1, pp. 305-308, 2023.

[19] C. Möller, H. C. Schmidt, N. H. Shah, and J. Wollnack, "Enhanced absolute accuracy of an industrial milling robot using stereo camera system," *Procedia Technology,* vol. 26, pp. 389-398, 2016.

[20] M. Grudziński and Ł. Marchewka, "A stereovision system for three-dimensional measurements of machines," *Zeszyty Naukowe Akademii Morskiej w Szczecinie,* no. 58 (130), pp. 16--23, 2019.

[21] W. Liu *et al.*, "Binocular-vision-based error detection system and identification method for PIGEs of rotary axis in five-axis machine tool," *Precision Engineering,* vol. 51, pp. 208-222, 2018.

[22] A. Traslosheros, J. M. Sebastián, J. Torrijos, R. Carelli, and E. Castillo, "An inexpensive method for kinematic calibration of a parallel robot by using one hand-held camera as main sensor," *Sensors,* vol. 13, no. 8, pp. 9941-9965, 2013.

[23] A. Tabia, "Pose estimation with event camera," Université Paris-Saclay, 2024.

[24] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose estimation from corresponding point data," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 19, no. 6, pp. 1426-1446, 1989.

[25] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[26] opencv. "Understanding Lens Distortion." https://learnopencv.com/understanding-lens-distortion/ (accessed 20/06/2024.

[27] T. Lei, Y. Rong, H. Wang, Y. Huang, and M. Li, "A review of vision-aided robotic welding," *Computers in Industry,* vol. 123, p. 103326, 2020.

[28] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 22, no. 11, pp. 1330-1334, 2000.

[29] Z. Zhang, "Calibration," in *Computer Vision: A Reference Guide*: Springer, 2021, pp. 117-118.

[30] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *International journal of computer vision,* vol. 4, no. 2, pp. 127-139, 1990.

[31] A. Wan, Y. Wang, G. Xue, K. Chen, and J. Xu, "Accurate kinematics calibration method for a large-scale machine tool," *IEEE Transactions on Industrial Electronics,* vol. 68, no. 10, pp. 9832-9843, 2020.

[32] G. P. Stein, "Accurate internal camera calibration using rotation, with analysis of sources

of error," in *Proceedings of IEEE International Conference on Computer Vision*, 1995: IEEE, pp. 230-236.

[33] W. Burger, "Zhang's camera calibration algorithm: in-depth tutorial and implementation," *HGB16-05,* pp. 1-6, 2016.

[34] H. Peng, L. Zhiwei, S. Zhenlian, G. Xinjian, and F. Jianguo, "A novel camera calibration method based on multiple calibration planes," presented at the Proceedings of the 2020 2nd International Conference on Big Data and Artificial Intelligence, 2020.

[35] Calib.io. "camera calibration chessboard pattern." Calib.io. https://calib.io/products/checkerboard?srsltid=AfmBOooLcuhhsx4GrybN1bCIqcttGKzpr4ns-57fFYllgtoo0_H4T6f_ (accessed 12/06/2024, 2024).

[36] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the seventh ieee international conference on computer vision*, 1999, vol. 1: Ieee, pp. 666-673.

[37] P. Kaur and R. Kant, "A review on: Comparison and analysis of edge detection techniques," *Int. Journal of Engineering Research and General Science,* vol. 2, no. 3, 2014.

[38] B. K. Shah, V. Kedia, R. Raut, S. Ansari, and A. Shroff, "Evaluation and comparative study of edge detection techniques," *IOSR Journal of Computer Engineering,* vol. 22, no. 5, pp. 6-15, 2020.

[39] R. Mwawado, B. Maiseli, and M. Dida, "Robust edge detection method for the segmentation of diabetic foot ulcer images," 2020.

[40] A. S. Ahmed, "Comparative study among Sobel, Prewitt and Canny edge detection operators used in image processing," *J. Theor. Appl. Inf. Technol,* vol. 96, no. 19, pp. 6517-6525, 2018.

[41] G. N. Chaple, R. Daruwala, and M. S. Gofane, "Comparisions of Robert, Prewitt, Sobel operator based edge detection methods for real time uses on FPGA," in *2015 International Conference on Technologies for Sustainable Development (ICTSD)*, 2015: IEEE, pp. 1-4.

[42] R. Maini and H. Aggarwal, "Study and comparison of various image edge detection techniques," *International journal of image processing (IJIP),* vol. 3, no. 1, pp. 1-11, 2009.

[43] K. Kanatani, Y. Sugaya, and Y. Kanazawa, *Ellipse fitting for computer vision: implementation and applications*. Morgan & Claypool Publishers, 2016.

[44] F. L. Bookstein, "Fitting conic sections to scattered data," *Computer graphics and image processing,* vol. 9, no. 1, pp. 56-71, 1979.

[45] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 21, no. 5, pp. 476-480, 1999.

[46] S. Pollard and J. Porrill, "Robust recovery of 3D ellipse data," in *BMVC92: Proceedings of the British Machine Vision Conference, organised by the British Machine Vision Association 22–24 September 1992 Leeds*, 1992: Springer, pp. 39-48.

[47] W. He, G. Wu, F. Fan, Z. Liu, and S. Zhou, "A Robust Real-Time Ellipse Detection Method

for Robot Applications," *Drones,* vol. 7, no. 3, p. 209, 2023.

[48] W. Gander, R. Strebel, and G. H. Golub, "Fitting of circles and ellipses least squares solution," in *SVD and Signal Processing III*: Elsevier, 1995, pp. 349-356.

[49] P. L. Rosin, "A note on the least squares fitting of ellipses," *Pattern Recognition Letters,* vol. 14, no. 10, pp. 799-808, 1993.

[50] R. Halır and J. Flusser, "Numerically stable direct least squares fitting of ellipses," in *Proc. 6th International Conference in Central Europe on Computer Graphics and Visualization. WSCG*, 1998, vol. 98: Citeseer, pp. 125-132.

[51] H. Ma. "Ellipse fit using geometric parameters based on Trust Region minimization scheme." https://connections.mathworks.com/matlabcentral/fileexchange/32107-fitting-an-ellipse-to-a-given-set-of-points-using-trust-region-method (accessed 18/12/2024.

[52] R. Matsuoka and S. Maruyama, "Eccentricity on an image caused by projection of a circle and a sphere," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* vol. 3, pp. 19-26, 2016.

[53] J. Liao, B. Buchholz, J.-M. Thiery, P. Bauszat, and E. Eisemann, "Indoor scene reconstruction using near-light photometric stereo," *IEEE Transactions on Image Processing,* vol. 26, no. 3, pp. 1089-1101, 2016.

[54] T. Tekla and H. Levente, "A Minimal Solution for Image-Based Sphere Estimation," ed: Springer, 2023.

[55] camcalib. "What is the Reprojection Error." camcalib. https://www.camcalib.io/post/what-is-the-reprojection-error (accessed 26/02/2024.

[56] M. H. center. "Using the Single Camera Calibrator App." Mathworks.inc. https://www.mathworks.com/help/vision/ug/using-the-single-camera-calibrator-app.html (accessed 2025).

APPENDIX A CALIBRATION PARAMETERS RESULTS AND COMPONENT SELECTION CRITERIA AND PROGRAMS

A.1 Recovered camera calibration parameters

The estimated camera calibration parameters are summarized in Table A.1.

Table A.1 Table Camera parameters recovered

```
                      Standard Errors of Estimated Camera Parameters
                      ---------------------------------------------
Intrinsics
----------
Focal length (pixels):   [ 2573.9990 +/- 2.4102     2572.6654 +/- 2.3419  ]
Principal point (pixels):[ 2034.8963 +/- 0.9878     1536.5415 +/- 0.7988  ]
Radial distortion:       [    0.0255 +/- 0.0012       -0.0937 +/- 0.0030  ]
Tangential distortion:   [    0.0006 +/- 0.0002        0.0005 +/- 0.0002  ]


Extrinsics
----------
Rotation vectors:
             [   -0.0595 +/- 0.0006     0.1071 +/- 0.0005    -0.0246 +/- 0.0001 ]
             [   -0.0615 +/- 0.0006    -0.1377 +/- 0.0005    -0.0208 +/- 0.0001 ]
             [    0.0170 +/- 0.0005     0.4447 +/- 0.0006     0.0002 +/- 0.0001 ]
             [    0.1905 +/- 0.0006    -0.0398 +/- 0.0005     0.0247 +/- 0.0001 ]
             [   -0.1590 +/- 0.0006     0.1687 +/- 0.0005     0.2331 +/- 0.0001 ]
             [   -0.0990 +/- 0.0005     0.2234 +/- 0.0005    -0.3149 +/- 0.0001 ]
```

```
          [  -0.4327 +/- 0.0007      0.2576 +/- 0.0008      2.8396 +/- 0.0002  ]
          [  -0.1765 +/- 0.0004      0.6362 +/- 0.0005     -0.0482 +/- 0.0001  ]
          [  -0.1847 +/- 0.0005      0.0408 +/- 0.0005     -0.0893 +/- 0.0001  ]
          [  -0.1783 +/- 0.0003      0.3234 +/- 0.0004     -0.0668 +/- 0.0001  ]


Translation vectors (millimeters):
          [  -24.6494 +/- 0.0288    -17.4141 +/- 0.0238     77.9059 +/- 0.0762  ]
          [  -22.9194 +/- 0.0286    -17.6938 +/- 0.0225     72.1630 +/- 0.0690  ]
          [   -9.0674 +/- 0.0314    -18.6811 +/- 0.0244     82.3218 +/- 0.0755  ]
          [  -34.7072 +/- 0.0301    -14.4345 +/- 0.0245     75.5960 +/- 0.0780  ]
          [  -23.6350 +/- 0.0315    -22.4800 +/- 0.0257     85.3764 +/- 0.0819  ]
          [  -36.5793 +/- 0.0321     -9.8090 +/- 0.0267     87.4158 +/- 0.0845  ]
          [   23.7541 +/- 0.0300      9.8001 +/- 0.0259     82.6757 +/- 0.0694  ]
          [  -15.8768 +/- 0.0366    -15.7174 +/- 0.0291     95.2653 +/- 0.0763  ]
          [  -39.1341 +/- 0.0305    -14.7652 +/- 0.0250     81.8767 +/- 0.0817  ]
          [  -32.9228 +/- 0.0253    -15.5991 +/- 0.0207     68.8671 +/- 0.0648  ]
```

A.2 Component's criteria selection

Table A.2 summarizes the component criteria for the camera type and lens.

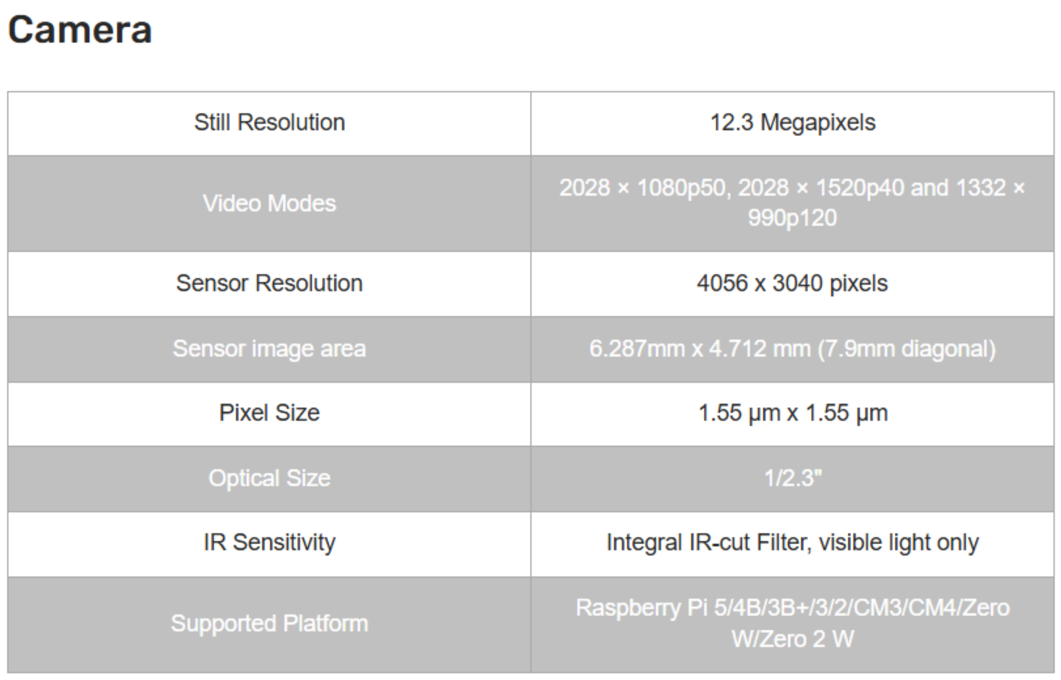
Table A.2 Criteria for selection of the camera type and lens

Criteria	Specifications	Values	Notes
Size	Minimal dimensions	<200*150mm (about 5.91 in)	
Output	File	PNG or JPG image file	

Precision	Camera resolution	Minimum 5MP	More pixels resolve in having a clearer image of the sphere
distortion	Amount of distortion	N/A	We need to minimize the distortion on the camera
Minimal distance	The minimal distance we can use is between the sphere and the camera for a clear image	30 cm (about 11.81 in) minimum but for this type of application less than 10 cm (about the length of the long edge of a credit card) should be ideal	
Image acquisition	Number of images that can take, save, and transferred	Not less than 100 images	
Automatization	N/A	N/A	Being able to take images without the intervention of a human being.
File transfer	Type of sharing data between the camera and the user	Wi-Fi or micro-SD card	
Portability	Minimal autonomy	2 hours maximum	This means we can use the sensor while the doors of the machine tool are closed
Dust and water jet	Protection of the electronic component	IP65	The sensor component should be protected against dust and water jet
LED lighting	Includes some LED if necessary	Yes	Being able to provide some source light during the images acquisition
Price	Total price	500$ CAD	

A.3 Camera specifications

Figure A.1 illustrates the main specifications of the camera sensor.



Camera

Still Resolution	12.3 Megapixels
Video Modes	2028 × 1080p50, 2028 × 1520p40 and 1332 × 990p120
Sensor Resolution	4056 x 3040 pixels
Sensor image area	6.287mm x 4.712 mm (7.9mm diagonal)
Pixel Size	1.55 μm x 1.55 μm
Optical Size	1/2.3"
IR Sensitivity	Integral IR-cut Filter, visible light only
Supported Platform	Raspberry Pi 5/4B/3B+/3/2/CM3/CM4/Zero W/Zero 2 W

Figure A.1 Raspberry PI HQ camera image sensor specifications

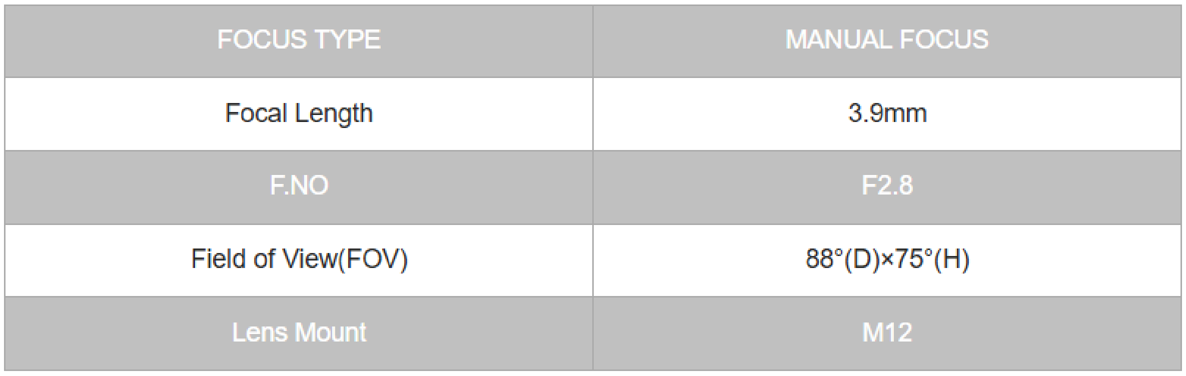
A.4 Choice of the lens

The lens chosen is a M12 lens, with less distortion:

It is an M12 camera lens, offering resolution up to 12.3MP, very compact and designed to limit distortion.

Figure A.2 A 75-degree cameras lens used on the Raspberry Pi HQ camera

Figure A.3 illustrates the main specifications of the M12 lens.



FOCUS TYPE	MANUAL FOCUS
Focal Length	3.9mm
F.NO	F2.8
Field of View(FOV)	88°(D)×75°(H)
Lens Mount	M12

Figure A.3 The lens specifications

Advantages:

Inexpensive ($16)

High quality

Small size

A.5 Motherboard selected

The selected motherboard is a Raspberry Pi, as shown in Figure A.4.

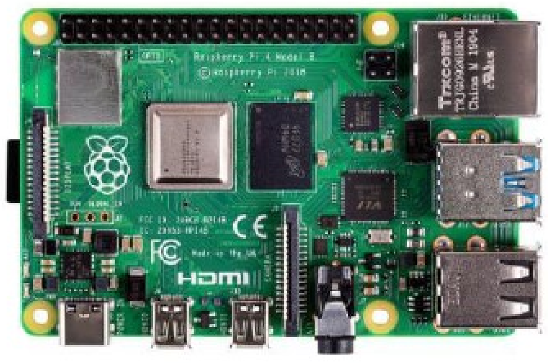


Figure A.4 Illustration of the Raspberry Pi with the included input and output ports

Figure A.5 illustrates the detailed pin layout of the Raspberry Pi.

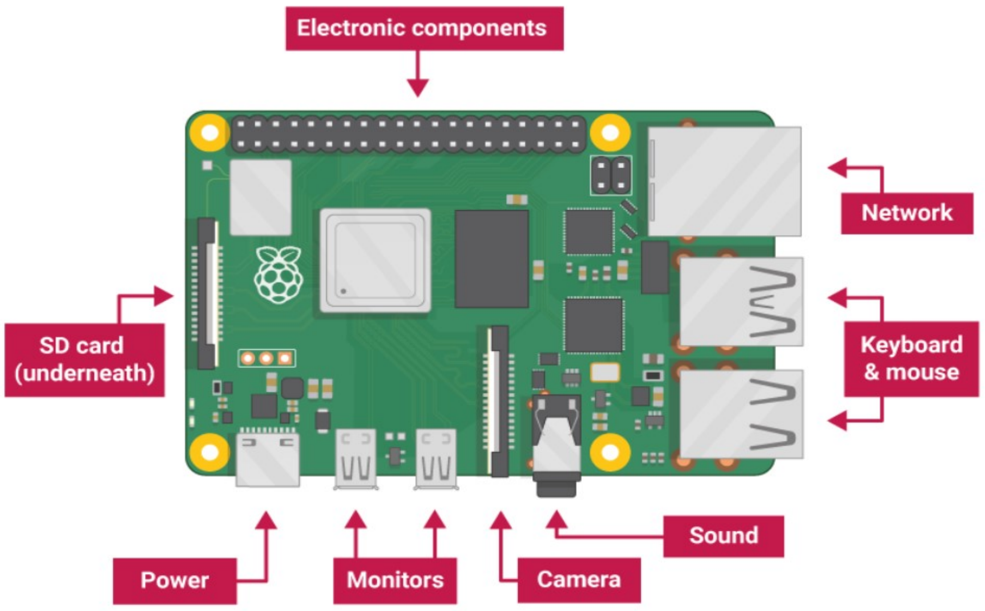


Figure A.5 The Raspberry Pi with detailed pin layout

Advantages:

Faster Processor

Multiple Sensors

Easy packaging

Cost 110$

Disadvantages:

Overheating, reliance on an SD card for storage.

A.6 Raspberry Pi HQ camera

This is the highest quality of camera for raspberry.

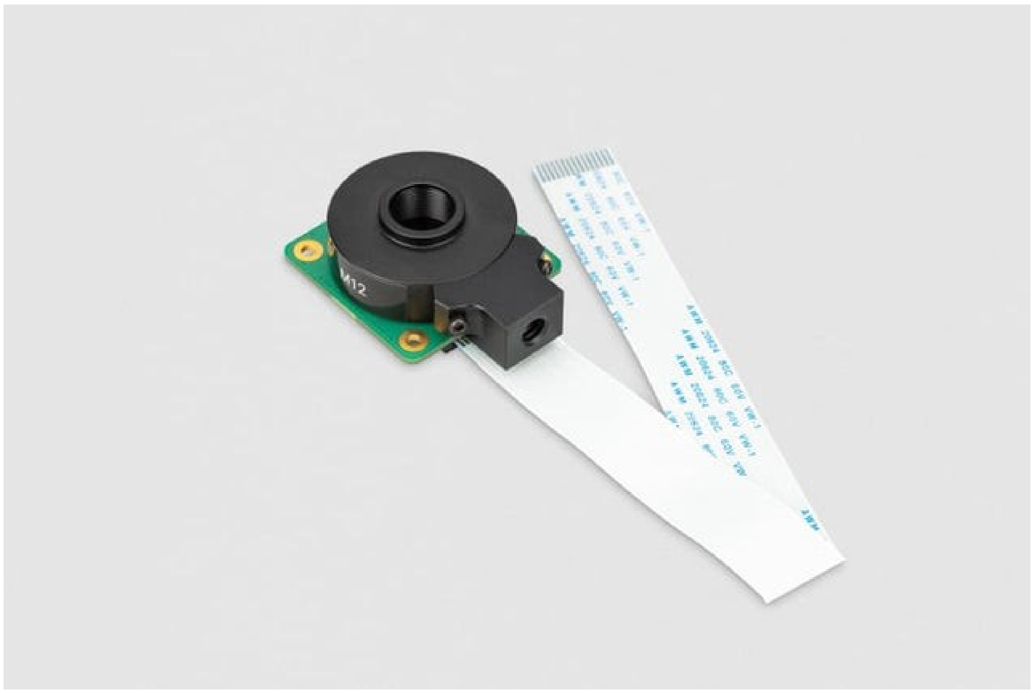


Figure A.6 Illustration of the Raspberry Pi HQ M12 mount-type

Advantages:

- Best quality on the market
- Customizable with different lens

Disadvantages:

- Expensive : 110$

A.7 Bill of materials

The bill of materials is presented below:

- Raspberry PI 4 + camera HQ M12 + M12 Lens+ Lithium-ion Battery + support:

Total: 291$ = 119$+100$ +16$ +38$+18$

- Procuration of component (website link)

1.https://abra-electronics.com/robotics-embedded-electronics/raspberry-pi-en-3/boards/sc1111pi5-4gb-raspberry-pi-5-4gb.html

2.https://abra-electronics.com/robotics-embedded-electronics/raspberry-pi-en-3/cameras-en-2/sc0870-official-raspberry-pi-high-quality-camera-module-for-m12-mount-lenses.html

3.https://abra-electronics.com/robotics-embedded-electronics/raspberry-pi-en-3/cameras-en-2/wave-23965-m12-high-resolution-lens-12mp-113-fov-2-7mm-focal-length-compatible-with-raspberry-pi-high-quality-camera-m12.html

4.https://abra-electronics.com/batteries-holders/battery-chargers-testers/2465-ada-powerboost-1000-charger-rechargeable-5v-lipo-usb-boost-1a-1000c.html

5.https://abra-electronics.com/batteries-holders/batteries-polymer-lithium-ion/bat-lipo-3.7-1200-lithium-ion-polymer-battery-3.7v-1200mah-clone.html

A.8 MATLAB-based Line of sight computation algorithm

```
function process_and_back_project_endpoints(imageDir, excelFilename, camera_params)
    % PROCESS_AND_BACK_PROJECT_ENDPOINTS
    % This function processes a series of images containing ellipses, fits each ellipse,
    % calculates the endpoints of the major axis, and back-projects these endpoints into
    % 3D rays using the camera intrinsic parameters. It estimates the cone axis from each
    % pose by averaging the direction vectors of these rays. The results are plotted and
    % saved to an Excel file.
    %
    % INPUTS:
    % - imageDir: Directory containing the image files (assumed format: 'image1.jpg', etc.).
    % - excelFilename: Full path to the Excel file where results will be stored.
    % - camera_params: Struct with fields 'focal_length_x', 'focal_length_y',
    %                  'principal_point_x', and 'principal_point_y'.
    %
    % OUTPUTS:
    % - Saved Excel file with ellipse and endpoint data.
    % - A 3D plot displaying cone axis vectors originating from the camera center.

    % Construct the intrinsic camera matrix K using focal lengths and principal point
    K_matrix = [camera_params.focal_length_x, 0, camera_params.principal_point_x;
                0, camera_params.focal_length_y, camera_params.principal_point_y;
                0, 0, 1];

    % Inverse of the intrinsic matrix, used to back-project image points into 3D rays
    K_inv = inv(K_matrix);

    % Prepare data storage for ellipse parameters and endpoints
    numImages = length(imageDir);  % imageDir should be a list or count of images
    data = cell(numImages, 8);     % Each row: filename, center x/y, axis, endpoints

    % Create figure for 3D visualization of cone axes
    figure;
    hold on;

    % Loop through each image
    for i = 1:numImages
        % Generate full path to image file
        filename = fullfile(imageDir, ['image' num2str(i) '.jpg']);

        % Fit ellipse to the image content (your own implementation)
        [ellipse_params, status] = fit_ellipse_from_file(filename);

        % Skip this iteration if ellipse fitting failed
        if isempty(ellipse_params)
            disp(['Failed to fit an ellipse for image ' num2str(i)]);
            continue;
        end

        % Extract ellipse parameters
        fitted_center_x = ellipse_params.X0_in;
        fitted_center_y = ellipse_params.Y0_in;
        semi_major_axis = ellipse_params.long_axis / 2;     % Convert to semi-major axis
        orientation = ellipse_params.phi;                    % Orientation angle (radians)

        % Compute endpoints of the semi-major axis in image coordinates
        end_point_1_x = fitted_center_x + semi_major_axis * cos(orientation);
        end_point_1_y = fitted_center_y + semi_major_axis * sin(orientation);
        end_point_2_x = fitted_center_x - semi_major_axis * cos(orientation);
        end_point_2_y = fitted_center_y - semi_major_axis * sin(orientation);

        % Store results in cell array for Excel export
        data{i, 1} = filename;
        data{i, 2} = fitted_center_x;
        data{i, 3} = fitted_center_y;
        data{i, 4} = semi_major_axis;
        data{i, 5} = end_point_1_x;
        data{i, 6} = end_point_1_y;
        data{i, 7} = end_point_2_x;
        data{i, 8} = end_point_2_y;

        % Prepare homogeneous image coordinates for each endpoint
        endpoint_1_image = [end_point_1_x; end_point_1_y; 1];
        endpoint_2_image = [end_point_2_x; end_point_2_y; 1];

        % Back-project to 3D rays by applying the inverse of K
        ray_1_3D = K_inv * endpoint_1_image;
        ray_2_3D = K_inv * endpoint_2_image;

        % Normalize the rays to unit direction vectors
        ray_1_3D = ray_1_3D / norm(ray_1_3D);
        ray_2_3D = ray_2_3D / norm(ray_2_3D);

        % Average the two rays to define the cone axis direction
        cone_axis = (ray_1_3D + ray_2_3D) / 2;

        % Plot the cone axis as a line from camera center to the 3D direction
        plot3([0, cone_axis(1)], [0, cone_axis(2)], [0, cone_axis(3)], 'b-o', 'LineWidth', 2);

        % Print cone axis to command window
        disp(['Pose ' num2str(i) ' - Cone Axis:']);
        disp(cone_axis);
    end

    % Finalize and label the 3D plot
    xlabel('X');
    ylabel('Y');
    zlabel('Z');
    title('Cone Axes from Camera Origin');
    grid on;
    axis equal;
    % Export collected data to Excel
    xlswrite(excelFilename, data, 'Sheet1');
end

% Dummy ellipse fitting wrapper for integration purposes
function [ellipse_params, status] = fit_ellipse_from_file(filename)
    % Example wrapper that fits a conic from ellipse boundary points
    % Inputs:
    %   filename - name of a .mat file containing 'x' and 'y' vectors (Nx1)
    % Outputs:
    %   ellipse_params - structure with algebraic and geometric parameters
    %   status - 'Success' or 'Failed'

    % Load x and y (should be Nx1 column vectors)
    data = load(filename);
    if ~isfield(data, 'x') || ~isfield(data, 'y')
        status = 'Failed';
        ellipse_params = [];
        return;
    end

    x = data.x;
    y = data.y;
    if numel(x) < 5 || numel(y) < 5
        status = 'Failed';
        ellipse_params = [];
        return;
    end

    % Fit conic
    XY = [x(:), y(:)];
    ParAini = [1, 0, 1, 0, 0, -1]';  % Reasonable starting point (circle-like)
    LambdaIni = 0.1;
    [ParA, RSS, iters, code] = fit_conic(XY, ParAini, LambdaIni);

    if code ~= 1  % Only continue if conic is a valid ellipse
        status = 'Failed';
        ellipse_params = [];
        return;
    end

    % Convert to geometric parameters
    [ParG, ~] = AtoG(ParA);  % [a, b, theta, x0, y0]

    ellipse_params.algebraic = ParA;
    ellipse_params.geometric = struct(...
        'a', ParG(1), 'b', ParG(2), 'theta', ParG(3), ...
        'xc', ParG(4), 'yc', ParG(5));
    ellipse_params.RSS = RSS;
    ellipse_params.iters = iters;

    status = 'Success';
```

```
end
```

A.9 Rigid body adjustment program algorithm

```
function [R, T, k, rmse_values, new_distances,distance_vectors] = bundle_adjustement(points,
direction_vectors, tau, origin, T_initial)
% DEVELOPED BY KANDOLO JIRE CHRISTIAN 2024
% This MATLAB function performs a rigid body adjustment.
% It iteratively estimates the rotation (R) and translation (T)
% that best align a point cloud with a corresponding set of 3D lines.
% The goal is to minimize the sum of squared perpendicular distances
% from each transformed 3D point to its associated 3D line (RMSE).
% origin:origin of the camera of the camera
%T_initial : intial transformation matrix
%tau : stopping criteria
%direction_vectors : Line of sight
% points : Grid
k = 0;
last_rmse = 0;
rmse_values = [];
T_matrix = eye(4)

while true
    % Step 1: Compute scalar distances from each point to its corresponding line
    di = calculate_distances(points, origin, direction_vectors);

    % Step 2: Compute normal vectors and closest points on the lines
    [ni, Q] = calculate_closest_points(points, origin, direction_vectors, T_initial, T_matrix, k
+ 1);

    % Step 3: Estimate rigid transformation parameters
    [R,T,rotation_matrix] = rigid_adjustment(di, points, ni);

    % Step 4: Apply transformation to points
    transformedPoints = apply_registration(points, rotation_matrix, T);

    % Step 5: Evaluate alignment quality (RMSE)
    [rmse, new_distances] = compute_rmse(transformedPoints, origin, direction_vectors);
    rmse_values = [rmse_values; rmse];

    % Step 6: Check for convergence
    if abs(rmse - last_rmse) < tau
        break;
    end
    last_rmse = rmse;
    points = transformedPoints;
    k = k + 1;
    if k >=15
        disp('Maximum iterations reached without convergence.');
        break;
    end
end
end

function distances = calculate_distances(points, origin, direction_vectors)
% Calculates perpendicular distances from points to lines.
    points = points';
    direction_vectors = direction_vectors';
    num_points = size(points, 1);
    distances = zeros(num_points, 1);
    for i = 1:num_points
        origin_to_point = points(i, :) - origin;
        distances(i) = norm(cross(origin_to_point, direction_vectors(i, :)));
    end
end

function [ni, Q_original] = calculate_closest_points(points, origin, direction_vectors,
T_initial, T_matrix, k)
% Compute closest points on 3D lines and direction vectors ni from each point to its line
% All directions and origins are assumed to be in the same frame as points.
% 'ni' is calculated from original points to the closest points transformed back to the original
frame.

nPoints = size(points, 2);
P = points;  % 3xN original points
Q = zeros(3, nPoints);        % Closest points in transformed frame
Q_original = zeros(3, nPoints);  % Closest points transformed back to original frame
ni = zeros(3, nPoints);       % Normalized direction vectors

% Determine full transformation matrix
if k == 1
    T_combined = T_initial;
else
    T_combined = T_initial * T_matrix;
end

% Apply transformation to original points
homogP = [P; ones(1, nPoints)];         % 4xN
P_transformed = T_combined * homogP;    % 4xN
P_t = P_transformed(1:3, :);            % 3xN

% Loop through transformed points
r0 = origin';                           % Line origin (3x1)
for i = 1:nPoints
    Pi = P_t(:, i);                     % Transformed point (3x1)
    v = direction_vectors(:, i);       % Direction vector (3x1)

    t = dot(Pi - r0, v) / dot(v, v);    % Scalar projection onto line
    Qi = r0 + t * v;                    % Closest point on line in transformed frame
    Q(:, i) = Qi;
end

% Transform Q back to original frame using inverse transform
Q_homog = [Q; ones(1, nPoints)];                      % 4xN
Q_original_homog = inv(T_combined) * Q_homog;         % 4xN
Q_original = Q_original_homog(1:3, :);                % 3xN

% Compute ni using original points and back-transformed Q
for i = 1:nPoints
    ni_vec = Q_original(:, i) - points(:, i);
    ni(:, i) = ni_vec / norm(ni_vec);   % Normalized vector
end

end


function [R, T, rotation_matrix,T_matrix] = rigid_adjustment(di, points, ni)
% Computes the optimal rotation and translation minimizing the point-to-line distances.
% Inputs:
%   di     - distances from points to lines
%   points - 3D points (3 x N)
%   ni     - unit direction vectors from point to closest point on lines (3 x N)
% Outputs:
%   R - small rotation vector (3x1)
%   T - translation vector (3x1)
%   rotation_matrix - rotation matrix from R

% Ensure correct shapes
P = points';       % N x 3
ni = ni';          % N x 3

% Extract components
xv = ni(:, 1);
yv = ni(:, 2);
zv = ni(:, 3);
xi = P(:, 1);
yi = P(:, 2);
zi = P(:, 3);

% Cross product terms for rotation part
term_3 = yi .* zv - zi .* yv;
term_4 = -(xi .* zv - zi .* xv);
term_5 = xi .* yv - yi .* xv;

% Build Jacobian matrix J and solve
J = [xv, yv, zv, term_3, term_4, term_5];
t = pinv(J) * di;

% Translation and rotation vector
T = t(1:3);
R = t(4:6);
```

```
% Convert small rotation vector to rotation matrix
alpha = (R(1));  % rotation about x
beta  = (R(2));;  % rotation about y
gamma = (R(3));;  % rotation about z

Rx = [1, 0, 0;
      0, cos(alpha), -sin(alpha);
      0, sin(alpha),  cos(alpha)];

Ry = [cos(beta), 0, sin(beta);
      0, 1, 0;
     -sin(beta), 0, cos(beta)];

Rz = [cos(gamma), -sin(gamma), 0;
      sin(gamma),  cos(gamma), 0;
      0, 0, 1];

rotation_matrix = Rz * Ry * Rx; % ZYX order

    % Convert rotation vector into matrix
    R = rotation_matrix
    % Build full 4x4 transformation matrix
    T_matrix = eye(4);
    T_matrix(1:3, 1:3) = R;
    T_matrix(1:3, 4) = T(:);  % Ensure T is a column vector
end

function [transformedPoints,transformationMatrix] = apply_registration(points, rotation_matrix,
T)
% Applies the estimated rigid transformation to a set of 3D points.
    nPoints = size(points, 2);
    homogeneousPoints = [points; ones(1, nPoints)];
    transformationMatrix = [rotation_matrix, T; 0, 0, 0, 1];
    transformedHomogeneous = transformationMatrix * homogeneousPoints;
    transformedPoints = transformedHomogeneous(1:3, :);
end
function [rmse, new_distances] = compute_rmse(transformedPoints, origin, direction_vectors)
% Computes the RMSE.
    points = transformedPoints';
    direction_vectors = direction_vectors';
    num_points = size(points, 1);
    new_distances = zeros(num_points, 1);
    for i = 1:num_points
        origin_to_point = points(i, :) - origin;
        dir_vec = direction_vectors(i, :);
        direction_length = norm(dir_vec);
        cross_vec = cross(origin_to_point, dir_vec);
        new_distances(i) = norm(cross_vec) / direction_length;
    end
    rmse = sqrt(mean(new_distances .^ 2));
end
```

A.10 Raspberry Pi set up program algorithm

- **Setup steps:**

1. **Install Raspberry Pi OS and Set up the camera module:** Follow https://www.raspberrypi.com/documentation/computers/configuration.html . Ensure the camera interface is enabled in raspi-config.
2. **Install the picamera library:** Open the terminal on your Raspberry Pi and install the picamera library using pip:

```bash
sudo apt update
sudo apt install python3-picamera
```

3. **Python script :** Create a new Python script (e.g., `capture_image.py`) and use the following code to capture an image:

•

```python
import cv2

# Open the camera (0 = default camera, 1 = external camera)
cap = cv2.VideoCapture(0)

if not cap.isOpened():
    print("Error: Could not open camera.")
    exit()

# Capture a single frame
ret, frame = cap.read()

if ret:
    # Save the captured image
    filename = "captured_image.jpg"
    cv2.imwrite(filename, frame)
    print(f"Image saved as {filename}")
else:
    print("Error: Could not capture image.")

# Release the camera
cap.release()
cv2.destroyAllwidndows()
```