ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Master Thesis

Active Camera Calibration for Robotic Systems

Spring Term 2016

Supervised by:
Enric Galceran Yebenes
Thomas Schneider

Author:
Ricard Boada Farràs

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Active Camera Calibration for Robotic Systems

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

Ricard	Boada Farràs

Student supervisor(s)

Enric	Galceran Yebenes
Thomas	Schneider

Supervising lecturer

Roland	Siegwart

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/plagiarism-citationetiquette.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Place and date	Signature

[1] Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Abstract

Camera calibration is an essential process embedded in any computer vision system that involves the extraction of spatial information from captured images. Traditional techniques for calibration require an expert to take several pictures from a target from different viewpoints. However, they usually suffer from lack of repeatability and insufficient accuracy if the selection of poses is not correctly done

This thesis seeks to make a completely autonomous camera calibration system by using an active robotic system in front of a fixed calibration target. The proposed method aims at calibrating a camera with the minimum number of poses required by selecting at each step the next best pose as the one that reduces the variance of the parameters the most. The algorithm consists of a deterministic initialization of camera parameters based on the maximization of distance among utilized poses and the maximum expected reprojection error algorithm for selecting next poses.

The results obtained confirm a promising performance of the algorithm in terms of the three categories considered: repeatability, fast convergence and high accuracy. Further experiments should be conducted with other robotic systems in order to assess the suitability of the method to replace current manual calibration techniques.

Acknowledgements

First of all, I would like to express my gratitude to Enric Galceran, who made my wish possible to come to the Autonomous Systems Lab (ASL) at ETH Zürich to write my Master's thesis in such a cutting-edge research group in robotics.

I want to give special thanks to my home university supervisor Ferran Marquès, and also to Xavier Giró, Miquel Àngel Farré, Hilario Tomé and Dr. Javier Alonso-Mora, who were very helpful and kind during my search of a research group. Without them probably I would not have been able to produce this work.

Additionally, I am very grateful to Thomas Schneider, my second supervisor, who has always given me good advice, and the people at Zürich Eye, specially Janosch Nikolic and Jörn Rehder, who always offered a hand with my calibration experiments. Also, thanks to my office colleagues who made my experience in the ASL very pleasant.

Finally, I could not finish these acknowledgments without mentioning my family and friends. Thank you very much for always supporting me.

Table of Contents

List of Figures

List of Tables

Symbols

Intrinsic and lens distortion parameters

f_x, f_y	focal length in both axes
u_0, v_0	principal point or image center
γ	skew coefficient
k_1, k_2	radial distortion coefficients
p_1, p_2	tangential distortion coefficients
K	intrinsic or projection matrix

Frame conventions

T_{AB}	transformation matrix from frame B to frame A
R_{AB}	rotation matrix from frame B to frame A
${}_At_{AB}$	vector from frame A to frame B expressed in frame A
${}_AM$	3D point expressed in frame A
m	2D point in the image plane
M', m'	3D and 2D points expressed in homogeneous coordinates respectively
T	target frame
B	robot base frame
W	robot wrist frame
C	camera frame

Other symbols

$\sigma_{f_x}, \sigma_{f_y}$	standard deviation of focal length

Acronyms and Abbreviations

ASL	Autonomous Systems Lab
ERE	Expected Reprojection Error
ETH	Eidgenössische Technische Hochschule
LED	Light Emitting Diode

MAV	Micro Aerial Vehicle
MRE	Mean Reprojection Error
MSE	Mean Squared Error
NN	Nearest-Neighbour
PnP	Perspective-n-Point
RANSAC	RANdom SAmple Consensus
RE	Reprojection Error
ROS	Robot Operating System
SLAM	Simultaneous Localization And Mapping

Chapter 1

Introduction

Camera calibration is a fundamental procedure in the field of computer vision that allows being able to interpret spatial information, e.g. depth or height of objects, from captured two-dimensional images. This process is also called *camera resectioning*, and consists on computing the specific parameters that characterize the camera lens and image sensor, with the possibility to account also for lens distortion coefficients, whenever a camera is going to be used for image or video interpretation purposes.

One of the immediate applications that come to one's mind when talking about camera calibration is the ability to correct the distortion of particular images, specially now that a huge camera market is available. Besides that, calibration of cameras is the first step to be done in any field involving the extraction of valuable information from images. These are applications highly researched nowadays, including perception for robotics, navigation systems in autonomous driving and 3D scene reconstruction for example.

Typically the process of calibrating a camera is manual. An expert user takes many images of an object of known geometry with the camera, typically a checkerboard with certain cell dimensions and spacing, from different points of view. Then, traditional algorithms are able to find correspondences between the detected target in the captured images and the known structure of the object, and from this rela-

Figure 1.1: Sample dataset used to calibrate a camera using a checkerboard. Source: *ROS.org*

Figure 1.2: ABB IRB 120 is the robotic arm used in this thesis for evaluation.

tion extract the value of the calibration parameters. For example, in the case of a checkerboard the algorithm would aim at detecting from the image the location of the corners for each cell and relate them to the previously stored geometry of the object, as shown in Figure 1.1.

This technique is still being widely used by the research community and many frameworks are available implementing it, like OpenCV, OpenGL and Matlab. However, the performance of these algorithms greatly depend on the selection of poses and usually results in low quality if the user is not an expert in the field. Furthermore, the procedure is also highly time-consuming and cumbersome, specially for new users.

1.1 Contribution

Nowadays the vast majority of camera calibration algorithms publicly available is passive, *i.e.* requiring the intervention of a human to move the target in front of a fixed camera. These procedures usually produce low accuracy results and take a long time to converge.

Conversely, the aim of this thesis is to propose a novel camera calibration method which is active, that is, a robotic system is holding the camera and moves it to certain poses automatically. These active techniques do not require the intervention of a human in a decisive manner. Therefore, highly repeatable results can be achieved, which is one of the main drawbacks of currently applied manual techniques.

Also, the process is optimized in order to visit only the poses that maximize the accuracy of the obtained camera parameters with the purpose of ensuring a fast convergence of the algorithm. Moreover, we have decided upon the use of AprilTags instead of other targets such as checkerboards. Unlike other approaches, AprilTags allow for detection even when individual corners are occluded.

This master thesis includes results with both simulated and real data. For the sake of demonstration, the work here has been proved to operate using the arm-robot shown in Figure 1.2, although it can be transported without much difficulty to any other robotic system.

1.2 Overview

The remainder of this thesis is organized as follows:

Chapter 2. Background.
This chapter aims to provide the reader with all the important background information needed to understand the following chapters and the work done. Also, the state-of-the-art research on camera calibration is discussed with a focus on the strengths and weaknesses of each available technique.

Chapter 3. Camera calibration.
The work done in the thesis is presented. This includes the batch and incremental camera calibration routines, which help to better introduce the active camera calibration algorithm proposed. For all approaches, the different steps are detailed in order to allow later reproductions.

Chapter 4. Experiment results.
This chapter shows the results obtained using the algorithm proposed in a series of conducted experiments. Furthermore, a comparison with other similar policies and traditional camera calibration techniques is provided.

Chapter 5. Conclusions.
Finally, the conclusions of the thesis are presented, paying special attention to the contributions and achievements. Also, some ideas on future work directions are given to expand the research on this topic.

Chapter 2

Background

Before going into more detail with the contents of the thesis some basics of camera calibration must be given. Also, the state-of-the-art algorithms in this field are carefully analyzed in this chapter.

2.1 Basics of camera calibration

Camera calibration or camera resectioning is the process of estimating the parameters that characterize a certain camera model[1] in order to extract 3D metric information from two-dimensional captured images. The choice on the camera model is dependent on the targeted application, but essentially they can be classified according to their capability to capture perspective or not.

2.1.1 Pinhole camera model

In this thesis we focus on the simplest perspective camera model available but also the most used one, called *pinhole* camera model. A deep analysis on all the camera models available can be found in [1] [2]. In the pinhole model, the image plane is placed at some distance from the camera center point as shown in Figure 2.1. The image produced contains the projection of all points in the environment whose straight line with the camera center intersects the image plane. In the pinhole camera model there are five parameters defining the characteristics of the camera, called *intrinsic* parameters:

- focal length in both axes (f_x and f_y),
- 2D principal point (u_o and v_o), and
- skew coefficient (γ).

These parameters are gathered in the intrinsic or projection matrix, K, which relates a 3D point in the camera frame, denoted by ${}_cM = [{}_cx, {}_cy, {}_cz]^T$ in meters, to 2D image points in pixels, denoted by $m = [u, v]^T$ in the image plane. Eq. (2.1) describes the transformation procedure between these two frames, also called *projection*.

$$m' = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K\,{}_CM = \begin{bmatrix} f_x & \gamma & u_o \\ 0 & f_y & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}_cx \\ {}_cy \\ {}_cz \end{bmatrix} \tag{2.1}$$

[1] A **camera model** is a function that maps 3D world points onto the 2D image plane, and is designed to closely model the physical camera.

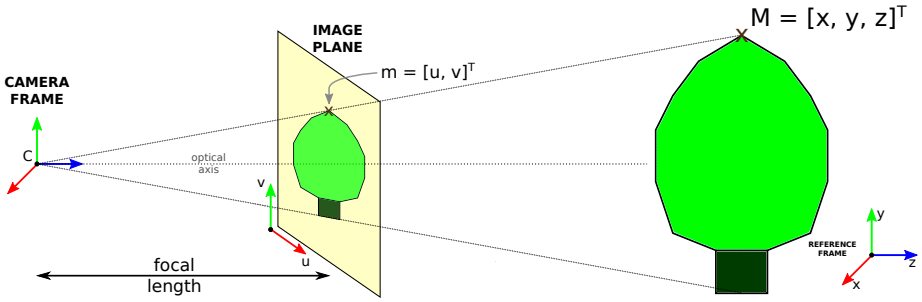

Figure 2.1: Description of different frames in the pinhole camera model.

In order to be able to deal with a transformation from a 3D point to a 2D point we need to introduce the concept of *homogeneous or projective coordinates*. While in the Euclidean space any two parallel lines lying in the same plane will never intersect, in the projective space parallel lines always intersect in a vanishing point. Homogeneous coordinates represent N-dimensional coordinates with N+1 coefficients as expressed in (2.2), allowing to work with points at infinity if $w = 0$. Throughout this thesis, points expressed in homogeneous coordinates will be denoted by the *prime* symbol ('), such that point m in Euclidean coordinates become point m' in homogeneous coordinates.

$$m = [u, v]^T \rightarrow m' = [x_1, x_2, x_3]^T \text{ any such that } u = \frac{x_1}{x_3} \text{ and } v = \frac{x_2}{x_3}. \tag{2.2}$$

Once we have the point expressed in the camera frame (${}_cM$) we need to transform it to the world reference frame (${}_wM$). This is done using a combination of rotation and translation between both frames, as described in (2.3).

$${}_CM = \begin{bmatrix} {}_cx \\ {}_cy \\ {}_cz \end{bmatrix} = \begin{bmatrix} R_{CW} & {}_Ct_{CW} \end{bmatrix} {}_WM' = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} \tag{2.3}$$

R_{CW} is a 3x3 matrix that symbolizes the rotation of vectors from the world reference frame to the camera frame. On the other hand, ${}_Ct_{CW}$ is the 3x1 vector that represents the translation from the camera to the world frame expressed in camera frame.

Finally, the whole transformation of points in 3D world frame to the 2D image plane can be summarized as outlined in (2.4).

$$m' = K \begin{bmatrix} R_{CW} & {}_Ct_{CW} \end{bmatrix} {}_CM = \begin{bmatrix} f_x & \gamma & u_o \\ 0 & f_y & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} \tag{2.4}$$

2.1.2 Lens distortion model

In addition to computing the camera parameters, we can also account for the lens distortion in the same optimization. Again, there exist many models for distortion, like the *equidistant*, *fish-eye* and *radial-tangential*. In this work we have considered only the radial-tangential model, which takes into account the two most relevant distortion effects. A much deeper description of other distortion models can be found in [3].

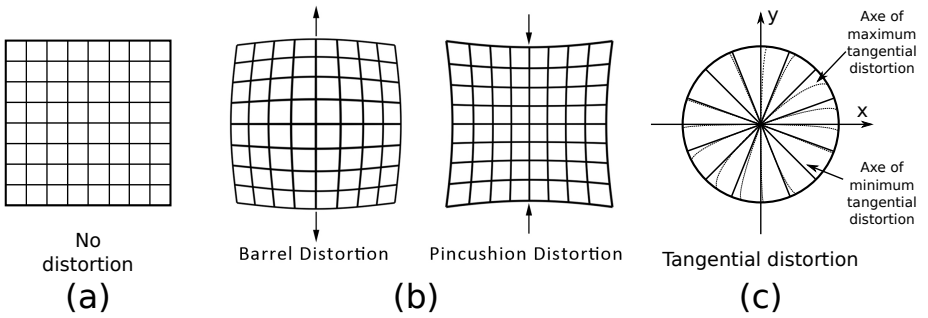

Figure 2.2: Distortion models considered. (a) shows a grid without any distortion type, (b) displays the two different radial distortion models, and (c) exhibits the effect of tangential distortion on an image.

As shown in Figure 2.2 (b), radial distortion occurs when light rays bend more near the edges of a lens than close to its optical center. We can distinguish two types of radial distortion according to its shape: barrel and pincushion distortions. In barrel distortion image magnification decreases with distance from optical axis, while in pincushion distortion the effect is the opposite.

On the other hand, tangential distortion is the effect of having the image plane and the lens plane not parallel to each other. This produces the effect depicted in Figure 2.2 (c), where the concentric axes are not exactly straight and suffer from little deviation depending on its angle.

The radial-tangential model can be solidly approximated by four parameters (k_1, k_2, p_1 and p_2). The first two parameters account for the radial distortion as described in (2.5), while the last two represent the tangential model, (2.6). Although in each case more coefficients could be taken, working with the main two gives sufficiently good results.

$$\begin{aligned} x_{corrected\_rad} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + ...) \\ y_{corrected\_rad} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + ...) \end{aligned} \tag{2.5}$$

$$\begin{aligned} x_{corrected\_tan} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{corrected\_tan} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \tag{2.6}$$

The procedure to correct the lens distortion effects is iterative. First, we compute the undistorted point due to radial distortion, and then using this corrected point we undistort the tangential distortion. This is usually done after projecting the points in the image plane.

2.2 State of the art

Camera calibration techniques are very variate in terms of calibration targets used, optimization methods adopted and camera and distortion models considered. However, most of them are based on the use of multiple views of a planar target, typically a checkerboard [4] [5] [6]. This is also the case for the popular toolboxes for MATLAB [7] and OpenCV [8].

Other approaches consider the usage of different calibration targets, such as a moving LED for wide-area multiple-cameras environment [9] and a custom-made cali-

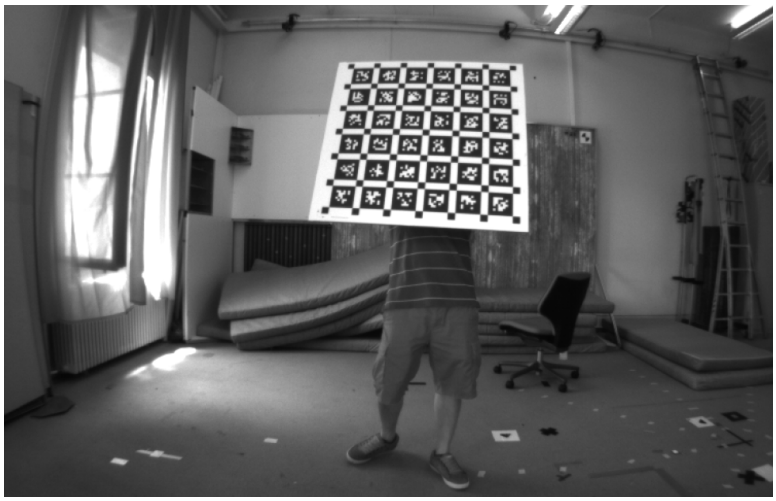

Figure 2.3: Camera calibration using Apriltags as calibration target.

bration cube [10]. It is even possible to calibrate a camera without using any target by capturing several images with the same camera and making correspondences between salient points in them [11], although it does not always provide reliable results.

Finally, a recent researched technique include the utilization of a novel visual fiducial system shown in Figure 2.3, called AprilTags [12]. These particular tags, similar in shape to QR codes, can be individually detected under severe lightning conditions and extreme viewing angles. Moreover, its special structure allows calibration frameworks to work with partially observed targets [13].

Most calibration algorithms built upon the work by Zhang [4], which consists in computing an analytical closed-form solution of the parameters followed by a non-linear optimization based on maximum-likelihood criterion. The analytical solution is found by assuming that the target plane is on $Z = 0$ and calculating the homography between the target plane and the image plane. Then, by imposing orthonormality among columns in the rotation matrix we can arrive to a closed-form of the intrinsic camera parameters. The subsequent maximum-likelihood estimation minimizes the squared distance between the detected target points in the image with the reprojected target points using the estimated target pose, as described in (2.7), where m_{ij} denotes location of corner j in image i and $\hat{m}(K, R_i, t_i, M_j)$ is the projection of corner M_j on the image plane given intrinsic matrix K, rotation matrix R_i and translation t_i. This optimization is used to refine the accuracy of the calibration and at the same time include other effects such as lens distortion. Zhang also introduces a second procedure to estimate the lens distortion by alternation [4], although its convergence is slow. These approaches involving a non-linear optimization can be easily implemented using a sparse matrix solver, similar to the ones used in bundle adjustment problems [1] [14] [15] [16].

$$\sum_{i=1}^{n}\sum_{j=1}^{m} ||m_{ij} - \hat{m}(K, R_i, t_i, M_j)||^2 \tag{2.7}$$

Regarding methods not including a maximum-likelihood optimization, a proposal of a different method to estimate the camera parameters of a fish-eye lens, based on the estimation of vanishing points, is given in [6]. Finally, for wide-area applications another approach suggest working with a jointly estimation of camera and

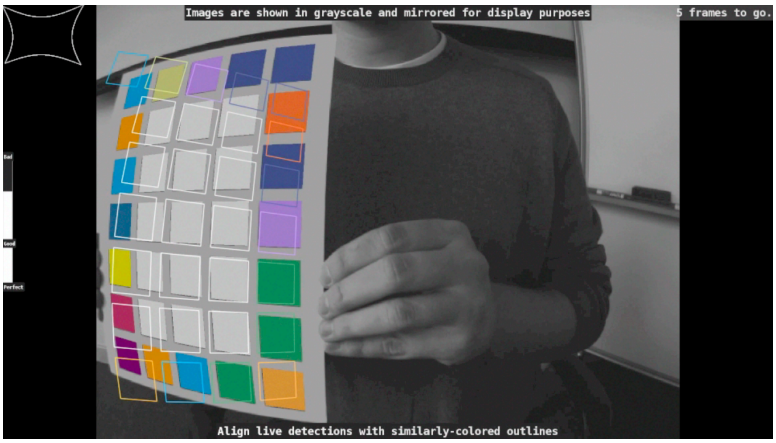

Figure 2.4: Suggestion given to the user by the AprilCal toolbox.

lens distortion parameters using a factorization approach [9].

Most of the techniques mentioned above share the same working scheme. First, the user collects a set of images of the target in order to run a batch offline calibration on them afterwards. However, these techniques are prone to produce inaccurate calibrations specially if the selection of images is not optimally distributed in the image plane. Indeed, in order to obtain good calibration results the algorithms should use images which homogeneously reproject their points in the image plane, as demonstrated by Hu and Kantor from a statistical learning perspective [17].

Our work instead uses an incremental algorithm for calibration, continuously gathering new target observations until a certain convergence threshold is achieved. Similar incremental approaches were already introduced for SLAM problems [18] [19]. In this method, and following the similar work in AprilCal [13], the target images are taken to maximize the accuracy of the parameters and, therefore, obtain a fast calibration. AprilCal is an interactive calibration process consisting of a fixed camera and a moving target. The method provides suggestions to the user of where to move the target such that it reduces the parameter variance the most, as shown in Figure 2.4. The procedure is repeated until the calibration has reached the specified accuracy. This interactive approach aims to maximize the calibration accuracy while ensuring fast convergence and repeatability.

Unlike AprilCal, the proposed method does not require the intervention of a user moving the target in front of a fixed camera. Instead, the target is kept fixed in the environment and a robotic arm holding a camera is in charge of moving to the specified positions to capture images of the target. A robotic arm has already been used in the literature to manipulate a calibration target automatically [17], without including a specific algorithm for selecting next best poses.

Chapter 3

Camera calibration

This chapter describes the core theory of this thesis. First, an explanation of the camera calibration algorithm followed is given with special emphasis in the batch and incremental algorithm workflows.

Next, the active calibration procedure is detailed in depth. Specifically, we present the method for generating a collection of possible poses in the environment and also the suggestion procedure for next best poses.

3.1 Batch calibration

Batch calibration aims to calibrate a camera using a full collection of target images. The more variate this images are, in terms of positions and angles used, the more accurate the calibration will be. This procedure is the basis for the posterior incremental and active calibration implementations.

The proposed batch calibration method is structured in the four main stages shown in Figure 3.1. Each of them will be explained in a different subsection.

3.1.1 Collection of observations

As mentioned in section 2.2, the choice on the image database used for calibration is crucial to get sufficiently accurate parameters. Therefore, the collection step is probably the most important in any camera calibration procedure.

Once the user has captured enough images of the target, the method iteratively recognizes the target in each image and produces a collection of observations. Each observation contains the location of the corners detected in the image together with the characteristics of the target used, which is the only information needed to run the calibration. The algorithm responsible for the detection of AprilTags given an input image falls beyond the scope of this thesis, although a good amount of documentation is available, specially in its main paper [12], and even a ROS implementation is distributed [20].

3.1.2 Initialization of camera parameters

The initialization of camera parameters is based on the work by Zhang [4]. This step does not include any estimation of lens distortion coefficients. They are computed in a later step in the non-linear least-squares optimization.

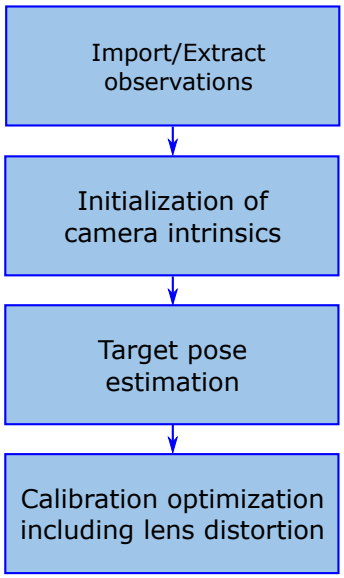

Figure 3.1: Workflow of the batch camera calibration algorithm.

Considering the pinhole camera model, (2.4) describes how a point in the world reference system (in meters) can be transformed to a 2D point in the image (in pixels). Without loss of generality, one can assume that the target is placed in $z = 0$. Then, the previous equation becomes (3.1), where r_i denotes the i-th column of the rotation matrix R_{CW}.

$$m' = K \begin{bmatrix} r_1 & r_2 & r_3 & {}_C t_{CW} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ t \end{bmatrix} = K \begin{bmatrix} r_1 & r_2 & {}_C t_{CW} \end{bmatrix} \begin{bmatrix} x \\ y \\ t \end{bmatrix} = H \begin{bmatrix} x \\ y \\ t \end{bmatrix} \tag{3.1}$$

The goal of the method is therefore to find the homography $H = K \begin{bmatrix} r_1 & r_2 & {}_C t_{CW} \end{bmatrix}$ that relates a point in the world frame (assumed at $z = 0$) with an image point. Afterwards, we need to decompose this estimated homography H into the camera intrinsics and the target pose, as defined in (3.2), where h_i represents the ith column of the homography H and λ symbolizes the scale factor imposed by the homogeneous coordinates.

$$\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = \lambda A \begin{bmatrix} r_1 & r_2 & {}_C t_{CW} \end{bmatrix} \tag{3.2}$$

This can be done if we impose two constraints concerning the ortho-normality of rotation components r_1 and r_2, as described in (3.3) (3.4).

$$h_1^T A^{-T} A^{-1} h_2 = 0 \tag{3.3}$$

$$h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \tag{3.4}$$

It is possible to express the components of the matrix $B = A^{-T} A^{-1}$ in terms of the intrinsic parameters as follows:

$$\begin{aligned} B = A^{-T} A^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} = \\ = \begin{bmatrix} \frac{1}{f_x^2} & -\frac{\gamma}{f_x^2 f_y} & \frac{v_0\gamma - u_0 f_y}{f_x^2 f_y} \\ -\frac{\gamma}{f_x^2 f_y} & \frac{\gamma^2}{f_x^2 f_y^2} + \frac{1}{f_y^2} & -\frac{\gamma(v_0\gamma - u_0 f_y)}{f_x^2 f_y^2} - \frac{v_0}{f_y^2} \\ \frac{v_0\gamma - u_0 f_y}{f_x^2 f_y} & -\frac{\gamma(v_0\gamma - u_0 f_y)}{f_x^2 f_y^2} - \frac{v_0}{f_y^2} & \frac{(v_0\gamma - u_0 f_y)^2}{f_x^2 f_y^2} + \frac{v_0^2}{f_y^2} + 1 \end{bmatrix} \end{aligned} \tag{3.5}$$

B is a symmetric matrix, and can be defined by the 6×1 vector (3.6).

$$b = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T \tag{3.6}$$

If the ith column vector of H, h_i, is $h_i = [h_{i1}, h_{i2}, h_{i3}]^T$, then we can express (3.3) as

$$h_1^T B h_2 = v_{ij}^T b \tag{3.7}$$

where

$$v_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j_1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T \tag{3.8}$$

Using the notation introduced in (3.6), it is possible to express the two orthonormality constraints as a homogeneous system .

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{3.9}$$

By stacking n equations such as (3.9), one for each target image, we obtain (3.10), where V is a $2n \times 6$ matrix.

$$Vb = 0 \tag{3.10}$$

To solve this system we need in general at least 3 images. However, with two images is still possible to work imposing zero skew ($\gamma = 0$). The solution is given by the the eigenvector of $V^T V$ corresponding to the smallest eigenvalue. Once b is estimated we can univocally extract from it the intrinsic parameters as in (3.5).

3.1.3 Estimation of the target pose

The procedure of estimating the pose of a calibrated camera given a set of 3D points in the world frame and their correspondent 2D projected points in the image is also called Perspective-n-Point (PnP). It consists on discovering the 3 degrees-of-freedom (DOF) associated to rotation together with the three-dimensional translation vector once the intrinsic parameters are known.

In this work we have used the PnP solver included in the OpenGV toolbox [21]. First, a perspective-3-point (P3P) algorithm [22] is used together with RANSAC for rejection of outliers, by introducing intermediate camera and world reference frames and parameterizing their relative transformation. Next, a general PnP method is implemented using the remaining inliers to discover the relative target-camera pose.

3.1.4 Calibration optimization including lens distortion

Although the calibration does not necessarily need a final optimization step, this is normally included to achieve higher accuracy results. Moreover, it allows to include other effects such as lens distortion, which helps to characterize better the system reducing the overall reprojection error.

As the number of images utilized increases the problem becomes large in terms of residuals developing into a large-scale bundle adjustment problem [14]. This optimization is usually formulated as a non-linear least squares problem, where the goal is to minimize the squared difference between the observed corner location in the image and projection of the corresponding 3D target point on the image plane of the camera, as described in (3.11), where w_{ij} is an indicator function equal to 1 if point j is visible from camera i. All error terms are equally weighted.

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} ||m_{ij} - \hat{m}(K, R_i, t_i, M_j)||^2 \tag{3.11}$$

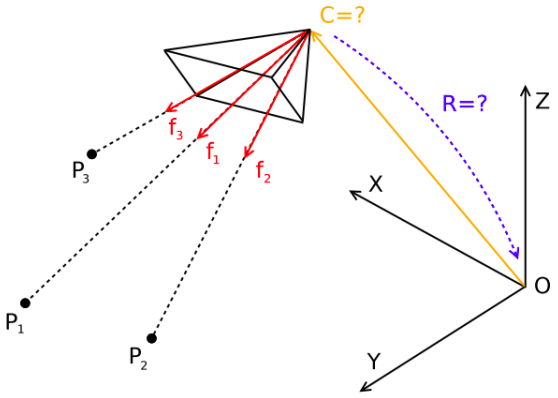

Figure 3.2: General target pose scenario. $\{O, X, Y, Z\}$ represents the world reference frame, C symbolizes the position of the camera, and P_1, P_2, P_3 are three viewpoints with f_1, f_2 and f_3 being vectors pointing respectively the three viewpoints from the camera [22].

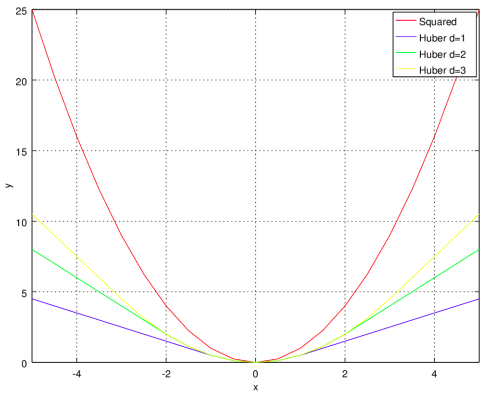

Figure 3.3: Huber Loss function for different values of δ. Squared error is also shown for comparison.

The goal of the minimization is to find the four camera parameters (f_x, f_y, u_0, v_0), the four coefficients characterizing the lens distortion (k_1, k_2, p_1, p_2) and seven components for the pose of each image consisting in the rotation quaternion (x, y, z, w) (see Appendix A for a detailed explanation) and the translation vector (t_x, t_y, t_z). We have omitted the skew-coefficient γ in the optimization for implementation purposes.

Although in (3.11) we have expressed the distance between the 2D point and the reprojected 3D point as a L2 norm, several different implementations for this loss function can be applied. Particularly, using the Huber loss function, defined in (3.12) and depicted in Figure 3.3, allows better treatment of outliers in front of squared errors, leading to a most robust result.

$$L_\delta(x) = \begin{cases} \frac{1}{2}x^2 & for|x| \leq \delta; \\ \delta(|x| - \frac{1}{2}\delta) & otherwise. \end{cases} \tag{3.12}$$

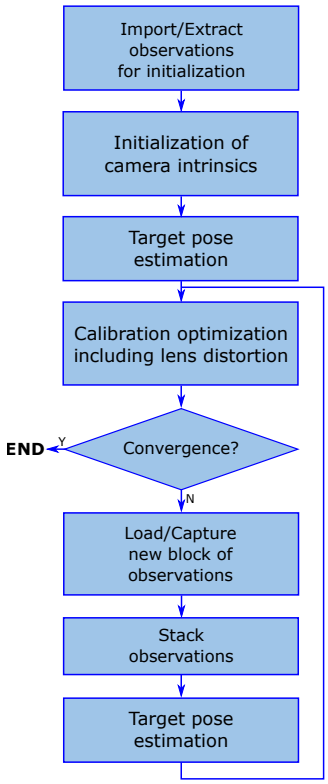

Figure 3.4: Incremental calibration algorithm workflow.

In this work, we have used the open-source C++ optimization framework *Ceres Solver* [23], in production by Google, to implement the calibration optimization routine. By constructing the individual residuals, one per corner in every image, the engine is capable of minimizing the cost function according to one of various possible solvers, such as Levenberg-Marquardt.

The inclusion of the lens distortion effects is done in the computation of each residual, following the explanation in Section 2.1.2. Also, we need to numerically compute the jacobian of each residual. More details on the calculation of the jacobian matrix and the effect of quaternions on it are given in the Appendix A.

3.2 Incremental calibration

A further step towards the accomplishment of an active camera calibration method is the implementation of an incremental algorithm, which allows to consecutively add new images while calibrating the parameters. The more images added, the more accurate the parameters should be. Specifically, this effect can be seen as a reduction in the variance of the parameters.

The incremental algorithm builds upon the batch calibration routine for initialization of camera and lens distortion parameters, and adds additional steps for new target images. As illustrated in Figure 3.4, the new images are directly added to

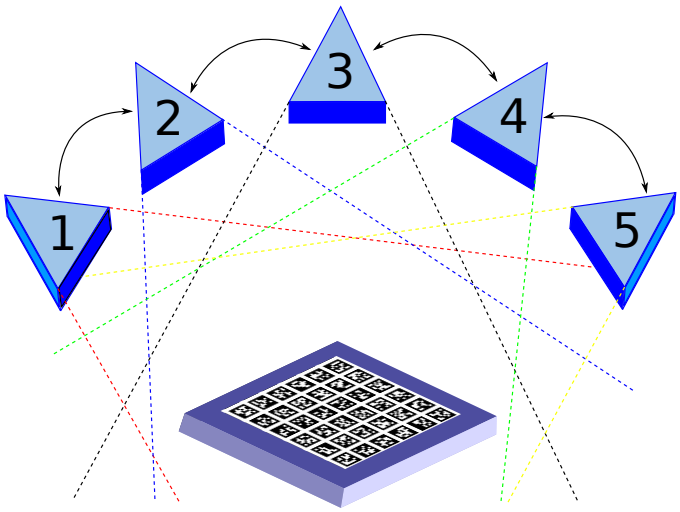

Figure 3.5: Active calibration scenario where the target is fixed in the world reference frame and a moving camera takes pictures from it.

the non-linear optimization, after an initial estimation of the parameters has been obtained. Obviously, the result of this method is dependent on the quality of the initialization images used. If the initialization images were not variate enough, then the algorithm could fall in a local minimum of the cost function instead of the global minimum desired.

3.3 Active calibration

In the literature there have been many attempts to make the process of camera calibration more robust and repeatable. However, most of the techniques developed targeted passive calibration. To our knowledge, AprilCal [13] constitutes the first publication introducing an interactive calibration method, in which the user is given suggestions on the location where to move the target in order to best optimize the current camera parameters.

The setup used in AprilCal consists of a fixed camera in the world reference frame and a moving target. In this case is the user who is responsible for moving the target to match the suggestions done by the software, as shown in Figure 2.4. Conversely, the proposed method in this thesis aims to work in the opposite way: the target is fixed in the environment while the camera is movable in front of it, as schematically depicted in Figure 3.5.

The proposed method for selecting new poses is based on the Max ERE strategy and adapted to work under the scenario depicted in Figure 3.5. The workflow of this algorithm is given in Figure 3.6. It is divided first in an offline step consisting in the definition of poses where the camera can be moved to inside the working area the robot, and after an online stage in which the robot starts visiting iteratively the different selected poses and keeps capturing new observations until convergence is reached.

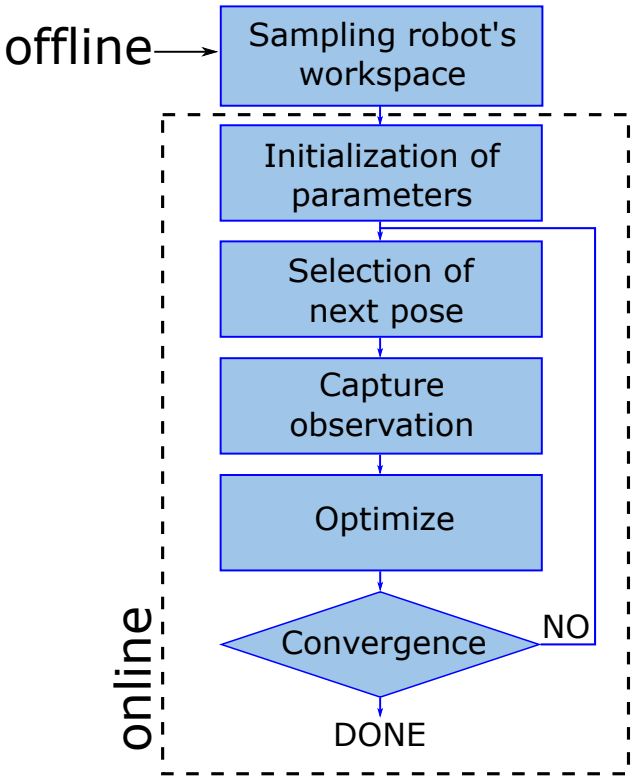

Figure 3.6: Proposed active calibration algorithm workflow.

3.3.1 Generation of poses

The generation of poses is probably the most important step in the algorithm and a good selection of available poses greatly affects the accuracy of the resulting calibration.

In camera calibration best results are obtained when poses reproject homogeneously all over the image plane, as described in [17] from a mathematical perspective. Hence, the approach taken here is to simulate poses homogeneously in the environment so that their reprojection spreads homogeneously over the image plane. Following the *Inverse transform sampling* [24] theorem we are able to generate the poses inside the working area of the robotic system we are using, which, in the case of a robotic arm, can be roughly approximated as a semi-sphere. Figure 3.7 shows, for the robotic arm case, the hypothetical quarter of sphere containing the poses.

We parametrize the R-radius semi-sphere volume using spherical coordinates (r, θ, ϕ) such that the original cartesian coordinates can be recovered as expressed in (3.13), where $\theta \in [0, \pi/2]$, $\phi \in [-\pi/2, \pi/2]$ and $r \in [0, R]$.

$$\begin{aligned} x &= r\sin(\theta)\cos\phi \\ y &= r\sin(\theta)\sin\phi \\ z &= r\cos(\theta) \end{aligned} \tag{3.13}$$

However, by sampling uniformly the spherical coordinates in their range of operation does not result in an homogeneous distribution of points in the sphere. To achieve this uniformity we need to apply the aforementioned theorem such that we transform

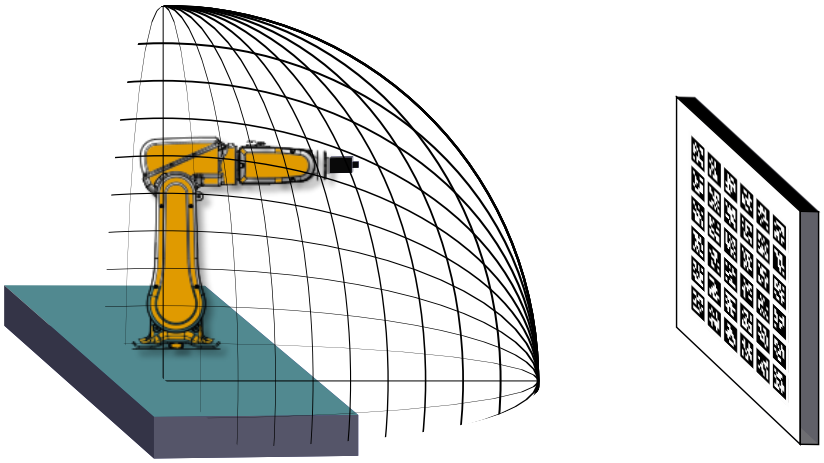

Figure 3.7: Quarter of a sphere containing poses around a robotic arm.

the sampling to the following ranges:

$$\begin{aligned} \phi &\in [0, \pi] \\ \cos(\theta) &\in [-1, 1] \\ u &\in [0, 1] \end{aligned} \tag{3.14}$$

From (3.14), we can obtain the spherical coordinates as:

$$\begin{aligned} \theta &= \arccos(\cos(\theta)) \\ \phi &= \phi \\ r &= R\sqrt[3]{u} \end{aligned} \tag{3.15}$$

Figure 3.8 shows an example of the generated poses around the robotic arm. In this case, the working area of the robot has been limited to $\theta \in [0, \frac{5\pi}{12}]$, $\phi \in [-\frac{5\pi}{9}, \frac{5\pi}{9}]$ and $r \in [0.3, 0.7]$. The target is placed at 1.3 meters from the robot base.

From all this set of possible positions the algorithm plans them using a motion planner and discards the unfeasible ones. Only the reachable positions will be maintained in the list of available poses for the online part of the algorithm.

Regarding their orientation, the algorithm selects two different rotations randomly at each position such that the camera is always pointing towards the target. Specifically the ranges used for roll, tilt and pan are between -20º and 20º for tilt and pan, while roll can span all the 360º.

3.3.2 Initialization of camera parameters

The initialization of camera and lens distortion parameters in the active calibration routine is exactly the same as in the incremental version. In a batch manner, a few observations are stacked and subsequently optimized.

The selection of poses in the initialization procedure has great importance on the final calibration results. A bad selection of poses, for example poses which reproject to the exact same region of the image plane, might make the algorithm converge to a local minimum of the cost function instead of the expected global minimum.

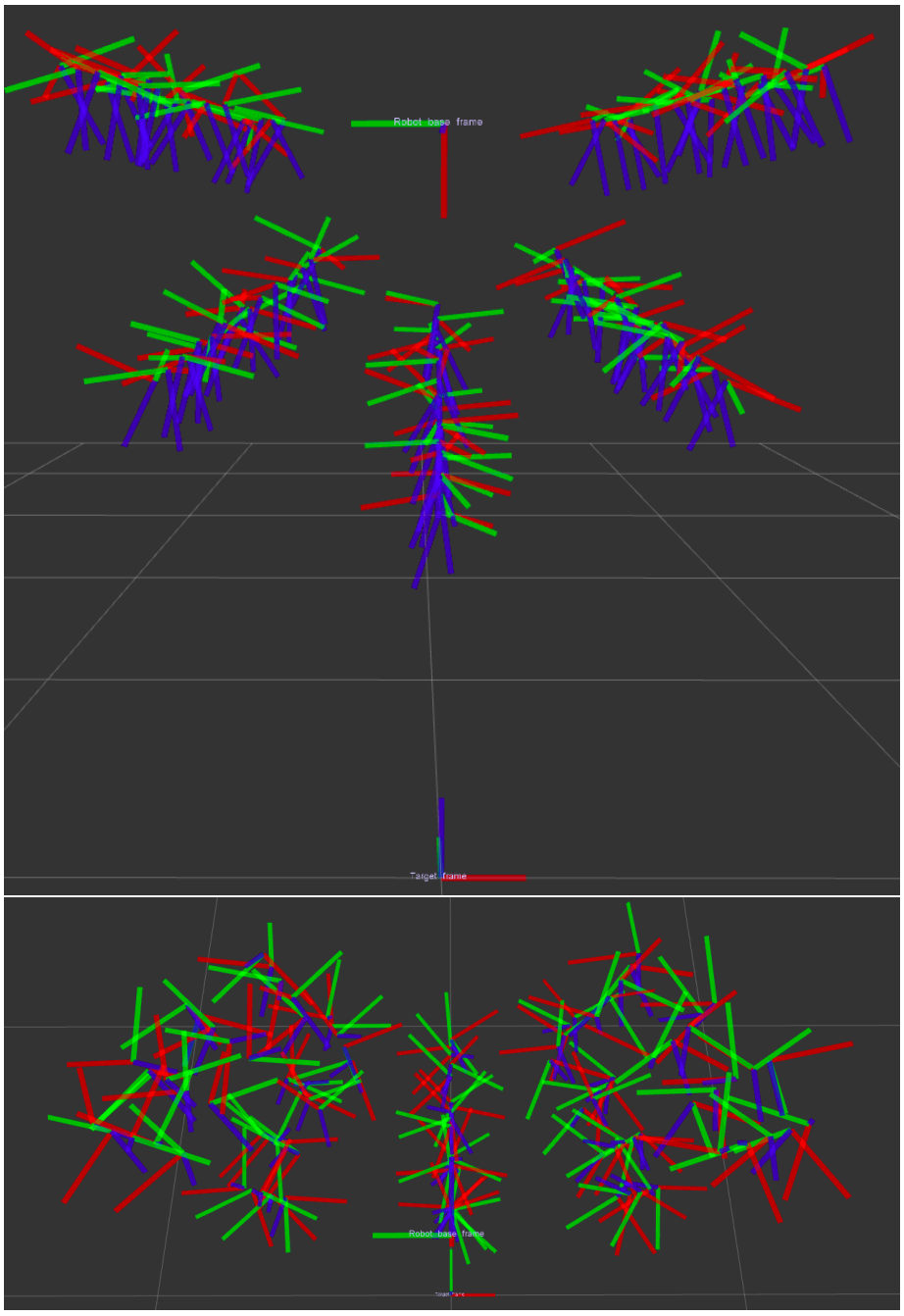

Figure 3.8: Uniformly distributed poses within a sphere around the robot base. The generated poses are pointing towards the target, which is also depicted as a reference.

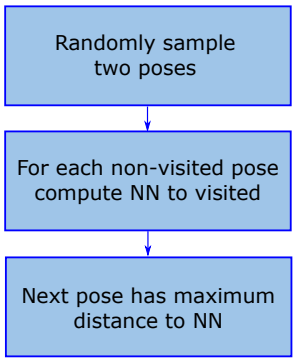

Figure 3.9: NN-based selection of poses for initialization of camera parameters.

This work includes two independent methods for initializing parameters, whose experimental results will be contrasted in Section 4.2.4. First, a completely random sampling among the set of available poses. Also, in order to avoid the possibility of sampling very close poses that would not provide much information to the calibration routine, a method based on the Nearest-Neighbour (NN) approach has also been developed. This method aims to select poses which fall the further apart possible from already visited poses, and therefore maximize the different views obtained of the target. To achieve this goal, the algorithm computes the distance among all the non-visited poses to all the visited poses and select the non-visited one which offers the highest distance to its closest visited pose, by using the NN algorithm. This algorithm is shown schematically in 3.9.

3.3.3 Selection of next pose

The proposed algorithm selects the next pose among all the non-visited poses as generated in the first step (described in section 3.3.1). The selected pose is the one which diminishes the variance of the camera parameters the most.

The outline of this algorithm is shown in Figure 3.10. In the first step, the mean and variance from the camera parameters are obtained and used to generate a distribution of parameters. Then, sampling this distribution results in a collection of 'sampled' cameras.

For each sampled camera and candidate camera pose the target points are reprojected into the image plane. By comparing the reprojection on these sampled cameras with the camera containing the current estimates of the parameters we achieve a measure of the expected reprojection error for this specific camera pose. The procedure is repeated for all the available poses and ends with the selection of the pose which maximizes the ERE. The max ERE pose is the most uncertain position at this calibration point and, therefore, is the one that can lead to a highest improvement of the parameters' accuracy. The algorithm is described schematically in Algorithm 1.

The max ERE metric was introduced in AprilCal and, unlike other alternatives such as Mean Reprojection Error (MRE) and Mean Squared Error (MSE), is a pixel-based calibration measure based on the testing error instead of the training error. This is achieved by working with the posterior distribution of the camera parameters, which is assumed as multi-variate Gaussian. By working with the testing

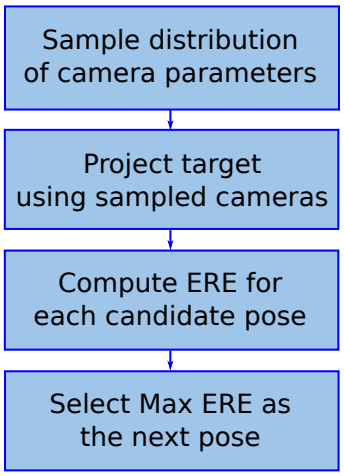

Figure 3.10: Max ERE algorithm to select next pose.

error, the parameters can generalize the data and do not overfit the given dataset, which can happen if the training error is used.

Also, other policies for computing next pose have been implemented for comparison. Namely, a totally random policy and a NN-based policy, which selects poses which maximize the distance to the already visited ones. These procedures will be further explained in Section 4.

Algorithm 1 Computation of next pose using Max ERE

$listPoses = \text{GETNONVISITEDPOSES}()$
$(x, \Sigma) = currentCal.\text{GETMODELPOSTERIOR}()$
$sampleCalList = [sampleCal_0(x, \Sigma), \ldots, sampleCal_1(x, \Sigma)]$
$MaxERE = 0$
$nextPose = 0$
for all $p \in listPoses$ **do**
  $ERE = 0$
  for all $sampleCal \in sampleCalList$ **do**
    $ERE+ = \frac{1}{N}|currentCal.\text{PROJECT}(target) - sampleCal.\text{PROJECT}(target)|$
  end for
  if $ERE \geq MaxERE$ **then**
    $MaxERE = 0$
    $nextPose = p$
  end if
end for

3.3.4 Optimization of next pose

Once the next pose has been computed, the robotic system has to move the camera to the specified position in order to take a picture of the target. Next, the picture is analyzed and the target is detected. This information feeds the calibration algorithm in an incremental manner producing updated results on the parameters.

3.3.5 Convergence of the algorithm

The active calibration method is iterative: it keeps capturing new observations and adding them into the calibration routine until a certain accuracy metric is achieved.

In this work we have decided upon the variance of the parameters as the accuracy metric. Specifically, we impose a threshold on the maximum standard deviation of the focal lengths among the two dimensions, as defined in (3.16). The value of the threshold can be easily expressed in pixels and determines the accuracy of the calibration.

$$\text{finishes if } \max(\sigma_\alpha, \sigma_\beta) < \sigma_{threshold} \text{ [px]} \tag{3.16}$$

Chapter 4

Experiment results

This chapter includes all the experimental results we have obtained, both on simulation and real scenarios. First, the performance of the batch algorithm, which is the basis of the active camera calibration method, is analyzed on a calibration dataset. Then, after introducing the setup of the experiment used, results gathered using the active technique are presented and compared against the traditional procedures.

4.1 Batch calibration

In order to assess the performance of the batch calibration algorithm we have used the EUROC calibration dataset [25]. This dataset employs AprilTags to account for partially occluded targets and includes ground truth regarding intrinsic parameters and also target poses.

Table 4.1 shows in the first column the expected camera and lens distortion parameters for the EUROC calibration dataset. These have been obtained considering the pinhole camera and radial-tangential distortion models. In the middle column the same parameters output by the batch calibration algorithm also using the pinhole camera and the radial-tangential distortion models are shown. Their absolute difference is given in the right-most column, showing very close performance between both approaches.

Table 4.1: Calibration results on the EUROC calibration dataset [25].

	EUROC calibration	Batch calibration	Absolute difference
α	458.655	459.657	1,002
β	457.296	458.411	1,115
u_0	367.216	369.096	1,880
v_0	248.375	246.539	1,836
k_1	-0.283	-0.279	4.00e-2
k_2	0.074	0.070	4.00e-2
p_1	1.936e-4	5.825e-4	3.89e-4
p_2	1.760e-5	-3.765e-4	3.94e-4

4.2 Active calibration

The active calibration algorithm is built on the basis of the batch and incremental routines. To assess its performance in terms of speed of convergence and repeatability we have compared calibrations using the Max ERE policy for selecting new poses against other alternatives, such as randomly sampling and distance sampling, in subsection 4.2.4. Also, the results obtained with the active routine are contrasted to traditional calibration algorithms in subsection 4.2.6.

4.2.1 Experimental setup

Figure 4.1 shows the setup of the experiments with the robotic arm. It is worth noting that the target is at a fixed position in front of the manipulator, and the camera is attached at the tip of the arm focusing at the target.

Before starting any experiment an important hand-eye calibration has to be done. Specifically, the transformation between target (T) and robot base (B), called T_{BT}, and the transformation between the arm wrist (W) and the camera (C), called T_{CW}, have to be found. For example, the rotation matrices involved in the scenario shown in Figure 4.1 are as follows:

$$\begin{aligned} R_{BT} &= \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \\ R_{CW} &= \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \end{aligned} \tag{4.1}$$

Regarding the translations included in the transformation matrices, they describe, respectively, ${}_{B}t_{BT}$ and ${}_{C}t_{CW}$. These distances can be measured as the distance from base to target expressed in the robot base coordinate system and the distance from camera to wrist expressed in the camera frame respectively.

These transformations will remain intact during the experiment and are only used to select the best next poses as described in the Max ERE algorithm. Although we could include their computation in the calibration optimization, the effect on the results' accuracy does not compensate for the increment in the overall complexity.

The robotic arm used in this experiment is the industrial IRB 120 from ABB company, while the camera is a monochrome Point Grey Grasshopper 3, model GS3-U3-32S4M, with a resolution of 3.14 megapixels (2048 $\times$ 1536). Figure 4.2 show some pictures of the experiments conducted in this thesis.

Additionally, the characteristics of the AprilTag target used have to be given to the calibration procedure. In particular, the number of tags per row and column, and the tag size and the space in-between tags, both expressed in meters. Finally, other required arguments are the camera ROS topic in which the camera publishes the images taken and the threshold on the maximum standard deviation of the focal lengths that the algorithm will use to determine convergence, expressed in pixels.

4.2.2 Recording dataset with the robotic arm

In order to be able to run experiments offline it is possible to, first, record a dataset using the arm, generate a rosbag and afterwards, run on it the calibration procedure.

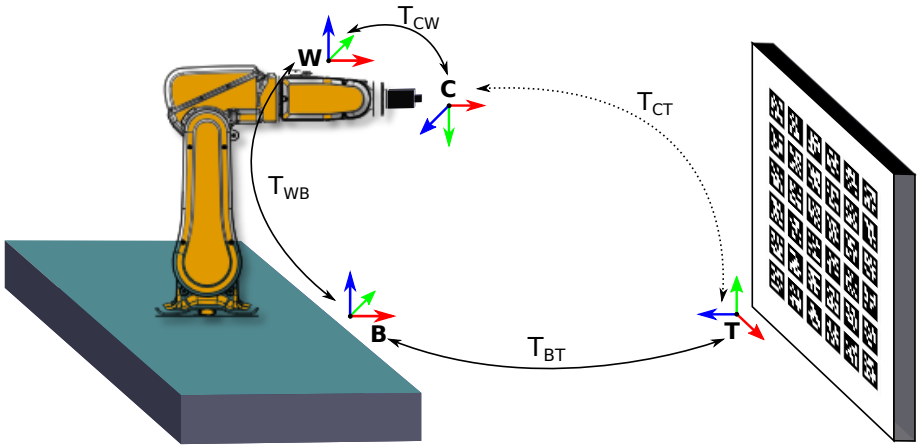

Figure 4.1: Experiment setup with a robotic arm. The different coordinate frames are drawn, together with the transformations between them.

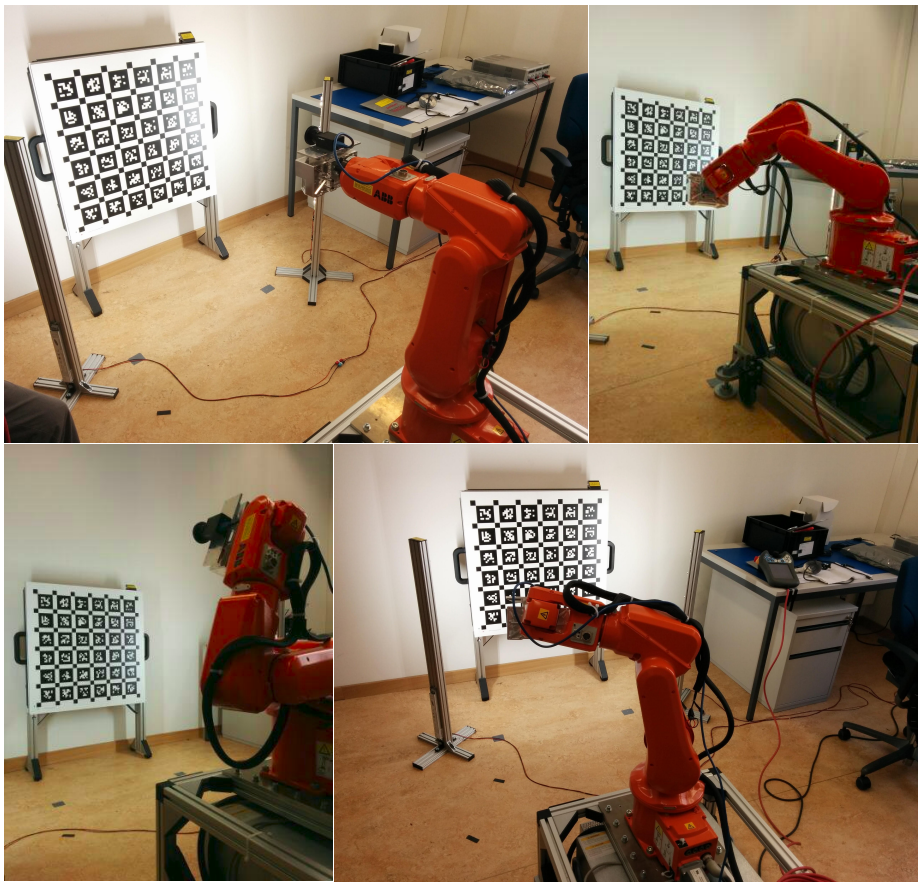

Figure 4.2: Real experimental setup used, with a ABB IRB 120 and a monochrome Point Grey Grasshopper 3 camera mounted on its tip.

Before running the method, one needs to generate a list of available poses, as described in Section 3.3.1. This list is stored in a CSV file which is then fed into the calibration algorithm.

The algorithm visits all the list of available poses in batch mode and for each of them takes a picture of the target. All the gathered images are stored in a rosbag file. Also, after running the AprilTag detection algorithm, the list of the obtained observations, containing the position of the detected corners in each image, is serialized into a Protocol Buffer [26], which allows for a much faster loading as compared to rosbag.

4.2.3 Proposed strategy

The proposed strategy for calibration consists in using a non-random initialization, based on a nearest-neighbour approach for selecting poses, together with the Max ERE algorithm for computing next best poses.

One of the main advantages of this method over the traditional manual calibration techniques is the repeatability. Table 4.2 shows the results of the camera parameters obtained in five different realizations of the calibration, together with the mean and standard deviations for each of them. We can observe that the standard deviation for the camera parameters is quite low (less than 2 pixels in all cases except for one component of the principal point), giving a very consistent calibration among different executions. Regarding lens distortion parameters we achieve a high level of accuracy in all the parameters.

Table 4.2: Proposed strategy results for five calibrations of the same dataset. The mean of the parameters for the five realizations are shown in bold. The last row of the table shows the number of poses needed to calibrate the camera, excluding the initialization phase.

	1	2	3	4	5	mean	std
α	1004.02	1008.20	1007.07	1005.98	1006.84	**1006.42**	1.56
β	1003.56	1007.60	1006.49	1005.61	1006.68	**1005.99**	1.53
u_0	1054.30	1055.08	1054.83	1055.26	1054.99	**1054.89**	0.36
v_0	748.00	746.70	747.63	747.79	747.01	**747.43**	0.55
k_1	-0.19	-0.19	-0.19	-0.19	-0.19	**-0.19**	5.41e-4
k_2	5.12e-2	5.10e-2	5.18e-2	5.16e-2	5.23e-2	**5.16e-2**	5.15e-4
p_1	-9.20e-5	-1.43e-4	-1.09e-4	-1.01e-4	3.27e-6	**-8.82e-5**	5.46e-5
p_2	1.72e-4	4.56e-5	1.57e-4	7.17e-5	2.89e-5	**9.51e-5**	6.55e-5
#poses	19	14	20	18	20	**18.2**	

Also, the last row of Table 4.2 shows the number of observations needed in each case to achieve convergence. The variability in this case is caused by the initialization phase, which has a random component that affects the overall performance. In average the algorithm can calibrate a camera with only 18 observations.

4.2.4 Comparison of Max ERE with other strategies

Apart from implementing the Max ERE approach, and in order to contrast its performance in detail, the algorithms included in Table 4.3 have also been taken into account in the analysis. They are characterized by the initialization technique used and the method employed to select next best poses.

Table 4.3: Strategies used for the active algorithm. The proposed method (Max ERE) is added in the table for comparison.

	Initialization	**Selection of next pose**	**Number of poses**
Random	Random	Random	27.6
Random Max ERE	Random	Max ERE	21.8
Nearest-Neighbour	NN	NN	21.0
Max ERE	NN	Max ERE	**18.2**

Firstly, the completely random procedure (both initialization and selection of next pose at random) yields a very slow algorithm in terms of required poses to converge apart from giving a result with high variance among different realizations. Approximately the standard deviation is three times the obtained for the Max ERE approach, as shown in Figure 4.3.

The second proposed strategy also contains a random initialization and, although the results regarding rate of convergence and repeatability are slightly better than the completely random procedure, they are still far from the other two techniques.

Finally, the NN-based approach and the proposed strategy achieve similar results in terms of repeatability among different realizations of the algorithm, although the Max ERE algorithm is a bit more accurate than the NN-based approach for all parameters.

However, the big difference between the NN-based and Max ERE strategies is on the number of poses required in other to reach convergence of the calibration. The Max ERE strategy can calibrate the same camera using approximately three less poses than the other strategies in average, including the NN algorithm. This feature shows very high repeatability as compared to manual techniques.

4.2.5 Comparison with the complete dataset

Another interesting analysis is the comparison between the results obtained by calibrating the camera with only the required poses as selected with the active approach (using the Max ERE algorithm) and the calibration parameters obtained using the dataset containing all the observations captured from all the poses from the list of poses generated in the first step prior to the online algorithm.

The results are shown in Table 4.4. In it we can observe that both algorithms are able to get similar results for all the parameters using a much smaller number of observations for the active case. Using the Max ERE algorithm the routine requires 18 poses in average, while the complete dataset consists of 88 different poses. This is a reduction of almost five times the number of poses required, with the consequent improvement in terms of calibration speed.

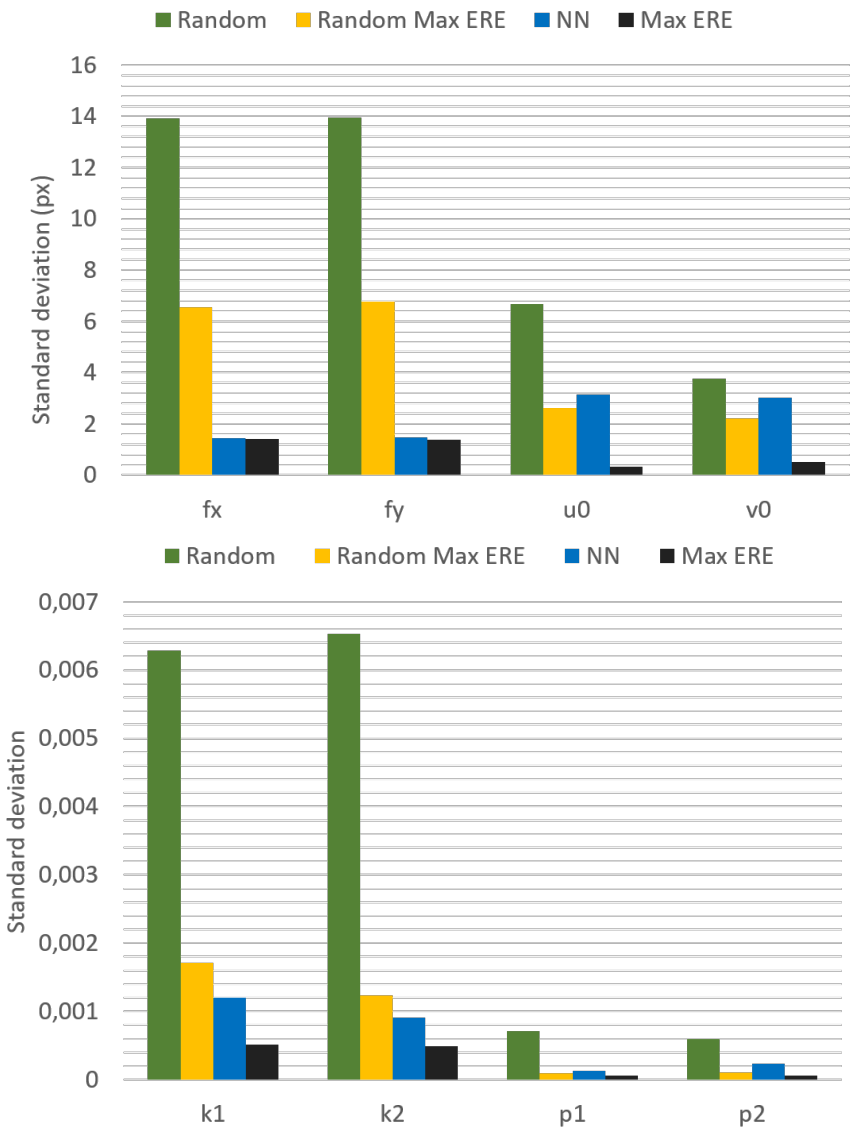

Figure 4.3: Variability of camera and lens distortion parameters for the different strategies analyzed: Random, Random Max ERE, Nearest-Neighbour and Random Max ERE. Ten realizations are considered in this analysis.

Table 4.4: Comparison of the active algorithm with a batch calibration of the complete dataset.

	Batch calibration	**Active calibration**	**Absolute difference**
f_x	1007.65 px	1006.33 px	1.32 px
f_y	1007.13 px	1005.92 px	1.21 px
u_0	1052.82 px	1054.87 px	2.05 px
v_0	750.03 px	747.50 px	2.53 px
k_1	-0.18	-0.18	5.74e-3
k_2	4.97 e-2	5.16e-2	1.90e-3
p_1	-5.95e-5	-1.01e-4	4.15e-5
p_2	-5.23e-6	8.45e-5	8.97e-5
Number of poses	**88**	**18**	

4.2.6 Comparison with manual calibration

Finally, in order to evaluate the suitability of the active algorithm for replacement of the manual calibration techniques the accuracy of the camera and lens distortion parameters obtained with the Max ERE approach has been compared with the obtained with a traditional calibration tool. The calibration toolkit utilized for this purpose is Kalibr [27], which runs offline on a manually recorded dataset.

Three different datasets have been used independently for calibration. All of them have been recorded by an expert user with the same camera (*Point Grey Grasshopper 3 monochrome*) and with the same target characteristics as the robotic arm experiment. In these conditions, ideally the camera parameters obtained should be very similar to the results achieved with the active calibration technique. Sample snapshots of one of the manually recorded datasets are depicted in Figure 4.4, showing different perspectives of the target for higher accuracy calibration.

The results of this comparison are shown in Table 4.5, with the mean and variance of the parameters obtained using both techniques. We can see a noticeable difference of about three pixels for the estimated focal lengths and principal points, while the distortion is well approximated. This disagreement can be well explained by exploring the characteristics of the datasets used in each calibration.

Figure 4.5 shows the reprojection of target points on the image plane for both datasets (manually recorded and active). While in the manually recorded datasets we show an homogeneous distribution of reprojected points on the image plane as expected, in the active calibration dataset the poses do not reproject homogeneously on the image plane although the generation of poses was done to accomplish this distribution. This effect can be specially seen in the corners and borders of the image plane. The use of a robotic arm has a very limited working area around the robot, which reduces the possibility to capture extreme poses. If the robot arm is placed very close to the calibration target, extreme poses can be captured easily but the amount of target corners observed is significantly reduced. On the other hand, if the target is kept far away from the camera, most of the corners can be observed

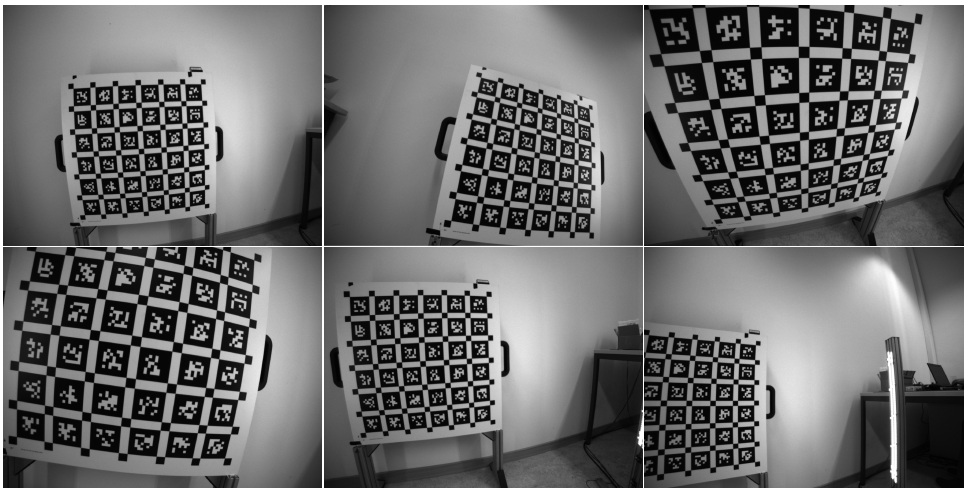

Figure 4.4: Dataset recorded for manual calibration with Kalibr. Different perspectives are crucial to obtain accurate calibration parameters.

from all poses at the expense of not capturing variate poses which supply the majority of valuable information for the calibration. In this case, other robotic systems such as UAVs can benefit more from this active calibration since their working area is not that limited.

Table 4.5: Comparison of the active calibration technique against a manual calibration using the Kalibr toolbox on three different datasets.

	Active calibration		**Manual calibration**	
	μ	σ	μ	σ
f_x	1009.94 px	2.09 px	1006.33 px	1.41 px
f_y	1009.87 px	2.32 px	1005.92 px	1.37 px
u_0	1055.36 px	1.65 px	1054.87 px	0.33 px
v_0	752.62 px	2.48 px	747.50 px	0.52 px
k_1	-0.18	9.98e-4	-0.18	5.08e-4
k_2	4.54e-2	6.85e-4	5.16e-2	4.88e-4
p_1	-2.15e-5	1.41e-4	-1.01e-4	5.79e-5
p_2	3.83e-5	6.95e-5	-8.45e-4	6.41e-5

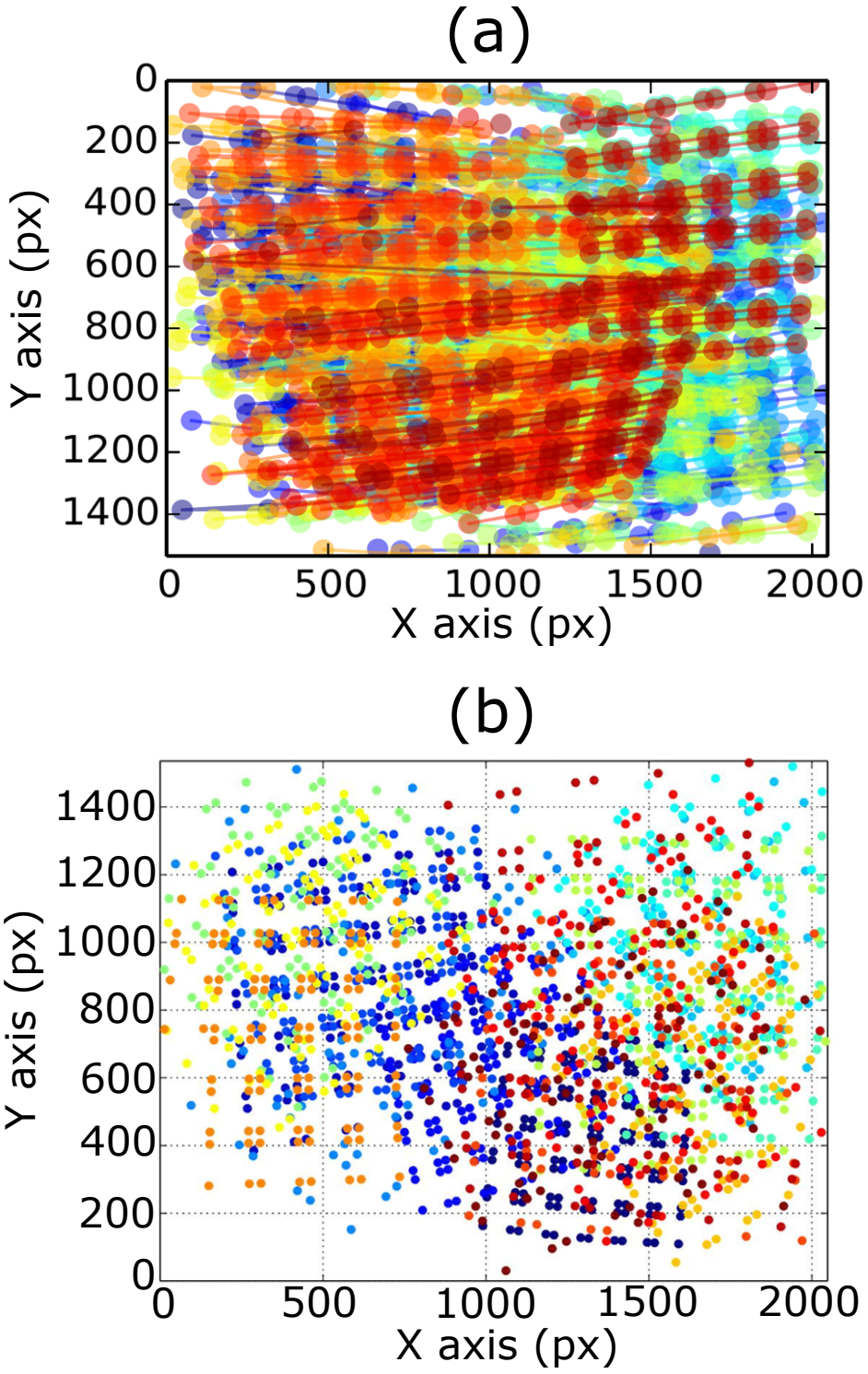

Figure 4.5: Reprojections of target points on the image plane for the manual and active calibration datasets: (a) shows the reprojections for the manual dataset and (b) for the active calibration dataset.

Chapter 5

Conclusions

Camera calibration is a fundamental step to be done in any computer vision work that includes interpreting spatial information from an image or video. However, the techniques used still require the decisive intervention of an expert user which greatly determines the final achieved accuracy of these algorithms.

In this thesis we have proposed a novel calibration method which, as opposite to the typical calibration routines, is active. Therefore, it does not require the intervention of a human in a decisive manner, producing very consistent results among different calibrations.

uilding upon the work by AprilCal [13], the implemented method selects at each step the next pose from where to capture a new image that best improves the accuracy of the camera parameters, measured as a reduction in their variance. This ensures a fast convergence of the algorithm as compared to manual techniques. The selection of next best poses is based on the maximum expected reprojection error (Max ERE), which is a calibration metric that considers the testing error as opposed to the usual training error, by sampling camera parameters from their posterior distribution.

By using a grid of AprilTags instead of typically used targets, such as checkerboards, the technique allows good operation even with partially observed targets. This circumstance is specially relevant for capturing extreme poses, which greatly influence the accuracy and convergence of the calibration algorithm.

Also, the inclusion of a non-random initialization method based on the Nearest-Neighbour algorithm helps to ensure a fast convergence in front of other random techniques.

The proposed method has been experimentally applied to a robotic arm, showing better performance compared to other implemented strategies and traditional techniques. Regarding accuracy, the Max ERE calibration parameters have almost three times more accuracy than the random policy and double accuracy than the NN-based approach, as measured in terms of variance. Also, the results obtained are very consistent with the manual Kalibr calibration. Further experiments should be conducted in order to test the suitability of this method as a replacement of the manual calibration procedures.

Furthermore, the introduced method is easily reproducible in any other robotic system, not only in industrial manipulators, like micro-aerial vehicles (MAV) for in-

stance. These robotic systems could benefit more from the active algorithm thanks to its larger working area, which allows for better generation of poses.

All these mentioned characteristics make this algorithm suitable to be used in any scenario where camera parameters need to be constantly updated and accurate.

5.1 Future work

While this thesis has demonstrated promising results in the camera calibration field, it can be further expanded in the following directions:

- **Consider other lens distortion models.** Currently the system only allows the radial-tangential distortion, which covers the majority of available cameras. However, and specially for high precision cameras, other models should be implemented such as the equidistant lens distortion model.
- **Adapt to other robotic systems.** Although the current system only supports robotic arms, the adaptation to other robotics systems should be straightforward. The core of the algorithm would remain the same, but the generation of poses and the hand-eye calibration should be modified accordingly. A good next step is to apply the same algorithm for a Micro-Aerial Vehicle (MAV) and prove its performance.
- **Improve the generation of poses.** As mentioned throughout the report, the list of available poses greatly determines the success of the algorithm. In the literature, researchers have demonstrated that an homogenous spreading of reprojected targets in the image plane produces more accurate results. In this thesis we have opted for a homogeneous distribution of poses in the working area of the robot, expecting also a homogeneous distribution of their reprojection in the image plane. Although this is true, the algorithm could incorporate a better sampling procedure, putting more effort in the extreme poses, which reproject to the corners of the image plane, than in the central ones, whose reprojection is very similar.
- **Examine other loss functions in the core calibration algorithm.** At this point, the calibration algorithm by default uses the Huber loss function to minimize the impact of outliers in the optimization. Also, the outlier rejection procedure in the base of the calibration method has a great influence in the achievement of accurate results, since it modifies the global cost function optimized. Therefore, modifying these two aspects might lead to better results regarding reprojection error and accuracy of the final parameters.

Bibliography

[1] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[2] P. Sturm, S. Ramalingam, J.-P. Tardif, S. Gasparini, and J. Barreto, "Camera models and fundamental concepts used in geometric computer vision," *Foundations and Trends® in Computer Graphics and Vision*, vol. 6, no. 1–2, pp. 1–183, 2011.

[3] C. Ricolfe-Viala and A. J. Sanchez-Salmeron, "Lens distortion models evaluation," *Appl Opt*, vol. 49, no. 30, pp. 5914–5928, Oct 2010.

[4] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, Nov 2000.

[5] P. F. Sturm and S. J. Maybank, "On plane-based camera calibration: A general algorithm, singularities, applications," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1, 1999, p. 437 Vol. 1.

[6] C. Hughes, P. Denny, M. Glavin, and E. Jones, "Equidistant fish-eye calibration and rectification by vanishing point extraction," *IEEE Trans Pattern Anal Mach Intell*, vol. 32, no. 12, pp. 2289–2296, Dec 2010.

[7] J. Y. Bouguet, "Camera calibration toolbox for Matlab," 2008.

[8] G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

[9] J. P. Barreto and K. Daniilidis, "Wide area multiple camera calibration and estimation of radial distortion," in *The fifth Workshop on Omnidirectional Vision, Camera Networks and Non-classical cameras*, 2004.

[10] M. Li, *Camera calibration of a head-eye system for active vision.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 541–554.

[11] Q. T. Luong and O. D. Faugeras, "Self-calibration of a moving camera from point correspondences and fundamental matrices," *International Journal of Computer Vision*, vol. 22, no. 3, pp. 261–289, 1997.

[12] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).* IEEE, May 2011, pp. 3400–3407.

[13] A. Richardson, J. Strom, and E. Olson, "AprilCal: Assisted and repeatable camera calibration," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, November 2013.

[14] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment - a modern synthesis," in *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ser. ICCV '99. London, UK, UK: Springer-Verlag, 2000, pp. 298–372.

[15] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[16] T. Svoboda, "Quick guide to multi-camera self-calibration," Tech. Rep., 2003.

[17] H. Hu and G. Kantor, "Instance selection for efficient and reliable camera calibration," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4335–4340.

[18] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. on Robotics (TRO)*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[19] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Journal of Robotics and Autonomous Systems (RAS)*, vol. 57, pp. 1198–1210, Dec. 2009.

[20] "AprilTags C++ Library," http://people.csail.mit.edu/kaess/apriltags/.

[21] L. Kneip and P. Furgale, "Opengv: A unified and generalized approach to real-time calibrated geometric vision," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1–8.

[22] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, June 2011, pp. 2969–2976.

[23] S. Agarwal, K. Mierle, and Others, "Ceres Solver," http://ceres-solver.org.

[24] L. Devroye, *Non-uniform random variate generation*. Springer-Verlag, 1986.

[25] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016.

[26] Google Inc., "Protocol Buffers - Google Developers," https://developers.google.com/protocol-buffers/.

[27] Furgale, P., May J., Rehder J., and Schneider T., "The Kalibr calibration toolbox," https://github.com/ethz-asl/kalibr.

[28] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep. 2005-002, Mar. 2005.

[29] F. S. Grassia, "Practical parameterization of rotations using the exponential map," *J. Graph. Tools*, vol. 3, no. 3, pp. 29–48, Mar. 1998.

Appendix A

Quaternions

Quaternions are a system of numbers that extends the complex numbers $(a + bi)$, invented by William Rowan Hamilton in 1843. A quaternion can be described as (A.1), where i, j and k are hyperimaginary numbers, following the JPL notation [28] which has been used throughout the thesis.

$$\bar{q} = q_4 + q_1 i + q_2 j + q_3 k \tag{A.1}$$

The quaternions lie in the quaternion space, denoted by $\mathbb{H}$, and follow the fundamental quaternion algebra, described in (A.2).

$$\begin{gathered} i^2 = j^2 = k^2 = ijk = -1 \\ -ij = ji = k,\ -jk = kj = i,\ -ki = ik = j \end{gathered} \tag{A.2}$$

They can also be expressed in vectorial form as:

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix} \tag{A.3}$$

In (A.3), q_4 is referred to as the real or scalar part and $\mathbf{q}$ as the imaginary or vector part. An extended analysis of their properties can be found in [28].

Particularly, quaternions can be used among other applications to model 3D rotations, expressing a 3-dimensional rotation using a 4-dimensional vector. In this case the quaternion fulfills (A.4), describing a rotation of θ around the vector $\mathbf{q}$. Also, it is a unit quaternion satisfying $|\bar{q}| = \sqrt{\bar{q}^T \bar{q}} = \sqrt{|\mathbf{q}|^2 + q_4^2} = 1$.

$$\begin{aligned} \mathbf{q} &= \begin{bmatrix} k_x \sin(\theta/2) \\ k_y \sin(\theta/2) \\ k_z \sin(\theta/2) \end{bmatrix} = \hat{\mathbf{k}} \sin(\theta/2) \\ q_4 &= \cos(\theta/2) \end{aligned} \tag{A.4}$$

The representation of a 3D rotation using a 4D vector allows an optimizer to change up to four directions, making possible to move the quaternion off the quaternion sphere, resulting in a non-rotation [29]. To avoid this scenario, we need to introduce a local parameterization of quaternions in the optimization that allows to compute the jacobian of the rotation using quaternions, getting rid of this fourth dimension that could lead to bad-conditioned results.