A UAV-ENABLED CALIBRATION METHOD FOR REMOTE CAMERAS ROBUST TO LOCALIZATION UNCERTAINTY

A Dissertation
Presented to
The Academic Faculty

By

Domitille Commun

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering
Department of Aerospace Engineering

Georgia Institute of Technology

December 2021

© Domitille Commun 2021

A UAV-ENABLED CALIBRATION METHOD FOR REMOTE CAMERAS ROBUST TO LOCALIZATION UNCERTAINTY

Thesis committee:

Prof. Dimitri Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Graeme Kennedy
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Cédric Pradalier
School of Interactive Computing
Georgia Tech Lorraine

Dr. Michael Balchanos
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Olivia Fischer
School of Aerospace Engineering
Georgia Institute of Technology

Date approved: August 9, 2021

ACKNOWLEDGMENTS

I have come now to the end of my journey through graduate school and I would like to acknowledge the people to whom I owe great appreciation in aiding me on this journey.

I am grateful to the members of my thesis committee for their help in preparation of this work. I would like to thank my advisor, Prof. Mavris, for giving me this immense opportunity to join ASDL, and be part of many fascinating projects. I owe Prof. Mavris a great appreciation in helping me grow and become a real scientist and creative thinker.

I would like to express my deep gratitude to Prof. Pradalier for his invaluable guidance, for the amazing knowledge I gained thanks to him, for challenging me to strive for excellence, and inspiring me to become a robotics scientist. I would like to express my thanks to Dr Fischer and Dr Balchanos for their guidance and feedback through this process. Your insightful comments helped me greatly to formulate my ideas, thank you! My gratitude also goes to prof. Kennedy for his time meeting about my PhD and shading new light on my ideas. Prof. Kennedy, thank you very much for your amazing optimization classes that are at the origin of my thesis topic idea!

Many thanks to my family who supported me during this journey. I would like to thank my mother who encouraged me to pursue a PhD, and my father who is always eager to talk about science and engineering topics with me. I would like to thank Alex Braafladt for his meticulous feedback on my work and his help reviewing publications. I could not have gone this far without you!

I would like to recognize my lab-mates and friends who supported me, and gave me great feedback on my thesis work. Thank you Pierrick, Coline, Stella, Darshan, Vincent, Eva, Sam and both Alex! For all your time, all the enhancements and quality you brought to this thesis, merci!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Several video applications rely on camera calibration, a key enabler towards the measurement of metric parameters from images. For instance, monitoring environmental changes through remote cameras, such as glacier size changes, or measuring vehicle speed from security cameras, require cameras to be calibrated. Calibrating a camera is necessary to implement accurate computer vision techniques for the automated analysis of video footage. This automated analysis enables the ability to save cost and time in a variety of fields, such as manufacturing, civil engineering, architecture and safety. The large number of cameras installed and operated continues to increase. A vast portion of these cameras are ”hard-to-reach” cameras. ”Hard-to-reach” cameras refer to installed cameras that can not be removed from their location without impacting the camera parameters or the camera’s operational use. This includes remote sensing cameras or security cameras. Many of these cameras are not calibrated, and successfully being able to calibrate them is a key need as applications continue growing for the use of automated measurements using the video provided by the cameras.

Existing calibration methods can be divided into two groups: object-based calibration, which relies on the use of a calibration target of known dimensions, and self-calibration, which relies on the camera motion or scene geometry constraints. However, these methods have not been adapted for use with remote cameras that are hard-to-reach and have large field-of-views. Indeed, the object-based calibration method requires a tedious and manual process that is not adapted to a large field of view. Furthermore, the self-calibration requires restricted conditions to work correctly and is thus not scalable to a large type of hard-to-reach cameras, with many different parameters, and various viewing scenes. Based on this need, the research objective of this thesis is to develop a camera calibration method for hard-to-reach cameras. The method must satisfy a series of requirements caused by the remote status of the cameras being calibrated:

- Be adapted to large fields-of-view since these cameras cannot be accessed easily (which prevents the use of object-based calibration techniques)
- Be scalable to various environments (which is not feasible using self-calibration techniques that require strict assumptions about the scene)
- Be automated to enable the calibration of the large number of already installed cameras
- Be able to correct for the large non-linear distortion that is frequently present with these cameras

In response to the calibration need, this thesis proposes a solution that relies on the use of a drone or a robot as a moving target to collect the 3D and 2D matching points required for the calibration.

The target localization in the 3D space and on the image is subject to errors, and the approach must be tested to evaluate its ability to calibrate cameras despite measurement uncertainties. This work demonstrates the success of the calibration approach using realistic simulations and real-world testing. The approach is robust against localization uncertainties. It is also environment independent, and highly automated, on the contrary to existing calibration techniques.

First, this work defines a drone trajectory that covers the entire field of view and enables a robust correspondence between 3D and 2D key points. The corresponding experiment evaluates the calibration quality while the 2D localization is subject to uncertainties. It demonstrates using simulations for several cameras that the use of the moving target following this trajectory enables the collection of a complete training set, and results in an accurate calibration with an RMS reprojection error of 3.2 pixels on average. This error is smaller than 3.6 pixels which is a threshold derived in this thesis, and which corresponds to an accurate calibration.

Then, the drone design is modified to add a marker to improve the target detection accuracy. Experiment 2 demonstrates the robustness of this solution in challenging conditions, such as in complex environments for the target detection. The modified drone design leads to improvement in calibration accuracy with an RMS reprojection error of 2.4 pixels on average, and is adapted for detection despite backgrounds or flight conditions that introduce complication in the target detection.

This research also develops a strategy to evaluate the impact of camera parameters, drone path parameters, and 3D and 2D localization uncertainties on the calibration accuracy. Applying this strategy to 5000 simulated camera models leads to recommendations for path parameters for the drone-based calibration approach and highlights the impact of camera parameters on the calibration accuracy. It demonstrates that specific sampling step lengths lead to a better calibration, and demonstrates the relationship between the drone-camera distance and the accuracy. This experiment results in recommendations for the drone path. It also evaluated the RMS reprojection error for the 5000 cameras. The average of this error is equal to 4 pixels. Linking this result to the speed measurement application, 4 pixels error corresponds to a speed measurement error smaller than 0.5km/h when measuring the speed of a vehicle 15 meters away using a pinhole camera of focal length 900 pixels.

The knowledge gained from these experiments is applied in a real-world test, which completes the demonstration of the drone-based camera calibration approach. The real test is made using a commercial drone and GPS, in an urban environment and in a challenging background. This hardware experiment shows the steps to follow to reproduce the drone-based remote camera calibration technique. The calibration error equals 7.7 pixels, and can be reduced if a RTK GPS is used as 3D localization sensor.

Finally, this work demonstrates using an optimization process for several simulated cameras that the sampling size can be reduced by more than half for a faster calibration while maintaining a good calibration accuracy.

CHAPTER 1
MOTIVATION

There is an increase in the number of video cameras used for environmental and wildlife monitoring purposes, manufacturing, architecture, engineering, or safety. This provides many opportunities for the automated analysis of video feeds, which, in turn, allows for the automated detection of objects or the measurement of attributes of interest. This eventually contributes to improvements in engineering methods, industry efficiency, safety, or geo-mapping, to name a few.

Obtaining accurate and reliable information from video feeds in an automated fashion requires that cameras be properly calibrated [1].

This chapter first presents photogrammetry, and computer vision, as fields that enable the measurement of metrics of interest from images. Then, it introduces camera calibration, as a first step toward accurate computer vision applications and image-based measurements. Finally, this chapter discusses the benefits of camera calibration across various applications of camera-based measurements.

1.1 Camera-based measurements

Cameras are commonly used for the modeling and measurement of engineering structures and for quality control. Techniques developed to enable measurements from images have been developed by the photogrammetric and computer vision communities. These two fields and their connections to image-based measurement are presented in the three following sections.

1.1.1 Photogrammetry

Photogrammetry is the science of making measurements from pictures. It processes photographs to provide a map, a drawing, or a measurement as an output [2]. The main principle behind photogrammetric measurements is the geometrical-mathematical reconstruction of the paths of light rays from the object to the camera [3]. Photogrammetry techniques usually focus on a distant object, such as the Earth seen through satellite imaging [4]. When the image processing techniques are applied to an environment closer to the camera, then the corresponding techniques are developed as part of the Computer Vision field.

1.1.2 Computer Vision

Computer vision is a set of techniques that aims at achieving human-level capability in the extraction of information from image data. Its main applications are object recognition, object and environment modeling, and navigation for autonomous vehicles [5].

Recent advances in Machine Learning (ML) techniques allow for the automated analysis of videos. Analytics on video footages help reduce human errors and make human tasks more efficient, contributing towards reduced costs. While humans sense and understand activity within their surroundings, computers observe images as arrays of pixel intensity. Analytics on video footage must be conducted to interpret these inputs, understand images and provide accurate information about detected objects as output. Computer Vision provides techniques that enable the development of artificial systems to obtain information from images [6]. In order to develop performant machine vision systems that can detect objects with good accuracy or provide measurement information from images only, camera calibration is necessary.

1.1.3 Introduction to camera calibration

Photogrammetry and Computer Vision share the same theoretical basis [4] and several goals, such as camera calibration, pose determination or model projection [7]. Photogrammetry and Computer Vision techniques can be used to make real-world measurements from cameras. Making measurement from images requires to establish the geometric relationship between the image and the object, to then extract information from the object using the image. This process is called camera calibration [8].

Camera calibration is the determination of the geometric relation between coordinates in the world and coordinates in the image. A short description of camera calibration is given below. More details about the mathematics behind calibration are provided in Chapter 2. Camera calibration aims to estimate a camera's geometric parameters, such as its focal length, the length distortion, or the location and orientation of the camera [9]. These parameters are used to compute the relation between an object location in the space and its location on the image captured by the camera [10]. When the camera captures an object, its location on the image is given by a 2D vector whose components are in pixel units. Its location in the space is a 3D vector whose components are in distance units (inch, meter). Figure 1.1 illustrates the path from the 3D coordinates of an object to its 2D coordinates. Calibrating the camera is equivalent to determining the mapping between this 3D vector and the 2D vector [11].

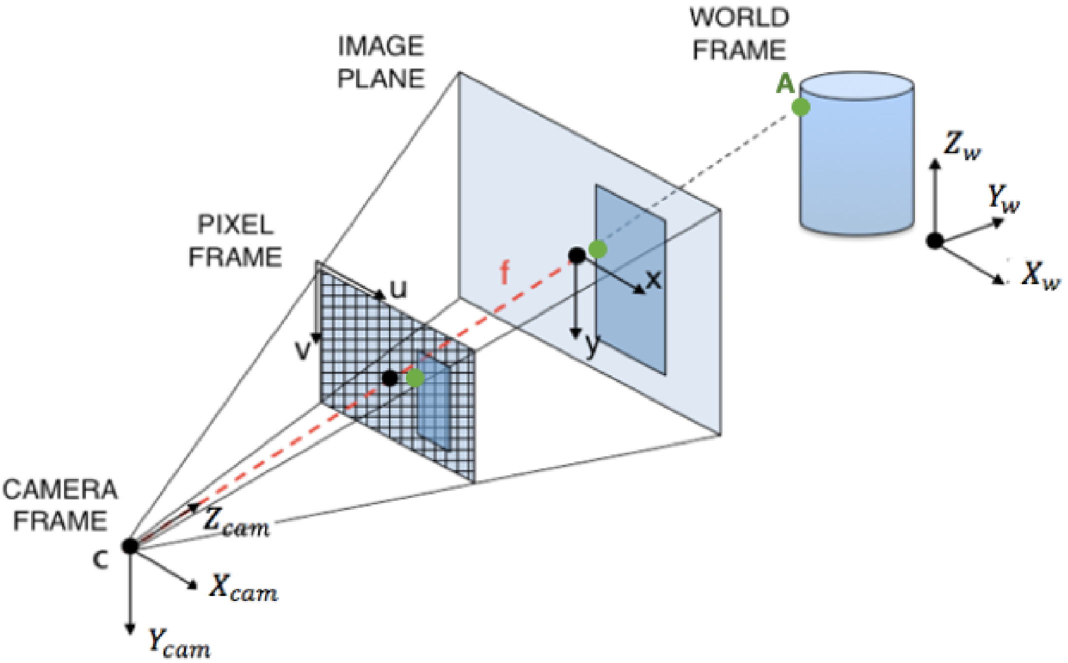


Figure 1.1: Camera calibration: obtain relation between 3D world coordinates and 2D image coordinates

In this example, the 3D object is a cylinder. Let us consider point A on this cylinder. Let us call A_w the coordinates of A in the world frame. The image of this point is, represented in 2D as the green dot in the pixel frame, which will be called a_p. Calibrating the camera consist of determining the relation between A_w and a_p, i.e., determining the mapping from 3D to 2D.

Once the calibration is done, information about an object such as its geometry, its dimensions and its speed can be extracted using the image only.

1.1.4 Summary

The overall performance of computer vision or photogrammetric implementation relies on the accuracy of the camera calibration step [12]. For instance, object detection and their 3D position estimates depend on camera calibration [13]. This step is critical to the acquisition of accurate geometric information such as height or position from the image only [14].

Once a camera is calibrated, an image can be corrected for distortion, reducing error in object detection through computer vision [15]. The following sections provide examples of photogrammetric and computer vision applications that require camera calibration.

1.2 Camera-based measurements in engineering and industry

Photogrammetry is used in civil engineering when a large number of measurement points is required [16]. Photogrammetry enables the measurement of displacements and deformations, which is critical to evaluating potential structural damages. While other means, such as strain gauges or inductive displacement transducers, can be used to perform such measurements, their use is not suited if many points are required. Hence, in such scenarios, cameras are preferred. Photogrammetry is also applied in civil engineering for crack detection and propagation purposes, allowing for the width of potential cracks to be determined [17]. In addition to its use in civil engineering, photogrammetry is also commonly applied in manufacturing. For instance, it is used to manufacture new parts for existing products located in a remote place and for which measurements need to be taken. Examples include covers or decking for boats or new liners/covers for pools [18].

The sections above have provided examples of how cameras can be used for engineering, manufacturing, and modeling purposes. The next section provides a more detailed description of an application of photogrammetry.

1.2.1 An application example in the aerospace industry: store separation testing through photogrammetry

Store separation testing through photogrammetry in aerospace engineering is an application of interest when focusing on the calibration of hard-to-reach cameras. During store separation testing using photogrammetry, two fixed-orientation cameras are used to determine the trajectory of the store that separates from the aircraft to collect data about the safety of the separation [19]. This requires camera calibration.

These two cameras are calibrated using an object-based calibration technique in a dedicated laboratory. After their calibration, they are installed in a photographic pod attached to the aircraft, behind a window, which impacts the calibration made using a 3D object in the laboratory [20]. The traditional calibration technique used for store separation testing through photogrammetry is not efficient because of the large set-up it requires, and leads to a key limit described in the next paragraph.

The calibration is performed before installing the camera behind a window on a part of the aircraft. Thus, the calibration does not account for the window property, the impact of the window on the light transmission, as well as other potential noise such as blurring.

Calibrating the cameras after their installation in the photographic pod should lead to a more accurate store trajectory determination. The calibration would consider additional distortion, potential blurring, reflection, or other noise due to the window [20]. The solution to this obstacle would require a calibration method suitable for hard-to-reach cameras. This is a key observation in this thesis, and is highlighted in the following statement:

> Observation 1: A calibration technique adapted to cameras already installed, that can not be reached easily, would benefit industry

In addition to their benefits for industry applications, cameras are a key tool used for cartography as it is described in the next section.

1.3 Cartography

Cartography is defined as the production of maps. It has various applications such as street-level imagery, 3D modeling, 3D mapping, and geographical survey. Common cartography methods use mobile mapping systems, which rely on using a sensor installed on a moving vehicle to acquire data [21]. Airborne remote sensing systems fall into this category and are used for instance, for agricultural analysis or BIMs development. Three examples of cartography through cameras are presented below.

Street level imagery: Cameras are the primary tool used by Google Street view. They are combined with a transport system such as a car, a bike, or a snowmobile to provide a view at the street level [22].

3D Mapping and localization: 3D mapping aims to build a 3D map of an environment or a 3D model of an object. As described in [23], Red-Green-Blue (RGB) cameras are an efficient tool to build a 3D mapping of the environment. One technique to build the 3D mapping is called structure from motion and requires the camera to move [24]. If only one camera is available, then it is possible to determine a ray in the 3D space where the object must lie on. If two or more views of the objects from two or more cameras or the same camera at different locations are available, then the 3D location of the object can be determined by intersecting the rays [25]. This observation is the key idea behind 3D objects reconstruction.

Geographical survey: A geographical survey aims at recording the coordinates of points of interest. If an infrared sensor is used, problems such as a gas pipeline leak, electric power transmission line weakness, or unexpected pollution can be detected [26]. Cameras are commonly used sensors for geographical surveys [27]. For instance, to perform an aerial survey, it is common to use an IR or visible spectrum camera mounted on a UAV. In [28], topographical changes on coastal areas are measured through this method. UAVs combined with cameras are helpful for agriculture applications. [29] presents two use cases that focus on determining harvesting site locations and inspecting forestry operations. In [30], cameras are mounted on UAVs to enable real-time traffic management.

This last application of camera-based measurement belongs to the field of remote sensing, which represents a key field when considering the applications of this thesis. Since cameras are the main sensor used for remote sensing [21], the next two sections provide some background about cameras as a remote sensing technology.

1.4 Cameras for remote sensing

Remote sensing refers to acquiring information about an object without being in direct physical contact with the object [31]. Remote sensing technologies provide observation and localization of an object of interest, such as a construction or vegetation, at a specific time [21]. Thus they enable data acquisition for topographic mapping. These technologies are used, for instance, to map large forest fires and help rescuers gain knowledge about the fire extend. They are also an enabler to weather prediction by providing cloud imaging [32].

Remote sensing technologies can be divided into the two following groups:

- Terrestrial and airborne platforms, which is the focus of the first subsection
- Spaceborne platforms, which is further detailed in the second subsection

1.4.1 Remote sensing from ground and low altitude platforms

This section introduces remote sensing technologies for environmental changes monitoring and cartography applications as it is a key application of this thesis work.

While airborne platforms have been used for more than a century, static platforms are a newer approach. There were enabled by the advancement and affordability of webcams that can work without requiring maintenance for several months [33]. Remote cameras, such as webcams, are an important enabler for environmental changes monitoring [34]. For example, remote cameras are used to monitor glaciers' retreats due to climate change. Figure 1.2 illustrates one of these static platform used to monitor environmental changes. A key portion of these cameras are not calibrated. Also, weather variation impact the camera internal structure and parameters. These cameras are hard to reach, thus Observation 1 holds true for cartography and remote sensing from static platforms' applications:

Observation 2: A calibration technique adapted to cameras already installed, that can not be reached easily, would benefit remote sensing from static platforms

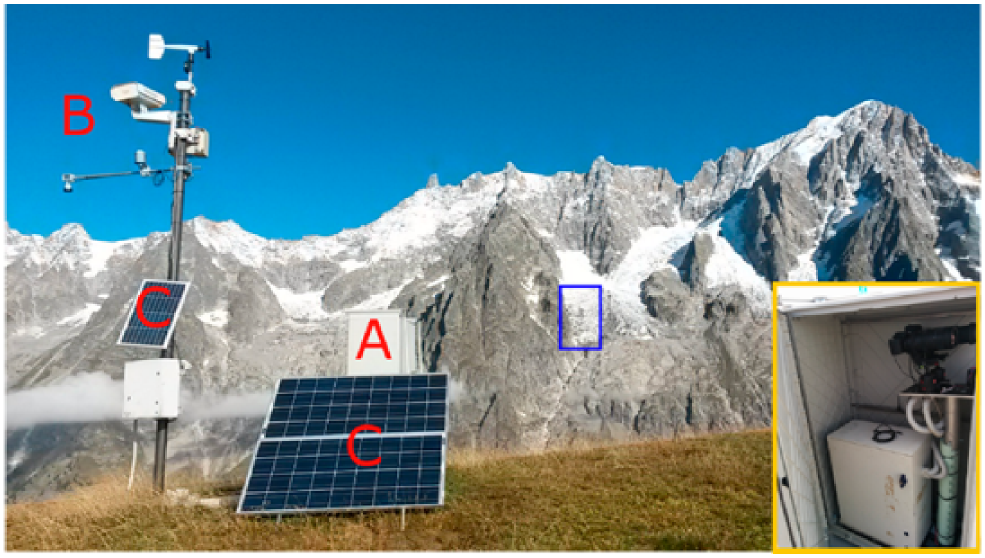


Figure 1.2: Static platform to monitor glacier changes with a camera [35]

Remote sensing technologies are also used for observation of objects from larger distances, such as for Earth Observation. The next section introduces briefly satellite remote sensing through imaging.

1.4.2 Space imaging

The use of satellites for Earth observation purposes started at the end of the fifties, and there are currently more than 150 Earth-observation satellites orbiting. This section briefly introduces some satellite images application since a use case of this thesis is the calibration of those cameras.

Imaging obtained through satellites enable to monitor climate and weather. They collect environmental imaging such as cloud, pollution, or sand storm images. Satellite imaging also detects environmental changes, such as vegetation and urbanization changes, which helps understanding drought or climate changes. Satellites such as MOMS, SPOT-5, ALOS, or ACRTOSAL-1 are major enablers of Earth mapping [36].

Most cameras used on the satellite are linear array cameras. Several line sensors are combined to build a larger virtual linear array camera [37]. These linear array sensors scan the ground while the satellite moves around its orbit, leading to a wider image[38].

These cameras are pre-calibrated before sending them in space, but the conditions during the spacecraft launch, such as changes in air pressure, temperature, or vibration, impact their geometry [39]. The conventional method for their on-orbit calibration relies on the use of control ground points. However, control points might not be available due to weather conditions [39]. Another set of methods rely on autocollimation made possible through modification of the internal camera structure [40]. For instance, the authors in [38] suggest adding a micro-transceiver in the camera interior. However, this requires modifying the camera's internal structure and is not adapted for cameras already installed. Hence, a method suitable for the calibration of hard-to-reach cameras would benefit space camera calibration.

1.4.3 Leverage camera use

While the applications presented previously are existing engineering applications, similar capabilities could be added to many more installed cameras. For instance, the use of security cameras could be leveraged to help reducing the resources required for obstacles localization. If these obstacles are temporary, then it would help taking short-term measures to avoid road accidents. Let's take the example of a tree falling on a road. If one could obtain an estimate of its actual location from the security camera, then the part of the road damaged by this incident could be notified to road users in an automated way. If the obstacle is permanent, such as a fire hydrant system along the road, then one could locate it from the camera and determine its actual location in the world frame to update a map of non-authorized spots to park. Providing the capability to make measurements from images to security cameras, would allow measuring vehicle speed from the installed cameras. In turn, this would reduce the cost required for radar systems. This potential applications rely on the automated processing of security camera images using computer vision techniques. The next section further investigate the advantage of camera-based measurement for safety applications.

1.5 Camera-based measurement for safety applications

1.5.1 Growth in the number of security cameras for safety purpose

Security video cameras help to minimize response time of rescue teams, maximize operational effectiveness and help to reduce the number of incidents. Indeed, strategies can be immediately deployed when certain situations arise that are detected using video cameras. For this reason, cameras help to reduce the number or impact of incidents. Moreover, they improve situational awareness, which helps manage and coordinate responses because they provide real-time access to visual information [41].

The number of security cameras continuously increases worldwide. In 2012, North America had around 33 million cameras. In 2016 this number had reached 62 million. Figure 1.3 provides a snapshot of the evolution of the number of security camera bases installed [42]. A base can have multiple cameras.

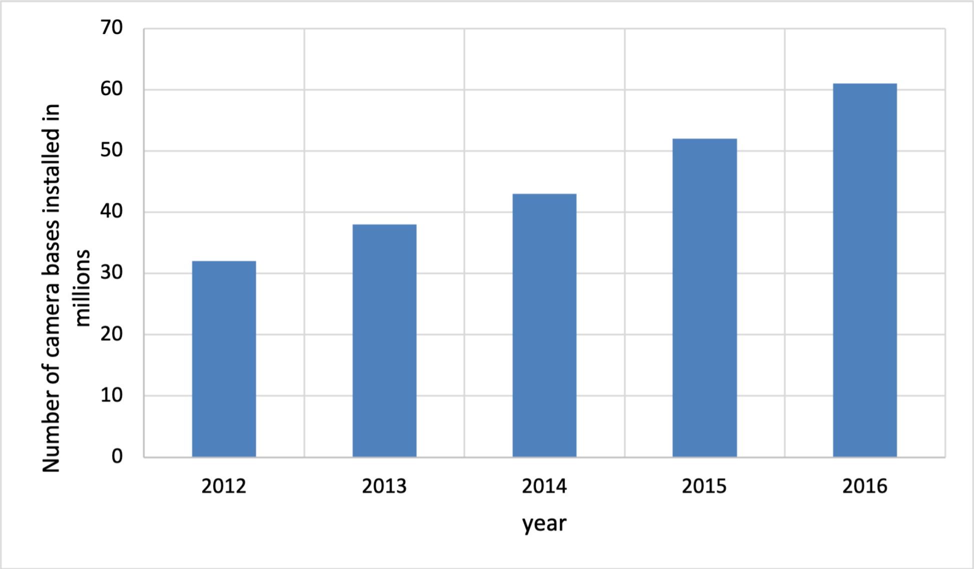


Figure 1.3: Number of security camera bases installed in North America from 2012 to 2016

Large cities across the world count hundreds of thousands of security cameras. For instance, London has spent more than 300M£ to install closed-circuit television (CCTV) units between 2007 and 2010 to help reduce crime and antisocial behavior. London counts more than 500,000 CCTV units, while the city has around 10 million people [43]. More than 30,000 cameras are used by the police in Chicago to monitor the city and help reduce violence [44].

The location and use of security cameras are broad: such places include train and subway stations, airports, offshore platforms, office buildings, schools, natural environments. New uses of security cameras are developed frequently. For instance, in France, wildlife monitoring cameras were deployed recently in the Fontainebleau forest to detect and track illegal polluters [45]. The same cameras are used in some African natural parks to reduce poaching. As mentioned earlier, capabilities could be added to these cameras for safety purposes, such as facilitating accurate object detection or measuring vehicles' speed.

Security cameras help to minimize the response time of rescue teams, maximize operational effectiveness and help to reduce the number of incidents. Their use could be further optimized by provided automated analysis of their video.

1.5.2 Multi-task, cost reduction and safety improvement through calibrated security cameras

Accurate object detection has many applications when working with security cameras. As an example, images could be corrected from their distortion, improving the object detection step, and in turn helping the detection of threats, such as fire or non-authorized vehicles. Calibration is also a requirement to get metric information from images. Thus, the calibration of security cameras is an enabler for obstacles height or location estimation or automated vehicle speed measurement [46]. Since these cameras are already installed, adding these capabilities would optimize their use and reduce resources currently required for the tasks mentioned above.

Security cameras are rarely calibrated. Indeed while describing the procedure for their installation, providers do not specify any requirement concerning calibration [47] [48]. For example, Atlanta's Georgia Tech campus counts more than 2000 cameras, but none of them are calibrated. Their calibration would enable to optimize safety applications as described previously. In the future, it could also provide capabilities aside from safety purposes to these cameras.

Some studies on the automated analysis of security camera footage focus on congestion detection using cameras [49] or vehicle speed measurement [46]. These last safety applications could be expanded to the large network of security cameras already existing, but this would require that security cameras are calibrated.

1.5.3 Speed measurement

Security cameras are used to monitor traffic and are capable of measuring the speed of vehicles [50] [51]. First, the vehicle is automatically detected using a computer vision technique. Then, the vehicle location on the image is determined, and the vehicle flow, which is the speed in pixels per second, is obtained. Finally, using the correspondence pixel to metric, the speed in kilometers per hour can be computed [50].

However, if the camera is not calibrated, the speed measured is not accurate, as is explained in [52] and [51]. In [53], several vehicle speed measurements are made using an uncalibrated camera. The camera-based measurement is compared to the speed measurement made using a GPS, which is considered as ground truth in this paper. The accuracy of the vehicle speed measurement made in [53] is lower than 70 percent across several trials. Indeed, while the GPS provides a vehicle speed equal to 43km/hr, the camera-based measurement provides a speed equals to 66km/hr. Another trial provides a GPS-based speed of 38km/hr and a camera-based speed of 55km/hr. Hence, the method applied on uncalibrated cameras leads to errors, and calibration is required for better speed estimation. In [54], the camera is calibrated, and the error of the vehicle speed measurement is equal to 3.5 percent for the worst trial, which is an improvement compared to the method where no calibration had been conducted [53].

Millions of cameras, and security cameras, in particular, are already installed but not yet calibrated. Additional capabilities could be enabled for these cameras. However, to do so, cameras must be calibrated and then computer vision techniques implemented.

> Observation 3: A calibration technique adapted to cameras already installed, that can not be reached easily, would benefit safety applications

1.5.4 Summary

Today, cameras contribute to improvements in a number of domains such as manufacturing, civil engineering, safety, environment monitoring. It is possible to facilitate the processing of videos and, consequently, of the use of cameras, by implementing automated analytics of video footages from calibrated cameras.

The photogrammetric and computer vision applications presented in this chapter share a common characteristics: they rely on the use of cameras that cannot be reached easily. The following section provides a definition for hard-to-reach cameras, and summarizes the benefits of their calibration.

1.6 The benefits of hard-to-reach camera calibration

”Hard-to-reach cameras” refer to installed cameras that can not be removed from their location without impacting the camera parameters or the task operated by the camera. This group includes, among others, remote sensing cameras and security cameras. These cameras are already installed outdoor or indoor, are usually high, and can not be reached easily. Hard-to-reach cameras refer to a broader range of cameras whose extent does not stop to security cameras. It also refers to cameras used for remote sensing. For example, it includes the ones on static platforms for glacier monitoring or satellites’ cameras. This group includes as well cameras used in a set-up that makes them difficult to access, such as cameras used for store separation testing. A technique adapted to the calibration of hard-to-reach cameras would benefits many fields, such as the aerospace industry through aerial remote sensing or store separation through photogrammetry, or the safety sector with automated speed measurement or obstacle localization through security cameras. This would also benefits environmental changes exploration. Remote sensing is an essential tool for environmental monitoring and for determining shapes and dimensions of objects' of interest located on the ground [36] [21]. Two types of remote sensing approaches use cameras that are hard to reach: static platforms and satellites. Accurate use of these cameras require that they are calibrated.

However, a key portion of these cameras are not calibrated. For instance, most security cameras are not calibrated, and their number was over 60 million in North America in 2016. In [35], authors installed a camera for glacier monitoring but they did not calibrate the camera before using it, which limits the automated analysis of its video footage. This lead to this main observation:

> Main observation: An efficient technique for the calibration of those hard-to-reach cameras would have benefits in many domains such as for environmental changes monitoring, in safety, and in industry.

In addition to the need for calibration, the internal structure of the camera is impacted by external factors, requiring the re-calibration of the camera. Thus, an efficient method for the calibration of hard-to-reach cameras would help calibration and re-calibrating the many cameras used for remote sensing, safety, and some industry applications.

1.6.1 Need for re-calibration

Temperature changes impact the focal length and the principal point location, which is the intersection of the optical axis and image plane. Figure 1.4, Figure 1.5, and Figure 1.6 illustrate the impact of weather on these parameters for three different trials (blue, red, and green plots) using the same camera and implementing the same increase in temperature [55].

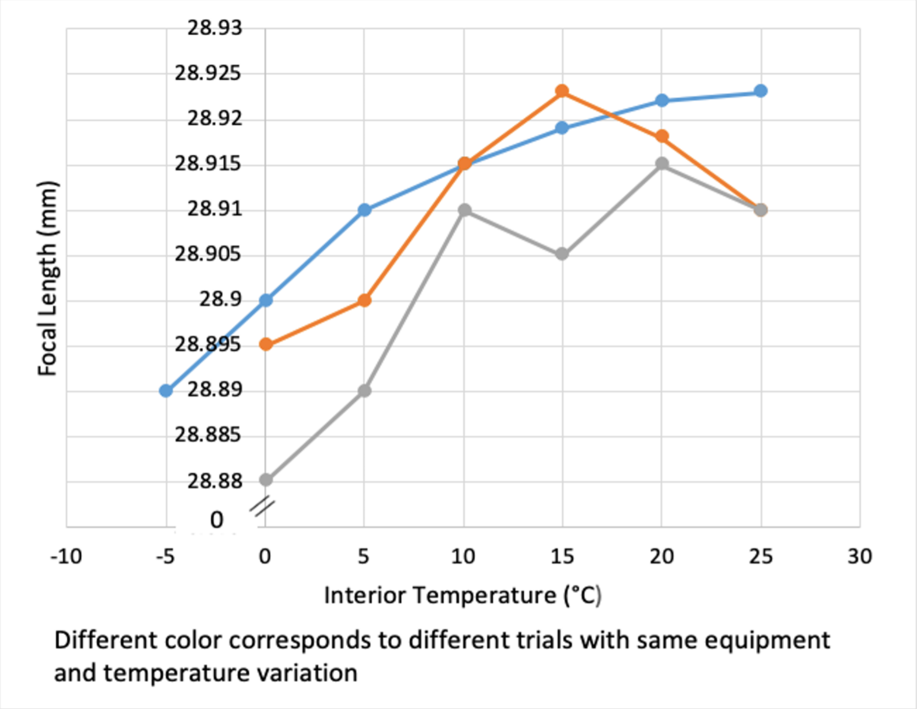


Figure 1.4: Effect of the temperature on the focal length

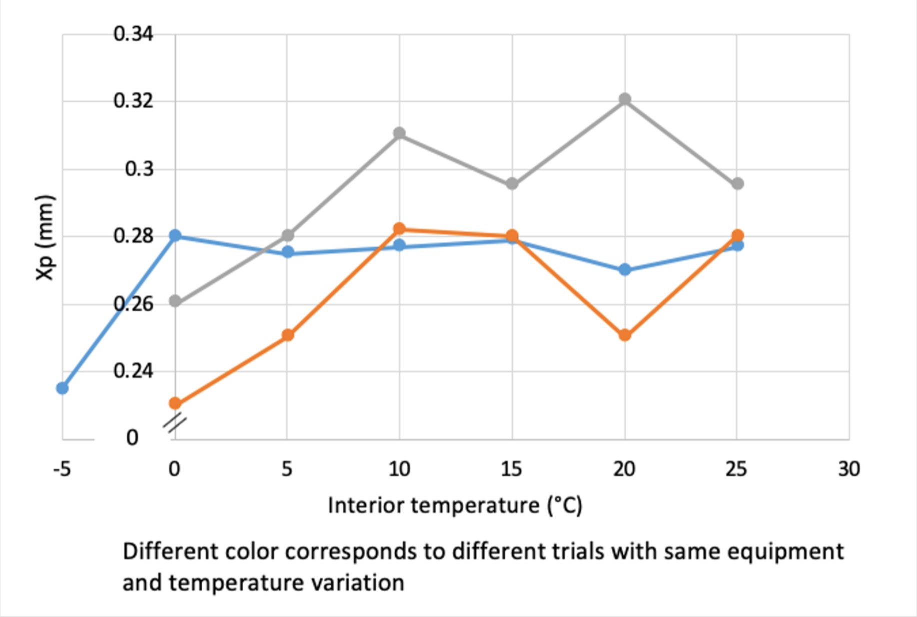


Figure 1.5: Impact of the temperature on the horizontal position of the principal point

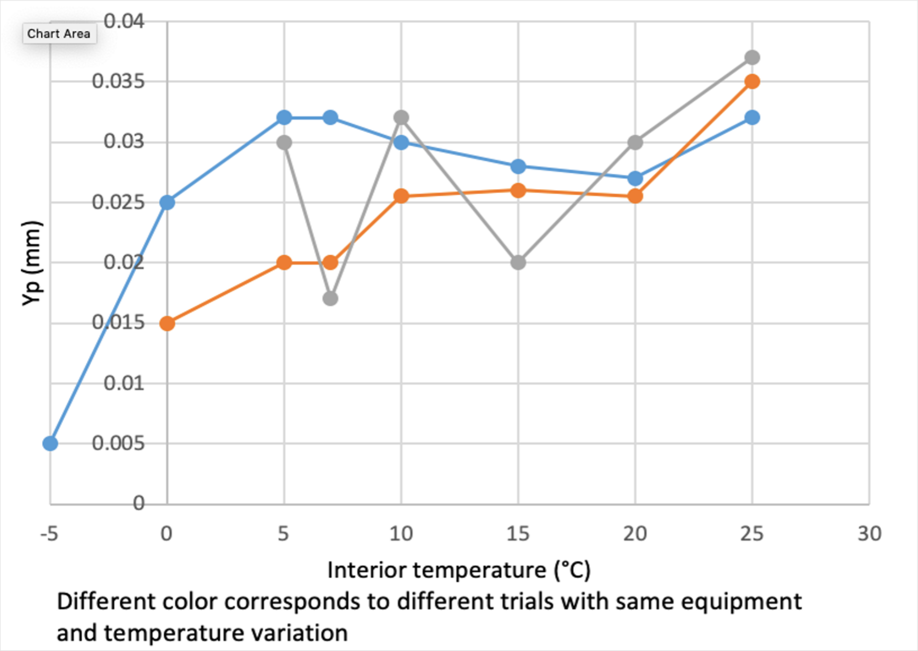


Figure 1.6: Impact of the temperature on the vertical position of the principal point

The weather has an impact on camera geometry, as shown with the trend observed for temperature changes, and consequently, it has an impact on camera parameters [56][57]. These parameters are involved in the mapping 3D-2D, i.e., the calibration. This requires that re-calibration be conducted. Consequently, there is a need for methods to re-calibrate cameras already installed, and located in a large variety of environments.

Environmental conditions also impact satellite cameras. Temperature changes and drying out effect are responsible for the modification of the camera's parameters. Indeed, they lead to unpredictable changes in the camera's internal geometry [37]. Consequently, satellites' cameras are periodically calibrated when moving on their orbits [58].

1.7 Overarching Research Objective

This chapter presented the advantages of camera calibration, which is a necessary step to make measurements from images only about objects located in the real world. From a monocular camera which is calibrated, and one known geometrical dimension in the world frame (this can be knowing the object is on the ground), the actual location of an object can be obtained. This chapter focused on hard-to-reach cameras such as remote sensing cameras on static platforms, or security cameras. An efficient method for their calibration, and re-calibration, would benefit many applications in environmental changes monitoring, safety, or industry. It would allow to provide additional capabilities to these cameras, which are already installed. Thus, the research objective of this thesis is the following:

Research Objective: Developing an efficient calibration technique for hard-to-reach cameras

1.8 Thesis structure

This present chapter motivated the need for calibration of hard-to-reach cameras. Chapter 2 further presents the camera calibration process and identifies challenges that must be overcome to calibrate hard-to-reach cameras and so to reach the research goal. It proposes a solution for the calibration of hard-to-reach cameras, and presents the challenges that the solution must overcome to reach the Research Objective. The remaining chapters formulate Research Questions whose answers provide an evaluation, improvement and validation of the proposed solution. These remaining chapters also present the Experiments run to test Hypotheses that are proposed answers to the Research Questions. This set of Research Questions, Hypotheses and Experiments lead to the validation of the proposed calibration solution. In a last chapter, the main contributions of this thesis are summarized and directions for future work are provided.

CHAPTER 2
RESEARCH FORMULATION

The Overarching Research Objective is recalled in the following statement:

> The research objective of this thesis is to develop an efficient camera calibration method for hard-to-reach cameras.

The key word efficient refers in this context to a method adapted to the large diversity of environments where these cameras are located: from urban areas for security cameras, to natural environments for remote sensing static platforms. It also refers to a method that can be implemented rapidly due to the large number of already installed cameras, and the need of re-calibration.

This chapter presents the mathematics behind camera calibration to understand better what is to be solved to reach the objective of this research. It presents existing calibration methods and their limits for their application to hard-to-reach cameras. Based on the drawbacks of existing calibration methods, the research focus of this thesis is refined, and an Overarching Research Question is formulated.

2.1 Mapping world coordinates to image coordinates

In a first sub-part, camera calibration and the transformations involved in the process are presented. In a second sub-part, the equations that relate 3D coordinates to 2D ones are derived.

2.1.1 Transformations between world frame and image frame

Calibrating a camera consists of determining the relation between the 3D space coordinates of an object and their projected 2D image coordinates [12]. Figure 2.1 depicts the projection of the 3D coordinates PO into the 2D coordinates Po through a pinhole camera. A pinhole camera is a simple camera that does not have a lens but a small aperture instead (point O on the figure).

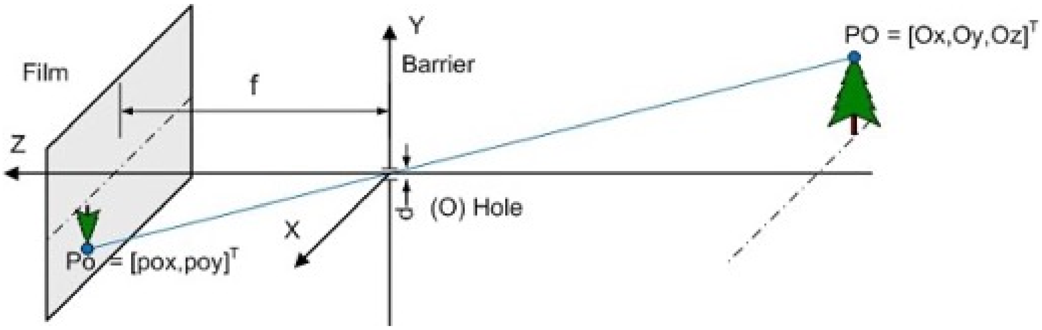


Figure 2.1: Light path for a pinhole camera [59]

The mapping between the 3D space coordinates and the 2D camera coordinates is derived using a succession of transformations:

- The first one is from the world coordinates X_w, Y_w, Z_w, to the camera coordinates X_{cam}, Y_{cam}, Z_{cam}. The world coordinate system is a global one, while the camera coordinate system depends on the camera location and orientation, and has for origin the camera center. Figure 2.2 illustrates this change of coordinates. Any transformation between two 3D coordinate systems is made of translations and rotations. Hence the relation between these two frames can be expressed using rotations and translation matrices multiplied as developed later in this chapter.

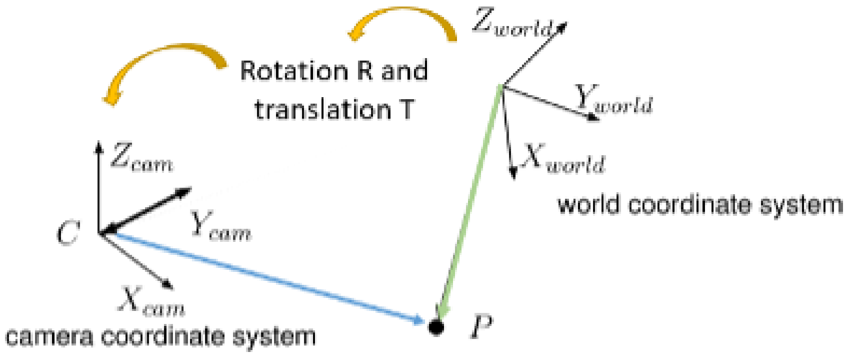


Figure 2.2: Point P seen from the world coordinate system (green vector) and the camera coordinate system (blue vector)

- The second transformation is from the camera coordinates (X_{cam}, Y_{cam}, Z_{cam}) to the image coordinates (x, y). This transformation is a projection of the object on the 2D image plane to which the lens distortion effect is added. Figure 2.3 depicts the transformation from 3D to 2D coordinates when it is assumed no distortion (projection in the image plane).

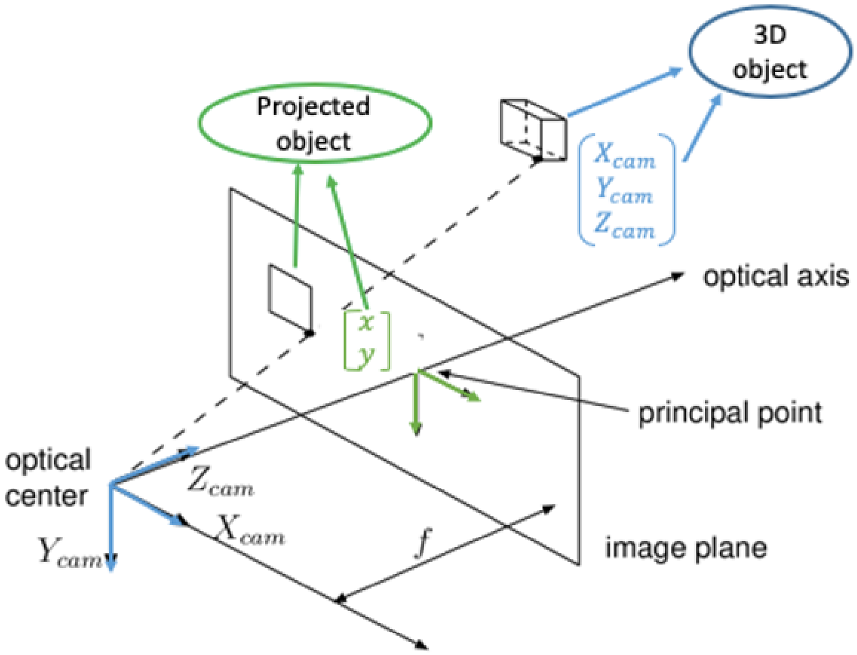


Figure 2.3: Projection 3D to 2D [60]

- The third transformation model the image deformation due to lens properties or camera orientation. This deformation is called distortion

- The last transformation maps the image coordinates (x,y) to the pixel coordinates (u, v). This last frame has an origin located at the top left corner of the image, and its axes are in pixel unit [61]. Figure 2.4 illustrates the difference between the image coordinate system and the pixel one.

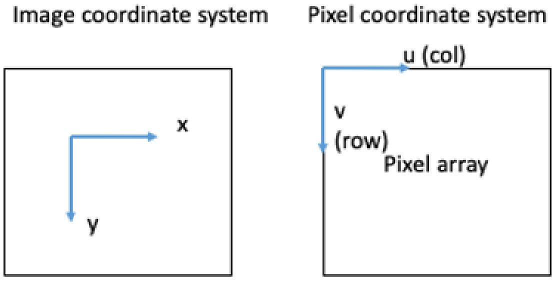


Figure 2.4: From image coordinates system to pixel coordinates system

- Figure 2.5 and Figure 2.6 below illustrate this succession of transformations for a cylinder located in the world frame. For each point of the cylinder, its coordinates are transformed from the world system to the camera system. Then the point is projected into the image plane, and finally, its coordinates are transformed into pixel coordinates.

The first transformation can be described by a matrix of rotation R and one of translation T, which define the location and orientation of the camera. The components of these matrices are parameters that are external to the camera geometry. The three other relations: the projection, deformation, and transformation affine, are made of internal camera's geometric and optical characteristics such as the focal length, the central point location, or the lens distortion coefficients. Hence, the relation between the 3D and 2D coordinates relies on internal and external camera parameters. Calibrating the camera consists of determining these parameters.

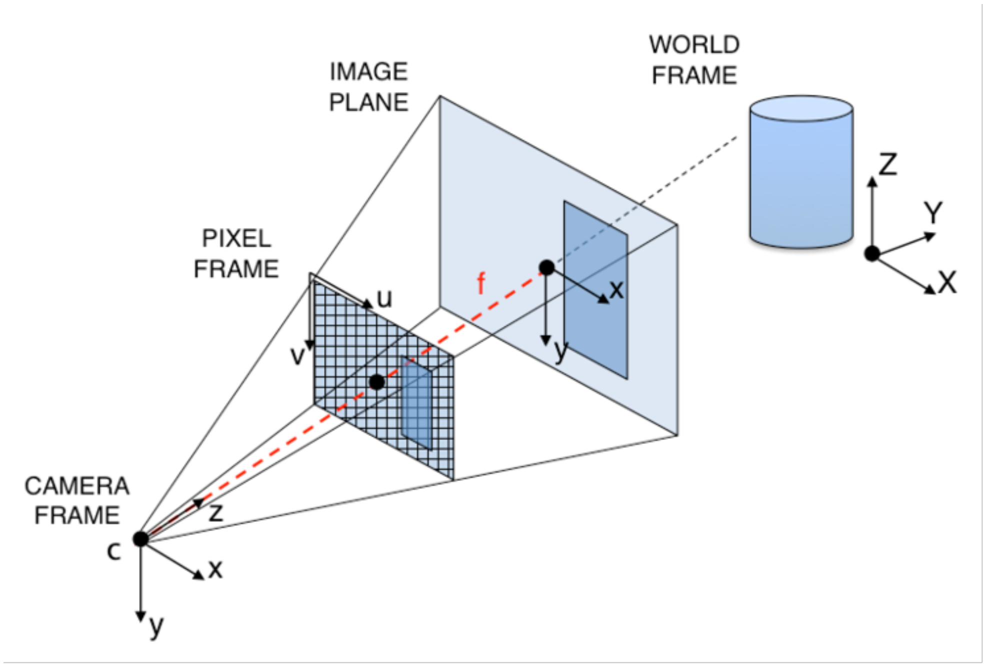


Figure 2.5: World, camera, image and pixel frames [62]

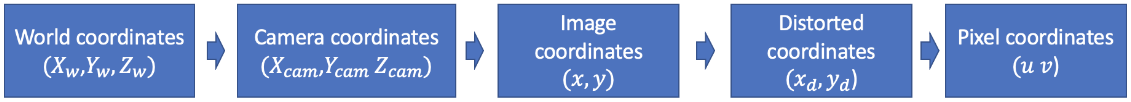

World coordinates (X_w, Y_w, Z_w)
Camera coordinates $(X_{cam}, Y_{cam}\ Z_{cam})$
Image coordinates (x, y)
Distorted coordinates (x_d, y_d)
Pixel coordinates $(u\ v)$

Figure 2.6: Succession of transformations

These parameters are sorted into two groups: Intrinsic and Extrinsic parameters. The following section provides equations for the transformations presented above that are made from cameras' parameters.

2.1.2 Camera parameters and equations from world to image frames

As mentioned, the camera parameters are divided into two groups. Extrinsic parameters define the location and orientation of the camera and are stored in a matrix [R T], where R is a rotation matrix and T a translation matrix. Intrinsic parameters are related to the camera's interior components and are involved in the mapping between the camera and the pixel coordinate systems [63]. The intrinsic parameters encompass:

- The focal length 'f'. For a thin length, it is the distance between the center of the lens and the focal point. The focal point (F in Figure 2.7) is the point where rays parallel to each other meet after passing through the lens [64].

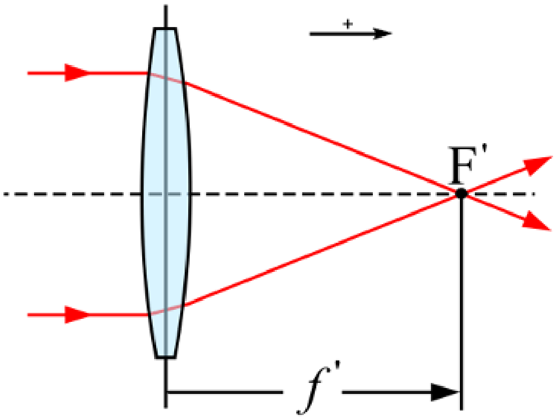


Figure 2.7: Focal length F

- The principal point location: it is the intersection of the optical axis and the image plane. It is the point pp on Figure 2.8 [65].

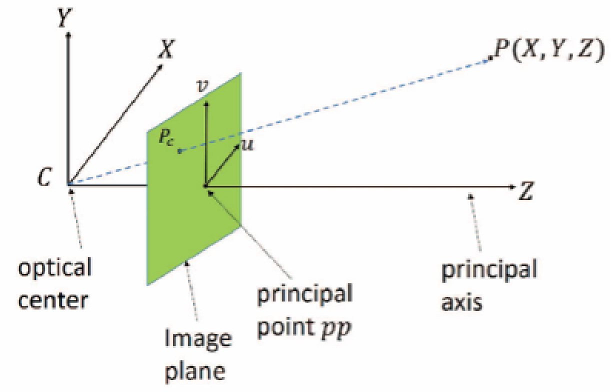


Figure 2.8: optical axis and principal point [66]

- The distortion or image deformation: Perspective distortion is due to the camera orientation. Optical distortion is due to the design of the lens used in the camera. The prominent lens distortion effect is called radial distortion, and can be divided into two groups: barrel and pincushion distortion [67]. These two lens distortion effects are depicted in Figure 2.9. The left image represents a picture without distortion. The middle one represents a picture deformed due to barrel distortion that causes straight lines to appear as convex curves. The right image depicts pincushion distortion that causes straight lines to be bowed inwards. In the equations derived later in this section, the distortion coefficients are written k_i, and p_i.

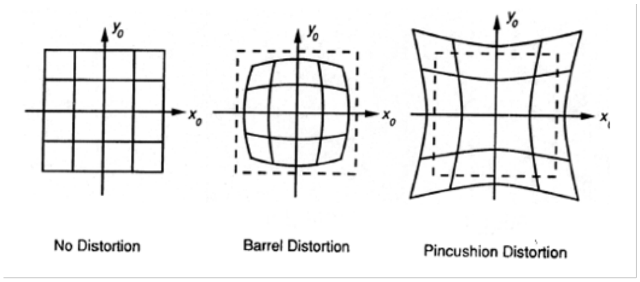


Figure 2.9: Distortion caused by lens [68]

- Skew coefficients: it is the number of pixels per unit length in each direction. It is often written as $\frac{1}{s_x}$ for the horizontal direction and $\frac{1}{s_y}$ for the vertical direction.

With the intrinsic and extrinsic parameters I introduced, the equations relating 3D coordinates to 2D pixel coordinates can be derived. The derivation is presented below because these equations are central to the calibration process. Also, their derivation is necessary to identify which part of the calibration process is well defined and which part could include innovations to enable a calibration approach adapted to hard-to-reach cameras. This derivation is also necessary to clearly understand most hypotheses.

First, the linear model is introduced. This model is derived using the pinhole camera model, which is depicted in Figure 2.10. This type of camera replaces the lens with a small aperture, and thus, their model does not take the blurring and geometric distortion into account [69].

This model assumes no distortion, enabling the representation of all intrinsic parameters within a matrix called K. Using a single matrix K to represent intrinsic coefficients is not possible when including the lens distortion. Indeed, in that case, the relation between (x,y) and (u,v) is not linear.

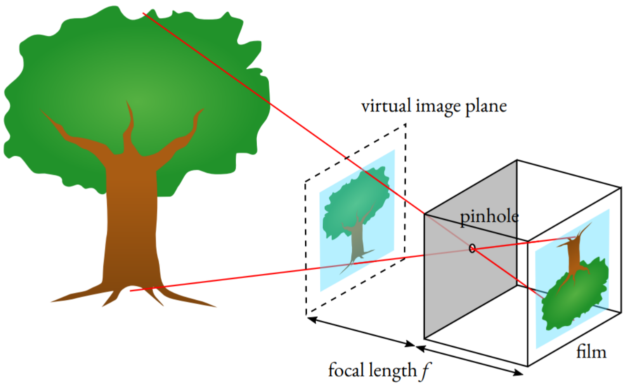


Figure 2.10: Pinhole camera model [70]

Figure 2.11 summarizes the transformations between the frames using the intrinsic parameters (matrix K) and the extrinsic ones (matrices R and T).

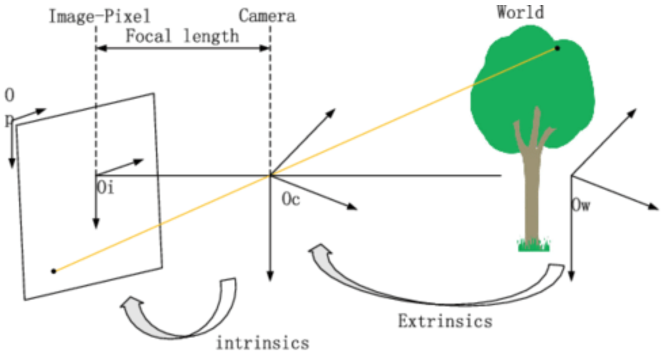


Figure 2.11: Intrinsic and extrinsic parameters to map 3D coordinates to 2D pixel coordinates [71]

A succession of rotations and translations define the transformations from the world frame to the camera frame. The translation vector is written $\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$ where t_x, t_y, and t_z are the translation along X_w, Y_w, and Z_w. The rotation matrix coefficients m_{ij} are obtained by multiplying the three matrices that define the rotation along X_w, Y_w, and Z_w. Consequently, the coefficients m_{ij} are functions of the angle of rotation along X_w, Y_w, and Z_w. For example, m_{11}=$\sin{(\alpha_x)} * \sin{(\alpha_y)} * \cos{(\alpha_z)} - \cos{(\alpha_x)} * \sin(\alpha_z)$

where α_x, α_y, and α_z are the rotation angles around X_w, Y_w, and Z_w. The axes and the angles mentioned here are depicted in Figure 2.12.

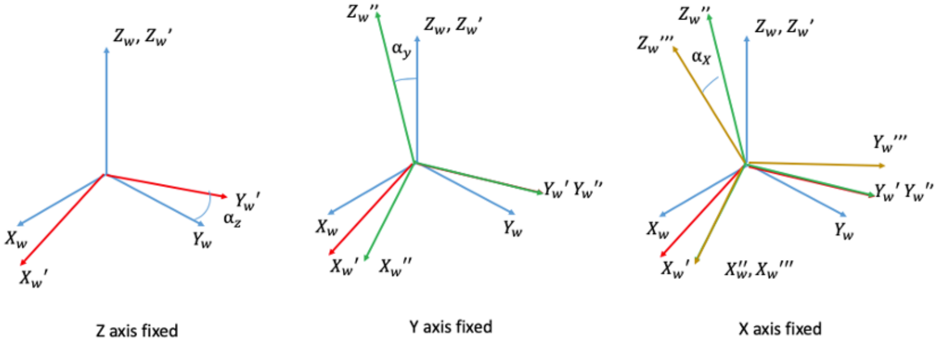


Figure 2.12: World coordinate frame and Euler angles

From the rotation and the translation coefficients, the relation between the world frame and the camera frame is derived and is given by the following equation [38]:

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} * \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} + \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} * \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

By writing t_1, t_2, t_3, the translation coordinates in the camera frame, the above equation can be written as:

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} * \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Figure 2.13 provides a drawing of the simple path followed by the light through a pinhole camera. From this picture, the relation between the camera coordinates and the image coordinates is derived using Thales' theorem or similar triangles:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z_{cam}} * \begin{bmatrix} X_{cam} \\ Y_{cam} \end{bmatrix}$$

So far, the relation from the world coordinate system to the camera one and the pro-

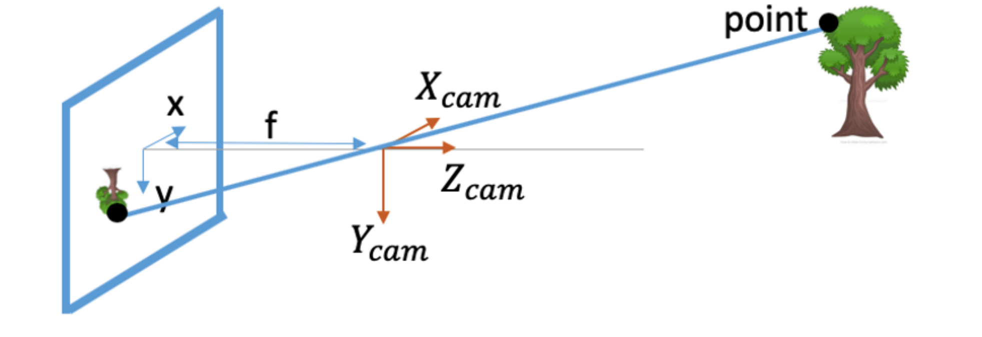


Figure 2.13: Light path from object to its image for a pinhole camera

jection from the camera frame to the image frame have been derived. Now the relation between the image frame and the pixel frame will be derived. The main difference between the image frame and the pixel frame is the axes center location. The center of the axes is shifted from the principal point to the top left corner, as typical in computer vision [38]. Figure 2.14 depicts the frame center shift.

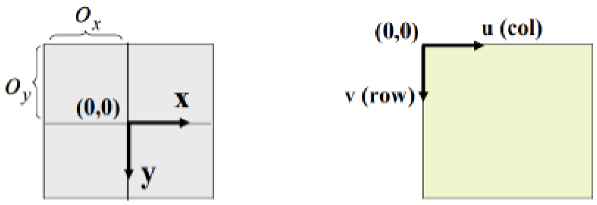


Figure 2.14: Frame center shift

Hence, the new coordinates become $x+O_x$ and $y+O_y$ where $\begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z_{cam}} * \begin{bmatrix} X_{cam} \\ Y_{cam} \end{bmatrix}$. Finally, the metric unit is converted into a pixel unit. For this purpose, the skew coefficients $\frac{1}{s_x}$ and $\frac{1}{s_y}$ are used, and the pixel coordinates can be defined as follow:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{1}{s_x} * (x + O_x) \\ \frac{1}{s_y} * (y + O_y) \end{bmatrix} = \begin{bmatrix} \frac{1}{s_x} & 0 & \frac{O_x}{s_x} \\ 0 & \frac{1}{s_y} & \frac{O_y}{s_y} \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{s_x} * \left(\frac{f}{Z_{cam}} * X_{cam} + O_x \right) \\ \frac{1}{s_y} * \left(\frac{f}{Z_{cam}} * Y_{cam} + O_y \right) \end{bmatrix}$$

The equations for each transformation derived above assumed no lens distortion. These equations can be summarized by a linear equation using matrices and homogeneous vectors. Homogeneous points are useful to write linear equations that include vectors of dimension n and dimension n+1 [48] (in this thesis dimension 2 for the projection and dimension 3 for the real world coordinates system). For instance, the point $\begin{bmatrix} u \\ v \end{bmatrix}$ will often be written using the homogeneous point $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$.

Using matrices and homogeneous coordinates, the following equations are obtained:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{s_x} & 0 & \frac{O_x}{s_x} \\ 0 & \frac{f}{s_y} & \frac{O_y}{s_y} \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \frac{X_{cam}}{Z_{cam}} \\ \frac{Y_{cam}}{Z_{cam}} \\ 1 \end{bmatrix} => \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f}{s_x} & 0 & \frac{O_x}{s_x} & 0 \\ 0 & \frac{f}{s_y} & \frac{O_y}{s_y} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} \frac{X_{cam}}{Z_{cam}} \\ \frac{Y_{cam}}{Z_{cam}} \\ 1 \\ \frac{1}{Z_{cam}} \end{bmatrix}$$

Let's write $f_x = \frac{f}{s_x}$ (respectively $f_y = \frac{f}{s_y}$) the focal length in pixel, $C_x = \frac{O_x}{s_x}$ (respectively $C_y = \frac{O_y}{s_y}$), the image center coordinates in pixels. The system of equation can be written as:

$$Z_{cam} * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix}$$

From Figure 2.14 or by using the Thales theorem or from equation 3, it can be stated that the points that belong to the ray $s * \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix}$ have the same projection $\begin{bmatrix} u \\ v \end{bmatrix}$, so they are equivalent [57].

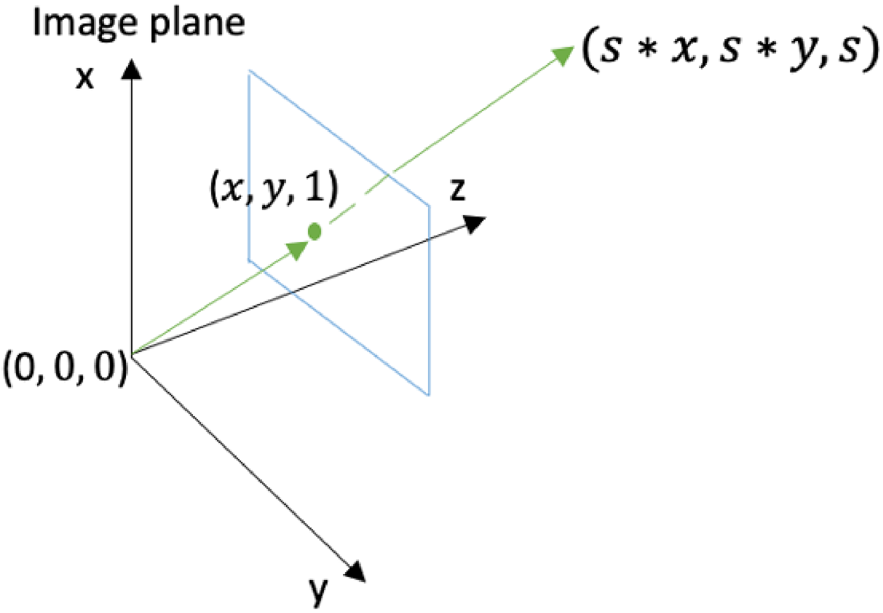


Figure 2.15: Same projection $(x, y, 1)$ for all points $s * (X_{cam}, Y_{cam}, Z_{cam})$

Consequently, the relation between camera coordinates and projected coordinates can be written, using s a real number, as:

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix}$$

Since the camera coordinates are related to the world coordinates with this relation:

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_1 \\ m_{21} & m_{22} & m_{23} & t_2 \\ m_{31} & m_{32} & m_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

The relation between pixel coordinates and world coordinates for a pinhole camera is given bellow:

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_1 \\ m_{21} & m_{22} & m_{23} & t_2 \\ m_{31} & m_{32} & m_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

This equation is central in the calibration process: it shows the relation between 3D and 2D coordinates. This relation depends on intrinsic and extrinsic parameters which must be determined to calibrate the camera. To do so a regression is implemented using several points of known 3D and 2D coordinates [12]. Note: a point in the 2D image is treated as a ray in the 3D frame. Once the camera is calibrated, it is possible to determine a ray in the 3D space that the object must lie on [72]. However it is not possible to get back to the 3D coordinates from the 2D coordinates if only one camera is available. With two views taken by two cameras or by the same camera in two locations it is possible to determine the exact position of an object by intersecting the two rays [72]. The model derived above is a linear model and is an approximation that does not include the lens effect. However, most cameras have a lens. The derivation is completed below to include the effect of lens distortion.

First, let us start from the equation below that relates camera coordinates to pixel coordinates and was derived earlier in the chapter.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} \frac{X_{cam}}{Z_{cam}} \\ \frac{Y_{cam}}{Z_{cam}} \\ 1 \end{bmatrix}$$

To obtain a more accurate model, the pinhole model is combined with the correction for both radial and tangential distortions. Radial distortion is due to the spherical shape of the lens, tangential distortion is caused by the difference between the normal to the lens components and the optical axis [73]. The most common distortion is the radial one. It is modelled using parameters k_1 and k_2 that need to be estimated. k_1 accounts for a large part of the distortion (usually 90 percent of it) [74]. Let us write $x' = \frac{X_{cam}}{Z_{cam}}$, and $y' = \frac{Y_{cam}}{Z_{cam}}$, the relation for radial distortion is given by:

$$\delta x'_{radial} = x' * (k_1 * r^2 + k_2 * r^4)$$

$$\delta y'_{radial} = y' * (k_1 * r^2 + k_2 * r^4)$$

$$r^2 = x'^2 + y'^2$$

Another common distortion is the tangential distortion and it is modelled as follow, where p_1 and p_2 are coefficient for tangential distortion [12].

$$\delta x'_{tangential} = 2p_1 x' y' + p_2 \left(r^2 + 2x'^2 \right)$$

$$\delta y'_{tangential} = 2p_2 x' y' + p_1 \left(r^2 + 2y'^2 \right)$$

These corrections are then added to the image coordinates (x’,y’) which lead to the following corrected coordinates:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} x' + \delta x'_{radial} + \delta x'_{tangential} \\ y' + \delta y'_{radial} + \delta y'_{tangential} \\ 1 \end{bmatrix}$$

The shape of the function that relates 3D world coordinates to their 2D pixel coordinates has been derived above. The camera parameters are unknown and need to be determined to build this function and thus calibrate the camera. The calibration problem is similar to a non-linear regression problem [75]. Indeed calibration consists of determining unknown parameters of a function, as illustrated with Figure 2.16. The following section describes the model fitting steps and their application for camera calibration purposes.

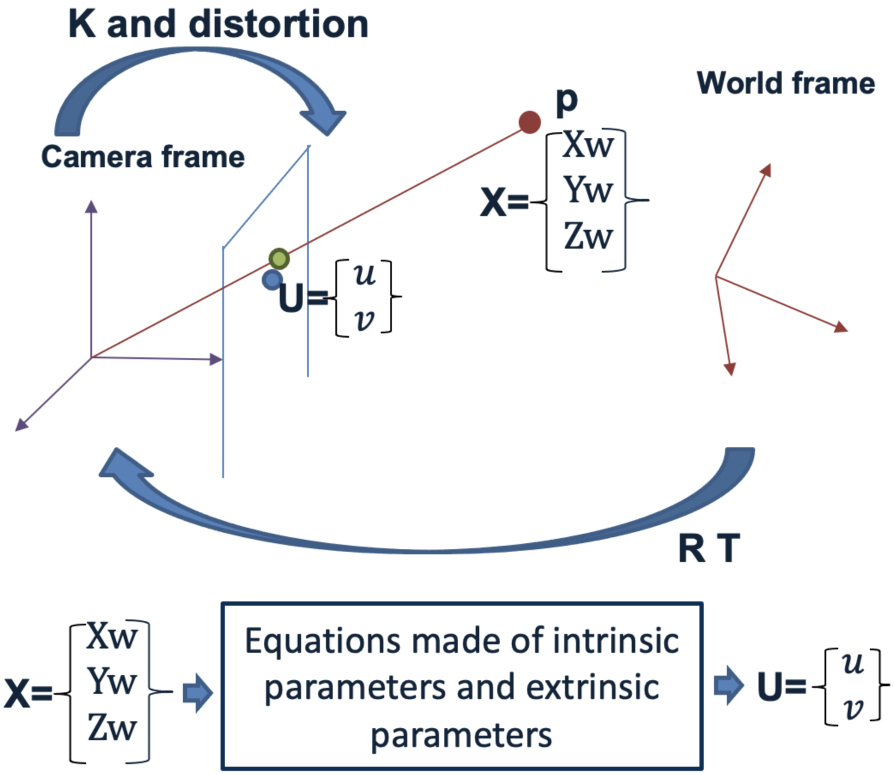


Figure 2.16: Calibration is a ”model fitting” problem

2.2 Non-linear Regression

The process of model fitting is decomposed into three steps [76]:

- Points sampling: build a set of input and output values and divide it in training and validation set
- Optimization: Use the training set of points to fit the unknown parameters
- Model evaluation: Use the testing set of points to fit the unknown parameters

The three following sub-sections provide general knowledge about these three steps and present their applications during the camera calibration process.

2.2.1 Sampling

This step focuses on building a set of input data points and get their function value. This set is then divided into a training set and a testing set. Because calibration is a non linear regression problems, points must be sampled in the entire design space, including in regions of complex behaviors to correctly capture the non-linearity of the model [77].

In the specific application of camera calibration, the set of points is made of 3D world points, and their function values are the 2D pixel coordinates of their projection of the image. Building the mapping 3D to 2D for a pinhole camera requires solving 11 unknowns, as can be seen from equations in the previous section. Indeed, the homography matrix that relates a point (X,Y,Z) to its pixel component (u,v) is a 3*4 matrix and is defined within a scale factor so it has 11 unknowns. A point in 3D and its 2D location provides two equations (one for each image component). Thus, a minimum of six training points must be generated to calibrate a camera. These points should not be colinear in the camera frame; otherwise, they provide the same information, i.e., the same set of equations.

The main take-away of this step for camera calibration purpose is summarized in the following observation:

Observation: The camera calibration process requires the measurements of at least six (3D,2D) points.

2.2.2 Training

The key points obtained during the sampling are used as an input of an optimization method which returns an estimate for the camera parameters. For the camera calibration purpose, the literature presents two optimization approaches: non-parametric regression and parametric regression.

Non-parametric regression:

Non-parametric regression estimates the function without assuming any parametric form, and the model can result in higher performance compared to parametric methods [78].

These methods present a main advantage compared to parametric method. They do not rely on a pre-determined shape for the function to model. In the specific case of camera calibration, it would enable modeling the camera while including impacts dues to manufacturing error, lens blurring, environment impact such as potential degradation due to weather or vandalism.

Non-parametric methods usually requires a large number of data points. In [90], a Support Vector Machine (SVM) technique is used to calibrate the camera and enables an accurate and fast calibration. The method uses a linear kernel function to model pinhole cameras and a radial basis function to model lens cameras. Because the kernel function type depends on the camera model, the SVM approach is not easily scalable to various cameras. In addition to this drawback, the method requires the estimation of the first guess for some scaling factors, which requires manual processing. When considering the large number of security cameras already installed, the calibration method should require as little human knowledge as possible. A neural network was used in [79] to build a 3D-2D relation independent of the camera geometry. In [80], a neural network is implemented to avoid using a predetermined model and account for the manufacturing error. However, training a neural network requires a large number of sample points, which is time-consuming to collect. The parametric methods have shown good accuracy and require a smaller number of points to sample. Thus, this work focuses on parametric regression methods, and the neural network method could be future work.

Parametric regression:

As derived in section I of this chapter, the explicit shape of the relation between 3D and 2D points is given by the following set of equations:

$$s*\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_1 \\ m_{21} & m_{22} & m_{23} & t_2 \\ m_{31} & m_{32} & m_{33} & t_3 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$u_{distortion} = f(u,\ v,\ C_x,\ C_y, f_x,\ f_y,\ k_{1,}k_{2,}p_{1,})$$

$$v_{distortion} = g(u,\ v,\ C_x,\ C_y, f_x,\ f_y,\ k_{1,}k_{2,}p_{1,})$$

where f and g are non-linear functions.

The literature presents many methods to solve the unknown parameters, and the three main ones are described below. The fastest one assumes the linear camera model and ignores lens distortion. In that case, a one-step optimization can solve for the unknown parameters. This method fits the unknown parameters by minimizing the least square error for an overdetermined problem [72]. This method is also used as a first step of more complete calibration methods.

The calibration equations for the linear model can be written as follow, where A=(a_{ij})

is called the projection matrix:

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} * \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

The parameter “s” can be any positive value. Indeed “s” is the distance along the ray that joins the camera center to the actual 3D point. All points on this ray have the same projection on the camera sensor, leading to the same pixel.

The above system of equations can be expended leading to the following equation:

$$s * u = a_{11} * X_w + a_{12} * Y_w + a_{13} * Z_w + \ a_{14}$$

$$s = a_{31} * X_w + a_{32} * Y_w + a_{33} * Z_w + \ a_{34}$$

$$a_{31} * X_w * u + a_{32} * Y_w * u + a_{33} * Z_w * u + \ a_{34} * u = a_{11} * X_w + a_{12} * Y_w + a_{13} * Z_w + \ a_{14}$$

$$a_{11} * X_w + a_{12} * Y_w + a_{13} * Z_w + \ a_{14} - a_{31} * X_w * u - a_{32} * Y_w * u - a_{33} * Z_w * u - \ a_{34} * u = 0$$

The same can be done for the coordinates v, and this leads to a second equation:

$$a_{21} * X_w + a_{22} * Y_w + a_{23} * Z_w + \ a_{24} - a_{31} * X_w * v - a_{32} * Y_w * v - a_{33} * Z_w * v - \ a_{34} * v = 0$$

Equations (a) and (b) can be used to solve for the unknown parameters. There are 12 unknown parameters, so it is necessary to have at least 12 equations to solve for these unknown parameters. Consequently, using equations (a) and (b) for at least 6 points, it is possible to solve for the parameters. In order to solve for these parameters, a matrix L is defined, such as for p in [0, N-1] where N is the number of points, the row of L are defined as follow:

$$L^T{}_{2p} = [X_p,\ Y_p,\ Z_p,\ 1, 0, 0, 0, 0, -X_p u_p, -Y_p u_p, -\ Z_p u_p, -u_p]$$

$$L^T{}_{2p+1} = [, 0, 0, 0, 0, X_p,\ Y_p,\ Z_p,\ 1\ , -X_p v_p, -Y_p v_p, -\ Z_p v_p, -v_p]$$

By writing a=$[a11, a12, a13, .., a33, a34]^T$, the parameters are a non-trivial solution to L*a=0, which can be obtained using least square or SVD decomposition. The parameters obtained from these steps are not directly the camera parameters but a combination of them. For instance, a_{11}=f_x*m_{11}+C_x*m_{31}. Calibrating the camera requires to estimate the camera parameters, thus, it requires to find f_x, f_y, m_{ij}, C_x, C_y, t_i, from the a_{ij} coefficients. This step is not trivial and a method to estimate the camera parameters from the a_{ij} coefficient is presented in Appendix B.

The method does not account for the lens distortion so far. The two methods introduced below do include lens distortion. A complete method is presented in [12] and takes the lens distortion into account. The key points obtained during the sampling are used as an input of a non-linear optimization method that aims at minimizing the mean square error $F = \sum_{i=1}^{N} (U_i - u_i)^2 + (V_i - v_i)^2$ where U_i and V_i are the actual values of the 2D key points and u_i and v_i are the model values that are functions of the unknown parameters and the 3D key points. By minimizing this function, an estimate of the unknown parameters is obtained and used to build the mapping that relates 3D to 2D points.

This method is more accurate than the first method because it includes the lens distortion into the model. However, this approach requires good initial values for some parameters to obtain a good model.

A third method, called the two-step method, enables to overcome this limit [81]. This approach is a combination of the two previous methods. It first uses a linear optimization technique to get some of the parameters. For instance, authors in [12] use the linear optimization method described first in this section. The camera parameters are then estimated from the estimated a_{ij} coefficients. Appendix B provides explanation about this step. Then the estimated camera parameters are used as first guesses for the nonlinear optimization. In the literature review about camera calibration, several methods are used for the nonlinear optimization process. This step, is also called ”Bundle adjustment” in computer vision or photogrammetry [82]. The bundle adjustment technique was developed to refine the key point coordinates measurements and the resulting camera parameters. The technique uses nonlinear optimization to minimize a cost function whose arguments are the key points and the camera parameters estimated in the previous step [83]. In calibration, the key points coordinates are fixed, and the non-linear optimisation minimize the cost function (reprojection error) by varying the camera parameters and keeping the key point coordinates constant. The non-linear calibration step is a specific case of the Bundle adjustment.

The non-linear optimisation step is generally a second order line search method: the search is conducted along successive lines that satisfy criterion based on the gradient and hessian values [83]. Commonly used method for this step use Netwon’s method. For the later, the condition between gradient, hessian and line search direction is the following, where H is the hessian, δ_f the gradient, p the line search direction, x the vector of parameters to be varied, and k is used to represents the iteration number [84].

$$H(x_k) * p_k = -\delta_f(x_k)$$

In [12], a variant of Netwon’s method, which is the Levenberg Marquardt optimization algorithm, is used for the non linear optimization. In [85], a genetic algorithm is implemented to obtain the calibration parameters. In [86], the authors use linear optimization of several functions, each of them corresponds to a function of a unique variable, which is a distortion parameter. The two-step method has shown good accuracy in literature. It is scalable to a large range of cameras as it does not rely on a pinhole model assumption and does not require a manual first estimate of the parameters. Thus, implementing the two-step calibration approach enables the calibration process to be automated and applied to pinhole and lens cameras. It seems that this method does not present major obstacles when working with hard-to-reach or remote cameras, as soon as a set of training 3D and 2D points is available. Because this method does not require manual tuning, it is scalable to the many installed cameras. Thus, this method is adapted for the calibration problem tackled by this thesis. This leads to the following claim.

> Observation: Implementing the two-steps calibration approach enables the calibration process to be applied to a large range of cameras because it is an automated method and does not rely on simple camera model assumptions or manual tuning.

Figure 2.17 summarizes the optimization steps for the camera calibration.

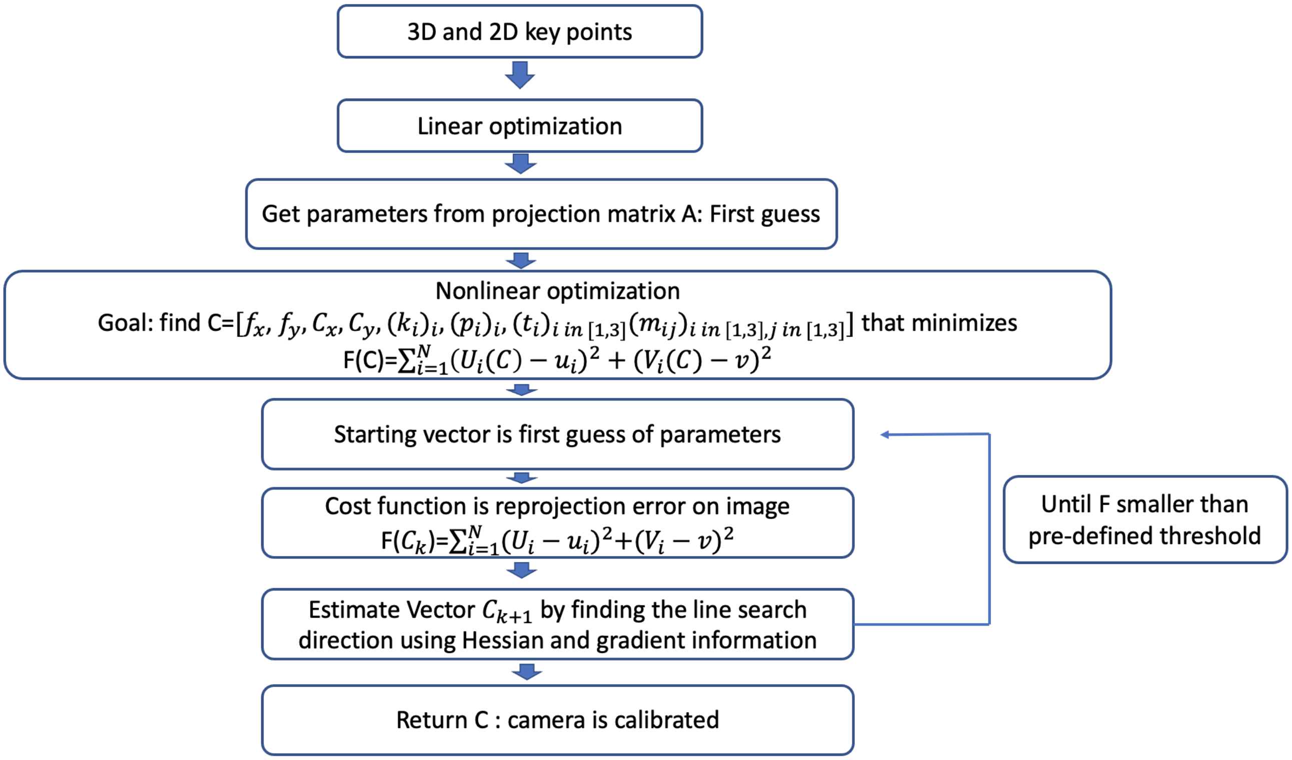

3D and 2D key points
Linear optimization
Get parameters from projection matrix A: First guess
Nonlinear optimization
Goal: find C=$[f_x, f_y, C_x, C_y, (k_i)_i, (p_i)_i, (t_i)_{i\ in\ [1,3]}(m_{ij})_{i\ in\ [1,3], j\ in\ [1,3]}]$ that minimizes
F(C)=$\sum_{i=1}^{N}(U_i(C) - u_i)^2 + (V_i(C) - v)^2$
Starting vector is first guess of parameters
Cost function is reprojection error on image
F(C_k)=$\sum_{i=1}^{N}(U_i - u_i)^2+(V_i - v)^2$
Estimate Vector C_{k+1} by finding the line search direction using Hessian and gradient information
Return C : camera is calibrated
Until F smaller than pre-defined threshold

Figure 2.17: Optimization process for the calibration

Once the model for the function has been estimated through the optimization process using the training set of points, the next step focuses on evaluating the accuracy of the camera calibration, i.e., the relation that maps 3D coordinates to 2D coordinates [87] [88].

2.2.3 Evaluation

This section introduces metrics used in calibration. It also presents an overview of common metrics used to evaluate accuracy of a regression model and their advantages and drawbacks for the camera calibration purpose. Finally, it lists the chosen metrics used in this thesis.

There is no systematic way to evaluate the camera calibration accuracy. For instance, for planar pattern-based calibration, Matlab offers to plot the relative locations of the camera and the calibration patterns to discover obvious errors [89]. This method can be applied to object-based calibration only. The authors in [90] chose a baseline calibration method as ground truth and compared the focal length and principal point location obtained through new calibration methods to the values obtained with the baseline method. In [91], the camera calibration is evaluated by measuring the root mean square error (RMSE) between image points obtained through the model and image points measured manually. In order to find an evaluation method suitable to the calibration problem tackled by this thesis, a review of functions of interest for evaluation of calibration accuracy and a review of regression evaluation methods was completed and is summarized below.

The accuracy of a regression model is measured by comparing the model value and the real function value at some points.

For camera calibration, the evaluation approach can be divided in two groups listed below [92].

- The first group measures the difference between the projection of 3D points obtained through the model fitted with the regression and the actual 2D coordinates on the distorted images.

- The second group compares the discrepancy between the 3D position estimated with the model and the actual 3D position. If the calibration function maps 2D points to 3D points, then it returns the ray where the 3D point is located as output. In that case, a method to calibrate the accuracy compares the distance between the actual

3D points and their predicted rays. If the calibration is from image to ground then the function maps 2D image coordinates to 2D real coordinates. The discrepancy between image points projected on the ground using the model and actual ground points enables to evaluate the calibration method.

Both accuracy evaluation methods are used in this work. The choice of the method is made depending on the purpose of the experiment. As an example, an application of this thesis work, as well as the last experiment focus on speed measurement of vehicles. For this specific case, the goal is to get the actual ground location from images, so the mapping from the image to the ground is computed. Thus, the second method is used to evaluate the calibration. Other experiments focus on the improvement of the approach. In that case, the work focuses on comparing camera models estimated through different calibration approaches. In that case, the accuracy is measured from the difference between the actual image points and the ones obtained through the estimated camera model.

Once the system used for accuracy evaluation has been determined, an evaluation metric (cost function) is computed using the difference between model points and real points. To select adapted metrics for this work, parameters used for regression analysis were investigated. Several methods to measure the accuracy of a regression model are listed below [93]:

- The coefficient of determination R^2 : $R^2 = 1 - \frac{SSE}{SST}$ where $SST = \sum_1^N (y_i - a)^T (y_i - a)$, $SSE = \sum_1^N (y_i - m_i)^T (y_i - m_i)$, a is the average of the y_i values: $a = \frac{\sum_1^N y_i}{N}$, y_i is the value of the true function at point i and m_i is the value of the model at point i [89]. a, y_i, m_i are vectors of two coordinates. R^2 is larger than 0 and smaller than 1 and it is closest to 1 for the best match between the model and the real function. The coefficient of determination is a relative measure of fit and is mainly used to compare two models and not for evaluation of a unique model.

- The actual VS predicted plot [93]: A good match is when the plot is close to the function y=x. This error metric assumes that the actual values are perfectly accurate which is not a correct assumption in camera calibration.

- Residual versus predicted plot [94]: the residual is the error between the real function and its prediction at a given point: $e_i = norm(y_i - m_i)$. If the model that is fitted to the data is correct, then the error should have a random distribution such as in the example below. The function modelled in this thesis is a function from R^3 to R^2. The predicted value is consequently a vector. So, two plots could be drawn: one that plots the error versus the abscissa value and one versus the ordinate value. However, using residual versus predicted plot for functions that model 2D vectors is not common. An example of plot for a function whose output is in the real space is given in Figure 2.18. If the residual displays a pattern, then the model fits the data poorly [95]. The metric is not well adapted to functions whose output is not in R but in R^n instead.

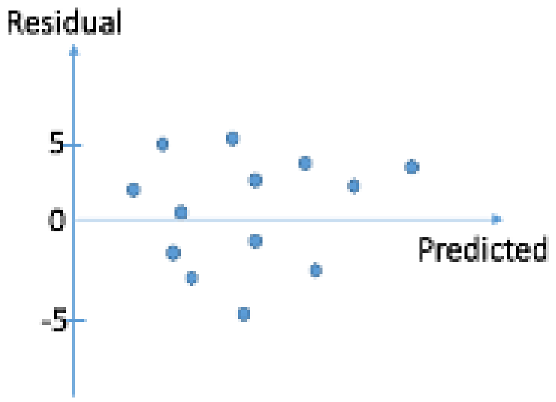


Figure 2.18: Light path from object to its image for a pinhole camera

- The root-mean-square error (RMSE): It is an absolute measure of fit, thus it is a commonly used metrics for calibration accuracy evaluation. The root mean square error has same unit as the output which allows to interpret easily the accuracy result. $RMSE = \sqrt{\frac{\sum_1^N (y_i - m_i)^T (y_i - m_i)}{N}}$

- Model fit error (MFE):[93] It is the error on training points. The model is good if the plot of the number of points with a given error versus the error value has a normal distribution.

- Model representation error (MRE) [93]: It is similar to the MFE but the error is measured using testing points instead of training points. The plot of the number of points for a given error versus the error value should have a normal distribution for a good fit.

These two last metrics prove that a model can be evaluated using training points, especially if testing points are not available.

The following observations summarize the Evaluation subsection,

Observation: Depending on the calibration purpose, two approaches are used in the literature to evaluate the calibration. The first one measures the discrepancy between actual image points and estimated projection of 3D points, the second ones compares a 3D points with the estimated ray.

Observation: The RMSE (or the average error) provides an absolute measure of fit and can be easily connected to the calibration application, thus it is an adapted metrics for this thesis work.

2.2.4 Take-away of the regression process for cameras calibration

The analysis of the regression steps for camera calibration purposes lead to four main observations that drive choices made while defining this thesis' hypotheses and while designing the experiments:

- Implementing the two-step calibration method for the optimization step enables the calibration process to be applied to many cameras. Indeed, it can be used for pinhole cameras and cameras with distortion without changes in the optimization process. Also, the two-step method does not require manual tuning of hyperparameters or a manual first-guess estimate. Thus, it is an automated method which is an essential factor when considering the large number of already installed cameras.

- The regression error can be measured through two different ways: by comparing 3D points and rays or by comparing image points and estimated projections. The choice of the method depends on the calibration application.

- RMSE and average error are adapted measures of fit for camera calibration. The error metric can be computed on training points (MFE) or on testing points (MRE).

- The optimization step relies on the availability of at least six points whose 3D and projected 2D locations are known. And the larger the number of points is, the higher the accuracy should be.

This last observation is the main challenge that must be overcome in this thesis. The following section provides a benchmark of existing methods that enable the collection of information about points in the world space and their corresponding images through the camera.

2.3 Benchmark of calibration methods

In this section, the conventional calibration methods will be introduced. It will be explained that these methods are not suitable for the calibration of hard-to-reach cameras.

2.3.1 Autocollimation method

This set of methods is used for the laboratory calibration of space cameras before the satellite launch. It is presented shortly since these methods are not used anymore for other types of cameras due to their tedious process. The approach relies on an expensive optical apparatus made of a collimator, a grid, and a bench to translate the collimator and the camera.

The collimator sends light to the grid, also called a filter. The actual grid shape and its image are used to determine camera parameters. This calibration method requires a large laboratory space or a large volume with know localization information [96]. It is a time-consuming process that can take a day of work for skilled technicians [96]. Authors in [20] designed an optical tool that replaces the bench to save room, but the tool requires easy access to the camera. Thus this set of methods is not adapted for hard-to-reach cameras.

2.3.2 Object-based calibration

These methods rely on the observation of points of interest on a calibration object, and the correlation of their pixel coordinates with their 3D coordinates. The calibration object's geometry is known, which enables the determination of the 3D coordinates of the points of interest [97]. According to the dimension of the calibration object, the method is classified as 3D object-based calibration, 2D object-based calibration, or 1D-object-based calibration.

- 3D object-based calibration: The calibration object is made of two or three planes orthogonal to each other [98]. The theory behind using a 3D calibration object is not complex, but the process is expensive to implement. Indeed, accurate 3D calibration objects are difficult to produce and maintain. Besides this limit, the object must cover most of the camera field of view to obtain a small calibration error which is not convenient when a camera with a large field-of-view is calibrated [99]. This calibration approach is expensive to set up, and that is why two other object-based calibration methods were designed [100].

- 2D object-based calibration: Zhang developed a method that uses a planar pattern that is moved in front of the camera at different orientations to get the camera's parameter and calibrate the camera [100]. It is the most common calibration technique used nowadays. An implementation of the method can be found on OpenCV.

It requires about 50 images of a known pattern. Several calibration patterns have been designed in the literature review, and the OpenCV method recommends using a chessboard-like pattern. The chessboard is placed at different locations and orientations in front of the camera so that it covers most of the field-of-view. The 3D world coordinates of some feature points are determined using the known geometry of the object. The OpenCV method detects and computes the corresponding 2D pixel locations in the image. These key points are then used as input of the black-box function to map 3D to 2D. However, this method is tedious to set-up because the calibration object must be manually rotated and translated in many positions. For large field-of-views, many calibration patterns can be located in the volume seen by the camera [101], but it requires a long and manual process. Authors in [102] demonstrates that their small calibration board must cover the entire field-of-view to ensure an accurate calibration, which is a very tedious process. This very manual process is not adapted for the calibration of cameras located in complex environments, such as the one for Glacier monitoring. Also, when considering the security camera application, more than 60 million cameras would need to be calibrated. This very manual process is not adapted for this purpose.

- 1D object-based calibration: An alternative solution for object-based calibration relies on using a 1D object, such as a wand that is moved in front of the camera. The calibration object is made of visible key points (markers) aligned on the wand [98]. These marker's locations on the wand are known [103]. This calibration object is moved in the field-of-view to collect the required 3D and 2D points. For instance, in [98], the wand movement is a rotation around a fixed point. Video sequences of the moving wand are acquired to obtain several observations of the 1D object and then are processed to calibrate the camera.

The three methods mentioned above require the use of an object that can cover the camera field of view. Indeed if the object covers only part of the field of view, then the calibration results in poor accuracy [104]. Since these cameras are hard to reach, their field-of-view covers a large area. Hence the object-based calibration is not adapted for a camera that is hard to reach. In addition to this constraint, two of the object-based approaches require the motion of the calibration object, which is complicated to implement when working with cameras that are hard to access. Because of the need of covering the entire field-of-view and the manual process required for object-based calibration methodology, these methods are not adapted for hard-to-reach cameras. Indeed, they do not solve the issue of remote camera calibration. For instance, they do not enable the calibration of cameras located on offshore platforms, since they require setting up calibration objects in their field-of-view, which is not easily accessible by a human. Finally, these processes are manual and require tedious human intervention, so they are not scalable to a large number of cameras. However, North American counts more than 60 million hard-to-reach cameras, whose calibration would enable great benefits, as explained earlier in this chapter.

Thus, the following gap can be formulated:

> Gap: The existing object-based calibration approaches are not adapted to large FOV and to hard-to-reach cameras, because they require a tedious and manual process

2.3.3 Self-calibration

Self-calibration methods do not use a calibration object. They rely on correspondences between image points, or on correspondence between real-world object characteristics and image features to compute the camera parameters.

There are a large variety of self-calibration techniques. First, the original one and the state-of-the-art ones are presented. Then, the main drawbacks of self-calibration are introduced.

The self-calibration principle relies on a virtual object called absolute conic. Let us consider a particular conic in the plane at infinity that is invariant under rigid motion and change of scale. Its image is the same when the camera is moving. This conic is called absolute conic and is used as a virtual calibration object that can be found in all scenes. Thus several views of this object are available and are used for the calibration [105]. The first self-calibration method developed uses the absolute conic principle and is described in [106]. This method requires camera motion, and consequently, it does not apply to static cameras, which is a key portion of security cameras. The self-calibration methods that are based on the absolute conic principle rely on Kruppa's equations [107] which have unknown scale factors that are difficult to solve, thus these factors are not estimated accurately. Also, some of these techniques require a first estimate for some parameters (intrinsic ones), which implies errors in the calibration process [105]. Consequently, these calibration techniques are not robust and tend to perform poorly [108]. In addition to lacking accuracy, some environment geometry is not conducive to self-calibration using the absolute conic. Indeed, a large number of well-distributed object points or highly convergent images are required to implement self-calibration techniques [109]. Consequently, these self-calibration methods apply only in a restricted type of environment. The state-of-the-art self-calibration method uses scene features called vanishing points [110]. Vanishing points are intersections in the image plane of parallel lines in the world frame. These lines are called vanishing lines [111] and rely on the following observation: parallel lines in the 3D frame usually intersect in the image frame. Indeed, the transformation between a 2D surface in the world and its image captured by a camera is described by the projective geometry. It does not preserve distances, angles, or parallelism. Projective geometry only preserves straight lines. A camera can be self-calibrated using three (or two for an approximated calibration) mutually orthogonal vanishing points [112]. Figure 2.19 illustrates vanishing lines in three orthogonal directions. In the world frame, the yellow lines are parallel to each other. It is the same for the purple lines and the green ones. On the image, they are not necessarily parallel, and they intersect at vanishing points.

Authors in [113] have demonstrated that this technique enables camera calibration from

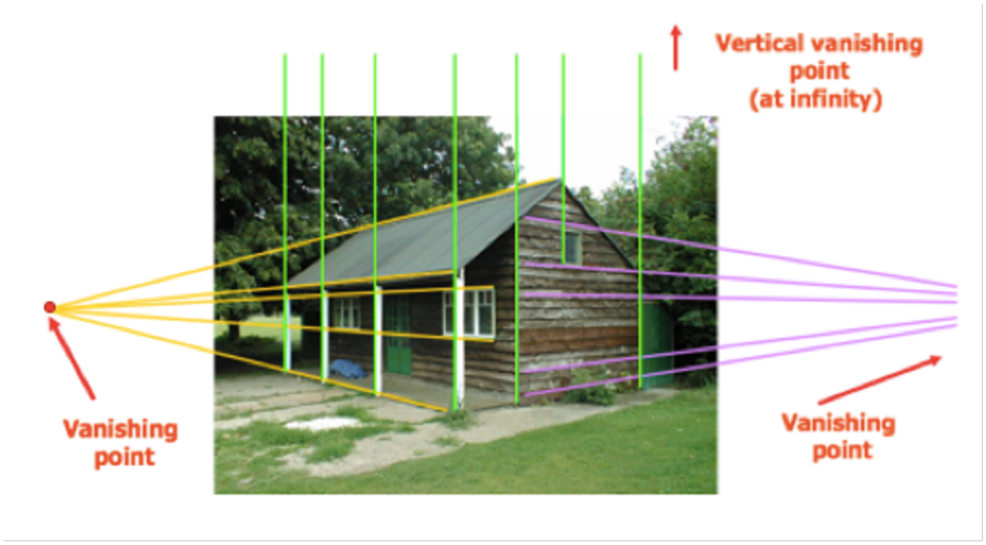


Figure 2.19: Vanishing lines and vanishing points

observing a walking human or a moving car to extract the vanishing points. However, it assumes a constant speed of the tracked objects and requires orthogonal lines marked on the ground. These requirements restrict the type of environments where the camera can be calibrated.

The main drawbacks of the calibration through vanishing points are listed below:

- Vanishing lines intersect at several points. It is then necessary to use a voting strategy to select a vanishing point. Consequently, the vanishing point location is inaccurate [114] [115]

- If two of the vanishing lines intersect at infinite vanishing points, then the internal parameters can not be obtained [110] [115]

- It requires at least two orthogonal vanishing lines [112] [116], and so it only applies to certain kind of scene [115]

- It often requires additional information about the scene, and sometimes it requires the use of an object set in the field-of-view [115]

Many techniques derived from the ones presented above can be found in the literature.

Self-calibration techniques are less accurate than object-based calibration techniques [106]. Also, they often require human processing [117], for instance, to identify a plane surface on the image and provide the information to the algorithm. Thus, they lack automation. Most importantly, they require specific view geometry, as for example, the presence of three orthogonal vanishing lines. Cameras in non-urban environments (offshore platform cameras, forest cameras to limit poaching or pollution, glacier monitoring cameras etc.) usually do not satisfy these scene constraints. Active vision-based self-calibration does not require those view constraints but the motion of the camera [105]. Thus, they are not adapted to static camera, such as intersection cameras that can be used for speed measurement application.

To summarize, the self-calibration techniques rely on constraints imposed on the camera motion or the scene geometry, which make them not adapted to calibrate cameras in a large variety of environments [105].

Gap: The self-calibration methods require restricted conditions to work correctly and are thus not scalable to a large type of hard-to-reach cameras

2.3.4 Summary

Figure 2.20 summarize the usual process used for model fitting problems, and their corresponding solution for the camera calibration purpose. The green check mark represents steps that are adapted for the calibration of hard-to-reach cameras, while the red cross indicates that the existing calibration methods are not yet adapted to the calibration of hard-to-reach cameras.

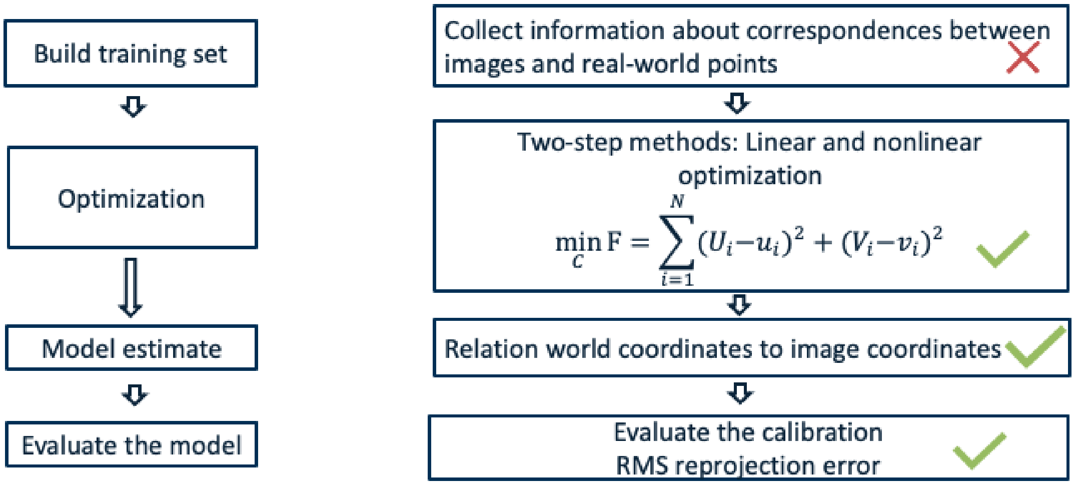

Build training set
Optimization
Model estimate
Evaluate the model
Collect information about correspondences between images and real-world points
Two-step methods: Linear and nonlinear optimization
$\min_{C} F = \sum_{i=1}^{N} (U_i - u_i)^2 + (V_i - v_i)^2$
Relation world coordinates to image coordinates
Evaluate the calibration RMS reprojection error

Figure 2.20: Model Fitting Steps and Calibration Steps

2.4 Overarching Research Question

This first chapter had provided information about the increasing number of cameras and their use in numerous fields. In particular, it stated the importance and the benefits of hard-to-reach camera calibration for various use cases and applications. One such use case is that of environment monitoring with static platforms. An other use case is that of object or obstacle detection and location identification using security cameras. This current chapter touched on the lack of calibration methods for cameras that are hard to reach. Existing calibration techniques are not adapted to the collection of the required information and correspondences between images and real-world points. The object-based technique being not applicable when dealing with a large field of view or complex camera access. The diversity of field-of-view environments is vast (forest, harbor, offshore platform, cities, etc.), making the self-calibration method not suitable. Thus, the following overarching research question is formulated:

> Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras?

Hard-to-reach cameras have a large field of views, making traditional object-based calibration approaches not suitable for gathering the key points required for the regression. Also, self-calibration methods can only be applied in a restricted type of scene. Thus, a way to collect 3D and matching 2D points for hard-to-reach cameras must be found. The following Chapter presents the proposed solution to calibrate hard-to-reach cameras and the challenges this method must overcome to be successful. It also defines criteria for successful calibration.

CHAPTER 3
PROPOSED SOLUTION AND CHALLENGES TO OVERCOME

A benchmark of the calibration methods was presented in the previous chapter. The existing methods are not adapted to calibrate already installed cameras, especially when facing a non-urban environment where strict geometric constraints are rare. Object-based calibration methods require a tedious process using a calibration object in the field of view of the camera. It is not adapted when camera access is difficult. Self-calibration techniques do not require a calibration object. However, their implementations either require human knowledge about the scene or require strict scene geometric constraints, that are rarely satisfied, such as three orthogonal lines. Thus, they cannot be scaled to the use of cameras located in a large variety of environments.

As explained in the Regression section, the fitting methods found in the literature are adapted to the calibration of hard-to-reach cameras if a set of 3D and 2D key points is available. However, existing calibration methods do not guaranty the availability of these key points. This lead to the overarching research question:

> Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras?

This Chapter proposes a solution to the collection problem adapted to hard-to-reach cameras. It also introduces the challenges that must be overcome to enable a successful calibration when using the proposed solution. Finally, it defines criteria for a successful calibration based on applications presented in the first chapter.

3.1 Moving target and challenges to overcome

The proposed method must satisfy the following requirements:

- Be adapted to large fields-of-view since these cameras cannot be accessed easily (which prevents the use of object-based calibration techniques)

- Be scalable to various environments (which is not feasible using self-calibration techniques that require strict assumptions about the scene)

- Be automated to enable the calibration of the large number of already installed cameras

- Be able to correct for the large non-linear distortion, that is frequently present with security cameras. This means the method must allow the coverage of the entire field of view

- Enable the collection of a large number of key points in the field of view to enable the use of non-parametric methods for calibration (in a future work)

A solution to these requirements can be provided with a system that can move easily in any 3D space, provides its location information, and be detected in an automated way on images. Using a GPS and a barometer allows the collection of 3D coordinates. The latitudes, the longitudes, and the altitudes obtained through these sensors can be transformed easily to world coordinates in a North, East, Up frame in the area of the camera frame, and in this way used easily for calibration purposes. It is now necessary to define a system that could carry these sensors and move them in the 3D space. The camera films this system, which is detected on each frame of interest. The resulting image localization is used as 2D points required for the calibration.

When Google Street View mapped the streets in the world, several options were investigated to capture images and their locations in various environments. Among them were pedestrians, cars, snowmobiles, and bicycles [118]. The same options are investigated to see which system can be used to sample points in a large variety of 3D spaces. Drones are added to the possible systems that can travel easily in a 3D space and provide their location data.

The ability of several options to satisfy the mission requirements is assessed using Table 3.1. Three systems that can carry localization sensors and move easily in front of the camera are identified. These three systems are the following: Pedestrian, Vehicle or Drone.

The calibration approach should be scalable to a large diversity of environments where remote cameras can be located. It should also allow the sampling of 3D points everywhere in the field of view (FOV) so that regions of complex behaviors can be accounted for when fitting the camera's unknown parameters. Thus, each system is evaluated compared to its ability to operate in various environments: Urban, sea, mountain, sky, and space environments, and its ability to cover the FOV, i.e., to enable large horizontal and vertical displacements (outside of the ground area).

Table 3.1: Evaluation of three systems for the calibration of hard-to-reach cameras

System that carries localization sensors	Urban environment	Sea	Mountain	Sky	Space	Can cover full FOV
Pedestrian	yes	no	yes	no	no	no - restricted to ground displacement
Vehicle	yes	yes (boat)	Yes (snowmobile, but large vibrations)	no	no	restricted to ground displacement
Drone	yes	yes	yes	Yes	Yes	Yes

Drones are the only system than satisfy all criteria. Indeed, a drone combined with a GPS and a barometer enables to sample points in a large variety of 3D spaces. These sampled points and their images through the camera enables to obtain the key points required for the calibration. In addition to satisfying these requirements, the drone can fly in an autonomous way which reduces the amount of manual processing and allows the approach to be scalable to a large number of cameras. These observations lead to the following claim, which is illustrated in Figure 3.1:

Claim: Flying a drone in the camera FOV enables the automated collection of the 3D and 2D key points required for the calibration of hard-to-reach cameras.

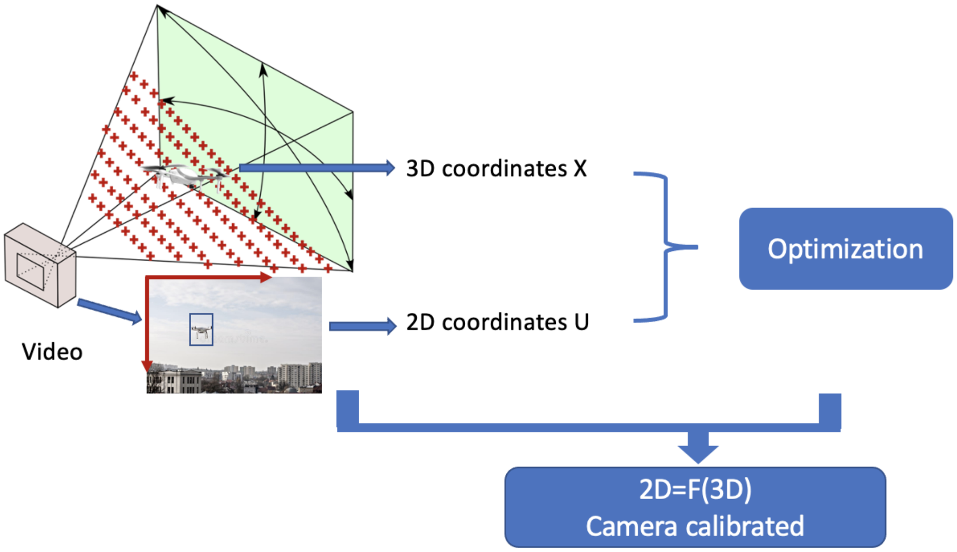


Figure 3.1: Use a drone as moving target for the collection of 3D and 2D key points

Traditional object-based calibration methods use a target whose geometry is well known, and thus the 3D localization of the sampled points can be obtained accurately. These methods are not adapted to large fields of view, and so to hard-to-reach cameras, because they require a tedious manual process. The proposed solution in this thesis relies on using a drone as a moving target. The localization of the moving agent is not known accurately. For example, random error is introduced by small drone movements at the sampled points, which might bias the 2D localization. If a GPS is used as a 3D localization sensor, random noise is introduced by events such as a signal blockage. Thus, the target localization is less certain when using a moving agent for the calibration. The proposed calibration approach must overcome the challenges raised by uncertainty in sensors' measurements. The calibration must be accurate despite the localization uncertainty, and image detection uncertainty. Thus, this work must test the drone-based calibration methodology and proposed improvements.

Before introducing and testing potential solutions to the calibration problem for hard-to-reach cameras, what is considered as ”successful calibration” or ”accurate calibration” is defined in the next section.

3.2 Threshold for calibration error

The camera calibration problem is an optimization problem whose goal is to obtain the best estimate for the camera parameters and, consequently, the most accurate relation between 3D points and 2D points. The function fit is not perfect, and thresholds for calibration errors must be defined to distinguish between successful calibration and poor calibration.

The evaluation of the calibration accuracy depends on its application. Obstacle localization and speed measurements are two key applications presented in Chapter 1. Thus, the definition of metrics to evaluate the calibration success are based on these applications.

Once the application of camera calibration is selected, two questions need answers:

- What is the maximum error acceptable to classify a vehicle speed measurement as accurate?
- What is the relation between the bounds defined above and the reprojection error bounds?

The answer to the first question is provided with the following thought process. The typical vehicle speed limit in urban environments is 50km/h, and the typical speed limit on highways is equal to 130km/h. An error of 1 percent when measuring these speeds corresponds to an error of 0.5km/h and 1.3 km/h, respectively, so an average of 0.9km/h, which is about 1km/h. So the maximum accepted error when measuring vehicle speed is taken equal to 1km/h=0.3m/s.

The answer to the second question is explained below. Figure 3.2 illustrates the true distance between the vehicle position at time t (point A) and its position at time $t + dt$

(point B), represented with the actual displacement vector u_a. It illustrates the measurement errors when determining the vehicle positions at time t and $t + dt$, called ue_1 and ue_2. The measured speed v_m is related to the actual speed v_a and the speed measurement error v_e as described below:

$$\overrightarrow{v_m} = \frac{\overrightarrow{u_m}}{dt} = \frac{\overrightarrow{u_{e1}}}{dt} + \frac{\overrightarrow{u_a}}{dt} + \frac{\overrightarrow{u_{e_2}}}{dt}$$

$$\overrightarrow{v_a} = \frac{\overrightarrow{u_a}}{dt}$$

$$\overrightarrow{v_e} = \frac{\overrightarrow{u_{e1}}}{dt} + \frac{\overrightarrow{u_{e2}}}{dt}$$

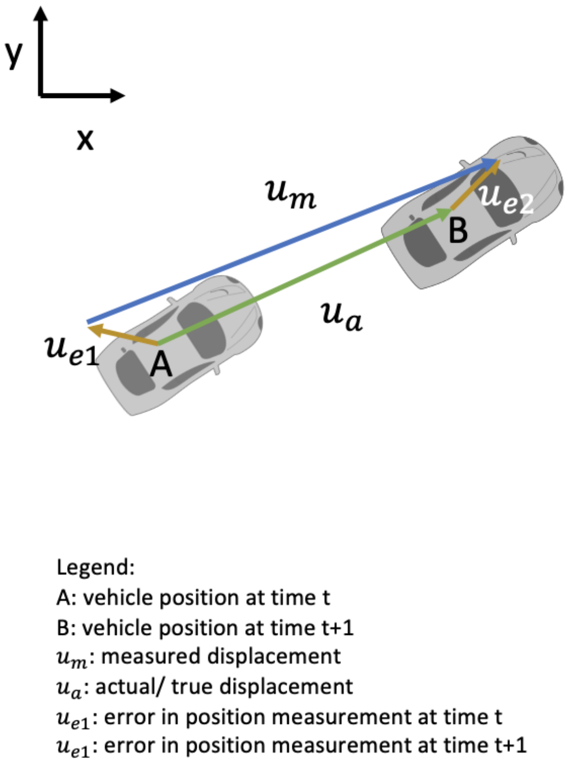

y
x
u_m
u_{e2}
B
u_a
u_{e1}
A
Legend:
A: vehicle position at time t
B: vehicle position at time t+1
u_m: measured displacement
u_a: actual/ true displacement
u_{e1}: error in position measurement at time t
u_{e1}: error in position measurement at time t+1

Figure 3.2: From speed measurement error to position measurement error

Since $\overrightarrow{v_e} = \frac{\overrightarrow{u_{e1}}}{dt} + \frac{\overrightarrow{u_{e2}}}{dt}$, the following equality holds

$$\|v_e\| <= \frac{\|u_{e1}\|}{dt} + \frac{\|u_{e2}\|}{dt}$$

Let us call u_{max} the maximum position measurement error, then the following equation holds:

$$\|v_e\| <= \frac{2 * u_{max}}{dt}$$

Common cars GPS update their position information every second [119]. Based on this observation, dt is taken equal to 1 second. If u_{max} is equal to 0.15 meter, then the following equation holds:

$$\|v_e\| <= 0.3$$

So taking a position measurement error on the ground of 0.15 meter allows for a maximum error speed of 1km/h=0.3m/s. Also, 15cm error would correspond to a great information for the obstacle localization application. This lead to the following claim

Claim: The camera calibration is successful if the corresponding measurement error on the ground is smaller than 15cm.

Let us define the relation between this ground measurement and its corresponding image measurement. This will lead to the maximum accepted discrepancy between actual image points and estimated projection of 3D points.

Figure 3.3 represents, for two different ground locations, the actual point on the ground (the blue one) and the estimated position when the measurement error is 15cm (the red point). Figure 3.4 represents the corresponding images seen by the camera whose parameters are given by the first line in Table 3.2.

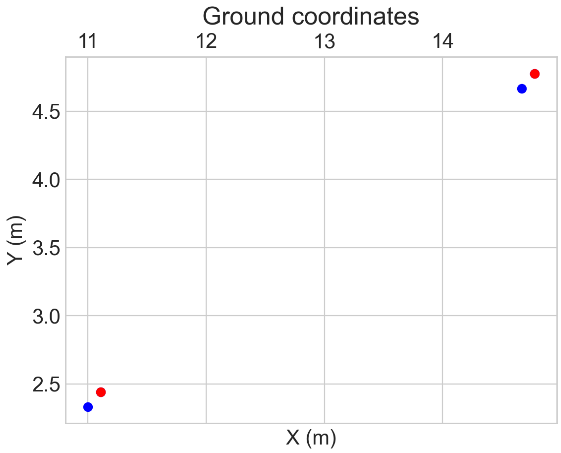


Figure 3.3: Ground measurement error equal to 15cm

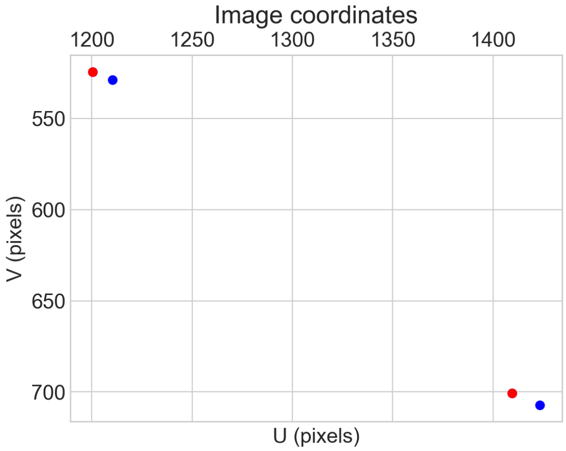


Figure 3.4: Corresponding image measurement error

Table 3.2: Parameters of different pinhole cameras

Camera id	f(mm)	w(pixels)	Height(m)	Orientation angle(83degree)	minimum accepted pixel error
1	13.3	3544	4.5	43	5.8
2	5.7	1860	5	67	0.9
3	7.5	3078	3.1	59	1.8
4	2.8	2153	3	83	1.8

An error of 15 centimeters (cm) on the ground is projected differently depending on the points' location. For a point close to the camera, the corresponding pixel error is large. It means that a large pixel error on the image does not transfer into a large measurement error on the ground. However, for points far from the camera, a small pixel error transfers in a large ground error. Figure 3.5 illustrates this observation. The blue cone represents the camera field of view, the red ray called ray_1, represents the actual ray where the point is located if they would be no image measurement error. The red ray called ray_2 represents the estimated ray, the one obtained after making an error called $Error_{image}$ on the figure. The error made on the image, translates to different magnitude depending on the object location compared to the camera. The farther the object is, the larger the resulting error is. Thus, the farther the object is, the worst is the impact of an image measurement error on the actual measurement.

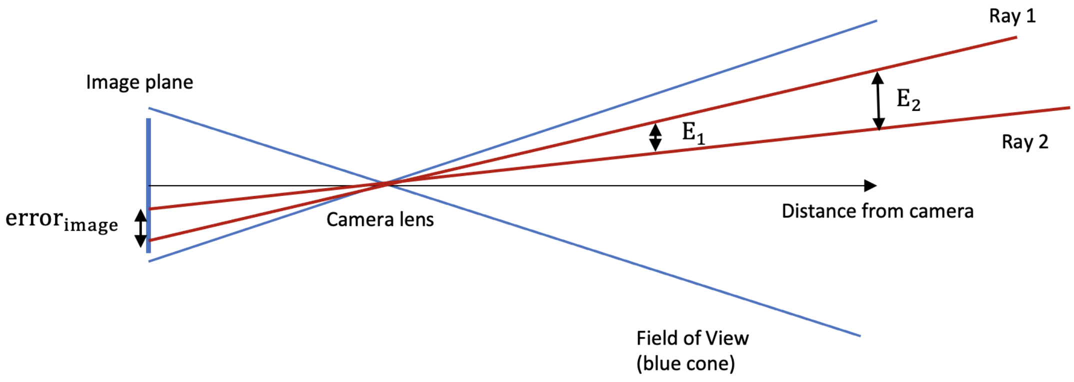


Figure 3.5: A given measurement error on the image impact badly the real world measurements when points are far from the camera

Thus, the error threshold is determined by measuring the pixel error corresponding to a ground error of 15cm for points far from the camera for different cameras. Table 3.2 provides these cameras' parameters and the error for the farthest point to the camera. This error is the hardest threshold to meet because it is the smaller one. The average of these smallest "accepted pixel errors" is 3.6 pixels. Thus the calibration of hard-to-reach cameras is considered accurate if the pixel measurement error is smaller than 3.6 pixels.

This leads to the following claim:

Claim: The camera calibration is successful if the corresponding measurement error on the image is smaller than 3.6 pixels.

In chapter 2, it was stated that RMS reprojection error is an adapted metric for the calibration problem because its unit is a pixel, the same as the image measurement unit. The average error is an adapted metric for camera calibration purposes, too, as it also uses pixel as a unit. On the contrary to average error (AE), RMSE gives more weight to large errors. RMSE better represents the regression success than the average error for applications where avoiding large errors is of significant importance. An upper bound for the RMSE is also an upper bound for the AE as showed with the equation below, where N is the number of points. This allows for a simple interpretation of the RMSE, which is a commonly used metric in calibration. If the RMSE is smaller than 3.6 pixels then the average reprojection error is smaller than 3.6 pixels and the calibration is accurate for the applications considered in this thesis.

$$AE <= RMSE <= sqrt(N) * AE$$

To following claim summarizes this section:

Claim: The calibration is evaluated as accurate if the corresponding re-projection error on the image is smaller than 3.6 pixels, or the corresponding re-projection error on the ground is smaller than 15cm.

The meaning of ”successful calibration” or ”accurate calibration” has been defined.

The proposed solution for the calibration of hard-to-reach cameras relies on the use of a drone as a moving calibration target. The uncertainty in 3D and 2D coordinates measurements impacts the calibration quality, and the technique must be tested to evaluate its ability to calibrate cameras. The research of this thesis focuses on the design, evaluation, improvement and validation of the proposed solution. To do so, the research plan is divided into four main parts, which corresponds to four separated chapters:

- Proof-Of-Concept: The calibration technique is tested for two simulated cameras. During this step, the 3D localization measurement is assumed accurate and the method is designed to allow for a good calibration despite 2D localization errors.
- Scalability to many cameras: During this step, the uncertainty due to the 3D localization sensor is included into the evaluation of the calibration technique. The calibration method is evaluated for thousands of different simulated cameras. A strategy is developed to understand the impact of path parameters on the calibration accuracy. This strategy is applied to the thousands of simulated cameras to provide recommendations for the target trajectory for a successful calibration.
- Hardware testing: The knowledge gained from the simulations is implemented in a real world test to complete the validation of the calibration technique and evaluate the smoothness of the real-world process.
- Optimization: This last part tests a strategy to optimize the target trajectory for a very efficient calibration technique.

Figure 3.6 illustrates the research plan followed in this thesis to design, evaluate, improve and validate the drone-based remote camera calibration technique.

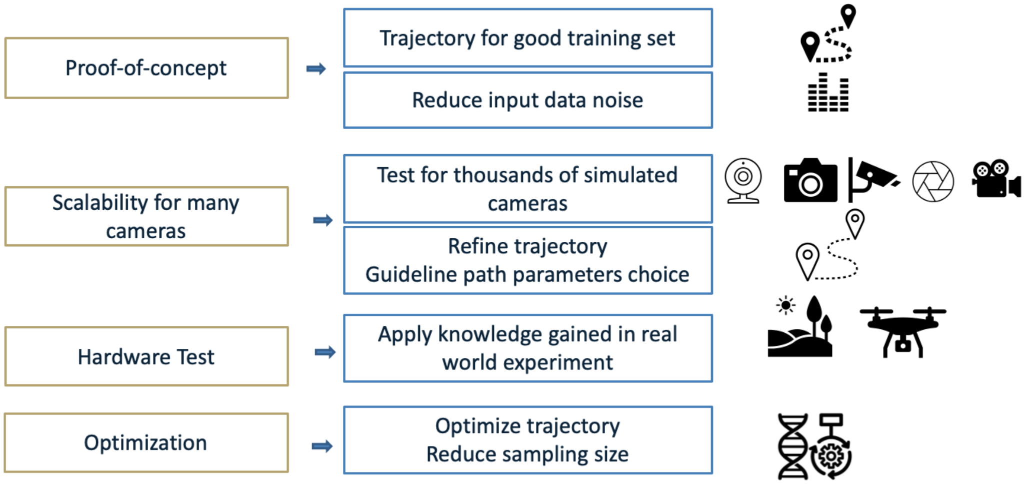

Proof-of-concept
Trajectory for good training set
Reduce input data noise
Scalability for many cameras
Test for thousands of simulated cameras
Refine trajectory
Guideline path parameters choice
Hardware Test
Apply knowledge gained in real world experiment
Optimization
Optimize trajectory
Reduce sampling size

Figure 3.6: Research Plan: proof-of-concept, scalability to many cameras, real testing, optimization

3.3 Summary

This section summarizes the main takeaway of this chapter. This thesis proposes a solution to the calibration problem for hard-to-reach cameras. The proposed approach relies on the use of a drone or a robot as a moving target to collect the 3D and 2D matching points required for the calibration.

The advantages of this approach, if successful, are listed below. First, it is adapted to the large FOV of hard-to-reach cameras, contrary to the object-based calibration method, which requires a tedious process. Secondly, it does not depends on specific scene geometry, contrary to self-calibration techniques, and is thus scalable to many environments. Thirdly, it is an automated approach for a process that would be otherwise tedious, especially when considering the large number of installed cameras. Finally, the approach enables the collection of a large number of key points located in the entire field of view and thus enables correcting complex non-linear behaviors such as the ones induced by the lens.

The target localization in the 3D space and on the image is subject to random noise, and the approach must be tested to evaluate its ability to calibrate cameras despite measurement uncertainties.

First, a proof-of-concept is demonstrated for two cameras using realistic simulations. The next Chapter presents the Research Questions and Hypotheses that must be answered and tested to evaluate the success of the calibration technique for these two cameras.

CHAPTER 4
STEP 1: PROOF-OF-CONCEPT

In a first step, it is assumed that the drone 3D localization can be obtained without error. Major improvements have been made recently for object localization and motion estimation. For instance, the combination of a Visual-Inertial Navigation system with SLAM methods enables accurate 3D tracking [120]. An alternative solution relies on the use of an external sensor, such as a single-chip millimeter-wave radar to obtain an accurate 3D position [121]. Thus, in a first time, the ability of the approach to enable a successful calibration is evaluated assuming perfect 3D localization information. In that scenario, image detection errors are the main source of error, and the calibration accuracy must be evaluated to ensure an accurate calibration despite errors in the localization of the target on the image.

Calibration is a regression process, and thus its quality is impacted by the input data quality [122][123]. The training data is made of the collected 3D and 2D key points. These key points are then input of the optimization for the calibration. The next section presents the requirements that a good quality training set must satisfy, and how they translate into requirements on the drone trajectory.

4.1 Training set design

Authors in [124] divide data quality into two groups: The first group is called the intrinsic group by authors in [124], and it is related to the accuracy and reliability of the data. For image processing, this group refers to measurement errors or labeling errors [125]. The second group is called the contextual group, and it is related to the completeness of the data, also called coverage of the data [126]. The training dataset must represent the diversity of items in the production dataset, so that the model can generalize well on unseen data [126].

For image processing, this group refers to an error during the design of the training sample [125].

Thus, the following claim and gap can be stated:

Claim: The calibration error is impacted by the sampling design choice and the sampling accuracy.

Gap 0: A good quality training set is required for a successful calibration. It must satisfy the coverage and accuracy requirements.

An analogy can be established between these two definitions of data quality and the regression error decomposition into bias, variance, and irreducible error (also called noise) σ:

$$error = bias^2 + variance + (\sigma)^2$$

The irreducible error is the error that can not be removed during the regression process. It is related to noise in the observation and measures the output error for the best possible model [127]. Hence, it is analogous to the first type of error introduced above, which is due to accuracy and reliability issue in the training data. The bias is taken as constant in this work. Indeed, the bias is a source of error that is independent of the particular learning set and is related to assumptions made by the model [127]. Biased error is due to the model being biased towards a particular type of solution [128]. In this work the explicit relation between 2D and 3D is known (as derived in chapter 2), and the steps to solve for them (two-steps calibration method) are not changed between experiments. Since the assumptions made by the model (model shape and regression steps) are kept unchanged in this work, the bias is not impacted by the way to collect the 3D and 2D key points.

This work focuses on the collection of the training set. The variance part of the bias-variance error decomposition is related to the design of the training sample. The variance measures how much the model is ”over specialized” to fit a particular training set, but does not generalize well to new data [128]. Hence, it is related to the second type of error introduced above, which is due to the completeness of the training set.

This analogy enables to refine Gap 0 into two more specific Gaps as follow:

Gap 1: The training set must satisfy the completeness requirement. It must be an accurate representation of the application sets.

Overcoming Gap 1 would result in a lower variance than the one obtained when using a restricted training set. Consequently, it would result in a smaller calibration error.

Gap 2: The training set must satisfy the accuracy requirement. The noise on training data must be reduced.

Overcoming Gap 2 would lead to a smaller noise term, and thus a smaller calibration error.

The following paragraph introduces a research question and hypothesis related to these Gaps.

The use of a drone as a moving calibration target allows the collection of 3D key points and their matching 2D key points required for the calibration. These key points are used as input of a calibration function, and the reprojection error on the image can be computed. The proposed approach enables a successful collection of training points if Gap 0, and thus Gap 1 and Gap 2 are overcome.

The training set results from the drone trajectory in the camera FOV. Hence, answering the following research question helps overcoming Gap 1 and Gap 2:

Research Question 1: What drone path would enable the collection of a good training set and thus a successful camera calibration?

The drone path must satisfy the requirements established by Gap 1 and Gap 2, which can be reformulated as follow:

- The training set and the application sets must be homogeneous. In other words, the trajectory must capture points in regions of complex behaviors, as well as in linear regions in the field-of-view

- The noise on 3D and 2D key points must be low to ensure small ”irreducible error”. Thus, a synchronization, or an alternative method must be designed, to match 2D key points with 3D key points with low error

To model the camera correctly and satisfy the first requirement (completeness requirement), it is necessary to collect points covering the entire field of view. Indeed the distortion does not impact points near the image center in the same way as extremity points [102]. Figure 4.1 illustrates this observation: it highlights the difference between image points obtained through a pinhole camera (blue ones) and image points obtained through a camera whose lens provides radial distortion (red ones).

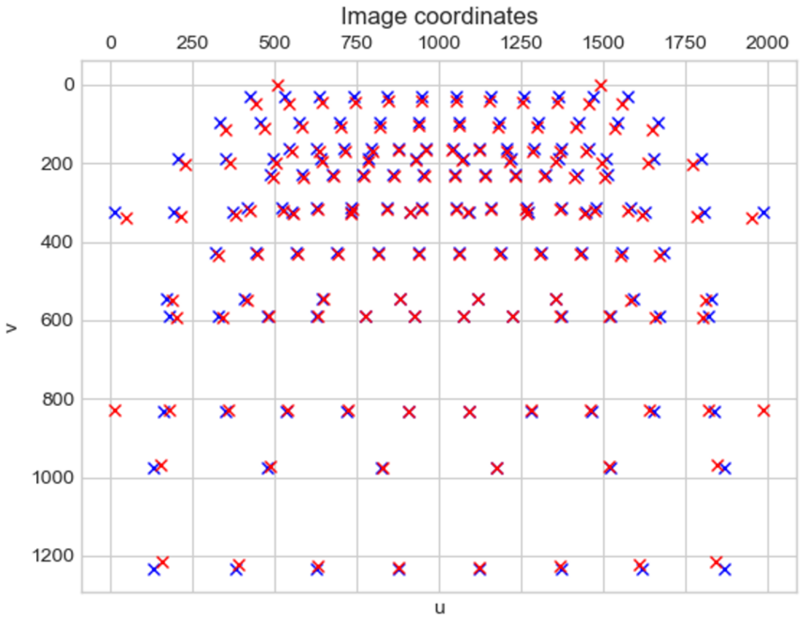


Figure 4.1: Radial distortion (red) versus no distortion (blue)

Also, using 3D key points located in a unique plane might bias the resulting camera parameters [100]. Thus, the drone path must satisfy the two following criteria to increase the likelihood of a successful calibration:

- FOV coverage
- Points collected in several planes

In addition to these criteria, the drone path must be kept simple in order to allow for an efficient calibration approach that is scalable to the large number of already installed cameras. To define such a path, a short introduction to path planning techniques is presented in the next paragraph.

Path planning tools enable finding a route that covers every point of a certain area of interest. Such methods have been developed for tasks such as surveillance or power line inspection using a drone. Usually, path planning starts with decomposing the space into cells. The center of a cell becomes a waypoint for the drone trajectory. If the coverage mission is performed over a regular-shaped and non-complex area, then the cell decomposition is not necessary [129]. In that case, simple geometric patterns are used to explore these areas, such as the ones in the Figure 4.2.

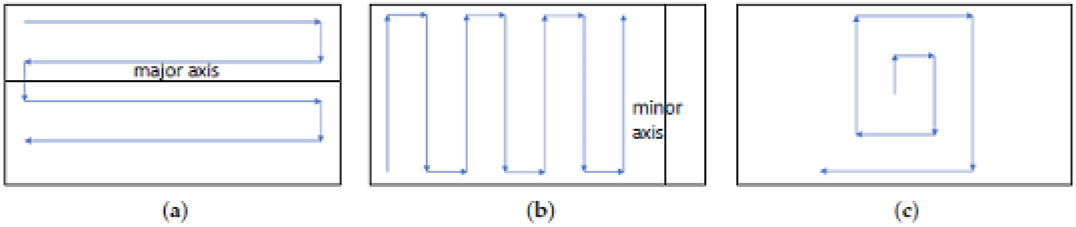


Figure 4.2: Simple path for area coverage

The field of view of a camera can be seen as a succession of simple rectangular planes, as shown in Figure 4.3 and Figure 4.4.

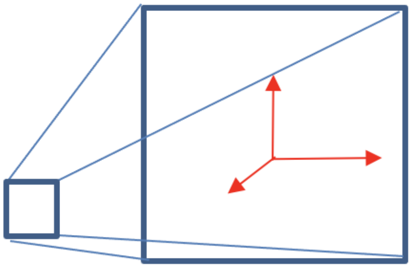


Figure 4.3: camera FOV shape

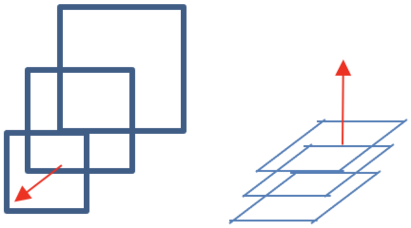


Figure 4.4: Division of camera FOV

From this field of view division and knowledge about traditional path planning methods, a simple path for the drone can be defined. The drone follows a back-and-forth path (Figure 4.2 a and b) in several vertical and horizontal planes contained in the camera's field of view.

Finally, the problem of 3D coordinates and 2D coordinates matching must be addressed. Errors in the correspondence between 3D and 2D points impact the training data quality, and thus the calibration quality. A simple method that tackles this problem is described below. First, the video time and the GPS time must be synchronized. Sensor synchronization requires determining the offset between the clocks of the data sources. In this work, the synchronization can be done by having the drone hovering at a location. When the drone starts, it is the initial time for the GPS and the video. Then, the 3D and 2D key points are obtained by taking the GPS location and detecting the drone on the image at synchronized times. When the drone reaches a waypoint, the corresponding time is obtained from the list of 3D locations and time provided by the localization sensors. The frame number that corresponds to this time is computed by using the video frame rate that can be obtained using OpenCV. Finally, the drone location on the corresponding frame is detected and provides the 2D pixel coordinates matching the 3D waypoint. Figure 4.5 illustrates the principle of GPS and image data matching.

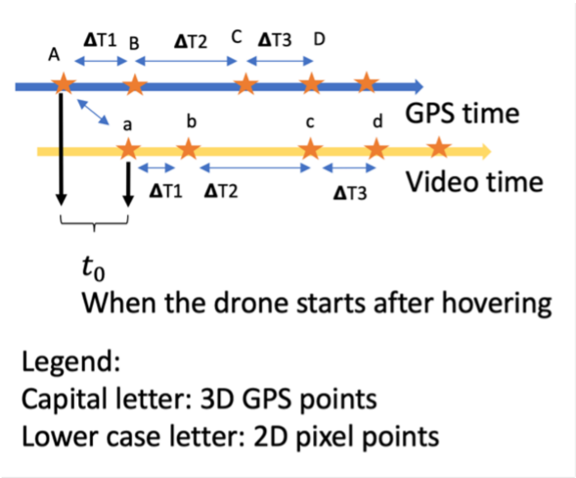

A
ΔT1
B
ΔT2
C
ΔT3
D
GPS time
a
b
c
d
Video time
ΔT1
ΔT2
ΔT3
t_0
When the drone starts after hovering
Legend:
Capital letter: 3D GPS points
Lower case letter: 2D pixel points

Figure 4.5: 3D and 2D points correspondence

However, this method is tricky because the GPS update rates and video frame rates might be very different. In that case, for a given GPS point, the corresponding image might not be captured by the video, which captures only specific frames per second depending on the frame rate. Inversely, the GPS update might be received after the actual position, which would bias the correspondence. This work proposes to enforce the drone to maintain its position stationary at waypoints to simplify the correspondence problem. So 3D and 2D waypoints are matched by extracting the list of points of zero speed in the GPS positions list and detecting the hovering points on the image. Then, the elements of the list of 3D stationary points are matched with the ones of the list of 2D stationary points. Figure 4.6 illustrates the approach proposed to overcome the correspondence problem.

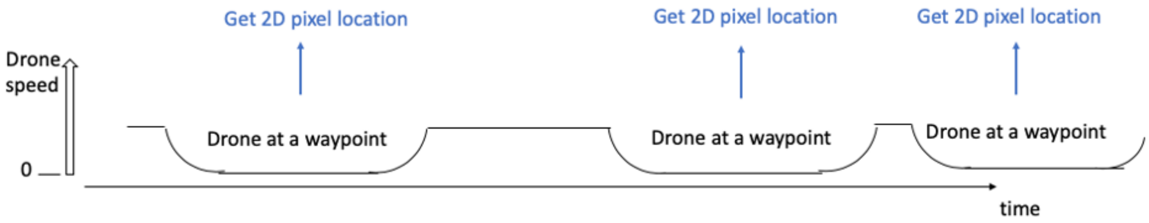


Figure 4.6: 3D and 2D points correspondence with hovering at waypoints

The three path criteria that must be satisfied to allow an efficient and robust drone-based camera calibration are recalled below:

- Simple path for short-time calibration
- Capture the regions of complex behaviors when the camera is not a pinhole
- Synchronization, or alternative, to match 2D key points with 3D key points without errors

This section explored ways to satisfy these criteria and thus enhance the likelihood of a successful calibration. The proposed path must be tested to learn whether the use of a drone as a moving target allows camera calibration despite image measurement uncertainties. The proposed path is summarized as follow:

> Hypothesis 1: IF the camera field of view is decomposed into several 2D planes and the drone follows a back-and-forth path in each plane while maintaining a stationary position at waypoints, THEN waypoints collected along this trajectory allow for a successful calibration.

The proposed matching process is expected to lead to a robust correspondence between 3D world points and image points. But this solution must be tested to make sure this matching process results in a noise on training data low enough for a successful calibration. Also, errors in the localization of the target on the image are introduced by small drone oscillations at waypoints, which causes blurring and small changes of the actual 2D key point location. Moreover, the proposed target can not be detected with a pixel accuracy on the contrary to a chessboard like calibration pattern. Indeed, for the later, the change in pixel intensities between its corners can be detected very precisely leading to a pixel accuracy. The detection of a key point on the drone will likely not be as accurate. Thus, the proposed solution in Hypothesis 1 must be tested. If the resulting calibration error is smaller than 3.6 pixels, then the use of a drone as a moving calibration target enables to build a complete training set that covers the FOV, and the matching process, as well as the image detection errors are low enough to lead to a successful calibration. Hypothesis 1 is tested in an Experiment whose design and results are described in the following sections. This experiment is made using simulations with ROS and Gazebo platforms. The next section introduces these simulations.

4.2 Simulation introduction

A simulation environment capable of modeling real weather and environment conditions for UAVs' applications is used to evaluate the calibration approach's feasibility. Indeed, such an environment models accurately the drone movements, such as small up and down oscillations when hovering, which impact the image detection accuracy. ”Robot Operating System” (ROS), combined with Gazebo, are adapted tools to model real-world conditions and UAVs flights [130]. Gazebo is a 3D simulator that enables robot simulations in complex indoor and outdoor environments [131]. The elements of a robot are created in Gazebo using Universal Robotic Description Format (URDF) files [132]. These files describe the kinematic and dynamic of a robot, its visual representation, and its collision model. Examples of parameters described in a URDF file are the rotor drag coefficient, the rolling moment coefficient, or the max rotor velocity. Some ROS libraries, such as Rotors [133] or hector quadrotor [134], provide URDF files to model UAV systems in Gazebo. Once the robot is created, it receives order from the Operating System (ROS) computing environment. ROS is made of executing files called nodes that communicate using messages called topics, services, or actions.

The simulation allows modeling a camera, a drone, and a chosen drone path in the camera FOV. The frames captured by the camera are saved, and the resulting video is used to detect the drone image location. Thus, 3D and corresponding 2D points are obtained from the simulations and then are input of a calibration function that computes the camera parameters. The relation from 3D to 2D is obtained using these parameters' estimates, and the corresponding reprojection error is computed.

Figure 4.7 summarizes the steps previously described.

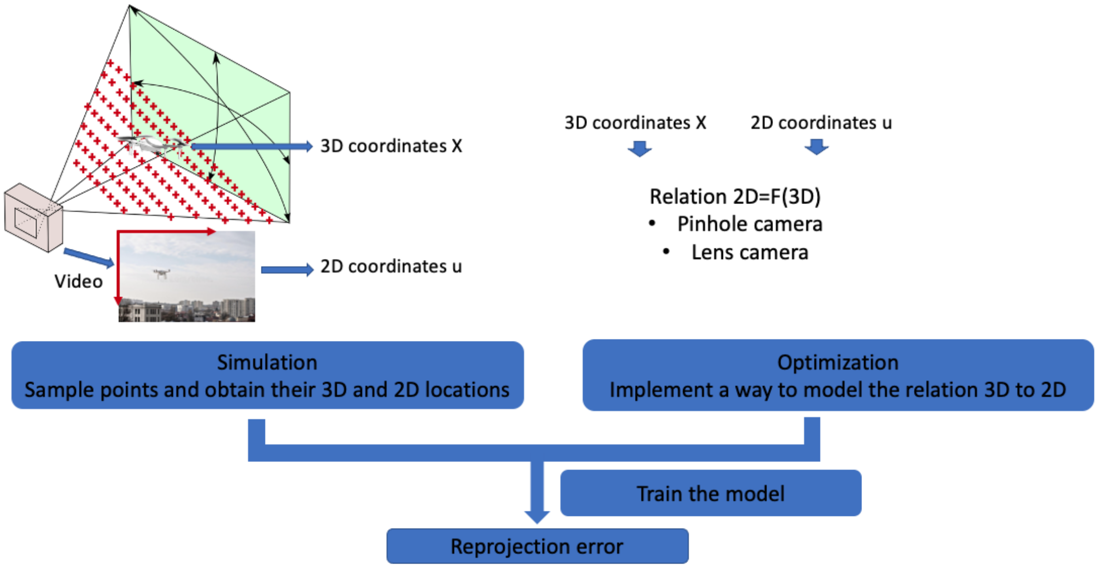


Figure 4.7: Simulation main steps

4.3 Experiment 1

This section provides a description of how the Gazebo and ROS simulations are used to test Hypothesis 1. Hypothesis 1 is tested for a pinhole camera first, as it is expected that the error would be smaller when there is no distortion. Once the method has been tested for this simplified case, it is tested for a camera with distortion. In the simulations, the drone flies to 3D waypoints in the camera field of view. The camera video is used to detect the drone where it remains stationary, which corresponds to the waypoints. In this experiment, the drone detection is made in a semi-automated way with a clicking method. The simulation enables the collection of a set of 3D and matching 2D points, which is used in the optimization process for the calibration. The reprojection error on the image is computed and is used as accuracy metric.

The general flow of Experiment 1 is summarized in Figure 4.8. The blue section corresponds to the simulation part, the pink section is the optimization for the calibration whose results is compared to the threshold of 3.6 pixels to classify the calibration as successful or not.

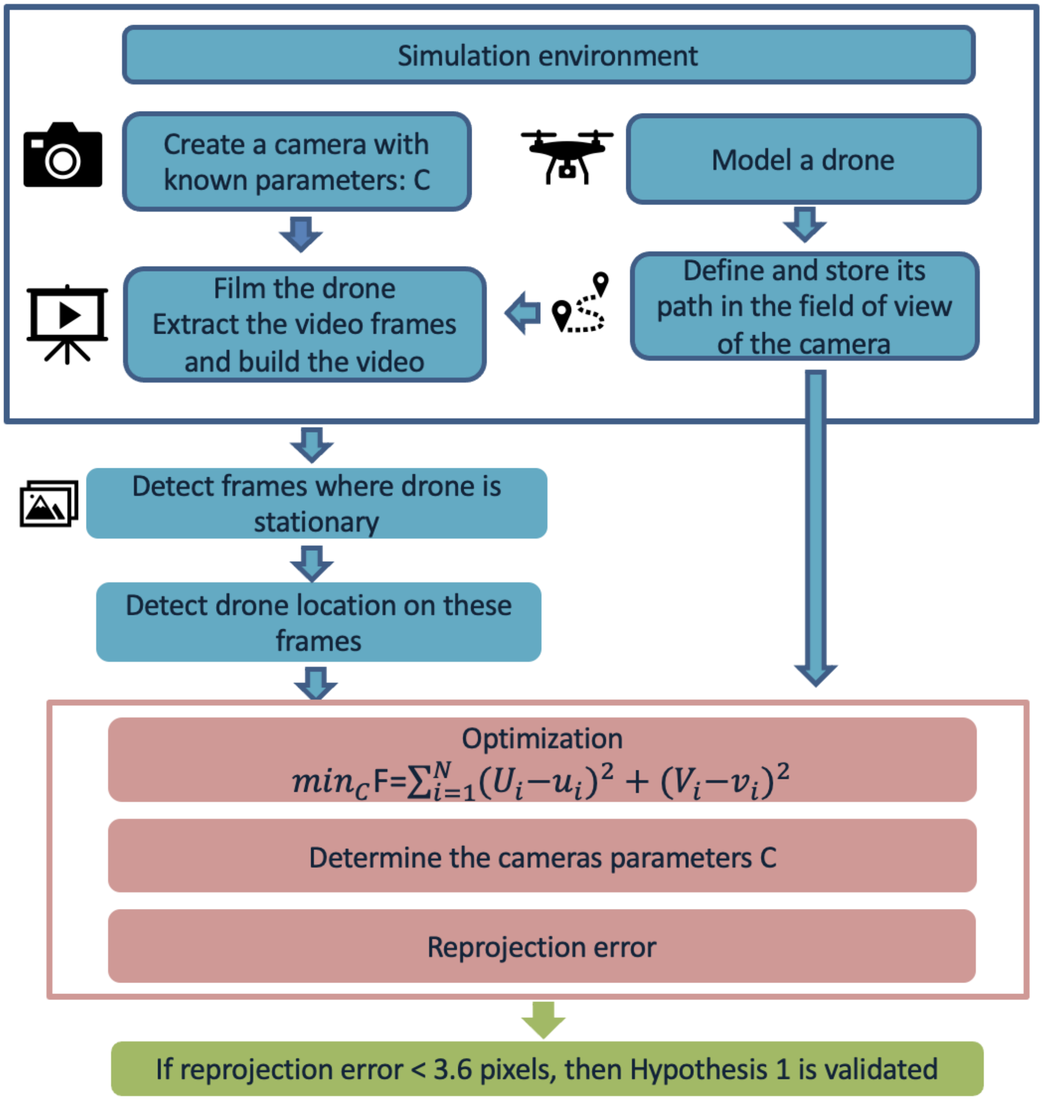
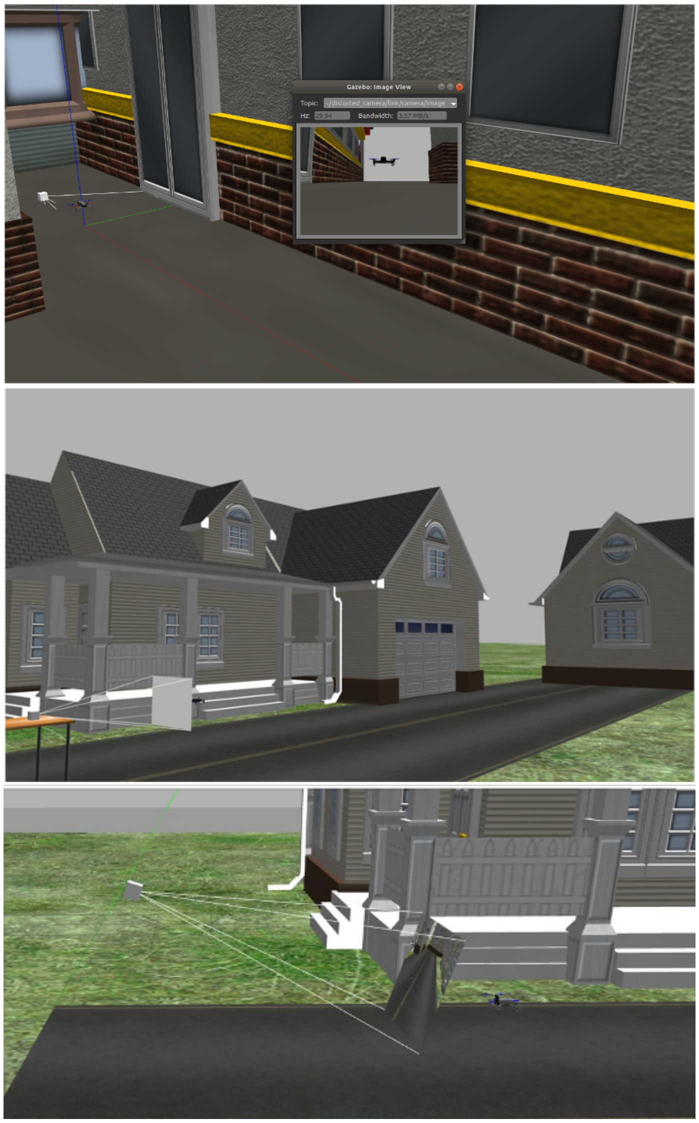

Simulation environment
Create a camera with known parameters: C
Model a drone
Film the drone Extract the video frames and build the video
Define and store its path in the field of view of the camera
Detect frames where drone is stationary
Detect drone location on these frames
Optimization
$min_C F=\sum_{i=1}^{N}(U_i-u_i)^2+(V_i-v_i)^2$
Determine the cameras parameters C
Reprojection error
If reprojection error < 3.6 pixels, then Hypothesis 1 is validated

Figure 4.8: Flow of experiment 1

In this work, the simulated drone is the Parrot Bebop 2, which is part of the Rotors package of ROS [135]. Urban environments with houses, roads, and traffic signs are created in Gazebo. Environments are called ”world” in the Gazebo framework and we would refer to this terminology in the document. When creating the world in Gazebo, a camera sensor is added. Gazebo enables to visualize the simulated drone, the simulated camera, and the world as illustrated with Figure Figure 4.9. The top illustration provides a close view of a simulated world. In this example, the camera is set on the ground and the small rectangle window represents the camera view. The middle illustration provides a more general view of a simulated world, the camera is set on a support (table here), and its field of view is represented with a white cone. The bottom illustration shows the drone flying in the camera field of view which is represented with a white cone in the simulation. The camera is the white bloc on the left.



Topic: ~/distorted_camera/link/camera/image

Hz: 29.94 Bandwidth: 3.57 MB/s

Figure 4.9: Simulated drone, camera and urban environment visualized in Gazebo

The camera parameters used in these simulations correspond to standard camera parameters. Two cameras, a pinhole and one with lens distortion are modeled in Gazebo. Their parameters are provided in Table 4.1. The extrinsic parameters are expressed in the world frame, which is the Gazebo frame. Figure 4.10 illustrates the world frame coordinates and the camera location. The X_w, Y_w, Z_w axes are represented with the red set of axes on Figure 4.10. The yellow point highlights where the camera is located on these axes.

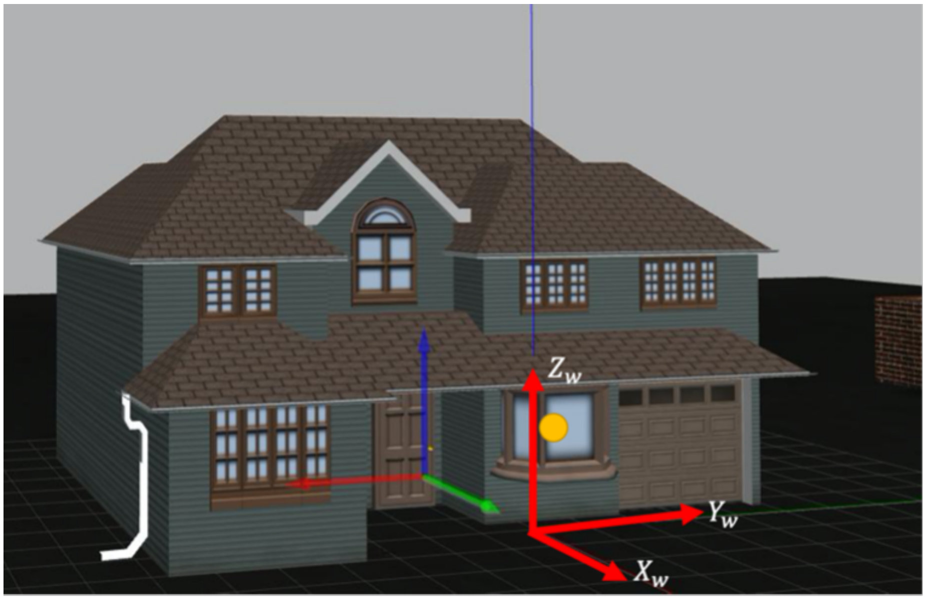


Figure 4.10: Illustration of the world axes

Table 4.1: Cameras parameters

Camera model	Pinhole camera	Camera with distortion
Focal length (pixel)	1110	1110
Image center coordinates (pixel)	(640, 480)	(640, 480)
Angle of rotation along y (degree)	30	30
Translation (m)	T_x=0.5, T_y=0, T_z=2	T_x=0.5, T_y=0, T_z=2
Radial distortion coefficient	none	k_1=-0.25
Tangential distortion coefficient	none	p_1=-28e-5, p_2=-5e-5

For each camera, the drone trajectory is defined in a ROS node before running the simulation. The trajectory follows the constraints imposed in Hypothesis 1 and recalled below:

- Coverage of FOV
- Back-and-forth path
- Hovering at waypoints
- Fly in several planes

Two paths are tested. The difference between the set of waypoints is their distance to the camera. Table 4.2 provides the waypoints for the first path tested, and Table 4.3

provides the waypoints for the second path tested. Figure 4.11 illustrates the first path, and Figure 4.12 illustrates the second path. On these figures, the navy blue points are the waypoints were the drone stops and the light blue edges is the path followed by the drone.

Table 4.2: 3D waypoint first path

Waypoint id	X_w	Y_w	Z_w
1	4	-1.5	0.5
2	4	-1.5	2
3	4	-1	2
4	4	-1	0.5
5	4	0	0.5
6	4	0	2
7	4	1	2
8	4	1	0.5
9	4	2	0.5
10	4	2	2
11	5	-1.5	1
12	5	-1.5	2
13	5	-1	2
14	5	-1	1
15	5	0	1
16	5	0	2
17	5	1	2
18	5	1	1
19	5	2	1
20	5	2	2
21	6	-2	0.5
22	6	-2	2.3
23	6	-1	2.3
24	6	-1	0.5
25	6	0	0.5
26	6	0	2.3
27	6	1	2.3
28	6	1	0.5
29	6	2	0.5
30	6	2	2.3
31	6	2.5	2.3
32	6	2.5	0.5

Table 4.3: 3D waypoint second path

Waypoint id	X_w	Y_w	Z_w
1	3	-1	0.5
2	3	-1	2
3	3	-0.5	2
4	3	-0.5	0.5
5	3	0	0.5
6	3	0	2
7	3	0.5	2
8	3	0.5	0.5
9	3	1	0.5
10	3	1.5	2
11	3	1.5	0.5
12	4	-2	0.5
13	4	-2	2
14	4	-1	2
15	4	-1	0.5
16	4	0	0.5
17	4	0	2
18	4	1	2
19	4	1	0.5
20	4	2	0.5
21	4	2	2
22	5	-1.5	1
23	5	-1.5	2
24	5	-1	2
25	5	-1	1
26	5	0	1
27	5	0	2
28	5	1	2
29	5	1	1
30	5	2	1
31	5	2	2

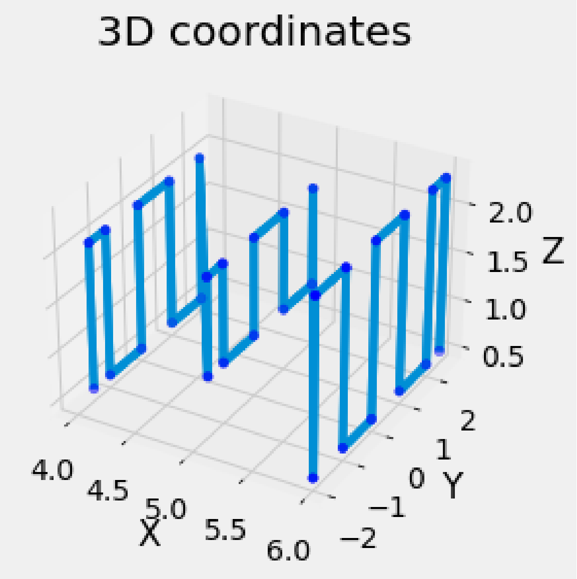


Figure 4.11: First trajectory

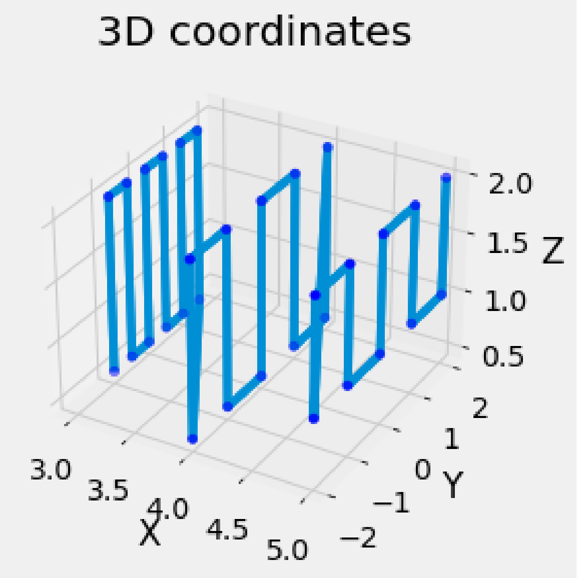


Figure 4.12: Second trajectory

The camera films the drone when it flies to these waypoints. The frames acquired by the camera are saved in a specific folder. The frame rate and the output folder name are initialized when creating the camera component in Gazebo. The frame rate is taken equal to 30 fps, as it is the usual security camera frame rate. The matching between 2D and 3D points relies on the detection of stationary position on the video. Also, in a real setting, the camera provides a video feed instead of separated images. Thus, a video is constructed from the set of frames gathered in the output folder. It is done with the ffpmeg function called in the command window. Figure 4.13 illustrates the set of frames obtained from the simulation, and the video after conversion.

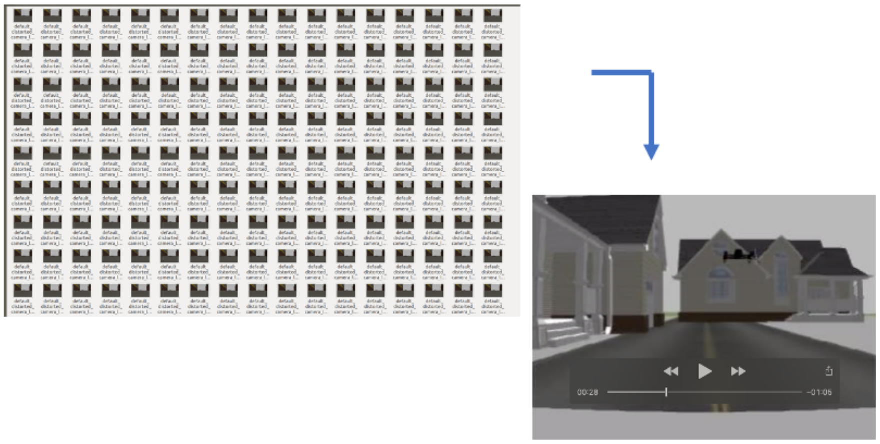


Figure 4.13: From the Gazebo output to the video

The video of the drone flying to waypoints is processed to collect the 2D key points. In this experiment, this step is done in a semi-automated way, that will be called ”clicking method” for the rest of the thesis. The choice of this clicking method is motivated as follow: if an automated drone detection method is used, such as template matching, the accuracy for the detection of a keypoint on the drone would be equal or worse to the one of the clicking strategy. This observation also holds if a convolutional neural network (CNN) is used to detect the drone. The label of the CNN would be the area selected by the cursor in the clicking method. The CNN detection error would be at least as large as the one from the labeling method, i.e. the clicking method. Based on these observations, the clicking method is first evaluated. If the calibration accuracy resulting from this method is deemed too low for the purpose of this thesis then an improved method will be investigated, and there is no need to implement a CNN for the drone detection.

An algorithm is implemented to read the video and allow the user to press keyboards key when he notices an event of interest. In this experiment, the drone keeping position stationary for a few seconds is the event of interest. Then, the algorithm prints a window with the frame that contains the event of interest. Finally, the user clicks on the drone image position with the cursor, as illustrated in Figure 4.14 and the algorithm stores the corresponding pixel coordinates in a table. This table is the final output of the algorithm, and corresponds to the list of 2D key points. Figure 4.15 illustrates the acquisition of 4 successive 2D key points, and the resulting table of pixel coordinates.

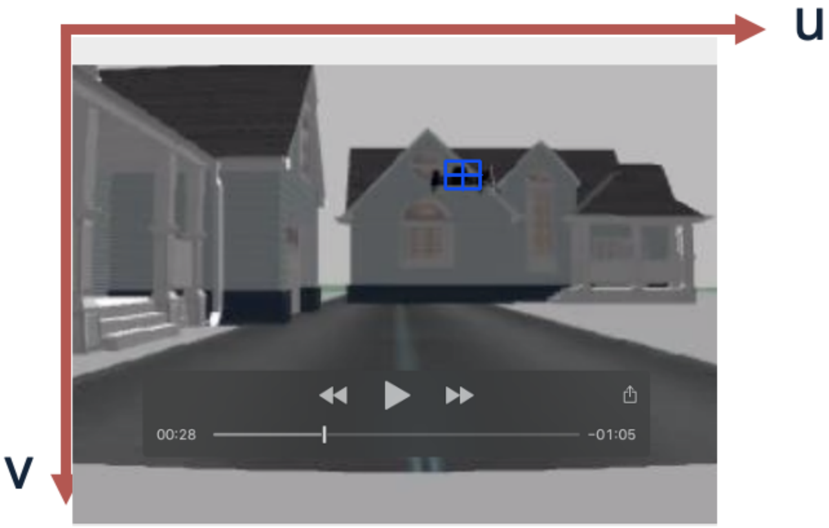
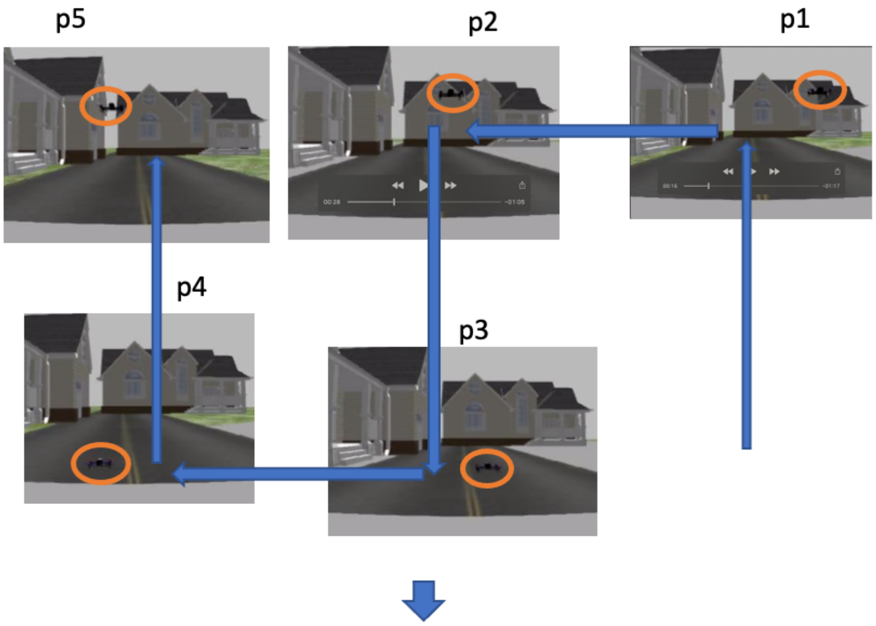


Figure 4.14: Clicking method



Point	u	v
p1	345	67
p2	305	67
p3	305	387
p4	265	387

Figure 4.15: Acquisition of the 2D pixel coordinates of the waypoints

At this stage of the experiment, the 3D and matching 2D key points have been collected. Figure 4.16 illustrates the 3D paths followed by the drone on the left image. The waypoints are the corners of this path. The corresponding 2D points are represented in red on the right image.

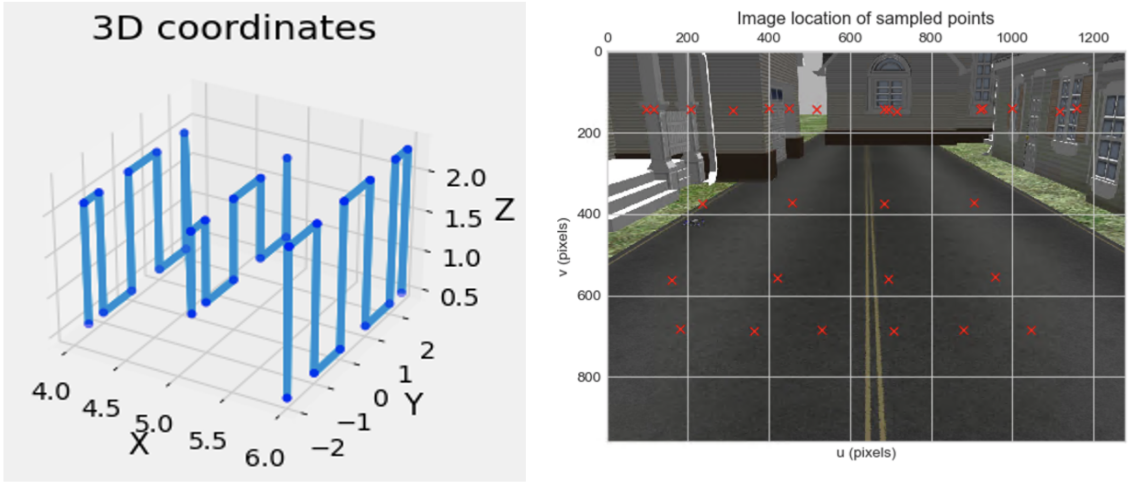


Figure 4.16: Simulation outputs: 3D and 2D key points

Once these key points are collected, they are used for the regression step. This step returns the camera parameters by minimizing the cost function $F = \sum_{i=1}^{N} (U_i - u_i)^2 + (V_{\;i} - V_i)^2$. U_i, V_i are the coordinates obtained through the model, and u_i, and v_i are the actual coordinates. For simplicity, the OpenCV calibration ”calibrateCamera” is used for this regression step [136]. Other methods were implemented and the details are in the Annex chapter, but they lead to similar results compared to the OpenCV function and were slower. Using the estimated parameters, the coordinates U_i, V_i can be computed for the 3D key points, and compared to the actual u_i, and v_i values. Figure 4.17 illustrates the actual 2D points with green color, and the reprojected ones with blue colors. The top left image is the output of Experiment 1 run for the pinhole camera and the farther path. The top right image is Experiment 1 result for the distorted camera and the farther path. The bottom left is obtained with the pinhole camera and closer path, the bottom right is obtained with the distorted camera and closer path.

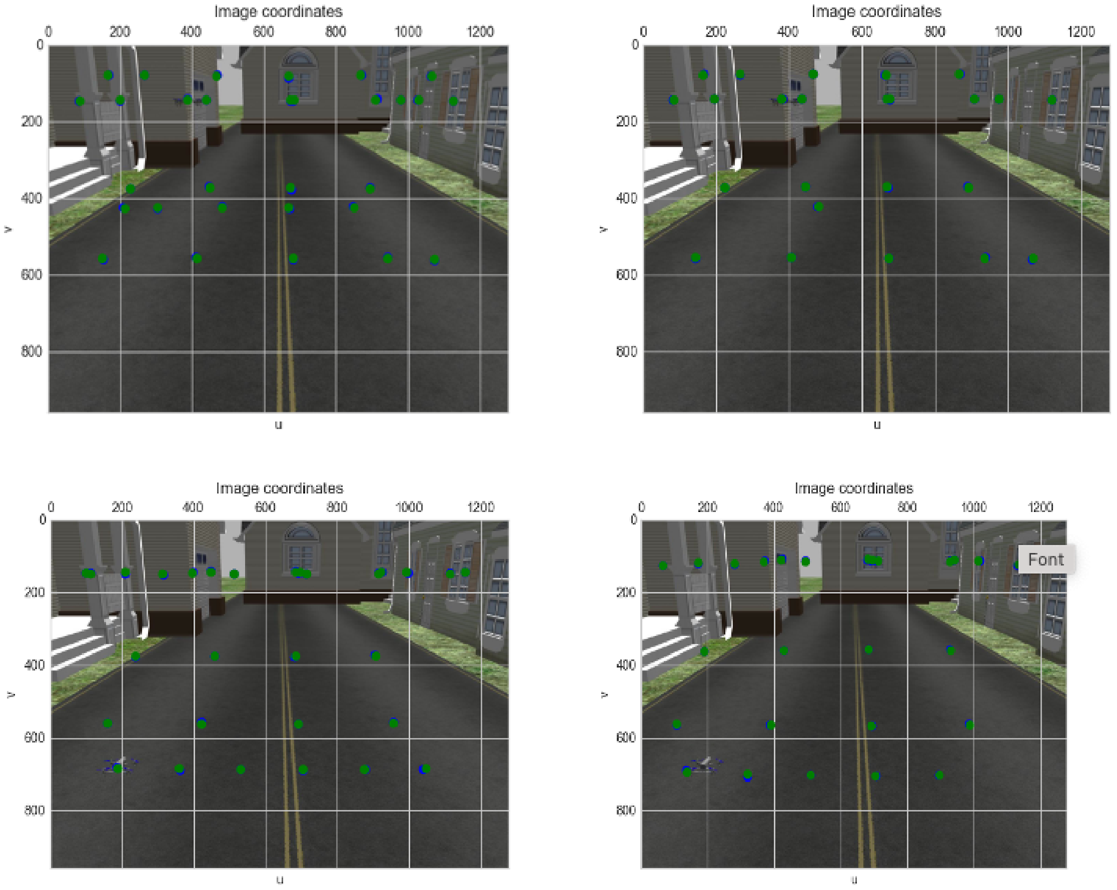


Figure 4.17: Actual points and their reprojection

The RMS of the reprojection error is computed at the end of the optimization. The training set is homogeneous to any application sets, because it covers the entire FOV. Thus, the accuracy can be estimated from the training set. Also, common calibration evaluation methods use the RMS reprojection error on the training set [137], so this support the choice of this evaluation metric.

The RMS reprojection errors obtained during this experiment are summarized in Table 4.7.

Table 4.4: RMS reprojection error

Path VS Camera type	Pinhole	Lens with distortion
Path 1	Error = 3.1 pixels	Error = 3.3 pixels
Path 2	Error = 3.19 pixels	Error = 3.28 pixels

All RMS reprojection error are smaller than 3.6 pixels. 3.6 pixels is the threshold defined in Chapter 3 to distinguish a successful calibration: error smaller than 3.6 pixels, from a calibration insufficient for speed measurement application: error larger than 3.6 pixels.

Result Experiment 1: The trajectory enables the collection of a complete training set. The resulting calibration error is smaller than 3.6 pixels for the tested cameras.

Thus, **Hypothesis 1 is validated**, and there is strong evidence towards the success of the drone-based calibration approach for hard-to-reach cameras. The next section summarizes Experiment 1 and its result leading to the validation of Hypothesis 1.

4.4 Take-away Experiment 1

The proposed solution to the calibration of hard-to-reach cameras relies on the use of a drone as a moving target. The uncertainties in the target image detection induce errors in the calibration. Thus, the proposed approach must be tested to evaluate the resulting error and the feasibility of the approach.

The drone trajectory is constrained by the FOV coverage need so that the model captures non-linear behaviors. Indeed, if only points at the center of the image are captured and the lens has a large radial distortion, then the calibration will have poor accuracy outside the image center. The proposed solution to this issue was stated in Hypothesis 1.

Experiment 1 implemented simulations using ROS and Gazebo frameworks. In these simulations, the drone follows a back-and-forth path to cover the FOV and collect 3D and matching 2D points representing well any application points. It follows such a path in several planes to avoid bias when computing the camera parameters. This path ensures the collection of an adapted training set for the regression part of the calibration process. The hovering condition ensures a simple matching between 2D and 3D key points, which is expected to be robust enough for the calibration purpose, i.e. the error resulting from this correspondence strategy would be low enough to enable a successful calibration.

Experiment 1 was run to test whether the calibration resulting from this set of 3D and 2D points is accurate despite non-linearity introduced by the lens and detection uncertainties. The detection uncertainties are caused by oscillations of the drone at waypoints, which makes the exact 2D localization more difficult to capture, and causes blurring and thus noise when identifying the 2D key points. Also, the detection error is larger than with traditional calibration objects for which a very good pixel accuracy can be reached through a robust detection of pixel intensity changes at corners of the chessboard-like pattern. That's why the proposed solution is tested in Experiment 1.

The errors obtained for the two tested cameras, and the two tested paths are all smaller than 3.6 pixels, which is the threshold for successful calibration. Consequently, **Hypothesis 1 is validated**.

HYPOTHESIS 1 IS VALIDATED: IF the camera field of view is decomposed into several 2D planes and the drone follows a back-and-forth path in each plane while maintaining a stationary position at waypoints, THEN waypoints collected along this trajectory allow for a successful calibration.

The flow between the main Gaps, the Overarching Research Question, Research Question 1 and its answer with the validation of Hypothesis 1 is summarized in Figure 4.18

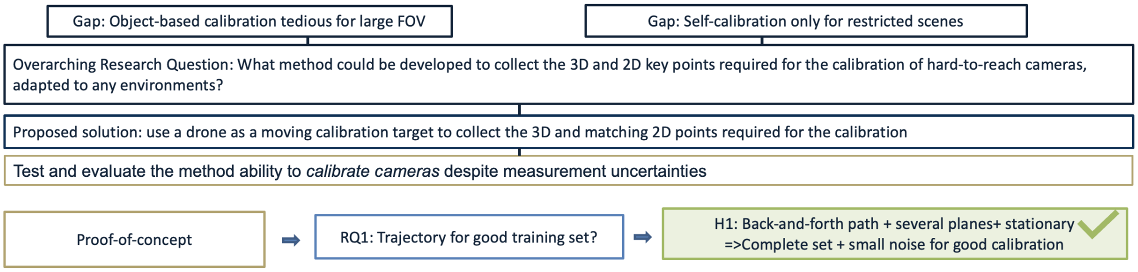
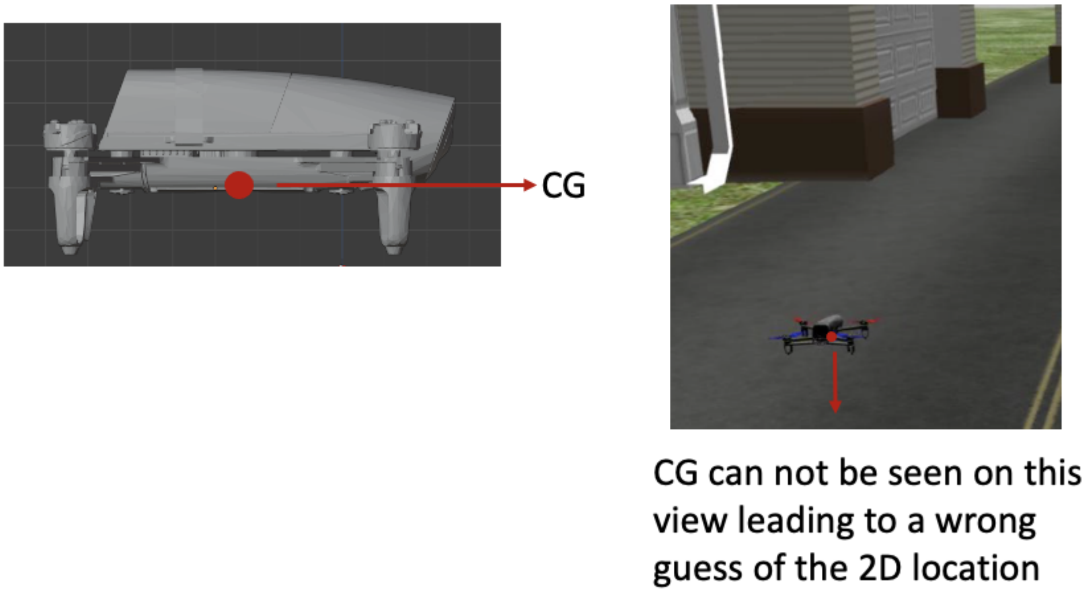

Gap: Object-based calibration tedious for large FOV
Gap: Self-calibration only for restricted scenes
Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?
Proposed solution: use a drone as a moving calibration target to collect the 3D and matching 2D points required for the calibration
Test and evaluate the method ability to *calibrate cameras* despite measurement uncertainties
Proof-of-concept
RQ1: Trajectory for good training set?
H1: Back-and-forth path + several planes+ stationary
=>Complete set + small noise for good calibration

Figure 4.18: Summary of Experiment 1 and validation of Hypothesis 1

The accuracy of the 2D sampling impacts the calibration accuracy. Random errors for the measurement of the drone image location are introduced by unpredictable events, such as small drone oscillations at waypoints. It is also impacted by the key point tracked on the drone. Indeed, when using a traditional calibration object, the key points are detected accurately by identifying large intensity gradients in several directions on the image, which are characteristics of the object corners, i.e. of the 2D key points. The method presented in Experiment 1 for the 2D key points detection must be adapted to reduce the 2D measurement error. This will in turn allows for a reduction of the noise term in the regression error decomposition into bias, variance and noise. The following section presents the related research question and hypothesis.

4.5 Reducing noise on training data

In the approach presented in the previous section, no specific drone detection method is specified. The simpler method is to detect the drone by clicking on it on the image. However, this method introduces human errors in the 2D sampling and is subject to target point occlusion for some views. Let us assume that the target point is the drone center of gravity (CG). The following observation is also valid for other target points such as the blades' center. There are some views where the target point can be seen and detected accurately; however, the same point can not be seen on some other views, and its estimated 2D location is inaccurate. Figure 4.19 illustrates a camera view for which the drone center of gravity can not be detected accurately. In addition to this observation, the precision of the target detection could be improved by detecting large pixel intensity changes in the same way as for traditional calibration objects.



CG can not be seen on this view leading to a wrong guess of the 2D location

Figure 4.19: View for which an accurate target point detection is not possible

During the optimization process, the model might learn the noise introduced by the data. Consequently, the quality of input data influence the quality of the calibration. The uncertainty in the target 2D localization could be reduced, and improving the sampling accuracy would improve the mapping accuracy. Consequently, Gap 2, which is recalled below, could be further overcome.

> Gap 2: The training set must satisfy the accuracy requirement. The noise on training data must be reduced.

Using landmarks as a target can provide reliable localization information [138][139][140]. In [19], the accurate tracking of the target, which is a store that separates from an aircraft, is made using several large markers printed on the store and aircraft. A marker can be easily detected with an identification algorithm [138], which reduces human processing, and in turn reduces human error.

QR tags and Aruco markers are the most frequent tags used for indoor localization. Authors in [138] demonstrated that Aruco markers lead to a better detection accuracy compared to QR tags. They also tested QR and Aruco markers against robustness to blurring, distortion and occlusion and noticed that these viewing conditions have a negative impact on the detection accuracy.

So adding a marker on the drone is a potential solution to enable the collection of a more accurate 2D sampling set. It is assumed that a drone with a marker can be designed so that performance constraints, such as weight or stability constraints, are satisfied. However, the solution of adding a marker to the drone would work if the marker is detected despite potential blurring due to small oscillations of the drone at waypoints, and distortion. Also, the drone should maintain a minimum distance to the remote camera being calibrated to ensure a safe flight. This distance corresponds to the obstacle avoidance distance that a moving drone must maintain when flying autonomously. Authors in [141] present research on computer vision techniques for obstacle avoidance of UAV using monocular vision. In their work, they set the safety distance between drone and obstacle between 3.5 and 4.5 meters (m). Authors in [142] present an obstacle avoidance strategy for UAVs, and test their method for safety distances equal to 1m, 1.5m, 2m and 3m. The average safety distance UAV-obstacle measured from the benchmark obtained from [141] and [142] is equal to 2.6 meters. The closest integer to this safety distance is 3 meters. Based on this observation, the drone must fly at least 3 meters away from the camera being calibration in this thesis work. The marker must be detected on the remote camera video despite a camera-drone distance larger than 3 meters.

These three observations lead to the following Gap:

Gap 2.1: The detection of the drone-marker system must be robust to distortion, blurring, and be detected for distances to the camera larger than 3 meters

In addition to the requirements stated in Gap 2.1, the solution marker-drone is adapted to the problem tackled in this thesis, if it allows an automated detection of the marker in a large variety of environments. Remote cameras are located in diverse environments that can lead to difficulty for the marker detection. The challenging scenarios include urban environments, where the background is often cluttered which can lead to false positive detection when identifying the marker, i.e. some object of the background are wrongly identified as a marker. The challenging scenarios also include background environments whose color are similar to the marker color. In this case, it is harder to detect the marker border [138]. This leads to the following Gap, which combined with Gap 2.1 results in Research Question 2:

Gap2.2: The detection of the drone-marker system must be adapted to the diversity of environments

Research question 2: Is the marker-drone solution adapted to the collection of 2D key points in diverse environments, and thus is an enabler for an improved calibration compared to the clicking method?

In Research question 2, the key word ”adapted” refers to the capability of the marker-drone solution to enable accurate 2D key points detection while overcoming the obstacles summarized below:

- Challenging backgrounds
- Blurring of target due to small drone oscillations at waypoints
- Distortion

- Distance marker-camera larger than 3 meters

Aruco markers are good candidates, since their design made of a binary pattern make their detection particularly robust, and the marker can be detected when rotated in the environment, which is of particular interest in this work since the drone might be slightly rotated compared to the camera axis [143]. Adding an Aruco marker at an extremity of the drone would allow collecting the 2D training set so that its detection accuracy does not depend on the drone orientation. This solution must be tested to evaluate the impact of the drone oscillations, and the camera distortion, on the automated marker detection robustness.

Also, the marker must allow a robust detection in a large variety of backgrounds, as remote cameras are located in different environments. Its pattern must allow a robust detection, without mistaken the marker for a background object (obstacle 1 in the aforementioned list of obstacles). This impacts the pattern choice for the Aruco marker. Aruco markers are made of a matrix of black and white squares framed by a black border [144]. For a fixed matrix size, there are different combinations of white and black squares. These combinations are stored in a dictionary enabling the detection of a large variety of Aruco markers. The matrix size varies between 4*4 and 7*7 blocks [145]. The larger the matrix is, the more patterns are available in the dictionary. Figure 4.20 illustrates Aruco markers of different matrix size: from left to right the sizes are 4*4, 5*5, 6*6 and 7*7.

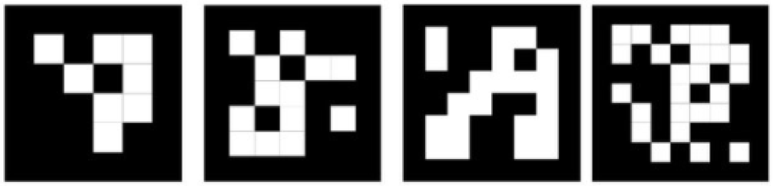


Figure 4.20: Aruco markers with different matrix sizes [145]

For the purpose of the drone-based calibration, a unique pattern is used and attached to the drone. So one could argue that there is no necessity of using a large size dictionary in this work, and so a large matrix size (the marker size is constant, but the block size inside the marker border would be larger for a smaller matrix size). However, the researchers who designed these markers recommend using a large number of bit transitions (i.e. a large number of color changes between blocks), so that the marker is less likely to be confused with an environment object [144]. On the one hand, a larger number of block reduces the false positive for the marker recognition. On the other hand, a larger matrix size requires a larger camera resolution to be identified correctly [146]. A matrix size equals to 6*6 is selected for this work as a trade-off between the false detection risk, due to the environment, and the resolution available. Aruco markers can be detected using an identification algorithm while getting a good accuracy [144]. Their detection is automated, which reduces error due to human processing during the 2D sampling step, and thus is expected to improve the calibration.

The marker is detected in an automated way using the following steps [144], that can be found already implemented in OpenCV [143] :

- Step 1: Image segmentation to extract the prominent contours
- Step 2: Filtering: Image contours that form a 4-edge polygons are kept, the others are discarded. So the algorithm detects potential markers by detecting rectangular contours
- Step 3: Marker identification: The inner region of these contours is analyzed to extract a binary code, and compare it to the dictionary. This comparison enables to discard false detection due to confusion with environment objects

It can be seen with the second step, that the marker recognition is made by detecting its contours at first, without considering the inner matrix. The choice of the number of blocks in the matrix is used to compare the potential marker, i.e. the detected black contour, to the marker one is looking for. In this work, the third step is used to discard environment objects that could have been mistaken with the marker. The accuracy of step 1 and 2 (detection of a 4-edge black border) for a marker far from the camera is impacted by the marker dimensions (in centimeters).

The larger the marker edge is, the easier is its identification on the image. However, the drone dimensions are constrained by major design criteria, such as its weight or dimensions. As an example, FAA flight authorization is based on the drone weight [147]. Dimensions also play an important role for the scalability of the method. The first object-based calibration method relied on the use of a 3D object, but its use was abandoned in favor of a planar calibration pattern because the 3D calibration object was too cumbersome [100]. Thus, it is preferred to keep the drone dimensions to conventional commercial drone dimensions. Ideally, a commercial drone would be used and slightly tuned to add a marker, which could be added on the drone directly, or in place of the camera to maintain the drone original weight. The marker must be smaller than the drone width. More precisely, for a simple solution it should fits between the propellers. This lead to the additional obstacle: The marker size is constrained by commercial drone dimensions.

The clicking method does not lead to false detection, however, the automated identification algorithm used to find the marker on the image can lead to false positive, which results in error in the 2D dataset. It is necessary to test whether the 6*6 Aruco marker can be identified correctly, without confusion with environment objects or missed detection, in the large diversity of environments where cameras can be located. It has been showed in the literature that the detection of Aruco markers work well when the background color is pale [144][138]. The identification of the marker is adapted to a large variety of environments, if it also works well in challenging environments. These environments include darker backgrounds, because the Aruco marker border is harder to detect due to its color similar to the background color. These environments also include cluttered backgrounds, such as urban backgrounds, where objects could be wrongly identified as the marker. Thus, the solution with the Aruco marker must be tested in challenging environments to answer Research Question 2.

To summarize the marker-drone solution must be tested to evaluate the impact of the drone oscillations, and the camera distortion, on the automated marker detection robustness. It must also be evaluated for challenging environments.

This leads to the following hypothesis:

Hypothesis 2: IF a 6*6 Aruco marker of restricted dimensions is added at an extremity of the drone, THEN the modified drone design enables an improved 2D points collection scalable to the variety of environments, robust to drone oscillations, to camera distortion, and accurate despite a safety distance of 3 meters

The associated experiment tests using simulations the marker-drone solution for cameras with distortion. The simulations model the drone oscillations at waypoint, which allows the evaluation of the marker detection robustness despite blurring due to small drone movements. The marker-drone solution must be tested against challenging backgrounds and distance drone-camera larger than 3 meters. If the solution leads to a successful calibration despite these two obstacles, then the marker-drone solution is adapted to the collection of 2D key points for the calibration, and Hypothesis 2 is validated.

4.6 Experiment 2

The drone detection method can be improved, by removing human error during the clicking method, and by locating a target marker designed for accurate detection. However, this marker approach is adapted to the collection of the 2D key points required for the calibration if the marker can be detected despite challenging backgrounds, blurring due to small oscillations at waypoints, and dimensions restricted by the drone dimensions. Experiment 2 is designed and run to evaluate the capability of the marker-drone system to collect the 2D key points, despite the obstacles mentioned above.

Experiment 2 is designed to answer Research question 2 that is recalled below:

Research question 2: Is the marker-drone solution adapted to the collection of 2D key points in diverse environments, and thus is an enabler for an improved calibration compared to the clicking method?

”Adapted” refers to the capability of the marker-drone solution to enable accurate 2D key points detection while overcoming the obstacles summarized below:

- Marker dimensions constrained by drone dimensions’
- Detection for distances marker-camera larger than 3 meters
- Robustness against small drone oscillations at waypoints (causing blurring for instance)
- Robustness against Distortion
- Marker detected in challenging backgrounds

To do so, simulations are designed to model a commercial drone (Parrot Bebop 2 drone) with a marker attached to it, and two types of challenging environments: urban environment, and dark background environment (mountain environment). The simulations also model drone oscillations at waypoints, thus the marker identification robustness to blurring caused by these small drone movements can also be evaluated. The simulations are implemented using ROS and Gazebo as described earlier in this chapter. The drone design is modified compared to the one used in experiment 1 to add a marker at its extremity. The design of the marker-drone system can be divided into two tasks: the choice of the marker location on the drone, and the marker size. These tasks were implemented using the Blender application, and resulted in several drone-marker designs. These drone-maker systems were tested for the calibration of cameras using ROS and Gazebo as in Experiment 1. The same paths and cameras (a pinhole camera and one with distortion) as the ones defined in experiment 1 were modeled in ROS and Gazebo simulations. Instead of identifying the drone on the image with a clicking method, the marker is detected in an automated way using the OpenCV Aruco library [143]. The number of image key points detected is used to adjust the marker size. Details about the two design steps: marker location on the drone, and marker size are described in the following subsections. Figure 4.21 summarizes the process and implementation steps of Experiment 2. The blue boxes refers to the steps implemented in Experiment 1, the red boxes refers to new steps being designed and implemented in Experiment 2.

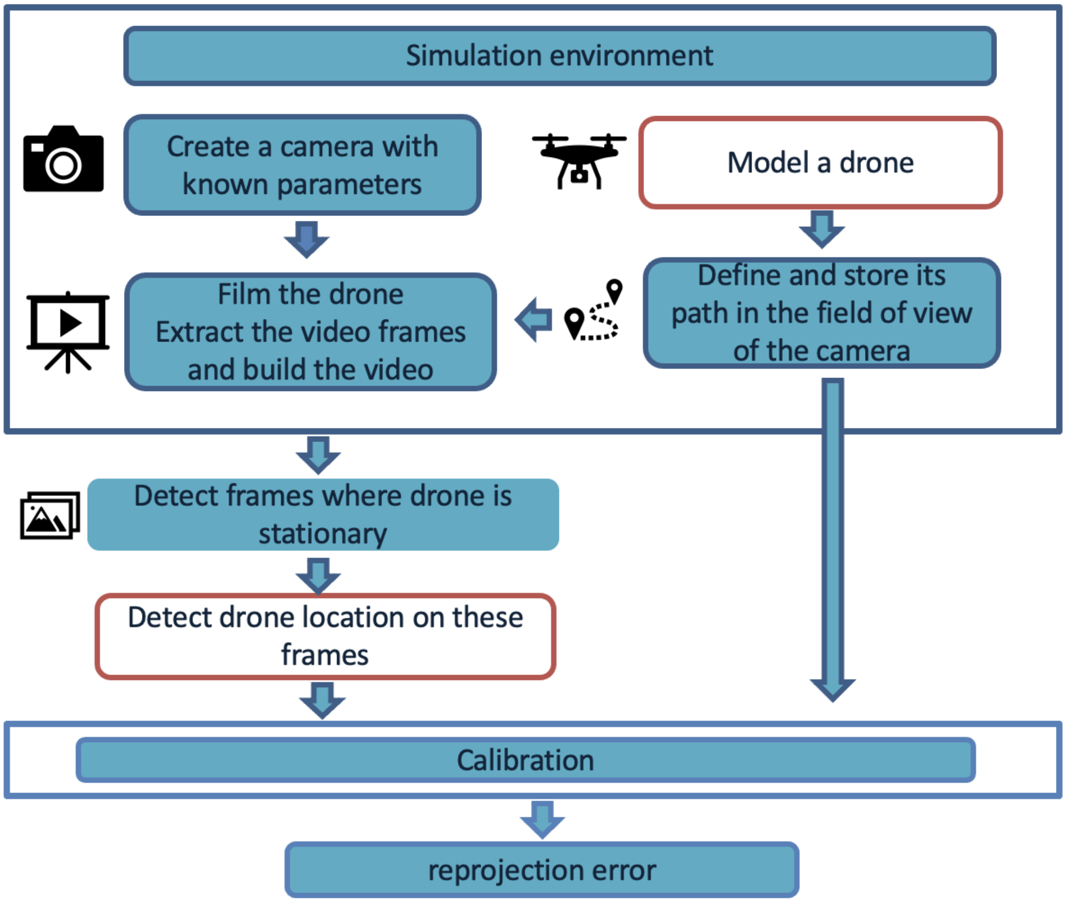

Simulation environment
Create a camera with known parameters
Model a drone
Film the drone
Extract the video frames
and build the video
Define and store its path in the field of view of the camera
Detect frames where drone is stationary
Detect drone location on these frames
Calibration
reprojection error

Figure 4.21: Process followed in Experiment 2

4.6.1 Marker location on the drone

Figure 4.22 represents a drone flying in front of a camera while following the path described in Hypothesis 1. Depending on the drone location, the drone sides seen by the camera at a location, might not be seen at an other location. For instance, the top side is seen for the drone location depicted on the top image, and on the third one on Figure 4.22. But it can't be seen on the second and last views. On this figure, the drone is highlighted in red. For each view, the drone is represented as a cube, whose visible sides are the same as the drone visible sides. One can see that for these four drone locations, the front side (represented in green on the right image) is always visible by the camera. Sometimes it is slightly rotated but Aruco markers are robust to rotation. Thus, if the marker is added at the front or back of the drone, the detection robustness of the target point on the image does not depend on the camera viewpoint, i.e. on small changes of orientation of the drone.

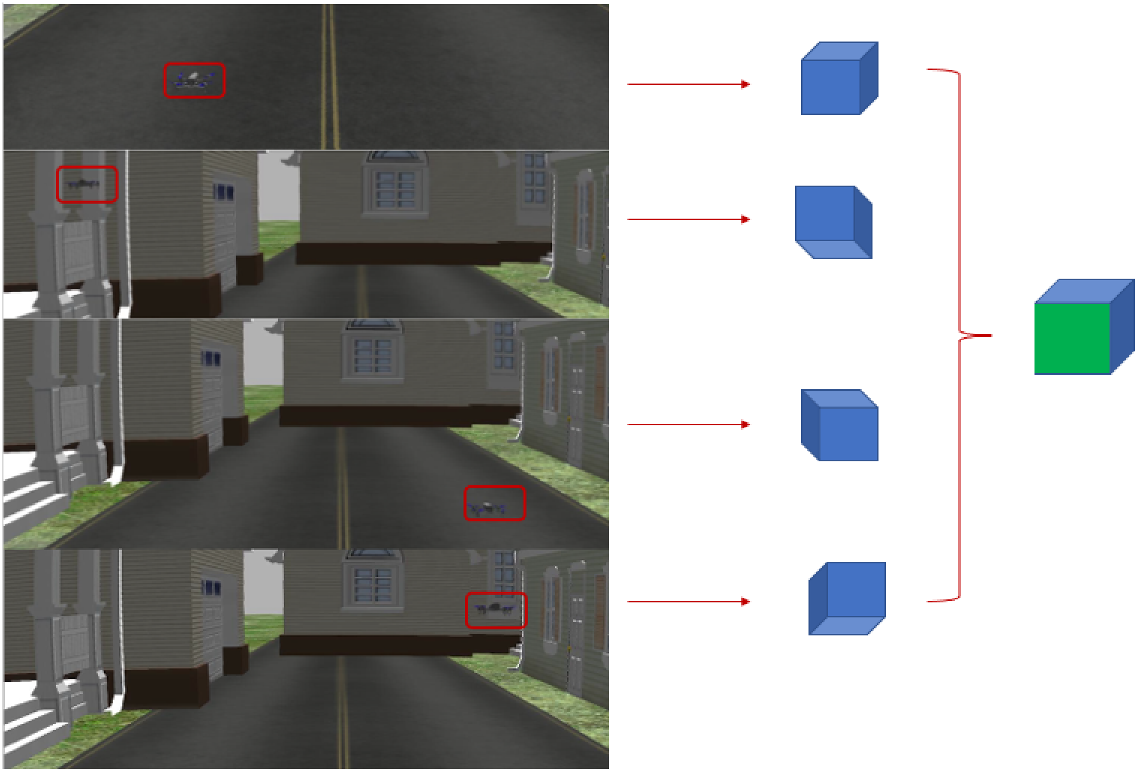


Figure 4.22: Choice of marker location

Realistic robot design in Gazebo are defined using mesh structures, which are a set of vertices, edges, and triangular faces. In this thesis, the mesh file of the Parrot Bebop drone used in Experiment 1 was modified to add a marker at the back of the drone. To operate this change of design, the Blender application is used. Blender is an open source software used to create 3D printed model, virtual reality and computer games [148]. First, a 3D rectangle shape, as represented on Figure 4.23 is created. On this figure, the parameter ”a” represents the edge dimension and this notation will be used in the next subsection. Then, the marker image is added to it, as can be seen on Figure 4.24. As explained earlier in this Chapter, the inner matrix size is taken equal to 6*6 to satisfy the trade-of between the false detection risk due to the environment, and the resolution available. The mesh file of the Parrot Bebop 2 drone is loaded in Blender and the marker and drone are merged together, leading to the system seen on Figure 4.25. The newly designed drone is saved as a .Collada file, so that ROS can use the drone-marker system physical description designed in Blender.

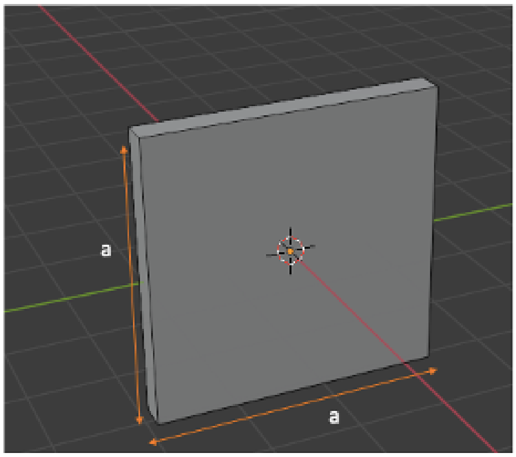


Figure 4.23: Support for the marker

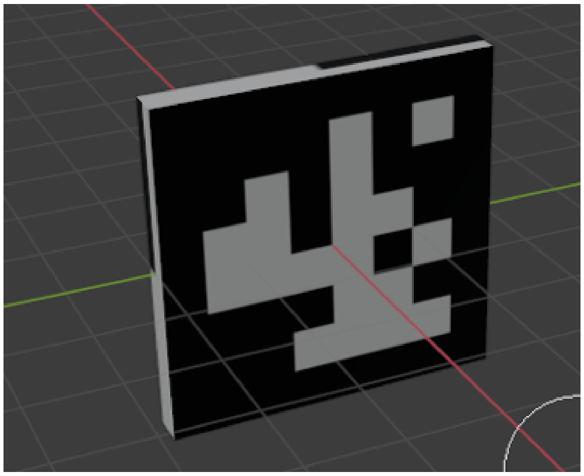


Figure 4.24: Marker created using Blender

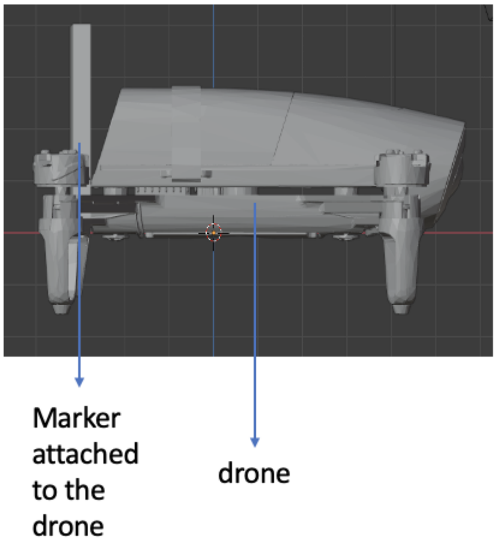


Figure 4.25: Marker-drone system

First, the Aurco marker was directly printed on the rectangle surface, and the simulation was run in an urban environment. However, due to the dark background of the urban environment whose color is very close to the marker border, the marker could not be detected for most views. The first steps towards the marker detection is the identification of edges, thus of large pixel intensity gradients. Then, if a set of edges form a polygon, the algorithm classifies the detection as a potential marker and further processing is done to refine the classification. Consequently, it is very important to detect the black edges of the marker, else the marker is not detected. Since the marker can be detected accurately on a white backgrounds [144], the idea is to reproduce the influence of a white background locally around the marker, so that the marker can be detected in pale and dark backgrounds. Figure 4.26 illustrates the initial marker design on the left, and the modified marker design with a white border on the right. The changed design was tested in the same simulation world (urban environment) and the marker location was detected correctly.

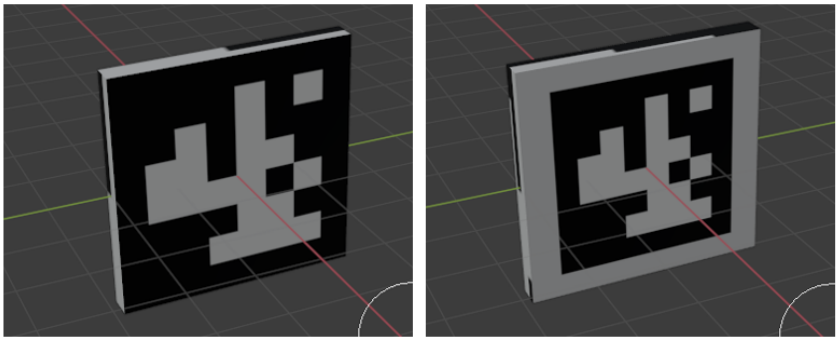


Figure 4.26: Marker design adjusted to improve its identification in a large variety of environments

First result of Experiment 2: A white border must enclosed the Aruco marker to enable the collection of 2D key points in challenging environments

A second parameter can be modified when designing the drone-marker system. This parameter is the edge dimension called ”a” and illustrated on Figure 4.23. The following subsection provides details about this parameter and its impact on the detection.

4.6.2 Marker size

The larger the marker is, the higher the chances are to be able to identify it in an automated way on the images. However, the marker dimensions are constrained by the drone dimensions. In order for a marker to fit on a commercial drone, its edge should be smaller than the distance available between the body width and the propeller length. The commercial drone taken as example in this work is the Parrot Bebop 2 drone. Its width measures 38 centimeters (cm), and its propellers' radius equals 11cm. The available space between the propellers is equal to $38 - 2 * 11 * cos(45) = 22.6cm$. To make sure the propeller rotation is not impacted by the marker, a safety distance of a few centimeters must be removed from this available length. So the marker edge should be of order of magnitude 10 cm to avoid impacting the propellers' rotation. Figure 4.27 depicts how the marker size is impacted by the drone dimensions.

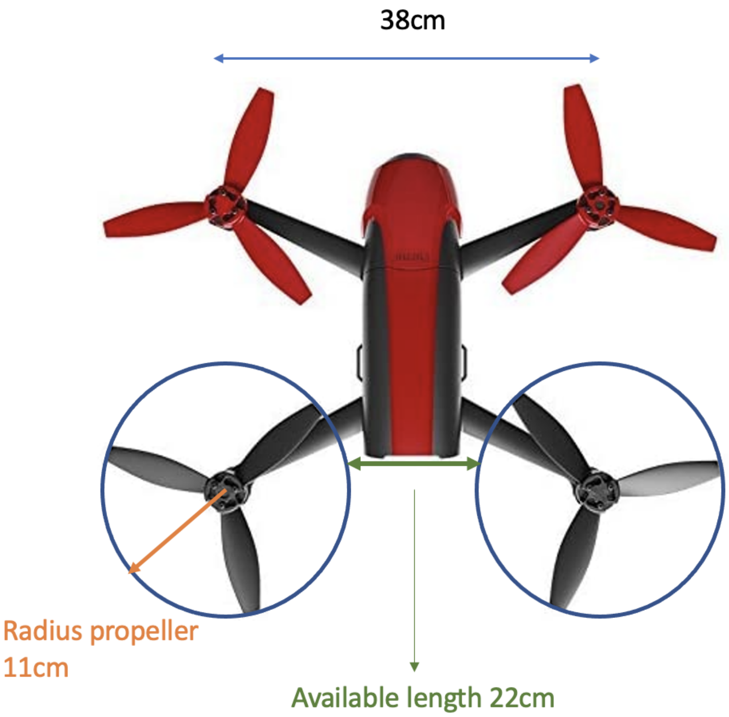


Figure 4.27: Drone dimensions

A marker of edge equal to 7.5 cm, and a marker of edge equal to 10cm are designed in Blender. The simulations with ROS and Gazebo are run for a pinhole camera, and for the first path presented in Experiment 1. The False Positive and False Negative rates are measured to evaluate the ability of the drone-marker system to collect the 2D points required for the calibration. The results are described in the next paragraph.

No object in the simulated world is mistaken with the marker, thus the number of False Positive detection is zero. However, there are frames where the marker is not detected, i.e. there are some False Negative. For the smaller marker size (7.5cm), only 67 percent of marker location are detected. Thus, the marker of edge 7.5 cm does not allow the collection of the 2D key points required for the calibration. For the larger marker size (10cm), there was a unique False Negative, leading to a detection rate of 98 percent. The results also demonstrated that the selected marker size enables the detection of the moving target for distances larger than 3 meters, since the minimum distance between the camera and the sampled points equals 3 meters for the tested path.

The following knowledge is gained from this part of Experiment 2:

> Second result of Experiment 2: A marker of edge 10cm enables the 2D points collection when the drone flies at least 3 meters away from the camera

Figure 4.28 illustrates the drone with the marker of 10cm, which is used in the rest of Experiment 2. Figure 4.29 provides the view of the drone and its marker seen on the video captured by the camera being calibrated.

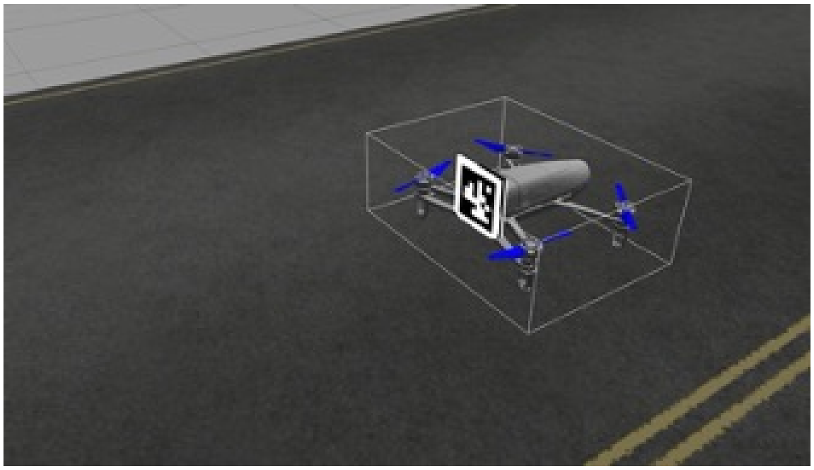


Figure 4.28: Drone with marker of edge 10cm, and using a white border for collection of image point required for the calibration in challenging environments

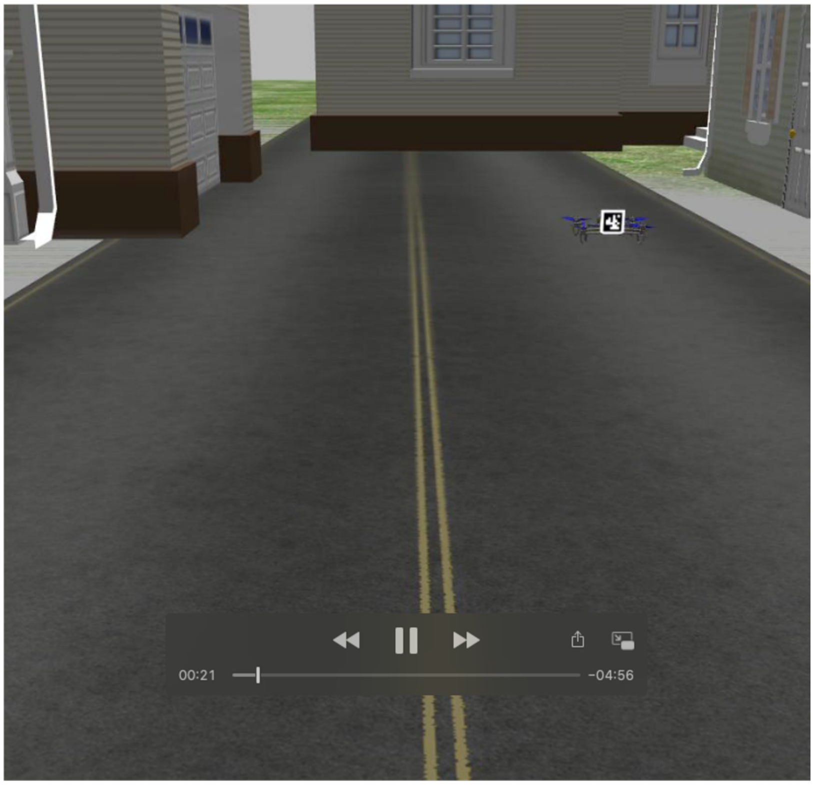


Figure 4.29: Drone with marker seen from the camera being calibrated

The design of the drone with a marker for calibration target purpose has been defined. Table 4.5 presents the criteria that must be satisfied by the marker-drone system for its validation on the left column. It presents the knowledge gained from Experiment 2 so far about the validation of these criteria in the second column, and the last column contains the additional work that needs to be demonstrated to fully answer Research Question 2.

Table 4.5: Knowledge gained and knowledge to acquire to answer RQ2

Criteria	Result	Additional work needed
Marker dimensions satisfy drone dimensions' constraints	Satisfied	/
Marker detected for distance larger than 3 meters	Satisfied	/
Robustness against challenging backgrounds	White border around the marker	Test for more environments
Robustness against small drone oscillations at waypoints	Marker detected	Look at the resulting calibration accuracy
Robustness against distortion	/	Test with distorted cameras

The marker-drone system was used in simulations in ROS and Gazebo to further demonstrate the efficiency of the solution for camera calibration. These simulations evaluate the success of the calibration approach in challenging environments, and for cameras with distortion. Explanations are provided in the next subsection.

4.6.3 Drone+marker calibration approach tested for different environments

An important criteria for the validation of this marker solution is the scalability of the approach to diverse environments where the cameras can be located. In the previous section, the marker solution has been tested in a urban environment. It lead to accurate identification of the marker with a detection rate equals to 98 percent. It also demonstrated that the chosen Aruco pattern enables to discard efficiently the objects that are not the marker since the False detection rate equals to zero, making the solution adapted to cluttered environments.

The environment tested had a grey background, and adding a white border around the marker was necessary for its detection. The automated marker detection steps are summarized earlier in Chapter 4 and it was highlighted that the Aruco marker is recognized on the image by identifying black contours with a 4-edge polygon shape. Thus, adding a white border around the marker does not impact its detection in light backgrounds since the black edges are unchanged. To further support this observation, authors in [144] demonstrated that the automated detection of Aruco markers works well on light color backgrounds. To provide more evidence towards the robustness of the marker-drone solution for challenging environments, i.e. for dark backgrounds, the proposed approach was tested in a darker simulated world. A mountain environment was chosen for this test, and is illustrated on Figure 4.30 and Figure 4.31. The drone follows the first path presented in Experiment 1, and the marker was detected on the image at all waypoints, validating the robustness of the marker-drone approach to diverse environments. Figure 4.30 are images of the simulation running, the white cone represents the camera field of view. Figure 4.31 are frames from the video captured by the camera, each frame shows a 2D keypoint. The background color around the marker changes during the drone trajectory: the top left frame background color is a light gray close to the marker border color, the two other frames have dark green and brown backgrounds. In this simulation, the drone and its marker are identified correctly at each waypoint, thus the marker solution is adapted to light and dark backgrounds.

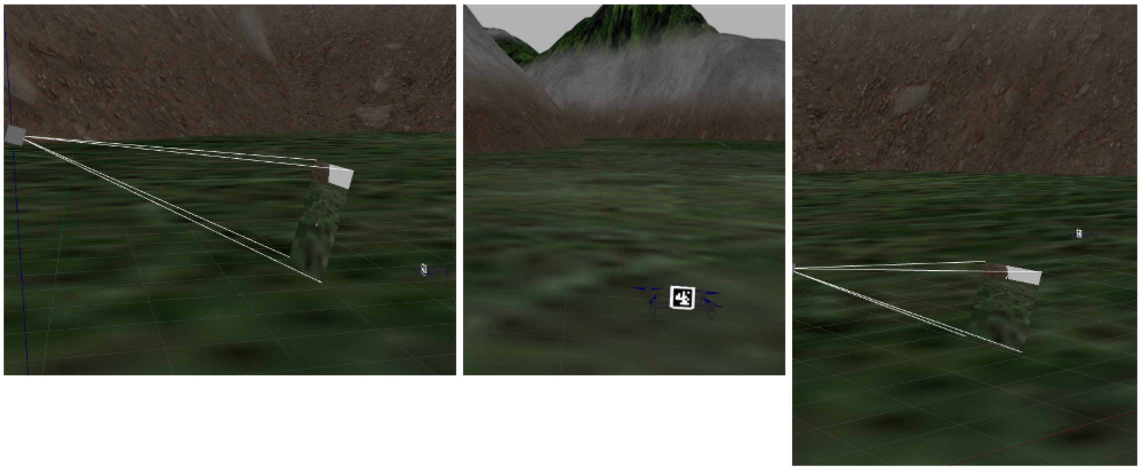


Figure 4.30: Simulation in a dark background environment

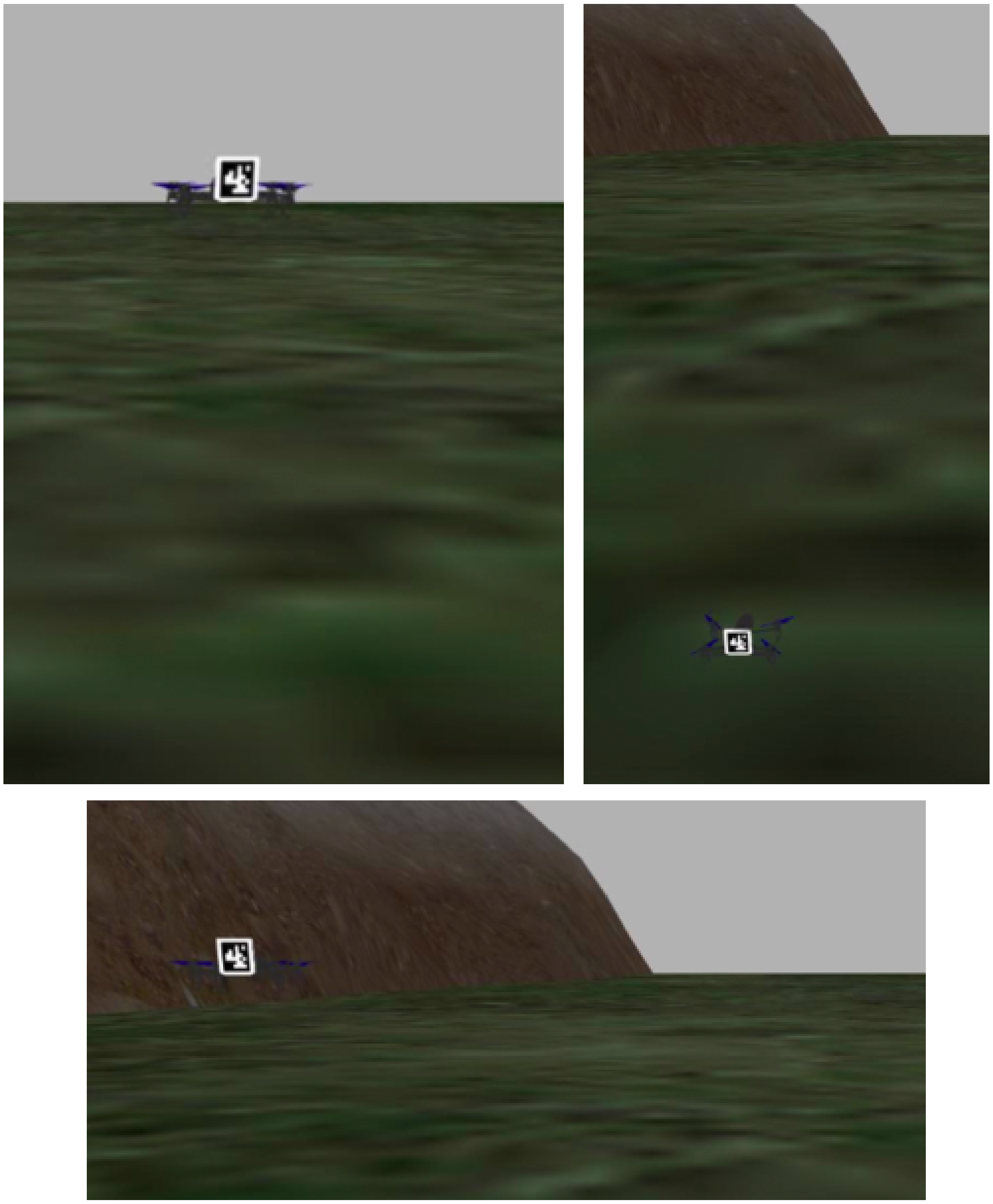


Figure 4.31: Sample of frames captured by the camera in the mountain environments

Table 4.6 summarizes the main results of Experiment 2 for the validation of the criteria to satisfy to validate Hypothesis 2. Most criteria are satisfied. In order to validate Hypothesis 2, the robustness of the approach against distortion must be demonstrated, and the resulting calibration accuracy must be measured. These results are presented in the following subsection.

Table 4.6: Knowledge gained and knowledge to acquire to answer RQ2

Criteria	Result	Additional work needed
Marker dimensions satisfy drone dimensions' constraints	Satisfied	/
Marker detected for distance larger than 3 meters	Satisfied	/
Robustness against challenging backgrounds	Satisfied	/
Robustness against small drone oscillations at waypoints	Marker detected	Look at the resulting calibration accuracy
Robustness against distortion	/	Test with distorted cameras

4.6.4 Drone+marker calibration approach tested for different paths, and different cameras

In Experiment 2, the same paths as the ones defined in Experiment 1 are tested. The same cameras as the ones defined in Experiment 1 are modeled (a pinhole camera, and one with distortion). These cameras capture a video of the drone with its marker. The drone flies at waypoints which are the corner of the blue path illustrated on Figure 4.32, and Figure 4.33. The green path on these figures represents the marker center path. The marker coordinates (in meter) compared (X_{marker}, Y_{marker}, Z_{marker}) to the drone center of gravity coordinates (in meter) (X_{drone},Y_{drone}, Z_{drone}) are the following:

$$X_{marker} = X_{drone} - 0.11$$

$$Y_{marker} = Y_{drone}$$

$$Z_{marker} = Z_{drone} - 0.07$$

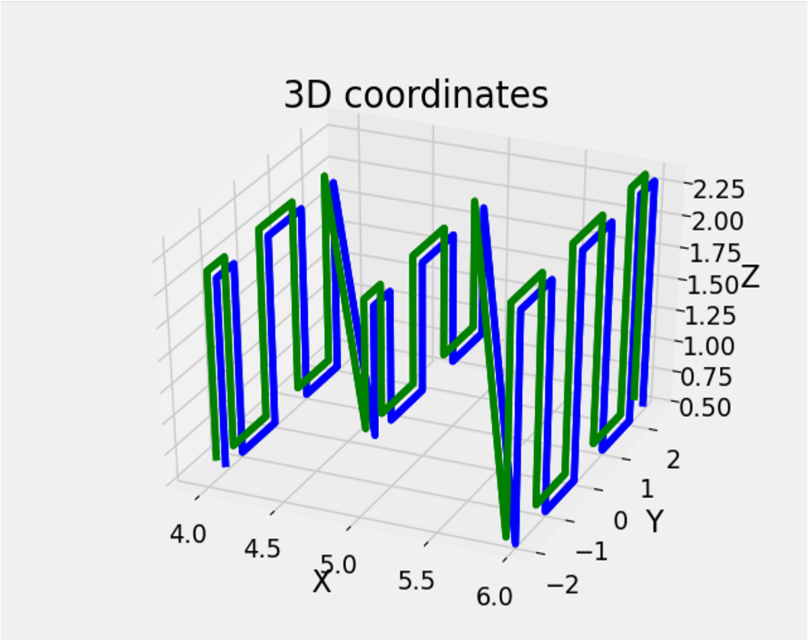


Figure 4.32: Drone and marker path 1

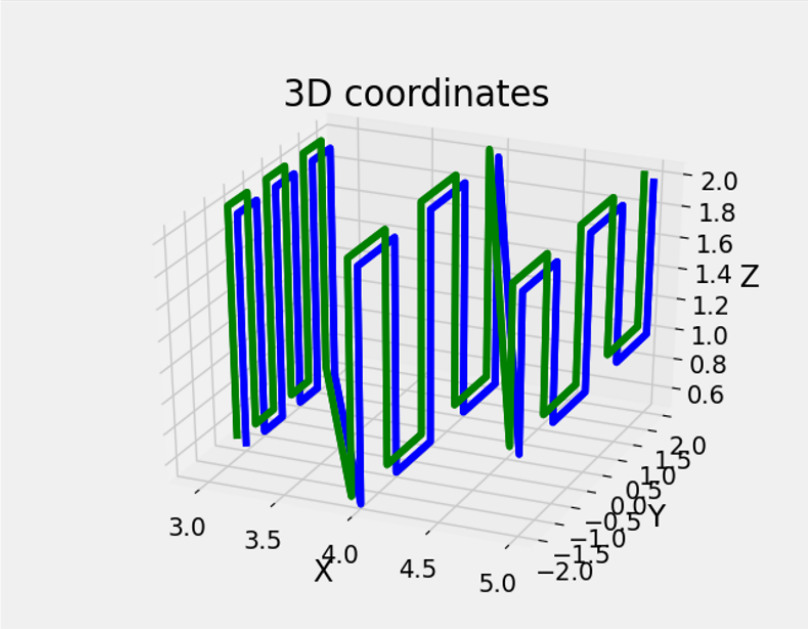


Figure 4.33: Drone and marker path 2

To further demonstrate the benefits of the marker approach, the calibration accuracy is compared between the clicking method and the marker method and is described below.

The RMS reprojection errors obtained during Experiment 2 are summarized in Table 4.7 The table also recalls the error obtained with the clicking method for comparison. Adding a marker decrease the reprojection error on the calibration by 24 percent.

Table 4.7: RMS reprojection error

Path VS Camera type	Pinhole	Lens with distortion
Path 1 - no marker	Error = 3.1 pixels	Error = 3.3 pixels
Path 2 - no marker	Error = 3.19 pixels	Error = 3.28 pixels
Path 1 - marker	Error = 2.12 pixels	Error = 2.28 pixels
Path 2 - marker	Error = 2.84 pixels	Error = 2.42 pixels

The lens distortion does not impact the marker detection, compared to the scenario where the camera is a pinhole one. Also, all resulting calibration errors are smaller than the threshold of 3.6 pixels, thus the calibration is successful. This observation enables to complete the criteria Table 4.8. It can be seen that the marker-drone solution satisfies all the requirements, and is thus adapted for the collection of the 2D key points.

Table 4.8: Knowledge gained and knowledge to acquire to answer RQ2

Criteria	Result
Marker dimensions satisfy drone dimensions' constraints	Satisfied
Marker detected for distance larger than 3 meters	Satisfied
Robustness against small drone oscillations at waypoints	Satisfied
Robustness against distortion	Satisfied
Robustness against challenging backgrounds	Satisfied

4.6.5 Validation Hypothesis 2

The research question addressed in the section is recalled below:

Research question 2: Is the marker-drone solution adapted to the collection of 2D key points in diverse environments, and thus is an enabler for an improved calibration compared to the clicking method?

The results obtained from Experiment 2 are summarized in Table 4.7, and Table 4.8. First, it was demonstrated that a 6*6 Aruco marker that satisfies the drone dimensions' constraints can be detected for distances larger than 3 meters. Then, it was demonstrated that the marker can be correctly identified despite small drone movements at waypoints. It was also shown that adding a white border around the marker improves its detection in challenging environments, and thus is required for the scalability of the approach. Experiment 2 also demonstrated the robustness of the approach against distortion. Finally, the calibration accuracy between the marker method and the clicking method are compared, and the marker method leads to a smaller calibration error. Thus, **Hypothesis 2 is validated**, and the marker-drone solution is adapted to the collection of 2D key points in diverse environments, and is an enabler for an improved calibration compared to the clicking method, which answers Research Question 2.

HYPOTHESIS 2 IS VALIDATED: IF a 6*6 Aruco marker of restricted dimensions is added at an extremity of the drone, THEN the modified drone design enables an improved 2D points collection scalable to the variety of environments, robust to drone oscillations, to camera distortion, and accurate despite a safety distance of 3 meters

4.7 Summary

Figure 4.34 summarizes the main observations, questions and hypotheses introduced so far.

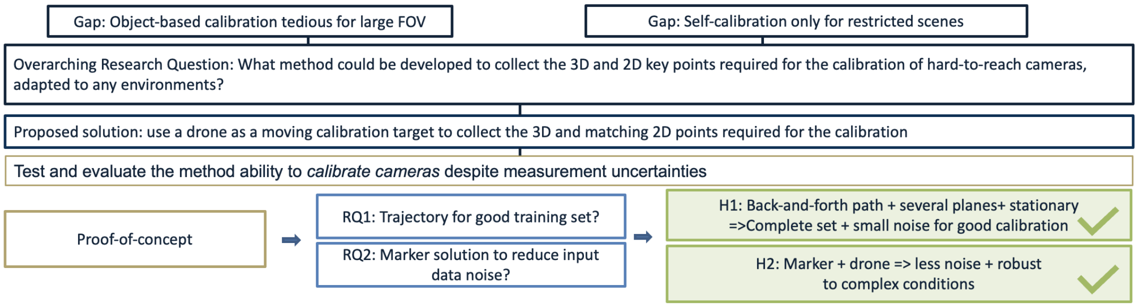

Gap: Object-based calibration tedious for large FOV
Gap: Self-calibration only for restricted scenes
Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?
Proposed solution: use a drone as a moving calibration target to collect the 3D and matching 2D points required for the calibration
Test and evaluate the method ability to *calibrate cameras* despite measurement uncertainties
Proof-of-concept
RQ1: Trajectory for good training set?
RQ2: Marker solution to reduce input data noise?
H1: Back-and-forth path + several planes+ stationary =>Complete set + small noise for good calibration ✓
H2: Marker + drone => less noise + robust to complex conditions ✓

Figure 4.34: Summary of Gaps, Research Questions and Hypotheses for the proof of concept

Experiment 1 demonstrated that if the drone flies following the three criteria listed below, then the calibration made with the key points collected by the drone is successful.

- The path covers all areas of the camera FOV
- The path covers points in several planes
- The drone stops at waypoints to allow for the correspondence between 3D key points and 2D key point

The two first criteria allow for the collection of a training set that is complete, i.e. that is an accurate representation of any application sets. For instance, this path includes points from regions of complex behaviors due to distortion, and points from regions of linear behavior. The third criterion allows the collection of a training set with low error when matching 3D key points with their images. The resulting training set leads to a calibration error smaller than 3.6 pixels and thus a successful calibration. Through Experiment 1, It has been proven for two common cameras, that the use of a drone as a moving calibration target enables the collection of the 3D and 2D key points to build a dataset complete and accurate enough for a good quality calibration.

Hypothesis 2 tests whether the error during the 2D sampling could be reduced by adding a marker on the drone. It is expected that adding a marker would reduce the 2D sampling error and thus the calibration error, if the marker can be identified accurately. The marker pattern and dimension influence its identification in the image. The marker-drone system must be detected in challenging environments, and be robust to oscillations, distortion, and scaling due to the camera distance. Thus, Hypothesis 2 was tested to find out whether the marker-drone solution is adapted to the flight conditions and the large diversity of environments where remote cameras are located. Experiment 2 demonstrated that the solution marker + drone overcome the obstacles listed above, and lead to a calibration accuracy improved compared to the one obtained in Experiment 1. Experiment 2 lead to the validation of Hypothesis 2.

To summarize the findings from Experiment 1 and 2: both a drone with a marker and one without a marker and a pinhole camera without distortion and a camera with distortion were tested. The tests were run in simulations using ROS and Gazebo frameworks, and by simulating two different drone trajectories. Both methods enable the collection of key points that cover the field of view. The calibration resulting from the 3D waypoints and their 2D images is evaluated by measuring the RMS reprojection error, and results are summarized on Figure 4.35. All RMS reprojection errors are smaller than 3.6 pixels, which is the threshold for an accurate calibration. Also, adding a marker at the back of the drone decreases the error by 24 percent on average. These results confirm that the choice of the detection method influences the calibration accuracy. These experiments' results provide strong evidence toward the success of the drone-based approach for the calibration of hard-to-reach cameras.

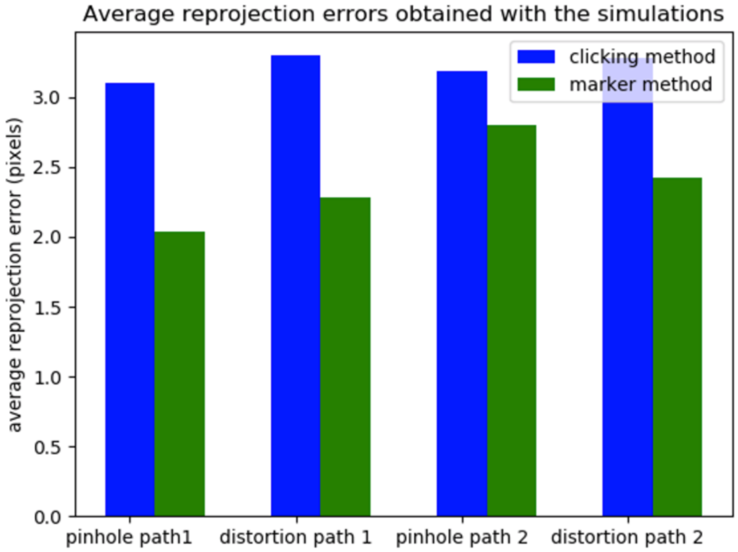


Figure 4.35: RMS reprojection errors

Results of Experiment 1 and 2 are a proof-of-concept for the drone-based calibration approach. The approach must be further demonstrated for a large range of camera parameters. Also, it was assumed that the 3D localization can be obtained without error. The impact of the measurement error on 3D coordinates must be included to better evaluate the calibration approach. Chapter 5 presents the work done to evaluate the approach for many different cameras, and for noisy 3D localization measurements.

CHAPTER 5

DEMONSTRATE THE SCALABILITY TO MANY DIFFERENT CAMERAS

Because Hypotheses 1 and 2 are validated, there is strong evidences that using a drone as a moving target enables the calibration of hard-to-reach cameras. The approach presented so far assumes perfect 3D localization information. The approach is scalable to the large number of existing cameras if it relies on commercial drones without the need to add complex sensors for the 3D localization. Thus, the measurement error due to a commercial 3D localization sensor, such as an RTK GPS, must be included to prove the scalability of the approach to a broad range of environments and cameras. Also, guidance must be provided to deploy the drone-based calibration approach easily for any type of camera. The following section presents the methodology defined to evaluate the influence of the GPS uncertainty on the calibration and provide guidance for the drone trajectory.

5.1 Strategy to evaluate impact of path parameters, uncertainties, and camera parameters on calibration quality

The drone-based calibration is scalable to the large number of installed cameras if it enables a successful calibration of a broad range of camera models while using commercial localization sensors, such as an RTK GPS. The error introduced by the GPS causes a calibration error, and the approach must be tested to evaluate whether the threshold defined in Chapter 3 is satisfied despite this additional error. Also, the impact of GPS uncertainty on the calibration quality depends on the localization of the 3D training points. Figure 5.1 illustrates this observation: objects A and B have the same height, but their respective images, a and b, have a different heights, which depends on the distance from the camera. Thus, the same error in 3D does not result in the same error on the image for two different distances camera-measured points. This is summarized with the following observation:

Observation: The impact on the calibration quality of the 3D measurement noise depends on the position of these 3D points

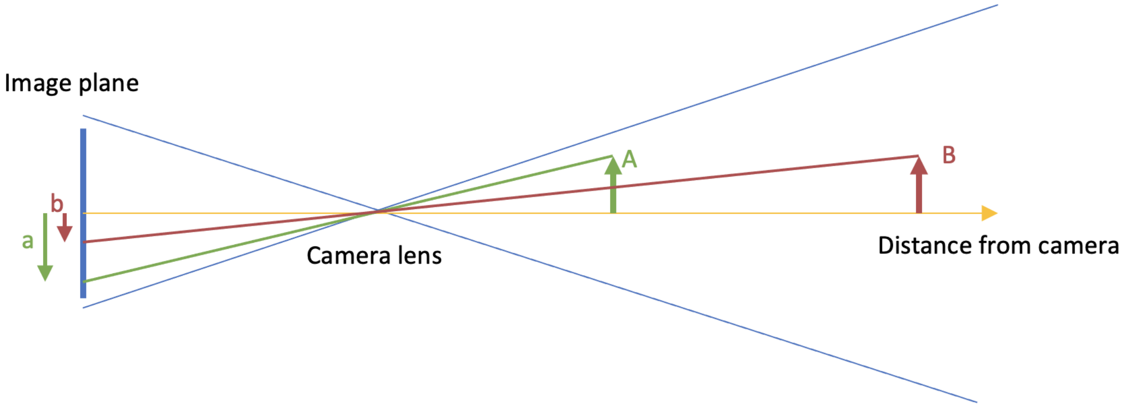


Figure 5.1: Impact of camera-object distance for the corresponding error on the image

In addition to the impact of the distance training point to camera on the calibration quality, the distance between training points might also impact the calibration accuracy. For a given number of training points, the density of points along the three dimensions available for the path can be varied. This change in densities might result in different calibration qualities, i.e., for a given camera, there might be an optimum step length that ensures an efficient sampling along the three directions. Since the 3D coordinates' measurements have uncertainties, once could assume that the larger the distance between two consecutive waypoints is, the less the uncertainty matters. Also, the closer two waypoints are, the larger are the risks to make correspondence errors between 3D points and their images.

Experiment 3 will investigate whether sampling step-length impacts the calibration quality.

Assumption: For a constant number of sampled points, the changes in the density of points along the three dimensions available for the path likely impact the calibration quality

The following paragraph summarizes the observation and assumption of this section.

The uncertainty resulting from GPS and image detection uncertainties must be evaluated to validate the drone-based calibration approach. The GPS measurement uncertainty impacts the calibration accuracy differently depending on the 3D training points localization. Also, the choice of sampling densities along the axes X, Y, and Z likely impacts the calibration quality. Thus, the choice of the path parameters is important to allow for a successful calibration, and there is a need for a guideline for the choice of path parameters to ensure a successful calibration for the large variety of cameras tested. This leads to the following research question:

> Research Question 3: How to pick the path parameters to enable a successful calibration of a broad range of cameras?

The calibration technique must be evaluated for thousands of camera models to demonstrate the scalability of the drone-based calibration solution. Due to the large number of tested cameras, the calibration technique is evaluated using simulations. The goal of these simulations is to model different cameras, the 3D training points, and their corresponding images on the camera. In addition, they should also model the 2D and 3D localization uncertainties.

Measurement error can be divided into two categories: systematic error, which is due to sensor calibration error and can be corrected, and random error, which is due to unpredictable fluctuations in environmental conditions or observer judgment error. Gaussian distributions of mean zeros can model the latter. The image localization uncertainty is caused by random events such as oscillations at the sampled points or shadows that make the marker detection less precise. Since it is caused by random noise, the 2D measurement uncertainty can be modeled by a gaussian distribution. GPS measurement error is also due to random phenomena such as signal blockage and is also modeled by a gaussian distribution. Due to the random behavior of the uncertainties, taking a unique sample to evaluate the calibration would not lead to a reliable metric. Instead, a Monte Carlo sampling enables to capture the impact of the input uncertainties on the calibration quality [149]. Indeed, it enables the evaluation of the reprojection error statistics, such as the estimates of the average error, the maximum error, or the uncertainty in measurements.

For a given camera, a set of 3D ”perfect” (without noise) training points is defined. The image 2D points are computed using the perfect 3D points and the camera model. This set of points is fed into a Monte Carlo sampling that draws k times a set of 3D and 2D measurement noises following Gaussian distributions. The 3D and 2D key points are modified by adding noise to each of them. This leads to a set of 3D and 2D key points that captures the measurement uncertainties. For each modified set of points, the optimization for camera calibration is run, and the error is stored. This process is repeated k times, which results in a statistical distribution for the calibration error that captures the measurement uncertainties. From this distribution, the estimate of the average error can be used for a reliable evaluation of the calibration approach. Once a reliable calibration error is computed, it is compared to the threshold defined in Chapter 3 to validate or reject the calibration methodology for the considered camera. Figure 5.2 illustrates this approach.

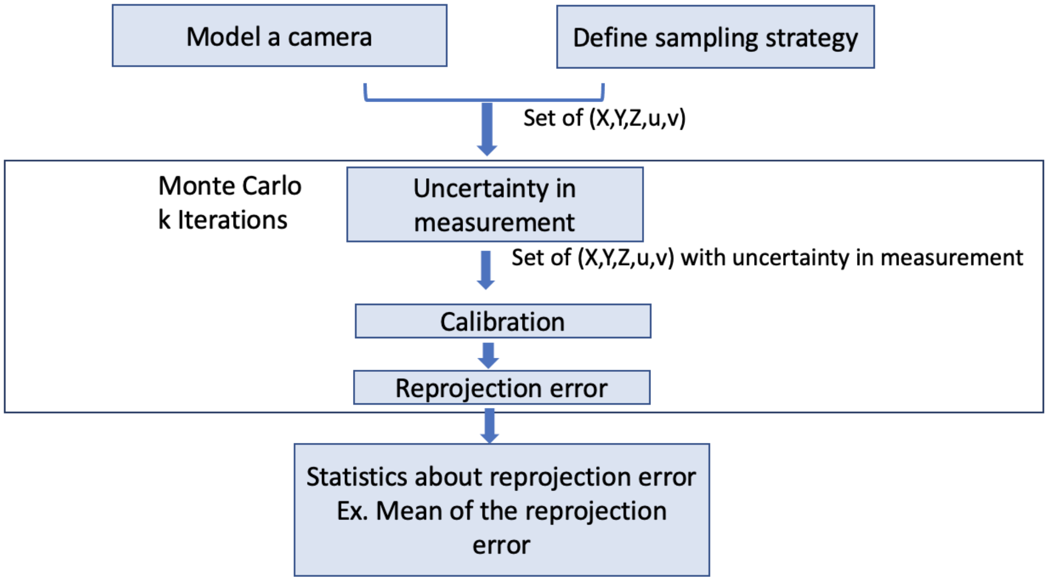

Model a camera
Define sampling strategy
Set of (X,Y,Z,u,v)
Monte Carlo
k Iterations
Uncertainty in measurement
Set of (X,Y,Z,u,v) with uncertainty in measurement
Calibration
Reprojection error
Statistics about reprojection error
Ex. Mean of the reprojection error

Figure 5.2: Monte Carlo sampling for a reliable evaluation of the calibration under uncertainties

The methodology described in the previous paragraph is followed for each tested camera to evaluate the scalability of the drone-based calibration technique. Testing the calibration approach for a broad range of camera models would also enable highlighting the influence of camera parameters on the calibration accuracy.

A method to define and model the large range of cameras must be defined. The modeling of a camera in the simulation introduced in this section relies on the input of camera parameters. Once the intrinsic and extrinsic parameters have been defined, the relation that relates world coordinates to image ones, as derived in chapter 2, is implemented to model the camera and its image points. So, modeling different cameras is equivalent to defining different sets of camera parameters. Bounds for intrinsic and extrinsic parameters are defined after searching for camera specifications. Table 5.1 summarizes the parameters which are varied to model diverse cameras. Figure 5.3 illustrates the impact of changes in the focal length and image dimension (the image width is related to the image height by an aspect ratio of $\frac{4}{3}$) on the field of view angle. Figure 5.4 illustrates the impact of changes in the elevation and the orientation along Y on the field of view orientation. In addition to the parameters introduced by Table 5.1, the distortion is varied between two levels: no distortion, and moderate distortion following the Brown equations [150].

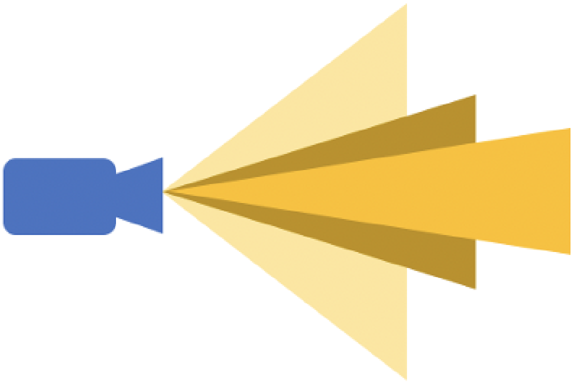


Figure 5.3: Variation of focal length and image width leads to variation of FOV

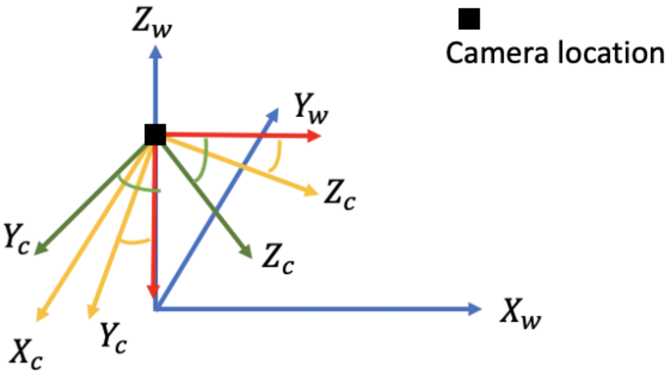


Figure 5.4: Variation of camera orientation and elevation

Table 5.1: Camera parameters design space

Parameter	focal length	image width	elevation	orientation	Distortion level
Lower bound	2mm	700 pixels	2m	0 degree (horizontal)	pinhole
Upper bound	15mm	4000 pixels	5m	90 degree (vertical)	Medium distortion using Brown model

The simulation is used to model thousands cameras by changing the focal length, the image dimensions, the distortion, the camera elevation, and orientation. The following paragraph describes the value selection for these five factors for the modeling of the many cameras.

Design of Experiments (DOE) techniques are sampling approaches that select points based on their distribution throughout the design space to optimize the knowledge provided by these sample points [151]. DoE techniques would enable the selection of values for the focal length, image width, camera orientation, and elevation so that thousands different cameras can be modeled in the simulations. The selection of an appropriate DoE technique depends on the type of input data collected. The set of varied parameters has two characteristics listed below that restrict the choice of a DoE.

- All parameters are varied in a continuous way, except for the distortion, which is varied between two levels. Thus, the DoE technique must handle continuous and discrete factors. Random sampling, Optimal design, Latin Hypercube, and space-filling are potential candidates for such a DoE [151].

- Since few cameras have a field of view angle equal to 180 or 20 degrees, a small ratio of points should be sampled at the design space extremities while the design space interior should be highly represented. Among the previously listed DoE techniques, Latin Hypercube is the one filling this criterion [152].

Thus, a Latin Hypercube design is selected to sample the design space made of the five factors: focal length, image width, camera elevation, orientation, and distortion level, and create thousands cameras. Then, for each camera, the methodology described in Figure 5.2 is run and, for a given path, the accuracy of the calibration is evaluated in a reliable way.

Once the cameras are modeled within the simulation, the Monte Carlo sampling approach described earlier in the section is applied for each camera.

At the beginning of this section, it was observed that the location of the 3D training points impacts the final calibration accuracy. Thus, there is a need to develop and apply a methodology to select the appropriate path parameters to ensure a successful calibration. This need is stated through Research Question 3: How to pick the path parameters to enable a successful calibration of a broad range of cameras?

The elements to answer the questions have been introduced in this section and are listed below:

- Thousands cameras are modeled by varying their parameters and selecting adequate focal length, image width, distortion, camera orientation, and elevation using a Latin Hypercube DOE

- For each camera and for a given path, the calibration quality is evaluated with a Monte Carlo sampling that captures the 3D and 2D measurement uncertainties. This enables a robust evaluation of the calibration quality for the thousands cameras while accounting for the uncertainties.

- Finally, the Monte Carlo sampling can be operated for each camera and different sampling strategies. Path parameters can be varied, leading to different 3D key points. Then for the latter key points and a given camera, the steps from Figure 5.2

are run. This leads to a reliable evaluation of the calibration quality for a given path and a given camera.

The second step listed allows to evaluate the calibration quality for a broad range of cameras and thus evaluate the scalability of the drone-based calibration approach. The last step listed above enables a comparison of path parameters' impact on the calibration accuracy. It is assumed that the combination of both steps would provide guidance for the choice of the trajectory to ensure a successful calibration and thus would answer Research Question 3. This assumption must be tested because the resulting calibration error might not satisfy the threshold defined in Chapter 3. Also, there might not be an optimum sampling that would lead to a sufficient calibration accuracy for the large range of cameras. Thus, this leads to the following hypothesis, whose test would answer Research Question 3.

Hypothesis 3: IF different sampling strategies with various step lengths and distances from the camera are tested in simulations where uncertainties are modeled with a Monte Carlo simulation and cameras are defined using a Latin Hypercube DOE, THEN a sufficient sampling strategy can be defined applicable to a large range of cameras

Figure 5.5 illustrates the methodology described in the section and the experiment designed to test Hypothesis 3.

The chart on Figure 5.6 summarizes the observations, question and hypothesis introduced in this section. Hypothesis 3 is validated if the approach described on Figure 5.5 allows the definition of sampling strategies whose corresponding calibration errors are smaller than the threshold defined in Chapter 3. The experiment, which test Hypothesis 3, evaluates whether the drone-based calibration approach allows a successful calibration for a large variety of cameras despite the 3D and 2D measurement errors. To do so, it tests different sampling strategies to find a sufficient one that allows a calibration error

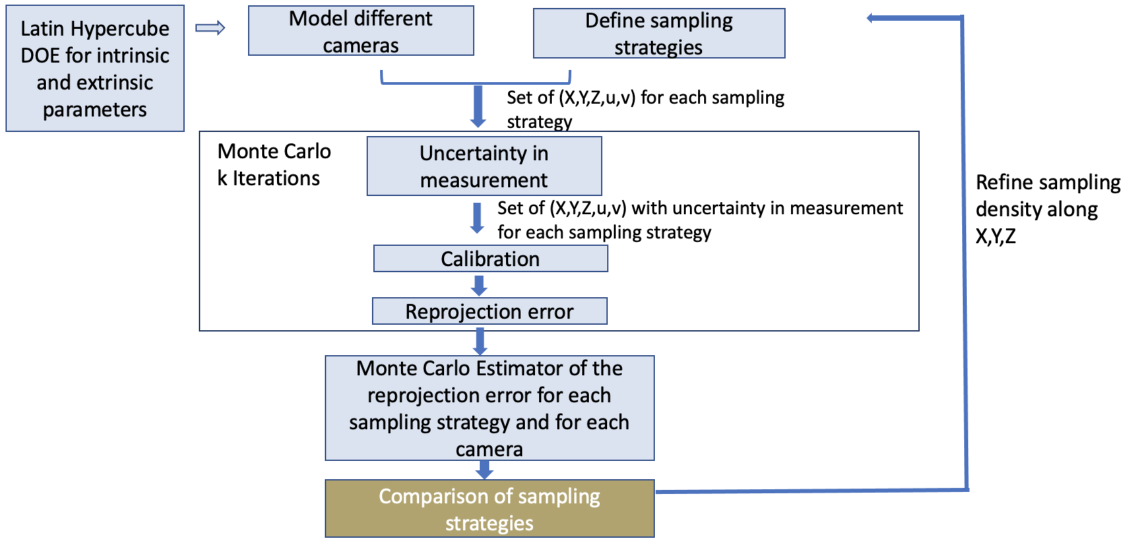
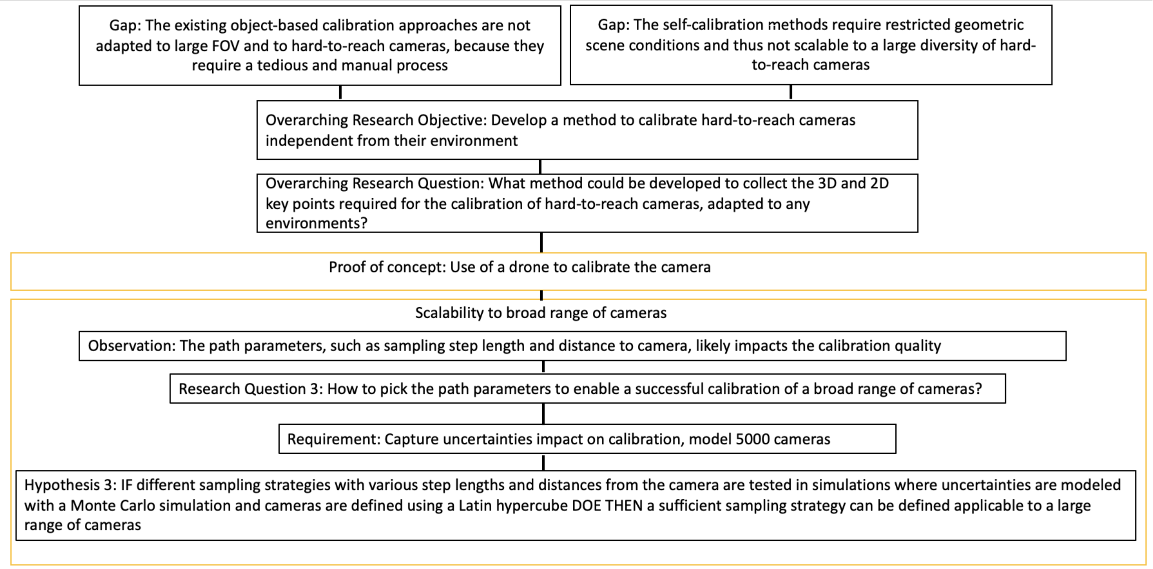

Latin Hypercube DOE for intrinsic and extrinsic parameters
Model different cameras
Define sampling strategies
Set of (X,Y,Z,u,v) for each sampling strategy
Monte Carlo k Iterations
Uncertainty in measurement
Set of (X,Y,Z,u,v) with uncertainty in measurement for each sampling strategy
Calibration
Reprojection error
Monte Carlo Estimator of the reprojection error for each sampling strategy and for each camera
Comparison of sampling strategies
Refine sampling density along X,Y,Z

Figure 5.5: Methodology designed to evaluate the calibration quality for a large range of cameras and for different sampling paths

small enough for speed measurement applications. The experiment is described in the next section.



Gap: The existing object-based calibration approaches are not adapted to large FOV and to hard-to-reach cameras, because they require a tedious and manual process

Gap: The self-calibration methods require restricted geometric scene conditions and thus not scalable to a large diversity of hard-to-reach cameras

Overarching Research Objective: Develop a method to calibrate hard-to-reach cameras independent from their environment

Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?

Proof of concept: Use of a drone to calibrate the camera

Scalability to broad range of cameras

Observation: The path parameters, such as sampling step length and distance to camera, likely impacts the calibration quality

Research Question 3: How to pick the path parameters to enable a successful calibration of a broad range of cameras?

Requirement: Capture uncertainties impact on calibration, model 5000 cameras

Hypothesis 3: IF different sampling strategies with various step lengths and distances from the camera are tested in simulations where uncertainties are modeled with a Monte Carlo simulation and cameras are defined using a Latin hypercube DOE THEN a sufficient sampling strategy can be defined applicable to a large range of cameras

Figure 5.6: Summary of observations, research question and hypothesis to evaluate whether the approach is scalable to a large variety of cameras

5.2 Experiment 3

Experiments 1 and 2 assumed the use of a localization system that enables very accurate 3D coordinates' measurements. The oscillations of the drone at waypoints were included in the simulations and are responsible for small 3D localization errors. These oscillations cause a slight difference between the recorded and the actual 3D path followed by the drone in the simulation. The first contribution of experiment 3 is the evaluation of the calibration quality despite GPS and altimeter noises. In Experiments 1 and 2, the calibration approach was tested on two types of cameras. The second contribution of Experiment 3 is to evaluate the scalability of the approach for thousands of cameras. The third contribution of Experiment 3 is a methodology and its application to select path parameters that ensure a successful calibration. Indeed, it was observed that the impact of 3D measurement noise on the calibration quality depends on the 3D points' position. It was also noticed that the step length along the three dimensions available for the path likely impacts the calibration quality.

Due to the large number of tested cameras, Experiment 3 is implemented using simulations. Because using ROS and Gazebo for the 5000 cameras, and the many sampling strategies, would take a long time, another simulation framework is built to reproduce the results of Gazebo and ROS simulations while running faster. These simulations are implemented in python 3, and they allow to model a camera using its parameters as input, to model a 3D trajectory, and its image captured by the camera. In addition, these simulations run a calibration of the simulated camera using the set of 3D and 2D points with noise and return reprojection errors and their distribution.

The RTK measurement uncertainty is modeled using a Gaussian distribution of mean zero and standard deviation 1.5cm along the vertical axis (Z axis). The measurement uncertainty along X and Y that forms the ground plane is modeled by a Gaussian distribution of mean zero and standard deviation 1cm along each axis [153]. The 2D localization uncertainty is not known and must be modeled to implement simulations that are a correct representation of the real world. This leads to the following gap:

Gap: The model of the 2D detection uncertainty must be estimated to build simulations that enable the evaluation of the calibration approach scalability.

Since it is caused by random noise, the 2D measurement uncertainty can be modeled by a Gaussian distribution of mean zero and standard deviation s, along the horizontal and vertical axes of the camera. The standard deviation s must be determined. With the available data from the simulations, it is possible to estimate the standard deviation s. The following section presents the methodology followed to determine s.

5.3 Model for image detection error

This section explains how results from Experiment 2 enable to find a model for the error in the selection of the pixel on the image that corresponds to the target.

The notations used in this section are introduced below:

- P_0 is the set of 3D training points
- S_0 is the set of pixel coordinates of the images of these 3D training points, i.e. it is the set of 2D training points
- S_1 is the set of 2D training points that were detected by the marker detection algorithm. If there were no noise, S_1 would be S_0. The points in S_1 are the images of the points in P_0, their location is not exact compared to the points in S_0 due to detection error. Also, some points in P_0 are not detected on the image. Thus S_1 has less points that S_0
- P_1 is the set of 3D training points whose images are in S_1

The set of equations used to model the camera in Gazebo is known [154] and is implemented in the python-based simulations. So the newly implemented simulations, and the Gazebo simulations, can model the same cameras.

For each marker location detected in the Gazebo simulation, the actual ground truth can be computed using the known 3D coordinates, the set of equations that model the camera, and the known camera parameters. This ground truth can be used to get the measurement error distribution once a marker is detected. However, the error in the exact localization of the marker is not the only noise involved in the selection of the target on the image. Figure 5.8 illustrates this observation. This step is impacted by the correct identification of the marker on the image. Indeed, there is noise due to lighting, oscillations, and other environmental factors, as is illustrated in the orange blocks on the right chart of Figure 5.7. This noise introduces errors in the image point detection: error in localization and classification error (some marker positions are not detected). The python-based simulations must account for this noise, and this is illustrated with the left orange block on Figure 5.7.

Figure 5.7 summarizes the two simulations steps. ”Simu 1” refers to the Pyhton-based simulations, and ”Simu 2” to the ROS and Gazebo-based simulations.

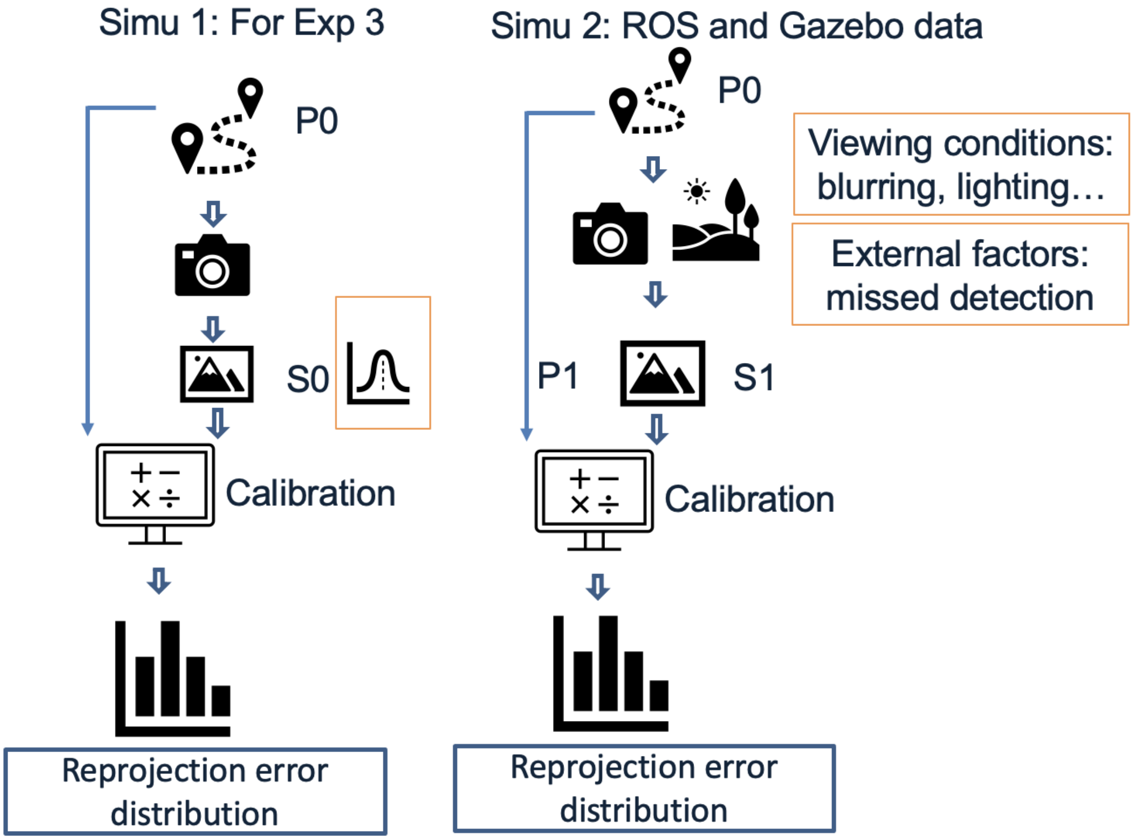
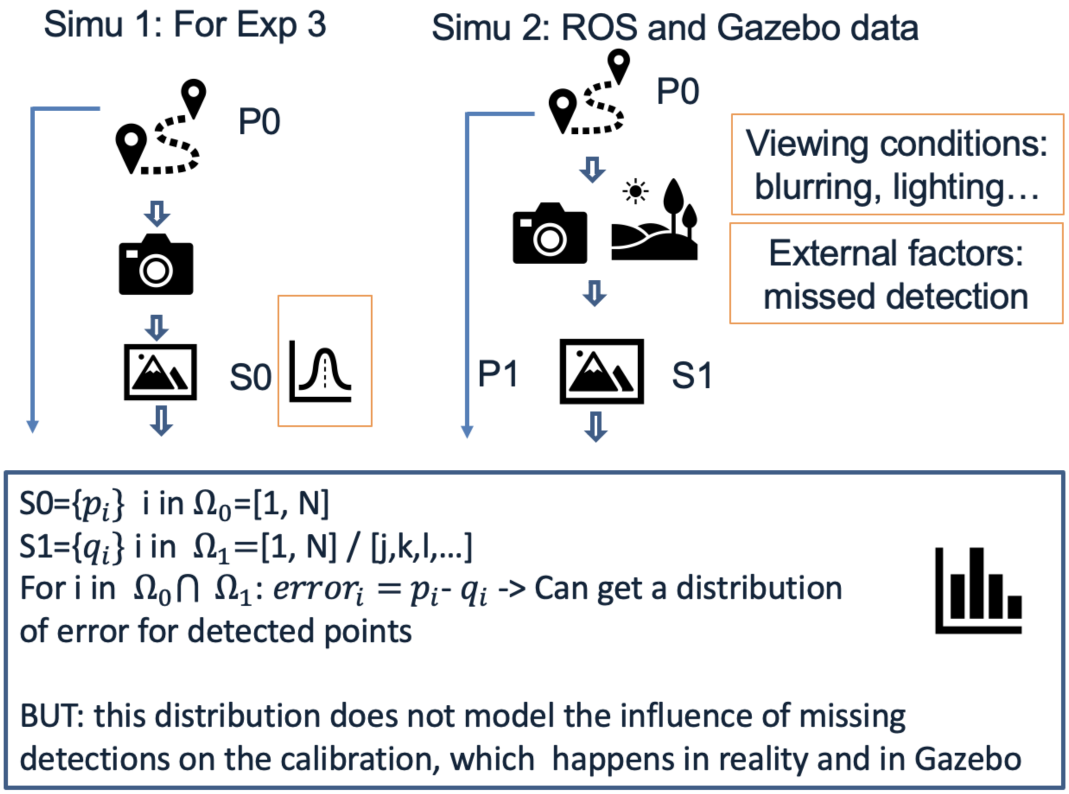

Simu 1: For Exp 3
P0
S0
Calibration
Reprojection error distribution
Simu 2: ROS and Gazebo data
P0
Viewing conditions: blurring, lighting…
External factors: missed detection
P1
S1
Calibration
Reprojection error distribution

Figure 5.7: Left image: Python-based simulations. Right image: ROS and Gazebo simulations

The comparison between detected image coordinates and actual image coordinates leads to a distribution for the image localization error. Figure 5.8 illustrates this process. However, this error distribution does not account for the influence of missed detection (marker location not detected due to lighting, clutter surroundings, etc.) on the calibration. The difference between S_0, and S_1 is also caused by marker that were confused with environment objects [144]. This noise is unknown and there are no ground truth available for it, as illustrated in Figure 5.8. Thus, a model is assumed for it. The goal is that the Python-based simulations reproduce accurately the results of the ROS and Gazebo-based simulations. So, the model is evaluated by comparing Gazebo simulation results and Python simulation results.


Simu 1: For Exp 3
P0
S0
Simu 2: ROS and Gazebo data
P0
Viewing conditions: blurring, lighting…
External factors: missed detection
P1
S1

S0=$\{p_i\}$ i in Ω_0=[1, N]
S1=$\{q_i\}$ i in Ω_1=[1, N] / [j,k,l,…]
For i in $\Omega_0 \cap \Omega_1$: $error_i = p_i$- q_i -> Can get a distribution of error for detected points

BUT: this distribution does not model the influence of missing detections on the calibration, which happens in reality and in Gazebo

Figure 5.8: Model localization error: difference between S_0 and S_1

There is a need for a detection uncertainty model that accounts for localization error and classification error. For the localization error, the available ground truth is the reprojection error obtained after the complete drone-based calibration process implemented in Gazebo. The proposed idea is to find a model for the noise on image points that would account for both localization and classification errors. The noise is added to the perfect set of image point S_0, and its impact on the calibration process must be equivalent to the impact of noise simulated in Gazebo (lighting, cluttering, oscillations...). The set S_0 + noise must be "equivalent" to the set S_1. Equivalent means here that the calibration outputs are similar. For this purpose, the reprojection error (calibration error) distributions obtained in Experiment 2 are used to evaluate the standard deviation s. Figure 5.9 illustrates the process followed to find a model for the pixel measurement uncertainty, using the available data from Experiment 2.

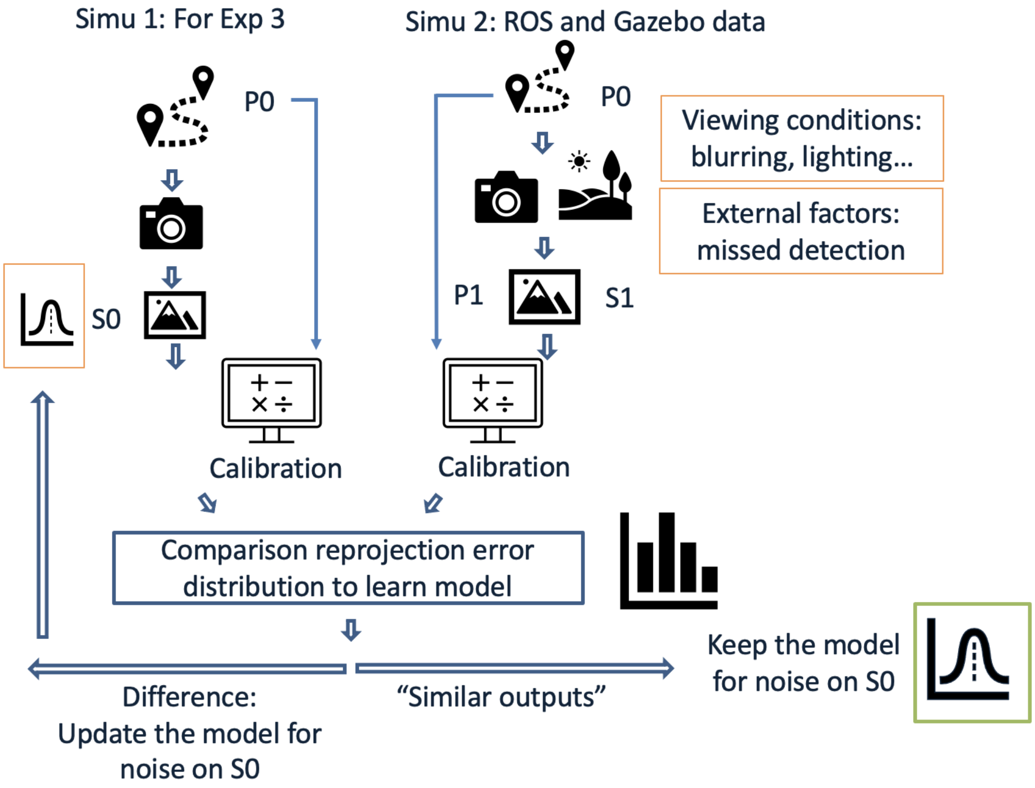

Simu 1: For Exp 3
Simu 2: ROS and Gazebo data
P0
P0
Viewing conditions: blurring, lighting...
External factors: missed detection
S0
P1
S1
Calibration
Calibration
Comparison reprojection error distribution to learn model
Keep the model for noise on S0
Difference: Update the model for noise on S0
"Similar outputs"

Figure 5.9: Model error when selection of 2D key point on the image

The new simulation and the Gazebo simulation are run for the same camera and the same path. The python-based simulation assumes a Gaussian noise of mean zero and standard deviation s for the image detection uncertainty. The optimization for camera calibration is run from the two sets of 2D and 3D points obtained through the Gazebo and the python simulations. The reprojection error is computed for each training point, leading to two distributions of reprojection errors. The first distribution is built using the reprojection error values obtained after the calibration of the camera simulated with the Gazebo simulation. The second distribution is built using the reprojection error values obtained after the calibration of the camera simulated in the python simulations. For a standard deviation value s, the reprojection distribution obtained from the python simulation and the one obtained from Gazebo are compared. The comparison is based on shape comparison and hypothesis testing. If the distributions are different, a new value for s is tested. Figure 5.10 illustrates the process followed to validate or reject the model tested.

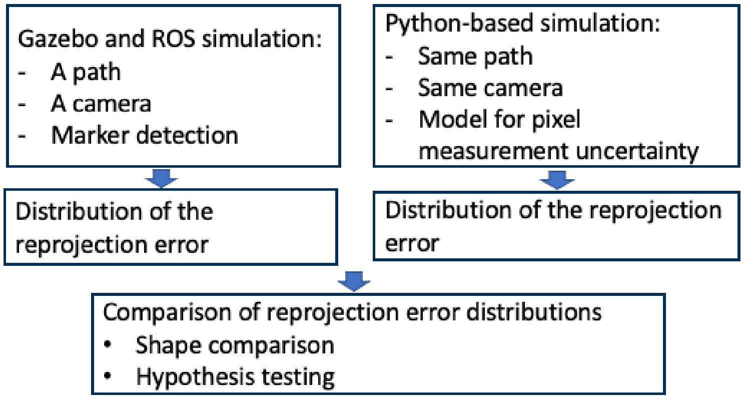

Gazebo and ROS simulation:
- A path
- A camera
- Marker detection
Python-based simulation:
- Same path
- Same camera
- Model for pixel measurement uncertainty
Distribution of the reprojection error
Distribution of the reprojection error
Comparison of reprojection error distributions
• Shape comparison
• Hypothesis testing

Figure 5.10: Validation process for the model for image detection noise

For a standard deviation value equal to 2 pixels, the distribution of reprojection errors obtained after the Gazebo simulation and the python simulation have similar shapes: they can bot be approximated as Gaussian, as seen in Figure 5.11. They also have similar box-plot representations, as seen in Figure 5.12. The two left box-plots are the reprojection error obtained for the simulated pinhole camera in Gazebo and in python. The two right box-plots are obtained for the simulated camera with distortion in Gazebo and in python. The medians are equal for pinhole (resp. distorted) cameras box-plots, and the allocation of points above and below the median are similar. In addition to the shape comparison, a T-Test is run to compare the mean of the reprojection error distributions obtained for the pinhole camera calibration in the two simulations. The Null Hypothesis states that the means of both distributions are equal. The p-value obtained from this t-test is equal to 0.64, which is larger than the p-value threshold of 0.1 [155]. Thus, there is strong evidence towards the Null Hypothesis. When combined with the qualitative shape comparison, this enables the validation of the 2D points detection uncertainty model for its use in Experiment 3. Figure 5.13 illustrates the distribution taken for the noise added to pixel coordinates in the u direction. The same distribution is used for the noise in the v direction.

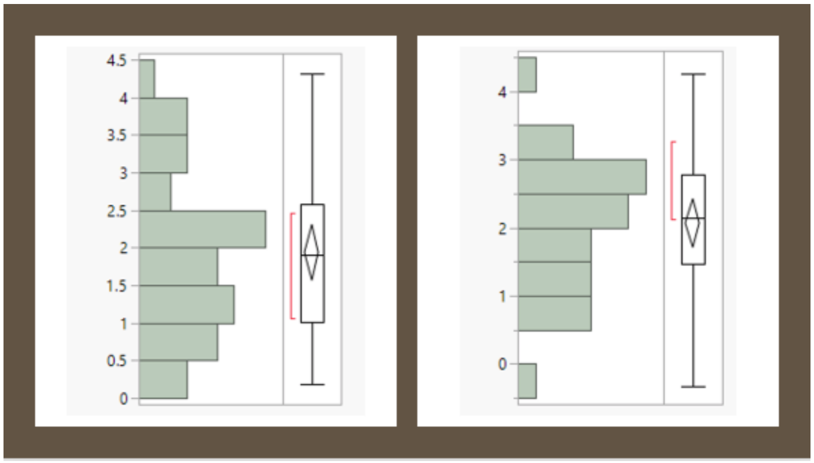


Figure 5.11: Shape comparison of reprojection error distributions: right distribution obtained with Gazebo simulations, left distribution obtained with python simulations. The y-axis is the RMS reprojection error value in pixels

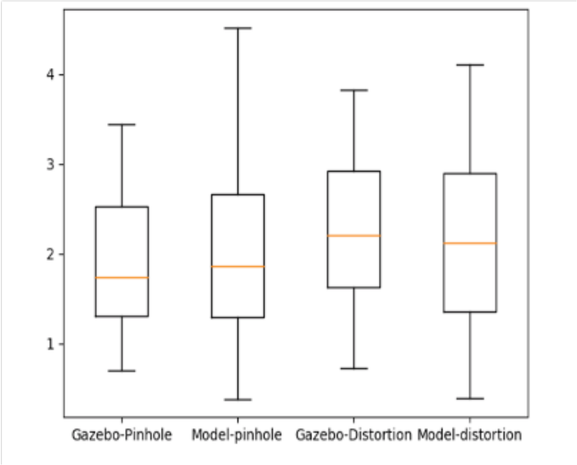


Figure 5.12: Qualitative comparison of box-plot distributions. The y-axis is the RMS reprojection error value in pixels

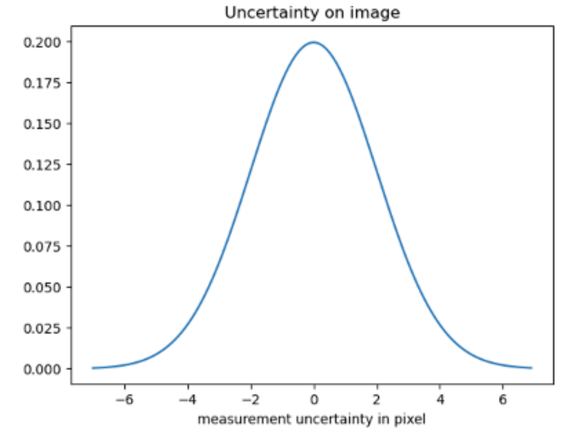


Figure 5.13: Model for noise in u (resp. v) direction

Result: The uncertainty for the identification and localization of the 2D points is modeled by a Gaussian of mean zero and standard deviation 2 pixels

5.4 Experiment 3: Implementation

Once the noise in 2D and 3D measurements can be modeled, the process described in Chapter 3 and recalled in Figure 5.14 is implemented for different sampling strategies. The goal is to find a sufficient sampling strategy applicable to the tested cameras. The optimum sampling strategy might depends on camera parameters. It is assumed that the the user of this calibration approach has a vague a priori knowledge of the camera parameters. For instance, one can guess the range to which belongs the FOV angle. If camera parameters impact the calibration quality, the user can adapt the trajectory to ensure a good calibration quality, using the camera parameters guess. The result of Experiment 3 would serve as a guideline for the choice of path parameters to ensure an accurate calibration for a large variety of camera parameters.

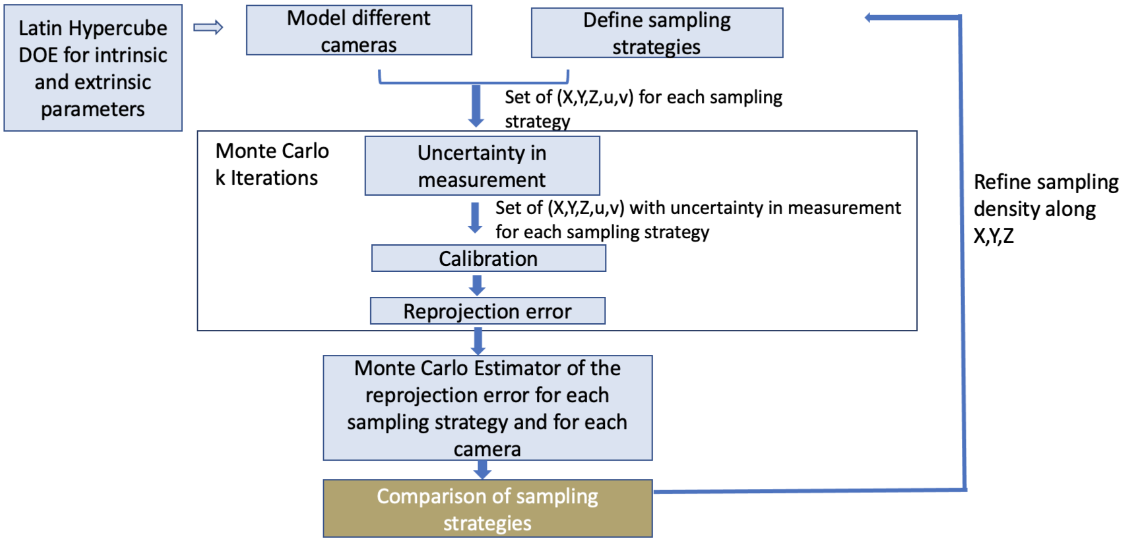

Latin Hypercube DOE for intrinsic and extrinsic parameters
Model different cameras
Define sampling strategies
Set of (X,Y,Z,u,v) for each sampling strategy
Monte Carlo k Iterations
Uncertainty in measurement
Set of (X,Y,Z,u,v) with uncertainty in measurement for each sampling strategy
Calibration
Reprojection error
Monte Carlo Estimator of the reprojection error for each sampling strategy and for each camera
Comparison of sampling strategies
Refine sampling density along X,Y,Z

Figure 5.14: Methodology designed to evaluate the calibration quality for a large range of cameras and for different sampling paths

The following pseudo-code presents the Monte-Carlo sampling, which was illustrated as a black-box on Figure 5.14. The parameter n is the number of pairs of 3D points P_i, and their image p_i. The noise on the 3D points and 2D points are assumed Gaussian with zero mean and covariance $S_i * Id_3$, and $s_i * Id_2$. Where S_i is the vector $\begin{bmatrix} 1 \\ 1 \\ 2.25 \end{bmatrix}$, and s_i is the vector $\begin{bmatrix} 4 \\ 4 \end{bmatrix}$ in this work.

Algorithm 1 Pseudo-code

```
Repeat k times:
    For i in [1,n]:
        Draw $Q_i$ from a normal distribution $N(P_i, S_i Id_3)$
        Draw $q_i$ from a normal distribution $N(p_i, s_i Id_2)$
    Estimate camera parameters C using the selected calibration method
    Compute and store reprojection error (or RMS reprojection error) $E(C)$ as $E_k(C)$
Compute statistics on the set of measurements $E_k(C)$: mean, max, variance...
```

The choice of k was set to 500 after observing the convergence of the Monte Carlo estimator for various cameras. Starting from 500 iterations, the amplitude of the oscillations of the estimator is decreased by 90 percent compared to the amplitudes for a small number of iterations. Figure 5.15 illustrates this trend for a camera with focal length of 1,000 pixels and image width and height of 1,200 pixels. Beyond this number of iterations, the gain in precision of the estimator was not worth the additional computation time.

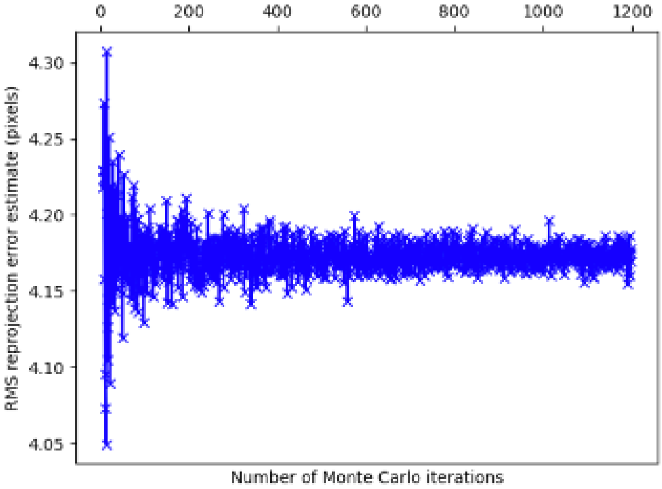


Figure 5.15: Convergence of the RMS reprojection error estimate with the number of Monte Carlo iterations

Because the simulations for 5000 cameras take several hours to run, the strategy was first applied to a reduced number of simulated cameras. Testing for 30 cameras were run in order to gain a first knowledge about which sampling step lengths, and which distance between 3D points and camera should be tested for the 5000 cameras.

5.5 Experiment 3: Pre-experiment for 30 cameras

First, a Latin Hypercube DOE is run to define 30 design points, in the design space made of the focal length, the image width, the distortion level, the camera orientation and elevation. The baseline sampling for obtaining the 3D key points is inspired by the path defined in the first experiment while adding more points along Z (axis vertical up). It is made of points every meter along X, and Y, where (X,Y) represents the ground plane, and every 50cm along Z. Figure 5.16 illustrates this baseline sampling.

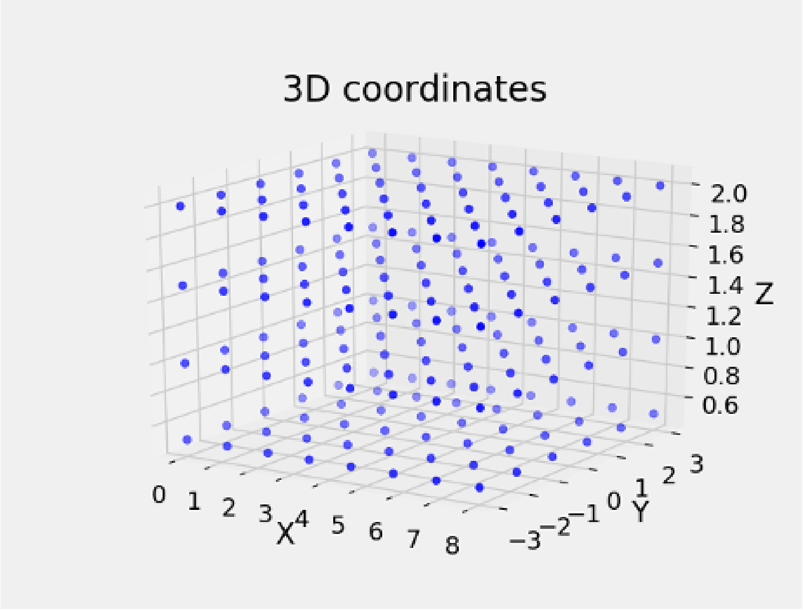


Figure 5.16: Baseline Sampling: the improvement in the calibration accuracy are measured against this sampling

The main takeaway of this pre-experiment is the evaluation of the influence of the sampling density along Z on the calibration quality. It was observed that changing the sampling density along Z from 2 points to 4 points, while sampling between 0.5 meters and 2 meters, did not significantly change the calibration error for the 30 tested cameras. Figure 5.17 illustrates this result: The abscissa represents the identification number of the tested cameras. For each camera, the blue histogram magnitude is the reprojection error measured when sampling two Z-values between 0.5 meters and 2 meters. The green histogram magnitude is the reprojection error when sampling 3 Z-values (0.5m, 1.25m, 2m), and the red histogram magnitude is the reprojection error obtained when sampling 4 Z-values (50 cm, 1m, 1.5m, 2m). Thus, the density of sampled points along Z does not significantly impact the calibration quality.

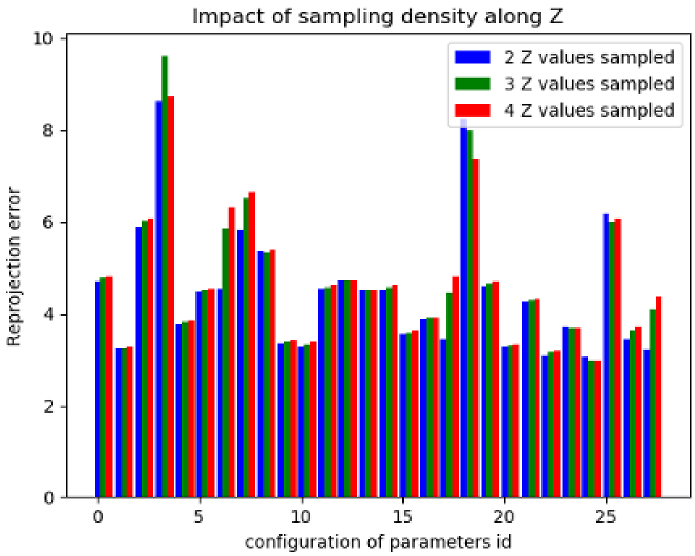


Figure 5.17: Reprojection error changes with change in number of sampled points along Z

While running this pre-experiment, it was observed that changing the sampling density along Y and X axes impacted the calibration quality. This observation has been further studied with thousands cameras and the results are provided in the next section.

5.6 Experiment 3: Final results

The observations made with 30 cameras where refined using tests on thousands of simulated cameras. The exact number of cameras is picked by tested several candidate values, and selecting the minimum value that ensure a dense coverage of the design space. The selected number of simulated cameras is 5000. Figure 5.18 illustrates the focal length, image width, orientation and elevation values returned by the Latin Hypercube for 5000 design points.

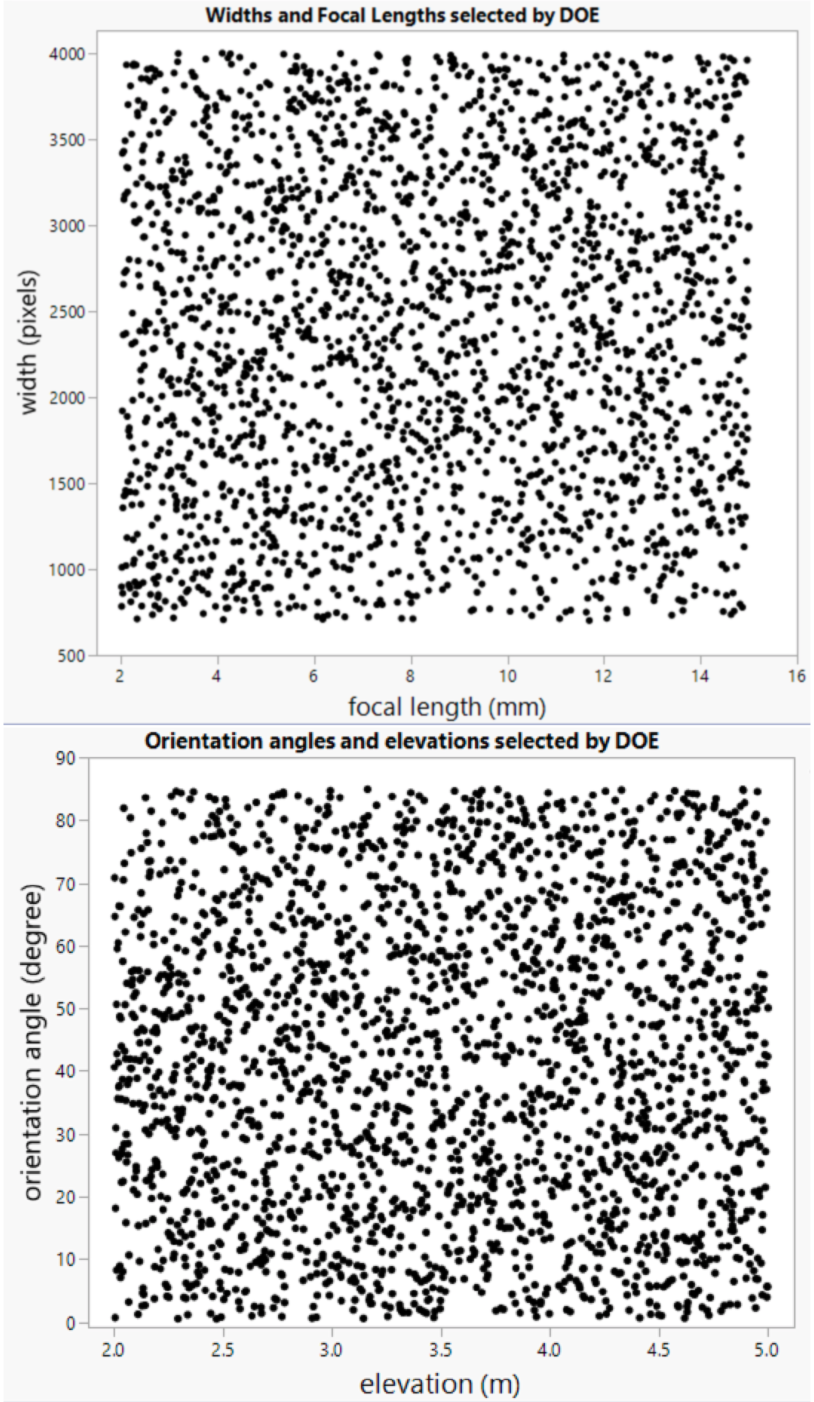


Figure 5.18: Camera parameters selected by DOE

5.6.1 Impact of sampling step-length

First, the results about the impact of density of points along Z from the pre-experiment are confirmed by running the methodology of Figure 5.14 for the 5000 cameras. Results confirmed the trend observed for 30 cameras. Thanks to the large number of tested cameras, a more precise evaluation is possible: 98 percent of camera models do not see an improvement in the calibration quality when the number of sampled points along Z is increased from 2, to 3 or 4 values. Sampling 2 Z values is preferred, compared to sampling 3 or 4 Z values. Indeed, it allows for a correspondence between 3D and 2D points using the changes of direction of the drone. This correspondence method could be used for robustness of the matching 2D-3D, while implemented in addition to an other method, such as hovering at waypoints, or time synchronization.

A second set of simulations compares a sampling step length along Y every 50 cm, with a sampling step length along Y every meter. It was found that using a step length of 1 meter leads to better calibration for 97 percent of cameras. Figure 5.19 illustrates the step-length choice for various focal lengths and image dimensions. On this graph, the contour value equals 1 (blue color) when a step-length of 1 meter enables a better calibration and 2 (red color) if the preferred step length is 50 centimeters. This last step length is advantageous for cameras with small focal lengths or large field of view angles. Figure 5.20 presents the contour plot with the same color code, for orientation angle and elevation changes. However, there are no clear trend seen with this plot. The trend observed when testing a sampling step-length along the X axis of 50cm, and of 1m, are similar to the ones described for the Y axis.

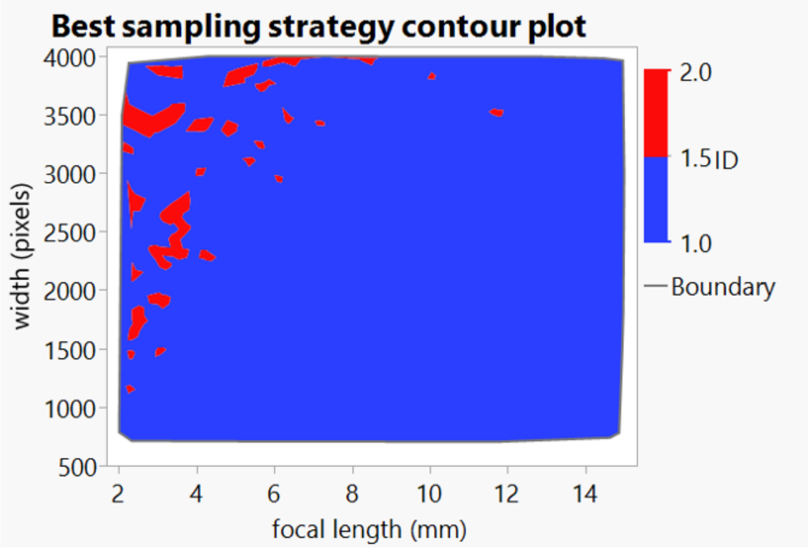


Figure 5.19: Comparison of sampling step-length along Y for various image width and focal length values

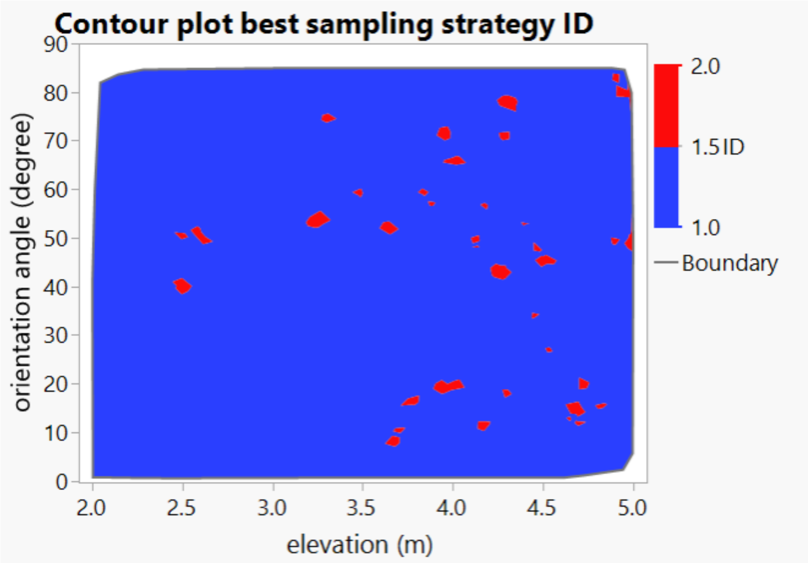


Figure 5.20: Comparison of sampling step-length along Y for various camera elevations and orientations

The main result obtained for the comparison of step-length values is the following:

First result Experiment 3: It is recommended to sample 3D points using a step-length equal to 1 meter along X and Y while sampling two Z-values. An exception exists for cameras with small focal length where a step length of 50 centimeters along Y is preferred.

For this recommended sampling strategy, the resulting RMS reprojection error is provided in Figure 5.21. This representation enables to highlight the impact of camera parameters on the calibration quality for the proposed path. The top figure presents the contour plot of the RMS reprojection error for various image width and focal length values. The calibration accuracy decreases when the image width increases or the focal length increases. The bottom figure depicts the contour plot of the RMS error for various orientations and elevations of the camera. It can be seen that the calibration accuracy is poorer for cameras with small elevation and large orientation angles (near vertical). For these configurations, the drone can only fly close to the camera. This causes the GPS uncertainty to have an important impact. The impact on the calibration of the distance between sampled points and camera is further studied with simulations and results are presented in the next paragraph.

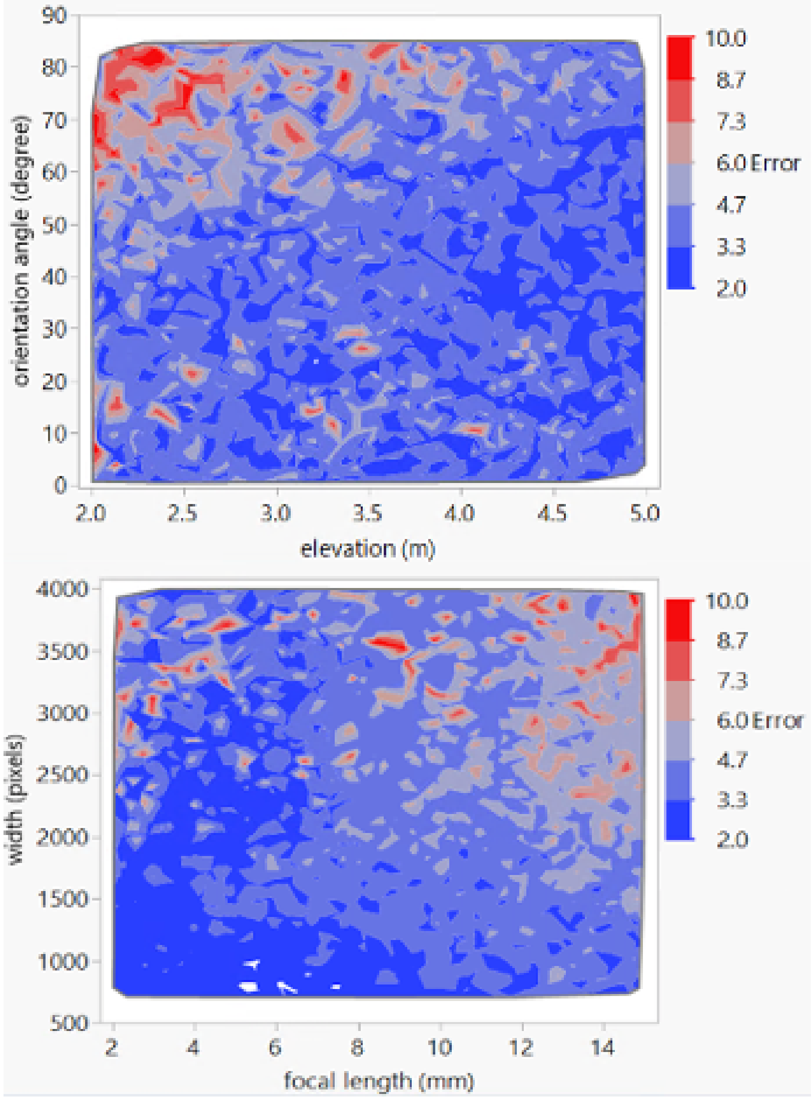


Figure 5.21: Variation of the RMS reprojection error for various camera orientations and elevations on the top and various image width and focal length values on the bottom

5.6.2 Impact of distance from camera

First, a test was made for a unique camera, whose parameters are in Table 5.2. For this camera, simulations were run while modifying the training set of points, such that the coordinates of the farthest points along X, i.e. the perpendicular distance from the camera, is increased. The camera is calibrated in these simulations, and the reprojection error is computed. Figure 5.22 presents the results: The red curve is obtained using 2D point location uncertainty only. The blue curve is obtained when including the GPS uncertainty in addition to the image measurement uncertainty. The impact of GPS uncertainty on the calibration error is represented by the distance between the two curves. It can be seen that when the points are sampled farther from the camera, the impact of GPS uncertainty on the calibration error decreases. This observation leads to the set of simulations described in the next paragraph.

Table 5.2: Camera parameters

focal length	image width	elevation	orientation	Distortion level
1000 pixels	1200 pixels	4 m	45 degree	pinhole

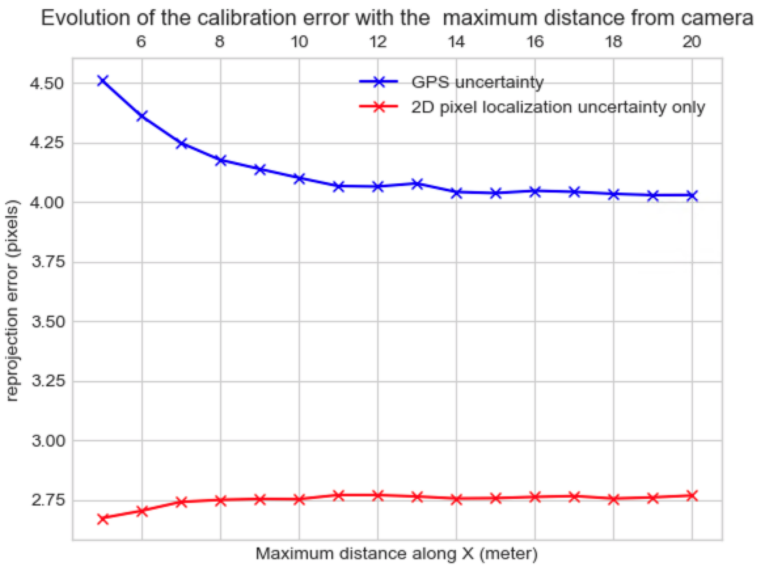


Figure 5.22: Impact on the calibration error of increasing the sampling distance from the camera

The simulations were run to compare a sampling close to the camera to one farther away from the camera. The first simulation used a sampling where values along the X-axis varied between 0 and 8 meters. In a second simulation X-values varied between 0 and 4 meters and the third simulation used X-values between 4 and 8 meters. According to the simulation's results, 79 percent of cameras have better calibration accuracy when using a sampling farther away from the camera. This result is summarized on Figure 5.23. The remaining 21 percent of cameras have common characteristics; they have extreme field of view angle values. Figure 5.24 and Figure 5.25 present contour plots where the red color corresponds to a sampling farther from the camera leads to a better calibration accuracy. The blue color corresponds to the scenario where the preferred sampling is made of points close and far. The grey color corresponds to the preferred sampling is made of points close to the camera. Some of the grey color is also obtained by interpolation in the contour plot, thus the actual number of grey areas is smaller than the one seen on this plot. The cameras who are nearly vertical (orientation angle of 80 degrees and above) have better calibration results when sampling points everywhere, instead of farther or closer to the camera. This result is actually biased, because for these cameras, there are no real notion of close and far since they are vertical and the distance to sample the points is limited by their height.

To summarize the trend observed on these plots, for most cameras, it is recommended to sample points farther from the camera, at a distance larger than 4 meters away to ensure a good calibration in spite of the GPS and altimeter uncertainties. An exception exists for cameras with very large or very small field of view angles. The results also demonstrated that sampling farther away from the camera reduces the average of the RMS reprojection error on all cameras by 38 percent, leading to an average RMS error of 4 pixels.

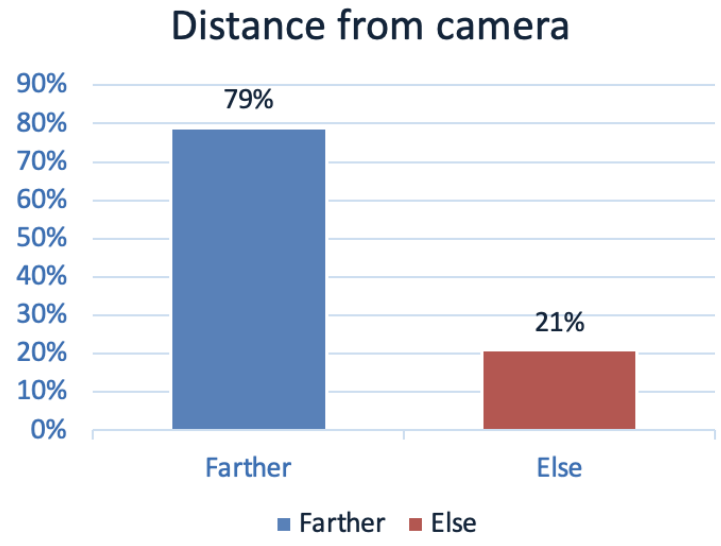


Figure 5.23: Impact of distance drone-camera on calibration accuracy tested on the 5000 simulated cameras

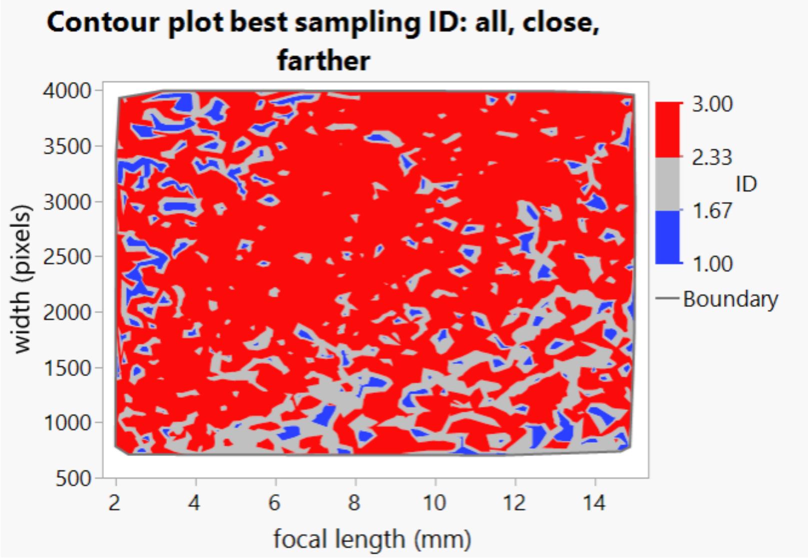


Figure 5.24: Impact on the calibration error of increasing the sampling distance from the camera

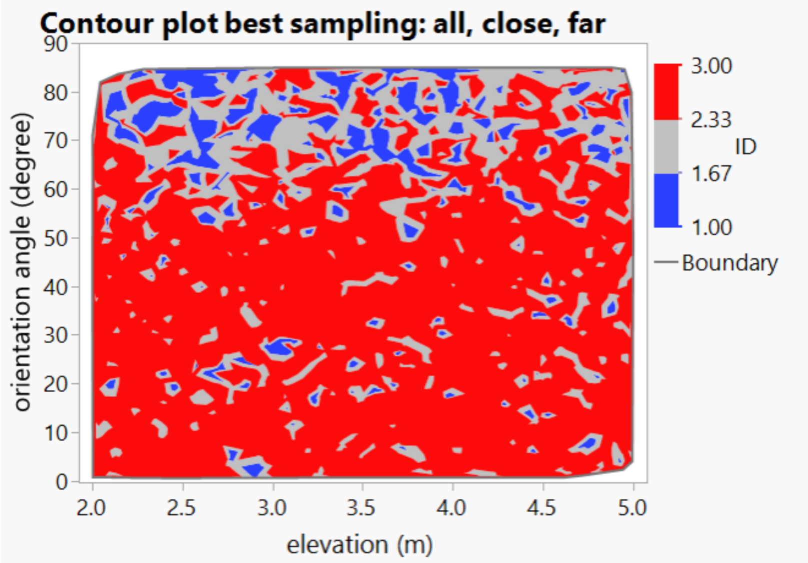


Figure 5.25: Impact on the calibration error of increasing the sampling distance from the camera

The main results of this set of simulations is summarized as follow:

Second Result Experiment 3: For most cameras, it is recommended to sample points farther from the camera, at a distance larger than 4 meters away to ensure a good calibration despite the GPS and altimeter uncertainties. An exception exists for cameras with very large or very small field of view angles.

Third Result Experiment 3: The average RMS reprojection error on all tested cameras is 4 pixels

5.7 Validation of Hypothesis 3

The distribution for the RMS reprojection error computed for the 5000 cameras is provided on Figure 5.26. The RMS reprojection error is an upper bound of the average reprojection error. The blox plot on the right of Figure 5.26 indicates a median of 3.7 pixels, which is very close to the threshold for successful calibration defined in Chapter 3. Also, 75 percent of cameras have a reprojection error smaller than 4.6 pixels, which is in the same order of magnitude compared to the 3.6 pixel threshold defined in Chapter 3. Thus, the drone-based calibration approach enables a successful calibration of the majority of tested cameras. These errors are obtained when the drone flies with a RTK GPS. If a better localisation sensor is used, then the error is reduced as demonstrated with Experiment 1 and 2. In that case, it is expected that the calibration is successful for all tested cameras.

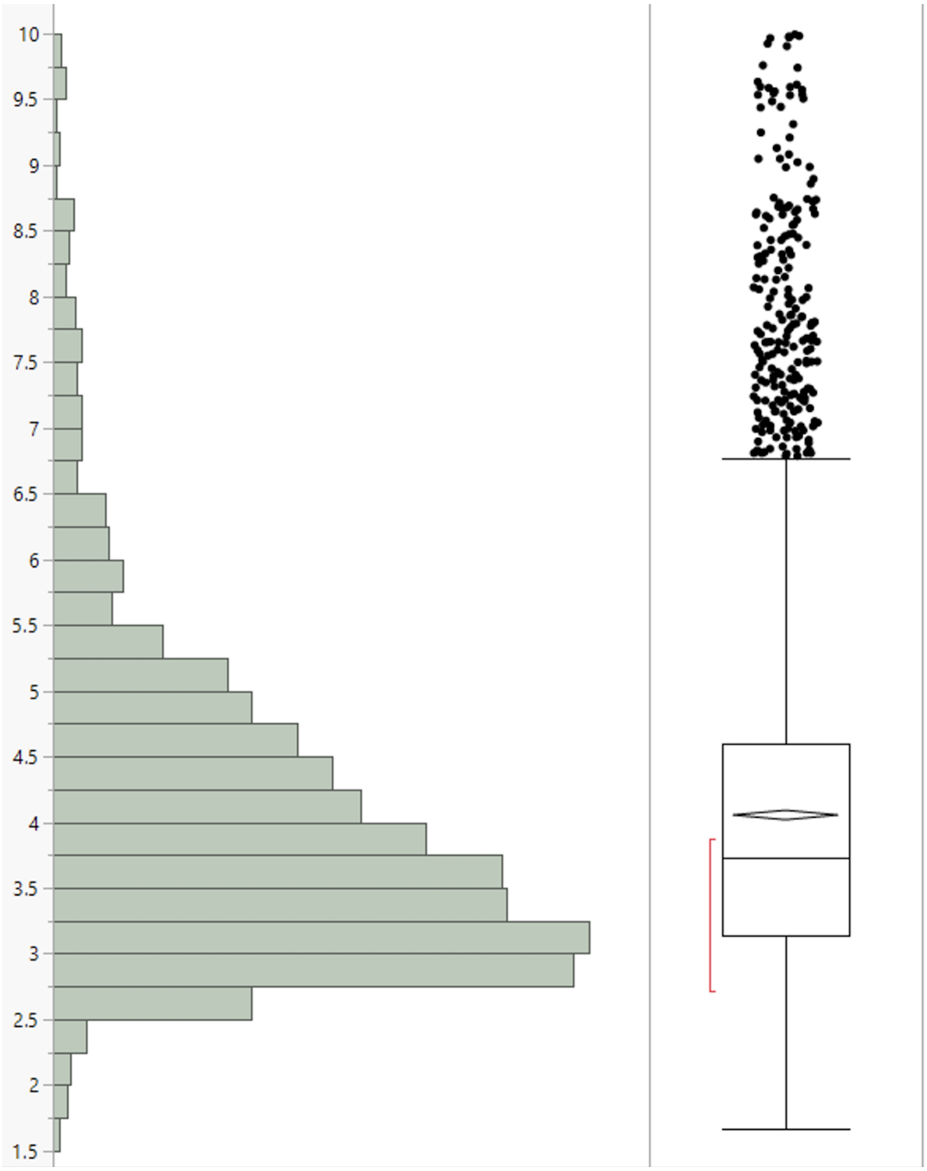


Figure 5.26: RMS reprojection error distribution and box plot for the 5000 cameras

The threshold defined in Chapter 3 to evaluate whether a calibration is successful is equal to 3.6 pixels. The average RMS reprojection error for the 5000 cameras is slightly larger than this threshold and equal 4 pixels, but it is in the same order of magnitude. Also, when going back to the speed measurement application, this reprojection error corresponds to a speed measurement error smaller than 0.5km/h when measuring the speed of a vehicle 15 meters away using a pinhole camera of focal length 900 pixels. It can be concluded that the proposed sampling strategy enables the calibration of cameras within a reasonable accuracy for speed measurement. Based on these observations, the simulations designed with a Monte Carlo sampling to capture uncertainties, and a Latin Hypercube DOE to model many cameras resulted in the definition of path parameters that allow a successful calibration for the majority of cameras tested. Also, this error can be further reduced by sampling points farther away from the camera, since the GPS error impact on the calibration decreases with the distance between sampled points and camera. **Hypothesis 3, which is recalled below, is validated**.

HYPOTHESIS 3 S VALIDATED: IF different sampling strategies with various step lengths and distances from the camera are tested in simulations where uncertainties are modeled with a Monte Carlo simulation and cameras are defined using a Latin Hypercube DOE, THEN a sufficient sampling strategy can be defined applicable to a large range of cameras

Results of Experiment 3 show that the drone-based calibration approach is scalable to a large variety of camera models. Experiment 1 and 2 demonstrated the success of the drone-based calibration approach for the calibration of cameras with large field of views, hard to access and located in challenging backgrounds. The results of these experiments were tested in a real setting. The next Chapter presents the real-world test made with a commercial drone on the Georgia Tech campus of Atlanta.

5.8 Summary

Testing Hypotheses 1, 2, and 3 allowed to demonstrate the success of the drone-based calibration approach for a large variety of cameras. Experiment 1 demonstrated that the drone-based calibration technique enables the collection of a low variance training set, i.e., of a training set that captures the behavior of any application points, and results in an accurate calibration despite the 2D measurement uncertainty. Hypothesis 1 was validated. Experiment 2 tested the feasibility of adding a marker on the drone to enable a more accurate 2D training set and thus a better calibration. Experiment 2 evaluated the method robustness against complex backgrounds, blurring, dimensions and distances constraints. Hypothesis 2 was validated, and the solution of adding a marker to a commercial drone enables an improved calibration adapted to challenging environments and flight conditions. Experiment 3 evaluated the calibration quality despite 3D and 2D measurement errors. It tested several trajectories to find some path parameters allowing for a successful calibration for many cameras. The test was made for 5000 simulated cameras to evaluate the scalability of the method. Experiment 3 demonstrated that the use of a drone as a calibration target is a successful solution for the calibration of many different cameras, and the impact of 3D localization uncertainty can be managed by choosing the right path parameters. Hypothesis 3 was validated.

The knowledge gained from these Experiments demonstrates the success of the proposed solution for the calibration of hard-to-reach cameras. Figure 5.27 summarizes this thesis findings presented so far.

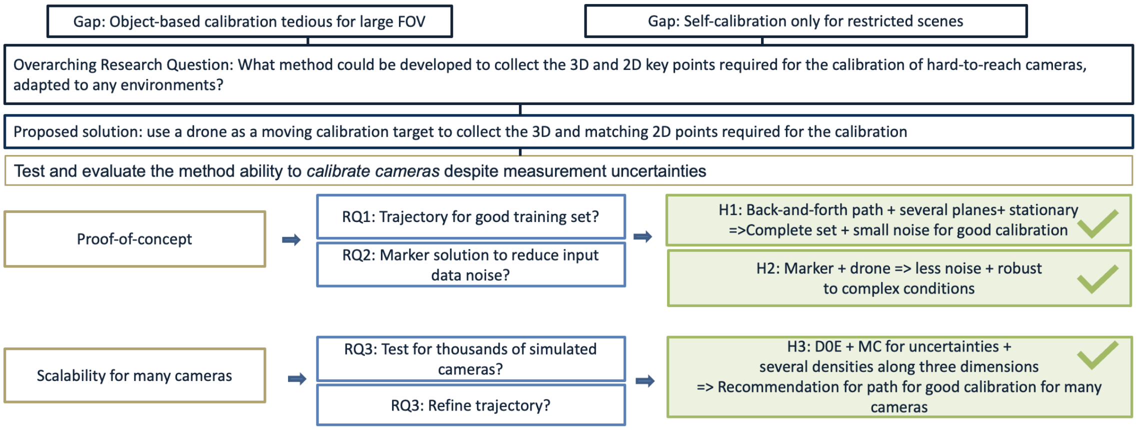

Gap: Object-based calibration tedious for large FOV
Gap: Self-calibration only for restricted scenes
Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?
Proposed solution: use a drone as a moving calibration target to collect the 3D and matching 2D points required for the calibration
Test and evaluate the method ability to calibrate cameras despite measurement uncertainties
Proof-of-concept
RQ1: Trajectory for good training set?
RQ2: Marker solution to reduce input data noise?
H1: Back-and-forth path + several planes+ stationary =>Complete set + small noise for good calibration ✓
H2: Marker + drone => less noise + robust to complex conditions ✓
Scalability for many cameras
RQ3: Test for thousands of simulated cameras?
RQ3: Refine trajectory?
H3: DOE + MC for uncertainties + several densities along three dimensions => Recommendation for path for good calibration for many cameras ✓

Figure 5.27: Summary of Research Objective, Research Questions, and Experiments presented so far

The knowledge gained from these experiments has been implemented with real hardware to further demonstrate the success of the drone-based calibration technique. The next Chapter presents this Real world experiment and a use case of the calibration of hard-to-reach cameras.

CHAPTER 6

REAL WORLD EXPERIMENT AND USE CASE 1

A key contribution of this thesis is the demonstration of a calibration approach adapted to remote cameras. The calibration method relies on the use of a drone to collect data required for the optimization part of the calibration. The use of a drone as a moving target enables to collect points in any area or volume size covered by the field of view of a camera. This solution satisfies the constraints of hard-to-reach cameras calibration, which are recalled below:

- Be adapted to large fields-of-view since these cameras cannot be accessed easily (which prevents the use of object-based calibration techniques)
- Be scalable to various environments (which is not feasible using self-calibration techniques that require strict assumptions about the scene)
- Be automated to enable the calibration of the large number of already installed cameras
- Be able to correct for the large non-linear distortion that is frequently present with security cameras

Experiments 1 and 2 presented a proof-of-concept for the calibration approach. Experiment 3 analyzed the impact of GPS uncertainty on the calibration accuracy, provided recommendation for path parameters to ensure a successful calibration, and evaluated the calibration approach for many different cameras.

The proof-of-concept that results from the two first experiments relied on ROS and Gazebo simulations. The use of simulations enabled to learn about the calibration approach without requirement costly equipment's. The Gazebo environments was designed to represent realistic scenarios [156]. However, it is interesting to test the approach with real hardware, and see how the simulation results deviate from the real experiment results. Consequently, the drone-based calibration approach, as described in Experiment 1, was run with an actual drone, camera, and in an outdoor environment. The drone used is a DJI Mavic Air [157], and the camera is a Nikon d3500 [158]. The drone used does not have a RTK GPS, and the experiment is run in an urban area where blockage of the GPS signal by buildings is likely to happen. Thus, the calibration error is expected to be larger than 3.6 pixels. In addition to evaluate the calibration quality, this test is also used to demonstrate the flow and smoothness of the drone-based calibration technique. The environment used for this real world test is Couch Park on the Georgia Tech campus of Atlanta. The next section describes the steps of the experiment.

6.1 Run experiment: fly the drone to waypoints

First, the drone was registered to FAA and a flight authorization was asked to the Georgia Tech Police Department. Then, the experiment was set-up. The camera was set on a tripod at the location called ”camera location” on Figure 6.2. which is a google map view from Couch Park.

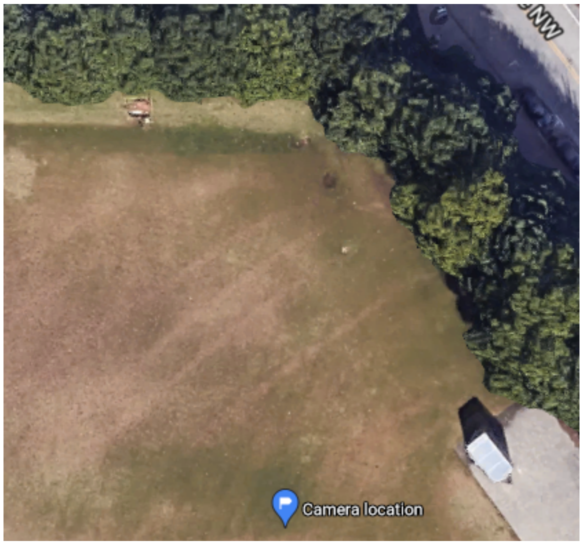


Figure 6.1: Satellite view of Couch Park and the camera location

The camera location was picked before running the experiment in the Park, and it was used to determine the drone trajectory. The knowledge gained from Experiment 3 about the distance camera-drone, and its impact on the calibration accuracy was used to define the drone starting point. In Experiment 3, it was observed that the farther the drone is, the smaller is the impact of its GPS noise on the calibration. In particular, it was recommended to fly the drone at least 4 meters away from the camera. Thus, on google map, a staring point for the drone is picked so that this distance condition is satisfied, and such that the drone can be seen through the camera (i.e. is in the field of view cone). Figure 6.2 is a satellite view of Couch Park, the area where lines are drawn represents the potential location of the starting point.

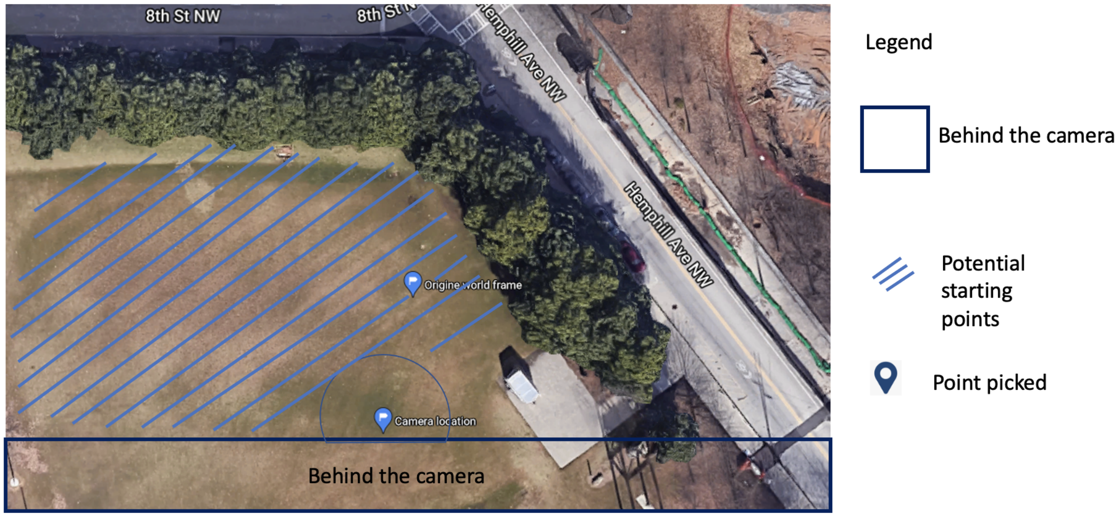


Figure 6.2: Potential starting points

Note: there are many points that satisfy the 4 meters condition, and the point was picked randomly. In a scenario where the camera being calibrated is a security camera, and its location can not be identified on Google map, then the trajectory starting point should simply be picked in the field-of-view of the camera, at an approximate distance of 10 meters from the camera. Any point in the field of view would work, and the farther is the point, the smaller is the GPS noise impact. One need only to make sure that the drone can be seen from the camera if a large distance is picked.

The starting point latitude and longitude are loaded in a .csv file. The .csv file is completed with the other waypoints latitude, longitude and altitude (Z_w=0 corresponds to the ground). The waypoints coordinates are determined based on observations made in Experiment 1 and 3: the waypoints crosses a distance from East to West large enough to make sure the image width will be covered by key points, and only two Z-values are sampled, while sampling different X, and Y values. Two trajectories were tested, and the corresponding waypoints latitude, longitude, and altitude are provide in Table 6.1 and Table 6.2.

Table 6.1: Waypoints - Path 1

Waypoint ID	Latitude (degree)	Longitude (degree)	Altitude (m)
0	33.77930009	-84.40263868	2
1	33.77930009	-84.40263868	4
2	33.77929508	-84.40265343	4
3	33.77929508	-84.40265343	2
4	33.77928728	-84.40266751	2
5	33.77928783	-84.40266751	4
6	33.77928114	-84.40268092	4
7	33.77928133	-84.40268099	2
8	33.7792739	-84.40266721	2
9	33.7792739	-84.40266721	4
10	33.77927724	-84.40265514	4
11	33.77927836	-84.40265514	2
12	33.77928114	-84.40264106	2
13	33.77928114	-84.40264106	4
14	33.7792856	-84.40262765	4
15	33.77928616	-84.40262765	2

Table 6.2: Waypoints - Path 2

Waypoint ID	Latitude (degree)	Longitude (degree)	Altitude (m)
0	33.77931744	-84.4026427	2
1	33.77931744	-84.4026427	4
2	33.77931242	-84.40265746	4
3	33.77931242	-84.40265746	2
4	33.77930462	-84.40267154	2
5	33.77930517	-84.40267154	4
6	33.77929849	-84.40268495	4
7	33.77929867	-84.40268502	2
8	33.77929124	-84.40267124	2
9	33.77929124	-84.40267124	4
10	33.77929458	-84.40265917	4
11	33.7792957	-84.40265917	2
12	33.77929849	-84.40264509	2
13	33.77929849	-84.40264509	4
14	33.77930294	-84.40263167	4
15	33.7793035	-84.40263167	2

The drone GPS, and google map, provide latitude and longitude coordinates. For the calibration purpose, the world coordinates must be defined in a Cartesian frame, locally, in the camera surrounding. These local coordinates are the 3D key points used in the calibration optimization. The use of coordinates in this projected frame corresponds to a normalization of the input, often performed in regression.

The world frame (camera local frame) is called (X_w,Y_w,Z_w): It is an East-North-Up (ENU) frame at the surface of Earth, in the neighborhood of the camera. The frame origin is taken as the ground point where the drone starts, such as the first waypoint visited by the drone is at X_w= 0, Y_w= 0. The third coordinate is the altitude (vertical up) of the first waypoint, and it is set to 2 meters above the ground. Figure 6.3 illustrates the relation between longitude λ, latitude ϕ, and the reference frame coordinates ENU (X_w,Y_w,Z_w).

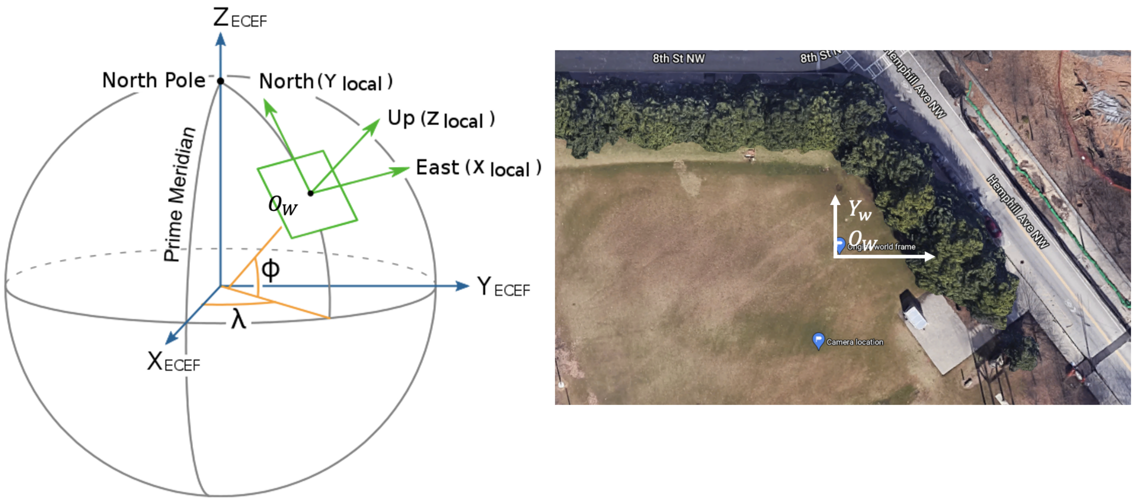


Figure 6.3: From GPS coordinates to local frame used for calibration

The Latitude and Longitude coordinates are converted to ENU coordinates (i.e. world coordinates when working on camera calibration) by first converting latitude and longitude into Cartesian coordinates in the Earth Centered Earth Fixed Frame (ECEF), and then converting the ECEF coordinates in ENU coordinates. This last conversion requires the definition of the origin of the ENU frame. For simplicity, the origin is taken at the starting point of the drone trajectory. The conversion between latitude (ϕ) and longitude (λ), and ECEF Cartesian coordinates is made using these relations:

$$X_{ECEF} = R * cos(\lambda) * cos(\phi)$$

$$Y_{ECEF} = R * sin(\lambda) * cos(\phi)$$

$$Z_{ECEF} = ratio * R * sin(\phi)$$

Where R is equal to the Earth Radius added to the altitude of Atlanta, and the ratio is given by: $ratio = (\frac{a}{b})^2$ where a=6378.137 km, b=6356.752km.

The conversion from ECEF to ENU requires defining an origin first, whose coordinates in the ECEF frame are (X_0, Y_0, Z_0), and latitude and longitude ϕ_0 and λ_0. In this work, the origine of the Frame is the drone starting point. The conversion from ECEF coordinates to ENU coordinates X_{ENU}, and Y_{ENU} is made using the following equation, and Z_{ENU} is equal to the altitude defined in .csv files described earlier:

$$X_{ENU} = -sin(\lambda_0) * (X_{ECEF} - X_0) + cos(\lambda_0) * (Y_{ECEF} - Y_0)$$

$$Y_{ENU} = -cos(\lambda_0)*sin(\phi_0)*(X_{ECEF}-X_0)-sin(\lambda_0)*sin(\phi_0)*(Y_{ECEF}-Y_0)+cos(\phi_0)*(Z_{ECEF}-Z_0)$$

This conversion enables to get the 3D key points input of the calibration. The result for Path 1 can be seen in Table 6.3, and on Figure 6.4, where the navy blue points are the waypoints where the drone stops, and the light blue edges represent the path followed by the drone.

Table 6.3: Waypoints in ENU cooridnates - Path 1

Waypoint ID	X (m)	Y (m)	Z (m)
0	0	0	2
1	0	0	4
2	-1.369721536	-0.557748932	4
3	-1.369721536	-0.557748932	2
4	-2.677183241	-1.425358324	2
5	-2.677183224	-1.3633862	4
6	-3.922385029	-2.107051259	4
7	-3.928865487	-2.086647166	2
8	-2.649293865	-2.912689407	2
9	-2.649293865	-2.912689407	4
10	-1.52861225	-2.540856864	4
11	-1.528612231	-2.416912603	2
12	-0.221150479	-2.107052078	2
13	-0.221150479	-2.10705208	4
14	1.024051086	-1.611275021	4
15	1.024051079	-1.549302897	2

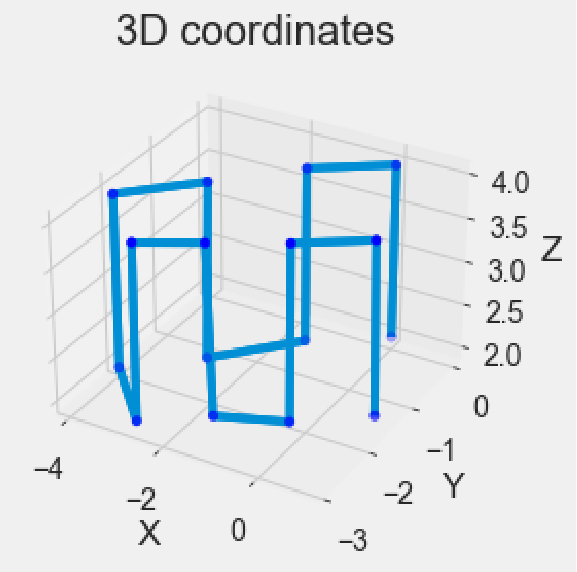
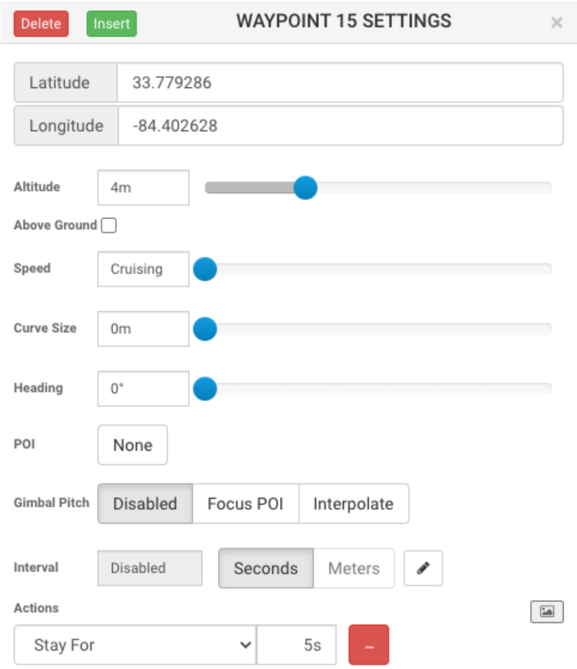


Figure 6.4: Trajectory in the ENU frame

The waypoints coordinates in latitude, longitude, and altitude stored in the .csv files, are loaded on an application called Litchi, that was developed to facilitate autonomous mission for UAVs. The application exists on phones (Litchi app), and on computers (Mission Hub) [159]. Once the .csv file which contains the waypoints is opened on Mission Hub, some additional characteristics can be added to the trajectory. A stopping duration of 5 seconds is added at each waypoint, so that the matching between world coordinates and image coordinates is convenient (no need of time synchronisation). Figure 6.8 provides the Mission Hub interface used to set a waiting time of 5 seconds at each waypoint.



WAYPOINT 15 SETTINGS

Latitude 33.779286

Longitude -84.402628

Altitude 4m

Above Ground ☐

Speed Cruising

Curve Size 0m

Heading 0°

POI None

Gimbal Pitch Disabled | Focus POI | Interpolate

Interval Disabled | Seconds | Meters

Actions

Stay For 5s

Figure 6.5: Set a waiting time of 5 seconds at waypoints

Mission Hub allows to visualise the waypoints using a satellite view. The drone flies to GPS waypoints (latitude, longitude, altitude), and these waypoints are converted to ENU coordinates for the calibration. Figure 6.9 illustrates the GPS coordinates with the left chart. The right image is taken from Mission Hub interface [159], the purple points are the waypoints where the drone flies in this experiment see from the top (altitude not visible on this image).

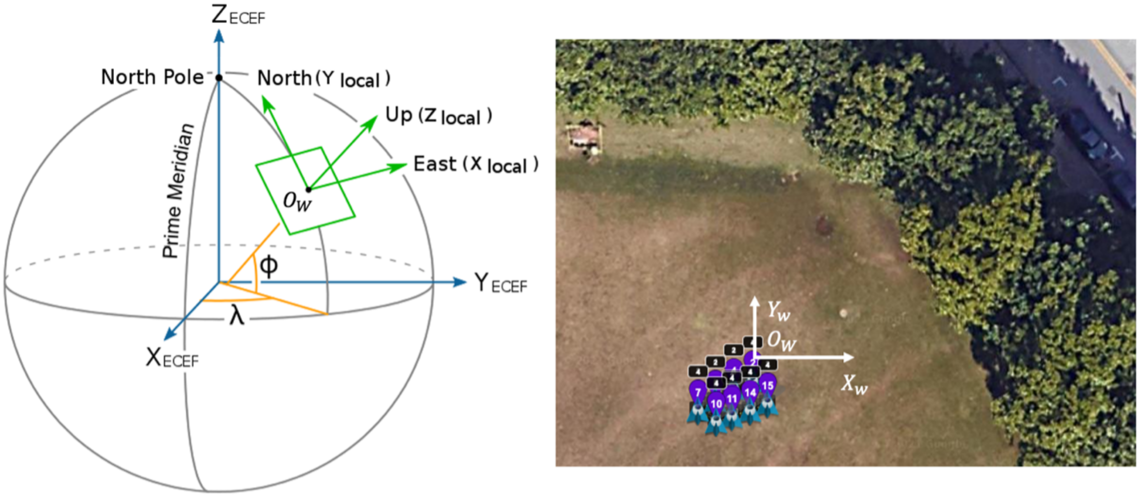


Figure 6.6: Relation between GPS coordinates, and world coordinates for the calibration

Once the waypoints are defined in the .csv file, and loaded on the Litchi app, the drone can start the mission and flies to the waypoints in an automated way. When it is done, it lands to a home point defined by the user. The camera recorded the drone trajectory. The footage is then used to extract the 2D key points for the calibration. The next section described the processing of the experiment data and evaluation of the calibration accuracy.

6.2 Process experiment data: process video and calibrate the camera

The experiment is run in challenging conditions: in a urban environment where trees and buildings might cause GPS blockage, and in a cluttered background with trees that make the drone identification harder. The experiments were run on a day with nice lighting and low wind, and on a windy day. These challenging conditions were chosen on purpose to get an estimate of the ”worst case” accuracy.

Figure 6.7 represents a cropped portion of the video where the drone is highlighted in red. From the video, the drone is detected with the clicking method described in Experiment 1. Since the key point selected and tracked on the drone is not provided by a marker, the resulting 2D coordinates measurement error is similar to the one observed in Experiment 1, and larger than the one obtained in Experiment 2. Errors are also introduced by the DJI localization system. According to the specification, the localization system error is 10 centimeters vertically and horizontally [157]. In this experiment, the drone flies in an urban environment, and buildings and trees introduce errors on GPS measurements. So the 3D localization error is expected to be larger than the one provided in the specifications. Also, the wind deviates the drone when it hovers, and thus introduce additional localization errors. The exact localization error is not known, but the impact resulting from the image detection error and the GPS error on the calibration can be evaluated with the reprojection error.

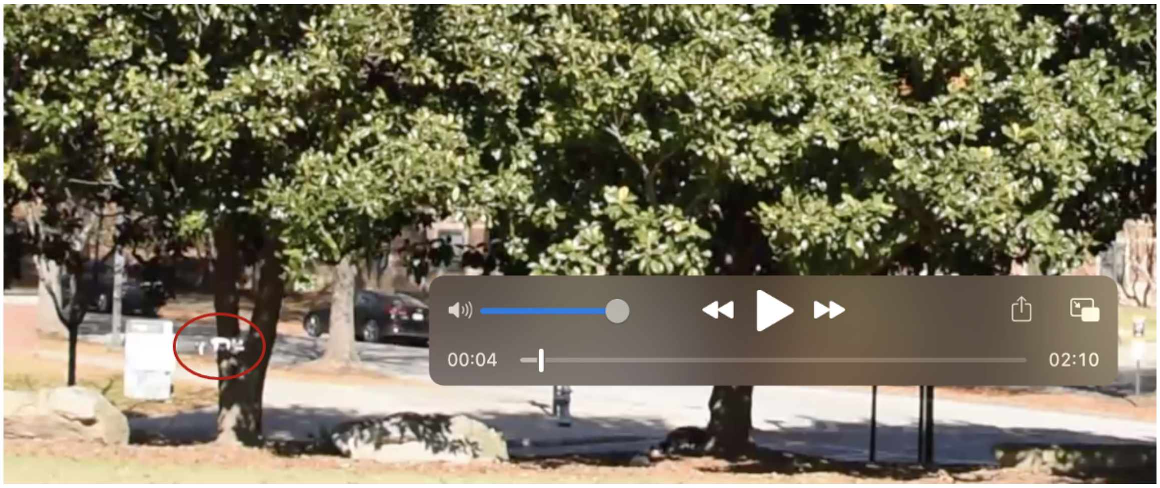


Figure 6.7: Detect the drone on the video captured by the camera being calibrated

Once the key points are detected on the image and their coordinates is stored in a table, the training set made of 3D key points in the ENU frame and their corresponding images is complete and can be used for the optimization steps.

The optimization steps estimate the camera parameters. From these estimates, the relation world to image is computed, and the rerojection error on the training points is computed. Table 6.4 provides the RMS reprojection errors in pixels for the two real testings, and provides the flight duration.

Table 6.4: RMS reprojection error

Path	Time (minutes and seconds)	RMS reprojection error
Path 1 - windy	2’10”	21.6 pixels
Path 2	2’01”	7.7 pixels

The experiments were done in challenging conditions: windy, cluttered background, and urban area, which induced noise in 2D and 3D key points. The error obtained are ”worst case scenario” errors. The reprojection error is equal to 7.7 pixels for the less windy experiment. For the calibrated camera, this error is equivalent to a measurement error on the ground equal to 43 cm. Thus, the calibration is suitable for the localization of large objects. Using a RTK GPS, and a marker on the drone would reduce the error as demonstrated in Experiment 2 and 3, and in turn would reduce the corresponding measurement error on the ground. The next section presents an application of this calibration. The chosen use case is the obstacle localization from the calibrated camera.

6.3 Use case: obstacle location

The objective is to determine an obstacle location in the world frame from its 2D location on the image taken from the remote camera who has been calibrated with the drone-based calibration approach. Thus, whose intrinsic and extrinsic parameters are known.

An application of this process is the determination of the location of an obstacle on the road, such as a tree that falls due to a storm and blocks part of the road. Figure 6.8 depicts such a situation. The obstacle is being captured by a remote camera, such as a security camera. The question answered by this use case is: Where is this obstacle actually located, in the street frame of reference or in the ECEF frame? This information would be useful for road safety, and for autonomous driving technologies. The obstacle location would be automatically sent to a database. From this information, a safety radius around the obstacle can be determined, leading to a deviated and safer path of the vehicle, which would be determined before the vehicle arrives on site.

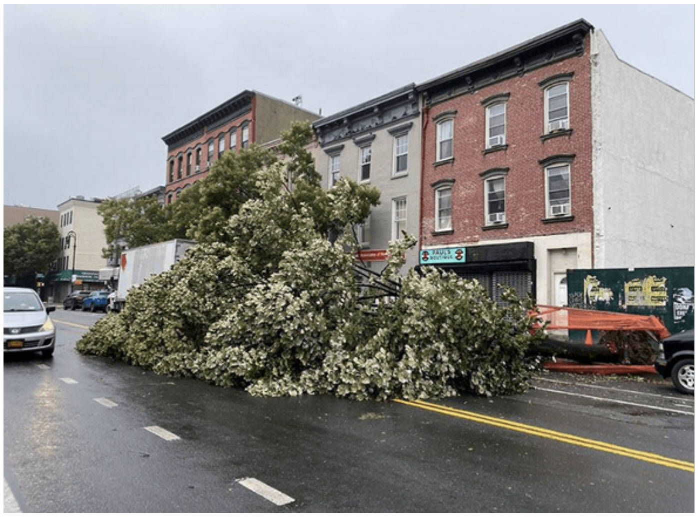


Figure 6.8: Temporary obstacle which can lead to safety concerns

6.3.1 Database of obstacles' location for safety purposes

The long term objective is the real-time collection of images, obstacles, and metadata about obstacles and their location. This data would be included in a database for road safety purposes. The collection of a database of images, with their metadata is not new. Indeed, aerial mapping from satellites or planes results in set of images associated with location data [160]. The idea demonstrated with this use case is to set-up a similar system using security cameras. The advantage is that security cameras are already installed, so it would reduce cost and C02 emissions due to airplanes use. Also, these cameras present an advantage compared to satellites sensing, since they are monitoring areas continuously on the contrary to satellites who point towards changing spots on Earth while moving on their orbits. Thus, the system would provide recent localization information about potential safety issues detected through remote cameras.

6.3.2 Demonstration from the real-world testing

To demonstrate how works this use case, an experiment was run to detect a fire hydrant system captured by the camera being calibrated using the drone-based calibration technique. The location of the ENU origin, the water hydrant, and the camera, are represented with blue flags on Figure 6.9. The goal is to get the ground coordinates of the fire hydrant system in the ENU frame. The ground location determined with this technique is compared to the ground location obtained with Google map. This comparison can be done in GPS coordinates, or in the ENU frames. Because the difference in coordinates using latitude and longitude might be hard to observed, the ENU coordinates are used. First the origin, and the water hydrant latitude and longitude are obtained from Google map, then they are converted in the ENU frame as described earlier in this chapter. The fire hydrant Google map coordinates are X_{ENU}=-1.5m and Y_{ENU}=43m.

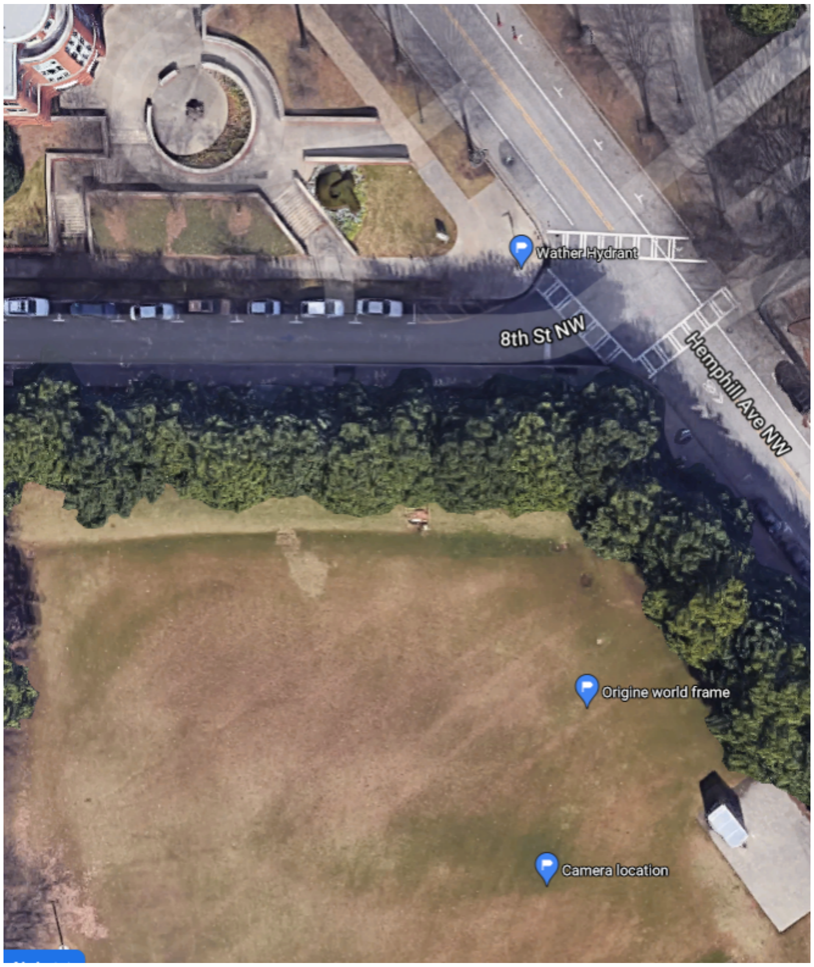


Figure 6.9: Points of interest - Fire hydrant use case

The camera is calibrated using the drone-based calibration technique. The internal camera parameters obtained through the calibration process are used to undistort the image. The steps describe next are implemented using the undistorted image. The fire hydrant is detected on the image to obtain its pixel coordinates. Figure 6.10 illustrates the fire hydrant seen through the camera that has been calibrated with the drone-based calibration approach. The pixel frame is represented with the u, and v axes drawn on the image.

The camera location and its intrinsic parameters (focal length, image width and height) have been estimated with the calibration. From it and using the undistorted image, the mapping world to image can be computed using the equation below: u and v are the coordinates on the undistorted image.

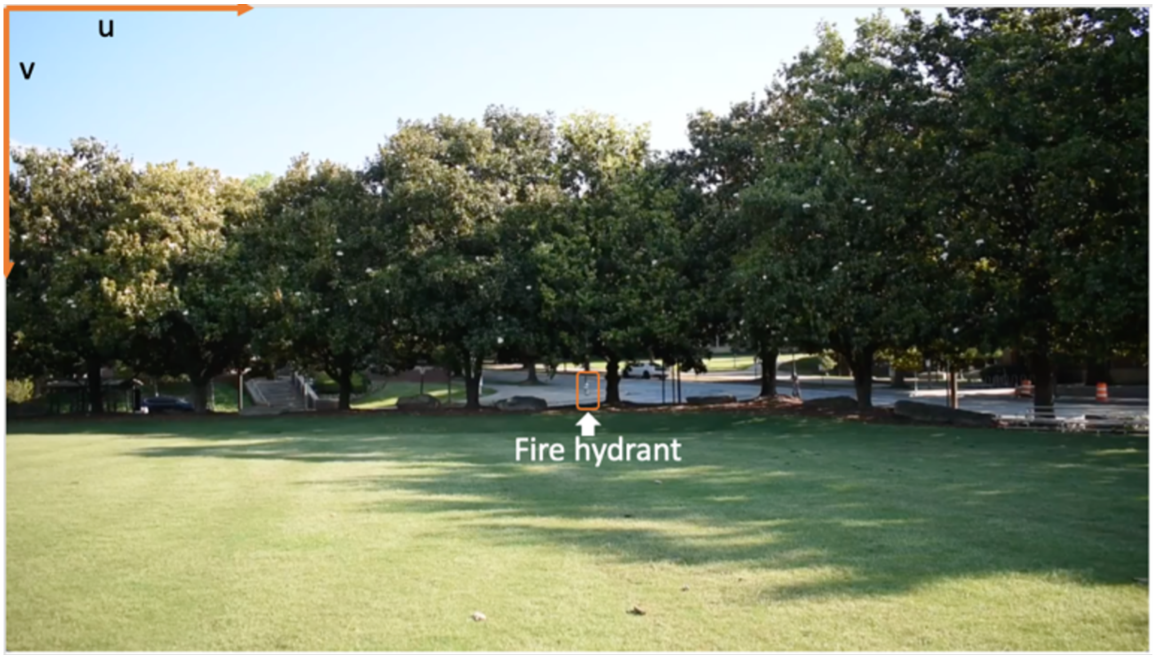


Figure 6.10: Camera view and fire hydrant image

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = K * [R\,T] * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \begin{bmatrix} f & 0 & \frac{w}{2} & 0 \\ 0 & f & \frac{h}{2} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

The water hydrant is located on the ground, which enables to set Z = 0 and build a reduced H that is square and can be inverted to compute X and Y.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{14} \\ h_{21} & h_{22} & h_{24} \\ h_{31} & h_{32} & h_{34} \end{bmatrix} * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = A * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

so

$$scalar * \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = inv\left(A\right) * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Using this last set of equation, the coordinates on the ground of the water hydrant are determined. The result is X=-0.9m and Y=36m. The coordinates obtained with Google map were X_{ENU}=-1.5m and Y_{ENU}=43m. The distance difference between the fire hydrant position measured with the proposed technique and the one found with Google Map is 7 meters. Note that Google map accuracy for satellite views is 10 meters [161], so Google map is not a ground truth. However it is used to compare the order of magnitude of the output from the method presented in the application, with another source of localization.

6.3.3 Additional use case: speed measurement

Following the same process, the speed of an object can be measured. The object is detected at two different locations in a time interval known, or that can be measured with the frame rate. The image location is converted in ground location as described with the water hydrant problem. Finally the average speed is measured as the difference between the two locations divided by the time.

Determining the ground location is not the only way to measure the speed. Once the camera is calibrated its parameters can be used to convert directly the speed in pixel per second in actual speed in meter per second. The equations that allows this conversion are provided in the following paragraph. It is assumed that the camera is horizontal, but the same steps can be followed for other camera orientations.

The speed in meter per second can be determined using the camera frame or the world frame. Indeed both frames are fixed the one compared to the other, so the speed is the same in both frames. The paragraph below explains why the norm of the speed is the same in the world frame and in the camera frame. The change of frame relation for speed is given by:

$V(M)_R = V(O\prime)_R + W^O_{R'/R}{}'M + V(M)_{R\prime}$ where R is the world frame and R' the camera frame. Since the camera frame is fixed compared to the world frame then the speed of the center of the camera frame in the world frame is zero: $V(O\prime)_R$ and the rotation velocity $W_{R'/R}$ is zero too. Consequently, $V(M)_R = V(M)_{R\prime}$.

This does not take into account the frame in which the speed coordinates are written. However it implies that the speed in the world frame and the speed in the camera frame have same norm and have same components if written in the same frame of reference. The next paragraph provides details on how to determine the speed in the camera frame. The speed in the camera frame is the derivative of the location $\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$.

The vector $\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$ is determined by first computed the undistorted image coordinates x and y. Using the undistorted images, the coordinates of an object detected in this undistorted image can be related to the metric image coordinates as follow: $u = f * \frac{X_c}{Z_c} + C_x$ $v = f * \frac{Y_c}{Z_c} + C_y$

These metric image coordinates are then used to get the camera coordinates using $\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \frac{Z_c}{f} \begin{bmatrix} u - C_x \\ y - C_y \\ f \end{bmatrix}$. It can be seen with this equation that Z_c must be computed. To do so, the equation $Y_c = \frac{Z_c}{f} * v$ is used and the information about the camera height is used. Hence $Z_c = f\frac{h}{v - C_y}$ and $\frac{dZ_c}{dt} = -f\frac{h}{(v-Cy)^2}\frac{dv}{dt}$. Using this information, the speed in the camera frame can be computed as a function of pixels coordinates u and v, intrinsic parameters f, C_x, C_y and camera height: h. First the speed components are derived and then they are used to compute the norm of the speed which is of interest here.

$$\frac{dX_c}{dt} = \frac{-h}{(v - C_y)^2}(u - C_x) * \frac{dv}{dt} + \frac{h}{(v - C_y)}\frac{d_u}{dt}$$

$$\frac{dY_c}{dt} = 0$$

$$\frac{dZ_c}{dt} = \frac{-f * h}{s * (v - C_y)^2} * \frac{dv}{dt}$$

$$V = \sqrt{\frac{dX_c}{dt}^2 + \frac{dY_c}{dt}^2 + \frac{dZ_c}{dt}^2}$$

$$= \sqrt{(\frac{-h}{(v - C_y)^2}(u - C_x) * \frac{dv}{dt} + \frac{h}{(v - C_y)}\frac{d_u}{dt})^2 + (\frac{-f * h}{(v - C_y)^2} * \frac{dv}{dt})^2}$$

Using the series of pixel locations for a tracked object (vehicle), the speed in pixels coordinates is determined: $\frac{d_u}{dt} = \frac{u_{t+1} - u_t}{dt}$, $\frac{dv}{dt} = \frac{v_{t+1} - v_t}{dt}$, where t+1 is the location at a frame n+k, t at a frame n and dt=fps*k. Then using the formula for speed derived above, the speed in meter per second can be computed. This method requires the estimation of less camera parameters compared to the first method presented to compute the speed measurement.

From this last equation, one can look at the impact of parameter estimation error on the final speed measurement. Let us assume a vehicle moving horizontally (i.e. along X_{cam}), then the actual vehicle speed is

$$V = \frac{h}{(v_{constant} - C_y)}\frac{d_u}{dt}$$

The impact of an error in the elevation estimate on the final speed can be computed by varying h and letting all other parameters constant. The impact of an error in h on the velocity error is represented on Figure 6.11. In a similar way, the error in the estimate of the image center coordinate C_y is represented on Figure 6.12.

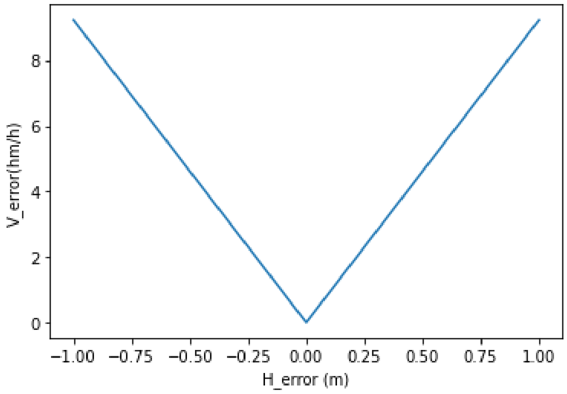


Figure 6.11: Impact of the error in the estimation of H on the resulting speed measurement error

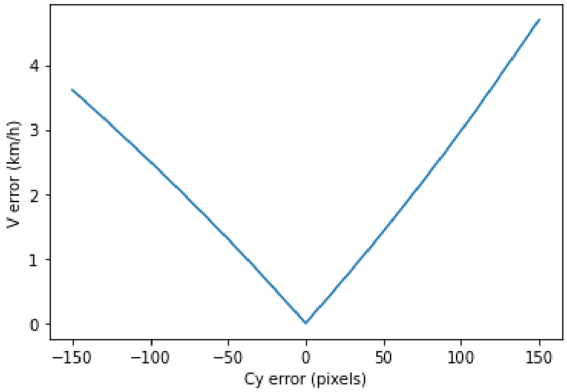


Figure 6.12: Impact of the error in the estimation of C_y on the resulting speed measurement error

6.4 Summary

This chapter presented a test of the drone-based remote camera calibration using real hardware. The test was done with a commercial drone, a GPS (not a RTK GPS), and a commercial camera. The drone trajectory was implemented based on the knowledge gained from each Experiment. The calibration RMS error was 7.7 pixels. Given that the 3D localization sensor used was a GPS in an urban environment, where GPS signal tends to be blocked frequently, the accuracy result is very promising. If a RTK GPS replaces the GPS, the error will be reduced.

The chapter also presented a use case example, which was the obstacle location from a calibrated camera. This application would help reducing traffic incidents by locating obstacles in real-time. Finally, this chapter presented how the speed in km/h can be computed from image information. This use case is further investigated in the next chapter.

The real testing was also done to verify the flow and smoothness of the drone-based calibration technique. A methodology to reproduce the drone-based camera calibration is presented in the next section.

6.5 Methodology to reproduce the work

From this real-world experiment, knowledge was gained about the methodology to reproduce the drone-based calibration. Figure 6.13 summarizes the methodology. First, a starting point for the drone is selected. This could be done by picking a point on Google map that is in front of the camera, and collecting its latitude and longitude. Then, this point is taken as the origin of the projected ENU frame. A set of points that satisfy the conditions established in Experiment 1, 2 and 3 is added to this starting point. If the user has no idea of the camera field of view size, then he should plan a very large path from East to West to make sure the drone covers a horizontal distance bigger than the FOV. The ENU coordinates are then converted into GPS coordinates. Depending on the framework used to fly the drone in an autonomous way to waypoints, the next step might varies. If one use Mission Hub or Litchi, then a condition for stopping at waypoints can be added easily. Once the path is entirely determined, the drone mission starts and the drone flies in front of the camera to the waypoints. Once the drone is back, the video is collected and the image waypoints are identified and their pixel coordinates stored with the 3D waypoints coordinates. From the key points, the calibration function is run, and the camera parameters estimated. Then, once the camera is calibrated, it can be used to detect obstacles' location, and the tool for this step is provided in the Annex 3.

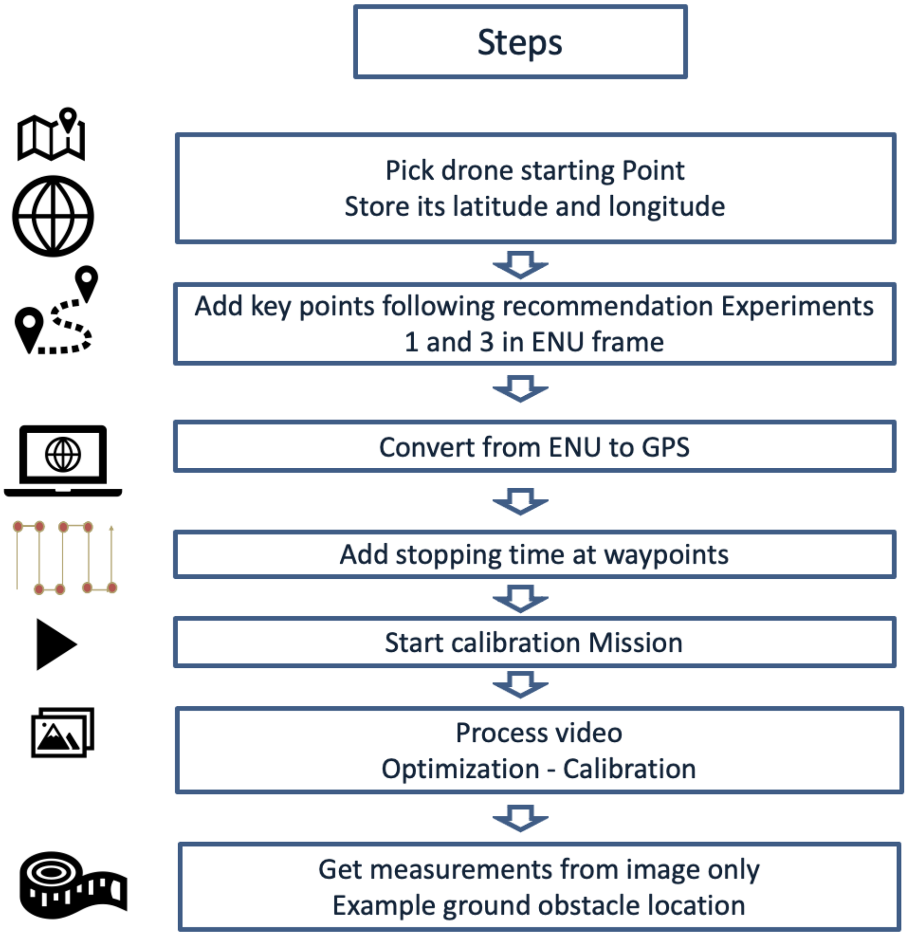

Steps
Pick drone starting Point
Store its latitude and longitude
Add key points following recommendation Experiments 1 and 3 in ENU frame
Convert from ENU to GPS
Add stopping time at waypoints
Start calibration Mission
Process video
Optimization - Calibration
Get measurements from image only
Example ground obstacle location

Figure 6.13: Methodology for the drone-based remote camera calibration

6.5.1 Discussion: One step method VS two steps method

This section discusses the way to implement the second step in the previous chart: ”Add key points following the recommendation in Experiments 1 and 3”. A key constraint on the drone trajectory was established in Experiment 1 and is recalled below:

> Gap 1: The training set must satisfy the completeness requirement. It must be an accurate representation of the application sets.

To answer this constraint, the drone must cover the entire field of view of the camera, to capture regions of complex behaviors, and regions of linear behaviors. This section provides guidance to help a potential user to reproduce the drone-based calibration technique, and especially to capture points in the entire field of view.

The first way, which is the one tested in the real hardware experiment, over-sample the volume in front of the camera. The path along East to West is designed to be much larger than the actual camera field of view. Figure 7.3 depicts the setting of the real-world experiment. The red triangle is a top-view of the camera field of view, the orange line represents the distance covered by the drone with a back-and-forth path that follows the guideline provided at the end of Experiment 1 and 3.

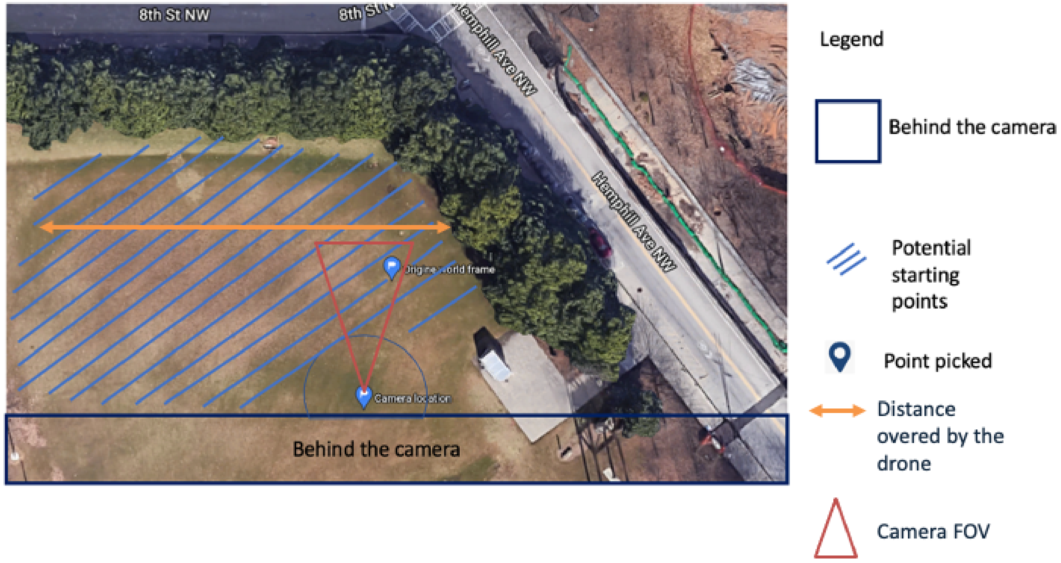


Figure 6.14: One step Method: over-sampling

The second way requires two steps. This method assumes a synchronization between the camera clock and the GPS clock. First, the drone flies in front of the camera from East to West. It is detected in an autonomous way, which could be done with a template matching step for instance. The frontiers East and West of the FOV are determined by detecting the locations where the drone appears on the image, while for all previous frames the drone where outside of the FOV, so not visible on the image. The video time at which the drone appears is collected and the GPS location for this video time is collected. Experiment 3 demonstrated that the density of points along the Z axis does not impact the calibration quality. So the process described in this paragraph is applied to detect West and East frontiers only. In this way, the East and West boundaries for the drone path are known before starting the calibration. Then, the back-and-forth path that satisfies the conditions stated in Experiment 3 is implemented within these boundaries.

This process is summarized in Figure 6.15.

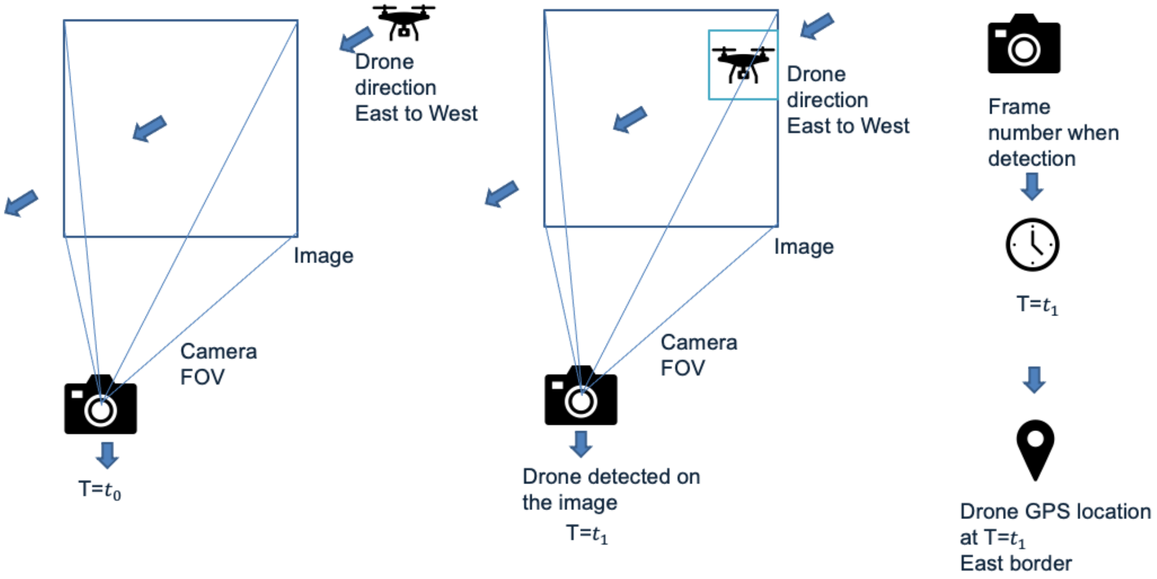


Figure 6.15: Two steps Method: Detect the boundaries East and West and then define the Back-and-forth path

CHAPTER 7

TRAJECTORY OPTIMIZATION

7.1 Need for an efficient way to map ground to image

More than 70 million security cameras have already been installed in North America, and very few of them are calibrated. Calibration would add capabilities to these cameras. For instance, it would enable measuring vehicles' speed or obstacle's location on the ground, such as the ones caused by storms. In turn, this would improve road safety. The calibration methodology is adapted to the large number of already installed cameras and business applications if it relies on a fast set-up, low cost, and robust process. This section focuses on optimizing the calibration approach to enable a simple calibration target path and accurate calibration.

For the specific application of speed measurement, the target points used to measure the speed can be taken on the ground. Once these points are detected on the image, their speed in image coordinates can be converted into speed in the real-world coordinates using the image to ground relation for the security camera. Consequently, camera calibration is required. More precisely, the relation between the image planes and ground planes must be determined. As demonstrated in the previous chapter, this relation can be obtained by first determining the relation from 3D to 2D and then simplifying it. If the environment where the camera is located is windy, then the drone-based calibration is more challenging and might result in lower accuracy compared to non-windy conditions, as demonstrated with one of the test introduced in the previous chapter. For this reason, the present Chapter proposes to replace the drone with a moving target that is more stable for windy conditions. This target is a wheeled robot moving on the ground. Using this moving target does not enable to determine all parameters introduced in Chapter 2. However it enables to determine the parameters necessary for the speed measurement application in the plane where the robot will drive. The equation for this relation are derived below.

The relation from ground points to image points for a pinhole camera is derived from the relation world to image, and setting Z to 0, so the third column of the homography matrix can be removed, which leads to:

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} m_{11} & m_{12} & t_x \\ m_{21} & m_{22} & t_y \\ m_{31} & m_{32} & t_z \end{bmatrix} * \begin{bmatrix} X_W \\ Y_W \\ 1 \end{bmatrix}$$

So the relation ground points to image points is given by the following equation where H is a 3*3 matrix:

$$s * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H * \begin{bmatrix} X_W \\ Y_W \\ 1 \end{bmatrix}$$

The relation image to ground is the inverse relation. It is written as follow where p is a scalar value, and M is a 3*3 matrix. It can be seen with this equation that the matrix M is defined up to a scale factor, thus among its 9 coefficients, there is one fixed, and 8 unknowns.

$$p * \begin{bmatrix} X_W \\ Y_W \\ 1 \end{bmatrix} = M \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

A point (X, Y, u, v) provides two set of equations to solve for these 8 unknowns. Thus, at least 4 points are required to enable a calibration of the mapping image to ground.

The relation image to ground for a camera with distortion can be completed by including the inverse distortion that maps $(u_{distorted}, v_{distorted})$ [162] to (u, v), and using the previous equation to map (u, v) to (X, Y).

Gap and Research Question

For the same reason as presented in Chapter 1, existing methods are not adapted for this calibration problem because they require a long and tedious process, or rely on strict scene conditions such as vanishing lines or overlapping camera views, which are not satisfied by most static security cameras, also called intersection cameras. Thus, the method of this thesis is preferred to tackle this calibration problem. The proposed solution relies on using a robot and its GPS sensor as a moving calibration target. This work aims to find an optimized robot path that is simple, fast, and leads to an accurate calibration. This would enable to satisfy the requirements of fast set-up, low cost, and robustness mentioned previously. Reducing the number of training points generally increases the variance of a regression model and thus its error. This leads to the following gap:

> Gap: There is a tradeoff between the cost of the method, which corresponds to the time of use of the robot or the number of training points sampled, and the accuracy of the method.

The following research question addresses this Gap:

> Research Question 4: Is there an optimum sampling set for a given camera that enables accurate calibration and reduced trajectory time?

The process to build the mapping between image points and ground points is a light modification of the calibration process presented in chapter 2. The main differences are highlighted in the list of steps below:

- Point sampling: The input points are image points, and the outputs are their corresponding ground locations

- Optimization: the training points are used to fit the unknown parameters using the same optimization steps as in Chapter 2. The difference relies on the equations used.

Since Z is zero because points are on the ground, the homograpy matrix that relates world points to their projection on the image before distortion is a 3*3 matrix instead of the 4*4 matrix. The function being minimized is $F = \sum_{i=1}^{N} (X_{_i} - x_i)^2 + (Y_{_i} - y_i)^2$ where X_i and Y_i are the actual values of the 2D ground points used for the training and xi and y_i are their model values that are functions of the unknown parameters and the image key points.

- Model evaluation: The goal of this work is to reduce the training set size. Thus, the training set will not be a homogeneous representation of application sets. It will not cover the full FOV, but be restricted to only a few points in the FOV. So the calibration cannot be evaluated using the training set, and the accuracy will be evaluated using a testing set of points on the ground

7.2 Definition of the optimization problem

Starting from a training set that enables a good calibration, the goal is to find a subset of points that would lead to a similar accuracy. The accuracy is evaluated using a validation set and measuring the maximum reprojection error on the ground. Indeed, the application of interest in this section is the vehicle speed measurement from static cameras. The ground reprojection error is directly connected to the speed measurement error. For cameras with large distortions, using a training set close to the image center causes a large error at the image extremities. Using the maximum reprojection error on the validation set as accuracy metric better represents the challenges faced when working with a small training set. Since the goal of this optimization is to find a smaller size training set for the regression while keeping a good calibration accuracy, the objective function depends on the number of sampled points and the maximum reprojection error on the validation set. The proposed approach starts from a given baseline set that enables an accurate calibration. A discrete optimization technique would compare the calibration accuracy enabled by different subsets of training points. The optimization must account for the image and ground measurement uncertainties. The Monte Carlo sampling described in Experiment 3 can be integrated in the optimization procedure to capture the impact of input uncertainties on the calibration error. The observation made in this paragraph leads to the following requirements for the optimization process:

- The optimization compares subset of points together and to a baseline set of points known to lead to good accuracy. So, the comparison relies on discrete optimization
- Uncertainties in measurement must be captured to enable a robust evaluation of the error. The proposed solution relies on the implementation of a Monte Carlo simulation. Since Monte Carlo simulation requires large number of iteration and is computationally expensive, an optimization method that enables parallelism in the search for the optimum is needed to balance the computational time

On the contrary to traditional optimization algorithms, genetic algorithm can explore the space in different directions simultaneously and is consequently adapted to parallel processing during its implementation [163]. Thus they satisfy the second requirement listed above. A genetic algorithm is also suitable solution for discrete optimization problems [164]. If N is the baseline set size, then a vector of size N with binary values can be taken as the chromosome. For the gene at position K in [0,N-1], the value is set to 1 if the point is used for the training subset and 0 if it is excluded. The choice of this chromosome enables to handle a discrete design space which satisfies the first requirement. The computation of the fitness function does not require the knowledge of the explicit function formula and is adapted to the computation of the Monte Carlo estimate of the maximum reprojection error on the validation set. In addition to these advantages, the properties of mutation and crossover of the genetic algorithm enables an efficient exploration of the baseline set.

This algorithm should enable an efficient exploration of the baseline set to look for a reduced size training set while maintaining the calibration accuracy. This would enable to evaluate whether the number of sample points can be reduced and the requirements of fast set-up, low cost and robustness introduced at the beginning of this section can be satisfied. Consequently, this leads to the following hypothesis:

> Hypothesis 4: IF a genetic algorithm is implemented, with a Monte Carlo sampling to capture uncertainty, and a chromosome that represents the use of a baseline point or not, THEN a reduced sampling that ensures accurate calibration can be identified

Figure 7.1 summarizes the observations, research question and hypothesis introduced in this section. Hypothesis 4 is validated if the optimization designed as described in this hypothesis enables to identify a training set of reduced size and low error. If reducing the training set size means getting an error larger than the 15 centimeters bound defined in Chapter 3, then the Hypothesis is rejected.

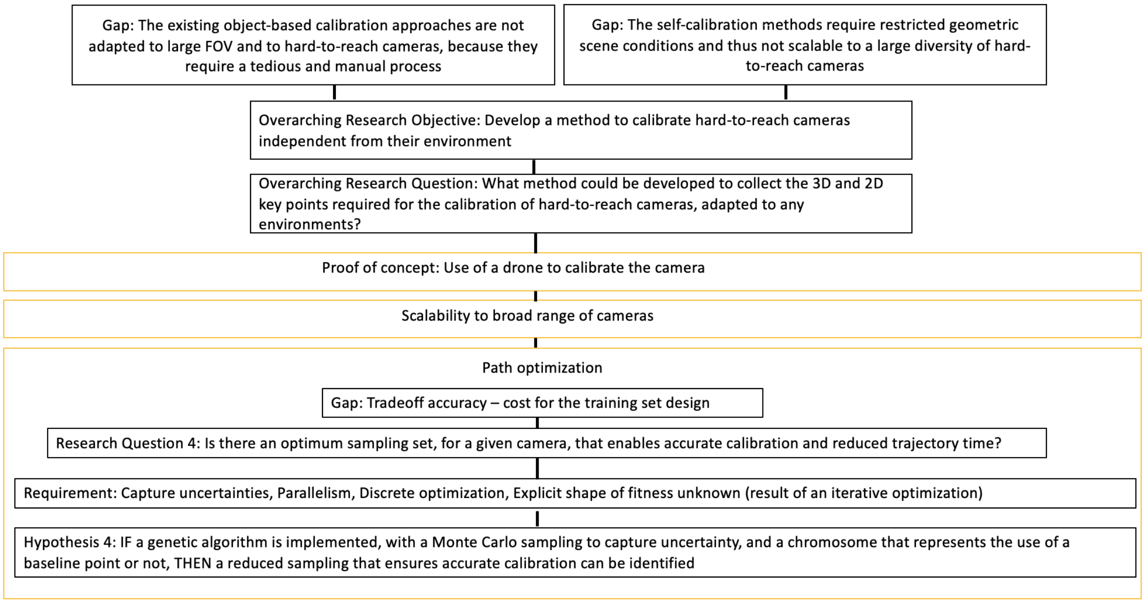

Gap: The existing object-based calibration approaches are not adapted to large FOV and to hard-to-reach cameras, because they require a tedious and manual process
Gap: The self-calibration methods require restricted geometric scene conditions and thus not scalable to a large diversity of hard-to-reach cameras
Overarching Research Objective: Develop a method to calibrate hard-to-reach cameras independent from their environment
Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?
Proof of concept: Use of a drone to calibrate the camera
Scalability to broad range of cameras
Path optimization
Gap: Tradeoff accuracy – cost for the training set design
Research Question 4: Is there an optimum sampling set, for a given camera, that enables accurate calibration and reduced trajectory time?
Requirement: Capture uncertainties, Parallelism, Discrete optimization, Explicit shape of fitness unknown (result of an iterative optimization)
Hypothesis 4: IF a genetic algorithm is implemented, with a Monte Carlo sampling to capture uncertainty, and a chromosome that represents the use of a baseline point or not, THEN a reduced sampling that ensures accurate calibration can be identified

Figure 7.1: Summary of observations, research question and hypothesis to test whether an optimum set of points can be sampled while maintaining a good calibration accuracy

7.3 Experiment 4

Experiment 1, 2 , 3 and the real hardware testing demonstrated that the drone-based calibration technique is accurate, efficient and adapted to hard-to-reach cameras. However, in windy conditions, the calibration lacks accuracy, and the proposed solution relies on the use of a robot moving on the ground as a calibration target. The robot is slower than the drone, and considering the very large number of cameras already installed, a very efficient method for the sampling on the ground would represent an advantage for the actual use of this methodology.

Experiment 4 focuses on the mapping image-to-ground and the reduction of the number of sampled points to simplify the robot's path. In Experiment 4, research is done to test whether the trajectory could be simplified, allowing for an efficient calibration. It would allow to reduce calibration time and cost and be adapted to business applications. This work is divided into several steps. First, the baseline set mentioned in Hypothesis 4 is defined, enabling the definition of the individuals of the genetic algorithm. Then, the genetic algorithm is implemented and adapted to the problem tackled in the experiment by defining a robust fitness function. Finally, the genetic algorithm is run for several camera models designed using a Latin hypercube DOE.

7.4 Experiment 4: Baseline set and population

The goal of the genetic algorithm is to reduce the training set size. Thus the training set does not represent the application set anymore. So the accuracy must be evaluated using a validation set that is a homogeneous representation of application sets. For this purpose, a dense validation set is used so that it homogeneously covers the ground. This validation set is represented in Figure 7.2. The estimate of the maximum reprojection error on the ground is used as an accuracy metric so that the complex non-linear behaviors is accounted for in the calibration process.

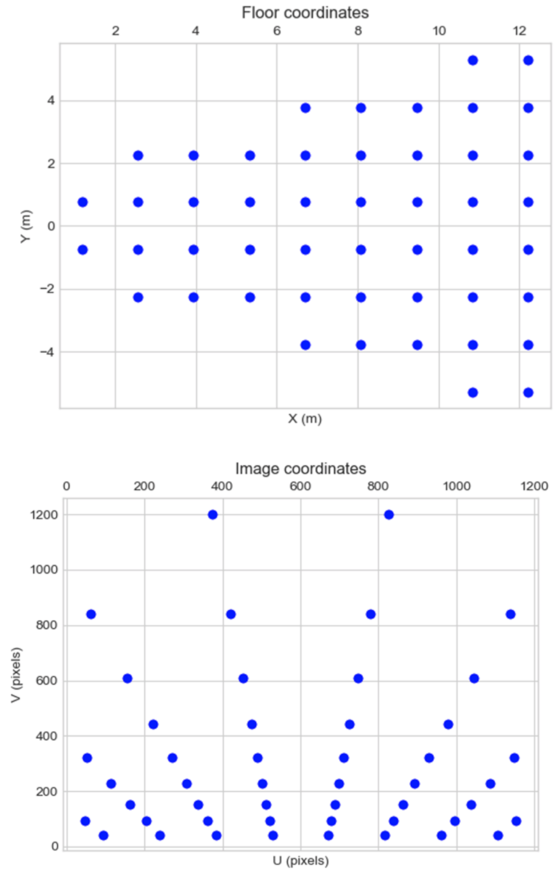


Figure 7.2: Homogeneous coverage of the ground by the validation set

In Experiment 4, a baseline set that enables accurate calibration is first defined. Then, several subsets of this baseline set are tested for the calibration using the genetic algorithm, and more details are provided in the next section. The baseline set must be defined to determine the genetic algorithm chromosomes. The points from this baseline set belong to the projection of the image on the ground. Thus the candidate points for the baseline set are enclosed in a trapezoidal surface on the ground as illustrated in Figure 7.3. On this Figure, the image frame has its origin at the top left corner of the image and the image points coordinates (U,V) are measured in pixels. The ground frame is defined by two orthogonal axes called X and Y in the ground plane and its origin is such that the 3D camera position is (0,0,Z) where Z is the camera elevation.

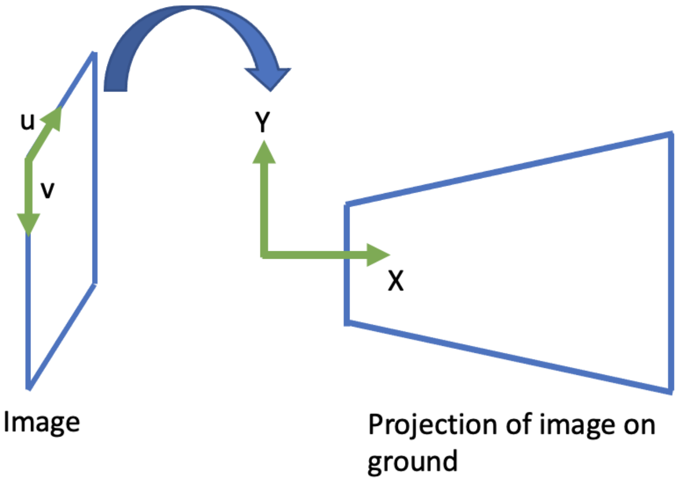


Figure 7.3: Mapping from image to ground

Before running the genetic algorithm, a first comparison measures the error obtained through a simple path and a random path. The simple path is made of a sequence of points organized along a back-and-forth path enabling the robot to follow straight lines. It is compared to random samplings, which require the robot to change directions frequently. Since building a fast calibration approach is a motivation for Research Question 4, the organized sampling is preferred to the random one if the loss of accuracy can be neglected. Figure 7.4 left represents the organized sampling, and Figure 7.4 right represents a random one.

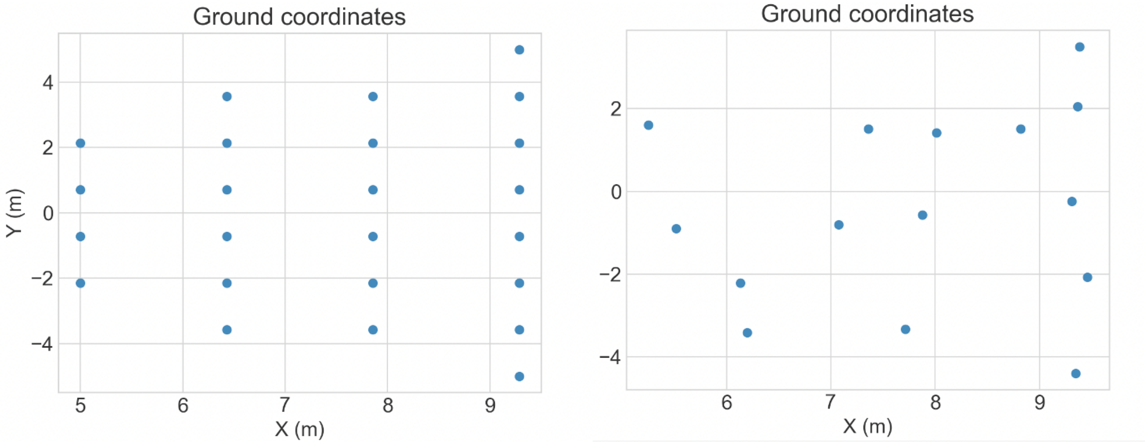


Figure 7.4: Ground samplings: organized on the left and random on the right

The comparison is made through simulations that model different cameras and the error in the measurement of ground points and image points. These simulations are close to the ones described in Chapter 5, but the relation world to image is slightly different, as in Experiment 4 the mapping described the relation between two 2D spaces. Thus, the cost function of the calibration optimization is different, but the steps followed by the optimization are the same as in the Experiment 3. In particular, the uncertainty in 2D and 3D coordinates' measurements is captured with a Monte Carlo sampling. The relation from image to ground is made of 8 unknowns instead of 11 unknowns for the relation 3D to 2D. A point (X, Y, u, v) provides two set of equations to solve for these 8 unknowns. Thus, at least 4 points are required to enable a calibration of the mapping image to ground. Ideally, the calibration would rely on only four points.

The comparison between organized training set and random training set was run for three different cameras for an increasing number of training points. The results are illustrated in Figure 7.5. It can be observed that the organized sampling leads to better accuracy compared to the random one. In addition to being more accurate, the back-and-forth path (also called sequence sampling on the graphs) is more convenient when running the robot, thus it is preferred compared to a random sampling, and it is taken as the baseline set for the genetic algorithm part.

First result of Experiment 4: An organized sampling on the ground leads to a better calibration accuracy compared to a random sampling. Since it leads to a simpler robot path, this trajectory is preferred

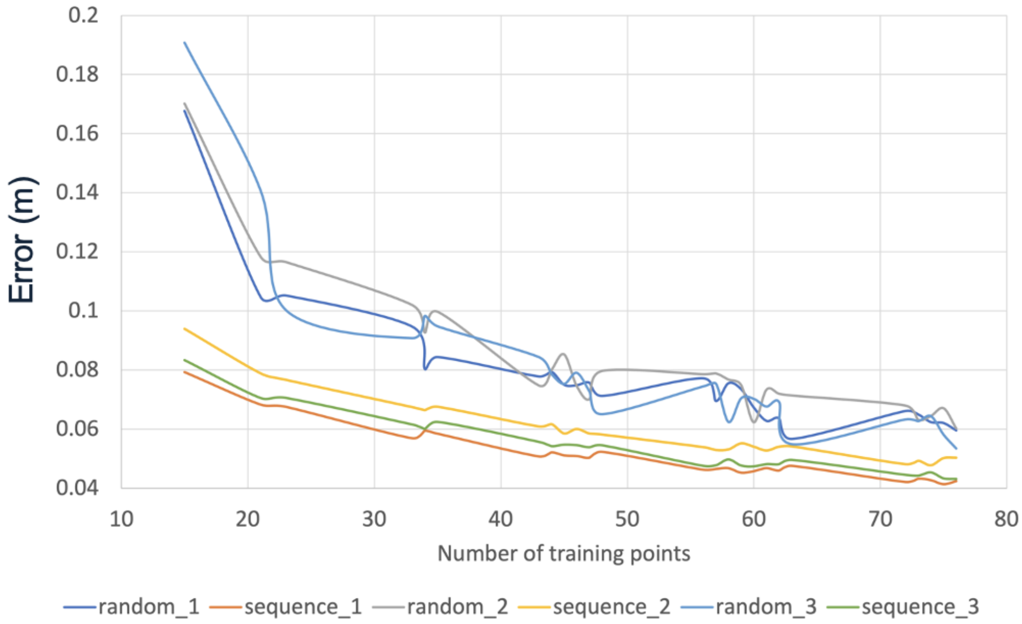


Figure 7.5: Comparison accuracy organized and random sampling

The trend observed is independent of the GPS uncertainty level. Figure 7.6 represents the mapping error for different GPS uncertainty levels when the robot follows an organized sampling. It can be seen that beyond 30 sampled points, there is no major improvement of the mapping. The genetic algorithm optimization approach tests whether this sampling size could be further reduced. Its goal is to evaluate the possibility of reducing the number of sample points and maintaining the mapping accuracy compared to the organized sampling taken as a baseline.

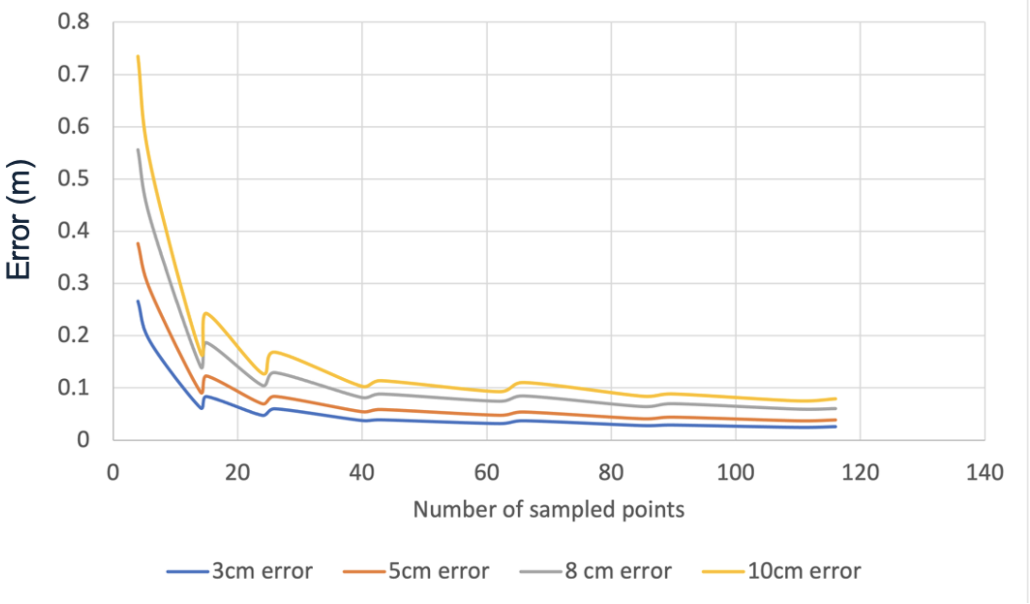


Figure 7.6: Error for GPS uncertainty along X and Y varying between 3cm and 10cm

The next step is the implementation of the Genetic algorithm with the exact definition of its fitness. This step is described in the next section.

7.5 Experiment 4: Robust fitness evaluation

The main contribution of this experiment is the methodology defined to compute the fitness value of the genetic algorithm. The fitness function is a weighted average between the number of genes at 1 in the chromosome, i.e., the number of training points used for the calibration and the accuracy metric. The accuracy metric used in Experiment 4 is the Monte Carlo estimate of the maximum reprojection error on the ground, computed with the validation set. Figure 7.7 illustrates the process followed in computing this metric. The training set is built from the chromosome: if the gene is at 1, the corresponding point is included in the training set. Else, the corresponding training point is not part of the training set. The training set provides ground coordinates (X, Y), which are the input of the function that models a camera. This function returns the image of the training points (u, v). A Monte Carlo sampling is run to capture the measurement uncertainty influence on the calibration from the set of (X, Y, u, v) points (called set T). Noise is added to the set T, for each iteration of the Monte Carlo sampling, the camera calibration optimization is run. The reprojection error on the ground is computed for each validation point from the camera parameters estimated through the calibration process. These validation points are defined before starting the Genetic algorithm process, and are represented on Figure 7.4. They are a perfect set of ground and image points, i.e., no noise has been added to these points.

The maximum reprojection error on the validation set is stored in a list for each iteration of the Monte Carlo sampling. At the end of the sampling, statistics are obtained from the aforementioned list. The average of the list values, i.e. the average of the maximum reprojection error on the validation set obtained at each Monte Carlo sampling iteration, is the accuracy metric used in the fitness. The Monte Carlo sampling allows for a robust evaluation of the accuracy metric while accounting for input measurement uncertainties.

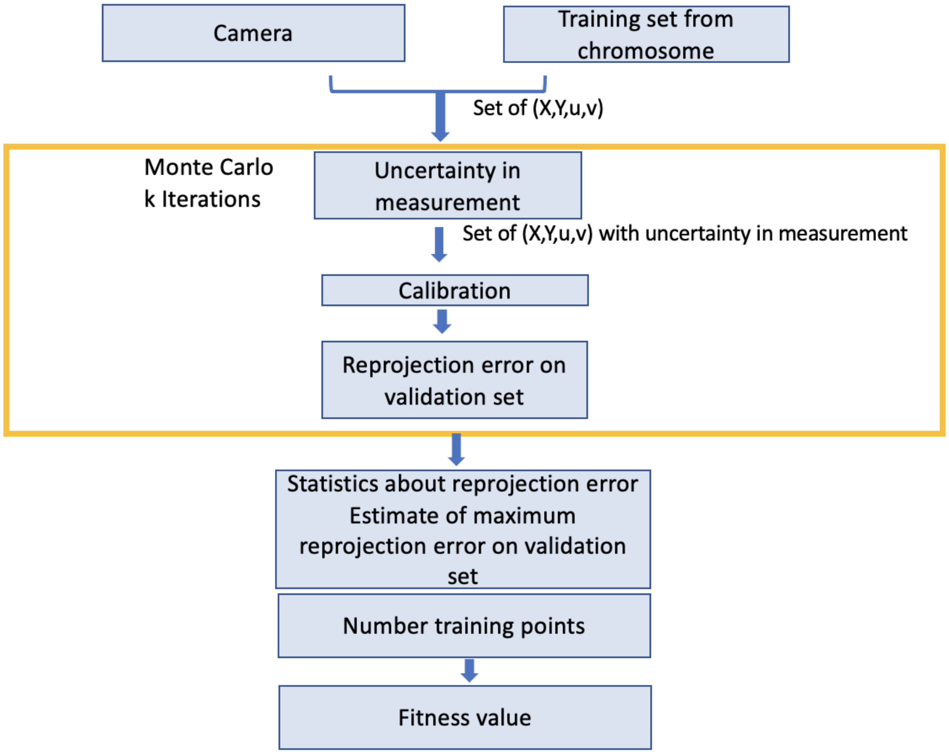

Camera
Training set from chromosome
Set of (X,Y,u,v)
Monte Carlo k Iterations
Uncertainty in measurement
Set of (X,Y,u,v) with uncertainty in measurement
Calibration
Reprojection error on validation set
Statistics about reprojection error Estimate of maximum reprojection error on validation set
Number training points
Fitness value

Figure 7.7: Fitness computation

The weight in the fitness function is determined by manual tuning by looking at the number of sampled points and accuracy metric returned by the genetic algorithm for several cameras. The fitness function in this experiment is equal to: f=5*Number of sampled points + maximum reprojection error. The genetic algorithm was run for 14 camera models. The cameras where modeled using the DOE approach to select their parameters. The fitness weights were chosen to limit the increase in the error due to the reduction of the sampling size. When using the baseline set to calibrate each camera, the average error on the ground is 3.4 centimeters. The threshold for accurate calibration defined in Chapter 3 is 15 centimeters. Here, a stronger constraint is used. The calibration error must not be doubled when reducing the set size, else the sampling size reduction is not worth the loss of accuracy. This condition was satisfied for the tested cameras, if the weight in front of the training set size was taken to 5, and the one in front of the error to 1. This weighted average led to an average error on the ground of 6.14 centimeters when the calibration is made using the reduced sampling set. Thus, the choice of this fitness function leads to an increase in the average error smaller than two times the average error obtained with the baseline set. However, one must test whether keeping a low error on the ground enables to reduce the sampling set size, or whether the number of sampled points is kept high to maintain a low error.

Once the fitness function is defined, it is used in the genetic algorithm's Tournament selection and best individual selection steps. Figure 5.25 illustrates the main steps of the genetic algorithm [163]. On Figure 5.25, the steps highlighted in green are the ones that were adapted for this research purpose. The initial generation is built from the baseline set defined earlier, by selecting combination of points among the baseline set points. The fitness computation use the Monte-Carlo sampling and calibration function, as described in Figure 7.7. The blue steps follow traditional rules for the genetic algorithm, and they are mentioned very briefly in the next paragraph.

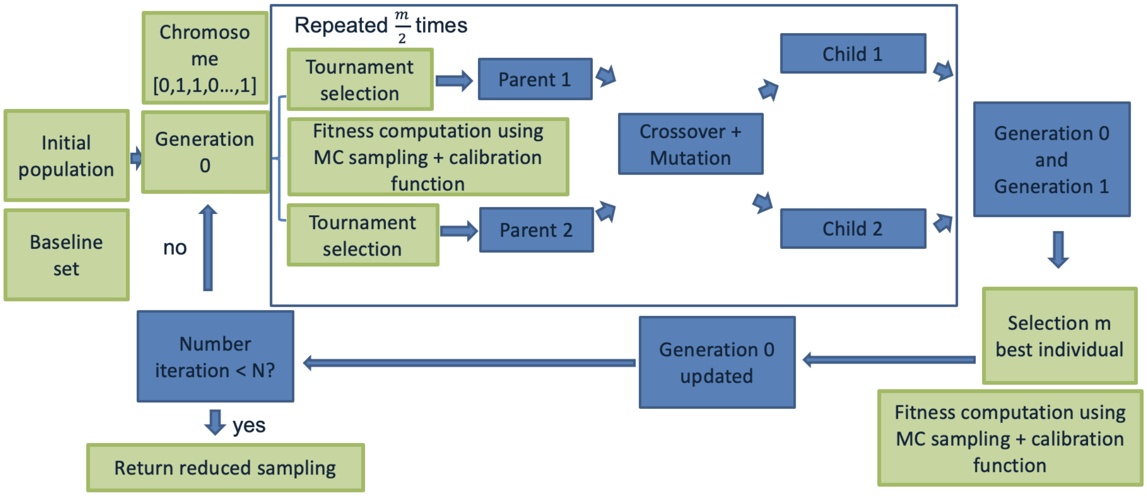


Figure 7.8: Genetic Algorithm steps

First, an initial population of size $m = 2 * n$ is defined, where n is the chromosome size, i.e. the baseline set size. The initial population forms what is called the Generation 0 on the graph: this is the parent generation. Two individuals are randomly selected from the Generation 0, and their fitness value is compared. The individual that leads to the best fitness value is selected for reproduction with probability p=0.8. This process is repeated a second time, leading to two individuals selected by the tournament. These two individuals lead to two children through crossover, which occurs with a probability equal to 0.8, and the crossover point is chosen randomly. Then, some gene are randomly mutated for each children, leading to the final two children called child 1 and child 2 on the chart. The formation of children is done n times, leading to the Generation 1 made of these children. The individuals in Generation 0 and 1 are compared to select the m individuals with the best fitness value, and they form the new Generation 0. This entire process is repeated 500 times. The choice of the genetic algorithm parameters are made based on the recommendations in [165].

The genetic algorithm explores different subsets of points used to build the relation image-to-ground and returns a subset that best satisfies the trade-off between number of sampled points and relation accuracy. The results obtained with cameras modeled in simulations are presented in the following section.

7.6 Experiment 4: results for several cameras

The genetic algorithm explores subsets of points among the baseline set to find a training set, that is made of a reduced number of points, while ensuring accurate calibration. The use of a smaller training set would help reducing the time of use of the robot, and thus the calibration time.

The first test is made for a pinhole camera with focal length 1100 pixels, image width 1200 pixels, image height 1200 pixels, camera elevation 4 meters and camera orientation along the Y axis equal to 45 degrees.

The validation set is represented on Figure 7.9. The top figure represents the ground validation point, and the bottom figure represents their images captured by the camera. The optimization though the genetic algorithm returns 4 points represented in red on Figure 7.10. On the same Figure, the set made of red and blue points together represents the baseline set. The maximum reprojection error on the ground equals 4 centimeters before running the genetic algorithm, i.e. when the mapping is built with all points from the baseline set. The maximum reprojection error on the ground, measured on the validation set, is equal to 7 centimeters once the genetic algorithm is run. Both errors are smaller than the threshold of 15cm used to evaluate the calibration quality. So the reduced set leads to a successful calibration.

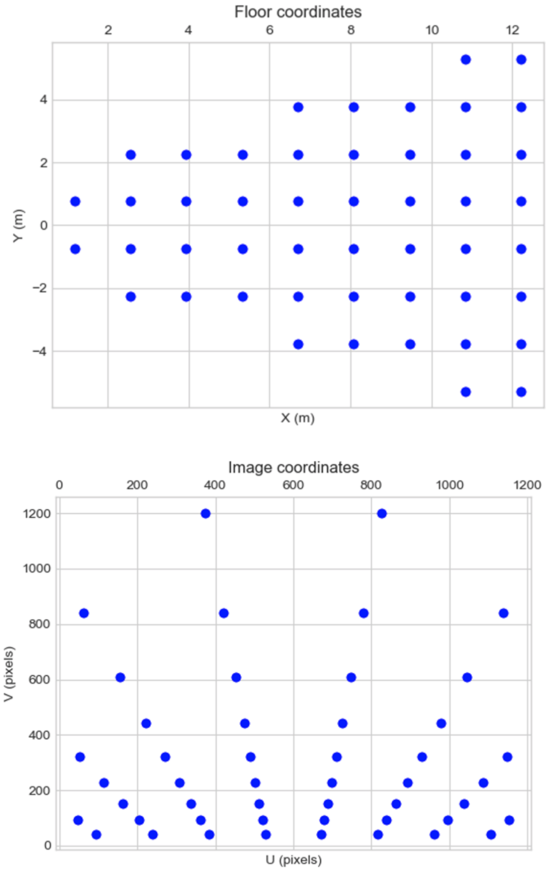


Figure 7.9: Ground and image validation points

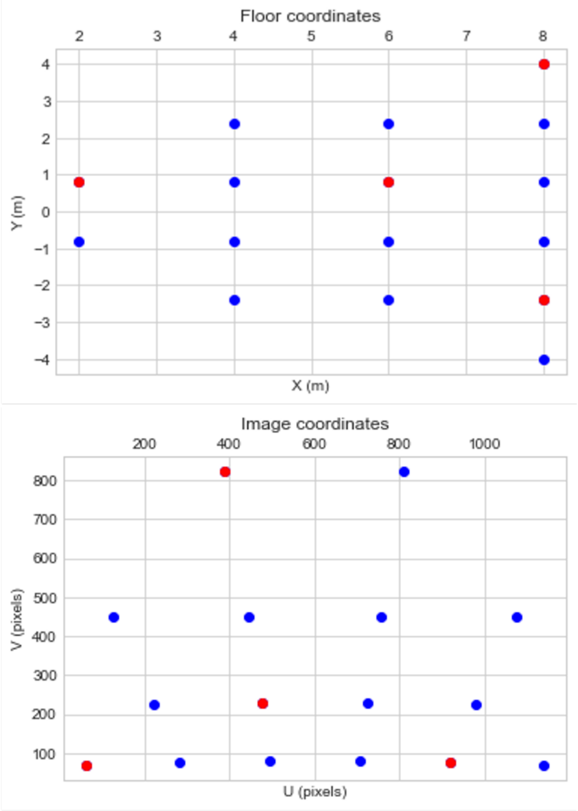


Figure 7.10: Result of the genetic algorithm

The experiment was run for various camera parameters. First it was run for several orientation angle values of the aforementioned camera, leading to reduced number of training points, and close calibration error value between the baseline set and the set returned by the GA. For a rotation along Y of 60 degrees, the number of sampled points is reduced to 4. The maximum reprojection error, computed on the validation set, when using the baseline set for training, is equal to 2 centimeters. When using the reduced training set it is equal to 3 centimeters. Both errors are smaller than the threshold of 15 centimeters so the calibration is successful. The selected ground points and their corresponding image points are represented in red on Figure 7.11.

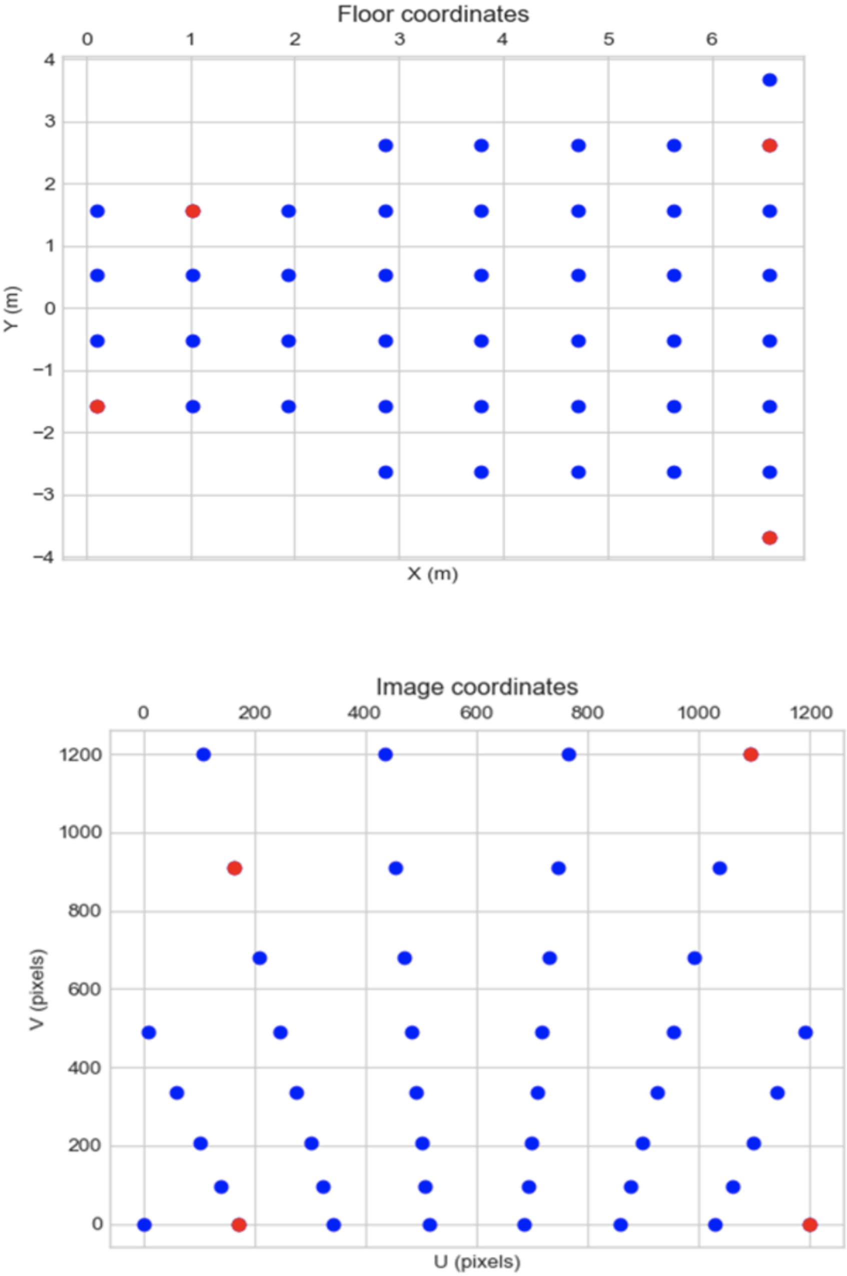


Figure 7.11: Result of the genetic algorithm for an angle along Y of 60 degrees

The experiment was also run for different focal length values. Figure 7.12 represents the result obtained for a focal length of 900 pixels. The maximum reprojection error when using the baseline set for training is the same as the one obtained when using the reduced set for the training, represented in red on the Figure.

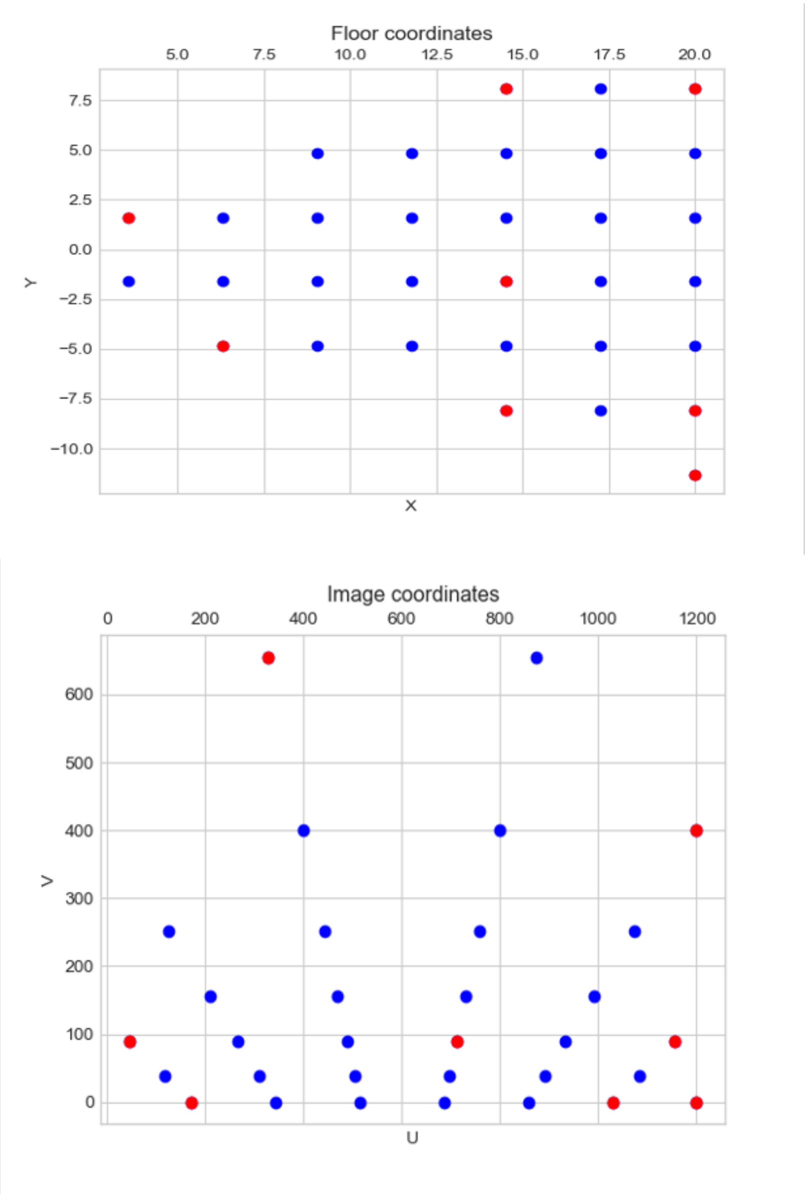


Figure 7.12: Sampling found by the GA (in red) compared to baseline sampling (blue and red)

The experiment was run for cameras with distortion, and an example of results is provided on Figure 7.13. The trend observed for pinhole cameras is different from the one observed for cameras with distortion. For cameras with large distortion, the reduced sampling captures points on the sampling space extremities, in the same way as for pinhole cameras. But on the contrary to pinhole cameras, a key portion of the points selected by the genetic algorithm are located at the sampling space center. The non linearity introduced by the distortion is likely responsible for this behavior.

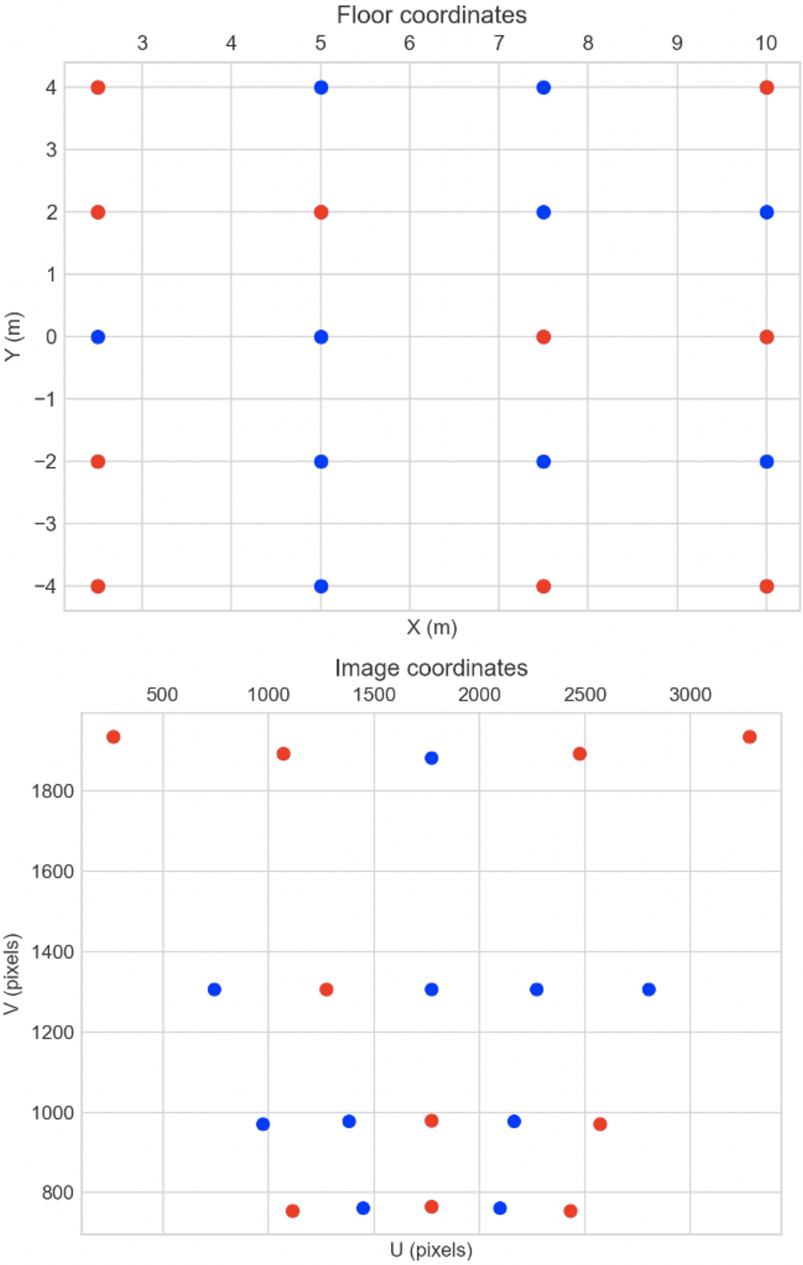


Figure 7.13: Sampling found by the GA (in red) compared to baseline sampling (blue and red)

7.7 Validation Hypothesis 4

In average, the maximum reprojection error on the validation set obtained for the calibration that uses the reduced sampling returned by the genetic algorithm as training set is equal to 6.14 centimeters for the tested cameras.

The number of points to sample returned by the genetic algorithm is equal to four in most tested cases where the camera is pinhole. When comparing to the baseline set size, the genetic algorithm allows a reduction of the number of sample points by 83 percent. The results observed for cameras with large distortion is not as good. The average size of the reduced sampling set, for the tested cameras, is 30 percent smaller compared to the baseline set size. Figure 7.14 illustrates the average sampling size, in percent of the baseline set size, obtained through the process used in Experiment 4.

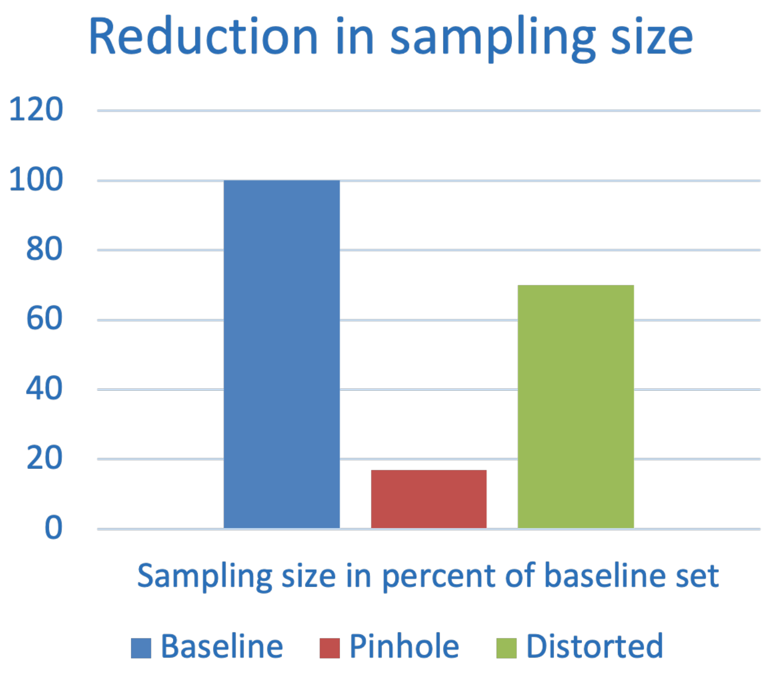


Figure 7.14: Sampling size reduction obtained with the Genetic Algorithm, for pinhole cameras and cameras with distortion

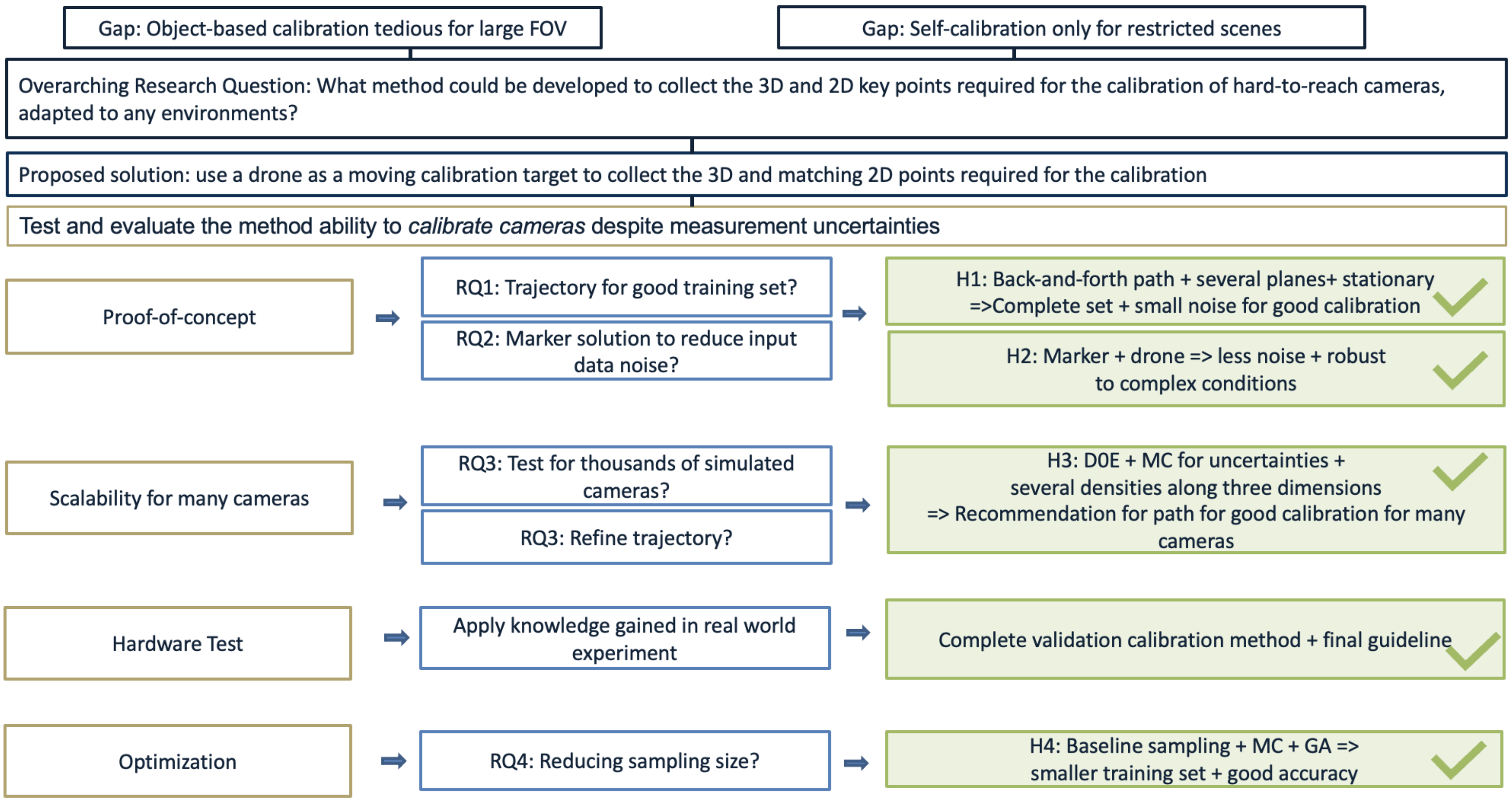
Main result of Experiment 4: The average reprojection error on the ground equals 6.14 centimeters for the tested cameras when calibrating these cameras with the reduced training set. The average size of the reduced training set is 56 percent smaller compared to the baseline set.

The number of points is significantly reduced and the average calibration error with the reduced sets is kept below twice the error with the baseline set. This error is also smaller than the threshold of 15 centimeters defined in Chapter 3. The number of points to sample is reduced thanks to the use of the genetic algorithm and so the trajectory time is reduced. **Hypothesis 4, recalled below, is validated.**

HYPOTHESIS 4 IS VALIDATED: IF a genetic algorithm is implemented, with a Monte Carlo sampling to capture uncertainty, and a chromosome that represents the use of a baseline point or not, THEN a reduced sampling that ensures accurate calibration can be identified

Further analysis is done to look for a pattern between camera parameters and reduced training set. This work is presented in the next section.

The chart on next page summarizes the main gaps, the overarching research goal, the research questions and hypotheses validated in this thesis.



Gap: Object-based calibration tedious for large FOV

Gap: Self-calibration only for restricted scenes

Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?

Proposed solution: use a drone as a moving calibration target to collect the 3D and matching 2D points required for the calibration

Test and evaluate the method ability to *calibrate cameras* despite measurement uncertainties

Proof-of-concept ➡ RQ1: Trajectory for good training set? RQ2: Marker solution to reduce input data noise? ➡ H1: Back-and-forth path + several planes+ stationary =>Complete set + small noise for good calibration ✓ H2: Marker + drone => less noise + robust to complex conditions ✓

Scalability for many cameras ➡ RQ3: Test for thousands of simulated cameras? RQ3: Refine trajectory? ➡ H3: DOE + MC for uncertainties + several densities along three dimensions => Recommendation for path for good calibration for many cameras ✓

Hardware Test ➡ Apply knowledge gained in real world experiment ➡ Complete validation calibration method + final guideline ✓

Optimization ➡ RQ4: Reducing sampling size? ➡ H4: Baseline sampling + MC + GA => smaller training set + good accuracy ✓

7.8 Experiment 4: looking for a common pattern

The complexity of the genetic algorithm defined for Experiment 4 is high, and thus the run time of the algorithm was slow. Consequently, the optimization approach of Experiment 4 is not tested for the 5000 cameras modeled in Experiment 3. The approach has been tested for an increasing number of cameras until conclusions could be identified about the sampling pattern returned by the genetic algorithm.

The optimization approach is tested on 14 cameras (7 pinhole cameras and 7 cameras with distortion), whose parameters are designed with a Latin Hypercube DOE in the same way as in Experiment 3, and are provided in Table 7.1. The optimization return a training set of reduced size.

Table 7.1: Camera parameters design space

camera id	focal length (mm)	image width (pixels)	elevation (m)	orientation (degree)
0	11.2957138	2977	3.63455932	32.527073
1	8.6718969	781	2.1525422	47.5879831
2	5.66768784	1860	4.9564201	66.6233502
3	13.3174566	3544	4.50139815	42.7456446
4	9.60559733	1195	3.8879974	76.9556946
5	7.469889	3078	3.10262839	58.7508124
6	2.25634501	2153	2.84824918	83.0400908

The reduce training set returned by the genetic algorithm for the pinhole cameras are presented on Figure 7.15 .

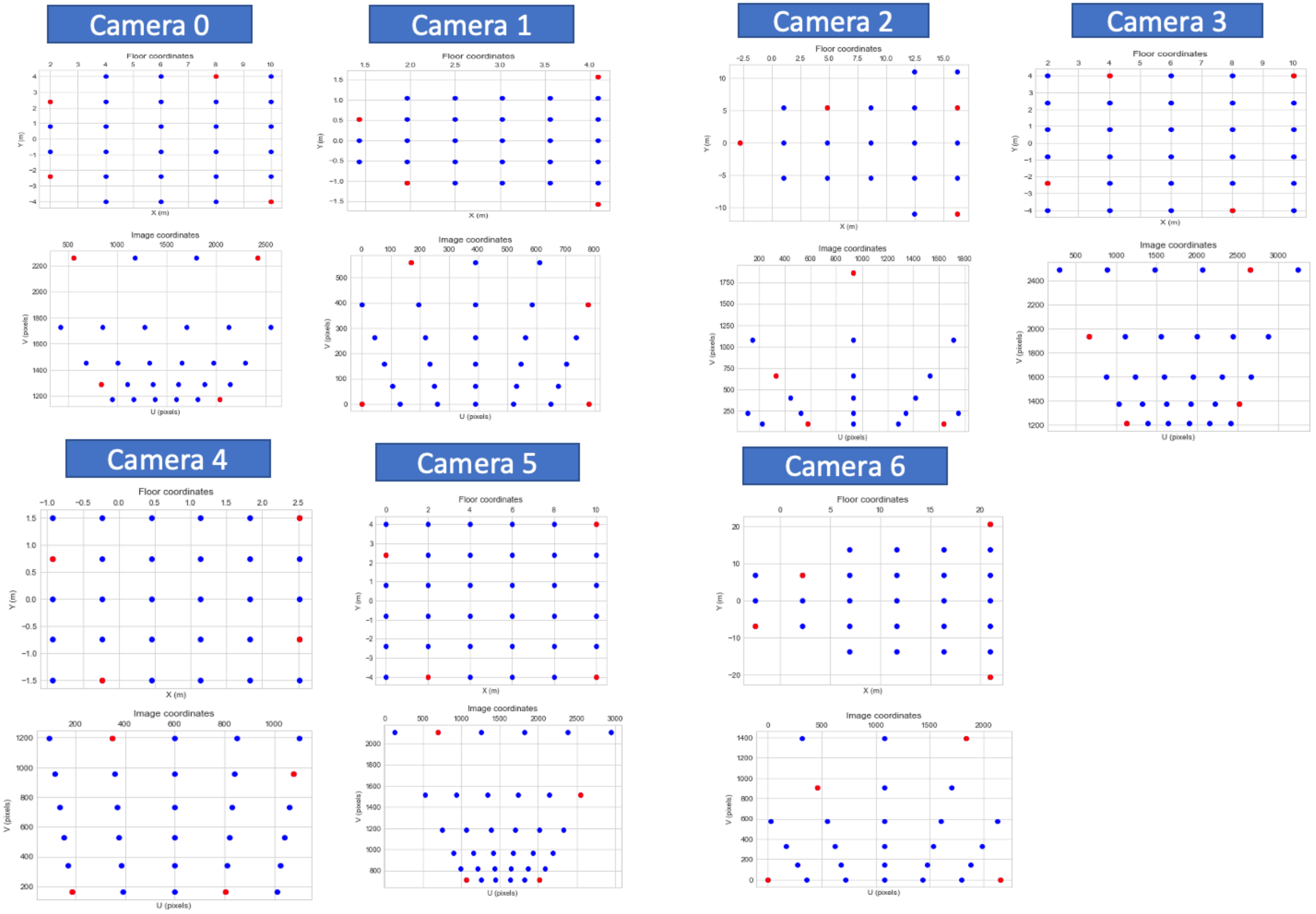


Figure 7.15: Reduced training set for pinhole cameras

It can be observed that the genetic algorithm always returns two points located on the farthest row of the baseline set, and two points located closer to the camera; on the first or second row. The influence of GPS uncertainties on the calibration quality decreases if the distance between sample points and camera increase. This trend was already observed for the calibration 3D to 2D in Experiment 3. This trend is a potential explanation for the genetic algorithm behavior: sampling points farther from the camera helps reducing the noise caused by GPS measurement errors. When fitting a linear model from the the minimum number of training points required, the regression model accuracy can be decreased by taking points far apart. Indeed, the uncertainty in measurement is random, thus taking points close to each other might lead to a larger error in the regression model, compared to taking points farther apart. This observation is illustrated in Figure 7.16, where two points are used for a linear regression to estimate a line equation. The ground truth for the line equation is the green line. Points A_1 and B_1, are close to each other (abscissa close), the line equation obtained when using A_1 and B_1 for training is represented in orange. Points A_2 and B_2, are farther to each other (larger difference for abscissa) and have the same measurement error than A_1 and B_1. The line equation obtained when using A_2 and B_2 for training is represented in blue. It can be observed that the points A_2 and B_2 lead to a better estimate of the line, compared to A_1 and B_1. Thus, when fitting a linear model with the minimum number of required training points, and these points are under the influence of an homogeneous noise, then taking points far apart will likely decrease the error on the regression. This observation is a potential justification for the genetic algorithm behavior with pinhole cameras. The latter collects points far away to reduce the impact of GPS uncertainty on the calibration, and complete these points with points located near the camera, i.e. far from the previously sampled points, to decrease the resulting error on the regression model.

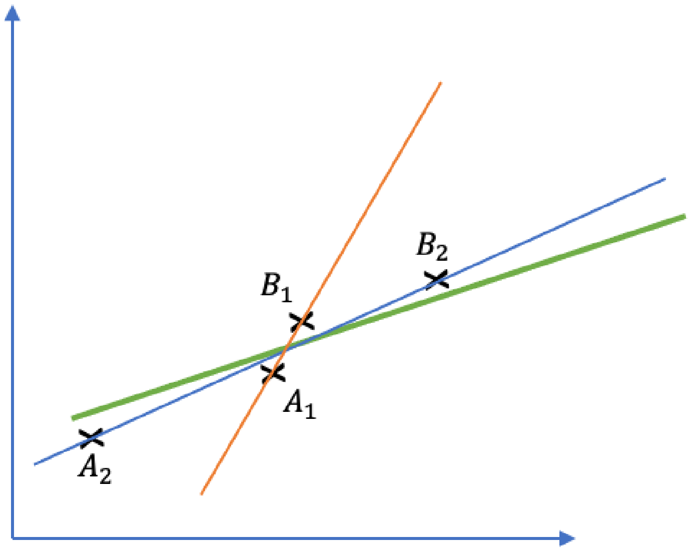


Figure 7.16: Impact of distance between training points for linear model fitting under uncertainty

The behavior of the genetic algorithm, that picks two training points far from the camera, two training points close to the camera, and none in the middle of the baseline set, can not be generalized to cameras with distortion. Indeed, the lens distortion causes a non-linear behavior that can not be captured with only four points. The results obtained for cameras with distortion are presented in Figure 7.17. It seems that all region must be covered by the training set to make sure the linear behavior at the center and the non-linear behavior at the image extremity are accounted for. Thus, there are no general patterns observed for cameras with distortion.

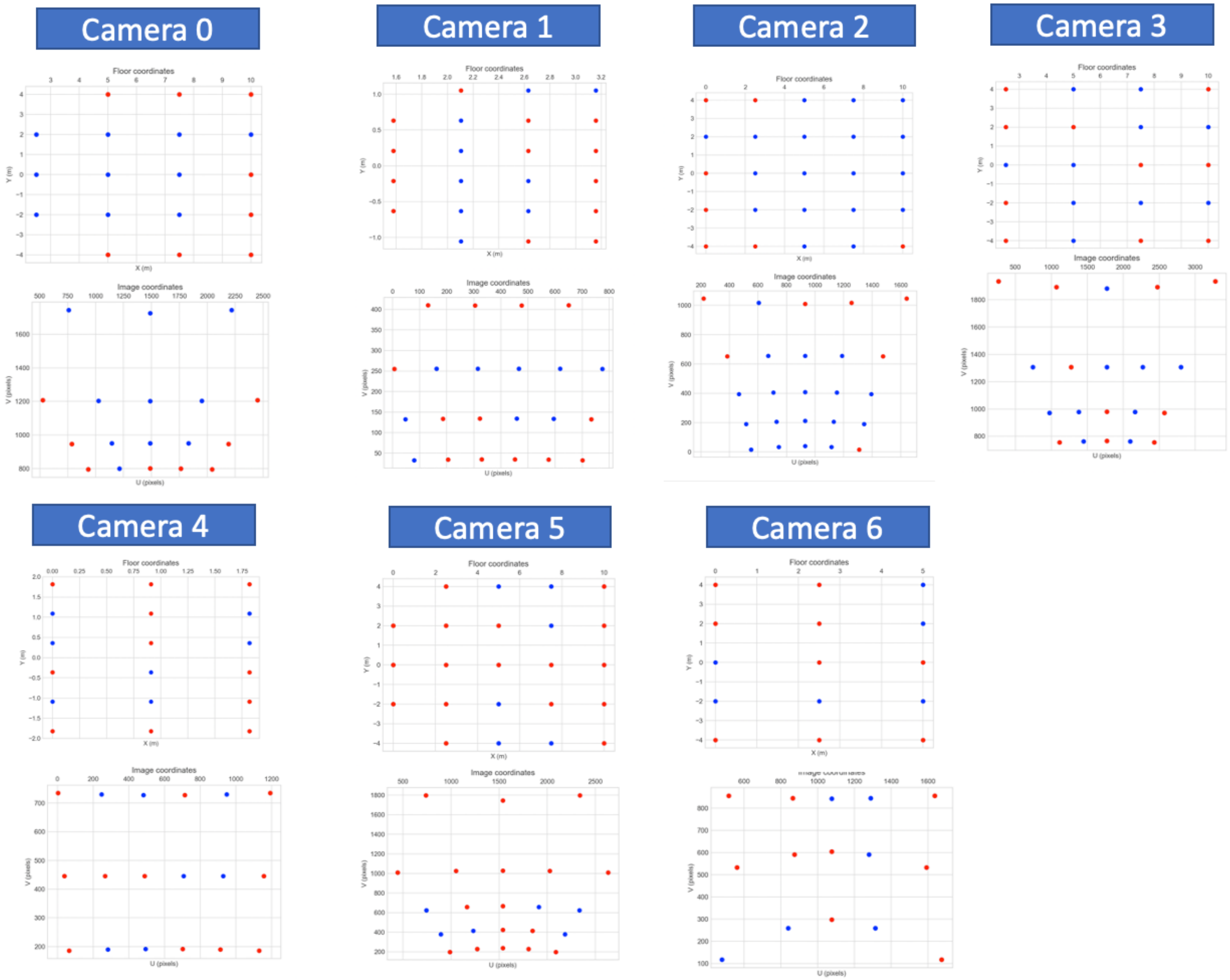


Figure 7.17: Reduced training set for cameras with distortion

There is no common pattern for the points selected by the Genetic Algorithm for the tested cameras that have distortion. It means that the camera parameters must be estimated to run the offline optimization, and get a path with reduced set of points. On the one hand, the genetic algorithm approach leads a reduced number of points to sample compared to the baseline approach. On the other hand, using the offline optimization approach requires pre-processing time, which balances the advantage of the faster path.

7.9 Experiment 4: application to a real camera

In order to provide a methodology to reproduce this offline optimization for the calibration with a reduced sampling, a test was done with a real camera. The paragraph below presents how the offline approach was run for a webcam at Georgia Tech Lorraine. First, the camera extrinsic parameters were estimated by using a satellite view of the area where the camera is located, obtained through Google map. Figure 7.18 illustrates this view, the X_w, and Y_w world coordinates represent the ground plane, and Z_w is vertical up.

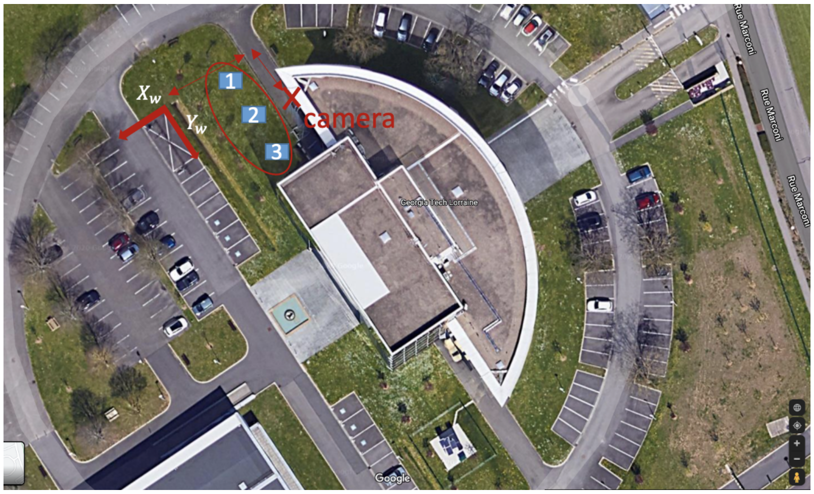


Figure 7.18: Satellite view of the camera area

The image captured by the camera was used to estimate its world coordinates. The trees seen through the camera were also identified through the satellite view to infer the camera location on the satellite view. The indices 1, 2 and 3 seen on Figure 7.19 highlight these trees' location. The outdoor light height seen through the camera was also used to infer the camera height. The camera world coordinates obtained with this process are the following: X_w=-16m, Y_w=9m, and Z_w=6m.

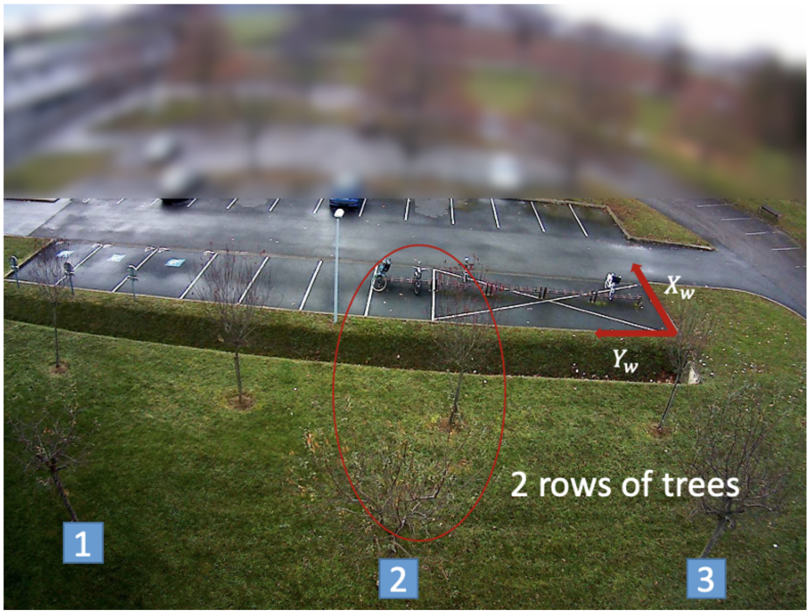


Figure 7.19: Image captured by the camera

Once the extrinsic parameters were estimated, the camera was modeled, points that are aligned on the world frame, i.e. on the ground were modeled and their image printed on the camera view. The intrinsic parameters were then manually tuned to align these lines to the actual lines seen through the camera, which are the parking spot white markers. Figure 7.20 illustrates this alignment process.

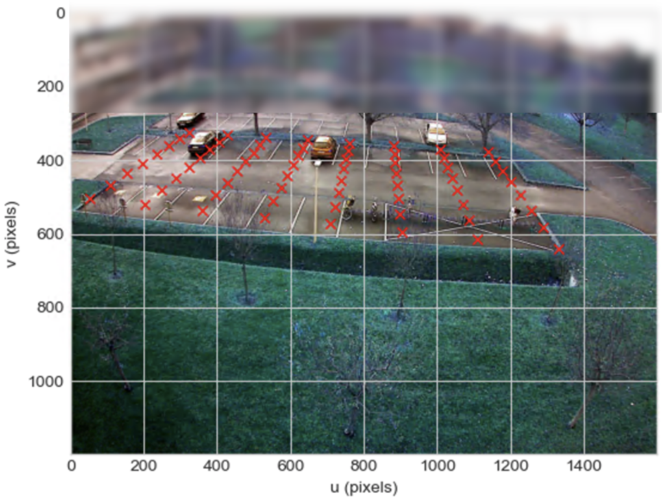


Figure 7.20: Tuning of intrinsic parameters by aligning modeled points to actual lines

Once the parameters are estimated, the camera is modeled in the simulation, and the Genetic algorithm is run.

The genetic algorithm returned the red sampling on Figure 7.21, while blue and red points together represent the baseline sampling. The calibration accuracy was evaluated with the Maximum reprojection error on the validation set, which equaled 16 centimeters.

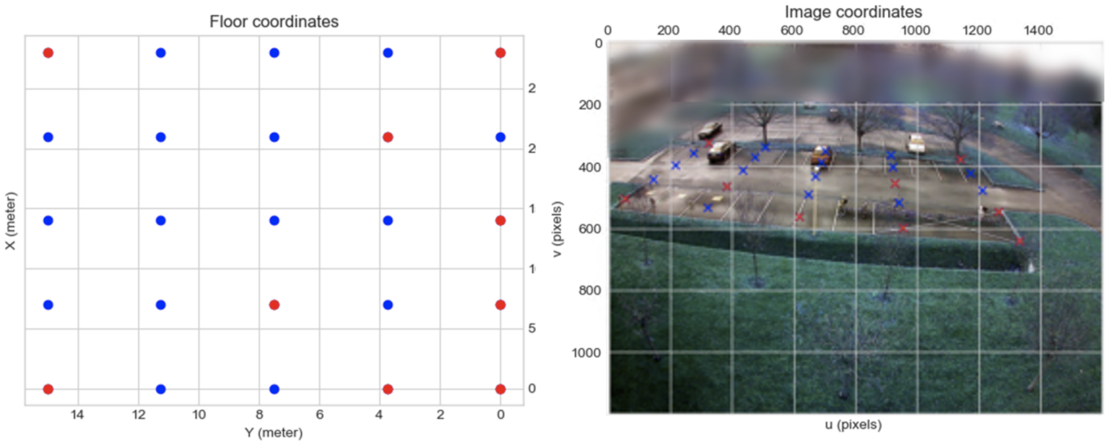


Figure 7.21: Reduced number of sampled points for the calibration of the camera

The requirement of the pre-processing step to estimate the camera parameters is an important limit of this approach. It is suited for cameras that have already been calibrated once, and need to be recalibrated due to internal parameter changes caused by weather condition or vandalism. Then, an estimate of the camera parameters is available, and the Genetic algorithm can be run offline to optimize the autonomous target path. For cameras that are calibrated for the first time, the best approach is to cover the entire ground, since it would spare the pre-processing which is time consuming and introduce approximation errors. Also, if the robot covers a large number of points on the ground, then a Machine Learning approach can be implemented for the calibration, which would model unexpected camera parameters' changes.

A second approach was investigated because of the drawback of the offline optimization approach for cameras with distortion, which requires a first estimation of the camera parameters. This approach relies on an adaptive sampling: first, the robot visits four sample points close to the image center, and then, it explores points in an autonomous way. The next point visited by the robot is chosen based on the calibration uncertainty obtained when using the previously sampled points, and a measure of gain of information. This method requires the access to the video footage in real-time, for an online calibration made using the ground location visited by the robot and its image through the video. The work on this approach was done in collaboration with Dr Pradalier and will be published in 2021.

7.10 Summary

This Chapter presented the last step of this thesis work. This step focused on the optimization of the moving target trajectory to ensure a simple setup, low cost, and accurate calibration. Research Question and Hypothesis 4 focused on the mapping image-to-ground and the reduction of the number of sampled points to simplify the robot's path. Experiment 4 demonstrated the existence of a small size training set for the calibration of the mapping from the image to the ground while allowing accurate calibration. For pinhole cameras, the reduced sampling is made of four points, with two points located far from the camera, and two points located near to the camera. This section also presented an application of the sampling strategy on a real camera. For pinhole cameras, the reduced sampling captures also points located in the middle of the design space. It was explained that for distorted cameras, using the baseline set with the robot moving along straight lines is preferred compared to the reduced sampling.

This last experiment completed the Research plan followed in the thesis and recalled on Figure 7.22.

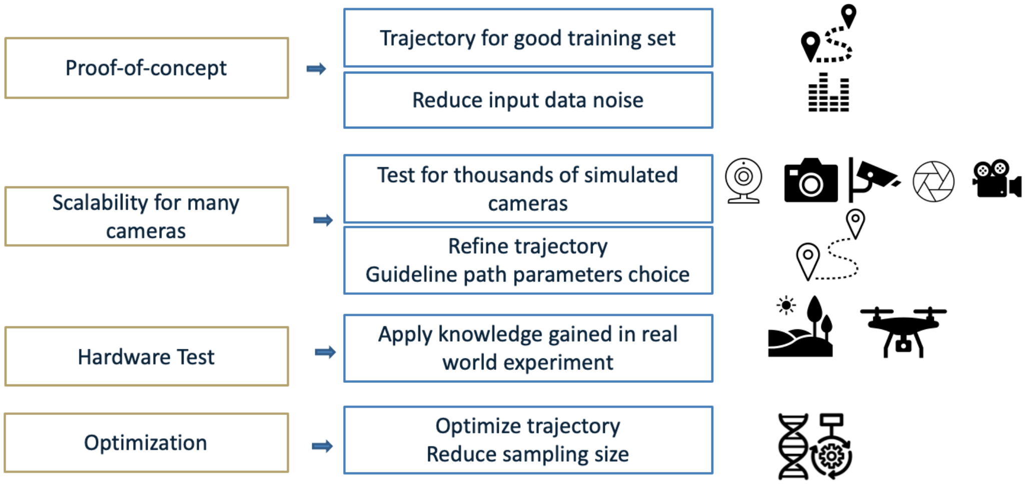

Proof-of-concept
Trajectory for good training set
Reduce input data noise
Scalability for many cameras
Test for thousands of simulated cameras
Refine trajectory
Guideline path parameters choice
Hardware Test
Apply knowledge gained in real world experiment
Optimization
Optimize trajectory
Reduce sampling size

Figure 7.22: Research Plan: proof-of-concept, scalability to many cameras, real testing, optimization

The chapters were articulated around Research Questions originating from Gaps and Hypotheses that were proposed solutions to these Research Questions. Each Chapter presented the Experiments designed to test the Hypotheses and their Results. First, A proof-of-concept was designed to evaluate the feasibility of the approach. This step tested whether the calibration is successful despite restrictions on the 3D sampling and 2D detection uncertainty (RQ 1). In order to capture the complex behaviors due to non-linearity in the model, the 3D sampling must follow the steps described in Hypothesis 1. Once the 3D key points and their corresponding image coordinates are collected, the camera is calibrated, and its error is compared to the threshold for successful calibration. The calibration error was below the threshold of 3.6 pixels, thus Hypothesis 1 was validated. Experiment 1 provided strong evidence for the approach's ability to calibrate hard-to-reach cameras.

Experiment 2 evaluated the feasibility of a solution to reduce the noise during the image detection step. It evaluated the possibility of adding a marker on the drone, while satisfying constraints due to commercial drone dimensions. Hypothesis 2 tested the feasibility of the marker-drone combination for the calibration. The marker-drone solution was proven robust against blurring due to small drone oscillations, distortion, challenging background. This solution respected the restrictions of dimensions and visibility at 3 meters. Since the marker-drone solution satisfied all criteria defined in Hypothesis 2, the later was validated.

In a second step, the calibration technique is evaluated for many cameras. The uncertainty due to RTK GPS measurements is included in the evaluation process. Experiment 3 demonstrated the impact of path parameters and camera parameters on the calibration accuracy. It resulted in recommendations for the drone trajectory to enable a successful calibration for many cameras. Based on this result, Hypothesis 3 was validated.

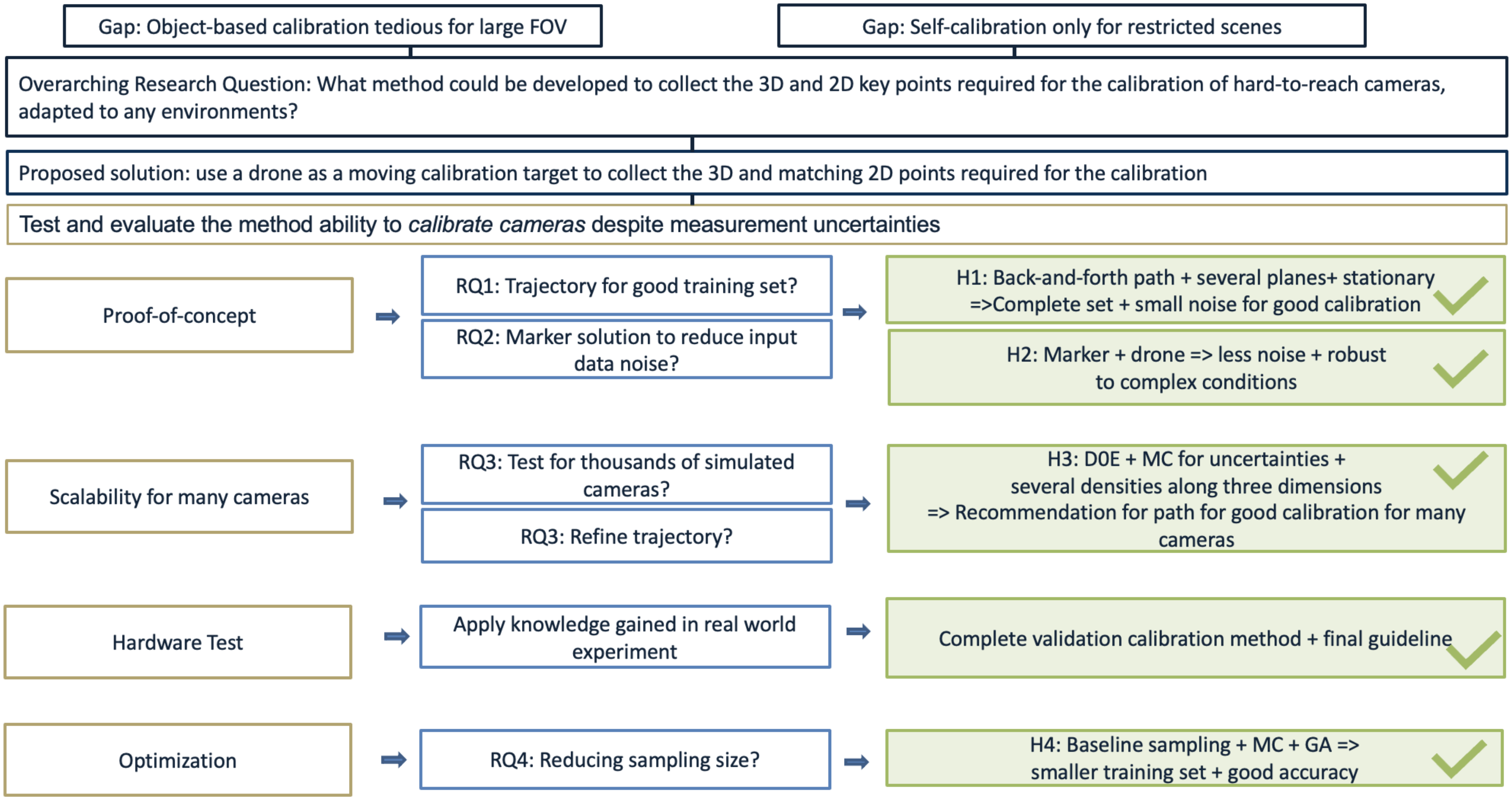
In a third step, the knowledge gained from the three first experiments is implemented with real hardware. The calibration is successful, and a use case for obstacle localization is demonstrated. This real-world testing showed the ease of deployment of the calibration technique and resulted in the definition of a methodology to reproduce the drone-based calibration.

Finally, the last step investigated the possibility to reduce the sampling set size for the mapping image to ground, while obtained a small calibration error. The sampling size can be reduced while keeping good accuracy, and Hypothesis 4 was validated using an optimization process based on a genetic algorithm.

The experiments of this thesis focused on closed range camera-based measurements, such as obstacle localization or vehicle speed measurement from security cameras. Enabling the calibration of remote sensing cameras on static platform is a key interest when developing this calibration strategy. This would allow for an accurate measurement of environmental changes, such as actual glacier size decrease. That is why this thesis experiments focused on closed range camera-based applications.

The calibration using a moving target that can cover the camera field-of-view can also be applied to the observation of objects from a larger distance, such as with satellite cameras. This use case is introduced in the next Chapter.

The figure on the next page depicts the flow of this research and summarizes its main results.



Gap: Object-based calibration tedious for large FOV

Gap: Self-calibration only for restricted scenes

Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras, adapted to any environments?

Proposed solution: use a drone as a moving calibration target to collect the 3D and matching 2D points required for the calibration

Test and evaluate the method ability to *calibrate cameras* despite measurement uncertainties

Proof-of-concept ⇒ RQ1: Trajectory for good training set? RQ2: Marker solution to reduce input data noise? ⇒ H1: Back-and-forth path + several planes+ stationary =>Complete set + small noise for good calibration ✓ H2: Marker + drone => less noise + robust to complex conditions ✓

Scalability for many cameras ⇒ RQ3: Test for thousands of simulated cameras? RQ3: Refine trajectory? ⇒ H3: DOE + MC for uncertainties + several densities along three dimensions => Recommendation for path for good calibration for many cameras ✓

Hardware Test ⇒ Apply knowledge gained in real world experiment ⇒ Complete validation calibration method + final guideline ✓

Optimization ⇒ RQ4: Reducing sampling size? ⇒ H4: Baseline sampling + MC + GA => smaller training set + good accuracy ✓

CHAPTER 8
USE CASE: CALIBRATION OF SATELLITE CAMERAS WITH A MOVING TARGET

A key application of remote sensing is environmental change monitoring through satellite sensors. Imaging obtained through satellite cameras enables us to understand climate change by providing pictures of major environmental events such as urbanization, vegetation changes, storms or pollution. The imaging system used on these satellites is pre-calibrated before launching the satellite. However, the conditions during the spacecraft launch, such as significant changes in air pressure or temperature, impact the camera geometry. Thus, these cameras are re-calibrated once they are in orbit. The conventional method relies on the use of ground control points as a calibration target. However, weather conditions, such as cloudy weather, prevent seeing the ground points on the satellite imaging system.

This thesis calibration technique can be applied to the calibration of satellite cameras and solve the issue of occluded ground control points. Two solutions are proposed and described in the two following sections.

8.1 First solution: airplane as a moving calibration target

This proposed solution relies on the use of an airplane, or a large helicopter, as a moving calibration target. The drone is replaced by an airplane, on the top of which a marker would be printed, as recommended in Experiment 2. The airplane would fly in the satellite field-of-view as represented on Figure 8.1. A radar located on the ground would provide an accurate information for its world coordinates. Figure 8.2 depicts the 3D localization system recommended in this use case. The plane, or its marker, would be detected on the images obtained from the satellite camera. Thus, the moving target would allow the collection of 3D and 2D key points required for the calibration. This moving target would replace the ground control points used for the calibration of satellite cameras. Because the target is moving and can take-off at anytime, the occlusion caused by weather conditions would not be a drawback for the calibration of satellite cameras anymore. The next section proposes a different solution to the calibration problem for satellite cameras. The solution relies on multiple moving calibration targets located on orbit.

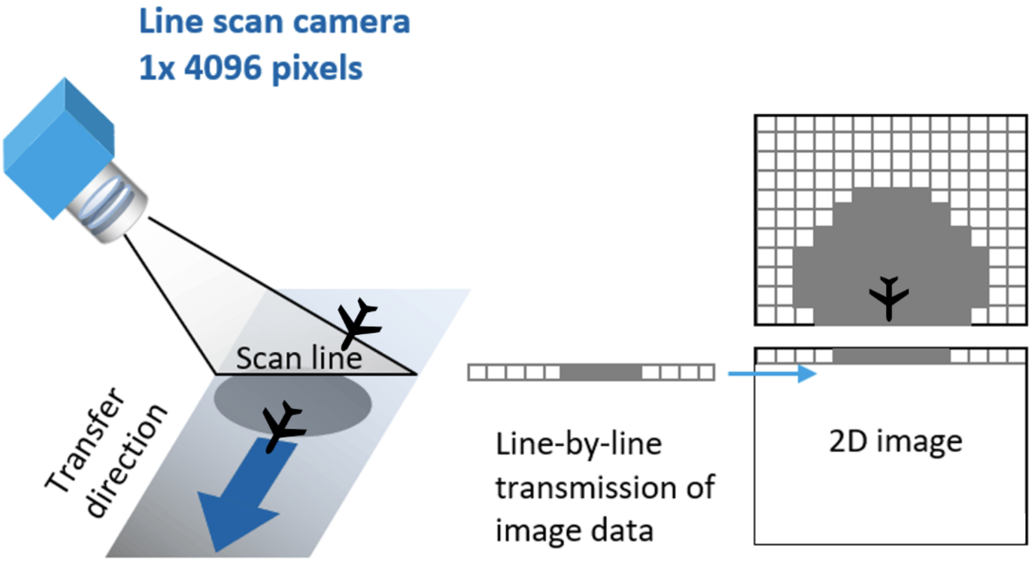


Figure 8.1: Airplane as a moving calibration object: 2D key points collection

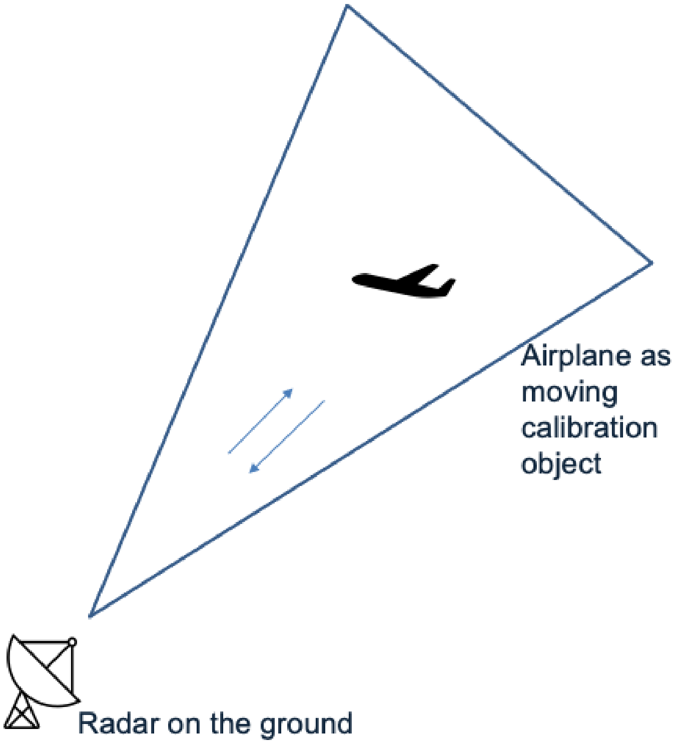


Figure 8.2: Airplane as a moving calibration object: 3D key points collection

8.2 Second solution: Cubesat as moving calibration targets

This proposed solution relies on the use of autonomous control points visible by the satellite camera and orbiting on a lower orbit compared to the satellite. The autonomous control points would replace the ground control points and be visible when needed for the re-calibration because their location on-orbit would make their detection independent from environmental conditions. The same set of autonomous control targets could be used by many satellites as long as they are visible by their cameras, at a given time, during their rotation on their orbit. This solution would then be adapted to new satellites and to satellites already orbiting around Earth, as long as they can see the targets at some point.

This use case is an application of this thesis calibration approach as it relies on the use of moving autonomous calibration targets.

When a satellite would be launched in orbit, it would launch at the same time a set of

CubeSats that would become the calibration targets. The cost of launching these CubeSats with the satellite is negligible compared to the cost of launching the satellite, which makes this solution cost-effective. The CubeSats would orbit on a lower orbit compared to the satellite, as illustrated Figure 8.3. The rotation period of a satellite on an orbit of radius R is given by $\frac{T^2}{R^3} = \frac{4*\pi^2}{G*M_{Earth}}$. Thus, the CubeSats period would be smaller than the satellite's one, as the radius where they are located is smaller (lower orbit). Consequently, the satellite would see the calibration targets several times in one rotation period and thus its camera can be re-calibrated several times during a rotation.

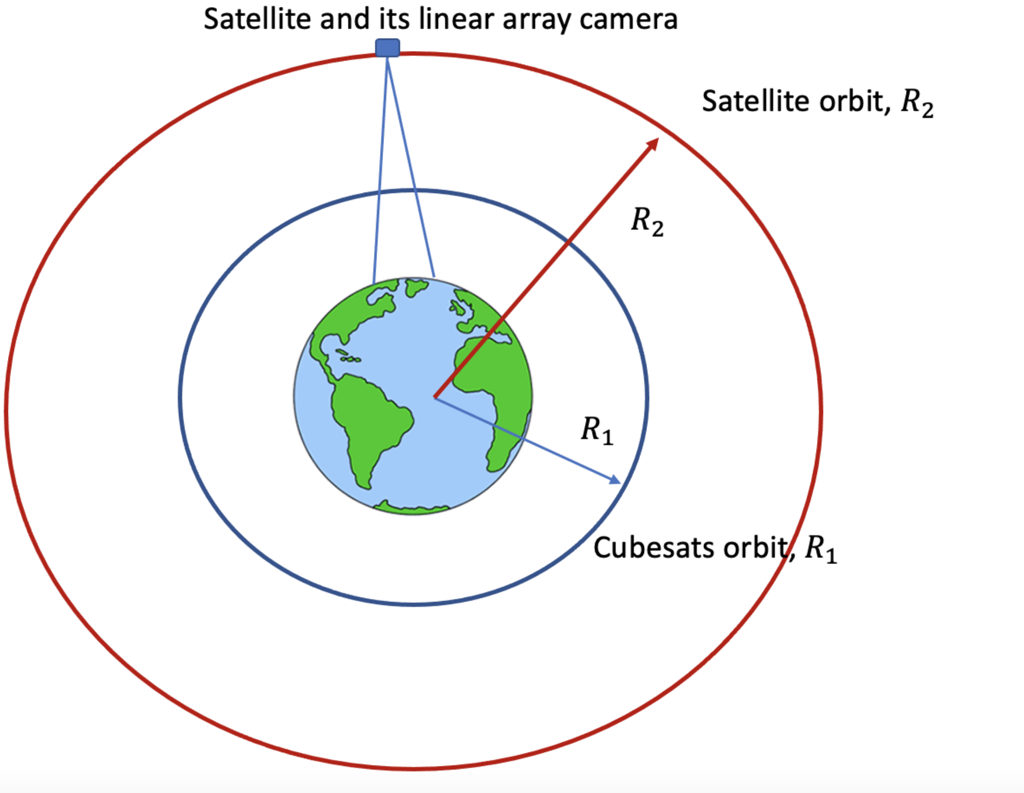


Figure 8.3: CubeSats as autonomous calibration targets on a lower orbit

The CubeSats 3D localization would be determined using ground radar. Their images on the camera would provide the corresponding 2D key points. Figure 8.4 illustrates this idea. From the training set of 3D and 2D key points, the re-calibration would be conducted.

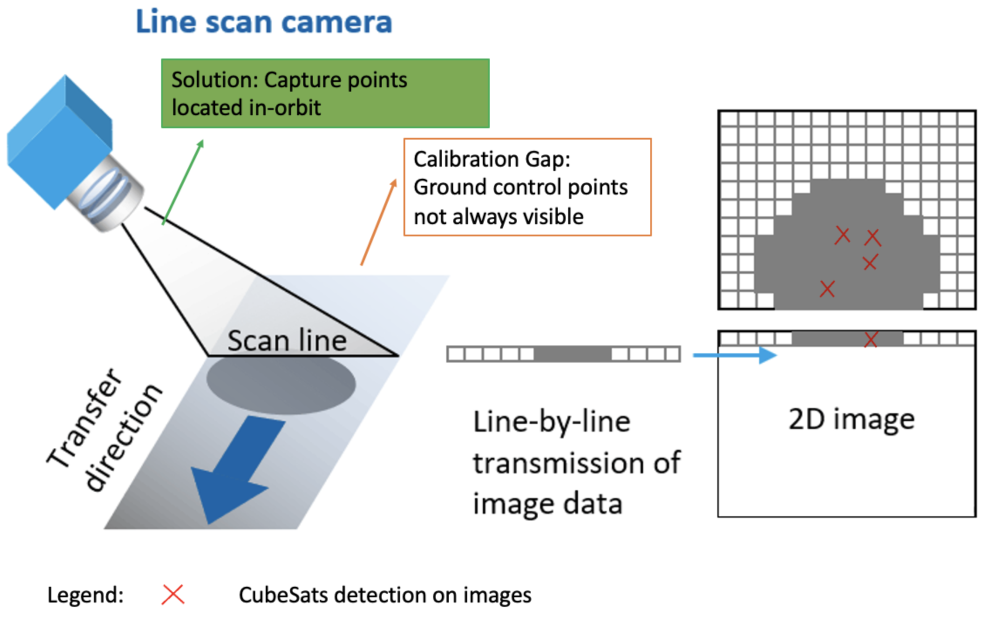

Line scan camera
Solution: Capture points located in-orbit
Calibration Gap: Ground control points not always visible
Scan line
Transfer direction
Line-by-line transmission of image data
2D image
Legend: × CubeSats detection on images

Figure 8.4: Calibration of satellite camera with orbiting control points

Once orbiting, the CubeSats can not stop their movements. Thus, the simplified solution proposed in Experiment 1, which relies on the autonomous target stopping at waypoints to simplify the correspondence between 3D and 2D, would not apply here. In that use case, the correspondence between 3D and 2D key points would rely on the synchronization of the satellite clock and the ground radar clock.

The strategy followed in Experiment 2 applies to this use case. Indeed, the CubeSats must be accurately identified in the cluttered background (Earth visible in the background). Several solutions could be tested, such as using a LED with a specific color on the calibration CubeSat or the use of a marker as presented in Experiment 2.

Future work to fully demonstrate this use case would require proposing a plan to clean the space from the CubeSats once the associated satellite whose camera is being calibrated would not be used anymore. Also, the orbit of the calibration CubeSats must be computed based on a trade-off between lower orbiting speed to avoid blurring issues during the detection and larger orbiting speed to avoid occlusion of Earth observation by the Cubesat, i.e., to make sure the satellite camera can capture the ground points below the CubeSat after the CubeSat went in the camera field-of-view. Also, Future work would need to design the CubeSat based on criteria such as visibility from the camera.

This use case relies on the use of autonomous calibration targets (CubeSats) with image and 3D localization uncertainties. Thus, the method developed in this thesis can be applied to solve the calibration of satellite cameras.

CHAPTER 9
CONCLUSION

Several video applications rely on camera calibration, a key enabler towards the measurement of metric parameters from images. For instance, monitoring environmental changes through remote cameras, such as glacier size changes, or measuring vehicle speed from security cameras, require cameras to be calibrated. Calibrating a camera is necessary to implement accurate computer vision techniques for the automated analysis of videos. This automated analysis would enable to save cost and time in a variety of fields, such as manufacturing, civil engineering, architecture and safety. The vast number of cameras installed and operated continues to increase. A key portion of these cameras are ”hard-to-reach” cameras. A calibration method to hard-to-reach cameras must overcome constraints due to the remote characteristics of these cameras. First, the method must be adapted to large fields of view since these cameras cannot be accessed easily. This requirement prevents the use of object-based calibration techniques. The method must be environment independent due to the large diversity of environments where these cameras are located, which is not feasible using self-calibration techniques since they require strict assumptions about the scene. Consequently, the existing calibration approaches are not adapted with the use of hard-to-reach cameras.

In addition to these two requirements, the method must be at least partly automated to calibrate the large number of already installed cameras, and be easily deployed for large field of views. Indeed, the method must allow the coverage of the entire field of view of the camera being calibrated. This condition is required to model correctly non-linear behaviors due to the lens for instance. Finally, the method must enable the collection of many key points in the field of view to enable the use of non-parametric methods for calibration to model cameras that would not verify the standard calibration equations. This needs lead to the overarching research question:

> Overarching Research Question: What method could be developed to collect the 3D and 2D key points required for the calibration of hard-to-reach cameras?

This thesis work designed and evaluated a solution for the calibration of hard-to-reach cameras. The proposed solution relies on the use of a drone as a moving target. The drone flies in the camera field of view to waypoints that are the 3D training points required for the calibration. The images of these waypoints through the camera are the 2D training points used for the calibration. The collected key points are the input of an optimization process that returns an estimate of the camera parameters. Once these parameters are obtained, the camera is calibrated, and it is possible to make measurements in the world frame using image information only.

The localization of the moving agent is not known accurately due to random noise in the image detection and 3D localization sensor measurements. Despite these localization uncertainties, the resulting calibration must be successful. Figure 9.1 illustrates the calibration technique designed, evaluated and validated in the thesis.

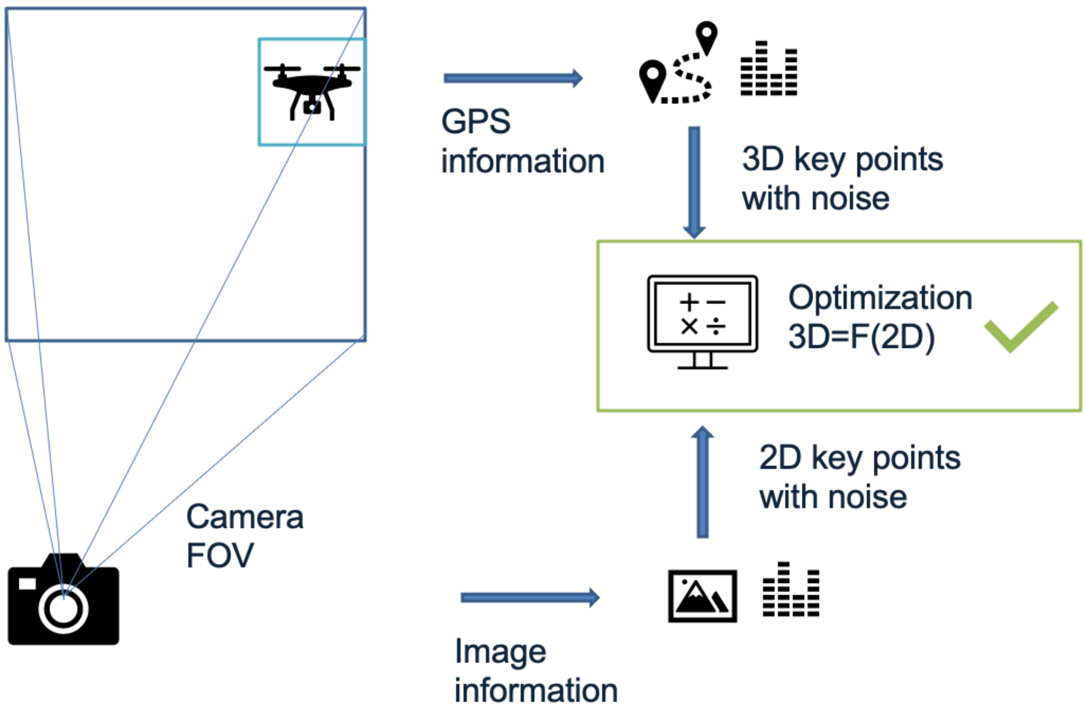


Figure 9.1: Use a drone as calibration object

This work demonstrated the success of the calibration technique, which enables the calibration of hard-to-reach cameras despite 3D and image localization uncertainties. To do so, this work first demonstrated the solution for two simulated cameras. For this step, the 3D localization is assumed accurate, and the uncertainty is introduced by image localization errors. In this step, the drone trajectory is designed so that the collected 3D and 2D key points form a complete training set with noise low enough for a good calibration. This step is divided into Experiment 1 and Experiment 2. The results of these Experiments demonstrated the success of the remote camera calibration technique under image uncertainty. In a second step, the technique is evaluated for thousands of simulated cameras, and the error introduced by the 3D localization system is accounted for. Experiment 3 identified sampling strategies that enable a successful calibration for many cameras. This step demonstrated the scalability of the technique to many cameras. The knowledge gained from Experiment 1,2 and 3 is implemented with real hardware. This real world testing completed the validation of the calibration technique and concluded with a methodology for the reproducibility of the calibration technique. Finally, a last Experiment focused on the mapping ground to image with distortion, and showed that the sampling set size can be reduced while maintaining a good calibration accuracy. This last step concluded with guidelines to reproduce the offline optimization technique with a real camera.

Figure 9.2 summarizes the research plan followed for the demonstration and validation of the UAV-enabled remote camera calibration technique under localization uncertainties.

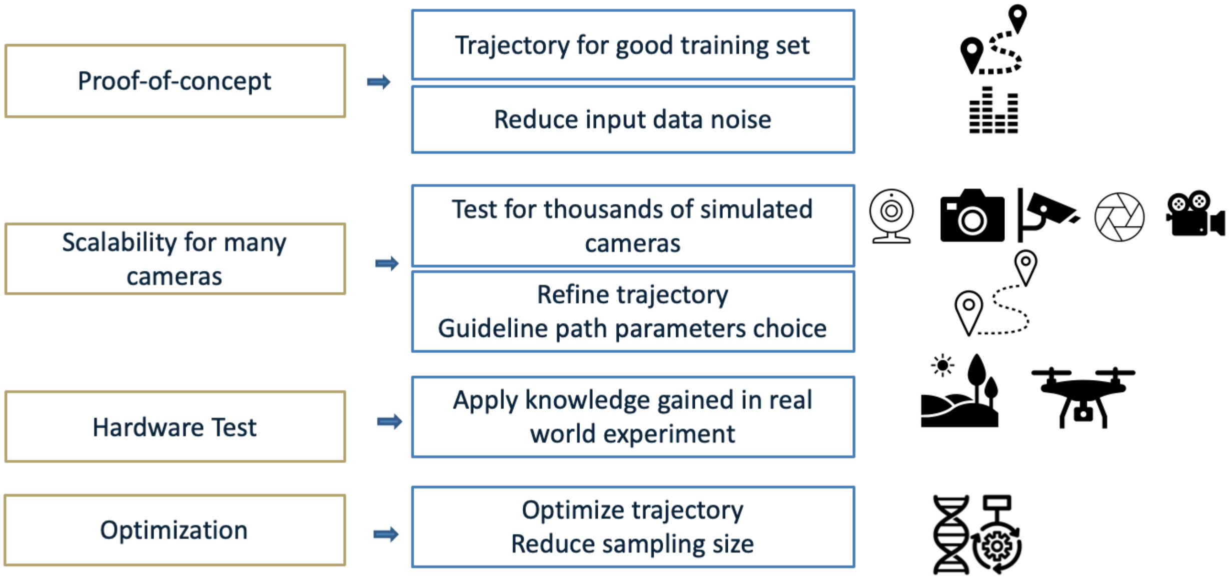

Proof-of-concept
Trajectory for good training set
Reduce input data noise
Scalability for many cameras
Test for thousands of simulated cameras
Refine trajectory
Guideline path parameters choice
Hardware Test
Apply knowledge gained in real world experiment
Optimization
Optimize trajectory
Reduce sampling size

Figure 9.2: Research Steps

The rest of this chapter summarizes the findings and contributions of this work, by looking back to each Experiment.

9.1 Experiment 1

The drone-based calibration approach must allow the collection of a complete training set that is an homogeneous representation of the application set. In addition to this requirement, the calibration resulting from the collected image points and world points must be accurate despite the localization errors for the drone on the images. These two requirements lead to Research Question 1 (RQ1), which is recalled below.

Research Question 1: What drone path would enable the collection of a good training set and thus a successful camera calibration?

In order to demonstrate the drone-based calibration concept, a set of simulations were run using ROS and Gazebo frameworks to model realistic environments, cameras and drone. In order to answer RQ1, Hypothesis 1 proposed a drone trajectory that covers the entire field of view, including regions of complex behaviors, and enables a convenient matching between 3D and 2D points allowed by the drone stopping at waypoints. Such a path leads to the collection of a training set that is a complete representation of any application sets. The use of a drone as a calibration object comes with a cost: error is introduced in the image detection due to oscillations at waypoints, use of a less accurate target compared to a chessboard-like pattern, and need to create a robust correspondence method for the 3D and 2D key points. Experiment 1 tested whether the calibration resulting from the use of the drone as a moving calibration object and the proposed trajectory leads to a good accuracy, despite uncertainty for the drone detection on the image. The RMS reprojection errors obtained through these simulations were smaller than 3.3 pixels, demonstrating that the calibration is successful despite image localization uncertainties. Indeed, it was demonstrated in Chapter 3 that the calibration is accurate enough for the remote camera applications if the reprojection error on the image is smaller than 3.6 pixels.

The flow of Experiment 1 is depicted on Figure 9.4. The red box represents the obstacles to overcome and the green boxes represent the solution tested and validated.

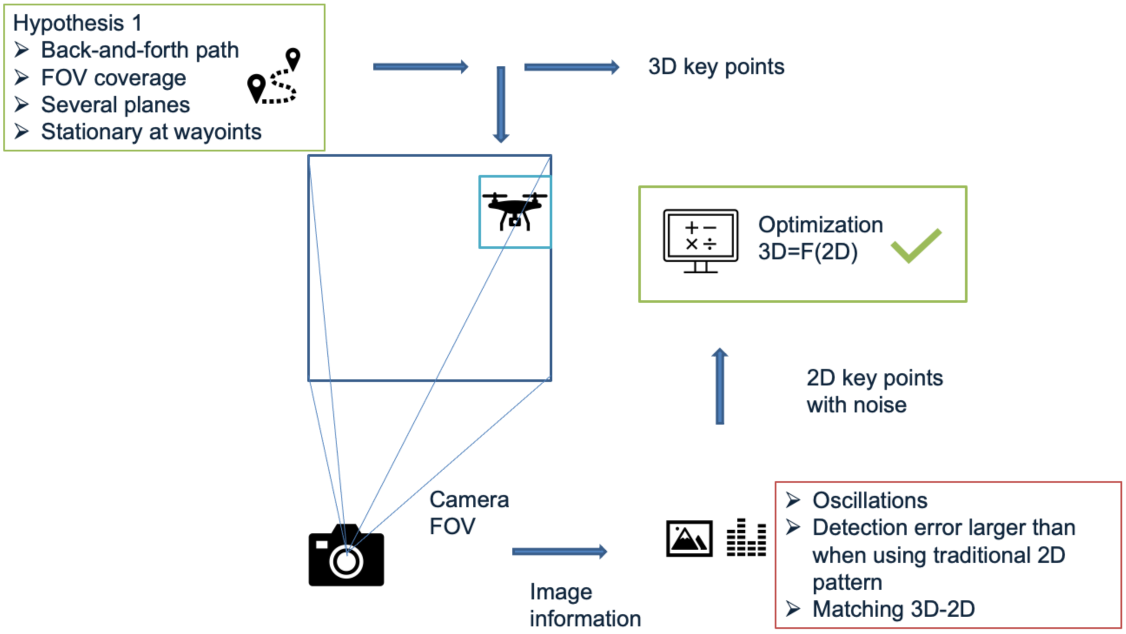


Figure 9.3: Experiment 1

The image localization error can be reduced if the method to detect the drone on the image focuses on the identification of an exact keypoint, instead of an area of the drone that takes several pixels on the image. That's why a marker is added on the simulated drone. But this marker-drone solution must overcome obstacles to enable a successful calibration. This solution was tested in Experiment 2.

9.2 Experiment 2

In order to improve the quality of the training set, Hypothesis 2 proposed to add a marker on the drone to reduce the image localization error. This strategy must allow a calibration despite complex environments where the marker is harder to detect, and complex conditions due to the use of a drone: oscillations, dimensions, obstacle avoidance distance. This need lead to research question 2.

Research question 2: Is the marker-drone solution adapted to the collection of 2D key points in diverse environments, and thus is an enabler for an improved calibration compared to the clicking method?

Experiment 2 tested whether the new design enables the collection of 2D keypoints while overcoming the following challenges:

- Environments that make the drone or marker detection difficult
- Blurring due to small drone oscillations at waypoints
- Dimensions constrained by drone dimensions
- Distance from the camera larger than 3 meters
- Automated detection robust to distortion

This experiment results demonstrated that the marker-drone solution is adapted to the collection of 2D key points in diverse environments because it overcame all obstacles listed above and lead to a reprojection error smaller than 2.84 pixels, with an average reprojection error on the tested cameras smaller by 24 percent compared to the Experiment 1 method.

Figure 9.4 summarized the steps of Experiment 2

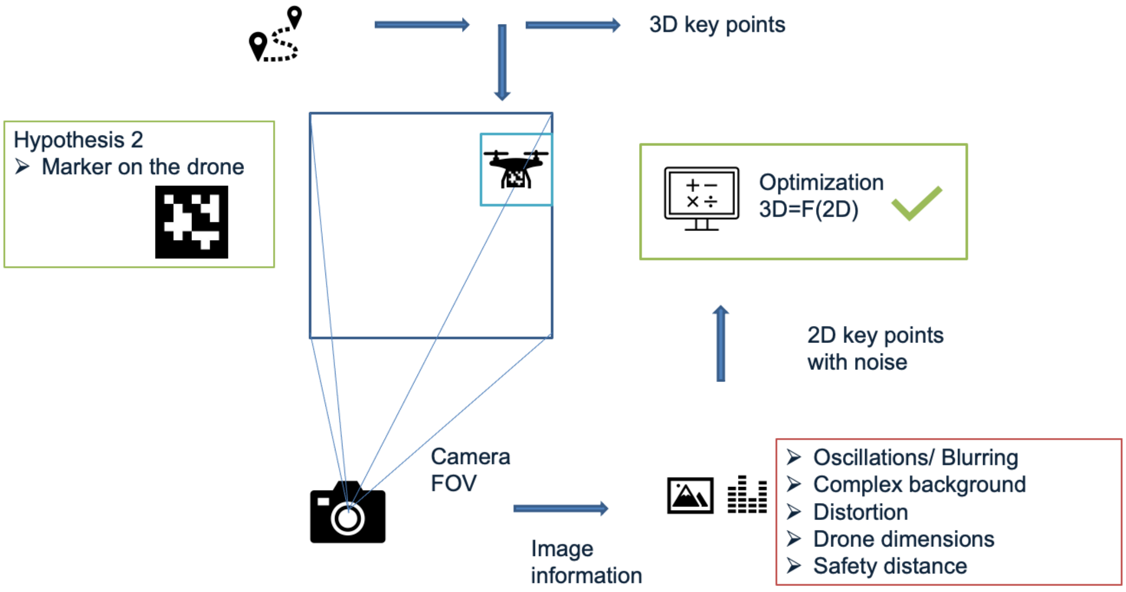


Figure 9.4: Experiment 2

Experiment 1 and Experiment 2 results demonstrated the success of the drone-based approach for the calibration of hard-to-reach cameras under image localization uncertainty. In Experiment 1 and 2, the calibration technique was tested using simulations for two cameras and assuming accurate 3D localization measurement. The approach is further demonstrated for a broad range of cameras parameters and noisy 3D localization measurements in Experiment 3.

9.3 Experiment 3

Hypothesis 3 defined a strategy to evaluate the impact of path parameters, 3D and 2D localization uncertainty, and camera parameters on the calibration accuracy for many cameras. In order to limit the impact of GPS noise on the calibration accuracy, different trajectories are investigated by applying this strategy to thousands of simulated cameras and to answer Research Question 3.

Research Question 3: How to pick the path parameters to enable a successful calibration of a broad range of cameras?

5000 simulated cameras are modeled by varying their parameters within ranges determined from the literature and using a Latin Hypercube DOE to efficiently select the parameters. The localization uncertainties are captured with a Monte Carlo sampling of 500 iterations. At each iteration, uncertainty values for the 3D and 2D coordinates are drawn from a distribution. The model for 3D uncertainty is taken from the literature, and the model for 2D uncertainty is derived in Experiment 3 and validated with statistical comparison to an available ground truth. The application of this process to 5000 simulated cameras and different sampling strategies lead to recommendations for path parameters for a successful drone-based calibration. Experiment 3 demonstrated the success of the calibration approach for many cameras, and its robustness to 3D and 2D localization uncertainties.

Figure 9.5 summarized the steps and contributions of Experiment 3. The red boxes highlight the obstacles to overcome, the green boxes are the solutions.

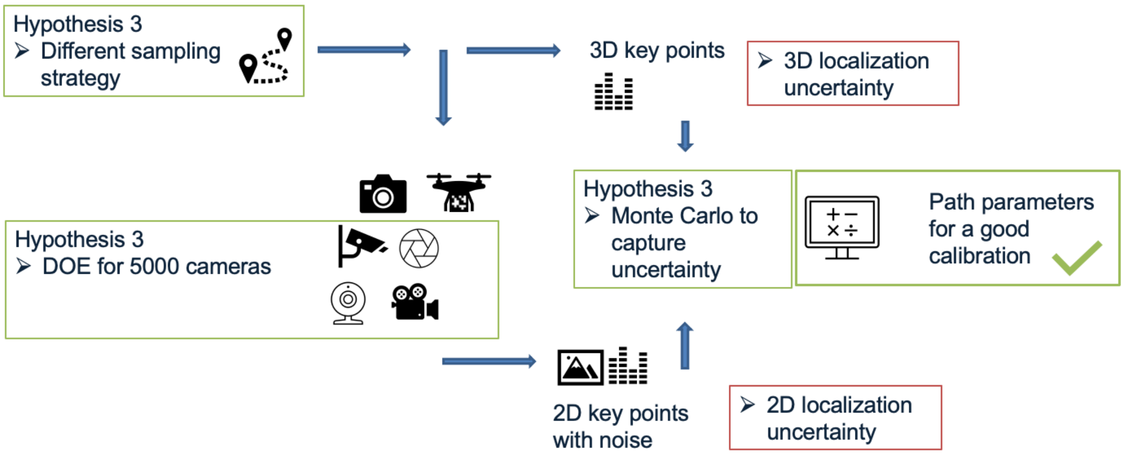

Hypothesis 3
➢ Different sampling strategy
3D key points
➢ 3D localization uncertainty
Hypothesis 3
➢ DOE for 5000 cameras
Hypothesis 3
➢ Monte Carlo to capture uncertainty
Path parameters for a good calibration
2D key points with noise
➢ 2D localization uncertainty

Figure 9.5: Experiment 3

The knowledge gained from Experiment 1, 2 and 3 was implemented with real hardware and results are presented in the next section.

9.4 Real world testing

To complete the demonstration of the drone-based remote camera calibration technique, the proposed approach was tested in a real setting, using a commercial drone and camera in an urban environment. The resulting reprojection error equals 7.7 pixels, and the drone flies to all waypoints in less than 3 minutes. The use of a commercial drone and the fast flight duration are interesting advantages of the approach when considering the large number of installed cameras. The real testing part concluded with a methodology to reproduce the drone-based calibration approach.

Figure 9.6 summarizes the steps to follow for an autonomous camera calibration with a drone as moving target. It is recommended to plan on a horizontal coverage larger than what the camera field of view might actually be. So that the points at extremity of the field of view can be covered and the sampling is homogeneous to application sets.

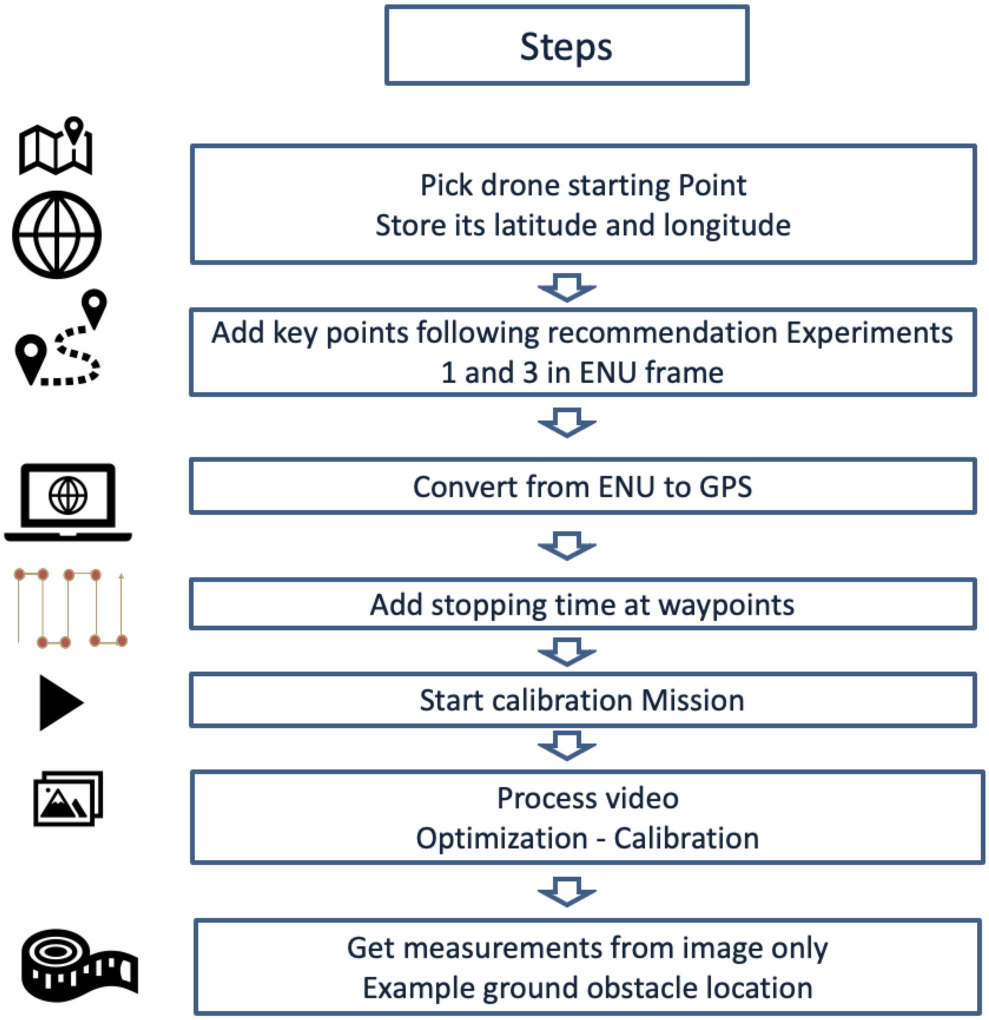

Steps
Pick drone starting Point
Store its latitude and longitude
Add key points following recommendation Experiments 1 and 3 in ENU frame
Convert from ENU to GPS
Add stopping time at waypoints
Start calibration Mission
Process video
Optimization - Calibration
Get measurements from image only
Example ground obstacle location

Figure 9.6: Process for the drone-based calibration

In order to reduce the calibration effort while maintaining a good accuracy, the feasibility and the benefits of an optimized trajectory was investigated in Experiment 4.

9.5 Experiment 4

Experiment 1,2 and 3 demonstrated the success of the drone-based calibration approach. In order to investigates whether the calibration effort can be minimized, an additional question is raised:

Research Question 4: Is there an optimum sampling set for a given camera that enables accurate calibration and reduced trajectory time?

Experiment 4 investigated the possibility of optimizing the autonomous target path while maintaining a good calibration accuracy for the mapping between image points and ground points. An offline optimization process is defined in Hypothesis 4. A genetic algorithm is implemented to explore different subsets of a baseline set made of 40 points. The influence of the localization uncertainties on the calibration is captured in the accuracy metric, which is the Monte Carlo estimate of the maximum reprojection error on the ground. Experiment 4 results showed that there is a subset of points that lead to an optimum calibration for a given camera. This subset of points depends on the camera model. Thus, when no prior parameters' knowledge is available, it is suggested that the moving target follows the baseline trajectory, which is made of straight lines and lead to an accurate calibration as demonstrated in Experiment 4.

Figure 9.7 summarizes the flow and finding of Experiment 4.

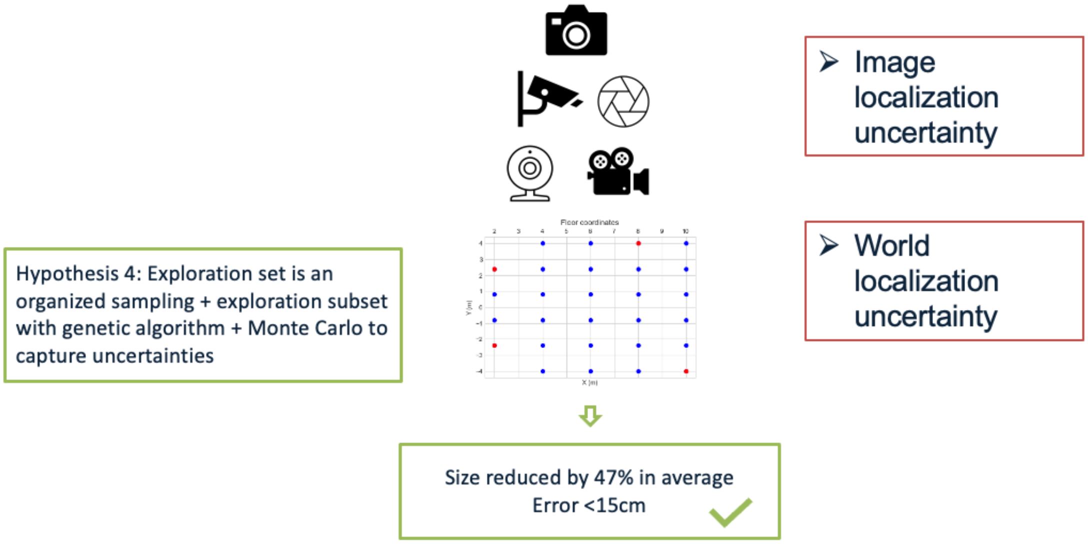


Image localization uncertainty
World localization uncertainty
Hypothesis 4: Exploration set is an organized sampling + exploration subset with genetic algorithm + Monte Carlo to capture uncertainties
Floor coordinates
X (m)
Y (m)
Size reduced by 47% in average
Error <15cm

Figure 9.7: Experiment 4

9.6 Summary of contributions

This section summarizes the key contributions of this thesis. This thesis demonstrated the success of the drone-based technique for the calibration of hard-to-reach cameras. The drone is used as a moving target that flies in the camera field of view and enables the collection of a training set used for the calibration. Its GPS coordinates provide the 3D key points and their corresponding images captured by the camera are the 2D key points. Using a drone as a moving calibration object comes with a cost: the calibration technique must be robust to 3D and 2D localization uncertainty. The results of the four experiments and the real testing demonstrated that the drone-based solution successfully enables the calibration of hard-to-reach cameras. This solution presents advantages compared to traditional methods as is summarized in the next subsection.

9.6.1 Solution for the calibration of hard-to-reach cameras

The proposed solution enables the collection of points that cover the entire field of view while avoiding a tedious and manual process, on the contrary to the object-based calibration technique. Experiment 1 demonstrated that this solution enables the collection of a complete training set and lead to a calibration error smaller than the threshold of 3.6 pixels. The proposed solution is environment independent, on the contrary to the self-calibration method that depends on the scene geometry. This property has been demonstrated with Experiment 2 and the real-world testing that showed that the technique overcome obstacles caused by challenging backgrounds and environments. The real world testing demonstrated that the method is automated which is an advantage when considering the large number of installed cameras. Consequently, this solution overcomes the limits of traditional calibration techniques, and is adapted to the calibration of hard-to-reach cameras.

Experiment 2,3 and 4 enabled to improve the drone-based calibration strategy which was first demonstrated in Experiment 1. Their contributions is summarized in the next subsection.

9.6.2 Enhancement of the solution

Experiment 2 demonstrated that adding a marker on the drone that satisfies drone dimension constraints, and distance to camera constraints enables to improve the calibration accuracy. This experiment demonstrated that this solution is robust against complex conditions, such as challenging backgrounds, oscillations of the drone, and distortion.

Figure 9.8 illustrates the final drone design recommended for an improved calibration accuracy compared to Experiment 1.

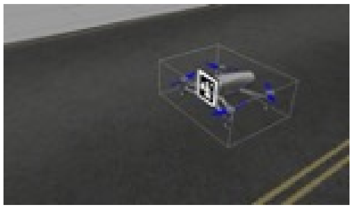


Figure 9.8: Drone-Marker system

Experiment 3 demonstrated the scalability of the technique to many different camera parameters. It resulted in guidelines for the choice of path parameters to enable a successful calibration robust to 3D and 2D localization uncertainty. An example of such guidelines is recalled with Figure 9.9 that represents the preferred sampling strategy along Y (blue for Y sampled every meter and red for Y sampled every 50 centimeters) for various intrinsic parameters.

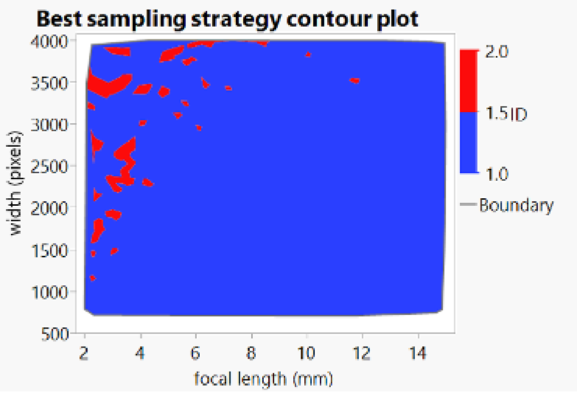


Figure 9.9: Preferred sampling density along Y for different camera parameters

A real testing was implemented to fully demonstrate the drone-based calibration methodology. This experiment was run in Couch Park at Georgia Tech with a DJI drone as showed on Figure 9.10.

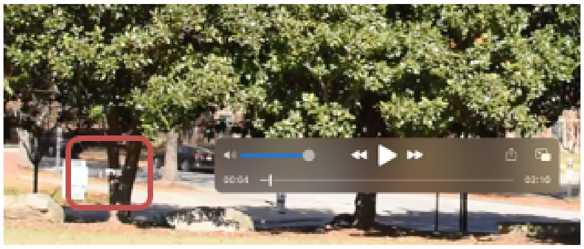


Figure 9.10: Real world testing

Finally, Experiment 4 focused on the relation image to ground for distorted cameras, and demonstrated that the sampling size can be reduced for a faster calibration, while maintaining a good calibration accuracy. Figure 9.11 recalled the reduction of the sampling size obtained by running Experiment 4 for 14 cameras with and without distortion.

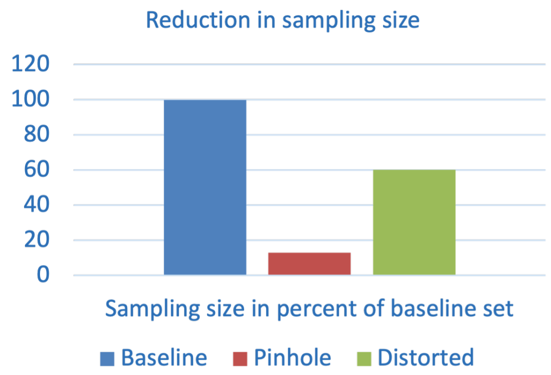


Figure 9.11: Reduced training set size

9.7 Proposition of Future Work

In the work presented in this thesis, the correspondence between 3D and 2D key points relies on the drone stopping at waypoints. The strategy for the correspondence between key points could be further investigated with a method where the drone would not need to stop at waypoints. In a future work, the GPS and video time could be synchronized. Any points along the trajectory would be a candidate training point, enabling the collection of a very large training set, and possibly a dense set of points. It would enable the application of machine learning techniques that require a large training set for the model fitting part, such as an artificial neural network. In turn, it would allow modeling cameras that do not verify the traditional equations from 3D to 2D, due to external factors such as vandalism or blurring due to weather damages.

Appendices

APPENDIX A

A CALIBRATION SOLUTION SCALABLE TO ZOOM-LENS CAMERAS

The thesis does not focus on non-static cameras. These cameras have the ability to move or to zoom. As an example, Pan-tilt-zoom (PTZ) cameras are capable of tracking activities over a large scene and are able to capture high resolution images around objects [166].

When the zoom varies, some camera parameters are fixed. For instance, extrinsic parameters and the pixel width s_x and s_y do not change when the zoom is activated. However several intrinsic parameters are impacted by the zoom change, such as the focal length, the principal point location and the distortion coefficients [167] [168].

If a camera has a zoom-lens it is then necessary to determine these variable parameters for different lens settings. Most techniques for zoom-lens camera calibration determine the camera's parameters for different zoom settings and populate a table with the corresponding parameters. Then an interpolation for other zoom values is implemented using the data gathered in the table [169]. In [166], the intrinsic parameters are first determine at the camera's lowest zoom setting. Then the variable parameters are determined at discrete steps by monotonically increasing the zoom. Once parameters value are obtained at different zoom levels, an interpolation is done to obtain the variable parameters at zoom values for which the camera has not been calibrated. In [170] a spline interpolation is implemented, in [171] a least square approach is used to model the variation of internal parameters as a function of zoom. The thesis focuses on obtaining an accurate calibration, at a fixed zoom level, for cameras that are already installed and located in complex environments. A solution to expand the thesis approach to zoom-lens cameras is to implement calibration at several zoom levels using the technique developed in this thesis. And then an interpolation can be implemented to obtain a calibration for every zoom levels.

APPENDIX B

CALIBRATION FUNCTION: THE STEPS BEHIND THE ”BLACK BOX” FUNCTION

The key steps of the calibration process was provided in Chapter 2. However, it was not explained how the camera parameters are obtained from the result of the least Square regression. The corresponding steps to retrieve the rotation, translation, focal length, image center parameters from the optimization coefficients were presented as a black box function. This appendix details a method derived in [172] and implemented in this thesis. One might want to know how the parameters are computed from the estimated product of intrinsic matrix, rotation and translation, since these parameters are required for the use cases presented in Chapter 7.

The result of the optimization for calibration is made of coefficients that are function of the camera parameters. The camera parameters are then estimated from these coefficients. The steps provided below enable to compute the camera parameters from the least square regression result.

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = M * (\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} - \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix})$$

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} m11 & m12 & m13 & -m11t1 - m12t2 - m13t3 \\ m21 & m22 & m23 & -m21t1 - m22t2 - m23t3 \\ m31 & m32 & m33 & -m31t1 - m32t2 - m33t3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

$$A = \begin{bmatrix} f_x & 0 & C_x & 0 \\ 0 & f_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} m11 & m12 & m13 & -m11t1 - m12t2 - m13t3 \\ m21 & m22 & m23 & -m21t1 - m22t2 - m23t3 \\ m31 & m32 & m33 & -m31t1 - m32t2 - m33t3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

$A = \lambda * V^{-1} * B^{-1} * F * M * T$ where $V^{-1}B^{-1}F$ contain the intrinsic parameters and $M * T$ the extrinsic parameters

The steps to compute this decomposition into matrices V^{-1}, B^{-1},F,M and T and to compute the camera parameters are described below.

- Compute $abs(\lambda) = \sqrt{{a_{31}}^2 + {a_{32}}^2 + {a_{33}}^2}$.
- Take A' which is the top-left 3*3 submatrix of A and compute A''=$\frac{A'}{abs(\lambda)}$ = A'
- Compute the RQ decomposition of A''
- Modify R to ensure that all its diagonal components are positive. Multiply R by the matrix $J = \begin{bmatrix} \substack{+\\-}1 & 0 & 0 \\ 0 & \substack{+\\-}1 & 0 \\ 0 & 0 & \substack{+\\-}1 \end{bmatrix}$ to satisfy this condition.
- Compute $R' = J * R$
- Build $S = \begin{bmatrix} cosa & sina & 0 \\ -sina & cosa & 0 \\ 0 & 0 & 1 \end{bmatrix}$ where $tan(a) = \frac{-r\prime_{12}}{r\prime_{11} + r\prime_{22}}$
- Compute $G = RJS = R'$. Some intrinsic parameters can be obtained using G:
- $F = 2\frac{g_{11}g_{22} - {g_{12}}^2}{g_{11} + g_{22}}$
- $Cx = g_{13}$
- $Cy = g_{23}$

- Compute $M' = G^{-1}A'$ and $\lambda = det(M')$

- MT=$det(M')$ M' and as seen from the equation at the beginning of the document, MT=$\begin{bmatrix} m11 & m12 & m13 & -m11t1 - m12t2 - m13t3 \\ m21 & m22 & m23 & -m21t1 - m22t2 - m23t3 \\ m31 & m32 & m33 & -m31t1 - m32t2 - m33t3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$

- From MT, get the rotation matrix $R = [R_1, R_2, R_3]$ and the translation vector $T = [t_1, t_2, t_3]^T$:

- $R_i = M[:, i]$

- T=[t1,t2,t3]T is obtained by solving

  $$\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} * T = \begin{bmatrix} -m_{11}t_1 - m_{12}t_2 - m_{13}t_3 \\ -m_{21}t_1 - m_{22}t_2 - m_{23}t_3 \\ -m_{31}t_1 - m_{32}t_2 - m_{33}t_3 \end{bmatrix}$$

  Where $\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$ and $\begin{bmatrix} -m_{11}t_1 - m_{12}t_2 - m_{13}t_3 \\ -m_{21}t_1 - m_{22}t_2 - m_{23}t_3 \\ -m_{31}t_1 - m_{32}t_2 - m_{33}t_3 \end{bmatrix}$ are obtained when computing MT in the previous step.

  Once the focal length, the camera center coordinates, the rotations and translations parameters are estimated, they are inserted into the equation 3D to 2D that include the distortion coefficients. An iterative optimization process is run to refine the parameters computed with the method above, as well as estimate the distortion coefficients.

  Packages in Matlab or python offer an implementation of these steps.

APPENDIX C

TOOL FOR THE OBSTACLE LOCALIZATION USE CASE

Appendix B presents the interface built to enable the obstacle location application presented in chapter 6.

Jupiter notebook for the application of the drone-based remote camera calibration approach The Use case implemented in this notebook is called "Get Obstacle Location"

0.1 Get obstacle location

Determine an obstacle location in the world frame from its 2D location on the image taken from the remote camera who has been calibrated with the drone-based calibration approach. Thus, whose intrinsic and extrinsic parameters are known. The object is on the ground, or one of its dimensions is known (example height).

0.2 Packages to load

[2]:

```
import math
import numpy as np
from scipy import optimize
import cv2
import pandas as pd
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
```

0.3 Load the key points obtained from flying the drone in front of the camera

[3]:

```
L=pd.read_csv('P4_Calib_2.csv')
L=L.values.tolist()
```

0.4 Camera calibration

Load a frame of the video to compute the resolution

[4]:

```
im=mpimg.imread('IMG.jpg')
size=im.shape[:2]
w0=size[1]
h0=size[0]
Cx0=int(w0/2)
Cy0=int(h0/2)
f0=h0

#Calibration
OP=np.array([[l[1:4] for l in L]]).astype(np.float32)
IP=np.array([[l[4:6] for l in L]]).astype(np.float32)
# Initial guess
cameraMatrix=np.zeros((3,3),dtype=np.float32)
cameraMatrix[0,0] = f0;
cameraMatrix[1,1] = f0;
cameraMatrix[2,2] = 1;
cameraMatrix[0,2] = Cx0;
cameraMatrix[1,2] = Cx0;
```

```
rms, camera_matrix, dist_coefs, _rvecs, _tvecs = \
        cv2.calibrateCamera(OP, IP, (w0, h0), cameraMatrix, None, \
        flags=cv2.CALIB_USE_INTRINSIC_GUESS)
print("RMS reprojection error: %f" % rms)
rvec=np.array(_rvecs)
tvec=np.array(_tvecs)
dist_coefs=np.array(dist_coefs)
rotation_mat, _=cv2.Rodrigues(rvec)
rvec2=cv2.Rodrigues(rvec)[0]
```

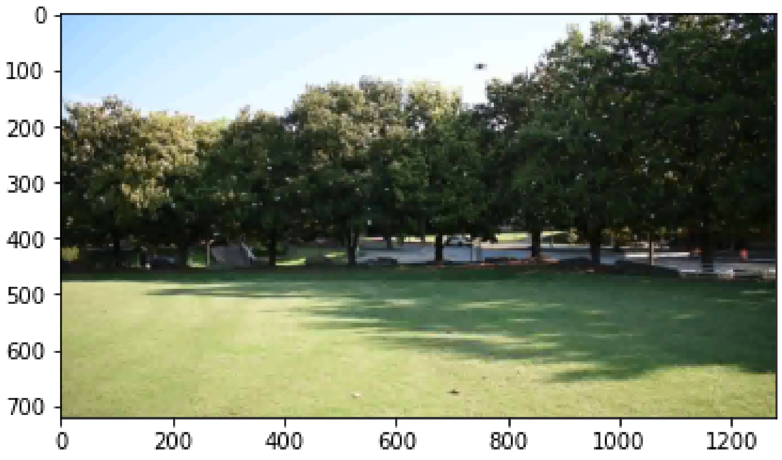
0.5 Undistort image

```
[5]: im=mpimg.imread('IMG.jpg')

     im2= cv2.undistort(im, camera_matrix, dist_coefs)

     imgplot = plt.imshow(im)
     plt.show()

     cv2.imwrite('UndistorerImg.jpg', im2)
```



```
[5]: True
```

0.6 From the undistorted image, detect the point to measure

[ ]:

```
im=mpimg.imread('UndistorerImg.jpg')
r = cv2.selectROI(im)

u=int(r[0]+r[2])
v=int(r[1]+r[3])
```

0.7 Definition of the function that relates world points to image points on the undistorted image

The function Homography_Matrix returns the matrix H=K[R T] where K is the intrinsic matrix, R the rotation and T the translation vector.

The function world_to_image take as input a point M (vector OM) of coordinates [X,Y,Z], an homography matrix H and returns the projected point on the image [u,v] in pixels

[ ]:

```
def Homography_Matrix(K, R, T):

    M1=np.zeros((3,4))
    M1[0][0]=1
    M1[1][1]=1
    M1[2][2]=1

    R1=np.zeros((4,4))
    R1[0][0]=R[0][0]
    R1[0][1]=R[0][1]
    R1[0][2]=R[0][2]
    R1[1][0]=R[1][0]
    R1[1][1]=R[1][1]
    R1[1][2]=R[1][2]
    R1[2][0]=R[2][0]
    R1[2][1]=R[2][1]
    R1[2][2]=R[2][2]
    R1[3][3]=1

    T1=np.identity(4)
    T1[0][3]=T[0]
    T1[1][3]=T[1]
    T1[2][3]=-T[2]

    P1=np.dot(R1, T1)
    P2=np.dot(M1, P1)

    P3=np.dot(K, P2)

    return(P3)
```

```
"""
Compute pixel coordinates of point M seen by camera with homography H
"""

def world_to_image(OM, H):
    X=OM[0]
    Y=OM[1]
    Z=OM[2]

    l=H[2,0]*X+H[2,1]*Y+H[2,2]*Z+H[2,3]
    u = (H[0, 0] * X + H[0, 1] * Y + H[0, 2] * Z + H[0, 3])/l
    v=(H[1, 0] * X + H[1, 1] * Y + H[1, 2] * Z + H[1, 3])/l

    u=int(u)
    v=int(v)
    return([u,v])
```

0.8 Compute line in 3D for a projected point [u,v]

Compute the line equation where a point M is in the world frame when it projects on the image at m=[u,v]

The line equation is OM=OC+t*d where O is the world frame center, OM the vector from the world frame to the point M on the line, C is the camera center, d is the direction unit vector for the line that goes through C and M and t is the scalar (distance C to M along d)

The input of the function are the homography (mapping world to image) for the camera, the pixel coordinates u and v and the camera center location C=[X,Y,Z]

The output of the function is the line equation, i.e. the coordinates of the vector OC and the direction unit vector d

Note: the ray where the point [u,v] belongs in 3D intersects with the ground floor and also goes through the camera center. So its equation can be obtained by taking a point whose Z=0, computing the reducde homography such as [u,v,1]=[X,Y,1] (Z=0), invert it (possible since assumed Z=0 so square matrix), get [X,Y] from this inverted matrix and [u,v] and compute the unit vector that gives the direction between [X,Y] and Camera center C

[ ]:

```
def Ray(H1, u,v, C):
    #the ray can be obtained by computing the unit vector for the direction "camera - point on floor" - so Z is picked equal to 0 - new H with Z=0 is computed and inverted to get X and Y
    H2=np.zeros((3,3))
    H2[0,0]=H1[0,0]
    H2[0, 1] = H1[0, 1]
    H2[1, 0] = H1[1, 0]
    H2[1, 1] = H1[1, 1]
    H2[2, 0] = H1[2, 0]
    H2[2, 1] = H1[2, 1]
    H2[0, 2] = H1[0, 3]
    H2[1, 2] = H1[1, 3]
    H2[2, 2] = H1[2, 3]

    H3=np.linalg.inv(H2)

    #compute coordinates points on floor
    la=H3[2,0]*u+H3[2,1]*v+H3[2,2]
    X=(H3[0,0]*u+H3[0,1]*v+H3[0,2])/la
    Y=(H3[1,0]*u+H3[1,1]*v+H3[1,2])/la

    print('coord groundZ')

    print(X)
    print(Y)

    #the ray is the direction of vector joining camera center C and the point on␣
  ↪the floor F=[X,Y,0]
    CF_x=X-C[0]
    CF_y=Y-C[1]
    CF_z=-C[2]

    CF=[CF_x, CF_y, CF_z]
    print('Vector camera to turtle')
    print(CF)
    CF2=np.array(CF)
    normCF2=np.linalg.norm(CF2)

    d=[CF_x/normCF2, CF_y/normCF2, CF_z/normCF2] #direction vector for the line␣
  ↪"Camera - point that projects on (u,v)" - unit vector

    line=[C,d] #all point M on the line satisfy OM=C+t*d where t is a scalar, C␣
  ↪camera center, d direction and O world frame center
    return(line)
```

0.9 Compute the object location [X,Y,Z]

Input are the line equation for the ray "Camera center - object" - called L1, and the object height

The output is a vector: OM

```
[ ]: def world_location(L1, h):
         C1=L1[0] #camera center
         d1=L1[1] #unit vector direction ray

         h_from_cam=h-C1[2]
```

```
    #vector camera to object M
    CM=[t1*d1[0], t1*d1[1], t1*d1[2]]

    t1=h_from_cam/(d1[2])
    OM_x=C1[0]+t1*d1[0]
    OM_y=C1[1]+t1*d1[1]
    OM_z=C1[2]+t1*d1[2]

    return([OM_x, OM_y, OM_z])
```

0.10 Fire hydrant example

[ ]:

```
#camera location from calibration
Cx=tvec2[0]
Cy=tvec2[1]
Cz=tvec2[2]

C=[Cx, Cy, Cz]

#Homography
H=Homography_Matrix(camera_matrix, rotation_mat, tvec2)

#Compute ground coordinates
out=Ray(H,u,v,C)
print(out)
```

REFERENCES

[1] G. D. Scholes, G. R. Fleming, A. Olaya-Castro, and R. Van Grondelle, “Lessons from nature about solar light harvesting,” *Nature chemistry*, vol. 3, no. 10, p. 763, 2011.

[2] E. R. Thieler, “Photogrammetry,” in *Encyclopedia of Coastal Science*, C. Finkl Charles Makowski, Ed. Springer International Publishing, 2019.

[3] I. M. James S. Aber, “Small-format aerial photography and uas imagery,” 2019.

[4] “Editorial: Computer vision and photogrammetry: Interaction or introspection?” *The photogrammetric Record*, S. I. Granshaw and C. S. Fraser, Eds., 2015.

[5] J. L. M. Richard I. Hartley, “Relationship between photogrammmetry and computer vision,” *: Optical Engineering and Photonics in Aerospace Sensing*, 1993.

[6] L.-M. E. G., “Introduction to computer vision, university of massachussetts, amherts, department of computer science,”

[7] “Relationship between photogrammmetry and computer vision,” *Optical Engineering and Photonics in Aerospace Sensing*, R. I. Hartley and J. L. Mundy, Eds., 1993.

[8] “Chapter 34 - a topological approach for detection of chessboard patterns for camera calibration,” L. Deligiannidis and H. R. Arabnia, Eds., pp. 517–531, 2015.

[9] M. P. Gustavo Laureano, “Chapter 34,a topological approach for detection of chessboard patterns for camera calibration,” 2015.

[10] Malab, *What is camera calibration*, https://www.mathworks.com/help/vision/ug/camera-calibration.html, Technical.

[11] S. S. Kenji Hata, *Cs231a course notes 1: Camera models*, https://web.stanford.edu/class/cs231a/course_notes/01-camera-models.pdf, lecture.

[12] S. O. Heikkila J., “A four-step camera calibration procedure with implicit image correction,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1106–1112.

[13] T.-C. P. William B. Thompson, “Detecting moving objects,” *International Journal of Computer Vision volume*, 1990.

[14] M. S. V. P. Gustavo Teodoro Laureano, “Chapter 34 - a topological approach for detection of chessboard patterns for camera calibration,” 2015.

[15] P. M. Zhongwei Tang Rafael Grompone von Gioi, “High-precision camera distortion measurements with a ”calibration harp”,” *Journal of the Optical Society of America*, 2012.

[16] H. Maas and U. Hampel, “Photogrammetric techniques in civil engineering material testing and structure monitoring,” *Photogrammetric Engineering and Remote Sensing*, vol. 72, pp. 39–45, 2006.

[17] “Automatic crack monitoring using photogrammetry and image processing,” *Measurement*, vol. 46, no. 1, pp. 433–441, 2013.

[18] P. technologies, *Powerful and flexible photogrammetry software*, https://www.photomodeler.com/, online.

[19] L. N. de Vasconcelos Luiz, “Store separation: Photogrammetric solution for the static ejection test,” *International Journal of Aerospace Engineering*, 2019.

[20] B. Stark, B. Smith, and Y. Chen, “Electronic imaging conference,” in *A new dimension in geometric camera calibration*, 2020.

[21] G. J. Charles Toth, “Remote sensing platforms and sensors: A survey,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 22–36, 2016, Theme issue ’State-of-the-art in photogrammetry, remote sensing and spatial information science’.

[22] C. D. Dragomir Anguelov, “Google street view: Capturing the world at street level,” *Computer*, 2010.

[23] “Construction of a 3d map of indoor environment,” *Procedia Computer Science*, vol. 125, pp. 124–131, 2018.

[24] R. Szeliski, “Computer vision: Algorithms and applications,” 2020.

[25] O. Yilmaz and F. Karakus, “Stereo and kinect fusion for continuous 3d reconstruction and visual odometry,” 2013.

[26] E. L. Myrick, “Geographical surveying using cameras in combination with flight computers to obtain images with overlaid geographical coordinates,” 1989.

[27] S. Panu, “Geometric camera calibration in support of gis,” X. H. Shekhar Shashi, Ed., 2017.

[28] “Uav photogrammetry for topographic monitoring of coastal areas,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 104, pp. 101–111, 2015.

[29] B. T. G. Grenzdörffer Andreas Engelb, "The photogrammetric potential of low-cost uavs in forestry and agriculture," 2008.

[30] R. M. Kanistras Konstantinos Martins Goncalo, "A survey of unmanned aerial vehicles (uavs) for traffic monitoring," pp. 221–234, 2013.

[31] "Chapter 4 - remote sensing techniques," S. Gandhi and B. Sarkar, Eds., pp. 81–95, 2016.

[32] N. Jacobs, W. Burgin, and N. Fridrich, "The global network of outdoor webcams: Properties and applications," *International Conference on Advances in Geographic Information Systems*, 2009.

[33] A. Bicknell, B. Godley, E. Sheehan, S. Votier, and M. Witt, "Camera technology for monitoring marine biodiversity and human impact," *Frontiers in Ecology and the Environment*, vol. 14, pp. 424–432, 2016.

[34] E. S. Bradley, M. P. Toomey, C. J. Still, and D. A. Roberts, "Multi-scale sensor fusion with an online application: Integrating goes, modis, and webcam imagery for environmental monitoring," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 3, no. 4, pp. 497–506, 2010.

[35] D. Giordan, P. Allasia, N. Dematteis, F. Dell'Anese, M. Vagliasindi, and E. Motta, "A low-cost optical remote sensing application for glacier deformation monitoring in an alpine environment," *Sensors*, vol. 16, no. 10, 2016.

[36] G. D. Scholes, G. R. Fleming, A. Olaya-Castro, and R. Van Grondelle, "General design of space optical remote sensing camera based on high-precision surveying and mapping requirements," *Nature chemistry*, vol. 3, no. 10, p. 763, 2011, doi:10.1038/nchem.1145.

[37] K. Jacobsen, "Calibration of optical satellite sensors," 2006.

[38] S. L. Jin Li Yuan Zhang, "Self-calibration method based on surface micromaching of light transceiver focal plane for optical camera," *Remote Sensing*, 2016, doi:10.3390/rs8110893.

[39] R. W. Wei LiuFangming QianYuzhe Miao, "A method for self-calibration in satellite with high precision of space linear array camera," 2016, doi:10.5194/isprsarchives-XLI-B1-361-2016.

[40] Z. L. Jin Li, "Efficient camera self-calibration method for remote sensing photogrammetry," *Opt Express*, 2018, doi:10.1364/OE.26.014213.

[41] Motorola, “Eyes on the street: How wireless video solutions are transforming public safety,”

[42] N. Jenkins, “North american security camera installed base to reach 62 million in 2016,”

[43] C. Camera, *How many cctv cameras in london*, https:///www.caughtoncamera.net/news/how-many-cctv-cameras-in-london, News article.

[44] T. Williams, “To combat crime, chicago builds an arsenal of 30000 cameras,” 2018.

[45] A. Triboire, “Decharges sauvages: Les pollueurs traqués,” 2019.

[46] S. M. Vakili E. Shoaran M., “Single–camera vehicle speed measurement using the geometry of the imaging system,” *Multimedia Tools and Applications*, 2020.

[47] *How to install security cameras*, https://www.swann.com/blog/how-to-install-security-cameras/, website, 2018.

[48] *Security camera installation*, https://www.homedepot.com/c/ah/how_to_install_security_cameras/9ba683603be9fa5395fab903516a307, website.

[49] J. A. Akira Kinoshitaab Atsuhiro Takasub, “Real-time traffic incident detection using a probabilistic topic model,” *Information Systems*, vol. 54, pp. 169–188, 2015.

[50] C. Maduro, K. Batista, P. Peixoto, and J. Batista, “Estimation of vehicle velocity and traffic intensity using rectified images,” pp. 777–780, 2008.

[51] a. L. Jinhui Lan, “Vehicle speed measurement based on gray constraint optical flow algorithm,” *Optik*, vol. 125,

[52] J. S. Marketa Dubska, “Automatic camera calibration for traffic understanding,” *AUTOMATIC TRAFFIC UNDERSTANDING*, 2014.

[53] A. Nurhadiyatna, B. Hardjono, A. Wibisono, W. Jatmiko, and P. Mursanto, “Its information source: Vehicle speed measurement using camera as sensor,” 2012.

[54] N. L.-H. Do Viet-Hoa, “A simple camera calibration method for vehicle velocity estimation,” 2015.

[55] E. C. M J Smith, “The effect of temperature variation on single-lens-reflex digital camera calibration parameters,” *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2010.

[56] T. H. Jamieson, “Thermal effects in optical systems,” *Optical Engineering*, 1981.

[57] M. L. Jiahang Tian Jieqiong Lin, “Ambient temperature effect on the imaging quality of an aspherical airborne camera: Theoretical and experimental analysis,” *Applied Optics*, 2021.

[58] D. Mulawa and S. Louis, “On-orbit geometric calibration of the orbview-3 high resolution imaging satellite,” 2004.

[59] C. Jordi, https://jordicenzano.name/front-test/2d-3d-paradigm-overview-2011/camera-model/.

[60] Y. Morvan, *Projective geometry*, http://epixea.com/research/multi-view-coding-thesisch2.html, Research blog.

[61] R. Collins, *Cse486 lecture, penn state university*.

[62] C. Y. Cristofalo E Xu J, *Final project - cs 585 image and video computing- 3d scene reconstruction from video cristofalo*.

[63] C. d. University of Toronto, *Camera models and parameters*.

[64] C. Claire, *Mpsi optic lecture, pasteur preparatory classes*.

[65] U. o. W. Vision Faculty, *Cse, projection and image formation, lecture 5*.

[66] J. S. L. Abdulrahman S. Alturki, *Accurate estimation of principal point using three mutually orthogonal horizon lines*, 2016.

[67] L. J. Drap P, “An exact formula for calculating inverse radial lens distortions,” *Sensors*, 2016.

[68] P. G. Bebis, *Camera parameters, cs485/685 computer vision, university of nevada*.

[69] S. Liu, H. Fu, and Y. Wang, “Camera calibration based on divided region ls-svm,” in *2008 IEEE International Conference on Mechatronics and Automation*, 2008, pp. 488–492.

[70] *Camera model: Intrinsic parameters*, https://lhoangan.github.io/camera-params/.

[71] Z. S. Hongli Peng, *Foot modeling based on machine vision and social manufacturing research*, 2018.

[72] R. Tsai, “A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses,” *IEEE Journal on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[73] N. S. Swarninathan R, "Non-metric calibration of wide-angle lenses and polycameras," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 2, 1999, 413–419 Vol. 2.

[74] "Cse, projection and image formation, lecture 5," *CSE, vision faculty, University of Washington*,

[75] NIST/SEMATECH, *Engineering statistics handbook, process modeling*, http://www.itl.nist.gov/div898/handbook/.

[76] G. T. Dr German, "E 6310: Optimization for the design of engineered systems - lecture surrogate modeling,"

[77] "Smart sampling algorithm for surrogate model development," *Computers and Chemical Engineering*, vol. 96, pp. 103–114, 2017.

[78] C. Y.C., "Stat 535 lecture: Statistical machine learning, fall 2019, euniversity of washington,"

[79] M. Mendonça, I. N. D. Silva, and J. E. C. Castanho, "Camera calibration using neural networks," in *WSCG*, 2002.

[80] L. N. Smith and M. L. Smith, "Automatic machine vision calibration using statistical and neural network methods," *Image Vis. Comput.*, vol. 23, pp. 887–899, 2005.

[81] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar, "Distributed localization of networked cameras," in *2006 5th International Conference on Information Processing in Sensor Networks*, 2006, pp. 34–42.

[82] Y. Furukawa and J. Ponce, "Accurate camera calibration from multi-view stereo and bundle adjustment," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[83] s. o. I. C. G. Dr Pradalier, "Cs 4495 computer vision,"

[84] G. T. Dr Kennedy, "E 6310: Optimization for the design of engineered systems - lecture steepest descent, conjugate gradient andquasi-newton methods,"

[85] Q. Ji and Y. Zhang, "Camera calibration with genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 31, no. 2, pp. 120–130, 2001.

[86] J. Ma and B. Li, “A method of camera calibration by iterative optimization algorithm,” in *Proceedings of the 2013 International Conference on Advanced Mechatronic Systems*, 2013, pp. 302–305.

[87] “Analysis of camera calibration with respect to measurement accuracy,” *Procedia CIRP*, vol. 41, pp. 765–770, 2016, Research and Innovation in Manufacturing: Key Enabling Technologies for the Factories of the Future - Proceedings of the 48th CIRP Conference on Manufacturing Systems.

[88] K. Han and G. DeSouza, “Feature detection algorithm for autonomous camera calibration,” in *ICINCO-RA*, 2007.

[89] Matlab, *Evaluating the accuracy of single camera calibration*, https://www.mathworks.com/help/vision/ug/evaluating-the-accuracy-of-single-camera-calibration.html;jsessionid=43936be0664b39963b1c57d22bd7, website.

[90] H. Zollner and M. Kampel, “An evaluation of camera calibration methods using digital low cost cameras,” 2010.

[91] E. Rosten and R. Loveland, “Camera distortion self-calibration using the plumb-line constraint and minimal hough entropy,” *Machine Vision and Applications*, vol. 22, pp. 77–85, 2009.

[92] J. B. Joaquim Salvi Xavier Armangué, “A comparative review of camera calibrating methods with accuracy evaluation,” *Pattern Recognition*, 2002, doi:10.1016/S0031-3203(01)00126-1.

[93] G. T. Dr Mavris, *Introduction to quality engineering methods lecture*, 2017.

[94] D. P. Lain, *Stat 462 applied regression analysis pennsatet university*.

[95] Mathworks, *Residual analysis*, https://www.mathworks.com/help/ident/ug/what-is-residual-analysis.html.

[96] “Handbook of practical camera calibration methods and models optical metrology centre chapter 4 camera calibration methods,” 2003.

[97] Q. Fu, Q. Quan, and K. Cai, “Calibration of multiple fish-eye cameras using a wand,” *IET Comput. Vis.*, vol. 9, pp. 378–389, 2015.

[98] Z. Zhang, “Camera calibration with one-dimensional objects,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, pp. 892–899, 2004.

[99] M. Devy, V. Garric, and J. Orteu, “Camera calibration from multiple views of a 2d object, using a global nonlinear minimization method,” in *Proceedings of the 1997*

IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97, vol. 3, 1997, 1583–1589 vol.3.

[100] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, R. Maimon Oded, Ed., vol. 22, pp. 1330–1334, 2000.

[101] J. Hays, *Cs 6476: Computer vision, georgia tech*, Lecture.

[102] Y. F. X. Zhang C. Wang and H. Lu, "Global homography calibration for monocular vision-based pose measurement of mobile robots," *International Journal of Intelligent Robotics and Application*, 2017.

[103] J. Mitchelson and A. Hilton, "Wand-based multiple camera studio calibration," 2007.

[104] S. Ramalingam, P. Sturm, and S. Lodha, "Towards complete generic camera calibration," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, 1093–1098 vol. 1.

[105] E. Hemayed, "A survey of camera self-calibration," in *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.*, 2003, pp. 351–357.

[106] M. S. Faugeras O.D. Luong Q.T., "Camera self-calibration: Theory and experiments," in *ECCV*, 1992.

[107] R. Hartley, "Kruppa's equations derived from the fundamental matrix," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 133–135, 1997.

[108] C. Lei, F. Wu, Z. Hu, and H. Tsui, "A new approach to solving kruppa equations for camera self-calibration," in *2002 International Conference on Pattern Recognition*, vol. 2, 2002, 308–311 vol.2.

[109] F. Remondino and C. Fraser, "Digital camera calibration methods: Considerations and comparisons," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 266–272, 2006.

[110] S. Seitzs, *Calibration and single view meterology, lecture university illinois*, http://dhoiem.cs.illinois.edu/courses/vision_spring10/lectures/lecture3_projectivegeomtery.pdf, 2010.

[111] D. Parikh, *Cs 6476, computer vision class at georgia tech, lecture 14*.

[112] M. Antunes, J. P. Barreto, D. Aouada, and B. Ottersten, “Unsupervised vanishing point detection and camera calibration from a single manhattan image with radial distortion,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6691–6699.

[113] W. Kusakunniran, H. Li, and J. Zhang, “A direct method to self-calibrate a surveillance camera by observing a walking pedestrian,” in *2009 Digital Image Computing: Techniques and Applications*, 2009, pp. 250–255.

[114] D. Parikh, *Cs 6476, computer vision class at georgia tech, calibration, triangulation and structure from motion*, 2018.

[115] P. Vision and R. Group, *Camera geometry ii, cos 429*.

[116] H. Chang and F. Tsai, “Vanishing point extraction and refinement for robust camera calibration,” *Sensors (Basel, Switzerland)*, vol. 18, 2018.

[117] S. Tony, “Video metrology without the image-to-ground homography,” in *2010 International Conference on Digital Image Computing: Techniques and Applications*, 2010, pp. 335–342.

[118] A. Dragomir, D. Carole, and F. Daniel, “Google street view: Capturing the world at street level,” *Computer*, vol. 43, no. 6, pp. 32–38, 2010.

[119] N. L. A. Ahmad Abbas Al-Ameen Salih, “The suitability of gps receivers update rates for navigation applications,” *International Journal of Mechanical, Industrial and Aerospace Sciences*, 2013.

[120] G. Huang, “Visual-inertial navigation: A concise review,” *International Conference on Robotics and Automation (ICRA)*, 2019.

[121] B. W. Peijun Zhaoy Chris Xiaoxuan Lux, “3-d motion capture of an unmodified drone with single-chip millimeter wave radar,” *International Conference on Robotics and Automation (ICRA)*, 2021.

[122] M. V. Valerie Sessions, “The effects of data quality on machine learning algorithms,” *Proceedings of the 11th International Conference on Information Quality*,

[123] J. Tellinghuisen, “Statistical error propagation,” *The Journal of Physical Chemistry A*, 2001.

[124] N. S. A Matsumura, “Competing with quality information,” *Proceedings of the Conference on Information Quality*, 1996.

[125] R. A. Arthur Elmes Lyndon Estes, “Accounting for training data error in machine learning applied to earth observations,” *Remote Sensing*, 2020.

[126] J. H. V. Asudeh Abolfazl Jin Zhongjun, “Assessing and remedying coverage for a given dataset,” pp. 554–565, 2019.

[127] P. Geurts, “Bias vs variance decomposition for regression and classification,” *Data Mining and Knowledge Discovery Handbook*, R. Maimon Oded, Ed., pp. 749–763, 2005.

[128] K. Weinberger, *Lecture 12: Bias-variance tradeoff, cornell cs4780 sp17*, https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote12.html, Lecture.

[129] P. F. T. Cabreira L. Brisolara, “Survey on coverage path planning with unmanned aerial vehicle,” *Drones*, 2019.

[130] H. A. Idriss, L. Krichen, and F. Mohamed, “Simulation tools, environments and frameworks for uav systems performance analysis,” in *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2018, pp. 1495–1500.

[131] C. Jeff, M. Robin, and B. Jenny, “A survey of commercial amp; open source unmanned vehicle simulators,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 852–857.

[132] Gazebo, *Urdf in gazebo*, http://gazebosim.org/tutorials?tut=ros_urdf, Gazebo tutorial.

[133] ROS, *Rotors simulator, package summary*, http://wiki.ros.org/rotors_simulator, ROS tutorial.

[134] ——, *Hector quadrotor, package summary*, http://wiki.ros.org/hector_quadrotor, ROS tutorial.

[135] B. M. Furrer Fadri, in. 2016, ch. GRotorS—A Modular Gazebo MAV Simulator Framework.

[136] OpenCV, “Camera calibration and 3d reconstruction,”

[137] M. F. A. Hassan, I. Ma’arof, and A. M. Samad, “Assessment of camera calibration towards accuracy requirement,” in *2014 IEEE 10th International Colloquium on Signal Processing and its Applications*, 2014, pp. 123–128.

[138] C. S.-L. Mostafa Elgendy Tibor Guzsvinecz, "Identification of markers in challenging conditions for people with visual impairment using convolutional neural network," *Applied Science*, 2019.

[139] A. J. B. Daniel Scharstein, "Fast recognition of self-similar landmarks," *IEEE CVPR*, 1999.

[140] D. M. Negre Amaury Pradalier Cedric, "Robust vision-based underwater target identification and homing using self-similar landmarks," *Field and Service Robotics: Results of the 6th International Conference*, 2008.

[141] N. Aswini and S. Uma, "Obstacle avoidance and distance measurement for unmanned aerial vehicles using monocular vision," *International Journal of Electrical and Computer Engineering*, vol. 9, pp. 3504–3511, 2019.

[142] M. C. P. Santos, L. V. Santana, A. S. Brandão, and M. Sarcinelli-Filho, "Uav obstacle avoidance using rgb-d system," pp. 312–319, 2015.

[143] "Opencv: Detection of aruco markers," visited in 2021.

[144] F.-C. S.Garrido-Jurado R.Muñoz-Salinas, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[145] B. T. Zoltán SIKI, "Automatic recognition of aruco codes in land surveying tasks," *Baltic J. Modern Computing*, 2021.

[146] M.-S. R. Francisco J. Romero-Ramireza Rafael, "Speeded up detection of squared fiducial markers," *Image and Vision Computing*, 2018.

[147] FAA, *Part 107—small unmanned aircraft systems*, https://www.faa.gov/uas/commercial_operators/, Code of Federal Regulations.

[148] "Blender,"

[149] R. F. A. Ortega R. Galego, "Estimation of camera calibration uncertainty using lidar data," *European Conference on Mobile Robots (ECMR)*, 2013.

[150] D. Brown, "Close-range camera calibration,"

[151] D. Mavris, "Advanced design methods, chapter desing-of-experiment," 2017.

[152] D. M. J. Barros M. Kirby, "Impact of sampling technique selection on the creation of response surface models," *SAE transactions*, 2004.

[153] Y. Feng and J. Wang, “Gps rtk performance characteristics and analysis,” *Journal of Global Positioning Systems*, 2008.

[154] ROS, *Gazebo plugins in ros*, http://gazebosim.org/tutorials?tut=ros_gzplugins, Tutorial.

[155] A. Hayter, “Chapter 8: Inferences on a population mean,” *Probability and Statistics for Engineers and Scientists*, 2006.

[156] Gazebo, *Gazebo: Why gazebo, and gazebo features*, http://gazebosim.org/, Technical.

[157] DJI, *Dji mavic air specifications*, https://www.dji.com/mavic-air, Technical.

[158] Nikon, *Nikon: D3500*, https://www.nikonusa.com/en/nikon-products/product/dslr-cameras/d3500.html, website.

[159] Licthi, *Mission hub - litchi*, https://flylitchi.com/hub, app.

[160] R. Bose and J. Frew, “Composing lineage metadata with xml for custom satellite-derived data products,” pp. 275–284, 2004.

[161] D. Potere, “Horizontal positional accuracy of google earth’s high-resolution imagery archive,” *Sensors*, vol. 8, no. 12, pp. 7973–7981, 2008.

[162] J. L. Pierre Drap, “An exact formula for calculating inverse radial lens distortions,” *Sensors (Basel)*, 2016.

[163] X.-S. Yang, “Nature inspired optimization algorithms,” *SAE transactions*, 2014.

[164] K. V. Nguyen H.T., “Genetic algorithms:“non-smooth” discrete optimization,” *Applications of Continuous Mathematics to Computer Science*, 1997.

[165] I. S. Onur Boyabatli, “Parameter selection in genetic algorithms,” *Journal of Systemics, Cybernetics and Informatics*,

[166] S. N. Sinha and M. Pollefeys, “Pan-tilt-zoom camera calibration and high-resolution mosaic generation,” *Comput. Vis. Image Underst.*, vol. 103, pp. 170–183, 2006.

[167] S. M. Ayaz, M. Y. Kim, and J. Park, “Survey on zoom-lens calibration methods and techniques,” *Machine Vision and Applications*, vol. 28, pp. 803–818, 2017.

[168] Y.-S. Chen, S. Shih, Y. Hung, and C. Fuh, “Simple and efficient method of calibrating a motorized zoom lens,” *Image Vis. Comput.*, vol. 19, pp. 1099–1110, 2001.

[169] G. Thomas and O. Grau, “Virtual graphics for broadcast production,” *Computer*, vol. 42, no. 7, pp. 42–47, 2009.

[170] G. M. Jaime García Ramesh Thoranaghatte, “Calibration of a surgical microscope with automated zoom lenses using an active optical tracker,” *International Journal of Medical Robotics and Computer Assisted Surgery*, 2008.

[171] G. M. Jaime Garcia Ramesh Thoranaghatte, “Novel patterns and methods for zooming camera calibration,” *Journal of WSCG*, 2013.

[172] M. T., “Geometrical modelling and calibration of video cameras for underwater navigation,”