Online stereo camera calibration on embedded systems

Michal Szczepanski

Université Clermont Auvergne

Online stereo camera calibration on embedded systems

Michał Szczepański

supervised by

Mehdi Darouich

Engineer researcher in CEA

Erwan Piriou

Engineer researcher in CEA

Frédéric Chausse

Professor at the Université Clermont Auvergne

thesis defended 14.11.2019

in Gif-sur-Yvette, France

composition of the jury:

Berry François

Professor at the Université Clermont Auvergne

Ginhac Dominique

Professor at the Université de Bourgogne

Gouet-Bruenet Valérie

Research director of the National Institute of Geographic and Forest Information

Bouchafa-Brunea Samia

Professor at the Université d'Evry Val d'Essonne

Abstract

This thesis describes an approach for online calibration of stereo cameras on embedded systems. It introduces a new functionality for cyber physical systems by measuring the quality of service of the calibration. Thus, the manuscript proposes a dynamic monitoring and calculation of the internal sensor parameters required for many computer vision tasks. The method improves both security and system efficiency using stereo cameras. It prolongs the life of the devices thanks to this self-repair capability, which increases autonomy. Systems such as mobile robots or smart glasses in particular can directly benefit from this technique.

The stereo camera is a sensor capable of providing a wide spectrum of data. Beforehand, this sensor must be extrinsically calibrated, i.e. the relative positions of the two cameras must be determined.. However, camera extrinsic calibration can change over time due to interactions with the external environment for example (shocks, vibrations...). Thus, a recalibration operation allow correcting these effects. Indeed, misunderstood data can lead to errors and malfunction of applications. In order to counter such a scenario, the system must have an internal mechanism, a quality of service, to decide whether the current parameters are correct and/or calculate new ones, if necessary.

The approach proposed in this thesis is a self-calibration method based on the use of data coming only from the observed scene, without controlled models. First of all, we consider calibration as a system process running in the background and having to run continuously in real time. This internal calibration is not the main task of the system, but the procedure on which high-level applications rely. For this reason, system constraints severely limit the algorithm in terms of complexity, memory and time. The proposed calibration method requires few resources and uses standard data from computer vision applications, so it is hidden within the application pipeline.

In this manuscript, we present many discussions to topics related to the online stereo calibration on embedded systems, such as problems on the extraction of robust points of interest, the calculation of the scale factor, hardware implementation aspects, high-level applications requiring this approach, etc. Finally, this thesis describes and explains a methodology for the building of a new type of dataset to represent the change of the camera position to validate the approach. The manuscript also explains the different work environments used in the realization of the datasets and the camera calibration procedure. In addition, it presents the first prototype of a smart helmet, on which the proposed self-calibration service is dynamically executed. Finally, this thesis characterizes the real-time calibration on an embedded ARM Cortex A7 processor.

Kew words: online stereo camera calibration; smart glasses; extrinsic parameters; embedded systems; processing on embedded systems; calibration; auto-adaptation; self-healing; self-calibration; online camera monitoring; quality of services; real time;

Résumé

Résumé : cette thèse décrit une approche de calibration en ligne des caméras stéréo pour des systèmes embarqués. Le manuscrit introduit une nouvelle mesure de la qualité du service de cette fonctionnalité dans les systèmes cyber physiques. Ainsi, le suivi et le calcul des paramètres internes du capteur (requis pour de nombreuses tâches de vision par ordinateur) est réalisé dynamiquement. La méthode permet à la fois d'augmenter la sécurité et d'améliorer les performances des systèmes utilisant des caméras stéréo. Elle prolonge la durée de vie des appareils grâce à cette procédure d'auto-réparation, et peut accroître l'autonomie. Des systèmes tels que les robots mobiles ou les lunettes intelligentes en particulier peuvent directement bénéficier de cette technique.

La caméra stéréo est un capteur capable de fournir un large spectre de données. Au préalable, le capteur doit être calibrée extrinsèquement, c'est à dire que les positions relatives des deux caméras doivent être déterminées. Cependant, cette calibration extrinsèque peut varier au cours du temps à cause d'interactions avec l'environnement extérieur par exemple (chocs, vibrations...). Ainsi, une opération de recalibration permet de corriger ces effets. En effet, des données mal comprises peuvent entraîner des erreurs et le mauvais fonctionnement des applications. Afin de contrer un tel scénario, le système doit disposer d'un mécanisme interne, la qualité des services, pour décider si les paramètres actuels sont corrects et/ou en calculer des nouveaux, si nécessaire.

L'approche proposée dans cette thèse est une méthode d'auto-calibration basée sur l'utilisation de données issues uniquement de la scène observée (sans modèles contrôlés). Tout d'abord, nous considérons la calibration comme un processus système s'exécutant en arrière-plan devant fonctionner en continu et en temps réel. Cette calibration interne n'est pas la tâche principale du système, mais la procédure sur laquelle s'appuient les applications de haut niveau. Pour cette raison, les contraintes systèmes limitent considérablement l'algorithme en termes de complexité, de mémoire et de temps. La méthode de calibration proposée nécessite peu de ressources et utilise des données standards provenant d'applications de vision par ordinateur, de sorte qu'elle est masquée à l'intérieur du pipeline applicatif.

Dans ce manuscrit, de nombreuses discussions sont consacrées aux sujets liés à la calibration de caméras en ligne pour des systèmes embarqués, tels que des problématiques sur l'extraction de points d'intérêts robustes et au calcul du facteur d'échelle, les aspects d'implémentation matérielle, les applications de haut niveau nécessitant cette approche, etc. Enfin, cette thèse décrit et explique une méthodologie pour la constitution d'un nouveau type d'ensemble de données, permettant de représenter un changement de position d'une caméra, pour valider l'approche. Le manuscrit explique également les différents environnements de travail utilisés dans la réalisation des jeux de données et la procédure de calibration de la caméra. De plus, il présente un premier prototype de casque intelligent, sur lequel s'exécute dynamiquement le service d'auto-calibration proposé. Enfin, une caractérisation en temps réel sur un processeur embarqué ARM Cortex A7 est réalisée.

Mots clés : calibration de caméra stéréo en ligne; lunettes intelligentes; modèle de caméra; matrice essentielle; paramètres extrinsèques; systèmes embarqués; traitement sur systèmes embarqués; calibration; calibrage; auto-adaptation; self-healing; auto-calibration; surveillance de caméra en ligne; qualité des services; temps réel;

Acknowledgments

This thesis is the result of three years of work at within the Laboratory Adéquation Algorithmes Architecture Algorithm of the Institute d'Intégration des Systémes et des Technologies in the Commissariat à l'énergie atomique et aux énergies alternatives (L3A LIST CEA), in Saclay. I have been hosted by the Architecture and Design Department of circuits, Embedded Software (DACLE) in the Service Calculs et Systèmes Numériques.

I would like to thank Jean-René Làquepeys, Head of the DACLE, and Fabien Clermidy, Head of the SCSN, for their welcome and the means they provided me to accomplish this thesis in the best conditions. I would like to thank Thomas Dombek, Head of the Laboratory Adéquation Algorithmes Architecture Algorithm (L3A), for the trust and support he has given me throughout these three years.

I much obliged to thank Frédéric Chausse, Head of Dept. of Measurements and Instrumentation at Université Clermont Auvergne, for having accepted the direction of this thesis, for having participated in my jury and for help in this work. I would like to thank an each person from jury that accepted and judge my work: Berry François, Ginhac Dominique, Gouet-Bruenet Valérie, and Bouchafa-Brunea Samia.

It is hard to find words to express my gratitude to my mentor Mehdi, who helped me throughout the whole work, without whom it was impossible to complete this manuscript. Frequent discussions with him have contributed a lot to develop my thesis, skills and personalities. Big thanks for my second supervisor Erwan, who helped me in my work and on whom I could count in case of problems.

A very special gratitude goes out to my colleagues whose good humor, interesting discussion and brilliant ideas were part of my daily life: Alexandre, Erwan, Caaliph, Stéphane, Karim, Laurent, Thibault, Philippe, Youssouf, Eric, Maroun, Suresh, Christophe ... It was a great to share the same laboratory with you all over the last three years. I really appreciate my co-offices friends (past and present): Aziz, Dzila, Gregory, Fabrice, Khanh, Iman, Alix ... for their sympathy, good humor and for the interesting discussions that we had the opportunity to exchange.

In particular, I am grateful to Dominika who has assumed a number of duties to enable me to focus on work. My son who understood the nature of the final phase of writing the manuscript and was very forgiving. Thanks to my sister, parents and grandparents who provided me immeasurable help, moral and emotional support in my life and build-in curiosity to live. They made it possible for me to achieve my goals through continuous advice over the years. Finally, I would like to thank rest of all my family and friends for their support and interest in my work. Thank you to all the people I unfortunately forgot to thank.

Michał Szczepański

Contents

List of Figures

List of Tables

Chapter 1

Background and motivation

> Research is what I am doing when I do not know what I am doing.
>
> Wernher von Braun

The first chapter describes the guiding motivation and each of the three main contexts of this work. Then it briefly explains goals and challenges of each context. Next paragraphs in this section present the global approach to real-time stereo camera calibration on special devices. The chapter ends with a description of the structure and content of the entire manuscript. After this chapter, the reader should understand the main elements, problems and motivation of this thesis.

Objective :

Identify the main features of online stereo camera calibration applications in the embedded system and Cyber Physical System contexts. Presents the main motivation in the specific context of this work.

To do this, we :

- study basic concepts of the cyber physical systems.
- study stereo camera sensors, its parameters and limitations.
- study applications and hardware setups, where stereo camera calibration is required.
- present targeted system together with the application.

1.1 Cyber Physical Systems

In the twentieth century, the microcomputer evolution began and it continues. This sentence describes the rapid development of microprocessor-based computers. In the last fifty years, it completely changed our way of thinking, working conditions, communications, education systems and almost all aspects of our life. Despite many improvements, humans have not rested on our laurels. The Moore's law [120] is a perfect proof of such sentence. This observation says that the number of transistors in a dense integrated circuit doubles every two years (see Fig 1.16a). However, scientists expect that the law have its limits. Microprocessors have almost reached their limit regarding energy efficiency, according to [48] and [34]. The manufacture process in Semiconductor IC device fabrication is now (in 2018) 7 nm and it will reach 5 nm in 2020. CMOS scaling does not provide longer efficiency gains proportional to the increase in transistor density [3].

Nowadays, academic research and industrial work focuses more on the design of specialized hardware accelerators. It results in the recent development tendency, that those very popular personal computers (PC) used by us every day are becoming less and less important in our daily lives. The dedicated systems, which are called the Cyber Physical System (CPS) [2] have taken their role. Following chapter presents some of their aspects. One of the key results of CPS concept is that these systems are heterogeneous. Instead of one type of processors or core, the system achieves better performance not only by adding the same type of processor, but also by adding different processors that are dedicated to specific tasks. It is one of the main strategies for creating modern CPS [151].

CPS connects the physical world, through sensors or actuators, with the virtual world [95]. System typically consists of various components working together to perform missions and activities. Physical elements through a network or other communication technology interact with the environment [64]. An embedded system executes the programs and device's logic on different processors architecture due to heterogeneous technology. Fig 1.1 shows the standard CPS scheme.

Nowadays, in different areas such as: automotive, avionics systems, intelligent and smart buildings, Internet of Things (IOT), medical segment, automated and robotic manufacturing, devices to augmented reality and many others are based on the CPS. They can operate together on a large-scale system for many various purposes covering a wide range of applications. For this reason, they are becoming an important part of our lives. These systems capture more and more responsibilities in many areas of knowledge.

Their success and wide range of application are the result of significant price reduction compared to electronic systems a few years earlier. Today, some of the CPS performs specific tasks in real time, without the need for huge and powerful motherboards or cloud computing.

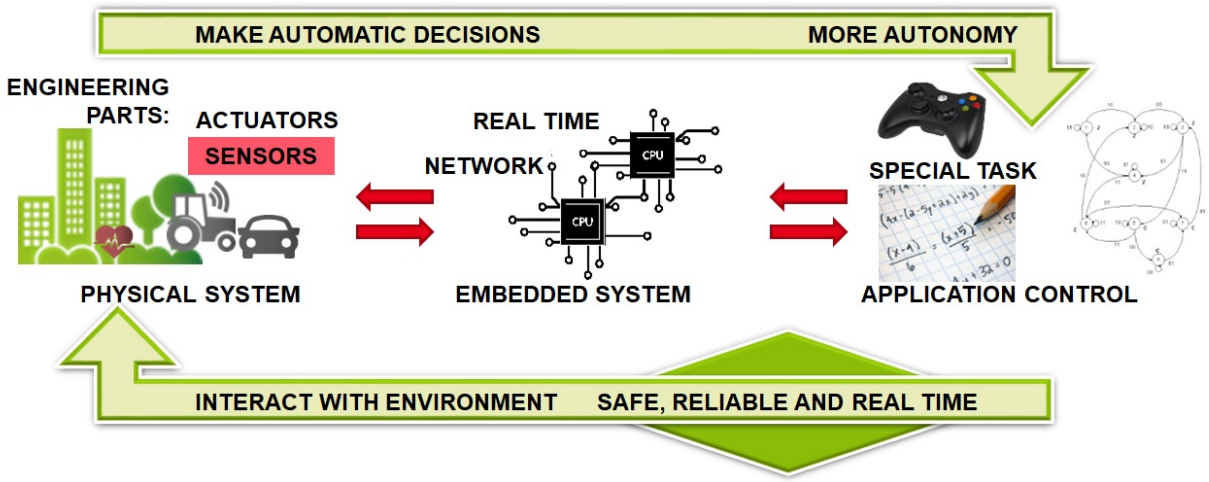


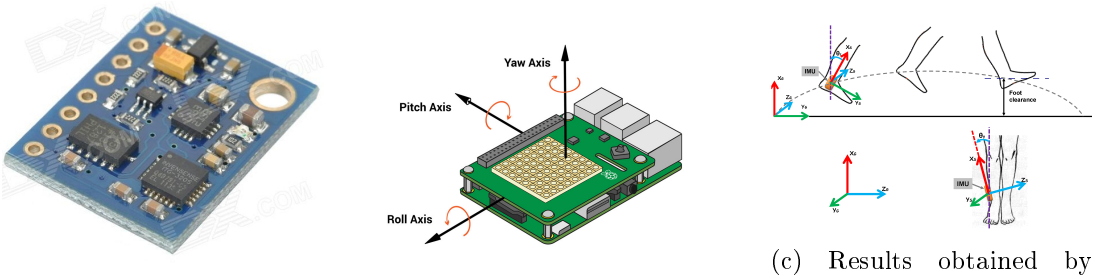
Figure 1.1: The standard cyber physical system (CPS) scheme.

Today, one of the main challenges for scientists and engineers is to create a modern CPS more independent, with a higher degree of autonomy. These systems should be automatic and sometimes even autonomous to allow mobility and portability, also to work in any environment in real time. To achieve these objectives, those systems must to be more intelligent. They have to increase perception, sense and better interact with the world around them. The acquisition, analyze and understanding of the environment must be properly satisfied. In order to carry mission, the CPS must monitor and control physical processes in the real time. Therefore, it is essential, that all components of the system operate as fast and accurately as possible to realize the task together.

1.1.1 Sensors in Cyber Physical Systems

There are many different approaches to increasing the ability to understand the world for the CPS. Many sensors designed for specific measurements can do this. All of them have some advantages and disadvantages. Regardless of their intended use, they have specific limitations and operating conditions. Therefore, there is no ideal sensor and solution. In most cases, it strongly depends on the application and devices. Certainly, the most popular methods provide a large number of sensors with sufficiently strong processing motherboard. Then CPS is usually continuously power, this solution works well. However, customers must take into account higher price of such a system. Moreover, it does not work for portable systems, which base on the smallest possible batteries. In addition, data fusion between multiple sensors requires good calibration. It is usually a complex and resources-intensive process. This subsection presents some of most interesting and popular sensors currently used in the CPS for localization and understanding the environment.

Inertial measurement unit knows as an IMU. An electronic device measures linear and angular motion, usually with a triad of gyroscopes and accelerometers [33]. It is presented is in Fig 1.2b and 1.2c. This commonly used sensor to collects data, which allows to position tracking by dead reckoning method [181], thus integrating angular velocity and acceleration in the sensor/body frame. Unfortunately, the measurement error of such sensor is significant and accumulated over time. Therefore, in the CPS, the GPS usually supports the IMU in order to correct the drift error. Depending on the requirements, especially such as precision, specific application uses different types of IMU. The price of a tool heavily depends on its precision and purpose. IMU may cost from few €like those one which are mounted in mobile phones to hundreds €for sensors in airplanes. For example: new IMU dedicated for drones and robotics proposed by Bosch (Fig 1.2a) is a 3×4.5×0.5 mm chip and it consumes around 5.2 mA.



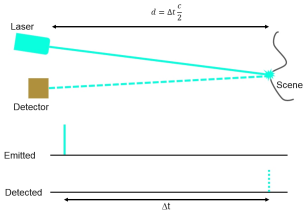
(a) IMU chip. (b) IMU ax explanation. (c) Results obtained by IMU.

Figure 1.2: IMU presentation.

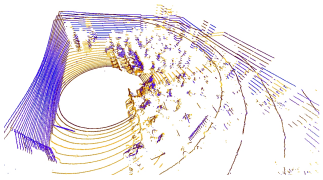
LIDAR is a sensor widely used in many devices and robots, to understand the local environment [7]. It measures distance to the target, by illuminating scene with pulsating laser light and by measuring the time from reflected impulses, Fig 1.3b shows this principle. It emits light in the near-infrared, visible or ultraviolet spectrum as opposed to radar, which operates on the same principles but in

the microwave domain. The received impulses are usually converted and interpreted as a 3D point cloud (Fig. 1.3c). Different precision and quality of measurements allow using this type of sensors in many fields such as: geodesy, archaeology, geophysics, robotics where it helps to detect and avoid obstacles [31]. The main disadvantage and the problem of the LIDAR is the computational load. It has to detect all points from the scene, optimize their position through time of light technique. If it is implemented using efficient hardware, it can work in the real time. This work [141] proves that a laptop equipped with 2.5 GHz quad cores and 6 GB memory, can handle necessary processing data, with some optimization. They propose to use a three-dimensional grid, which significantly reduced the number of points detected [187]. The various purposes of the sensors require different operating parameters, so there are various types of LIDAR available on the market. For example, the MRS6000 from SICK has an operating range of 0.5 m to 200 m. Its weight is a 2.2 kg, and require 20 W of power consumption. The URG-04LN HOKUYO operates at a distance of 0.6 cm to 40 cm and requires only 800 mA for full operation; the weight of this sensor is much smaller than the previous example and is about 0.2 kg. Fig 1.3a shows the Velodyne LIDAR, this model is widely used in the automotive industry. It is a sensor capable of delivering the most accurate real-time 3D data on the market 1.3a. The sensor creates a full 360-degree field of vision of up to 200 meters of environment. It requires 20 watts. Its size is significant 7 × 14 cm in size and weighs about 1 kg. The price of the sensor largely depends on the parameters of the detector and producer company. The reliable and widely used model on the market cost about 4 thousand €.

(a) Velodyne LIDAR.



(b) Principle of LIDAR work.

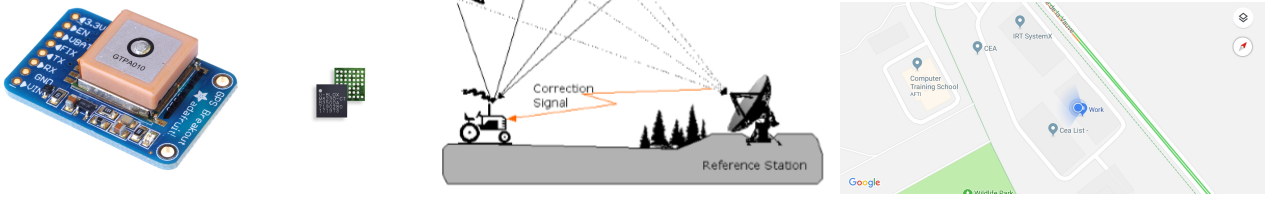


(c) Results obtained by LIDAR.

Figure 1.3: LIDAR presentation.

Satellite navigation is a system that uses an artificial satellites with radio waves to provide an autonomous principle of Geo-spatial positioning [70], as it is shown in Figure 1.4b. Based on the radio signal, it is possible to estimate the position given in latitude and longitude with an error between 6-12 m on the Earth's surface [127]. This position can be placed on the map, as it is shown in Fig 1.4c. Moreover, the continuous position of motion can be determined, wherever the signal is available. The most popular system is the Global Positioning System (GPS), but there are also many other alternatives, such as Galileo, Baid, A-GPS or GLONAS [9]. GPS signal receiver (see Fig 1.4a) is a module available in many modern devices, such as smartphones, laptops, mobile robots, cars, etc. Depending on the application, there is a very wide range of sensors with different performance,

precision, size and weight range. The NEO-6 u-blox 6, UBX-M8230-CT or module GTPA010 are a GPS chip, it reaches the size of a 3×3×0.4 mm, which consumes 20 mW for 30 minutes of track and requires power supply of 2.7 V - 3.6 V. The precision of this type of chips vary, depends on the model, and it is proportional to the price [1].



(a) Module GTPA010. (b) Principle of GPS work. (c) Results obtained by GPS.

Figure 1.4: GPS presentation.

Odometer is a method of measuring the distance, through translation determined by the position of the sensor or agent in relation to its initial position in time. It is not a basic sensor, but a method that uses impulses from actuators to estimate motion data [16]. For robot platform on wheels or legs, a mileage counter or rotary encoders can be an interesting and necessary source of data, for estimating the current and past locations. This allows estimating the relative position and the distance traveled from the starting point of your journey. Unfortunately, the odometer suffers heavily with precision problems. The wheels used to slip and slide on the floor, so the method accumulates the measurement error over time. On the other hand, the main advantage is cost. It does not require any special and advanced mechanical or electronic components. It can only process the pulse received on the wheel and send it to the microcomputer.

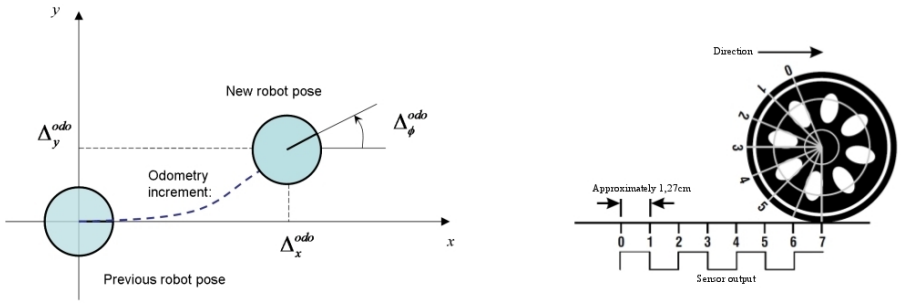


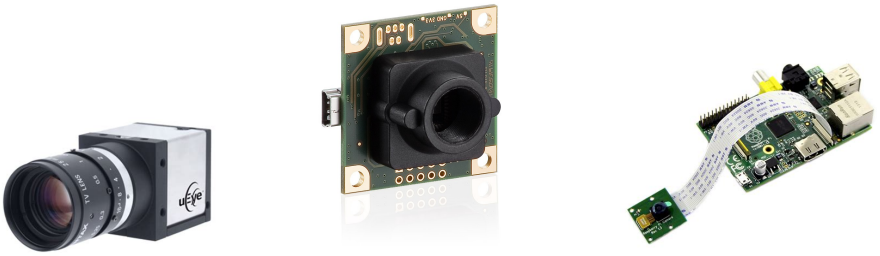
Figure 1.5: Principle of the odometry.

1.1.2 Cameras in Cyber Physical Systems

By interpreting the images provided by the camera, it is possible to obtain all the information necessary for understanding the surrounding environment and performing multiple missions. Object detection

and recognition, 3D mapping and location, navigation and many other tasks can be performed using camera's data. Today, these are widely used in many applications and CPS [56]. Many heterogeneous single-boards computer, such as the Raspberry Pi (RPi) (it is shown in Fig 1.6c) are already equipped with a camera and able to do simple image processing in the real time [45].

Active camera is usually complex system, equipped with many other sensors, where camera is only one of them. The Microsoft Kinect V1 & V2 [4] or Asus Xtion Pro [5] are an examples of the active camera sensor systems with heterogeneous architecture. Another common connection of sensors is a combination between LIDAR and camera [57] [50]. It is highly difficult to estimate the distances in the real environment by a single camera [75]. Therefore, the combination with LIDAR is suitable for this and used in many robots and mobile vehicles that require navigation. Another increasingly popular method of active vision is a combination of standard camera with infrared camera citeAlhwarin2014. For example, the smartphone may use it to improve facial detection and increase camera parameter settings [158] [91]. There are many different camera models on the market adapted to work with different sensors due to different applications associated with various restrictions and requirements. Active image processing and storage can take place on small, integrated circuits, in many devices such as smartphones. The size of powerful CMOS cameras (see Fig 1.6) is around 32×32×20 mm and may cost less than 100 €[53]. The weight of such sensors is less than 5 g, and power consumption is around 100 mA.



(a) IDS camera.

(b) IDS camera on chip.

(c) Camera connected with RPi Pi.

Figure 1.6: Cameras presentation.

Stereo cameras are a set of two parallel cameras called a passive 3d depth sensor. This can eliminate the need to use a LIDAR or other sensor in the depth estimation process. This type of solution can estimate distance in the local environment up to a certain range.

This sensor imitates a biological process and creates a vision system known as binocular (stereopsis) vision [19]. It can create a depth map, which is a three-dimensional image of the surrounding environment, obtained from a two-dimensional view, done from two vantage points of the cameras. The depth map helps to recognize and identify an object from the images, it allows compute a distance in the image, etc. Of course, it has its price, larger amount of data requires more data processing, in order to interpret information from images [66]. However, stereo cameras are becoming more and more popular

and promising sensors, due to the extremely large amount of data, they can provide [99]. Formula presented in 1.1 shows, that camera's parameters limit the depth extracted from images. Focal length, pixel specification and baseline (refer to real distance between two cameras) impact on measured data. As depth becomes greater, disparity tends to zero. The disparity and size of pixel represent the field of view. It has a significant impact on depth calculation, but higher resolution cameras compensate it [86].

$$\text{depth} = \frac{\text{baseline} * \text{focal length}}{\text{pixel disparity} * \text{pixel size}} \tag{1.1}$$

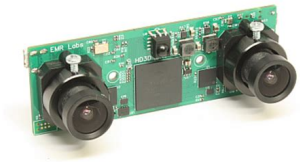
In traditional approaches of modern passive stereo camera systems (stereo cameras without additional sensor), the whole construction is usually mounted into one rigid and stable cage. The biggest disadvantages of this solution are that the baseline between the two cameras is strict, construction is heavy and big which does not fit many of applications. On the other hand, such a design prevents the camera from movement. This note is extremely important, because it assures that once determined camera's parameters (focal length, position etc. see at 1.2.1), are constant and does not change during any mission, where passive stereo vision is used.

There are some of passive stereo vision sensors available in the market. Fig 1.7b presents ZED sensor, it costs 450 €. Many applications from mobile robots to augmented reality devices can use it [8]. It provides two CMOS sensors with 4 M pixel resolution, which use 380 mA. The system provides depth resolution at a distance of 0.5-20 m where the stereo baseline is 12 cm. The size of the sensors is significant and equals 17×3×3 cm with weight of 160 g. It can operate with 60 frames per second with resolution 2560×720.

Fig 1.7c shows another interesting stereo sensor the Blaxtair [6] product. Its purpose is to distinguish a person from another obstacle in real time. Once detected, warn the operator in case of danger up to 6 meters. The sensor operates in harsh outdoor conditions mounted in a construction vehicle. There is a more passive stereo sensor in the market like sensor from Fig 1.7a but all of them are limited due to a fixed and constant position of cameras.

Another interesting stereo sensor is the Blaxtair product is shown in . Its purpose is to distinguish a person from another obstacle in real time. Once detected, warn the operator in case of danger up to 6 meters. The sensor is mounted on a construction vehicle that operates in harsh outdoor conditions. There are more passive stereo sensor in the market like 1.7a but all of them are limited due to fixed and constant position of cameras.

The infrared radiation camera (IR) is an active vision sensor independent of lighting conditions. Fig 1.8 presents this sensor where the main principle of working is similar to standard camera. However, the wavelength range in the infrared radiation camera spectrum is different and is between 700

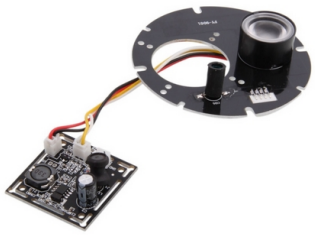


(a) HD3D Lite 3D FPV Camera.

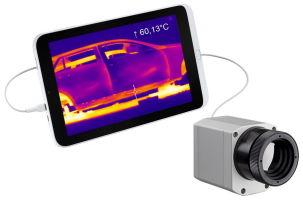
(b) StereoLabs ZED camera.

(c) Blaxtair camera.

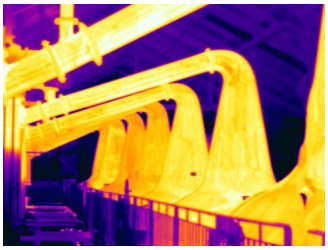
Figure 1.7: Passive stereo vision sensors (stereo cameras).



(a) Thermographic camera.



(b) IR camera with display.



(c) Thermograpic image.

Figure 1.8: Infrared radiation (IR) Camera presentation.

nanometers to 1 micrometer, where standard light visible camera operates from 380 to 700 nanometers. Significant reduction of sensor costs in recent years allows searching for new applications [139]. Infrared cameras can be used to improve image understanding from the standard camera, in processes such as face recognition, depth extraction, etc. 1.3.3. For example, the Microsoft's Kinect motion sensor uses the IR camera. Operating conditions, miniaturization and other parameters are similar to standard camera. Fig 1.8a presents model than costs less than 4 €and it is available in small chip's dimensions 2cm×2cm where weight does not exceed 50 g.

1.1.3 Conclusion

IMU, LIDAR, GPS, odometry, cameras and other sensors can provide a wide range of valuable information. This is required to react and accomplish many missions, in a known or unknown environment. However, higher amount of different sensors cause a complication in the CPS. More data from the sensors into the system requires more computing processing, consumes more time and resources [35] [47]. Different data spectrum provided by many sensors requires synchronization and calibration. This process is a challenging task, which requires many resources. In addition, as the number of sensors increases, the cost of the whole system growths proportionally. However, it is not always possible to place all sensors due to the constructions, maximal size, weight, required power supply energy, design etc. There are always specific scenario and device's requirements. For example, the size and energy required by LIDAR are significant for some of the devices, like smartphones. In addition, the price of the sensor can be higher than the whole device. We conclude that, the use of sensors correlates

and enforces the target devices and application. One of the main criteria in many aspects is price and simplicity, so the simplest and cheapest possible solutions are the most valued by industry and market.

The stereo camera is forward-looking sensor, which can provide enough data to understand and localize system in the surrounding world. It can eliminate and replace many other sensors. Obviously, it has some disadvantages, the biggest one is its standard design. Stereo camera sensor is in large, weight and rigid cage, which is usually very difficult to fix and install. After assembly the cage and use the two parallel set of camera sensor can be susceptible to many different external factors or forces. Then, it requires a human intervention such us maintenance or repair which is complicated or sometimes impossible. From this reason, the sensor requires tough and robust construction, which can guarantee the stability. Those cameras cannot move and change it pose in respect to each other. This solution results in high price on the market. Finally, this consideration hides the underlying crucial stereo camera problem. That the camera's pose exposes to unexpected changes. It must be guarantee that it does not happened, in order to work properly. For this reason, our work has focused on one of the classic problems, in the field of computer vision, which is online stereo camera calibration, in the specific CPS context. This is the subject of the next subsection.

1.2 Stereo camera calibration context

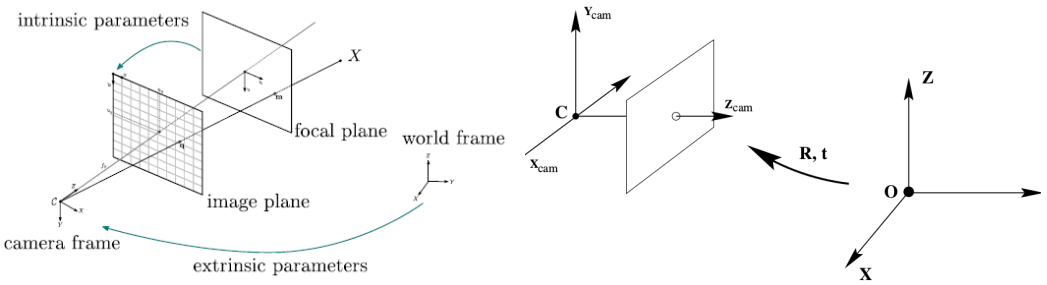
The calibration is a process that determines the relationship between the values of the measured quantity indicated by the measuring instrument and the corresponding values of physical quantities. The calibration is extremely important and is required for many activities and tasks [81]. Without this process, it is impossible to obtain a reference to actual values and correct interpretation of the results. For the same reason, the cameras require calibration. The camera calibration is the process, which provides a multiple parameters that define and relate to the specific characteristics of the camera [69]. These are critical during extracting 3D information from 2D images, measure object size in global units, visual-odometer process, reconstruct a 3D scene and many applications in computer vision and robotics domain citeSong2013 [72].

1.2.1 Cameras parameters

Fig 1.9 presents the simplest representation of the projection model known as the pinhole projection, which referees to camera. There is the light-sensitive surface (sensor) and the image plane with lens (projection) in a given position and orientation in the space. In order to describe it, there are two group of parameters.

First presented group is intrinsic or internal camera parameters. Those represents the relationships between the coordinates of pixels and the coordinates of the camera frame. They remove distortions caused by camera lens imperfections. They find the true center of the image and set correct distance of focal length.

The second group of parameters is extrinsic or external camera parameters. These express the relation between the coordinates of the different camera poses (Fig 1.9b). Moreover, in a stereo camera case, it represents the relation between two camera's positions. They allows reconstructing the 3D world model from cameras views. They represents the related position, which are used to the understanding of the environment.



(a) Pinhole camera model. (b) Extrinsic parameters.

Figure 1.9: Intrinsic and extrinsic camera parameters.

1.2.2 Limitation of standard approach

The computer vision application requires all camera parameters in order to work properly. Because of this, many studies investigated the camera calibration process. It led to the fact that calibration procedures run in the manufacturing process of the cameras. Many of calibration methods are dedicated to monocular camera calibration, which allows user to find only intrinsic camera parameters of one camera [188] [118] [190]. On the other hand, many stereo camera calibration methods try to estimate all the necessary, i.e. intrinsic (distortion) and extrinsic parameters. The most popular methods use classical and traditional approach [170] [144] [160] [189] [12]. Those methods try to find a known pattern or special calibration object with known size in the scene and link with the observed scene. Fig 1.10 shows the examples of 2D patterns.

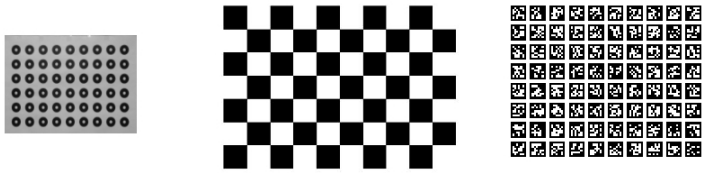


Figure 1.10: Different calibrations patterns

The camera construction guarantees constant focal length of the camera. If this is satisfied, the intrinsic camera parameters can be set once at production level. On the other hand, the extrinsic camera parameters should be constant if the camera's position does not change. This is true, if and only if, cameras are well fixed and mounted in the rigid frame. Fig 1.11b shows the example of such camera. However, real scenarios and the impact of environment expose the sensor on unpredictable strong shocks, vibration etc. In that case, camera's position can change. Then in the classical approach, the user must send back the sensor to the manufacturer. There are usually specially trained employees, who uses the traditional method to calibrate, thus compute a new camera pose/parameters. This type of the offline camera calibration consumes a lot of time. It requires special patterns, knowledge, etc., which it is not always available. This kind of method is not practical and it is expensive. This is why researchers are trying to find a new procedure, which allow realizing cheaper, easier and more general available camera calibration method.

1.2.3 Case of loosely attached cameras

The research subject in this work focuses on loosely mounted stereo cameras. Fig 1.11b shows the proposition of such type of sensor, where two cameras are not into a rigid cage. Many CPS such as smart glasses, vehicles or robot platform can benefit from such easily mounted cameras. In this setup, unfavorable environmental conditions and forces exposes cameras to at any time. The mechanical vibration, impact of obstacles, large temperature fluctuations, material tensile strength and many

others exist in real life scenarios and may change camera's parameters. Then, during a long-term mission, the system cannot assume and guarantee that the position of stereo cameras does not change.

Many functions and applications require well-calibrated camera parameters. Therefore, the system must be able to verify the parameters of the calibration at any the time. The procedure should run from time to time or online during the whole mission. In a real environment, it is difficult to determine the period when calibrate the cameras. It is because an unexpected change in the position of the camera may occur unexpectedly. In this situation, when the cameras are loosely connected, continuous online camera calibration seems to be necessary. The existing traditional approaches cannot handle this challenge.

Many calibration methods without patterns appear in the last thirty years [71] [170] [182] [160]. The self-calibration knows also as camera auto-calibration method. They use only camera motion in a static environment [111] [159]. This group of methods aspires to be online, i.e. executes while the system is working. The methods do not require any special calibration object in the scene. They seem to be a good candidate for that type of sensor in order to, calculate continuously extrinsic camera parameters. In theory, it is possible to perform these methods anywhere, if there are not any special method's limitations [121]. Successively some of the methods try to analyze camera motion in a stable environment, using Krupp equations, epipolar lines [80], absolute dual quadric and its projections. Unfortunately, some of them only calculate the intrinsic and distortion parameters of the camera, thus are dedicated for monocular camera.

1.2.4 Conclusion

Many computer vision processes requires knowledge of intrinsic and extrinsic camera parameters. They are very important and critical for many different applications, because of this they must update.

The intrinsic parameters that defines internal parameters of the camera can be set only once at the beginning. In this work, we determine that they will not change because you can be sure that the focal length of the camera is fixed.

(a) Stereo camera mounted in to a rigid metal cage.

(b) Loosely attached stereo cameras.

Figure 1.11: Two types of stereo cameras, loosely attached and fixed into rigid cage.

Unfortunately, it is not case for the extrinsic camera parameters. In some system, where cameras are loosely connected and attached to devices, see Fig 1.11b the camera's positions and orientations can change due to initial position. Then, the continuously extrinsic camera parameter estimation is required.

As presented in section 1.2, there are many different approaches to solve one of the classic problem in the computer vision - camera calibration. Unfortunately, the best, most practical and precise method does not exist. There are many of them dedicated to different specific CPS, application, environment or to a special scenario. In second chapter, we present a division into several groups of methods, which we explore and describe with the details.

Therefore, this work challenges the topic of online stereo camera calibration and tries to optimize and explore what can be done in a specific context. The main aim of this work is not to develop a new calibration method, but to explore existing methods and their possible use in a specific CPS. Such calibration should be performed online in real time during the mission, in an unknown environment, without special patterns and other attributes. It must be based solely on camera data.

This type of improvement has a positive impact on many aspects of the whole systems. It increases ability to perform operations for a long time without errors. Such systems equipped with this functionality will avoid returning to manufacturing when the recalibration process is required. This is a requirement improvement necessary to obtain the new functionalities and confidence in CPS. It will allow cameras mount without in a heavy rigid frame. The online calibration is the key to efficient and more accurate operation of the future equipment.

1.3 Application and devices context

The motivation of this work is to find one, most universal calibration method, that can be done online on specific CPS in an independent environment. The best stereo calibration method does not exist. Engineers strongly adjust many different procedures to the specific environment's and the application's requirements. In this section presents the analysis of different types of applications and missions.

1.3.1 Environment

The short study related to different types of environment is a very important context for the analysis of applications. We divide the main environmental categories in four main groups: known, unknown, static and dynamic areas.

Missions carried out in a static and known environment are less complicated because they are more predictable. The easiest scenarios to analyze are those where missions always take place in the same known locations. For example: devices on production lines or robotic mobile platform for warehouse can operate on the same path or repeat the action in closed, limited areas, in such situations, the environment can be upgraded, so that a calibration pattern is always available. This is perfect scenario for the traditional calibration method. This approach to solve the problem of stereo calibration is appropriate and often used. However, the system requires this specific and adapted environment. Therefore, it is not suitable for all type of CPS that may work in unprepared or random, undefined environments.

The systems, which realize mission in a dynamic and unknown environment, are much more complex. It is hard to predict and test all possible behaviors. The scenarios cannot rely on proper scene construction because it is impossible to ensure that some specific elements are available to use at any time. In a dynamic scenario, many different conditions can occur when using the devices. The system must be prepared and capable of reacting, so it must be more versatile and independent. Therefore, it may have a wider range of applications. Advanced autonomous systems should be able to work in a dynamic and unknown environment, without prepared a scene.

Of course, all the time several special conditions must be satisfied in each environment, in order to record correct images from cameras. The most important factor for a stereo sensor is sufficient lighting, to make the camera work efficiently. Night scenarios, or those with little light, require additional sensors to support camera data and in such case, the ability to differentiate elements on stage is required. A situation in which the image is uniform and homogeneous for a long time is unacceptable. For example, if there is a perfect white wall without any elements in the whole camera's view, it is impossible to distinguish any information. Then, this kind of data from cameras is not sufficient for any CPS mission.

The stereo camera calibration method without any limitations, except the sufficient light and

heterogeneous scenes is sought. It should have a possibility to run in any dynamic environment, i.e. internal: in buildings, station, public transport and external in a city, forests, roads, etc. Fig 1.12 presents how looks like a typical environment for camera calibration.

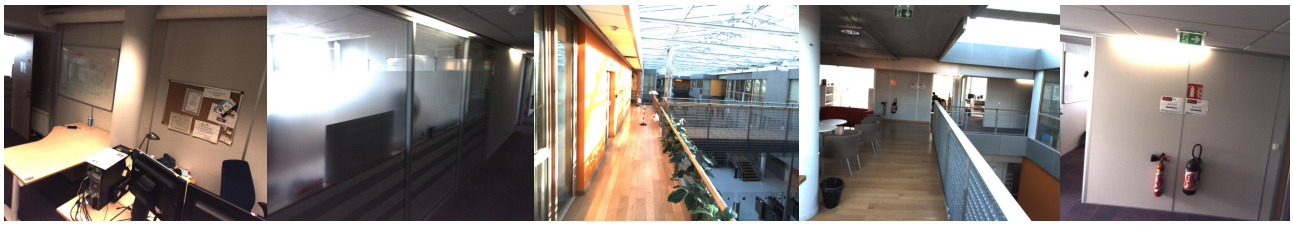


Figure 1.12: Examples of analysed type of scene - local environment.

1.3.2 Mobile robot platforms

Mobile robot platforms are CPS that need to interact with the environment. They expect to become increasingly useful in many different areas, such as autonomous vehicles, warehouse systems, cleaning robots etc. The wide spectrum of mobile robot application requires different approaches for stereo camera calibration.

Car and ADAS

Automotive production is one of the largest market values in the world [113]. Its economic potential constantly drives and enforces change. Continuous optimization improves many features of new car models. Road safety is a very important automotive aspect. According to World Health Organization (WHO), traffic accidents are one of the main causes of death in the world [132] because of this, there is a huge need to improve the car safety and reduce this statistic. Advanced driver-assistance system (ADAS) can help to solve this problem. It supports behavior on the road and improve driver safety. It can perform many activities such as human and road signs detection; calculate automatic maneuvers to generate collision-free trajectories, emergency stop systems to avoid collisions, etc. Fig 1.13c illustrates the center of the car in which the ADAS supports a driver. Today's cars often have more than 150 different subsystems. Many of them like: mirror and seat adjustment, air condition, brake control, etc. are a separate system usually controlled by CPU, such as ARM, Intel or other processor, like nVidia's DrivePX which is dedicated to the Artificial Intelligence (AI) [55] [129]. It leads modern vehicles, to have up to 50 processors and multiple sensors. Moreover, cars contain a huge amount of software that can have up to 10 million lines of code [149].

The 1rst models of vehicles equipped with a stereo camera sensor already appear in the automotive market [171]. Usually the current stereo camera exists as one sensor (one box), which makes it impossible to get a wide base line between the cameras. The most common location of such a sensor is under the front mirrors on the windshield. Nowadays, this sensor provides only additional data to

support more complex ADAS. Undoubtedly, car with motor can power all necessary electronics, which consumes a lot of energy, so the extra processor board does not complicate a system. The capacity and size of the whole car are sufficient comparing to the heavy cage, stereo sensors. Moreover, the use of a large number of sensors and electronics is possible because the car is a unique type of system for which many people are willing to pay more than for other products. Health and comfort are highly valued. Thus, in such type of the system, it can be equipped with expensive stereo camera systems. It is significant that the price of stereo camera in the comparison with the whole cost of a car (an engine, bodywork etc.) is negligible.

For these reasons, the automotive application is different from other CPS and the problem of the online stereo camera calibration is mostly overlooked because the cameras do not have the right to change their location in the expensive, rigid and complex stereo camera system. Nowadays ADAS try to propose a news solution, for example mount the cameras on the side mirrors. This solution increases a baseline between cameras, thus depth and field of view. In such case, it is mandatory to look at the online camera calibration because this type of installation requires continuous monitoring of the camera's position, because the mirrors are movable.

Currently, the system is full of data from various sensors, different applications, etc., which can be used to calibrate stereo cameras. The odometer and IMU data provide an additional location information; this data overcomes the limitations of GPS/GNSStechnology, the dead reckoning. This data helps when satellite signals are not available e.g. in tunnels, parking garages. The calibration method could use the road infrastructure. Some methods can work with standardized sizes of road signs, pedestrian and route lines, etc. and try to use them as a traditional calibration pattern. Moreover, if camera are attached in mirrors to aid different drivers, the movement of this is limited due to its construction. Onboard there are many different sensors. These type of methods can calibrate stereo cameras in the car. However, all of them are limited to this specific application context. They are not relatively universal and not possible to perform in real time. At the TED conference Elon Musk said, "Vision is the most critical sensor for the future autonomous driving system. Once you solve cameras for vision, autonomy is solved; if you don't solve vision, it's not solved" [124]. These words suggest that the vision becomes more popular and stereo camera can be the sensor providing required data.

Vacuum mobile robots

Fig 1.14 illustrates the vacuum mobile robot, which must have the intelligence to move and localize itself in the environment. New models comes with a new supplementary function, such as understanding a local area, ground, etc. In the current popular models, the micro-controller is the heart of a system. A bumper equipped with infrared sensors and an odometer to measure the traveled distance usually provides the data. It has a common DCpower unit consisting of a 12V battery, which allows vacuuming

(a) Mobile Robot Pioneer.

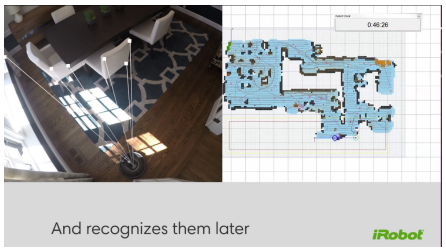
(b) Autonomous car.

(c) Advanced driver-assistance.

Figure 1.13: Different CPS where stereo camera can provide a key data.

(a) The Roomba 960 presentation.

(b) The Roomba 960 presentation.



(c) The Roomba 960 recognize visual landmarks and construct the floor plan.

Figure 1.14: Vacuum robot presentation.

and navigating. The price of the mobile vacuum robot is in ranges from 150 to 1000 €. Fig 1.14a shows one of the latest and most advanced models Roomba 960. This model is equipped with a camera sensor. It runs on built-in Linux, executed on ARM9 System on Chip, where 2 MByte FLASH and 16 MByte SDRAM is available. The system has a WiFi Router that allows to control it through a network.

Roomba 960 unlike to older generation robots does not use a random pattern to decide where to go. It uses information from the environment provided by camera and navigation algorithms [83] [134]. Thanks for this data, the robot understands where it has already been, where to go along the straight lines and not to repeat the same cleaning area, if it is not necessary. Roomba can slow down the speed of movement in front of the obstacle. Fig 1.14b shows application realized to create a map with detected objects such as a table, chairs, walls, etc. After making sure it has cleaned the entire surface, its navigation allows the robot to return to base and recharge the batteries.

Generally, customers do not want to spend a lot of money on consumer electronics devices. For popularization and more frequent use vacuum cleaning robot, it has to cost as less as it is possible. Expensive, heavy and rigid stereo cameras are not a good candidate as a main sensor. However, two loosely connected cameras can be a good and optimal alternative. Due to the commercial scope of the robot, technical documentation is not available. Probably the precise environment mapping is a result

of correct data fusion provided by the robot's wheel odometer, gyroscope, accelerometer, and camera. In the future, the stereo camera can provide a wide sufficient spectrum of data to the system. Thus, it can eliminate and replace other sensors and electronic equipment used in robot so far. The demanding customers require more precise and more complex vacuum robots for lower price. Two loosely attached stereo camera can reduce number of other sensors so minimize a price and increase perception of robot.

The last aspects that we analyzed in such context is an importance and priority of cleaning mission. It is definitely different from ADAS. If the mission goes wrong, there is no accident involving casual lives, unlike to vehicles or cars. For this reason, the vacuum robot calibration method has completely different limitations. It can be less complex and provide parameters with worse precision.

1.3.3 Virtual and augmented reality devices

The virtual and augmented reality devices collect, process and control data from internal and external sensors, in order to add the 2D, 3D figures or artificial information generated by computers to the display with real environments. Fig 1.18shows the example of such information. Fig 1.15 presents the special devices like portable smart glasses or helmets. Those are able to change the optical properties of the universal environment around us in the real time. This area is quickly gaining popularity in recent years.

(a) The Hololens proposed by Microsoft.

(b) The smart helmet proposed by our lab.

EXAMPLE OF SYSTEM	HOLOLENS 2
PROCESSOR	Holographic PU + CPU
MEMORY	64GB + 2GB RAM
BATTERY	2-3h of work
PRICE	3500 $

(c) The Hololens details.

Figure 1.15: Augmented and virtual reality devices presentation.

The first successes of virtual reality motion sensor, which try to capture the information, received in the real world and pass it to the computers was Microsoft's Kinect. It entered into the Guinness Book of Records as "the fastest-selling consumer electronics device" after more than 8 million copies were sold in the first 60 days [155]. It shows a huge potential and great perspective in the future for this kind of devices. The Kinect is equipped with two cameras. The first standard RGB video camera provides video images with a resolution of 640×480. The second camera is part of the whole sensor subsystem, which returns information about the depth of the local environment. It uses the infrared illuminator, which displays a cloud of points in front of the camera. The infrared camera sees its positions and size. Thanks to this data, it can calculate the distance to the local environment. The obtained 300 on

200 resolution of depth interpolates to the resolution of the video camera (640×480). The operating range of the distance sensor is between 0.4 and 6.5 m. This data allows detecting people and gestures through a software application. The use of the structural light method can flawless operation of the sensor only indoor - the reading is sensitive to excessive sunlight [114].

There are already several models of smart glasses or helmets available on the market. Fig 1.15a presents the Hololens, one of the most interesting models proposed by Microsoft [109]. It is a first generation of these kind devices, still large in size and weight (half a kilogram). This makes it uncomfortable for a long time of use. Intel ×32-bit architecture inside can run the Windows 10. Moreover, custom Microsoft Holographic Processing Unit supports the main processor. It has a 64 GB Flash and 2 GB glsram memory. It has a wide range of sensors: glsimu, 4 environment understanding cameras, 1 depth camera, 12 MP HD video camera, 4 microphones, 1 ambient light sensors and mixed reality capture. All sensors work together for the perception of the local environment. These elements and the expensive in term of power allow to active use during only 2-3 hours. Other companies offer commercial smart-glasses products such as Atheer [156] or Lumus [62].

Only during last 30 years, engineers were able to reduce the size and price of smartphones several times. That nowadays everyone can use it, due to its low costs and wide range of applications. If similar process will run for such virtual and augmented reality devices, those can be also widely use and change many aspect of everyday life. There is a high probability that the same history repeat. For this to happen, the price of such device must fall drastically. At present, an energy-intensive processor does not allow for long-term use, so the life of the device (without charging) should be extended. Loosely mounted cameras can replace multiple sensors that need costly hardware. It can eliminate expensive electronics and gain many computation operations. It can reduce the data fusion and other mathematical operation etc. Particularly, the device cannot be heavy and large, while the devices is located on the head. According to its use cases, it has to face many limitations. Such as in smartphones, the design, size and weight of the product are very critical. We could not imagine a walking with big phones carried in suitcases, but we do it with small devices fitting our pockets. Therefore, extending the size of battery increases the weight and size of the devices. Because of its proximity to the user's head location, we must consider the temperature and cooling process of the device. The many high complex processing has to reduced, accelerated and optimized in order to use less energy. Searching for less complex algorithms, optimizing the code and converting it to less powerful processors are desirable solutions. Devices of virtual and augmented reality must understand the surrounding environment. One of the best sensors in terms of size, power consumption, data spectrum and price is the stereo camera, in the form of two loosely connected cameras. Therefore, the question of their exact parameters estimation during the mission is important. Thus, the online

calibration of cameras can be a milestone for virtual and augmented reality devices.

1.3.4 Conclusion

Section 1.3 recalls some of the applications that can significantly benefit from the use of a stereo camera. It is a matter of time, when these mentioned examples use a stereo camera.

The mobile robot platform can use a stereo camera for navigation purpose. It can provide a wide range of data about the local environment. It can allow for safer movement, avoidance of barriers, etc. Depending on the platform, sensors and applications running on it, the calibration of stereo camera cameras may use different variants and methods.

Future augmented and virtual reality glasses and helmet can be much cheaper thanks to use only stereo cameras. This sensor can provide all the necessary information for proper functioning. The calibration method required by the augmented reality devices has definitely less data to use in the system. It has limited battery power, etc. than robots mobile platform such autonomous car. Depending on construction of smart helmets or glasses, it can be less susceptible to decalibration. Not less, certain form of verification whether the extrinsic camera parameters are correct, is required.

Section 1.2.4 shows existing methods with limitations. Additionally, this part specifies what kind of method is looking for in this work. The conclusion is that the ideal calibration method does not exist because it depends strongly on environment application and devices.

When exploring different calibration methods, it is important to find a specific context for the work. It is mandatory to know the target system, its possibilities, limitations and purpose in order to propose the best method. Therefore, for the purpose of this work, section 1.5 presents in detail our motivations.

1.4 Embedded system context

Embedded system is a special purpose computer system (controller programmed and controlled by a real-time operating system (RTOS)), which becomes an integral part of the computer's hardware. It must meet specific requirements strictly defined for its tasks. Therefore, it is not a typical multi-functional personal computer. Each embedded system is based on a microprocessor (or micro-controller) which is usually a part of System on Chip (SOC), and software programmed to perform a limited number of tasks (or even just one), very often with real-time computing constraints. Embedded systems are not always standalone devices. Many of them consist of small parts within a larger device that supply a more general purpose.

Over the past decades, the SOC has taken a big step forward. The Moore law is a good representation of these changes, it shows in Fig 1.16a how the number of transistors on the same surface in the processor, increase in relation to the limited period of time.

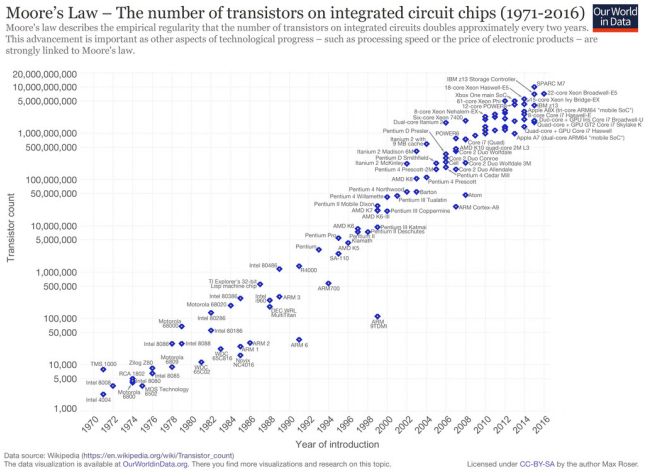
Fifty years ago, the huge computer occupied an entire room. They realized simple data process and had less computing power than a modern smartphone, which fits into pockets. Twenty years ago microwave had a simple CPU, which was high, modern embedded technology at that time. Nowadays, many children's toys use similar CPU, which costs less than 10 euro. SOC integrates all components of a computer or other electronic system used in an embedded system. These components typically include a complete system consisting of multiple electronic part such as a central processing unit (CPUs), graphics processing units (GPUs), multipliers, caches, memory, input/output ports, etc. Fig 1.16b shows SOC diagram, which presents ARM processor supported by all peripheral devices. The SOC is a complex embedded system, but fully integrated on one chip.

Nowadays, various technologies and languages implements specific programs depending on the task. The most common and popular programs use standard form of code that follows the instructions executed on the CPU. There are some of limitations and restrictions of such approach i.e. number of computation or speed. According to this, there are other solution, for example, use a data processing through other specific integrated circuit (ASCI) or field-programmable gate array (FPGA).

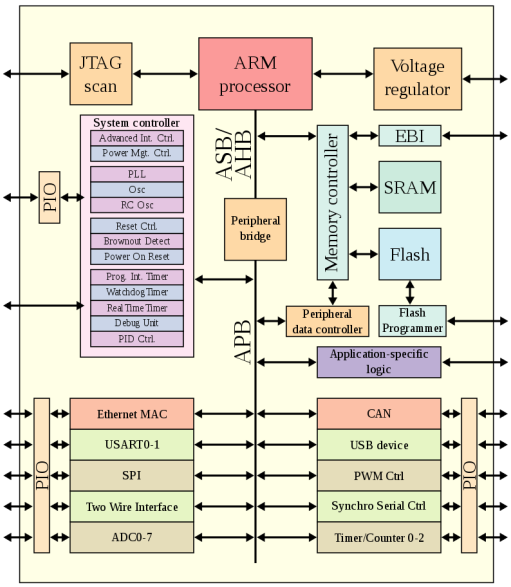
1.4.1 Vision vs Hardware

Many years ago, computer science had a high spectrum of knowledge. Development of this field of science created new specializations and communities. That has resulted in the fact that the knowledge and region of interest of the hardware development community that deals with embedded systems and computer vision community that focus on computer architecture significantly diverges.

The algorithm for vision applications and advanced systems require an extremely large number of mathematical operations so a lot of computational power. Moreover, the scientists and engineers are usually interested in the fastest and most precise so the most complex algorithms. Luckily, for them,



(a) A plot of CPU transistor counts against dates of introduction.



(b) Microcontroller-based system on a chip.

Figure 1.16: CPU presentation.

they have access to advanced, intelligent and heterogeneous hardware architecture that often consists of several powerful CPUs and GPUs. On such systems, they perform multiple complex processes simultaneously, without caring about many other restrictions. Modern hardware architecture performs and accelerates increasingly complex tasks. That is why nowadays, many algorithms are not well adapted and optimized to work on embedded systems. Many systems have reached the point where there is a strong need to use these computer vision algorithms and implement them in the new CPS with the low performance CPU. Therefore, the subject of research on the borderline between a computer vision and an embedded system is very significant and promising nowadays.

1.4.2 Personal Computer

The researchers of computer vision algorithms work on a standard personal computer (PC), and they do not worry about hardware limitations. The huge computational power is available due to expensive technology and high power consumption of such systems. It is possible thanks to many specifications such as the connection to a fixed power supply, a cooling fan because the temperature of processing units often exceeds 60 degrees Celsius, etc. The cost of this type of heterogeneous architecture often exceeds 500 €.

The standard PC contains the hardware, so all physical parts of computer and software, which provides instructions to realize different tasks. Thanks to today's solutions, it is hard to see the boundary between hardware and software in a modern PC. The motherboard connects all processors, memory, and peripherals together with embedded libraries and other solutions. RAM, graphics card and processor are in most cases mounted directly on the same motherboard. Today's the average

consumer does not need to know and understand their parameters to use them. In order to add a new hardware to PC, usually plug and play concept works. The most important software on each PC is the Operating System (OS). Mainly thanks to it, it is possible to communicate with the hardware. It helps and translates many instructions to specific computer languages. It allows to carry out specific tasks by use of particular commends. Due to this software, many of electronics elements and missions work. However, many processes require the operation system with special library, in order to use new tools and new user applications. There are available different OS, but the most popular for robots and CPS in the world is the Linux. It is free and open-source software, which gives great possibilities for internal interference.

1.4.3 Single-board (Ready-made) Computers

Fig 1.17 presents some of already-made worldwide-accepted boards such as the Arduino, Orange Pi, ZYNQ, Pynq or Raspberry Pi (RPi). This type of devices are perfect examples of SOC, which allows testing and creating new prototypes, very fast and cheap. It is interesting to see a performance of RPi, which has an Advanced RISC Machine (ARM) architectures family. Those CPU use 32 bits instructions sets to execute code. The reduced instruction set computing (RISC) allows having a lower number of cycles per instructions compared to most CPU used in modern PC, which usually have $\times$ 86 architecture.

In Fig 1.17a presents some processors characteristics, for example, the ARM Cortex A7 in the RPi 3 is about 7 times less powerful that CPU Intel 7 [20]. However, the price of the Cortex ARM family is much cheaper than the Intel processors. Moreover, the power consumption is significantly lower and it translates into less heat dissipation. These ARM's performances are ideal for lightweight, portable, battery-powered and low-cost devices. These processors can be dedicated devices for augmented reality, low-performances tablets and other CPS.

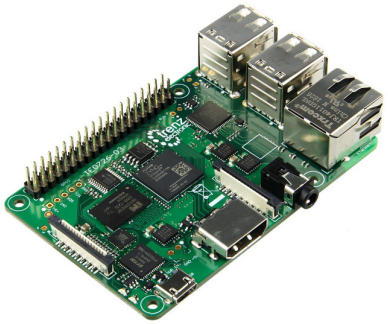
Of course, the construction of the CPU's system has some limitations. During the designing process, the engineers of embedded systems challenge the limitation of computational power and memory. As a result, it is impossible to implement the latest and the most complex algorithms in embedded systems based on ARM processors. This forces the search for a certain balance between simple and complicated algorithms. Moreover, we must consider the amount of data and the not linear calculations, in order to implement it on limited CPU.

1.4.4 Conclusion

Huge changes in an embedded system in recent years allow analyzing and creating new devices with new functionalities. Until recently, many functionality innovations have been outside the scope of technology. Fortunately, today's continuous improvements create a new possibilities and insights on complex subjects. One of them is to look at solving the stereo camera calibration during real-time on

	PC Intel Core i7 7500U	Embedded (Cortex A7)
IPS/ AT	x150 =49,360MIPS	2,850MIPS
MEMORY	x62500 = 32GB	512KB
TDP	x10 = 25W	0.5-1.9W
PRICE	x10 = 393$	30$

(a) Some characteristic between ARM Cortex A7 and Intel Core I7.



(b) ZYNQ Berry.

(c) Raspberry Pi - RPi

Figure 1.17: Embedded platforms equipped with ARM processor

embedded systems.

The typical SOC equipped with embedded processor has low power consumption, small size, rugged operating ranges, and low per-unit cost compared to the PC. This comes at the price of limited processing resources, which make them significantly more difficult to program and interact with other components. Those kind of integrated circuits equipped with ARM architecture on $\times 32$ bits can be acceptable candidates for many CPS.

In this work, we select an embedded systems platform with the ARM Cortex processor as a target system. For this reason, we consider only the most primitive methods of online stereo camera calibration. Many methods were developed and characterized on the $\times 86$-bit architecture. Therefore, this research tries to find and suggest a method with methodology, which executes in real time on selected limited embedded systems.

1.5 Our motivation

Finally, after presenting all four contexts of this work, we present our particular motivation. At this stage, we presented that the stereo camera is a good sensor for modern CPS, because it provides a high spectrum of data, on the other hand it can be cheap, small and require low energy. These features perfectly respects a low performance of embedded systems. Some devices and applications already use stereo camera, but it has still some limitations. One of them is camera calibration. There are many different camera calibration procedures. However, there is no ideal method to calculate the extrinsic parameters of a stereo camera in each possible scenario and applications. Moreover, there are not many studies about online calibration procedures.

The goal of this work is to show and illustrate an approach to extrinsic online stereo camera calibration that performs on embedded systems. It is important to mention that the goal is not to develop and create a new calibration method, but to investigate whether one of the existing procedures can be adapted and performed in real time on limited embedded systems on CPS.

1.5.1 Main aim of approach

In this thesis, we would like to execute the online stereo camera calibration in specific CPS with a particular mission. We consider the calibration method in the first generation of virtual reality devices seen as intelligent glass or helmet. Its main objective of such devices is the pedestrian guidance. Fig 1.18 presents possible scenario. This type of device must monitor and control the movement of the user, localize itself everywhere in the dynamic environment. In order to detect and avoid obstacles, interact, a high spectrum of data in the system is required. The important approach is that this CPS has a special dedicated group of recipients. It must support visual impaired people and facilitates their mobility. This type of system can be an alternative to a guide dog. The mission goal has a very high priority and it is important from the user's point of view. It must provide trusted and accurate information, because the health of the user and the success of many activities are important.

In this section, we consider limitations of such system. The smart glasses must be relatively small and lightweight to be portable and comfortable. It forces the use of a small battery, thus a system has a very limited amount of power. It requires the use of a small energy demanding processors. Moreover, in the glasses frame, it is hard to build a large CPU or GPU. These limitations led to the reduction of all electronics components. Sensors that consume too much energy or are too large do not fit into this type of system. It is important to remember that the vibrations, unpredicted forces and much more may affect the construction of the glasses.

1.5.2 Navigation

The system has a specific mission to realize, it supports the person during the guiding along route. The history of navigation is very long and has always been with humanity. Etymology of the word derives

Figure 1.18: Interface for application to led a pedestrian presentation.

from the Latin term for a sail. Over the centuries, humans distinguished many types of navigation: dead reckoning, which is a process of calculating one's current position by using a previously determined position, celestial navigation, which based on celestial bodies, radio, radar or satellite one. Section 1.1.1 describes some of the sensors used in navigation.

Nowadays, navigation referees as the classic problem of how to deal with the determination of the current location and optimal route to the destination for people, ships, land vehicles and other moving objects based mainly on satellite navigation. Due to its small size and weight, quite low price of the signal receiver chip, many devices such as smartphones, cars and planes obtain a satellite navigation module that can obtain signals from satellites. The GPS is an excellent type of navigation in an outdoor open space, environment, such as motorways or urban environment. Unfortunately, due to an error and lack of precision and signal in internal scenarios, this method is not sufficient for navigation requiring by a pedestrian guidance. Another problem that satellite navigation does not solve is information about the local environment and obstacles around the user. In a realistic dynamic scenario, in cities and inside buildings, there are a huge number of obstacles to avoid. It is an important aspect for the CPS, which must help navigate a visually impaired person.

There are sensors that can provide information about the local environment as part of smart glass. The LIDAR and radar seem to be the right choice for this type of task. Unfortunately, they have many disadvantages, especially in the context of glasses. This type of device must be portable, which makes it necessary to minimize the device with small batteries. The LIDAR sensor consumes a lot of energy and requires a huge amount of data processing. Mounting this sensor in glasses would be complicated due to its large size and embedded system's limitations. In addition, the cost of this LIDAR often exceeds what users want to pay for such product. Therefore, it is not ideal sensors for this type of equipment. Glasses cannot use a traditional odometer technique. Another sensor that can provide data to the system is the IMU. Unfortunately, the error in estimating the position due to the IMU accumulates over time. The price of this sensor is close to a price of the camera, but the spectrum of provided data is limited.

Two cameras as a stereo camera can provide sufficient data for a navigation mission. Such solution can reduce need of other sensors so the cost of the devices may decrease. It eliminates data fusion, thus

the whole system requires less computing power. The camera is an optimal sensor for smart glasses devices. The size and weight of the product is sufficient to mount it in the glasses frame.

Figure 1.19: Smart portable devices in our context of work.

1.5.3 Conclusion

The goal of this work is to realize the online stereo camera calibration on a smart glasses/helmet equipped with embedded systems. This CPS is dedicated to navigation of visually impaired people. The stereo camera provides data to the system. Its working conditions create many limitations. Due to the design and dynamic missions of the devices, the position of cameras can change. Therefore, it is necessary to calibrate extrinsic camera parameters all the time.

In order to accomplish its mission, the system must provide data, which allows moving from any location to another. Unlike other navigation devices, this type of system must be capable of understanding the surrounding dynamic environment. It must know the distances to the local environment, which help for obstacles detection and avoidance. It should be able to locate, track and measure distances traveled, select correct route, path, corridors, etc. Fig 1.18 presents an application, which leads the user to the closest exit and can display additional information.

To realize navigation, many missions must realize a depth map extraction, SLAM, visual odometer and path selection, etc. There may be many additional functions such as the object detection and recognition. The sound signal and many other lateral functions can help to avoid any barriers and obstacles. All these functions have to work together, process data in the real time. Immediate reaction and decision-making are the key in this type of system.

The online stereo camera calibration can be of great innovation for this CPS. It increases its reliability and creates the ability to carry out a long-term mission without the need for maintenance or operator assistance. As a result, research topic relates to self-healing and self-adaptation of the device. Auto-calibration (self-calibration) creates a continuous measurement of the stereo camera parameters in the real time independent of location or mission.

1.6 Summary of background and motivation

1st chapter gives all the necessary background information and motivations to understand the basic principles of online stereo camera calibration, it is divided into four different sections. Fig 1.20 sums up all the different aspects and contexts of this manuscript refereed in the first chapter.

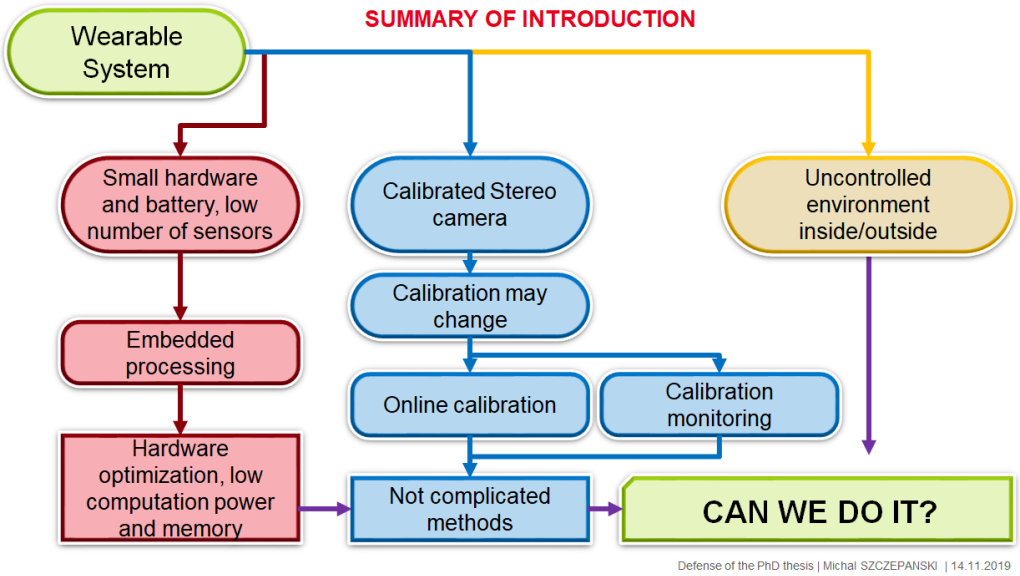


Figure 1.20: Summary of the first chapter in the diagram form.

Section 1.1 provides fundamental facts about the CPS. It shows many trends in the development of new systems. We point to the fact that, the modern systems need to be more intelligent, autonomous, independent and operate longer without the help of people. In order to carry many tasks, the CPS need to collect data from the environment. Section 1.1.3 presents the advantages and disadvantages of various types of sensors used in systems.

The one of the most future-oriented sensor is a stereo camera. Most of the models currently used are built into a rigid metal frame. However, due to its size, weight and cost, it does not t the most CPS. Removal of these restrictions by setting up loosely attached two cameras can be a big impulse to popularize and widely use this sensor in many new devices.

Part 1.1.2 shows that the use of stereo cameras can provide a very wide spectrum of data. Thus the system can reduce or even eliminate the need for other sensors thus reduce cost of many CPS. However, the processing based on the data from the stereo cameras has a some restrictions and limitations.

All computer vision processes require the intrinsic and extrinsic camera parameters, which describes the relation internal and external setup of stereo camera sensor. In order to calculate them, the camera calibration method is required. Section 1.2 presents the limitations of the standard, traditional calibration approaches. Unfortunately, removing rigid metal cage in the sensor creates new restrictions, such as the need for the continuous stereo camera calibration. The extrinsic parameters so a relative camera's position can change. In this case, the classical approaches are not suitable for the sensor.

In the literature, the perfect stereo calibration method does not exist, which can compute parameters repeatedly. The targeted application, devices or hardware strongly correlate and force the calibration procedure. Following, section 1.3 gives analyze of some of targeted environments and applications, which can significantly take profits of use online stereo camera calibration.

The next part of the 1st chapter shows embedded systems as another context of this work. The different CPS are based on various embedded CPU, those are usually a several times weaker than standard used in PC. The computing and memory limitations must be considered during calibration procedure executed on embedded systems.

The last part of this chapter describes our specific motivation to realize the online extrinsic camera calibration. Section 1.5 presents the smart-glasses with particular mission to realize, its working conditions and limitations. In this the selected CPS, the stereo cameras are mounted in a fragile design, which is exposed to many external factors that can cause camera's position movement. Therefore, the approach to real time calibration is extremely important. In such targeted CPS, an embedded system is an additional problematic aspect, due to its construction. For this reason, we consider the hardware limitations.

This PhD thesis is not focused on the creating a new camera calibration method, but in implementing and testing one selected procedure on the targeted embedded systems. Section 1.3.4 characterizes the particular method. This research is future-oriented in order to realize more autonomous and reliable CPS. The proposed method allows creating a system with the highly demanded functions such as: self-healing and self-adapting

Next 2nd chapter presents the state of the art. It presents the survey of much method. It explains the technical background and the basic concepts associated with the stereo camera calibration problems.

The 3rd chapter describes the developed approach to the online stereo extrinsic camera calibration. It presents an algorithm and the suggestion about the whole methodology of advanced calibration. In this chapter, we present the system improvements like: additional filtering or accumulation strategy. Moreover, many aspects of camera calibration, such as: frequency of execution, use of data from the system, acceleration of calculations, scale extraction, quality of calibration services are addressed and discussed in this chapter.

The 4th chapter presents different environments of tests. It shows the methodology used for realization of the dataset. In this chapter, we present the online stereo monitoring and camera calibration tests on the PC and the targeted embedded system. We characterize and comment the obtained result.

Last 5th chapter concludes and summarizes the work. It recalls the obtained results and the answers to the questions raised in the manuscript. This chapter presents a perspective of the whole work and proposes a future work.

Chapter 2

State of the art

The only source of knowledge is experience.

Albert Einstein

The considered custom method has to be a part of the whole CPS in a specific application domain. This chapter describes the existing stereo camera calibration methods. It describes the some specific parameters, limitations and constraints of presenting methods. The chapter ends with a description of the datasets available in the literature. After this chapter, the reader should know the most popular calibration methods.

Objective :

Selection of the best method for online calibration pipeline on embedded system in specific selected device and application context and the best database to test an approach.

To do this, we :

- study stereo camera calibration methods.
- study methods that aspire to real time execution.
- study methods that execute on an embedded system.
- study datasets for online stereo camera calibration.

2.1 Introduction to state of the art of calibration methods

In the last decade, scientists have studied various methods of calibrating cameras. This section provides an overview and in-depth analysis of some of these procedures. In Fig 2.1, we categorize the most important methods due to their limitations. We assess them according to the criteria presented in the first chapter, namely: embedded system and application context. The section 2.2 presents traditional methods, which use a calibration object in order to work. Many computer vision applications use them. This kind of group is popular when engineers assume that the camera parameters are fixed. In this thesis, we negate this statement. The section 2.3 presents self-calibration methods. These do not require any special calibration objects. However, they need a camera motion into static environment, which is usually the case for many computer vision processes. This group is less precise and stable. The third group represents these methods, which use a different constraint than calibration patter and motion, for example: the know rotation of camera, vanishing lines or information from additional sensors.

Various assumptions are obligatory depending on a specific method. Every calibration method requires well-synchronized images provided by cameras. The scenery must have sufficient light, in order to detect many features in the images. Another, very important aspect is the overlapping view of cameras. Both cameras must simultaneously observe the larger part of the image (scene). There are methods to calibrate cameras that do not overlap views such as [176]. However, this is not the research subject of this thesis to consider these methods.

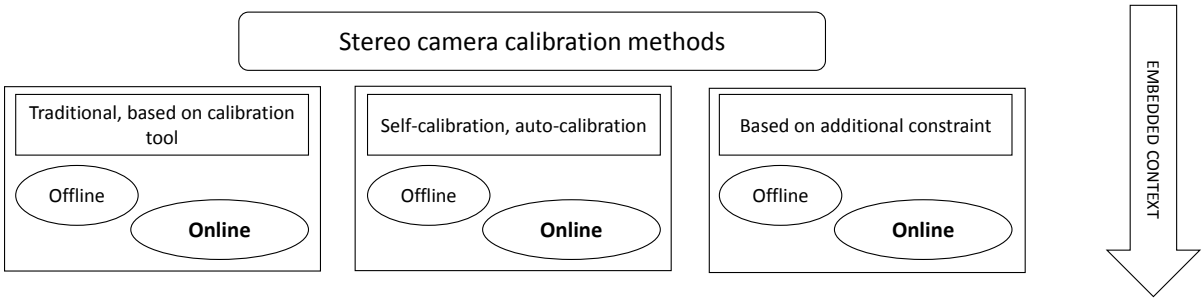


Figure 2.1: Global calibration methods characterization.

2.2 Traditional Camera Calibration

This section presents the most popular approaches of the traditional camera calibration methods. Those require a special calibration tool or pattern, in order to work. Fig 2.2 shows the four various forms of that tool. The pattern must be clearly visible (the whole tool) in the camera view from many different positions in relation to the static position of the camera. There is a reverse option in which the calibration tool has a stable position and it is observed from multiple camera positions (orientation). Depending on the different methods with a particular pattern, a various number of images from many perspectives is required. The traditional camera calibration method usually must stop application, system or current task to realize this procedure. Then, it must find and detect a calibration pattern, from many different views. Therefore, the traditional calibration method does not aspirate to have a potential to run in real time (during task). The procedure has to know the size of the calibration pattern. Each traditional method extracts clearly the coordinates of the calibration object. It creates a set of collinear equations thanks to its position. Then, the algebraic assumptions of the projection geometry solve this system of equations. Finally, the method usually estimates all camera (intrinsic and extrinsic) parameters. Linking the image views obtained from cameras with the particular calibration patterns can provide very accurate results not achievable by other methods.

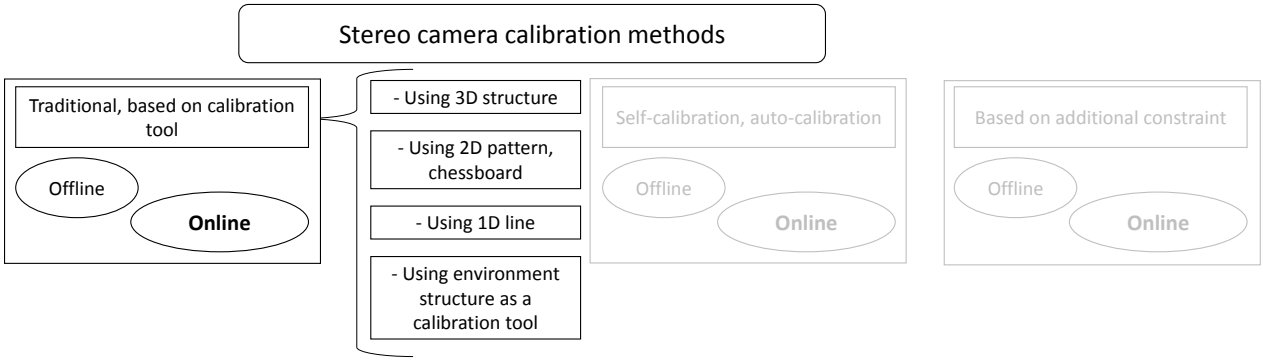


Figure 2.2: Traditional calibration methods characterization.

3-dimensional pattern structure

The first group of method presents calibration, which uses an apparatus or other calibration object with 3-dimensional geometry. The left part of Fig 1.3 shows the calibration object proposed by Faugeras [49]. Such apparatus normally consists of two or three perpendicular planes. Fig 1.3 presents type of calibration pattern with the 2-dimensional structure (in different planes), in order to imitate a 3-dimensional tool. This calibration room proposed by Geiger et al. realizes the procedure with only one camera shot (image) [58]. Thanks to the 3-dimensional of pattern, there is no need to move cameras and take images from different poses. Heikkila describes all necessary steps required by traditional method [71].

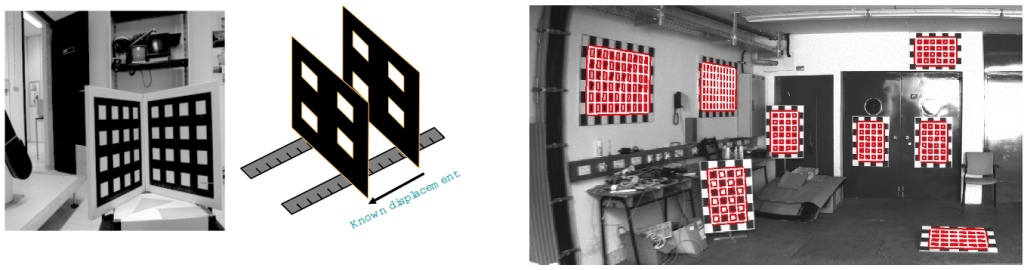


Figure 2.3: Different 3-dimensional apparatus for calibrating cameras.

2-dimensional pattern structure

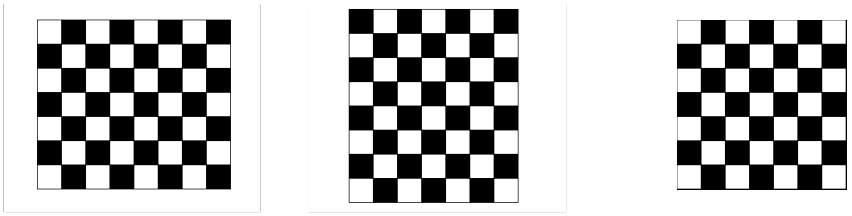
The second group is the most common type of the traditional calibration method, which uses 2-dimensional classical chessboard pattern. This group has a huge representation. During this thesis, we considered some of them [170], [144], [160], [189], [188], [12]. In the market, there are applications to calibrate stereo cameras. One example is the Matlab stereo camera calibration toolbox [108]. However, the license of such software is not free. The other one, the OpenCV is open source software, recommended by us [131]. Both applications have implemented the same Zhang method [189]. There is much more application available on the market. However, in this thesis, we use these two as the reference methods.

In these methods, the structure and parameters of the 2-dimensional calibration pattern is important. It has to be easy to detect, so the high contrast of pattern is required. We recommend printing the black squares on the white background page. Fig 2.4 shows how the most common pattern, which looks like the chessboard. We recommend using the chessboard, that contains an even number of squares and the other side has an odd number of squares. In this configuration, the program that detects the pattern will never confuse the left top corner with the bottom one. The calibration method must know the size of square, so the real scale and the precise value of the parameters can be determined.

The 2-dimensional pattern-based method is more flexible compared to another pattern-based method

because if a printer is available, it can be easily prepared. Of course, we should properly protect ourselves against bending the flexible sheet of paper where the pattern is present. To calibrate the cameras, we should only detect the pattern on one plane. Therefore, sheets of paper with a chessboard printed on should be on a solid, flat surface.

There are commercially patterns created on the solid material such as glass or metal one. Those calibration tools are precisely prepared to obtain the most accurate precision. They are much more expensive than those prepared by the standard printer. Moreover, their weight is much higher, which makes it more difficult to use.



(a) Good structure of pattern size 7x8.

(b) Good structure of pattern size 8x7.

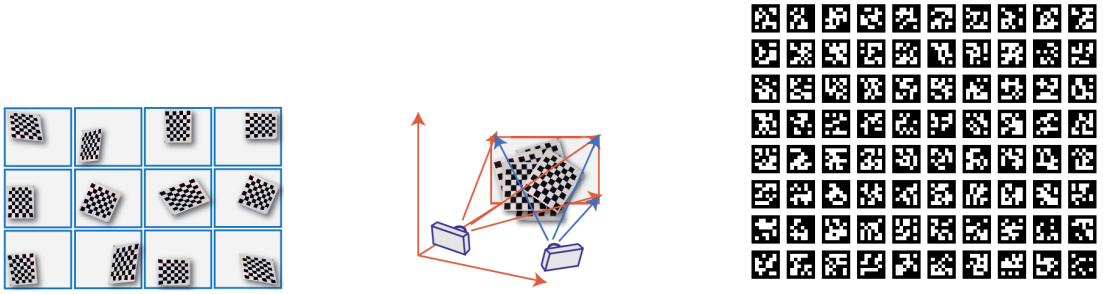
(c) Bad structure of pattern size 7x7.

Figure 2.4: Pattern examples.

The traditional calibration requires points detected from many different planes. The two-dimensional pattern plane, so it should be well distributed in each part of the image during the whole calibration sequence. It depends on calibration method but it is generally recommended to use 10 or 20 different images of a calibration pattern. For best results, the control board should be at an angle of less than 45 degrees to the camera plane and at different image depths. Fig 2.5a shows all these tips about the pattern's position and distribution. During calibration one element, camera or pattern position remains stationary, cannot move. If the camera is portable, the 2-dimensional pattern must be in a static position. If the camera is not moving, we must put the calibration structure well distributed in whole camera's view with various rotation, scale and skews. This is very important to achieve a high precision of calibration parameters. In the case of stereo camera calibration, the chessboard must be fully visible in both images of each sensor. Finally, as these methods use points from the image, so the high quality images (which allow for detailed detection of squares) increase the quality of the results. These methods are accurate and widely used in the field of computer vision.

Some traditional 2-dimensional calibration methods use landmarks as standard calibration patterns. Fig 2.5b presents one of the most popular landmarks - the aprilgrid patterns [130]. The Kalibr application [112] is a tool where the offline method recognizes these tags and calculate very precise camera parameters.

Other works also use landmarks system. Tang et al. [165] presents a work where the multi sensor system uses the landmarks to calibrate. He calibrates the cameras with laser, while the method proposed by G.Antonelli et al. [11] estimates simultaneously camera parameters and odometry param-



(a) Pattern distribution recommended. (b) April Tags

Figure 2.5: Pattern examples - landmarks that represent 2-dimensional calibration pattern.

eters.

1-dimensional structure

Another subgroup of traditional camera calibration, that has the least elaborate representation, is the one that require a 1-dimensional calibration pattern. This type of method requires a special calibration device and movement around the vicinity point. For this reason, it is far less practical than the methods based on standard 2 or 3 dimensional patterns. This method usually needs more measurements and still seems to be less precise. For example, Miyagawa et al. [118] and Zhang et al. [190] proposed approach, which use the distance measurements of a fixed-length object (stick), where at least two points on the objects are tracked and known. This method estimates the camera intrinsic parameters, while it observes a moving line around a fixed point. However, that method does not calculate the stereo extrinsic parameters.

Online approaches of traditional methods

The standard traditional camera calibration methods should be treated as offline procedure. They must be performed before the application or computer vision process. They require the use of a special calibration tool, 3, 2 or 1 dimensional, depending on the method. It cannot be guaranteed that a special apparatus, chessboard or other calibration pattern is always ready to use when needed. However, it is worth paying attention to the approach that attempts to adapt the working area. Some approaches realizes the mission in the same, custom place, then working conditions can be adjusted and patterns always visible. In special cases, when the place of mission is significantly limited, it is possible. Some methods do not adapt the environment but try to extract special elements of scene. Then some of particular objects can be used as calibration patterns, for examples: traffic sign or crosswalks. There are some approaches that mount calibration tool with camera sensor.

Qin et al. [135] equip the device with the attached calibration pattern. In such case, the pattern is always available and ready to use during the whole mission at any time. This approach proposes to

build a stereo vision system that has half-mirror with light source displaying the pattern in the real cameras scene. This sensor is mounted on top of the devices and it is presented in Fig 2.6.

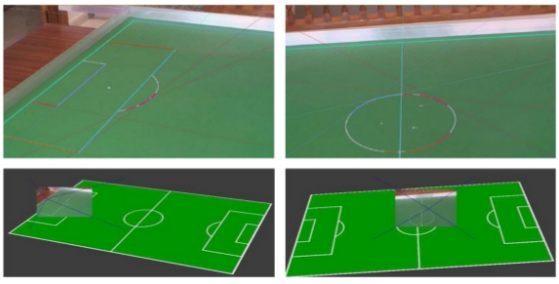
Figure 2.6: Embedded calibration stereovision system (left) and embedded stereo pair (right).

[85] and [153] propose a similar standard traditional calibration method which uses 2-dimensional pattern always visible by the camera. Calibration pattern is available at the robot's hands, it allows to be carried with its and use calibration whenever required. This type of approach can be performed online.

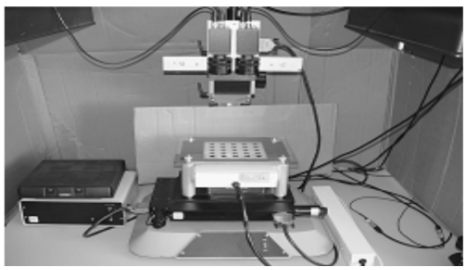
The same methodology is proposed in the another approach. Wang et al. [173] present the intrinsic and extrinsic camera calibration method. This procedure is based on traditional pattern procedure but they propose to replace the calibration tool by invisible infrared ray. It is displayed in front of camera all the time in each environment.

Many methods in this group consider only intrinsic parameters due to special video/television applications. A camera calibration method for stationary cameras designed to work in special environment is presented in: [85],[153],[41], [32] and [10]. Aleman et. al.[10] proposed to calibrate the camera for sports event scenarios. They took advantage of the lines on the pitches, their size, coordinates and position which are constant. This information allows to treat a pitch infrastructure as one large calibration pattern. The method based on the extraction of primitives corners of the image in this difficult terrain is able to cope with the problems of shaded regions and lens distortion.

The automotive industry looks at the stereo camera sensor very promising. Therefore a large part of the methods are dedicated to operate in that context. It is possible to detect and use many elements



(a) Scale model, showing line detection and reconstruction.



(b) The set up for stereo vision.

Figure 2.7: Examples of traditional online camera calibration methods.

of the road-infrastructure, which are standardized and occur very often or always, such as: horizontal road lanes or vertical traffic signs. In addition, vehicles have many sensors which provide additional information that can be used to calibration or scale extraction. Many of methods look for road markers as calibration patterns, [76], [63]. Martita et al. [107] propose to use road lines, where Zhaoxue [191] a crosswalks in order to calibrate camera.

The stereo cameras could be calibrated online base on the traditional method. However, this kind of approach is not universal and not perfect due to fact that the calibration pattern is always required.

It is worth paying attention to the fact that none of the above mentioned works pays attention to the important aspect of the possibility to be realized online. For each run not only specific pattern must be available, which is a major concern for most of the works, but also the process cannot be complicated to be done in the real time. Most of the presented processes do not describe the complexity of its algorithm. Additionally, none of the above mentioned work characterizes the method in terms of the time needed to obtain a parameters and the hardware on which it is performed. If any calibration method can be performed on PC or in cloud in real time, it is not synonymous that it can be realized on embedded systems which can be several times or even several hundred of times weaker in terms of computing power.

2.3 Self-calibration method

The second widely analyzed group of camera calibration is the self-calibration or auto-calibration methods. The procedures do not make usage of any particular calibration object, the researchers try to get rid of all calibration patterns in the contrast to the traditional group. Those methods can be considered as 0 dimension approach because typically only the set of corresponding POI across a several camera's views are required. Thanks to it, they can compute the parameters and poses of camera in any stable unknown scene. Methods do not require the user interaction. However, the motion of camera in the scene is required and obligatory. Maybank and Faugeras [111] described the theory of camera motion during the self-calibration procedure, then Sturm [159] characterizes and categorizes specific movements in that context.

Fig 2.8 presents a large variety of methods that can compute the camera parameters with computationally flexible approach and different constraints. Since they are based on the standard POI, it is possible to use those methods online, while performing other tasks. The most popular strategies such as bundle adjustment (BA) optimization or epipolar constraint are detailed explain in the following section.

There are some of self-calibration methods in the literature that try to calculate only intrinsic camera parameters, for example: those which are based on Kruppa equations [111], [80], [28] [105]. It is not the subject of this work, so those procedures are omitted.

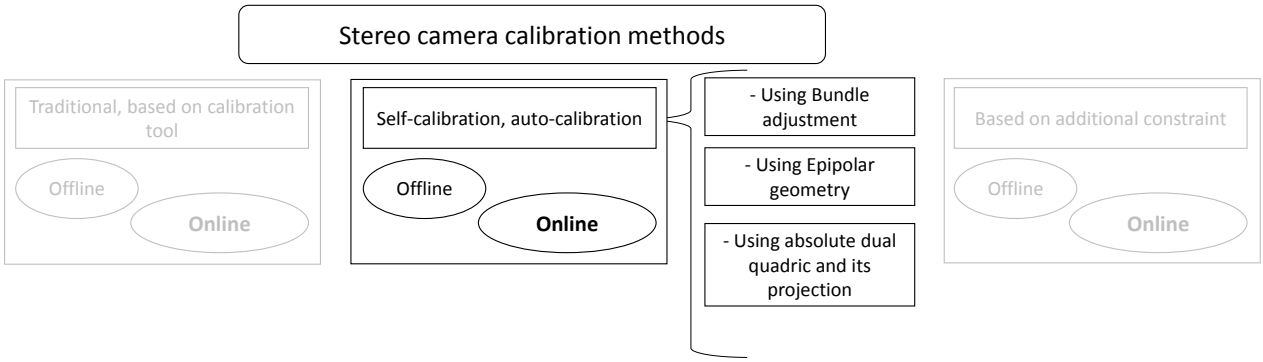


Figure 2.8: Self-calibration methods characterization.

2.3.1 Bundle adjustment camera calibration methods

The camera calibration can be achieved by optimization of POI position from different camera poses. The bundle adjustment (BA) method is the one of procedure which can realize it. This technique is adapted in the field of computer vision, where it calculates and optimizes the positions of multiple 3-dimensional POI from different view of the observer. Globally, the BA refers to a visual reconstruction where it creates the optimal 3-dimensional structure of the scene geometry. In addition, it can estimate

the relative motion and vision parameters (camera position and/or calibration)[169].

For the multidimensional optimization problem, the BA finds a set of camera parameters by minimizing the projection error between the measurements (all POI in each frame) and the predicted 3-dimensional position of observed points. The main difficulty is to find optimal parameters by minimizing cost of functions due to the scale problem. More precise camera parameters allow to minimize the error of POI positions and estimate the position of future frames and compare it with future measurements.

The 3-dimensional map of the world scenery reconstruction is created from the merge between the all of POI. As a result, the standard BA is expressed as the sum of the squares of a large number of non-linear functions that must be solved by appropriate algorithms. Due to the large size of problem, the several modifications and improvements have been developed on the BA context, such as: [90], [69] and [154].

The Levenberg-Marquardt algorithm has been proved to one of the most successful algorithm in computer vision to solve the BA due to its simplicity and availability in literature. It is an iterative algorithm that localize the minimum of a multidimensional function. It solve sum of squares of non-linear real value functions [96] [172]. It has become a standard technique commonly used in a wide range of disciplines where non-linear problems with the smallest squares has to be solve. There are some free C++ implementations available such as: [103] or [106]. There is a brief description of instruction how to implement this algorithm [104].

Offline camera calibration methods based on bundle adjustment

There are some calibration tools, which are based on solving BA by Levenberg-Marquardt algorithm. The MicMac [142] is a interesting, free, open-source solution for photogrammetry software for 3-dimensional reconstruction. It is a offline self-calibration method, that provides a very precise camera parameters. This powerful procedure does not require any characteristic scene on the image, but a several images of the same point from a few views. However, a high precision is burdened by computation amount which relates to a dozen minutes sometimes even hours on Intel core I7 to compute all parameters.

The main goal of Carrera et al. [25], [26] is to calibrate the relative transformation between multiple cameras on a robot platform up to scale. His approach calibrates the cameras with non-overlapping field of views. From this reason, the method requires precalibration movement (full rotation 360 degree) of the whole system. This ensures that cameras see the same part of environment. Carrera creates a globally consistent POI map for each camera. After, each feature is matched between each correspondences from pair of map via threshold matching between SURF descriptor. These steps can be consider as a local BA, based on 3-dimensional similarity transform supported by RANSAC [69] to

find inlier feature correspondences. The global BA optimizes camera, robot poses and 3-dimensional POI position. Moreover, this approach requires two optimization steps. First one it a local level between images view and second is a global level between different frames. It is very computation costly thus it is not considered as an online method. This approach can be precise in some scenarios like close and small environments but can fail in outdoor scenarios, when the majority of natural features are located far away from cameras.

L. Heng [74] proposes self-calibration method, which is very similar to Carrera work. His method does not need the overlapping view. He extends and upgrades the approach that it overcomes outdoor environment difficulties. He maximizes the number of POI correspondences between the images, after thanks to BA the most recent images are rectified to the common image plane.

Both described works [25] and [74] do not need a prior map. However, in the second [73] Heng's paper, he presents a new update of his work. The higher accuracy thanks to environment's infrastructure is achieved. It is possible due to a prior map which removes the need to find inter-camera POI correspondences and loop closures. It significantly accelerates the calibration procedure. His new method gets rid of global BA. It results in a simpler, more robust and faster algorithm, compared to his previous work. Unfortunately, this method is still not online thus to first local BA optimization.

T. Dang works on similar problem and proposes another method for estimating the relative transformation between multiple camera images to the external coordinate system - vehicle. This method compute the extrinsic parameters representing the whole camera system to global frame, not determine the extrinsic stereo parameters between two cameras [37],[38],[36]. [37] is based on geometric error criteria. It relies on a consistent derivation of a robust, recursive optimization scheme for Gauss–Helmert models. The algorithm allows to combine different geometric constraints in a common framework where implicit Iterated Extended Kalman Filter (IEKF) is used. The three main constraints work together: epipolar for stereo images, trilinear for image triplets and collinearity in the BA, that create a large number of computation.

Next similar paper is Pagel's work [133]. He targets non overlapping field of view for cameras on a mobile platform and calibration without using any pattern or known scene structure. The motion scale and extrinsic camera parameters are estimated due to BA. However, similar to Dang paper, it does not prove that stereo extrinsic parameters between two cameras can be established.

There are other approaches that solve calibration based on BA like [168] or [30]. Civera et. al. present a method that can calculate all parameters and camera pose for monocular camera. Tresadern's work does not determine all degrees of freedom of camera which represent extrinsic parameters.

Online camera calibration methods based on bundle adjustment

There are some approaches in the literature, that try to solve BA online. Hansen et al. [67] propose continuous online extrinsic re-calibration. Method obtains 5 degrees of freedom which represent the extrinsic pose of stereo frame. They estimate the whole camera's setup position, not each of camera separately. They assume that both cameras move with the same translation. Procedure is able to perform calculation in real time using only sparse stereo correspondences. It minimize the stereo epipolar errors by Kalman Filter (KF) [177]. The current extrinsic position estimated from each stereo pair is enable to remove a temporal drift. He says that enough correspondences (around 1000) is sufficient to realize a good calibration. This method is sufficient to compensate an odometry drift, and support navigation purpose.

Warren [176] work tackles the similar problem, the whole camera's setup position in the world frame reference. The method is based on modified BA algorithm that take advantages of rigidly-linked pair of cameras with overlapping views. Cameras do not have possibility to move their R and T between each other. He shows that, it is possible to recover an accurate setup position online from real world data by explicit by BA. Warren shows the ability to compute camera parameters with high precision online but does not provide a specification for hardware where algorithm is executed. Method seems to be a high computation load so it is not a good candidate for embedded processors.

Sappa et al. [146] [145] present an efficient technique for estimating the pose of an onboard stereo vision system relative to the environment's dominant surface area (it is supposed to be the road surface). This method can be used in vision-based ADAS. The procedure basically consists of fitting a plane to 3-D points belonging to the road and then determining the camera pose with respect to that plane. The road region is always in front of the vehicle (up to 50 m away). Then, it is approximated along frames as a piece wise linear curve, since the plane parameters are continuously computed and updated. Road data points are identified by assuming that the road surface is the most predominant geometry in the scene, which holds in most situations, but it cannot be guaranteed.

One of the newest and very interesting calibration work is method proposed by E. Rehder et al.[46]. They implement BA to recalculate extrinsic parameters in real time. This work shows good update of parameters on the fly and proves results on open-source dataset. Several accelerations step which optimize BA process are proposed in order to computes each camera parameter in real time. The method does not need any plane or other calibration tool. However, their experimental setup is realized on Intel 7 which can not be consider as a embedded processor for small CPS due to high electricity consumption around 90 W.

2.3.2 Epipolar geometry

Another most popular method from self-calibration group is based on epipolar geometry. The mathematical theory of epipolar constraint is explained in section 6.3.

It can be represented in the fundamental matrix (F). There are some algorithms that can calculate F or E, for example: eight point algorithm (8PA) [102], [68], [29],[14], [15], [94], five point algorithm [97],[128] or point to point algorithm. All of them have many improvements and have been implemented in many open-source library, for example: openGV C++ [88]. The obtained matrix has to be converted into essential matrix (E). Then, thank to the singular value property by singular value decomposition (SVD) can be transformed to the extrinsic parameters (R and T).

There are many papers that try use these algorithms. For example: Yan [184] work calculates the extrinsic parameters in binocular stereo vision of moving robot. The intrinsic parameters are assumed to be know. Based on matching stereo points, the F and E are calculated. This work provides neither execution time nor hardware used to do computation. Other work like [18] tries to overcome a limitation in image resolution and field of view. They propose continuously external camera calibration. Then, the linear estimation of E is used to convert it to relative pose, followed by a non-linear refinement incorporating depth ordering constraint in real time. Experimental testing was done on 195MHz MIPS R10K processor into small indoor sequences with stationary camera.

The fundamental matrix F

Many work based on the F, which is an algebraic representation of epipolar geometry. Given a pair of images, where to each point $x = (x, y, 1)^T$ in one image, there exists a corresponding epipolar line l' in the other image. Any point $x' = (x', y', 1)^T$ in the second image matching the point x must lie on this epipolar line l'. The epipolar line is the projection in the second image of the ray from the point x through the camera centre C of the first camera. Thus, there is a map $x \mapsto l'$ from a point in one image to its corresponding epipolar line in the other image. It is the nature of this singular map, which is a projective points to lines and represented by a F matrix. Its properties are described in section 6.3.1.

The F satisfies the condition that for any pair of corresponding points $x \leftrightarrow x$ in the two images:

$$x'^T F x = 0 \tag{2.1}$$

Let f be the vector representation (row-major) of F then each correspondences satisfies $p_r^T F p_l = 0$

$$\begin{bmatrix} x_i' & y_i' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{31} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = 0$$

that becomes

$$x_i x'_i f_{11} + x_i y'_i f_{21} + x_i f_{31} + y_i x'_i f_{12} + y_i y'_i f_{22} + y_i f_{32} + x'_i f_{13} + y'_i f_{23} + f_{33} = 0$$

$(x'x, x'y, x', y'x, y'y, y', x, y, 1)^T f = 0.$
f is a 9-vector and looks

$$f = \begin{bmatrix} f_{11} & f_{21} & f_{31} & f_{12} & f_{22} & f_{32} & f_{13} & f_{23} & f_{33} \end{bmatrix}^T$$

It is set up a homogeneous linear system with 9 unknowns variables.

The equation 2.1 is true because if points x and x' correspond, then x' lies on the epipolar line $l' = Fx$ corresponding to the point x. In other words $0 = x'^T l' = x'^T F x$. Conversely, if image points satisfy the relation $x'^T F x = 0$ then the rays defined by these points are coplanar. This is a necessary condition for points to correspond.

$$Af = \begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ : & : & : & : & : & : & : & : & : \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} f = 0$$

If there is more than 8 point matches, there is need to select the best points, it can be realized by RANSAC Robust estimation model.

Robust estimation Robust statistical methods have been established for many common problems, such as estimating pose, scale etc to not unduly affected by outliers and other external factors. Another motivation is to provide good performance methods in the case of small deviations from parametric distributions. In case where many points can create a different model, an very important step is to select the best one. In order to realize it the robust statistical method are used in epipolar geometry constraints. parametric distributions.

The set of correspondences $x_i / leftrightarrow x'_i$ are presented. In many practical situations the source of error arrives from many things, such as: the measurement of the point's position, mismatched etc. These points are outliers to the Gaussian error distribution. These outliers can severely disturb the estimated homography, and consequently should be identified. The goal then is to determine a set of inliers from the presented correspondences. The homography can be estimated in an optimal manner from these inliers using the algorithms described in the previous sections.

The Random sample consensus RANSAC is an one of example of robust iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers. It is recommended tool to use with the eight point algorithm.

Objective is to fit the best model to a data set S which contains outliers.

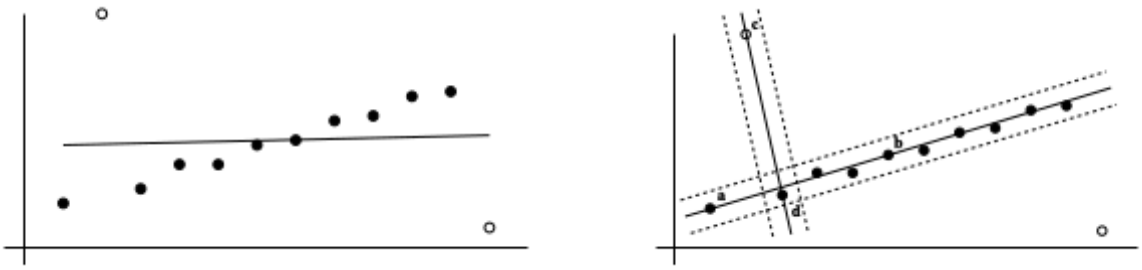


Figure 2.9: Robust line estimation.

- Randomly select a sample of s data points from S and instantiate the model from this subset.
- Determine the set of data points S_i which are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S
- If the size of S_i (the number of inliers) is greater than some threshold T , re-estimate the model using all the points in S_i and terminate.
- If the size of S_i is less than T , select a new subset and repeat the above.
- After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i

In the literature, RANSAC problem and optimization has been considered many times[119], [166], [161].

The essential matrix E

The results of the algorithm depends on the coordinates frame where points are expressed. The E is the special form of the F, which has a fewer degrees of freedom and additional properties, compared to the F. In the literature, it can be refereed while input points are normalized image coordinates by simple intrinsic parameters. Data normalization improves the accuracy of results. However, in the state of the art different normalization exists:

- intrinsic parameters
- isotropic
- bearing scaling

Intrinsic parameters normalization , consider a camera matrix decomposed as $P = K[R|t]$, and let $x = PX$ be a point in the image. If the intirnsic parameters of camera K are known, then its inverse to the point x to obtain the point $\hat{x} = K^{-1}x$ must be applied. Then $\hat{x} = [R||t]X$, where $\hat{x}$ is the image point expressed in normalized coordinates. It may be thought of as the image of the point X with respect to a camera $[R|t]$ having the identity matrix I as calibration matrix. The $K^{-1}P = [R|t]$

is called a normalized camera matrix, the effect of the known calibration matrix having been removed. The defining equation for the E is $\hat{x}'^T E\hat{x} = 0$. Replacement for $\hat{x}$ and $\hat{x}'$ gives $x'^T K'^{-T} E K^{-1} x = 0$. Comparing this with the relation $x'^T F x = 0$ for the F, it matrices is

$$E = K'^T F K \tag{2.2}$$

Isotropic scaling. As a first step of normalization, the coordinates in each image are translated (by a different T for each image) so it is required to bring the centroid of the set of all points to the origin. The coordinates are also scaled so that on the average a point x is of the form $x = (x, y, w)^T$, with each of x, y and w having the same average magnitude. Rather than choose different scale factors for each coordinate direction, anisotropic scaling factor is chosen so that the x and y-coordinates of a point are scaled equally.

The average distance of a point x from the origin is equal to $\sqrt{2}$. This means that the average point is equal to $(1, 1, 1)^T$. In summary the transformation is as follows:

- The points are translated so that their centroid is at the origin.
- The points are then scaled so that the average distance from the origin is equal to $\sqrt{2}$.
- This transformation is applied to each of the two images independently.

Bearing vector according to the openGV [88], the bearing vector is defined to be a 3-vector with unit norm bearing at a spatial 3D point from a camera reference frame. It has 2 degrees of freedom, which are the azimuth and elevation in the camera reference frame. Because it has only two degrees of freedom, it is frequently refereed to it as a 2D information. It is normally expressed in a camera reference frame. So points must be multiplied by intrinsic parameters, as it is shown in the E computation. Then the radial and tangent distortion must be removed, before the set of equation composition. Finally, the point is expressed in camera frame, it is normalized by sum of all elements, then it represents the bearing point.

According to E matrix properties described in section 6.3.2, the best E estimated in iterative eight point algorithm on the base of bearing vectors has to be converted to R and T. Its rank of E is greater than 8 then the least-squares solution can be found by use the singular value decomposition (SVD) [13]. This method is implemented in many programming tools or libraries such as: Matlab.

Determining the extrinsic parameters is realized in the following steps due to the least squares solution for E:

$$E = U \sum V^T$$

where U and V are orthogonal 3x3 matrices and $\sum$ is a 3x3 diagonal matrix with $\sum = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix}$ The diagonal entries of $\sum$ are the singular values of E which, according to the internal constraints of the E, must consist of two identical and one zero value. Define $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $W^{-1} = W^T = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and make the following model:

$$[t]_x = UW\sum U^T$$

and

$$R = UW^{-1}V^T$$

This kind of solution provides four possible choices for the second camera matrix, two different R matrices and T which have the same value, but opposite signs. It has to be determined which one is correct. It is usually realized by the reconstruction of point X in front of both cameras in one of these four solutions only (see section 6.3.2).

2.4 Method based on additional constraint

The two most popular camera calibration groups have been presented and discussed. There are many other methods that rely on other constraints. Some of them use another sensor, special environment structures, vanishing lines or compose from a several methods. Many constraints in the custom CPS with embedded contexts, eliminate the possibility of use those methods, because they require additional aspect of devices, environment or need many computation. However, some of them should be characterized and presented, in order to know why they can not be used.

2.4.1 Method which uses data from another sensors

In the CPS, vehicles and robots usually many senors supports camera. Some of methods try to use this approach and propose calibration procedure with the data from other sensor such as: LIDAR, IMU or GPS. These approaches have huge representation. However, they are dedicated to specific system with certain conditions.

There is many stereo camera calibration methods dedicated for the vehicles applications. For example, the modern car has many additional senors. There is a calibration method [84] which use a

GPS and IMU. Xu et al. [182] propose an hybrid procedure between traditional and self-calibration method which use the crosswalk corners connected with its real world positions. These detected points are localized and positioned by the GPS that knows their position (cross-walks) with great precision. Method requires the well detected infrastructure and GPS signal.

The modern cameras are mounted very often with IMU sensor. There are some methods [51],[117] and [98] that try to inject into calibration procedure data provided by this sensor. It can be an interesting approach for mobile robots and aerial vehicles, where IMU is one of the base sensor. The information can help estimate a 3-dimensional motion and be compared with visual odometry. Normally this approach is used in order to find the position of the whole agent in the monocular camera setup. They are more interested in global camera pose, which is not enough precise for stereo cameras position. Unfortunately, this approach usually suffer on precision and accumulated error which has to be corrected time to time.

Fleps's [51] proposes real-time capable and deliver very noisy data, but can be dedicated for drones and different flying vehicles which are already equipped with working IMU systems.

Tan et al. [163] and [164] publish automatic extrinsic calibration method for automotive domain in general drive conditions. Those approaches require the visible road surface to work properly. The method is based on the synchronization of the video stream with the position of the vehicle, which is estimated on the basis of IMU.

The continuous extrinsic online calibration for stereo cameras proposed by Mueller [122] estimate the relative 6 degree of freedom between the two cameras sensor. This algorithm runs in real time on Dual Intel Xeon e5-2667 and requires a high precision of IMU to find the global position of system according to the vehicle motion.

For many mobile robot and vehicles, the odometry is a base procedure for motion and navigation strategy. This data can be used to analyze the extrinsic camera calibration problem. All [116], [27] and [65] methods provide a global parameters in the reference to global frame, but not parameters of each (left and right) camera into stereo sets. Moreover, these procedures can be used only with cameras where odometry is available.

2.4.2 Pure rotation and translation

There are some methods that try to rely on pure rotation or translation. It is not universal type of method. However, this kind of methods are perfect candidates for camera mounted on manipulators or other robots with electrics engines. If motion can be perfectly controlled, it can be used for camera calibration [183], [174].

Moutinho et al. [121] proposes an online stereo calibration method, which rely on information from the cameras and the motor encoders. Procedure is dedicated to humanoid robot. It controls and can use an engine's rotation to calibrate camera.

2.4.3 Minimization of matching cost

The methods which are based on the minimization of matching cost (non-linear optimization) are very complex in term of computation and does not fit for embedded constraints. Procedures present a camera poses optimization while large number of correspondences points can be detected only from one stereo frame. There is no need for any calibration pattern, motion or additional data. Usually, the initial guess is required in order to compute the scale factor.

Spangenberg et al. [157] shows that number of matched pixels can be used as a valuable source of information to improve relative stereo calibration. The nonlinear problem is solved by Monte-Carlo algorithm.

Kuhn et al. [92] use a PatchMatch stereo with simultaneous Total Variation to achieve a reliable and accurate parameters re-calibration. This method relies on disparity map extraction and potentially detect inaccurate calibration parameters.

Ling [152] proposes online stereo extrinsic method that is adequate for block matching-based dense disparity computation in the whole processing pipeline. They execute the method on a Lenovo Y510 laptop with i7-4720HQ CPU which has significant higher power than standard embedded processors.

2.4.4 Vanishing points and lines

In contrast of using calibration patterns, it is possible to formulate another scene constraints for camera calibration. The perpendicularly vanishing lines are one of the example of such restriction, proposed and used by Caprile and Torre [24], [100]. However, these methods are not available when specific geometric structures are not guaranteed.

Grammatikopoulos et al. [61] present an approach for the automatic estimation of interior orientation from images with three vanishing points of orthogonal directions. The Tan [163] and [164] works was already characterized in other group but they also contains a vanishing point constraints.

These works [89], [178], [125] estimate the extrinsic camera parameters online using ground plane. This approach is developed only for monocular camera pose. It can be tested if those methods can be realized separate for two cameras and then merge it in the one global sensor.

Nedevschi's approach requires a special designed scene with road marking point. Moreover, it needs a flat ground, ruler, and something that can be used as the straight lines. The interactive operation is used to verify the locations of the calibration lines, which makes the method robust to different environments.

There are some calibration methods, which are dedicated for special, not common scenarios. The Nelson's [126] focuses on night situation when there is not enough light to detect and recognize any features in the scene. They adopt this method to localize a night artificial light source. This scenarios is extreme for specific mission and can not provide precise extrinsic stereo parameters.

2.5 Dataset

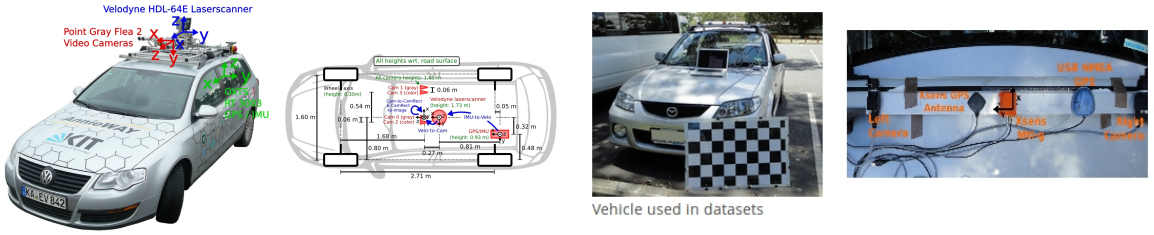
The dataset is a collection of statistical data normally presented in special, adapted form. The datasets for image processing and computer vision refers as the vision benchmark. There are many of them in the literature, because wide spectrum of applications in the computer vision domain requires many different data from various environments, scenarios and sensors, etc.

One of the most popular vision benchmarks for computer vision processing is one proposed by the Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago (KITTI) [59]. It provides many data recorded from different sensors such as stereo camera, GPS, IMU and LIDAR installed in the autonomous driving platform. Fig 2.10a shows the KITTI's setup, which has a wide range of scenarios recorded in different places, such as city, road, campus, etc. The stereo optical flow with other data gives possibility to verify many applications, such as visual odometry, 3-dimensional object detection, 3-dimensional tracking etc.

The University of Queensland's St Lucia proposes another vision benchmark dedicated for computer vision application [60]. Fig 2.10b presents their sensors setup to record high spectrum of data, similar to KITTI.

Swiss Federal Institute of Technology Zurich creates the EuRoCMAV - dataset recorded by drone [23]. It contains stereo images, synchronized IMU measurements, accurate motion and structure ground-truth.

Dosovitskiy et al. present a relatively new and very interesting work about CARLA [44]. It is the simulator with open source code to test, train and validate the autonomous urban driving systems. Wide range parameters in the simulation allow testing many aspects such as: weather conditions, unpredictable behavior of other cars, motorcycles and bicycles in traffic, etc. This is an extremely interesting project that can be used to test a various number of computer vision applications. Fig 2.11c presents a screen of work from CARLA simulation.



(a) KITTI sensor setup .

(b) STLUCIA sensors setup.

Figure 2.10: Sensors setup.

(a) KITTI view . (b) STLUCIA view. (c) CARLA view. (d) EuRoCMAV view.

Figure 2.11: Views form different datasets.

2.6 Summary of calibration methods

Based on this chapter, we sustain and continue the hypothesis that the ideal, universal stereo camera calibration method in the real time does not exist. The specific application or system adapts the calibration process that usually requires and depends on many factors. The calibration strongly relies upon in the input data form, environment, required precision, etc. The specific procedures usually create many limitations and constraints. The left part of Fig 2.12 shows that the computer vision community use a powerful PC and calibrate cameras considering it, as only first phase of more complex tasks. In this chapter, we present some of calibration methods for specific procedures, for special targets or environments. We categorized some methods taking into account different constraints such as if method can be executed everywhere, how much computation it needs. We verified if method satisfy a glasses context and provide a sufficient precision. We remind that the main motivation is to estimate the extrinsic parameters in the smart glasses context, with consideration that intrinsic camera parameters are constant. They can be stable due to the internal camera construction of the static focal length. However, in this thesis, we consider the glasses do not guarantee the stability of the extrinsic parameters, which describe the mutual position and orientation between two cameras in the three-dimensional space, because the cameras are subject to a number of different strengths and unforeseen conditions. However, we look on this problem in different way compare to current state of the art. In the right part of Fig 2.12 we presents how the existing methods see a calibration problem, as completely separate task from application. In our work, we would like to see a calibration process in the main application. The next chapter presents this point of view.

TYPE OF ALGORITHM	CONSTRAINTS			
	Executed any where	Low computation	Glasses context	Precision
Traditional pattern base methods (Zhang)	-----	X/-----	-----	X
Epipolar constraints (Hartley 8PA)	X	X	X	X/-----
Bundle adjustment (Rehder)	X	-----	X	X
Additional sensors such IMU/LIDAR (Zhou)	X	X/-----	-----	X/-----
Known object as pattern (Xu)	-----	X/-----	-----	X/-----
Known rotation or translation	-----	X/-----	-----	X/-----

SELF CALIBRATION METHOD BASED ON **NOT COMPLICATED ALGORITHM**

EIGHT POINT ALGORITHM (8PA)

Defense of the PhD thesis | Michał SZCZEPAŃSKI | 14.11.2019

Figure 2.12: Summary of main aspects after second chapter.

The first, presented group is the most numerous and concerns the traditional calibration methods. The advantage over other methods is that it simultaneously provides very accurate and stable intrinsic and extrinsic parameters. For this reason, in this work we use a traditional method as the reference method. Those methods implies the use of some kind of known calibration patterns in the camera's view. It is a huge constraint when dealing with real time applications, because it is impossible to ensure that calibration tool is available. Moreover, it is hard to automatize this kind of procedure, so the current mission usually must stop in order to run the traditional method. These methods perform offline before the first time camera use in computer-vision tasks. There are some of the traditional methods that try to simulate the calibration tool and use some standardize objects from the environment. However, it is impossible to ensure that the specific elements of the scene are always available. Moreover, it is hard to guarantee a good distribution of calibration objects in the scene, thus the obtained parameters are not always accurate.

The most interesting group in our context of work is the self-calibration methods. They get rid of the calibration tool and they require a moving camera in a stable environment. These procedures can work in an online context but they still need to respect some of additional constraints. They require a good feature detection and matching. We distinguish two main leading group in the self-calibration methods. The first group relies on epipolar geometry, which many algorithms can solve. Some of them are low complex and can compute the essential matrix that converts to extrinsic parameters. After the analyses of the state of the art, we would like to use the Hartley universal eight point. This algorithm seems to be efficient and constraint of low processing power because it does not require huge amount of calculations. Method requires small number of simple stereo matched point of interest, which exists in many computer vision's application pipelines. However, algorithm has some drawbacks described in literature as very sensitive and requires a scale factor from another source.

The second group relies on bundle adjustment, which is an optimization of the 3-dimensional point cloud from several 2-dimensional views. This estimation requires many computations and remains time consuming, especially in the larger environment, this is a big limitation of this method. The higher number of input POI in the process significantly increase the number of computations. Rehder et al. realizes implementation on powerful hardware - PC with Intel i7 CPU. It does not allow assuming that the same method can operate with the limitations of embedded systems. We consider the bundle adjustment as method too complex and demanding in terms of calculations. A subgroup of self-calibration methods seems to be the only choice, thus fulfills the assumptions presented in the Fig2.13. The methods, which use a known rotation or translation, do not fit into glasses context.

Finally, we must test selected Eight-point algorithm in the special dataset. In fact, some work that targets the same problem does not provide any dataset where such changes of camera's position happens. For this reason, we must create a reasonable dataset. According to system restrictions, selected method must recalculate extrinsic parameters: online, without special patterns (that can

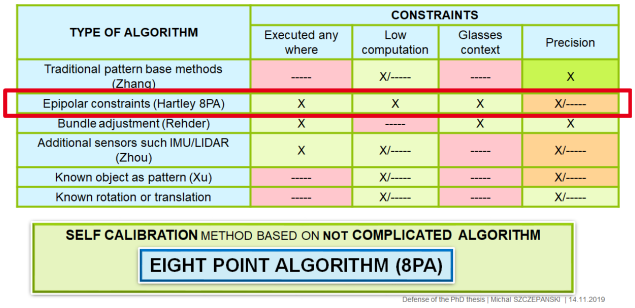


Figure 2.13: Summary of methods analyze in all context of work.

apply everywhere) and realize computation on embedded systems.

Chapter 3

Approach of online calibration pipeline on embedded systems

> Innovation distinguishes between a leader and a follower.
>
> Steve Jobs

This chapter presents the discussion about online calibration in the application pipeline. We create a new concept of camera calibration hidden in to system. It should verify if current extrinsic parameters are up to date. Moreover, this section propose and explain additional functions realized into calibration pipeline.

Objective :
Present the primitive and advance approach of online calibration in the whole application pipeline on an embedded system.
To do this, we :

- study different application pipelines.
- study low level computer vision processing.
- present results from the first primitive online calibration pipeline.
- study different optimization of online calibration pipeline.
- present advanced online calibration pipeline.
- study stereo camera monitoring.

3.1 Introduction

1st chapter describes the background and context of the entire manuscript. It explains the modern, global trends of most CPS. It shows that they require a higher autonomy, reliability and longer working time to realize more complicated mission. Section 1.1 presents a various model of sensors used to deliver data about the local environment to CPS. One of them is the stereo camera, which is becoming increasingly important. It can replace many sensors, providing a high data spectrum. It is relatively cheap, small, and compact; moreover, it does not consume much power compared to other sensors such as LIDAR. On the other hand, the section 1.2 explains the some limitations related to the use of stereo camera. One of them is the possibility to change the extrinsic camera parameters. In order to solve this problem, the system has to be able to recalculate continuously the parameters.

In addition, 1st chapter lists another essential contexts of this work, which must be repeated before explaining the approach of online calibration pipeline: the application and embedded system (sections 1.3 and 1.4).

Section 1.5 presents the targeted final system, which has unique needs and limitations. The smart glasses that require the online stereo camera calibration. They should guide and navigate pedestrians in complex, indoor and outdoor environments without human intervention. The whole system must understand and be capable of self localization, in order to lead a user correctly. In addition, the device must plan and decide, the right path in an efficient and safe mode, taking into account the dynamics unpredictable changes in the real world.

Moreover, it has embedded processors such as ARM Cortex, which is very constrained by memory and processing power. This CPS requires a low-power processor to perform tasks, due to low energy availability (small battery, portability needs). The selected embedded processor is a several times weaker in terms of mathematical operations per second than processors, which are widely used in a standard PC, such as Intel.

The 2nd chapter describes many methods of the camera calibration according to three main contexts of work: CPS, application and embedded systems. It allows distinguishing between different existing methods, which uses various criteria, contains advantages and disadvantages. The emphasis put several times attention on the fact that the best camera calibration procedure does not exist. Moreover, another important point shows that there is no sense in looking for the problem of camera calibration without context of the application and devices.

Section 2.6 describes the selected method, which run in the specific custom context of the smart glasses. The procedure uses only on stereo points and does not require a large number of complex mathematical operations. Therefore, it is suitable for an embedded system, which use only from stereo cameras data. The chosen procedure seems to be a universal and ready to realize in a general environment, without any special patterns or other known objects.

This method is appropriate for any type of application and CPS. The most important information from the 1st and 2nd chapter are presented in summary Table 3.1

Objective	Realize an online calibration pipeline on embedded systems	
Context	CPS, Embedded systems, Application	Camera calibration
Requirements	limited: processing power, memory, battery powered system, no additional sensors, universal at any environment and mission,	simplest method, hidden in pipeline system, without calibration pattern

Table 3.1: Table of summary.

Throughout this chapter, many of the critical questions and fundamental reflections related to the online calibration pipeline on embedded systems appears. We begin with study of the high-level applications, which need such approach with the updated extrinsic parameters. We analyze that the stereo camera calibration is not a key task of any devices and computer vision application. It is an input data, providing process, which feed the pipeline with the stereo camera parameters. The calibration method is not important from the user's point of view, because the procedure is usually not visible to the operator. If camera calibration is not principal purpose of any CPS and application, it cannot consume a lot of computing power and resources from the system.

In practice, in the real scenarios, the extrinsic parameters of a stereo camera can change at any time. Therefore, the system must constantly monitor and have the possibility to recalculate the parameters during the mission. This chapter presents the approach of the online calibration hidden in the application pipeline, which monitor and recalculate the extrinsic camera parameters. In order to know what kind of precision we can estimate from perfect points and real world, we implement and test the primitive online stereo calibration procedure on the standard PC. Thanks to conclusion from first tests, it allows to propose some of the optimization techniques for advanced online calibration pipeline. We elaborate these topics and many other related discussions in detail in this manuscript.

3.2 The whole navigation pipeline of the CPS

We divide the study of the whole navigation pipeline on the custom CPS in three main parts. The first presents some of specific applications used in to navigation pipeline. The second proposes the whole approach divided in special small part of functions. The third presents the low-level computer vision processes commonly used in the many applications and in the targeted smart glasses.

According to section 1.5, the navigation is a main goal of the custom smart glasses. This system requires the online calibration procedure in order to enhance the device's reliability, precision and safety.

Over the last few decades, the area of autonomous mobile systems has developed very successfully. Nowadays, these systems are able to carry out a navigation and many other complex missions on their own in real time [77]. It creates a new opportunities to hide the online calibration in these navigation pipelines.

3.2.1 The purpose and aim of the CPS

The standard navigation mission contains a several separate phase. First, it has to create a reliable map. Then it should localize itself, finally the decision making algorithm can select the path [138]. The custom target navigation system has to realize the same goals and some additional task such as object detection and recognition. The system must detect and avoid obstacles in the local environment in real time. In order to do this, we analyzed the various applications for custom navigation, in terms of input, output and complexity. We present them in the Table 3.2 and Fig 3.1 bellow.

Description	Application
Understanding of the local environment	Disparity and Depth map extraction
Understanding of the local trajectory	Visual odometry
Understanding of the global trajectory	Visual SLAM
Understanding of the local environment	Object Extraction and Tracking

Table 3.2: Table presents the most important application in CPS dedicated for navigation.

In all types of missions under analysis, the calibration of the stereo camera is not directly the main task. The calibration provides only some of the necessary input data (extrinsic camera parameters) required for each of these applications. In the following section, we describe each of these.

Visual Odometry

The visual odometry stands to VO, it is the process of estimating the position and orientation of monocular or stereocular camera's data. It increases navigation accuracy at any type of movement on any surface in robots and vehicles domain. VO calculates the 3D movement of the agent base on different input images from one or more cameras [186] [147] [54]. The relation between the reference

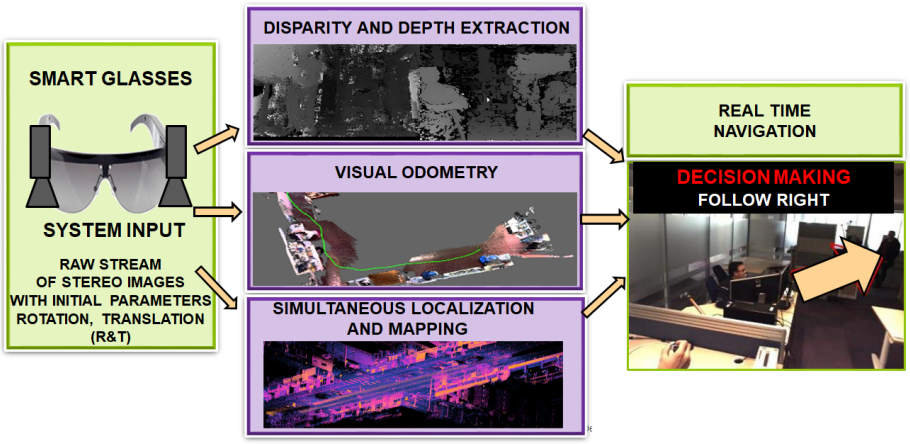


Figure 3.1: The whole navigation pipeline of selected CPS in the embedded domain.

frame's position and the current frame's position represents the continuous problem of optimizing the two 3D positions. Today, many solutions propose to use only small parts of the trajectory. Where each frame with respect to the previous one computers the translation and rotation between them. This kind of assumption allows for reduction a large number of data.

Fig 3.2 illustrates simplified schema and results of the VO. The first step detects and match points between different frames. In the next phase, some of optimization algorithms gradually estimate the path and the poses of the camera/robot. Usually the complex non-linear process calculates a new sensor position in a three-dimensional map. There are a many methods, which enable this type of process, such as pose-graph optimization, bundle adjustment and many more [52] [87].

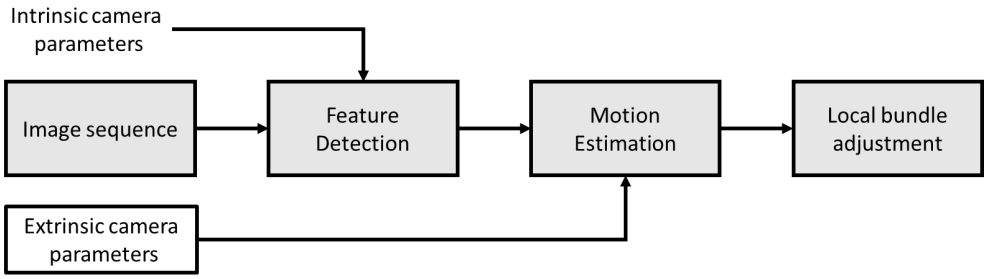
One of the most popular solution is the bundle adjustment (BA) that allows reconstructing the 3D structure and viewing parameter estimation [169]. It is a very complex problem, which tackles many additional constraints. Faessler et al. have significantly reduced costs of bundle adjustment by introducing many restrictions when navigating the unmanned aerial vehicle (drone) in space. Their implementation works in the real time [110]. Their vision-based system runs with a speed of 20 fps on ARM Cortex A9.

Specification	Description
Purpose of specific application	Local consistency of the trajectory
Input data	Stereo parameters, point of interests, features
Output data	Local pose estimation of the camera, position in local reference frame, trajectory

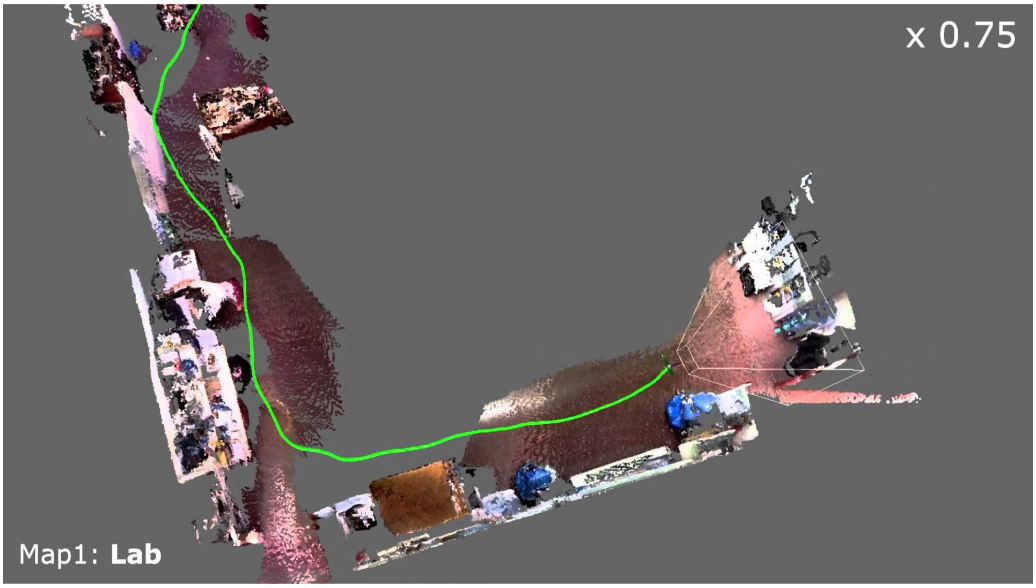
Table 3.3: Table presents the most important information about visual odometry.

Visual SLAM

SLAM stands to Simultaneous Localization And Mapping. This process allows locating itself in known or unknown area and built a map of this local environment at the same time. Many different approaches



(a) Diagram block shows the main components of visual odometry.



(b) Visualization of the output data.

Figure 3.2: Real time visual odometry VO as a high-level application developed in the laboratory.

use specific constraints and sensors in order to realize SLAM [93] [162] [42]. The Visual-SLAM is the one that relies only on visual information so data only provided by cameras. The high spectrum of camera's data can replace use of other sensors like LIDAR, sonar, etc. [115]. Nowadays, when the stereo cameras become cheaper and easier to use, the visual stereo SLAM is an ideal candidate for navigation's applications.

The main differences between visual odometry and VSLAM is the loop closure procedure and default data fuzzy from the other sensors [186]. While the first focuses on local consistency and only images data, the second tries to interpreted information from the mission to a previously reviewed area. This process increases the precision of navigation and localization missions. It reduces a drift in the position and trajectory estimations. However, the global optimization on huge data and loop closure requires a large number of processing that translate to very computationally expensive process. Moreover, it is important to remember, that camera still needs calibration.

Fig 3.3 presents simplified scheme of SLAM. It is one of the most fundamental process in the navigation pipeline. To achieve control and full autonomy of the system, it must have knowledge where it is located and the ability to explore its environment. Everything must realize without user intervention in the real time.

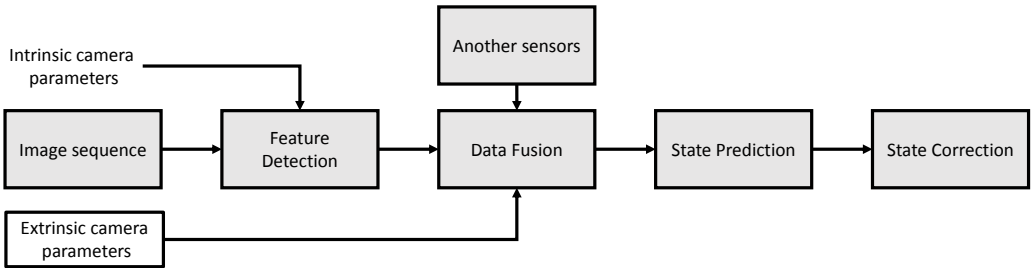


Figure 3.3: A block diagram shows the main components of SLAM.

Specification	Description
Purpose of specific application	Global consistency of the trajectory and map
Input data	Stereo parameters, point of interests, features
Output data	Global pose estimation of the camera, position in initial reference frame, trajectory

Table 3.4: Table presents a most important information about visual odometry.

Disparity and depth extraction

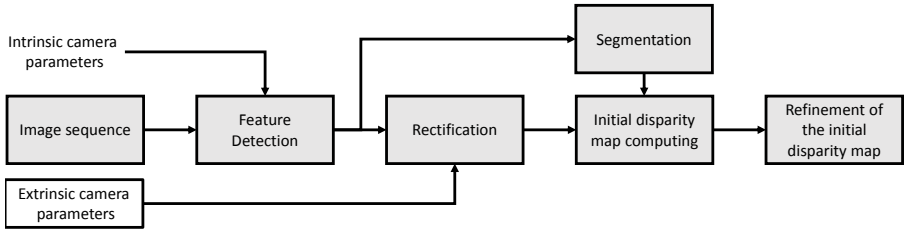
The visual SLAM and VO applications can deliver data for the navigation and localization mission. However, they do not provide any information about local environment such as the distances to the objects observed from camera (system). The depth map obtained by the disparity from the stereo images supplies this data. Data from cameras allows conversing the 2D images into the depth map, other words the 3D information about observed scene. Today, there are neural network procedures, simple algorithm based on geometric information and many different methods to extract depth map [40] [175] [123]. Fig 3.4a shows the scheme of disparity map extraction, which use simple computer geometry computation to get a depth map. Some publications show the real time depth extraction is possible due to hardware accelerators on the GPU or FPGA [140].

Specification	Description
Purpose of specific application	3D information around an agent in local environment
Input data	Stereo images + stereo parameters = rectified image
Output data	Estimating the 3D motion of the camera sequentially

Table 3.5: Table presents the most important information about disparity map.

Rectification is a step required before a depth map extraction while its algorithm uses the epipolar geometry. A transformation process projects input images into a common image plane. The goal of this is to simplify the correspondence problem, which search for matching points between left and right images [39]. If the planes of the image are co-planar, the images are directly in the epipolar geometry. If the centers of the images are in the same line, then the corresponding points from the left image is in the same parallel line in the right image, as shown in Fig 3.5a. It is explained in section 6.3.

Unfortunately, to the nature of the 3D world such situation does not happen in practice. The

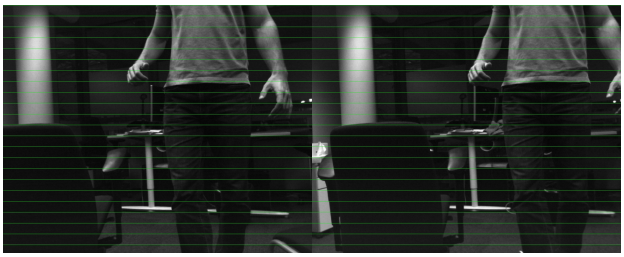


(a) A block diagram shows the main components of depth extraction.

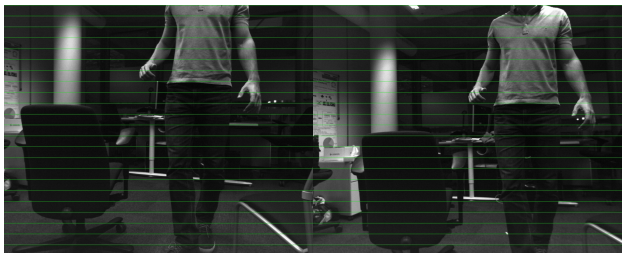
(b) Input image. (c) Depth map.

Figure 3.4: Real time disparity map computation as a high level application developed in the laboratory.

images need rectification (adjustment), in order to be in co-plane positions. Before rectification, the center of the image is not on the same line as shown in Fig 3.5b. The linear transformation can transform images into a common plane if and only if precisely knows extrinsic camera parameters.



(a) Two views in co-planar plane, after rectification process. Green lines are epipolar lines.



(b) Two views in not co-planar plane, before rectification process. Green lines are not epipolar lines.

Figure 3.5: Stereo images in the one plane with green lines on the same high of both images.

3.2.2 The whole navigation pipeline separation

This subsection, together with Fig 3.1 presents the global navigation pipeline for custom CPS, build from the previous mentioned applications. The VO, visual SLAM, depth extraction, object detection and tracking are sufficient to carry out navigation missions on the smart glasses. Data processed by these functions allows choosing a good trajectory with possibilities to avoid obstacle etc. Each of these applications realizes task at the same level of the importance. The analysis of the application presented in Tables 3.3, 3.4 and 3.5 allow drawing an important conclusion. Each table demonstrates that proposed functions use the same input data: the stereo image sequence, intrinsic and extrinsic camera parameters. We can separate and extract the first step of each pipeline from many other computer vision applications [143] as presented in the Fig 3.6.

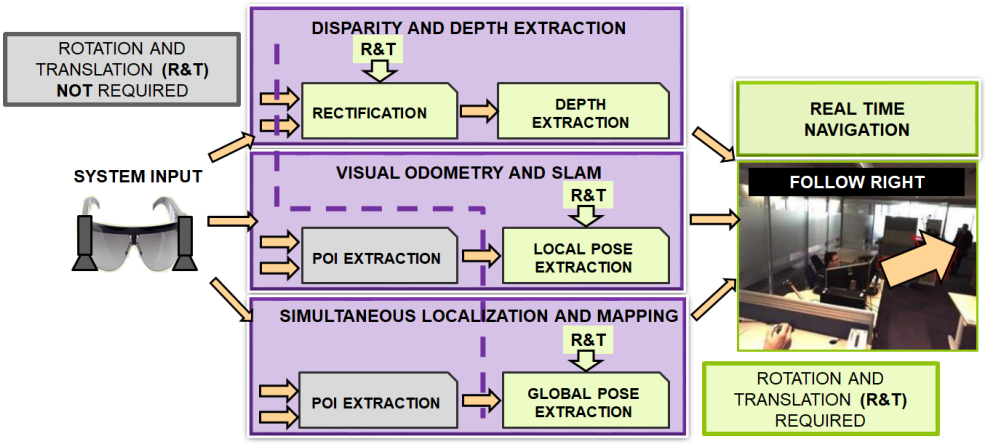


Figure 3.6: Navigation pipeline divided in small blocks.

The pre-processing data that detect extract and describe corners known also as point of interest detection (POI). These particular pre-processing functions are the most common functions for all image-based applications. This approach is fundamentally and very important, because in such system we do not need repeat some of calculations. It saves computational loads, thus creates less complex systems. In the practice, computing power and memory limits each application. In contrast to programs realized in the on the PC or computer cloud, where the limitations are not strict, the application parameters in an embedded system must be very restricted. The first proposition for whole pipeline is to use common pre-processing functions for all applications once. This methodology can accelerate a global navigation pipeline. If it is necessary, it can perform the specific adjustment of the output data. Appropriate precision of points detection and features descriptions have a significant impact on the quality of the computer vision application. Therefore, it is a very important step of the whole navigation pipeline. In order to accomplish pre-processing with higher precision, it is necessary to eliminate distortions and the impact of the focal length. For this reason, functions require initial intrinsic (internal) camera parameters [22]. Initial parameters are in the system and system guarantees the stability of these

parameters. In that case, the monitoring and recalibration of intrinsic camera parameters is not necessary. On the other hand, many forces can change the camera positions so the extrinsic camera parameters in the setup mounted in the custom smart glasses frame. Therefore, the whole system must have the possibility to update the extrinsic camera parameters while it is required.

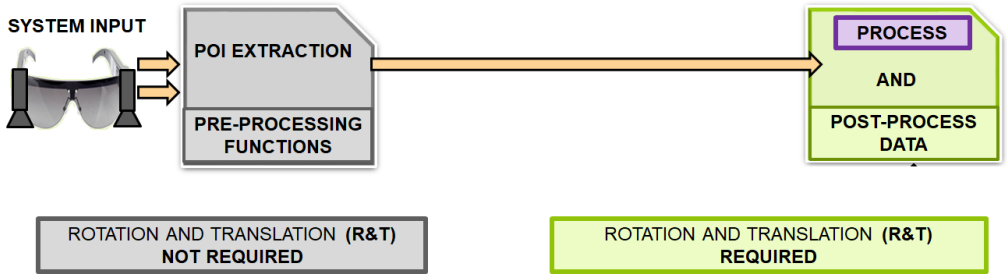


Figure 3.7: Navigation pipeline divided in two groups of functions, preprocessing that does not need extrinsic parameters and post process, which need.

Fig 3.7 shows that the pre-processing functions do not require the extrinsic camera parameters. Those are mandatory for post process so high-level applications that realize the navigation processing. Non-precise or not actual extrinsic parameters can lead to serious errors in the high-level application and fail the navigation mission. The online calibration must be included in the whole navigation pipeline. The traditional approach in the classical computer vision pipelines provides the intrinsic and extrinsic camera parameters offline once at the beginning. It is always before the first use sometimes during the production process. While the current state so the extrinsic parameters change, the pre-processing output is constantly true. This work propose to hide the online calibration between preprocessing and post processing data. The selected calibration algorithm from the second chapter can rely on the output from the pre-processing functions. In this custom approach, the calibration procedure does not require any input data preparation, because all necessary data are already in the system generated for other purpose by the pre-processing functions. This can significantly reduce the whole calibration costs and allow realizing it in real time on embedded system. We present the final simplified approach with online camera calibration service in Fig 3.8.

3.2.3 Low level monocular pre-processing functions

Pre-processing functions use only the stereo image sequence and intrinsic camera parameters. In practice, the left and right camera provides two-separated monocular raw video. Fig 3.9 presents the most basic and popular pre-processing functions for custom system, which deliver data for high-level application and CPS. However, in future, it is possible to expand the group of pre-processing methods by other additional procedures.

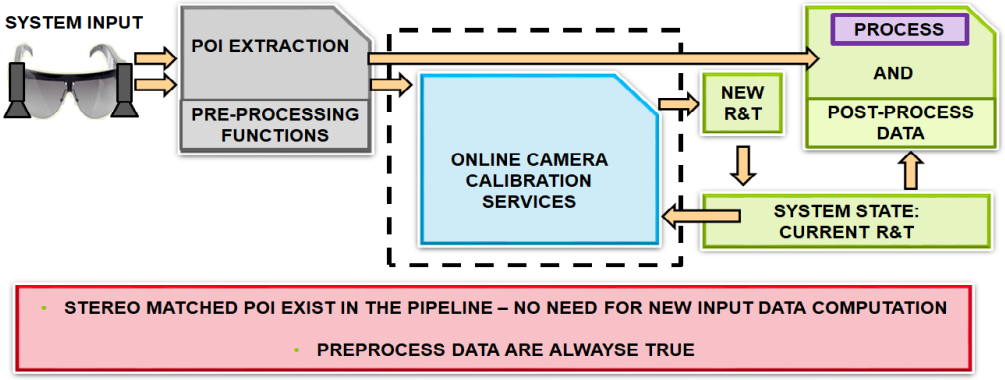


Figure 3.8: Simplistic navigation pipeline with online camera calibration block.

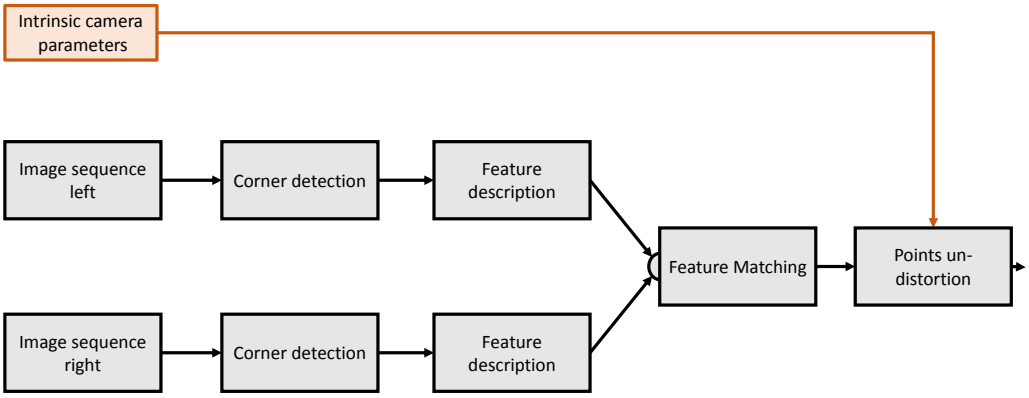


Figure 3.9: Pre-processing pipeline based on divided functions.

POI detection is the first function in our navigation pipeline. In the literature, there are many names used interchangeably for the same term: "Point of interest" (POI), "Corner" or "Feature". There are different method to extract points for example: Harris, Stephens, Plessey or Shi–Tomasi algorithms. They have different parameters and complexity [82]. The standard POI has a well-defined position in pixel coordinate. It usually represents as the intersection of two edges. It is possible to detect point without knowing the intrinsic parameters of the camera. However, because of the camera lens and radial distortions, the actual position of the point may be slightly different in reality.

POI description is required in order to distinguish and characterize the different detected POI. The local image structure around the feature (neighborhood) is rich in terms of data information contents that is used to specific corner descriptions. As for POI detection, various types of methods exist in order to describe a POI such as ORB, SURF, FREAK or BRISK. All of them have unique specification and different complexity [148].

POI matching is realized when POI are detected and described. Thanks to the unique descriptor parameter, point can connect the same corners from different views. Each descriptor contains local

information and orientation that helps to compare a different POI. Each time the closest value of descriptor is paired and the connection between points is established. This allow tracking similar features from more than two images frames. This process is necessary in order to track the specific elements of the scene.

Removing distortion from the corner position is an essential process to increase precision and achieve a true point position. The perfect camera's construction does not exist in the standard computer vision applications. Therefore, the system must eliminate the distortion caused by a camera lens. The most common image distortions are the radial and tangential distortions both can be eliminated from a point in pixel coordinates or for whole image. There is the Brown-Conrad model [78] which removes and corrects both distortion.

Hardware optimization is the way of solving and optimizing problems associated with many pre-processing functions of computer vision applications. The frame per second (FPS) parameter represents the number of images registered by camera during one second. The low-level functions use each of all delivered by the cameras. The applications set the FPS and size of the image. However, many cameras provide more than 20 images per second (20 FPS) with a size of 800 $\times$ 600 pixels. We can notice that it represents a huge amount of data. If we consider that on each image we have to realize much mathematical operation, it gives a lot of computation load. Then it cannot work in real time. The standard pre-processing functions on PC have a many resources to use and the Haris POI detection and SURF descriptions still take about 50ms while the image has resolution 640 $\times$ 480 [101]. Therefore, it seems interesting to accelerate this type of procedure with the help of modern hardware solutions. Linear mathematical operations can perform on the data flow the low-level functions such detection, description and matching. The hardware architecture relies on the GPU or FPGA accelerates many repetitive computations [150], [21].

3.2.4 Conclusion

The first section presents and describes the whole navigation pipeline of the custom CPS. The main goal and working condition place the purpose of devices in a global context. We analyzed the principal applications of custom navigation target in term of input, processing and output data. We propose the new approaches to divide the global navigation pipeline in two groups. Many complex functions use the same low-level functions. Therefore, the first group refers to pre-processing functions. Those require the monocular stream of one camera flow with the intrinsic camera parameters. They detect, describe, match and remove distortion of the point of interest. They do not need the extrinsic stereo camera parameters to work properly so they are always true for the whole time of mission.

The second group is the process and post process data of the navigation that rely on stereocular

data. They need the preprocessing functions and the extrinsic camera parameters in order to understand the relation between the two image streams. The analyzed applications in custom CPS such as VO, visual SLAM and depth extraction need the extrinsic parameters to work properly.

We propose to create and realize the online camera calibration block, just behind the pre-processing functions. The input data are available in such system. This approach can reduce additional computation, because the input data for calibration are already in the navigation pipeline. It allows hiding the online calibration in to completely custom pipeline and realizing the procedure as the background operation.

The online calibration of the system increases the safety, reliability and precision of the whole system. It can confirm if currently working system to realize a mission with high precision or not. The online self-calibration procedure in the system realizes the self-healing and self-adapting concept for custom CPS. Many devices may apply the same methodology, and gain profits for the stereo camera use.

3.3 Calibration based on primitive approach

As we presented in the 2nd chapter, we select the 8PA algorithm in order to solve the online calibration in the navigation pipeline. According to embedded system constraints, the low-complicated algorithm is required. The 8PA method uses only simple POI. The same kind of features are common to other applications, so they must be available in the system, where the computer vision process exists. As mentioned in section 3.2.3 the pre-processing functions deliver POI. This methodology allows simplifying and accelerating the calibration process, because input data are already calculated. We propose to use this methodology in the custom CPS. The low-processing functions that detect POI are in the out navigation pipeline. The proposed 8PA algorithm takes profits and hides in the whole processing pipeline. Fig 3.10 presents the most basic primitive implementation of the selected algorithm, detailed information about specific blocks can be found in section 2.3.2.

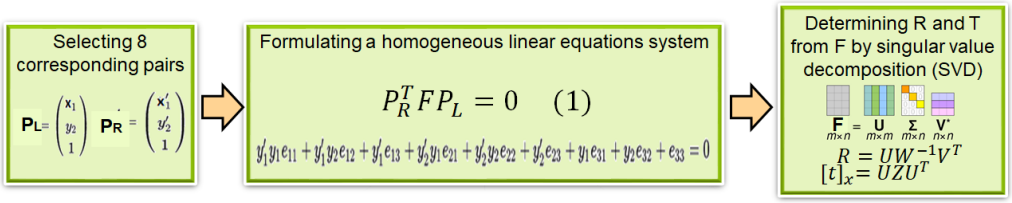


Figure 3.10: Pipeline of online calibration blocks.

Resuming, the epipolar geometry formulates the set of homogeneous linear equation. The formula uses eight different correspondences between stereo POI from the left and right images. The system computes the fundamental matrix (F), when POI are not normalized, so they are in pixel coordinate frame. On the other hand, the system calculates the essential matrix (E) if POI are normalized, so expressed in the camera system coordinates. The singular value decomposition converts E to rotation (R) and translation (T). It represents the normalized relation between two images. The section 6.3 explains those steps in details.

3.3.1 Preparing input data

The 8PA algorithm computes model from only eight correspondences POI. Therefore, choosing the best eight points is extremely important. More stable and precise features allow obtaining more accurate model. However, it is usually a difficult task since point localization is noisy, in order to multiple aspects such: incorrect illumination, distortion, false matches, etc. That is why an algorithm requires optimizations in order to select the best POI.

Removal of distortion

We use the pre-processing functions of the system to correct the distortion caused by imperfections of camera sensor. We use the Brown-Conrad distortion model to correct radial and tangential distortion [78]. Following equations 3.1, 3.2 undistort each point before 8PA block.

$$x_u = x_d + (x_d - x_c)(K_1 r^2 + K_2 r^4) + (P_1(r^2 + 2(x_d - x_c)^2) + 2P_2(x_d - x_c)(y_d - y_c)) \tag{3.1}$$

$$y_u = y_d + (y_d - y_c)(K_1 r^2 + K_2 r^4) + (2P_1((x_d - x_c)(y_d - y_c) + P_2(r^2 + 2(y_d - y_c)^2)) \tag{3.2}$$

x_u, y_u = distorted image point as projected on image plane using specified lens
x_d, y_d = undistorted image point as projected by an ideal pinhole camera
x_c, y_c = distorted center assumed to be the principal point
K_n = radial distorted coefficient
P_n = tangential distorted coefficient
$r = \sqrt{(x_d - x_c)^2 + (y_d - y_c)^2}$ Euclidean distance

Table 3.6: Table presenting the parameters used in Brown-Conrad distortion model.

Normalizing

The normalizing block is the next preprocess function used as the supplementary part of the 8PA pipeline, which increases the stability and precision of the estimated model. The section 2.3.2 describe and explain the four different normalization methods, which we studied the isotropic, non-isotropic, by intrinsic parameters and bearing scaling. We extract the normalization part of the pre-processing functions, Fig 3.11 shows that input data are already normalized as the first step in the completely online calibration process.

If the system detects the POI with infinite arithmetical precision, the normalization process does not affect the results. However, in real case where the noise exists. We detect POI with pixel accuracy, (alternatively we propose a sub-pixel precision) with camera distortion. In such case, the normalization has a significant impact on the precision of results. We tested each of normalization methods. In the custom navigation pipeline, we use the bearing normalization.

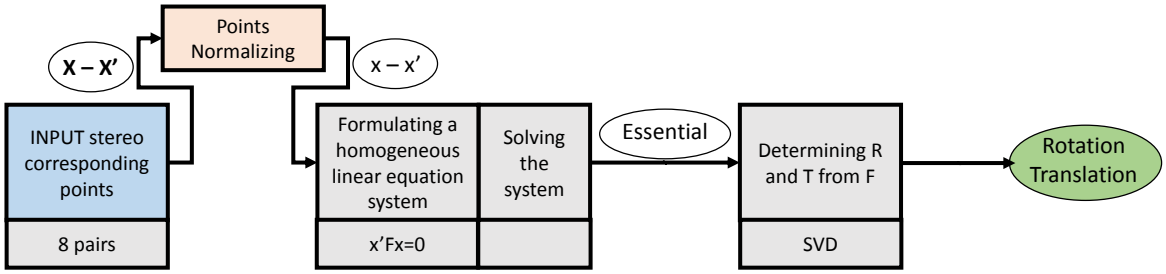


Figure 3.11: Pipeline of online calibration blocks with normalization block.

RANSAC, inliers and outliers

The preprocess functions detect and match many different stereo pairs of POI in each stereo image. If there is more than eight points, the algorithm must select the best points, in order to estimate the best model. It is a difficult task. In the real scenarios, the distance from the camera to the extracted POI play an important role. Faraway detected corners have less precision so its displacement in pixel coordinates is higher. The impact of the distortion on points detected farther from camera is much bigger. There are also many aspects, which lead to a wrong POI matching. With more input points, proportionally higher number of pairs are wrong match. Thus, many of them are less stable and lead to not precise model estimations. It is crucial to eliminate those aspects and select the best POI. In order to do that, it is popular to use a robust statistic tool such RANSAC. It allows to the robust estimation of the model. The section presents the mathematical background of RANSAC (section 2.3.2). It is usually better to estimate the model over the largest possible set of correspondences, but this has a significant impact on the length of the performed calculations.

The iteration is the one RANSAC's cycle. In the first cycle, the model uses an initial random set of eight POI to estimate the model. Then, the rest of the stereo pairs from the system verifies estimated model. Those pairs of point, which satisfied equation 1.4 and gives results smaller than the threshold, is the inlier. The result of such equation stands for epipolar error and we use this term name in the whole manuscript. The threshold value is an important feature in the RANSAC calculation [167]. In the future tests the default value of threshold is equal to $2.0 * (1.0 - cos(atan(sqrt(2.0) * 0.5/800.0))) = 7.8125e - 07$.

In contrast, the pairs of POI, which do not fit the current calculated model and give higher error than the threshold value, are the outlier. The algorithm keeps the model that contains the highest number of inliers compared to outliers, and with the smallest epipolar error. To prevent the infinite operations, RANSAC needs the maximal number iteration. Fig 3.12 illustrates the online camera calibration pipeline with the robust estimation RANSAC.

$$x_{left}^{'} M_{odel} x_{right} = 0 \tag{3.3}$$

$$x_{left}^{'} M_{odel} x_{right} < \text{threshold} \tag{3.4}$$

$$x_{left}^{'} M_{odel} x_{right} = epipolarerror \tag{3.5}$$

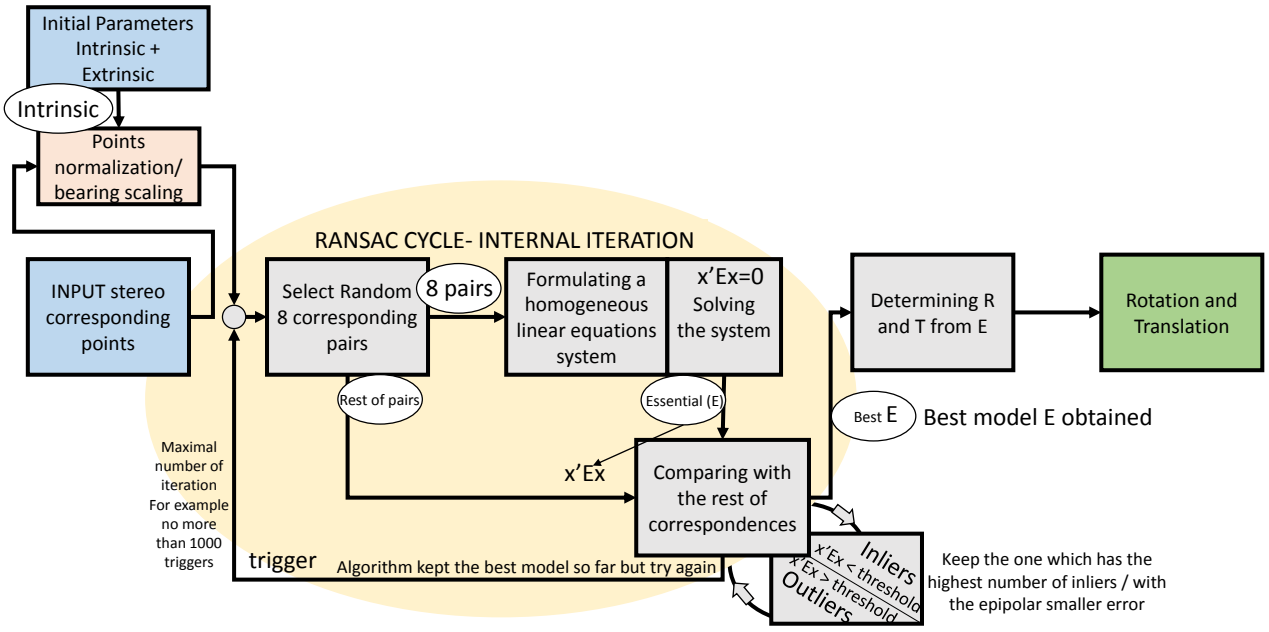


Figure 3.12: Pipeline of online calibration blocks.

3.3.2 Representing the results

We must extract the extrinsic parameters from the model, which the 8PA calculates. The translation T and rotation R express the actual distance and orientation between the two positions of the camera (for details see section 6.1.1). In order to verify if the extrinsic parameters are precise, we compare parameters found by online camera calibration method with parameters found by the offline camera calibration method. To measure their differential, it is necessary to have them in the same scale and norm. Comparing the simplest online method to the traditional offline method is ambiguous and unreliable. Due to the fact, that one method uses the excellent, well-distributed POI and extracted from known objects. In the contrast to the other, which uses features without guarantee of their distribution and stability. Nevertheless, we compare both method by comparing the extrinsic camera parameters in form of θ that represents the Error of Rotation and e_1 the Error of Translation. In the whole manuscript, we use the same, both parameters to present the precision of obtained results. The following sections 3.3.2 explains in details, how we compute those errors.

However, in the real system, it is not always possible to compare the extrinsic parameters found by the online and offline procedure. The offline method provides parameters in beginning of work. When during the mission the online method recalculates parameters, it has not new parameters found by offline method. That is why, the other possibility to verify and judge a precision of new camera parameters must exist in the online camera calibration pipeline. In this work, we analyzed the high-level (post process) application as another feasibility of parameters evaluation.

Error of Rotation express in θ

We realize all steps described by Huynh to compare between two R matrixes [79]. The first matrix is found by the offline traditional stereo camera calibration method (reference value), the second by online computation from 8PA andSVD. We use the Eigen library in order to convert both matrixes to quaternion form by equations 3.6, 3.7, 3.8 and 3.9. Then, the equation 3.10 normalizes the scalar of R. Finally, equation 3.11 compares two quaterians of R matrices. Where $(\langle Q1, Q2\rangle)$ is equal $a1 * a2 + b1 * b2 + c1 * c2 + d1 * d2$. The θ is in radians and express the Rotation Error. In the whole manuscript, for all tests, we use this error to evaluate quality of results. We can convert θ in to angular degree, by multiply by $180/\pi$.

$$a = (\sqrt{1 + R_{00} + R_{11} + R_{22}})/2 \tag{3.6}$$

$$b = (R_{21} - R_{12)}/(4 * qw) \tag{3.7}$$

$$c = (R_{02} - R_{20)}/(4 * qw) \tag{3.8}$$

$$d = (R_{10} - R_{01)}/(4 * qw) \tag{3.9}$$

$$Q1 = a + bi + cj + dk \text{ that satisfy } a^2 + b^2 + c^2 + d^2 = 1 \tag{3.10}$$

$$\theta = \arccos(2(\langle Q1, Q2 \rangle)^2 - 1) \tag{3.11}$$

Error of Translation express in e_1

We realize two vectors subtraction to compare between two T vectors. The offline traditional stereo camera calibration provides first reference vector and the 8PA with SVD calculates second vector. Both vector are in the same scale, normalized by equation 3.12. The equation 3.13 present error in e0 form when we consider that each ax represents the same direction in each vector. Therefore, we substrates each normalized in each axis separately. In the equation 3.14, we present second methodology, because during tests the continuing problem with axis sign appeared. In the error form of e_1, we subtract the absolute value. We call the e_0 and e_1 the Translation Error.

$$T_{Translation} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}; T_{Translation_{norm}} \begin{bmatrix} x_{norm} \\ y_{norm} \\ z_{norm} \end{bmatrix} = \frac{1}{\sqrt{x^2 + y^2 + z^2}} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.12}$$

$$e_0 = \sqrt{(x_{ref} - x_{cal})^2 + (y_{ref} - y_{cal})^2 + (z_{ref} - z_{cal})^2} \tag{3.13}$$

$$e_1 = \sqrt{(|x_{ref}| - |x_{cal}|)^2 + (|y_{ref}| - |y_{cal}|)^2 + (|z_{ref}| - |z_{cal}|)^2} \tag{3.14}$$

Error estimation based on high-level application

In the following section, we extend the idea that the high-level functions validate the accuracy of R and T. The particular CPS and application require different precisions of extrinsic parameters. There is not perfect stereo camera calibration method and universal precision quality that each high-level application requires. The precision of parameters is good enough, if the mission of the application works effectively. We analyzed some internal parameters and output data to judge precision of current extrinsic parameters in the navigation pipeline. The visual odometry can use a triangulation or re-projections error. In order to verify, if new estimated camera parameters are more precise than old one. The depth map extraction has internal parameters describing erroneous points. The system can use it for evaluation and new parameter verification. We develop this type of project at the laboratory,

so in the future, it will be possible to use it, for such purposes.

Due to the limited duration of doctoral thesis, we only consider a methodology for precision verification by depth map extraction, without any future tests. Fig 3.13 presents stereo camera system mounted on the helmet. It observes an adapted scene with specific objects with known standardize dimensions. We propose to compare those sizes with the depth extracted from the images. It is important to verify images from several different, relatively close distances from the camera. The system analyzes the dependencies between depth map and known objects in the scene, only at the specific moment. When the known objects, such as road signs, doors or other elements (which sizes can be standardized), are detected and recognized on the scene.

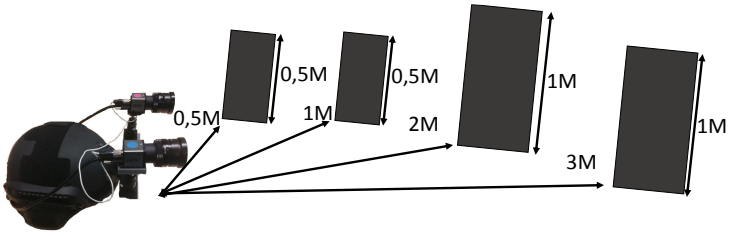


Figure 3.13: Scenario for precision of extrinsic camera parameters estimation based on depth-map extraction.

3.4 First tests based on primitive of calibration

This section presents the basic test of the naive approach of 8PA with bearing normalization and RANSAC. Fig 3.14 shows the primitive pipeline in order to verify and compare the offline and online method on two basic recorded dataset. For the first, the perfect input points, for the second, the points from one stereo frame are used. We present the results each time on the three charts. The figures illustrate the most important characteristic of the camera calibration parameters. We use the same data to illustrate the results in each test for the whole manuscript: the error of R in θ (see section 3.3.2), error of T in e_1 (see section 3.3.2), and number of iterations to estimate the best current model (see section 3.3.1). We present the parameters according to the number of input stereo pairs and number of inliers.

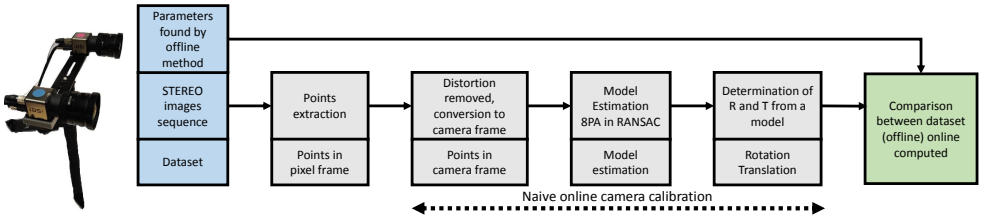
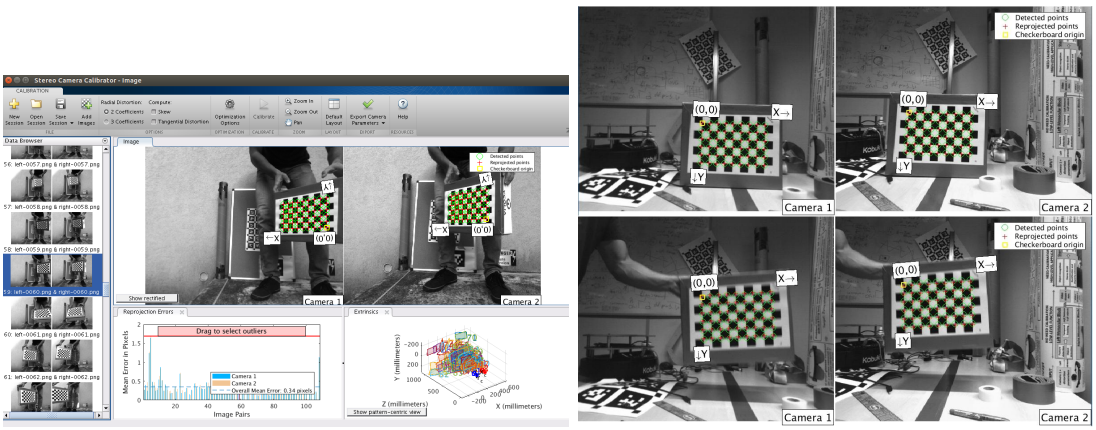


Figure 3.14: The naive pipeline shows two online camera calibration methods in parallel.

3.4.1 Parameters found by reference offline method - Matlab API

We record first dataset in order to verify primitive approach. It contains two image streams registered by a set of stereo camera shown in the left part of the pipeline from Fig 3.14. We use cameras, which are in one stable position. We put the calibration pattern (chessboard) in their field of view, as illustrated in Fig 3.15. In order to compare results to offline method, we realized the traditional calibration procedure from Matlab stereo application. It provides all camera parameters: intrinsic, distortion, extrinsic R, T), F and E matrix. Moreover, it gives the all POI detected from the camera chessboard. Fig 3.15a presents the screen of the application. The method implemented in the Matlab use the Zhang's traditional calibration procedures with classical chessboard pattern [189]. We know the number and size of squares. In theory, this traditional offline method requires at least three different poses of the calibration pattern. However, in practice, to provide precise results, it requires between 10 and 20 images. It is important to distribute chessboard's poses, in the whole scene, not far from the camera in order to obtain good parameters. In section 2.2, we describe this calibration method in details, as it is a reference method in this work.



(a) Matlab stereo camera application to calibrate cameras.

(b) Perfect, stereo points from chessboard (calibration pattern).

Figure 3.15: Traditional offline stereo camera calibration method in Matlab API. Detected points from right image are used as input for naive approach.

3.4.2 Points from the chessboard.

At the beginning, we test the naive (pure) version of the 8PA. We realize it on the same input points, which offline method used. To measure and verify, if with perfect input pairs of POI, this naive approach can obtain similar precision of the extrinsic camera parameters as offline method. The system accumulates and detects the stereo POI from few different planes of calibration pattern with a high float precision (such as: x=890.1125 y=752.5964). Fig 3.15b presents two different planes of chessboard at left and right image.

Fig 3.16 presents the average value from ten measurements of error in θ and in e_1. Fig 3.17 illustrates the average number of the iterations to estimate the best model. For each plot, blue column in Y-axis gives an average value of a measured parameter. The X-axis presents the number of test. We indicate the number of input stereo pairs by the orange line with the indication on the right side of the graph. The black line display the number of inliers to estimate a final model. Table 3.17a presents the highest precision obtained for those tests.

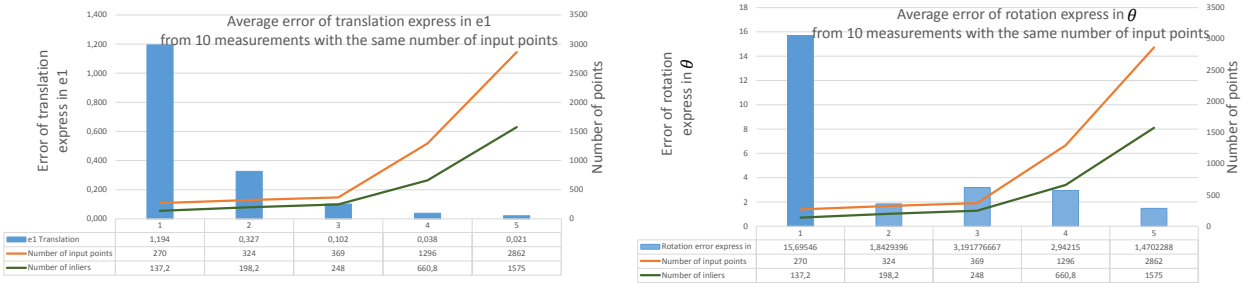
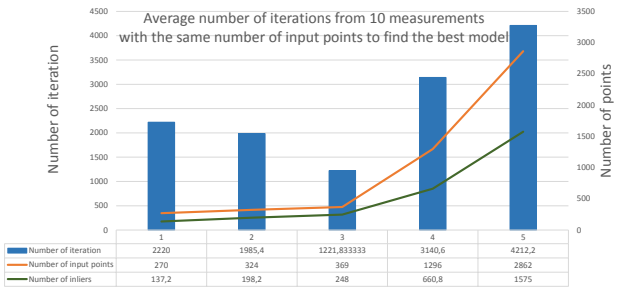


Figure 3.16: Average error of rotation in θ and translation in e_1 is calculated from 10 measurements, based on different number of points from chessboard pattern.

The higher number of perfect input POI (detected from calibration pattern) decrease the error of

R in θ and T in e_1. However, it increases the number of 8PA iteration required to estimate extrinsic parameters. Maximal tested value for primitive approach is 2862 input pairs. They allow calculating a relative right camera pose expressed in the left, similar to those, found by the offline camera calibration method. The differences in T error represented in e1 is equal to 0.02 of the Euclidean distances. For example, with this error if real distances between the two cameras is equal 15 cm then the online method recalculates 14.7 cm. The change between two R matrices is 1.5 degrees of θ.



Average error of R in θ	1.47
Average error of T in e_1	0.021
Average number of iteration	4212
Average number of inliers	1575
Input size of stereo match	2862

(a) The best precision obtained

Figure 3.17: Average number of iteration from 10 measurements and summary of the most precise method.

The perfect stereo points input for the camera calibration based on 8PA can provide the precise extrinsic camera parameters. Based on this we present the first hypothesis that the system must achieve similar precision of extracted POI from the real dataset in order to the online camera calibration works properly. In the next section, we will test it. From a practical point of view, in a single image, many of the detected features are usually in multiple planes. Therefore, the estimation of parameters must consider a large number of detected POI in a single stereo frame.

3.4.3 Points from one frame

The second dataset is recorded by the same stereo camera set, which was used in previous section 3.4.1. The naive calibration pipeline method run continuously, that the extrinsic parameters are computed from the pair of POI detected on only one stereo frame. Fig 3.18 shows points from the standard stereo frame of this custom dataset. The POI are not well distributed in each part of the image (due to the light and structures of the scene). However, in recorded scene, the features can be detected in many different planes. That is why, tests which are based on the one frame, theoretically, can provide a positive result.

The plots 3.19 and 3.20 illustrate calculated error of R represented in the θ and T in the e_1 in form of blue column. It is based on the same methodology, as in the previous test. The number of input stereo pairs from the current frame and a number of inliers are given in the X axis and in orange and black line. Moreover, next to each plot, there is a small table, showing an average value of each monitored parameter to increase readability.

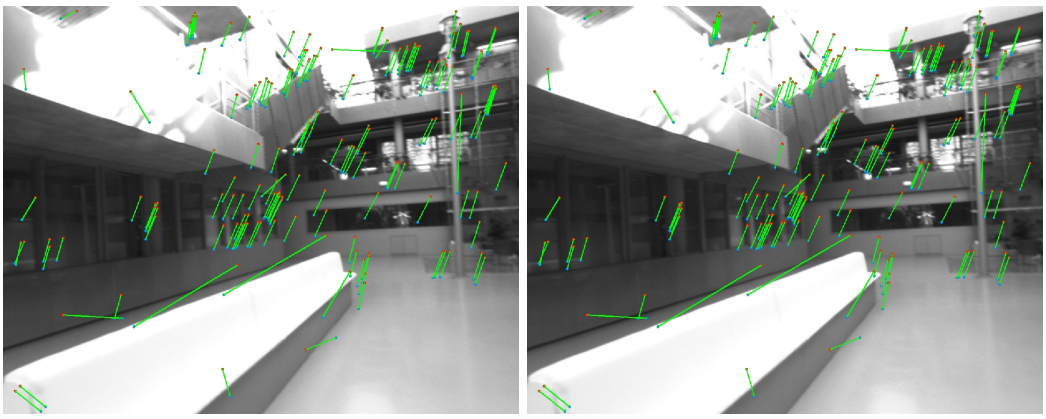
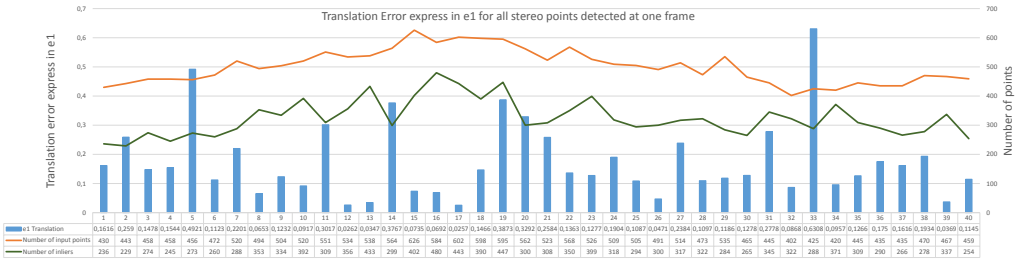
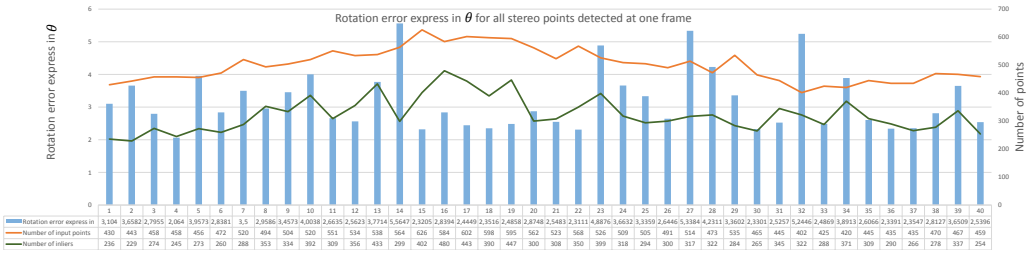


Figure 3.18: Input images with point detected on real stereo images recorded by custom camera set.



Average error of T in e_1	0.3183
Average number of inliers	324
Input size of stereo match	500

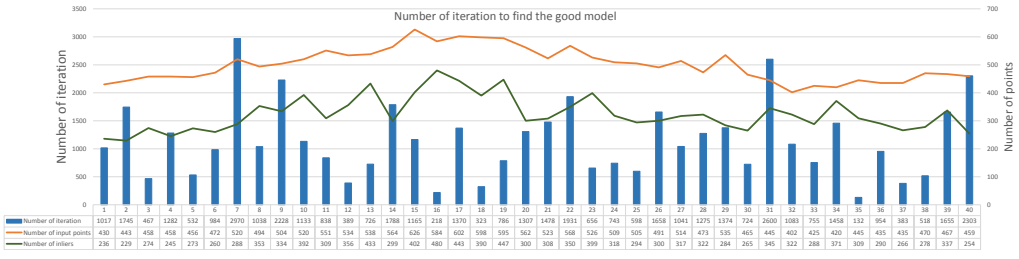
Figure 3.19: Error in translation expressed in e_1 calculated only on points from current frame.



Average error of R in θ	2.174
Average number of inliers	324
Input size of stereo match	500

Figure 3.20: Error in rotation expressed in θ calculated only on points from current frame.

Fig 3.21 presents blue columns which symbolize a number of algorithm iterations (RANSAC loops) to estimate the best model. The average time required to estimate the best model is around 200 milliseconds (0.2 second) during a test on classical PC.



Average number of iteration	1140
Average number of inliers	324
Input size of stereo match	500

Figure 3.21: Number of iteration required to estimate final model only on points from current frame.

The precision of extrinsic parameters obtained by online method, cannot be higher, than that, obtained while use the perfect input POI from calibration patter. Previous test 3.4.2 shows an error of R (θ) around 1.5 and error of T (e_1) around 0.02 of whole distance between cameras. That is why, the average error obtained during this test from points detected at one frame, is considered as very imprecise results, the θ equal 2.2 and e_1 0.32 (if the Euclidean distance is equal 15, method recalculate 10.2). Moreover, both errors are very unstable and vary a lot. Therefore, it is not possible to precise calibrate cameras when method is based on POI detected from only one, single image.

3.4.4 Discussion

The previous tests presented two different naive approaches of 8PA. The first shows the continuous calibration that performs the computation only on stereo POI detected in the current frames. The obtained results are unstable and not precise. They implicate that it is impossible to have a good enough POI in each of stereo images. On the other hand, the results of online calibration based on perfect POI gave precise parameters, when the input data accumulated from many frames and structures are distributed in the whole image's scene and many planes, etc.

We consider a hypothesis that the appropriate amount and precision of POI from many images of the real scene allow calculating the extrinsic parameters. For the reason, we present in the next section technique to accumulate POI for several or more than a dozen frames in order to perform calibration procedure. Moreover, we propose the function to distinguish the best and most stable POI from the whole group of input pairs. We consider that they must to be well distributed in a many different planes and distances from the cameras. To achieve, the same precision, system needs to have an ability to understand the POI with their properties (this point is close from the camera; this point is in the top-left part of the image, etc.).

The frequency of performing online camera calibration in the application pipeline is the other important aspect of this work. If system has to accumulate and collect sufficient number of POI, the question arrives, how to find a trigger, which inform that the system has already enough input data. System consider the POI in term of good distribution, various distances, high precision, etc. that allows performing calibration and obtaining the precise result.

We consider that maybe the camera calibration does not need to run each time, when there is enough point in the system, but run when there it is required. Then, we propose an additional functionality – the monitoring of extrinsic camera parameters. It triggers the algorithm in order to compute new parameters because the old one are not precise anymore. Performed results show that if there is more stereo POI, the results are more precise but algorithm requires more iteration, to estimates E so the R and T. Each algorithm iteration costs a computation power. It is important for embedded systems context to reduce the number of calculations to minimum. We realized the first tests on a PC equipped with an Intel 7. It is several times faster than the standard embedded system processor (ARM Cortex).

W present the detailed computer parameters into table 4.1.

Nevertheless, the algorithm needed approximately 200 milliseconds (0.2 second) to calculate the extrinsic parameters from the model, the time needed to prepare the data (normalize) is negligible on the architecture ×86-64 bits with 8 cores on the Intel 7 processor but must be considered for testing on the ARM processor in ×32 bits architecture. Next section presents the advanced approach to online calibration in the application pipeline on embedded system. It tries to propose a methodology to obtain as precise extrinsic parameters as it is possible and find the answers on those asked question.

3.5 Advanced stereo camera calibration approach

This part of the 3 chapter proposes different optimizations of the whole online calibration pipeline. We build the advance stereo calibration pipeline from the online stereo camera calibration procedure, which stands to OSCC and the stereo camera calibration monitoring SCCM. The main task of is to compute new extrinsic parameters, thanks to the 8PA. The second important task is to say if system requires the new calibration by verifying if current extrinsic parameters are precise enough.

We suggest accumulating the input POI with appropriate information. We propose a different filtering strategy, in order to choose the best stereo pairs and compute the precise extrinsic parameters. This should increase the precision of results and reduce the number of iterations to find the best model thus minimize the execution time of the calibration procedure.

3.5.1 Map Points of Interests, the accumulation strategy

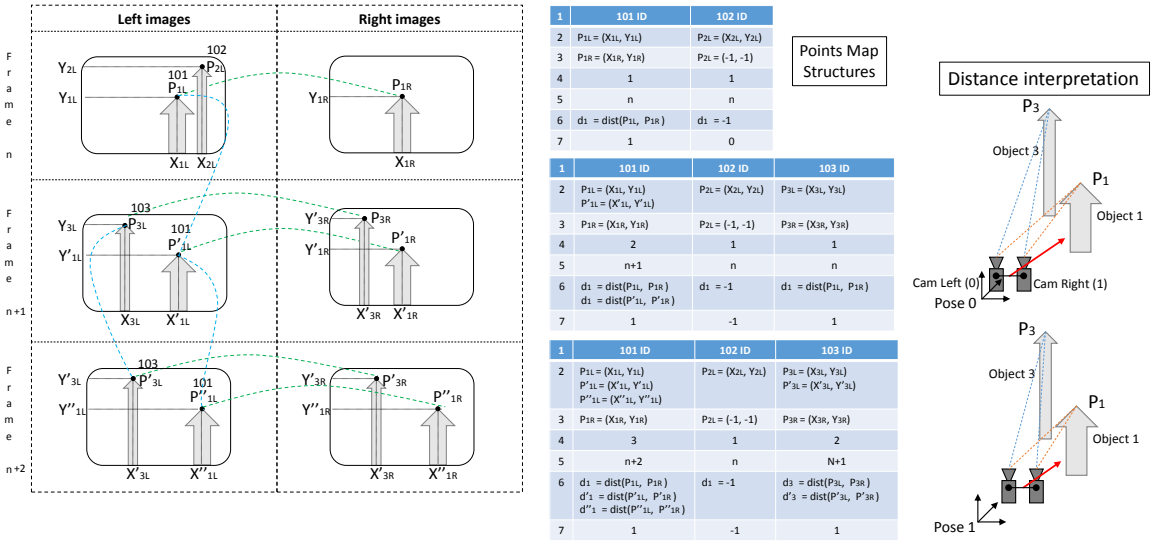
We use the same pre-processing functions from naïve calibration pipeline before the advanced stereo camera calibration. Stereo tracking is the detection of POI, matching between stereo frames. In the advanced approach, we propose simultaneously additional matching between current and previous frame. It stands to the temporal tracking and Fig 3.22 illustrates its idea. The system transmits the detected POI into the accumulation block. There, we create the "Map Points of Interests" (MPOI). It contains a basic information about points, which simple stereo-temporal tracking provides.

Long mission as a few minutes or hours, delivers a huge number of frames. For this reason, a simple accumulation and saving strategy of all stereo pairs from each frame is impossible. The high number of images significantly increases the number of inputs POI. The memory always limits the program, thus the reduction of an infinite number of points must appear.

The left part of Fig 3.22 explains the stereo tracking. On the top, there is the last frame, so the oldest, which arrived. The preprocessing functions detect two POI. Each of them gets a global identification data (ID) number, for example, P_{1L} obtains the 101 it is the first element in the table 3.7. It is unique and assigned to POI at the detection process. We push the global ID, with the position represented in the pixel into the coordinate's vector of features, the float value represents position and it strongly depends on the detector. For example, system allocates the first time detected POI: $P_{1L} = (X_{1L}, Y_{1L})$ as the first element of vector presented in the second row of the table 3.7.

If, this POI has an equivalent (matched) feature in the right image, it is the stereo track. When system matches the point P_{1L} with and P_{1R}, it pushes coordinates $P_{1R} = (X_{1R}, Y_{1R})$ to the vector of POI correspondence from right camera (element 3rd in the table 3.7). If, it does not have a matched correspondence (like examples P_{2L})), the $P_{2R} = (-1, -1)$ fills the right vector.

The new group of POI arrive with the next frame (middle frame n-1). Thanks to the temporal tracking, accumulation block knows if system tracks a new point in the previous left frame. If this



1	101 ID	102 ID
2	$P_{1L} = (X_{1L}, Y_{1L})$	$P_{2L} = (X_{2L}, Y_{2L})$
3	$P_{1R} = (X_{1R}, Y_{1R})$	$P_{2L} = (-1, -1)$
4	1	1
5	n	n
6	$d_1 = dist(P_{1L}, P_{1R})$	$d_1 = -1$
7	1	0

1	101 ID	102 ID	103 ID
2	$P_{1L} = (X_{1L}, Y_{1L})$ $P'_{1L} = (X'_{1L}, Y'_{1L})$	$P_{2L} = (X_{2L}, Y_{2L})$	$P_{3L} = (X_{3L}, Y_{3L})$
3	$P_{1R} = (X_{1R}, Y_{1R})$	$P_{2L} = (-1, -1)$	$P_{3R} = (X_{3R}, Y_{3R})$
4	2	1	1
5	n+1	n	n
6	$d_1 = dist(P_{1L}, P_{1R})$ $d_1 = dist(P'_{1L}, P'_{1R})$	$d_1 = -1$	$d_1 = dist(P_{1L}, P_{1R})$
7	1	-1	1

1	101 ID	102 ID	103 ID
2	$P_{1L} = (X_{1L}, Y_{1L})$ $P'_{1L} = (X'_{1L}, Y'_{1L})$ $P''_{1L} = (X''_{1L}, Y''_{1L})$	$P_{2L} = (X_{2L}, Y_{2L})$	$P_{3L} = (X_{3L}, Y_{3L})$ $P'_{3L} = (X'_{3L}, Y'_{3L})$
3	$P_{1R} = (X_{1R}, Y_{1R})$	$P_{2L} = (-1, -1)$	$P_{3R} = (X_{3R}, Y_{3R})$
4	3	1	2
5	n+2	n	N+1
6	$d_1 = dist(P_{1L}, P_{1R})$ $d'_1 = dist(P'_{1L}, P'_{1R})$ $d''_1 = dist(P''_{1L}, P''_{1R})$	$d_1 = -1$	$d_3 = dist(P_{3L}, P_{3R})$ $d'_3 = dist(P'_{3L}, P'_{3R})$
7	1	-1	1

Figure 3.22: Left part of image explains in graphic mode the methodology of stereo and temporal tracking. The center of image presents the MPOI of detected points (for details see a table 3.7). In the right part of image, the real situation is illustrated.

point already existed in MPOI, it contains the same global ID (it helps match points). When point occurs $P'_{1L} = (X'_{1L}, Y'_{1L})$ and it is a continuation match of P_{1L}, it pushes the new position of POI into vector. It allows having a history of the tracked points from first detection in the structure. If such POI like P_{3L} does not exist before, it adds the new ID to the MPOI in the system.

The temporal tracking data provides information how long certain matched points exist in the system. Each POI has a global age (element 4th in the table 3.7), which is set at one when points appear at the register. This value is incremented as long as continuous detection of feature is possible. When detection is lost the incremental of age stops. The system consider the longer detected and matched POI as more stable compared to that detected spontaneously.

The last frame appears as another parameter (element 5th in the table 3.7). It referees in which frame, the POI was available last time. Based on this parameter, the system removes and cleans the accumulated data about points if point appears sufficient number frames ago (from current frame).

In order to storage additional information, about POI in the MPOI, the system registers the Euclidean distance between two matched points in element 6-th of the table 3.7. Each matched point pushes the distance like ($d_1 = dist(P_{1L}, P_{1R})$) to the vector. The right part of Fig 3.22 illustrates this methodology. The objects 1 and 3 has position fixed into the scene. P1 is much closer to the cameras than P3. There is the same movement for both cameras in the system, from pose 0 to pose 1 in time t. If the POI has larger differences in the distance (as in the case of P1), then its location is closer to the observer. If the distance is stable and the changes are in small steps, it means that the feature is far from the camera. Based on this information, in the future, the appropriate POI can be prioritize. That the algorithm focuses on the points which are in the nearest from cameras. The features that are far on the horizon usually have the smallest accuracy, due to the difficulty of detection and their

precise description.

Last parameter in the MPOI is the status (element 7th in the table 3.7), which is only a control value, used for organization. This parameter allows to fast interpretation of the selected POI. If the values is equal "minus one", it represents that POI is not tracked any more. While "zero" means that point is tracked only in left image, and "one" means that point is matched between left and right frame.

Table 3.7 provides the whole structure of one POI with its temporal and stereo correspondence.

Data	Format
1) Global ID	unsigned int - global reference number
2) Position of the POI - camera left	vector of pair floating points - position of cam left points
3) Correspondence of the POI - camera right	vector of pair floating points - position of cam right points
4) Global age of point	unsigned int - from how many frames point is track and matched
5) Last frame appear	unsigned int - last number frame when point was seen
6) Distance between matched POI	vector of floating points - distance between left and right point
7) Status	unsigned int - what is current status of point

Table 3.7: Table presenting the most important information about points of interests.

3.5.2 Filtering points strategy

In the application pipeline, the MPOI contains all possible matches from last several frames, due to proposed accumulation strategy. It is an important and crucial to choose the most stable and precise POI for 8PA. The filtering with certain rules can search and choose the most appropriate points. The function should give them the proper priority, so that the algorithm can reject wrong, imprecise matched and base only on excellent POI.

The standard 8PA approach performs two loops. The first loop calculates the model from randomly selected eight stereo pairs from input POI. The second is to verify the current calculated model, through the rest of the POI pairs.

If the system during the first loop, thanks to filtering strategy, knows which POI are more stable and precise than others, can choose thus calculate more precise model faster. This methodology can eliminate a huge randomness thus it accelerates the search of the best model and reduces a number of loop iterations, so the whole computation. For the second loop, the filtering can reduce a number of input pairs so the algorithm has less point to verify, etc. In the following section, some of basic filtering methods are proposed:

Stereo-Temporal + age: the first filtering method uses two parameters: status (4th) and global age (7th) elements from the table 3.7. The POI must be stereo match at current frame and must appear at least from n frames. Points tracked for longer period of time (at least one frame) should be more.

Stereo-Temporal + age + key frames: this filtering strategy proposes to use tracked POI from only some frames, period called key-frame. We explain this on the one example, when the camera starts or drastically rotates and lose all previous tracked points. The points that appear in this new saturation can be more stable and more precise. We test all point filtering methods during the experimental phase.

Stereo-Temporal + age + key frames + distance: this strategy is similar to previous and inject the distance parameter in to account. When the motion of camera is in the direction of POI, the long tracked points start to appear far away and move closer to the camera. System can take the differences in distances in to account. This filtering method prioritizes from this distance and takes old points in the MPOI structure. On the other hand, the POI, which is far away, has lower precision in location detection. Therefore, system must find the appropriate balance between the length of tracking history (age of points, 4th element in the table 3.7) and the distance between the points (how far is point from the scene, 6th element the table 3.7)

In the 4 chapter, we explain and test the different strategy of filtering in the graphic form. We select the best method in term of results precision. The time required by various method is measured and its impact overall processing pipeline is verified. Filtering strategy should effect on the number of inliers (points satisfy a model) and outliers (points do not satisfy a model). Moreover, it has to reduce the number of RANSAC iterations and allows finding faster the best model.

This procedure should be meaningful while the code will transfer to embedded systems with a much weaker processor. The purpose of filtering is to select the best points in order to shorten the time of estimation of new extrinsic parameters of cameras.

3.5.3 Stereo Camera Calibration Monitoring

This section proposes a discussion about the stereo camera calibration monitoring (SCCM). At the beginning, it is worth to ask, whether the system must repeat calibration each time as separated thread running with low priority, or realized as main task only when it is required? The separated thread in an embedded system does not have many resources to disposition. That is why, on some small processors, it can be not sufficient to perform even low-complex algorithm. On other hand, if system realizes calibration only when it is necessary, it must recognize when parameters are not precise. How to realize functions which knows that the current extrinsic camera parameters are no longer valid?

Fig 3.23 presents the whole approach of online calibration pipeline. The point accumulation block creates MPOI just after normalization and preprocessing functions. Those points go directly to the SCCM block. POI from MPOI go to filtering process and then to online calibration block. The OSCC runs only if the trigger decision comes from monitoring. Additionally, the SCCM can received some information from the high-level application.

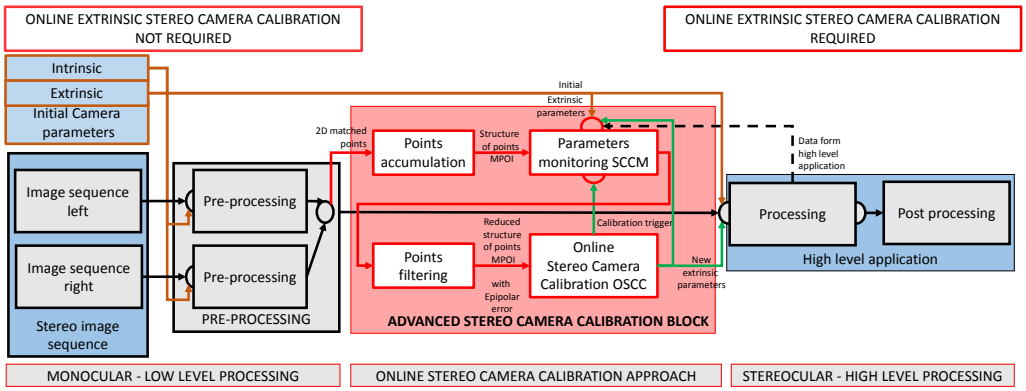


Figure 3.23: Final pipeline presenting advanced online stereo extrinsic calibration approach. The accumulation, filtering and online stereo monitoring of parameters are included in to calibration bloc.

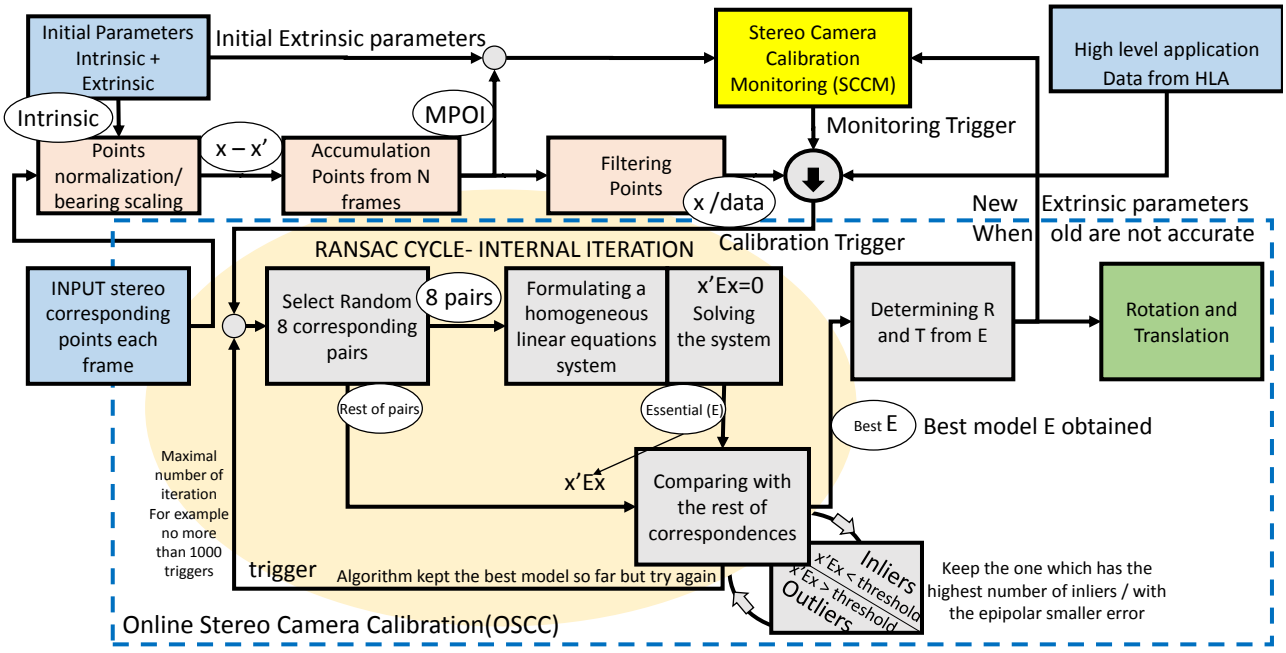


Figure 3.24: Zoom of the whole pipeline from Fig 3.23

We analyze two separate stereo camera calibration monitoring methods in this work. The section 3.3.2 presents the proposition that uses the high-level application. The dash line from processing block illustrates this feedback information from high-level. The second method relies on the epipolar geometry, described in section 6.3. The system verifies all filtered input points from MPOI by the same methodology used in the 8PA to select the best model. However, the method to check points need to use the current extrinsic parameters of the system. Equation 3.15 illustrates how we determine the threshold.

$$x'_{left} CurrentM_{odel} x_{right} = epipolarerror < \text{threshold} \tag{3.15}$$

This methodology seems to be universal for all types of applications, because initial parameters are usually available. Therefore, we describe this technique in details in the following section and test in the next chapter. The system requires initial extrinsic parameters in order to realize SCCM, which uses the epipolar geometry. The initial calibration provides the extrinsic parameters with the intrinsic parameters.

The table 3.7 presents the 8th element which represents the epipolar error and extends the table 3.7. System verifies by mathematical equalization of epipolar geometry $Error = x'_R E x_L$ each stereo POI. Then pushes obtained result to vector of errors. The system has to dispose the history of this point with correlated error. This error defines if point is inlier or outlier for a current existing model.

Data	Format
8) Error of correspondence points	vector of floating value- representing the $Error = x'_R E x_L$

Table 3.8: Table presenting a complemented value of following points.

We use specific dataset to test the SCCM approach. The two different calibration exists in one video stream. Fig 3.24 presents this kind of possible scenarios, where system has the calibrated and then uncalibration arrives. The axes shows on x-axis the time of action depending on the number of points fitting the current parameters on y-axis. The green line presents a hypothetical number of inliers and red line represents the number of outliers.

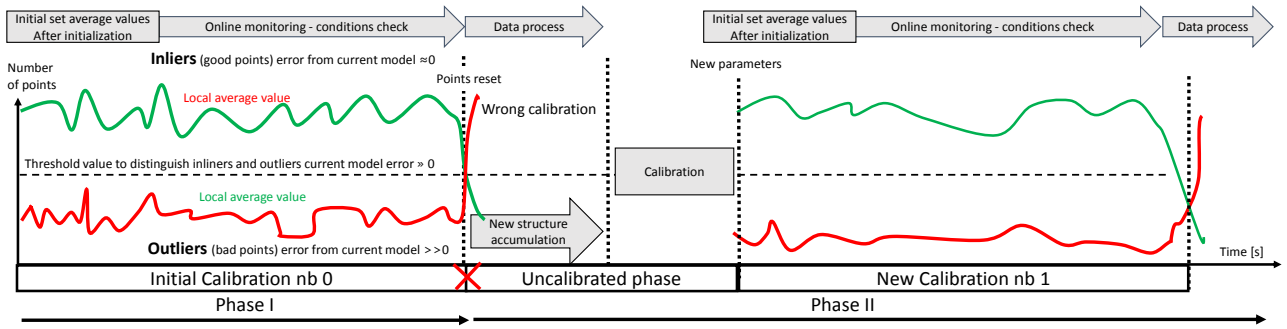


Figure 3.25: Scenario presents the whole approach of online camera monitoring.

In the beginning, system has a good initial calibration (nb 0) provided by an offline method. There should be much more inliers than outliers. The online monitoring measures a number of those points. At one moment, the uncalibration phase arrived. The ratio proportions between number of inliers to outliers and other parameters such as the average error and the median of error drastically changes. Based on those and other indicators the SCCM should be able to detect that extrinsic parameters of cameras changed.

In the MPOI structure, there is a history of points according to the old calibration. Therefore, the system cannot use it to estimate a new model. It needs to clean the old MPOI structure and starts to accumulate new points that satisfy current calibration. The system can perform the OSCC, once

there is enough new points accumulated. Once system calculates new extrinsic parameters, delivers to the pipeline (in Fig 3.25 nb 1). Once the precision of calculated parameters is high, the situation arrive to the similar state as with the initial parameters. The system is well calibrated, once the ratio and other parameters arrive to correct values. The SCCM can run again, but now it rely on the model calculated by OSCC.

3.5.4 Quality of Service current extrinsic parameters

Finally, in Fig 3.26 we propose the separate quality of service block that measure online camera current parameters and compute a new one. The main objective of the SCCM is to provide information on whether current external parameters are up to date with respect to previous excellent initial parameters. If they are not, the system triggers signal and sent it the system. The OSCC then tries to calculate the new parameters. Once they are delivered to the pipeline, they need to be verified how accurate they are.

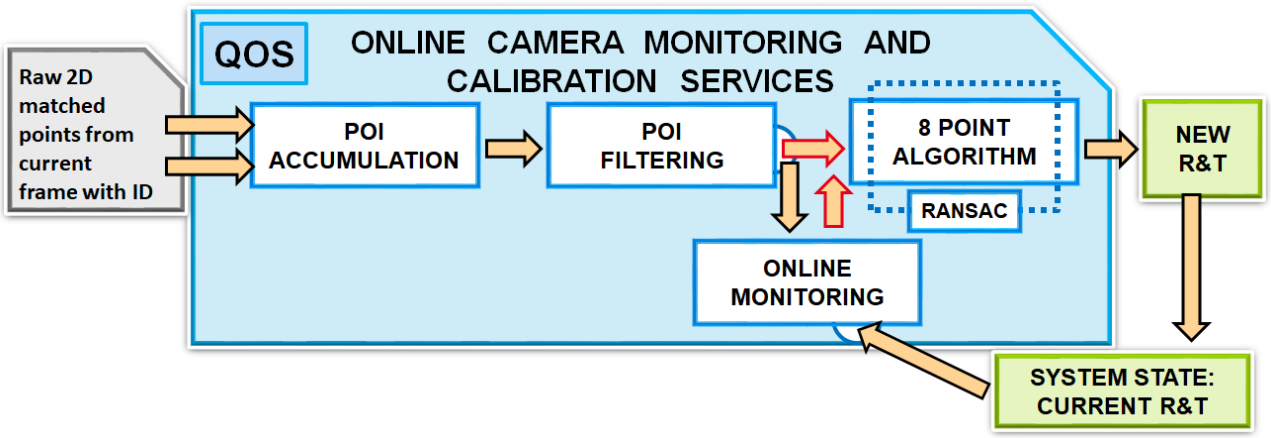


Figure 3.26: The camera calibration quality of service block.

We consider the whole calibration procedure as the quality of services (QoS) in the system. It provides the feedback data to the high-level application with information if the current system state so rotation and translation are precise. The system must verify the new calculated parameters, we propose the three levels of QoS.

- There is no OSCC function, then after decalibration phase detection, the system's mission must be continued in the hand mode, directly to the offline method calibration.
- There is OSCC function that provides temporal parameters. These are not precise, but the system can continue a mission in the safety mode. Then, the low quality of high-level application can guarantee that the mission does not fail. The user must escort the system to the place where it realizes the offline camera calibration, in order to provide new precise parameters.

- There is OSCC function that guarantees to provide a precise and high quality of the parameters. The mission and high-level application can continue the normal mode of work.

3.5.5 Scale Problem

The scale problem is one of the classic problems in the self-calibration method. It is the issue for monocular, stereocular visual odometry, SLAM and camera calibration. For applications, the scale provides information, which allows understanding the world. It can deliver the width of the road, the height of the corridor, traveled distance, etc. For calibration, scale is required to convert the extrinsic camera parameters from camera scale to real distances. We compare the obtained results (extrinsic camera parameters) in the proposed advanced calibration approach to an offline method in form of e_1 and θ, as described in section 3.3.2.

In the traditional calibration methods, the system uses patterns with known size, and then the scale problem is easily solvable. The each method realizes calibration thanks to connection of the observed world. However, the self-calibration methods must use different approach, when the calibration pattern is not available. We propose three main solutions in order to extract the scale:

- Scale extraction uses the sensor baseline. The distance between cameras cannot change, according to some devices.
- Scale extraction uses the objects in the scene. This approach is similar to the traditional calibration method but instead of known chessboard, it uses a known object. It is one proposition for the future work.
- Scale extraction uses data from the other sensors. We analyzed this approach however, we keep in mind that in the target devices we would like to limit the additional sensors.
- Scale extraction uses the deep convolution neural fields [185], because of an embedded system limitation, we do not consider this type of method.

Scale extracted from the baseline

In the beginning, the offline traditional calibration method provides the initial extrinsic parameters. The calibration pattern provides the real scale, when provide the bassline. The many CPS such as intelligent glasses do not allow change the camera's position significantly. In such case, the approach assumes that if the distance between the cameras essentially change, it destroys the system. In such case, the system does not need the online calibration. However, changes in the rotation of the camera in relation to the other camera is possible, if the construction of the glasses frame is delicate. This can lead to small differences in the camera are relatively often.

The system with loosely attached camera, without special metal cage makes it easy to connect to the robot or other CPS like smart glasses, helmet, etc. Fig 3.27 illustrates a mobile robot with

different setups of the stereo camera. The various orientations show what kind of camera position changes are possible. The position changes can be similar in the context of glasses, mainly R, but T changes minimal. With this assumption, we consider that the new calculated parameters may refer as the initial value, as presented in the section 3.3.2. In future tests, this approach is used. The recalculated parameters use the initial distance of the baseline between two cameras.

Figure 3.27: Possible settings of the camera without destroying the system.

Scale extracted from scene - object detection

We propose extract the scale factor directly from the objects localized in the scene. The principals of this methodology uses the same assumptions as the traditional method, i.e. finding a calibration object with known parameters. However, it should not be a classical calibration pattern such as chessboard etc. but some real object. The standardized object must appears broadly in the working scenario. The standardized and characterized object in term of size and shape is suitable for detection. This type of consideration is not generic and mainly based on the specific application, which works under certain scenarios in which it is used. Some objects widely appear in some group of scenarios.

We analyzed several types of elements: road sign, pedestrian lanes or doors. All of them appear in many environments and mission. They have constant standard sizes. Of course, it is hard to guarantee that such elements exist every time when it is required. Fig 3.28 presents a significant detection realized on the KITTI dataset. The system can obtain scale from time to time when an object is available.

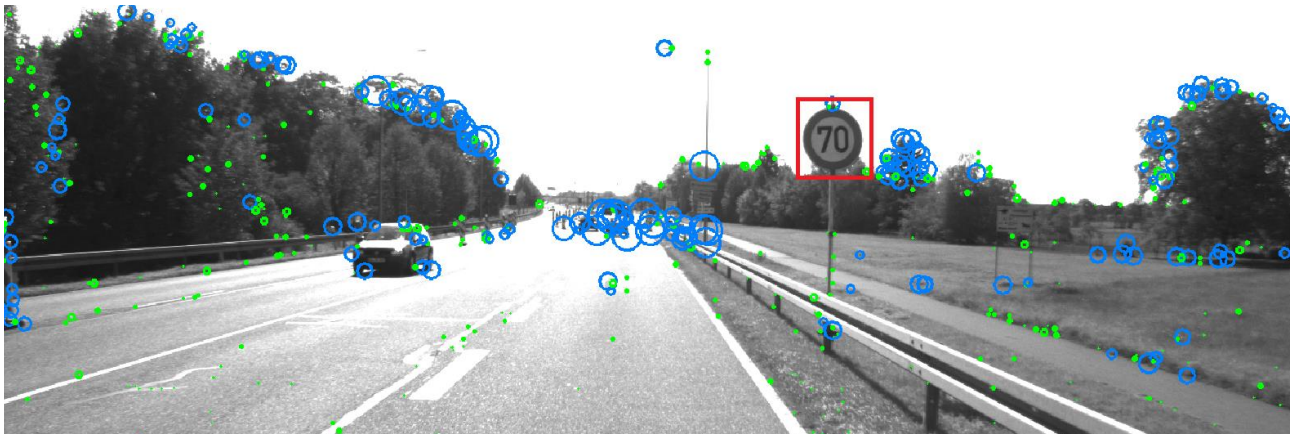


Figure 3.28: Traffic sign detection experiments.

Scale extracted from other sensor

In some particular examples, the computer vision application can combine different data from various sensors or application. We analyzed the visual data with the odometer that measure the traveled distance. The visual odometry can provide a different camera pose. The odometry realizes traveled distances. Then the relation between two distances may provide a factor in order to extract scale. The other example is the fuzzy of IMU data with the visual data. It provides similar information as odometer but in the short distances. Then the global application compares two distances. Other type of system, which can fuzzy different type of data, is the devices with LIDAR. The sensor provides information about the local environment. It allows confirming the distance to specific elements of the scene.

3.6 Conclusions about the presented approach

In the beginning, the methodology presents an approach to camera calibration, which hides in to processing pipeline, and use data from the system. The first section lists some of high-level applications that can run on custom smart-glasses context. We propose the CPS on the embedded system where input data comes from the loosely fixed stereo cameras. In order to satisfy a limitation of such devices we present an approach to online calibration in the application pipeline. We divide analyzed applications in pre-processing and post-processing group of functions.

The first pre-processing functions are similar for many computer vision applications. They realize low-level functions such as POI detection and description, etc. This group uses on monocular camera data, so it requires only intrinsic camera parameters, which are constant due to the camera construction. Then, the post-processing functions of high-level application use the stereocular data from two cameras. Therefore, they require all camera parameters (intrinsic, extrinsic) and data from low-level functions.

The section 3.2 proposes to use calibration method on the same data, which the post-processing functions used. The low-level functions provide sufficient data for some of self-calibration algorithm. Moreover, according to proposed methodology these data are available in the system, so calibration procedure can use it without additional processing. Thanks to this, the computer vision processing pipeline hides the calibration method. The next part of this chapter presents a primitive approach to extrinsic camera calibration, which standard literature presents. It describes the chosen algorithm: 8PA with basic steps. We recorded the dataset with sequences of stereo images by the custom stereo camera setup in order to validate our approach. The first test proves that perfect input data extracted from the chessboard (from offline camera calibration) allow estimating precise parameters. The obtained results are close to the offline method due to perfect input points. In future work, we expect to obtain the same similar precision of the extrinsic camera calibration.

After the second group of test realizes the same primitive approach of online stereo calibration on all points detected on one image scene. This has not sufficient predispositions to find and select the best points to estimate a precise model. The algorithm tries to calculate extrinsic camera parameters continuously after each frame. It leads us to section 3.4.4 where we try to answer, how often the system should perform the calibration. We consider that it should be possible to obtain a similar precision of parameters based on the points from the scene. Therefore, we propose additional optimizations to online stereo calibration pipeline in order to achieve more precise (closer to offline method) results. There are three main developed strategies described in section 3.5.

We propose the stereo camera monitoring strategy in the advanced stereo camera calibration pipeline. This technique allows realizing camera calibration only when it is required instead of continuous calibration. The special monitoring policies track all the input points. They use them to decide if current parameters are still valid.

This methodology significantly reduces the calibration costs. The system does not run calibration continuously all the time, but only then when it is necessary. For this strategy, the system needs initially calibration thus the intrinsic parameters of the first part of the pipeline (pre-processing), and the extrinsic parameters to start monitoring of the extrinsic parameters. We propose additional accumulation and filtering points in order to improve the monitoring and precision of the online calibration method. The accumulation saves the history and other basic information about the all tracked points. It provides input data not only from one, but also from many frames to algorithm. The system accumulates much specific information about point such as how far point is from the camera, how long it appears. After that block, we propose the filtering of such points based on many parameters. The main goal of this function is to reduce a number of input POI. Then it should select the most suitable points for the 8PA. As the result, it should stabilize and determine the results for the algorithm and reduces the workload and execution time.

We remind that the whole procedure runs on embedded systems. Thus, the autonomous, online stereo self-calibration method has to consume low processing power. It should allow for the use of loosely attached stereo camera, thus it enhances reliability, length of reliable operation of CPS, which is equipped with such sensor. The main aim of calibration is to recalculate (self-heal) and monitor extrinsic parameters in the system. The recalibration is a feedback loop in the whole system. The proposed methodology considers the calibration as the process in the background of the entire application. Moreover, it realizes a task in real time. In the next chapter, we present the results of our approach.

Chapter 4

Experiments and results

Nothing will work unless you do.

Maya Angelou

This chapter presents and comments the results of the online stereo camera calibration pipeline on different environment setups. We describe in detail the condition of work in the PC, RPi and CPS prototype. We present the methodology and realization of new dataset required for the test validation. This chapter realizes the online stereo camera monitoring, which use different policies and optimization blocks such as filtering and points accumulation in order to detect calibration need. We realize the continuous and triggered online camera calibration on two different datasets. The chapter finally presents the precision of the results and characteristic of the whole pipeline on an embedded system.

Objective :

Illustrate the results and methodology of the advance approach to the online stereo camera calibration in the application pipeline on an embedded system.

To do this, we :

- analyze different working environments.
- demonstrate the prototype.
- realize the custom dataset.
- explain the stereo camera calibration monitoring strategies.
- present the continuous and triggered stereo camera calibration.
- select best filtering method.
- realize the whole advanced online calibration.
- measure the impact on the high-level applications
- characterize the online stereo camera calibration on an embedded system.

4.1 Environment of experiments

We analyze the different environments of experiments and CPS, which need to realize the stereo camera calibration. We prepare the PC that is the most practical environment in our methodology. A high computational power with memory and easy interface are available. Additional, there is a large number of tutorials and community, which help to solve many problems related to compilation or connection of libraries for this environment. While PC can handle operation and mission with proper results. We select Raspberry pi as an embedded target. It is a popular low-cost single-board computer with a huge potential, widely supported by many open source projects. We propose some basic optimization and code improvement in order to satisfy constraints of the second environment. An embedded processor installed on board has similar parameters to the achievable processor in the third environments, which is a future prototype.

The code transfer from the PC to an embedded processor is usually the most complicated and tedious task. There appear many software and library's issues, compilation errors, memory leaks, lack of processing power and many other problems.

Finally, the code migrates to the final prototype when everything works well on the target embedded processor and the entire architecture. This means that the results must be close enough to the version released on the PC. This chosen methodology allows getting the best results in the shortest possible time. We use the Robot Operating System (ROS) to facilitate code transfer between different environments. This section describes all tested environments and interfaces with specific and detailed information.

4.1.1 ROS interface

Robot Operating System can confuse and mislead the user because, it is not real OS, but collection of libraries and tools that help software developers create and simulate robot applications or other CPS. ROS provides a hardware abstraction layer, device drivers, libraries, visualizers, message-passing, package management, etc. ROS perfectly fits into custom approach, because community develops the system on various machines with different architecture type (×86-64, ×32ARM). The stationary PCs or single-board computers, etc. has a specific dedicated version in order to run. The main programming language to write a code is the C++, but also C, python, java exists.

Thanks to ROS, it is possible to interface and cooperate with other open sources packages and with many more applications. For examples, a RQT software framework is available to visualize in the graphic way many different data available in the system. The machine vision provided by OpenCV packages allows creating a map, which localize agent in the environment, etc. A large community propose and support many other packages and applications.

The huge advantage of ROS is that every robot's component and function connect to specific pack-

ages due to distributed message to the whole system. We can divide the program in small independent blocks, known as the nodes. Those communicate and exchange data by publishing different messages to topics, between itself. A message is a data structure, and can contain each standard type such: integer, char, string, etc. in each structure such: vector, matrix, etc. The advantage of ROS is to receive and send packets of data with a customized form for each node. This activity is much easier to realize than the exchanging data between different programs not written in this environment.

Another interesting and useful command-line tool from ROS functionalities is the rosbag. It records a bag, which is a file format, which stores a ROS message data. There are varieties of tools that can store, process, analyze and visualize the bag. The rosbag player can select many parameters, while sending messages such as frequency, first frame, etc. in order to improve and accelerate results of the tests. It is a powerful tool, which can provide the identical input sequence. For example, it can play sequences of images with the same parameters and conditions, which allows monitoring and determining the same results each time. The section 2.5 presents the datasets provided by the vision community, each of them, we can convert to the rosbag, where the video stream in the ROS system plays the data flow.

Network configuration is another advantage of the ROS. It is possible to design and build a network consisting of many environments (devices) with different nodes. These nodes can listen to each other and exchange specific data, directly by network connection. The different proposed environments can communicate when its IP addresses is known. For example, it is possible to use a network, where the master host and peripheral slave devices can connect to the same local network, configured on one roscore server.

4.1.2 PC environment

The section introduces the first environment setup that uses the standard PC. The Linux OS is Ubuntu 16.04 distribution and others PC parameters, presented in Fig 4.1. Fig 4.2 shows synthesized view of the first working environments with ROS Lunar Loggerhead, where we implement the whole concept of the OSCC from the previous chapter. Following section, describe each separate node in the entire processing pipeline. We present the messages with the specific data structure used to send information between nodes in the arrow form. This approach makes it easier to create a calibration procedure, as other environments or devices can use and apply this solution. It provides a future possibility to extend and migrate the code to the other systems with similar environment.

- Dataset as known as rosbag feeds the pipeline. It contains of collected and synchronized time stamped data from the stereo camera with initial intrinsic and extrinsic parameters.

  OUTPUT: synchronized left and right image at demanded parameters as FPS, plus the initial intrinsic and extrinsic parameters.

Parameters	Setting
OS	Ubuntu 16.04
GNU C++ Compiler	g++ 5
CPU model	Intel(R) Core(TM) i7-2600 CPU @ 3.40GH
CPU MHz	1687.77
Memory size	8192 KB
Cache size	8192 KB
Core number of CPU	8
Architecture	×86-64

Figure 4.1: The parameters of the first test PC environment presentation.

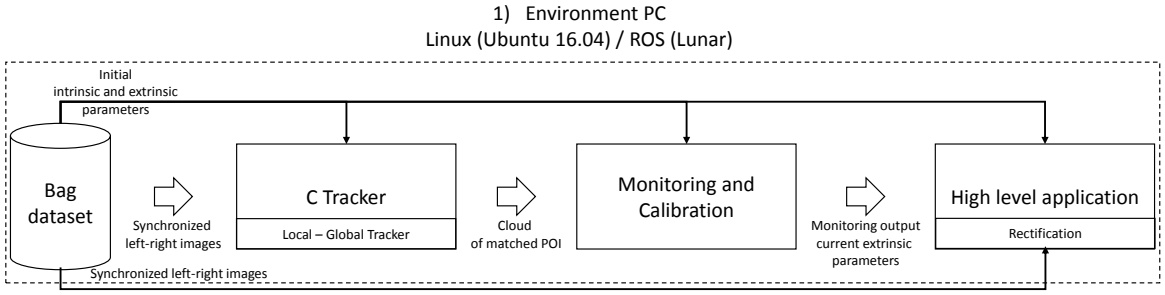


Figure 4.2: The whole processing pipeline executed on PC environment.

- Tracker contains functions to detect, describe and match POI between the synchronized input images: left, right and previous left frame. The section 3.2.3 presents the methodology. This approach divides the tracker in the local and global tracker. It returns the point cloud, which is a ROS structure used for POI descriptions. The section 3.5.1 details and explains the structure that contains all elements that allows realizing the point allocation. The system gives integer pixel value precision of the point. We develop the tracker in the laboratory, realized in C language with the hardware version realized on FPGA, described in section 4.1.4. During thesis, we use also tracker from OpenCV.

  OUTPUT: cloud of matched POI.

- Monitoring and Calibration block is responsible for the online calibration pipeline. It provides information when calibration is required and computes the new extrinsic camera parameters. The section presents the advanced approach with all the optimization. We implement all functions in C++ language and use the 8PA method from OpenGV library [88]. It is a dedicated library for solving calibrated central and non-central geometric vision problems.

  OUTPUT: new extrinsic camera parameters and monitoring output.

- In this particular scenario, we use the rectification process of the high-level application. It is a last block used to validate the calculated extrinsic parameters. It uses a geometric transformation to change camera configuration with non-parallel epipolar lines to the canonical one. The section 3.2.1 describes how it allows determining and finding a line-wise matching points. There is in

pre-installed default library into ROS, an OpenCV open source library [131], which contains functions to realize a rectification task. OUTPUT: two rectified images.

We use the identical nodes with the same form of messages in the other environments, thanks to the ROS environment.

4.1.3 Raspberry Pi 2B environment

The second working environment is small single-board computers known as Raspberry Pi. It stands to RPi. There are various models available on the market. We use 2B version with the Cortex A7 processor. The software setup in this embedded system platform is very similar to one used on thePC. The OS is Linux but the Ubuntu Mate 16.04 distribution with ROS Kinetic [136]. There are many tutorials, how to install and set up the whole environment on the ROS wiki web page. Table 4.3a presents the parameters of the second environment where OSCC runs.

Parameters	Setting
OS	Ubuntu Mate 16.04
GNU C++ Compiler	g++ 5
CPU model	quad-core ARM Cortex-A7
CPU MHz	900
RAM size	1000 KB
Cache size	20 MB
SWAP size	2 GiB
Core number of CPU	4
Architecture	×32

(a) Parameters.

(b) View.

Figure 4.3: RPi 2B the second environment presentation.

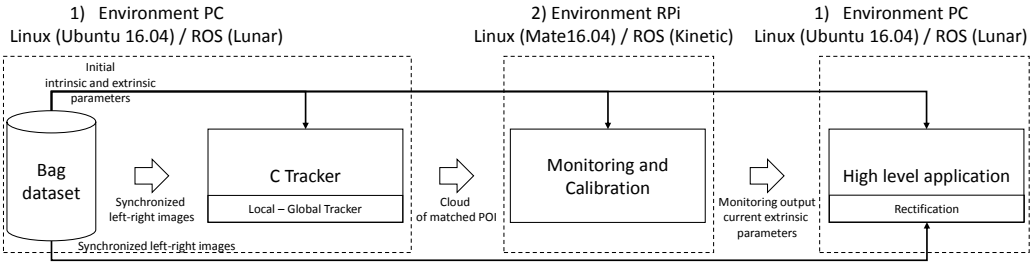


Figure 4.4: Pipeline executed in the PC and RPi environment.

The Fig 4.4 presents the processing pipeline in the second working environment. In this approach, we execute only monitoring and calibration node on the RPi. We characterize this node in term of execution time and precision. The tracker run on the PC and generate the cloud of points to the monitoring and calibration block. It returns the extrinsic camera parameters to the high-level applications, which as trucker runs on PC. To realize these tasks, the system needs a ROS network connection between different environments. In this section, we presents how to realize these steps. The last part of this section contains the instruction for lunching and compilation the entire pipeline in the second environment. The master host on PC and peripheral slave devices RPi creates one local network. The dataset with a tracker run on PC. It sends data packages to the RPi, where calibration executes. The Script 4.6a presents the network configuration.

Virtual memory (SWAP) allocation is required step in order to increase an available memory in

the system. Small, limited size of memory characterize the embedded systems as presented in section. The online camera calibration package requires compilation across several different libraries that use a complex structure. Thus, the system requires more that 2GiB RAM. On the RPi 2 B, only 1GiB memory is accessible, so the compilation is a challenge, in order to realize it we use a SWAP memory. It is a virtual memory able to stores the data in memory on disk. We are aware that reading from the disk is several orders of magnitude slower than reading from the memory, but we use it only for compilation level [17]. The Script 4.5c presents a code of bash.

We use the g++ 5 compiler to compile the monitoring and calibration node on ARM Cortex A7 processor in the RPi with the dynamic link to the. The system requires OpenGV library compilation, on the same embedded target. This step is not possible due its large size. From this reason, we cross-compile the library on the PC. Scripts 4.5a and 4.6b show the required commands. It is possible on Linux OS due to GNU C compiler ARM-linux-gnueabihf[43]. RPi documentation delivers the toolchain of ARM compiler, available in [137].

The execution of entire pipeline is possible due to the roslaunch tool. It is another interesting aspect of ROS. It allows launching multiple nodes locally and remotely via secure shell (SSH) settings. Each node from Fig 4.4 is launch from the same file. This technique simplifies the start of programs and allows you to predefined parameters for different functions. This is necessary for projects in which there are several work environments and many nodes. The roslaunch uses the XML configuration with the .launch extension.

```
add_definitions(-Wall -march=armv7-a -O2)
```

(a) Compiler option for OpenGV library for specific ARM processor with optimization -02.

```
    #roslaunch start
roslaunch calibration_whole_processing_pipeline.launch
```

(b) Examples, use of roslaunch tool.

```
#!/bin/bash
sudo fallocate -l 2G /swapfile && sudo mkswap /swapfile && sudo swapon /swapfile
```

(c) SWAP - Virtual memory allocation.

Figure 4.5: Scripts used to program compilation.

```
# addresses presented in master host roscore
export ROS_MASTER_URI=http://192.168.10.132:11311 #
      present the host's roscore services address
export ROS_HOSTNAME=192.168.10.132
export ROS_IP=192.168.10.132 # present the raspberry address
# addresses presented in slave devices connected to master
export ROS_IP=192.168.10.210 # present local address
export ROS_MASTER_URI=http://192.168.10.132:11311 #
      present host address
```

(a) Two scripts of network configuration for PC master host and slave RPI devices.

```
   #Toolchain - ARM processor
set(CMAKE_SYSTEM_NAME Generic)
set(CMAKE_SYSTEM_PROCESSOR ARM)

execute_process(
 COMMAND ${UTIL_SEARCH_CMD}
       ${TOOLCHAIN_PREFIX}gcc
 OUTPUT_VARIABLE BINUTILS_PATH
 OUTPUT_STRIP_TRAILING_WHITESPACE)

set(TOOLCHAIN_PREFIX arm-linux-gnueabihf-)
```

(b) Toolchain form Make option to cross-compile library on ARM A7.

Figure 4.6: Network and toolchain configuration scripts.

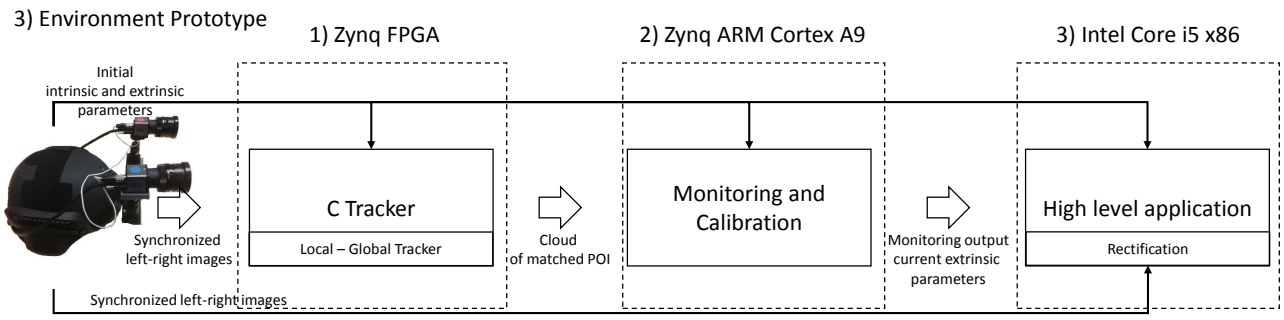
4.1.4 Prototype

We would like to dedicate the final device to help the visually impaired person in displacement. This CPS should be capable to localize and guide to the specific destination with abilities to new route and directions computations. The devices must be able to react to the local environment, as opposed to a standard navigation device. The detection of the scene's elements and changing dynamic data should enable on barriers and obstacle avoidance. The section 1.5 explains in the details the mission and device contexts, where the final version of the SCCM and the OSCC run. The third environment represents the software and hardware of this CPS prototype.

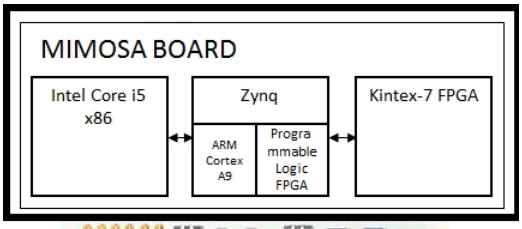
The hardware is the MIMOSA prototyping board designed and developed by CEA. It is equipped with two FPGA, Intel I5 and ARM Cortex A9. Fig 4.7b presents the approximate schematic of the whole pipeline. At this moment, the system realizes the low level processing functions such as point extraction, description and matching on the Kintex7. It realizes and returns exactly the same point cloud structure as the Tracker functions presented in the section 3.2.3. The same functions and output data create possibility to switch the environments very flexible and fast for testing purpose. The system can use the second FPGA for rectification and the depth map extraction. There is an ARM processor dedicated for the SCCM and OSCC procedures (see section 3.5). On the Intel i5, the system realizes the rest of high-level applications related to navigation.

The whole setup composes of the backpack and the helmet. Moreover, there is an electronic and battery hidden in the backpack as illustrated in Fig 4.7d. The electronic board with SD 500 GB drive is in the wooden box, visible in Fig 4.7e. The helmet is equipped with two stereo cameras presented in Fig 4.7f. Thanks to this, the whole system is portable. In this prototype, we test the SCCM with OSCC and present the results the following chapter.

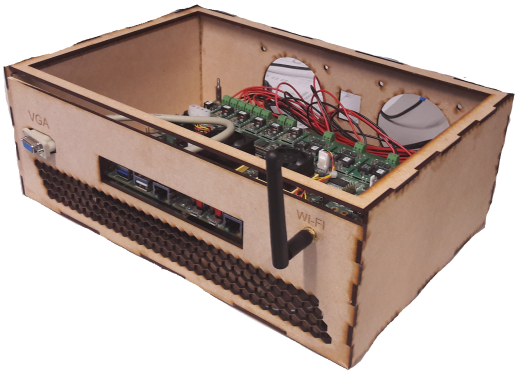
In future, instead of using this smart helmet prototype, we would like to propose the smart glasses concept. However, in order to realize it the system must reduce the size of electronics board and battery.



(a) Pipeline executed in the prototype environment.



(b) Overview of MIMOSA board developed by the CEA.



(c) Wooden box with MIMOSA board.

(d) Wooden box protecting the MIMOSA board.

(e) Backpack equipped with wooden box with batteries.

(f) Cameras mounted on the helmet which can change the position.

(g) Me with the whole prototype.

Figure 4.7: Prototype presentation.

4.2 Dataset

We studied different existing datasets from the literature. All of them are dedicated to the specific computer vision application. However, none of them depicts the scenario, where the realization of SCCM and OSCC make sense. The custom dataset requires scenario where the two different calibrations exists in one sequence. We compare the studied methods between each other in order to point the missing points for our scenarios. This section presents the specification of perfect dataset for this calibration purpose. Moreover, we propose and explain the methodology of handmade data acquisition.

4.2.1 The study of existing datasets

This section presents the main drawbacks from the datasets presented in the section 2.5. Each work provides the computer vision benchmark. The recorded data consists of many different sensors such IMU, LIDAR and the video flow of stereo cameras. It shows that existing datasets have a wide spectrum of applications. However, this work is interested only in the data from the stereo camera sensor, in order to perform extrinsic camera calibration.

The stereo sensor used in the dataset is usually mounted to the rigid metal platform. It is very reliable and stable approach. In all examples, it guarantees that the camera does not change intrinsic or extrinsic parameters. The standard approach ensures that the camera and its internal focus length are well fixed, thus the all camera parameters are constant from the beginning to the end of the mission. The moment of the decalibration does not appear, so the calibration is stable. Then, there is no need of re-calibration. Moreover, stereo camera position are usually ideally parallel to each other. In such cases, the R is in the identity matrix form so the obtained image planes are directly co-planar. This camera's setup is correct and desired by systems to simplify many calculations such as rectification process.

In additions, the Euclidean distance between the two cameras is characteristic for the majority of stereo sensors. The only significant distance is the one, along the X-axis. The other distances along Y and Z-axes are negligible. Those are disproportionately smaller compared to the X one, thus we omit them. However, this is a result of the most specific placement of the camera stereo sensor. Table 4.1 presents the main parameters of the analyzed datasets. We do not present the StLucia parameters in the table because the setup is similar the KITTI [60] [59].

From the analyzed dataset, the KITTI presents a very interesting collection of sequences, which provides images in high resolutions with many structures. Moreover, the dataset is widely used in the communities to present the computer vision results. The Fig 4.8 illustrates the three frames where the software program matches the stereo POI. The system continuously provides the data flow from the same level of the ground. There are many points in each part of the image, detected on different structures localized on various distances from the cameras. We use this dataset for future tests, even

if the moment of decalibration never happens in any sequences.

Dataset	KITTI	EuRoC [23]	Carla [44]
Realization	setup mounted on car	setup mounted on dron	simulation
Motion	to ground plane respect	unstable in any plane, fast dynamic movements	can be programmed
Additional sensors	LIDAR, IMU, GPS	laser, IMU	data similar to LIDAR
Stereo setup	stable, parameters constant	stable, parameters constant	the camera pose can be programmed
Camera Resolution	1382×512	1024×1024	can be programmed
Scenarios	many on roadway scenarios	many inside/outside scenarios	many, simulation on different conditions possible
Calibration sequence	is not provide, only one shoot with calibration patterns	rosbag with checkerboard 7x6	is not provided

Table 4.1: The main parameters of the analyzed datasets.

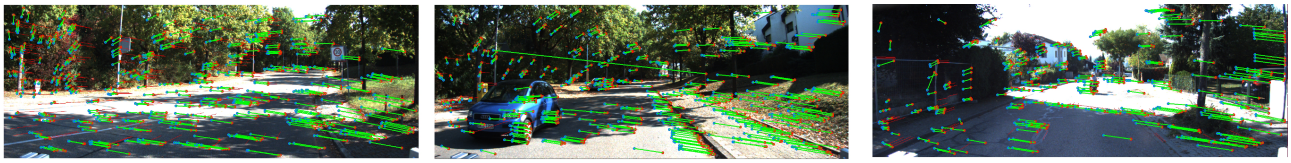


Figure 4.8: The recorded dataset by stereo cameras mounted on car. The distribution of scene (ground plane, sky) is very similar at each image frame. There is not a big differences in the rotation and high from ground compare to ground plane.

The existing datasets allow testing many computer vision applications. The all-traditional datasets consider each camera parameter as constant. This is the desired assumption by many computer vision applications. Some of their sequences are sufficient to test the camera calibration. This kind of dataset can provide good input data for recalibration testing. However, the decalibration phase does not occur, thus testing this kind of task does not make sense in those standard approaches. The system ensures that once perfect calibrated parameters are available for the whole sequence. The decalibration must appear in order to verify the sense of SCCM and OSCC. This kind of situation never happens in the mentioned dataset. It is the reason why there is a need for a new dataset to fully realize and test our approaches.

4.2.2 Specification of the perfect dataset for SCCM and OSCC

This section catalogs in three paragraphs all the most important aspects which dataset for SCCM and OSCC should provide. The most meaningful element of the dataset is the decalibration phase during the sequence. Therefore, at least two different calibrations must take place in an ideal dataset. The sequence should clearly separate them from each other. At the beginning, the sequence with a calibration pattern in the scene should appear, in order to realize the initial traditional calibration method. After, the system should realize the particular movement with the same calibration setup. In this period, the self-calibration methods can calculate the extrinsic parameters and confirm them with traditional method. These procedures do not require any calibration patter, but the motion of the camera into static environment. In the more advanced dataset, the system should monitor the

motion with the other sensor such as indoor GPS, GPS, system of lasers, etc. It allows verifying and confirming with high-level applications like SLAM or visual odometry the currently calculated camera parameters. This strategy can confirm that obtained parameters are precise.

We define the type of dataset's movements in two categories. The first, where the system realizes the motion mainly in a two-dimensional plane. The mobile robot platform with cameras can provide the example of such motion. It records the dataset while the pose of the camera respects constant position in the relation to the ground plane. The second type performs movement in three-dimensional space. The drone or camera mounted on the human provides this kind of dataset movement. This type of motion exposes to dynamic movements in different directions and at different angles. Both types of motion should be sufficient to realize the self-camera calibration procedures. However, the first group seems to be less complicated because some of extra limitation exists, the dynamic and rapid camera motion in up-down direction cannot appear.

The dataset must define precisely the decalibration moment at the time. To achieve it, we propose to use some characteristic and known objects in the scene, in order to verify and see the moment in the dataset. Controlled camera position change is advisable, but seems difficult to achieve in real situations. This process leads from the first calibration (1) with R1, T1, E1, F1 to the second calibration (2) with different parameters refined to R2, T2, E2 and F2. This work assumes that the intrinsic and distortion parameters are constant. However, in future, such group of dataset can realize the same scenarios with constant extrinsic parameters and change the intrinsic. Moreover, for the most complicated case, it is possible to change two groups of parameters. The movement after decalibration should be similar to these in the first phase, that the self-calibration can recalculate new parameters. We suggest realizing the motion in the closed loop. This strategy allows arriving to the starting point and it can be an additional verification step for high-level applications. The glsslam can provides the differences between the input and the output position. This can be a good indicator of the quality by parameters.

At the end of sequences, the dataset should have the calibration patter in the view in order to realize the final traditional calibration method. The initial and final sequences with calibration pattern should ensure that the sufficient number of frames with chessboard is available for the traditional calibration method to calculate the most accurate parameters.

At the end of sequences, the dataset should have the calibration patter in the view in order to realize the final traditional calibration method. The initial and final sequences with calibration pattern should ensure that the sufficient number of frames with chessboard is available for the traditional calibration method to calculate the most accurate parameters.

In accordance with the methodology of standard stereo camera calibration, the algorithm has to base on the overlapping views coming from the left and right camera flow. The dataset must guarantee that left and right images present the same part of scene at exactly the same time, in order to match the proper points. However, we present the different setups of cameras, to test, illustrated in Fig 4.9.

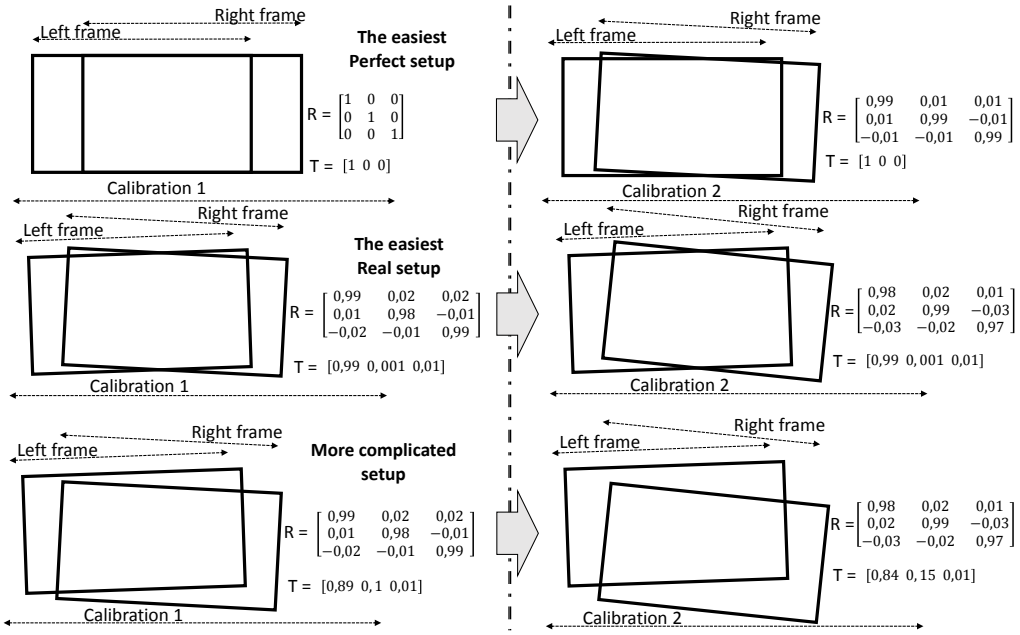
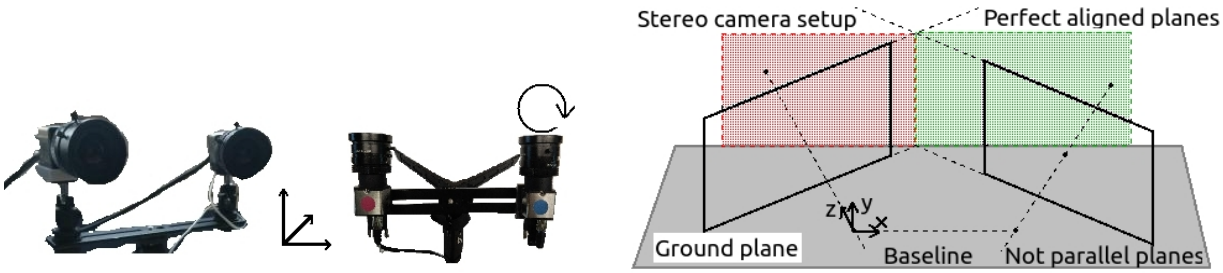


Figure 4.9: There different camera setups. The different systems can increase the various R and T parameters, thereby increasing the difficulty and differences to calculate.

The setup needs the perfect synchronization between two left and right flows. The dataset must record the camera stream with a certain speed. The higher FPS allows for smoother records of datasets, on the other hand the low FPS can lead to delays, lags and problems related to the continuity of frame data.

The dataset requires a high resolution of the input images, which is an equivalent of pixels in the one image. There are many standard display resolutions: 1024×768 (XGA/XVGA), 1280×1024 (SXGA), 1600×1200 (UXGA), etc. As a rule, a larger number of pixels allows for a more accurate representation of the reality. It affects in the better and more accurate POI detection. Therefore, we recommend using the maximum resolution. However, the system must consider that each pixel, depending on the format, takes up space. It significantly extends all the processes associated with the image analysis such as (corner detection, etc.). For this reason, an appropriate balance between the quality of the image and its size is necessary.

The dataset must meet certain additional constraints, in order to detect accurate and well-distributed points. The adequate illumination of the observed scene is a necessary condition for proper operation of the image analysis functions. The dataset should avoid the direct blindness from the light source, the reflection of rays and other special situation in order to create more useful and easy sequences. Another important point is to have a heterogeneous scenery. Where the each part of the image has clear and non-repetitive structures. These elements should be localized on the different distances from the camera. Finally, we recommend that the static scene should dominate the data set over moving objects in the frame, for more standard approaches. In order to extract the scale factor by the self-calibration methods additionally information is required. According to the section 3.5.5, the perfect dataset for



(a) Cameras mounted on the bar which can be directly attached to the helmet.

(b) The bar with camera mounted on the helmet.

Figure 4.10: Prototype of stereo camera used to realize custom dataset.

the OSCC, should have in the field of view several objects, which are suitable for easy recognition and have standardized sizes. Such objects can be traffic signs, doors, windows or people. We propose to use this type of scenery elements to extract the scale.

Dataset aspect	Conditions
Content	two synchronized overlapping stereo flows, high FPS, sufficient resolutions
Scene and environment	static scene, perfect lighting conditions, well distributed elements of scene, known objects in the scene
Calibration aspects	decalibration during sequences, all camera parameters for first and second calibration

Table 4.2: The table shows the summary of the specification for dataset dedicated for SCCM and OSCC.

4.2.3 Realization of custom dataset

Setup presentation

The Fig 4.10a presents the setup for dataset recording, where the stereo cameras are mounted in the metal bar, which can be attached directly to the helmet. For such system, the camera planes are not in the perfect parallel position to the ground due to the camera's setup settings. However, the distance between two cameras is contained only in one direction X-axis. During dataset recording, we hold the system in the hand so it is sensitive on motions in all directions. We compare this setup to a drone based system, which is very complicated due to irregular movement. The camera setup can move in the all degrees of freedom, in each: roll, pitch and yaw angular in the respect to the three directions.

Technical recording aspects

Images flow send the overlapping images straight to the PC, thanks to the u-eye ROS node [180]. This application allows setting various camera and dataset parameters such as exposure, frame rate, type of recording les (MONO8, RGB8, BAYER) and many others. Due the trigger technology build in cameras, it is possible to take already synchronized images from both cameras. The system sent the data with the time stamp, which enables to merge the corresponding frames. It is critical and necessary step for each image processing, in order to ensure that images present part of scene at the same time.

It allows correcting POI matches by low level processing functions. These data with certain parameters and specified frequency are received as an image messages, recorded in form of rosbag. During test, we play this as the movies for test purpose.

The resolution of images provided by stereo cameras is 1200 ×860 pixels. The weight of one standard frame with current resolution is around 3,5 MB. The system records the regular videos with 22 FPS. It is around 77 MB per second for each camera. It gives more than 4.5 GB each minutes for one camera. In the stereo camera setup, two must multiply this value. These values show how important and useful can be ow processing. Instead of copy and special mathematical operation on the image memory, the low-level image processing functions can realize computation on the stream of data at real time (see section 3.2.3).

For additional optimization requirements, we suggest using the Bayer format, which can reduce the size of the image. However, then system requires additional conversion to grey or to RGB the computer vision process.

Condition of recording aspects

We acquire the dataset outdoor close to the office. The sun is in the sky, so the lighting is not controlled but it is sufficient without any disturbance. The close surrounding environment has road, buildings and roadside elements. Fig 4.11 illustrates the view of camera, where for a few images the POI detection applied.

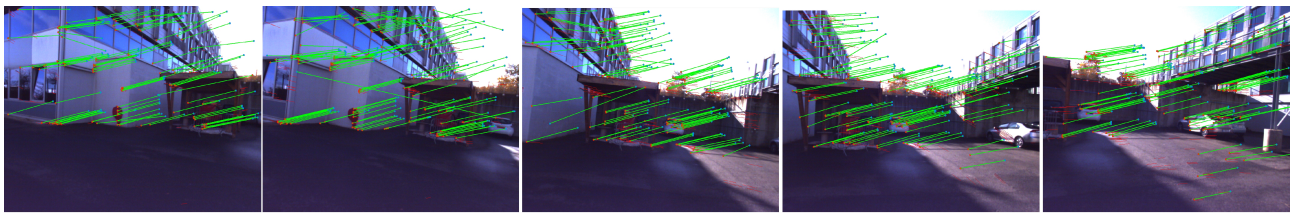


Figure 4.11: The recorded dataset by stereo cameras mounted on helmet. The points detection and stereo matching are applied. It is possible to see that, between image's frames position of camera is the same compared to the ground plane.

The recorded dataset is set on the 20 FPS in the RGB8 format. It allows considering less than 3 MB for one image. Depending on the sequence, the system records the different length of the rosbags. Each dataset consists of several small sequences. The next section explains this methodology. However, each sequence is no longer than a minute. We use to test the part of the dataset that has 30 s. The completely one registered sequence has less than 20×3×2×20=2.4 Gigabyte.

Custom dataset methodology

One registered dataset usually consists of three separate parts: two of them consist of an illuminated calibration standard for approximately 45s and one of approximately 30s of camera movement in a

static environment. The first one is the sequence at the beginning, where the system observes the chessboard. The second is the sequence at the end with exactly the same indication as the first.

The extrinsic camera parameters obtained by the OSCC compare to those calculated by offline traditional method, which is the reference procedure. For this reason, in the beginning and in the end of recording dataset, we present the pattern and record as the separate rosbag. Thanks to these sequences, system can find all necessary parameters: intrinsic, distortion and extrinsic.

In the custom-recorded dataset during around 30s, two different calibrations of the stereo camera setup exists. Moreover, the system perfectly knows the moment of decalibration in term of frame and time. The movement with the calibration 1 realizes during first 15 s of dataset. After that time, the position of the right camera is changed. It is due to the physical force, which touched camera. The images are continuously recorded during and 15s after decalibration. The second part of the dataset realizes the sequence with the calibration 2. The Fig 4.12 presents this strategy with the methodology and characteristic. During this sequence, the SCCM and OSCC can be tested and try to be proved, if approach from 3rd chapter works properly.

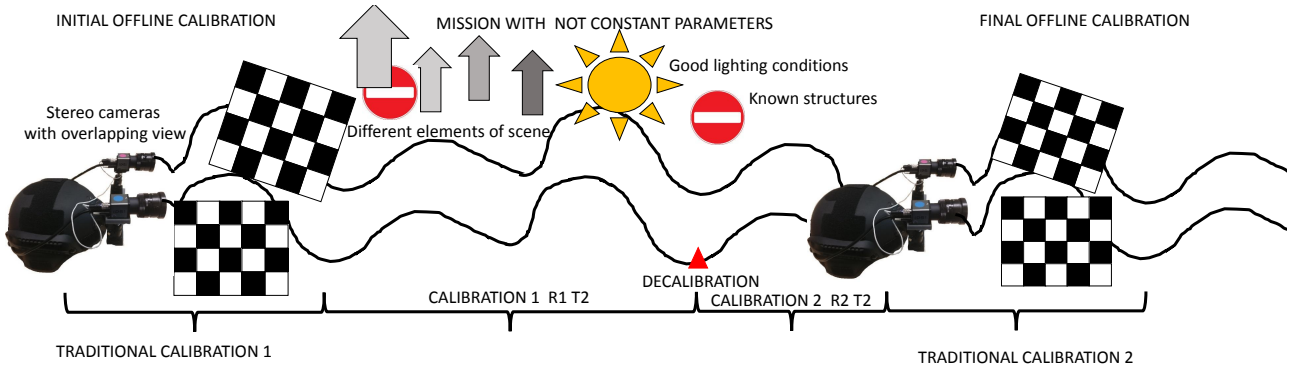


Figure 4.12: The specification of dataset for OSCC. At the beginning the traditional camera calibration (with pattern) is performed. Then during motion, one of camera is decalibrated. At the end of sequences, the calibration chessboard is shown again, in order to calculate second calibration of camera parameters.

Traditional calibration phase

The system must calculate the parameters during the recorded dataset. During the first phase, the system observes the calibration pattern during the sequence. The Fig 4.13 shows this process. The captured data from u-eye node fill the offline camera calibration node [179], which is based on OpenCV. We run this node with GUI presented in Fig 4.14. The Fig 4.15 shows the command to calibrate with several parameters related to calibration tool and image flow. The offline traditional calibration node provides intrinsic parameters of the left and right cameras, and the extrinsic parameters of the stereo pair. The R and T express the positon of right camera the left camera. This method computes the parameters during procedure. This methodology ensures that in the recorded rosbag, there is a

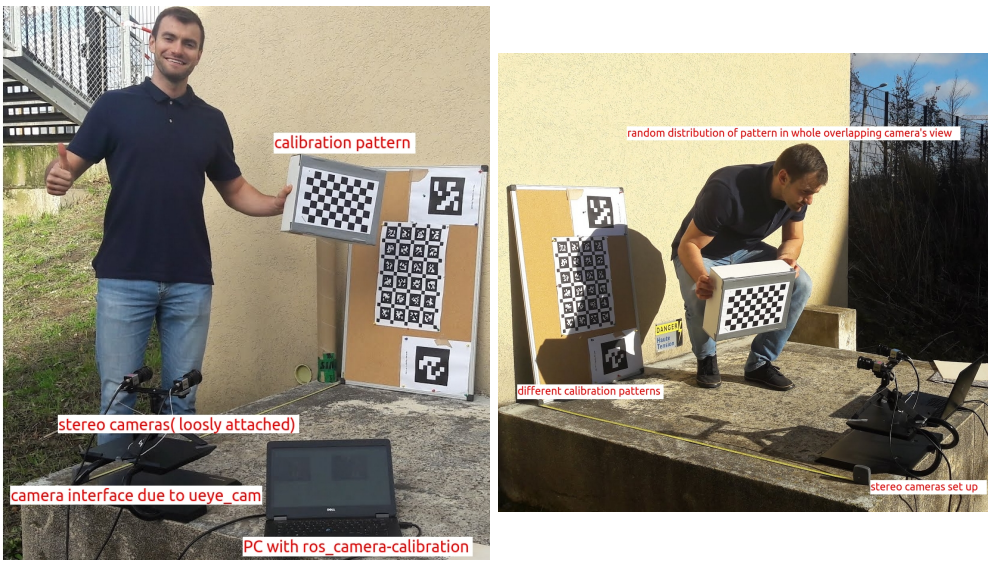


Figure 4.13: Dataset recording - the sequence for offline method is repeted on the beginning and on the end of one scenario.

(a) API collecting data from two images streams.

(b) API ready to calibrate parameters.

Figure 4.14: The GUI of ROS node used to traditional stereo camera calibration.

sufficient number of images of the calibration pattern to perform accurate calibration. The traditional calibration needs a sufficient number of frames with enough pattern variation to estimate the correct parameters (X, Y, size and skew, shown in Fig 4.14a). When the system has enough data to calibrate, the blue button calibration is ready to use, as presents Fig 4.14b. This node provides the R and T parameters and not the E matrix, which is required to compare selected approaches. It is possible to obtain the E from the extrinsic parameters. There are four solutions, but it is difficult to guarantee which one is the best. $E = R[t]_x$ or $E = -(R[t]_x)$ or $E = (R[t]_x)'$ or $E = -(R[t]_x)'$.

The offline camera calibration based on the Matlab application can calculate the same parameters plus E and F matrices (see section 3.4.1). However, this procedure does not inform the system, if there is already a sufficient number of images to realize a proper calibration during the recorded sequence. We use two different camera calibration methods during this dataset. It can ensure that the calculated extrinsic camera parameters are precise. Moreover, this approach provides F and E matrices, required by OSCC.

```
rosrun camera_calibration cameracalibrator.py --approximate 0.5 --size 9x6 --square 0.25 right:=/right/image_raw left:=/left/image_raw
    right_camera:=/right left_camera:=/left
```

Figure 4.15: The GUI of ROS node used to traditional stereo camera calibration. First parameter after size informs about chessboard size. Next the square represents a real size in cm of one black square. Following, right and left camera calls the name of image stream from ueye_ cam node.

4.3 Experiments on the Stereo Camera Calibration Monitoring -SCCM

This section presents characteristic and methodology of SCCM based on the approach from the section 3.5.3. This process monitors current quality of stereo camera calibration in the sense described in the section 3.5.4. The several different policies are used to de ne if current extrinsic camera parameters are still valid or the camera's position is changed.

The standard scenario is related to dataset from section 4.2.3 with two phases. During the first stage, a scenario is calibrated, and it uses initial parameters found by offline traditional method. The second phase in the middle of dataset starts with decalibration. The initial extrinsic parameters are not correct. The main task of SCCM is to detect when it happens. The system must detect this moment as soon as possible because the wrong extrinsic camera parameters can lead to many errors from high-level application.

4.3.1 Input data used to SCCM

The section 4.3.3 proposes and explains two different groups of policies in order to realize SCCM. The first group uses the amount of POI in a specific category. The second group of SCCM tactic depends on the value of epipolar error in selected group. The final decision, if the system needs calibration is based on a combination of those policies.

At this point, it is mandatory to remind the input POI after the filtering tracker strategy from the section 3.5.2. The whole detected and stereo matched points must satisfy two important conditions to be considered as the inlier or outlier. It has to be temporally (at least once before left and previous left) and the stereo (left - right) track. If it is not, then points move to “not considered" group.

Amount of POI in specific category

Fig 4.16 shows the number of points in each group. The grey bars on the background plot all stereo matched points from the current frame. For almost entire sequences, the value is quite stable from frame to frame and has between 100 and 200 points. Small irregularities occur in the frame range of 20th-29th, 221th-230th, 253th-260th and 275th-285th. During these frames, there are not enough points in the system. It is because imperfections of dataset (poor or too strong illumination, homogeneity of the image, etc.) Then, the green color represents the amount of inliers. Parameter is stable until 157th frame, when it starts dropping. The current number of outliers are marked with the red line, which

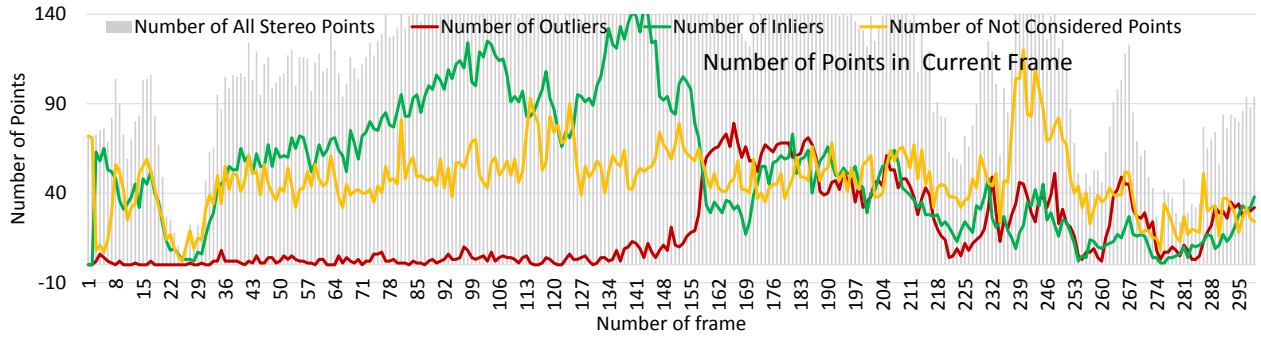


Figure 4.16: The plot shows the number of points on each group from every frame of the analyzed sequence.

significantly starts to increase when the number of inliers decreases, around 153th frame.

The last group of points does not satisfy the tracker condition (the point is not tracked at least on the frame before). In Fig 4.16, the yellow color plots the "not considered" points. We can see that the amount of such points is stable during 300 frames.

Value of epipolar error in specific category

The second group of policy uses the sum and ratios between the epipolar error in specific categories, in order to activate the triggers. The system must exceed the particular higher number relation between the initial/global average value and the local/current average of epipolar error calculated from stereo points. It calculates the epipolar error for each of the considered stereo pair, due to methodology presented in the section 3.5.1. The whole calibration pipeline uses the same conditions for tracker in this policy's group. To be stereo considered, the point has to be temporally (at least once before left and previous left) and the stereo (left - right) track.

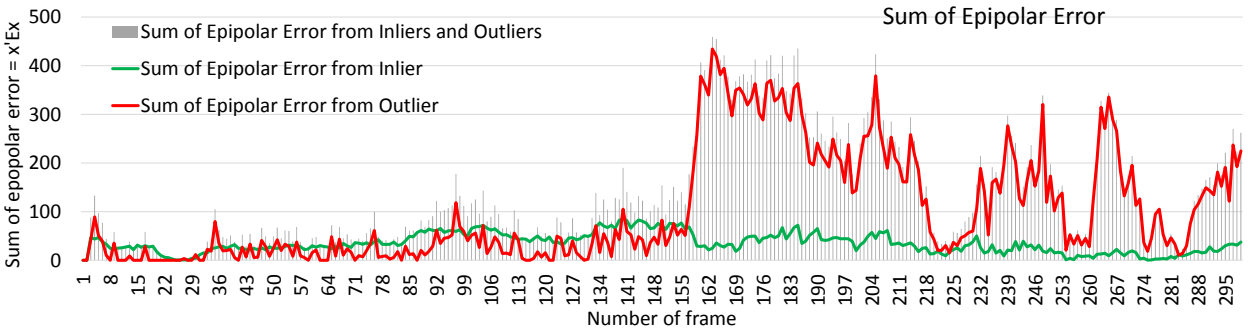


Figure 4.17: The plot shows the sum of epipolar error on inliers and outliers group from every frame of the analyzed sequence.

The Fig 4.17 shows the sum of epipolar error in each group: inliers and outliers. Moreover, the grey color bar form in the background plots the sum calculated from all stereo matched points at current frame. For almost entire first half of the sequences the value is quite stable and seldom exceeds 100.

For the second half of the dataset, the sum is a few times higher. Then, the green color represents the sum of error calculated from inliers. Parameter is quite stable for whole sequences. It is because the epipolar error defines the point category such as inlier. If an error is higher than the threshold value, the outlier is the point class. The current outliers are marked with the red line. The sum of epipolar error from outliers significantly starts to increase after 155th frame. An irregularity of epipolar sum for outliers occur in the frame range of 20th-29th, 218th-232th, 253th-260th and 275th-285th. During these frames, there are not enough points in the system, the dataset imperfections caused it.

The two diagrams shown in this section allow determining whether the current parameters are up to date. Based on these charts, the section 4.3.3 proposes the different techniques to detect the right moment.

4.3.2 Average value and trigger methodology

It is difficult task to determine the standard value of each monitored parameter from first or second group of policies. Because it is strongly dependent on dataset and low-level functions. For example, the various numbers of points detected on each image's frame significantly depend to particular detector or tracker, its method, internal parameters, specific condition etc.

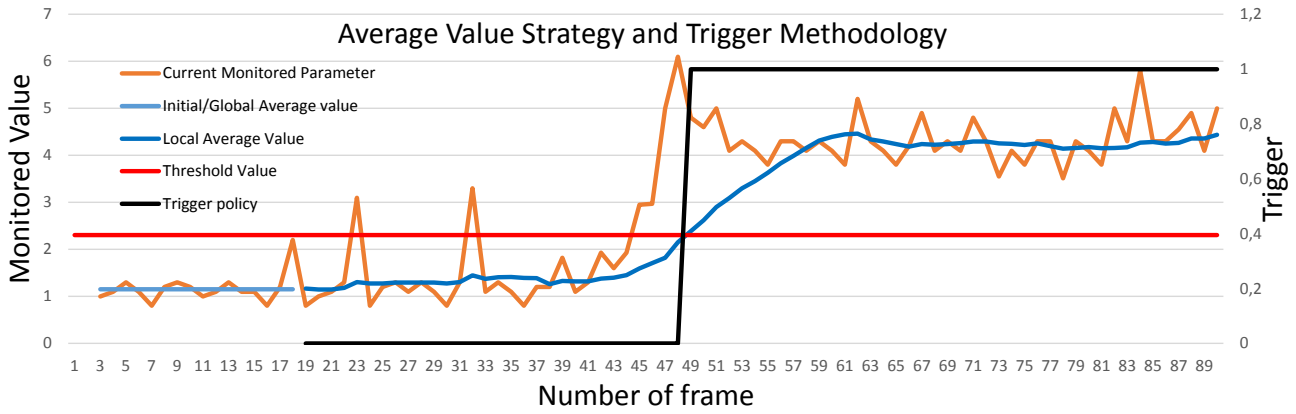


Figure 4.18: The plot explain the average construction technique and trigger simulation. It shows that this technology is able to eliminate a false data.

To minimize these criteria effects, at the beginning of each scenario, every monitored parameter is set as the initial or global (names are used alternatively) average value, it is based on the first 15 frames starting from the second image. The Fig 4.18 shows the initial average building process in a light blue line. The number of frames used to calculate an average is found by experiments and recording parameter - FPS. Thanks to this proposition, monitored value is independent from the current dataset.

The different SCCM policy requires threshold value, in order to activate a trigger. This is presented in the red line, its value is determined experimentally for each monitoring tactic. In the Fig 4.18, we can observe that system calculates the 200

For each following plot in this section, the orange line or the orange bar represent the currently

monitored parameter. We use these different formats in order to increase readability of plots. In the Fig 4.18, it intersects threshold line at 21th, 30th and 42th frame. However, due to not perfect and not a deterministic condition of the dataset, it can happen that measured values fluctuate very intensive. We propose the local average technique for SCCM, in order to avoid it, represented in the blue line. The system construct it from 15 frames, tests for this value passed.

The alarm triggers, when and only when, local average value (not monitored parameter) is higher than the threshold. This methodology eliminates false alarm at the 21th and 30th frame. It triggers policy at 46th frame. This may result in a small delay in relation to the real changes of parameters. The image's frames are taken at a speed of about 20 FPS (for this tested dataset), so delay is never longer than 1 second. Additionally, positive aspects that allow to avoid false alarms are higher than this small delay. This policy can detect a second phase of the scenario.

For each plot's figures in the following section, the X-axis represents the frame number. The system use the value interchangeably with time domain. However, in order to do it, it must recalculate. We record the dataset sequence with a speed of 20 FPS, so 1 FPS is 0.05 s.

In each of following plots, there is a policy trigger. If, it represents value zero, the current extrinsic parameters satisfy the particular conditions. When the value jumps to high value (1 or other positive value like 0.9. 1.2 etc. sometimes uses to increase a readability), it symbolizes that the tested monitored tactic detects decalibration and needs new extrinsic camera parameters.

The next section presents each policy with alarm (trigger) criterion. We present them on the appropriate sequences, recorded based on the meteorology from the section 4.2. To validate and show that this technique works properly, we test methodology on the KITTI and the inversion sequence of custom dataset. Finally, we comment the obtained results in the last section of this chapter.

4.3.3 SCCM policy

Policies rely on the number of points

We present the first group of policy, which bases on the ratios between the numbers of points in specific categories. The system requires exceed the particular relation between the initial/global average value and the local/current average number of points, in order to activate the triggers, as presented in the section 4.3.2. The precise definition of each point's category is available in the section 3.3.1.

The first presented policy is based on the ratio between number of points satisfy and do not fulfill the epipolar geometry. Fig 4.19 shows the decision making process based on this criterion. The orange bar presents the current monitored value. The presented ratio is very unstable and varies a lot in the range between 10th and 150th frame but it does not tend to zero in that window. The number of points causes a large variation. If there is only one current outliers (bad point in the considered group) in relation to one hundred inliers (good points),then the ratio is several times higher than when there is not one but several outliers (the same value divided by a higher number). The unfavorable relation begins when the amount of outliers coincides with the number of inliers. In such case, the ratio tends to reach 1, or less. It is clearly visible for orange bars after the 157th frame that the monitored parameter certainly drops and stabilize, and then it is close to 0.

The system composes the global average in the light blue line. Then, based on this the threshold (red color line) value is calculated. In this policy, it is 60 % of initial value. We select this value of the experiment. We apply the local average ratio technique, in order to avoid deviations of current measured parameters. The system compares the dark blue line with a threshold. In Fig 4.19, the local average intersects the threshold twice, at frame number 25th and 150th. It implies and turns on the ratio policy trigger, which presented in the black line in the plot. The ratio between 20th-29th frames tends to be small due to a small number of points. It is the false alarm, which the other policies must eliminate.

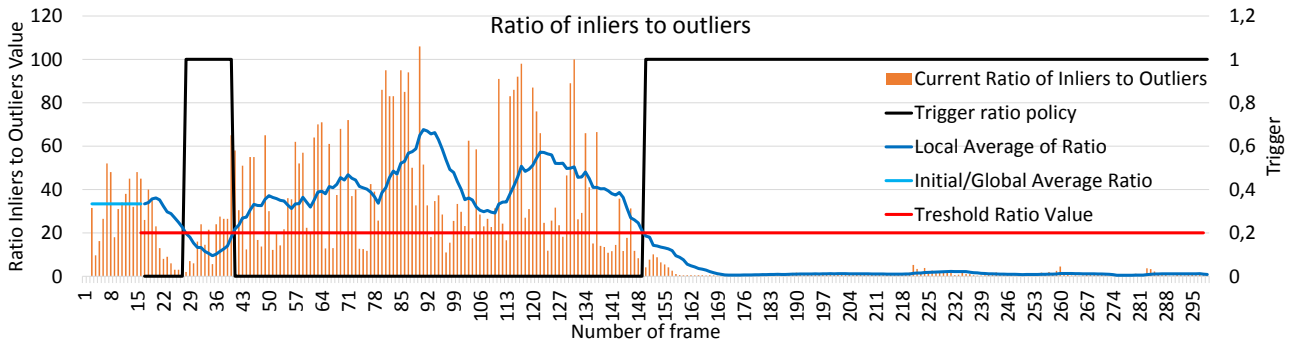


Figure 4.19: 1st SCCM policy is based on ratio between the amount of the inliers to outliers.

The next policy rests upon on the ration between numbers of inliers to all stereo points. In this policy, the threshold value is set on the 0.75% of global initial average. The system set this value because

during the standard sequences in the dataset, the number of points satisfying epipolar geometry should increase. If and only if the movements of the camera is smooth and camera's pose does not change. However, the drop of inliers can happen, for example: when some object obscures in the camera view or camera parameters changed. Fig 4.20 shows that in the first half of the test, the value of a measured parameter systematically arises. Then after 153th frame the current ratio of inliers of all stereo points decrease. The local average follows the same trend and activate a trigger at 161th frame. Moreover, there is similar false alarm between 29th and 35th frame as in the previous policy.

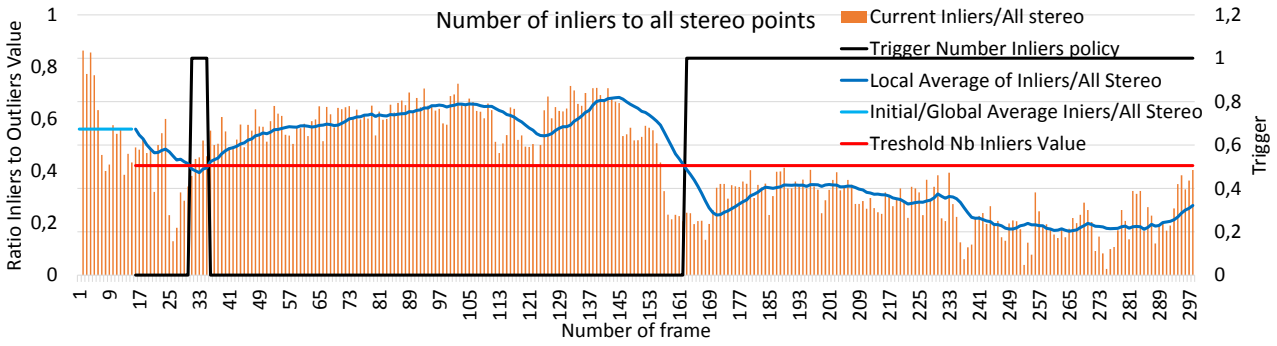


Figure 4.20: 2st SCCM policy is based on ratio of the inliers to all stereo points.

The last ratio policy in this category shows the number outliers to all stereo points. The Fig 4.21 illustrates monitored parameters and the decision making process with a similar methodology. The threshold value is set on the 5 times higher number than the initial average. This value is set high because, in the beginning of standard dataset the number of outliers should be very small. The system starts with perfect extrinsic parameters from offline method. Test shows that proposed value is well suited. For the first phase of sequences outliers ratio tends to zero, then during second phase when camera must recalibrate, the value is very high. In this policy, the system does not activate the false alarm. The trigger is turned on only once at 155th frame.

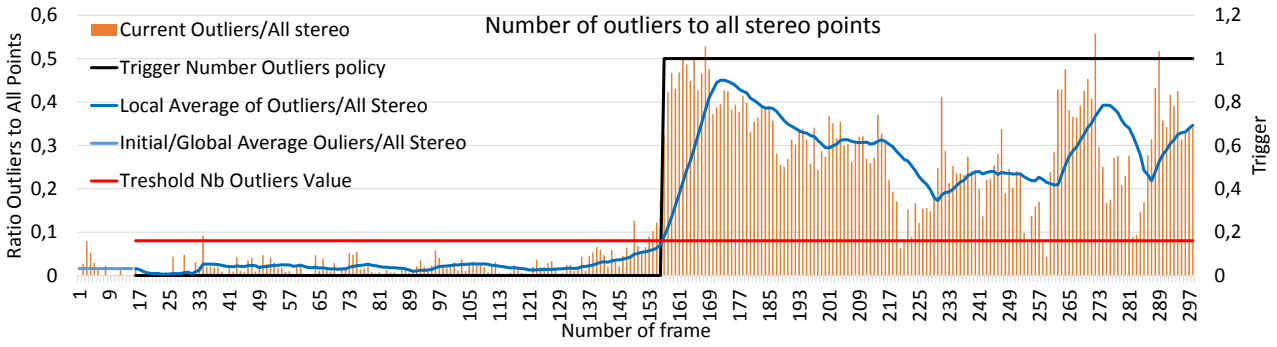


Figure 4.21: 3st SCCM policy is based on ratio of the outliers to all stereo points.

Policies rely on the epipolar error

The Fig 4.22 illustrates the decision making process according to median of epipolar error from all stereo considered points. As for each plot, the orange bars provide a value of monitored parameter from the current frame. The system calculates the local average median of error (dark blue color line) from last 15 frames.

We construct the threshold on 3 $\times$ higher value than the initial/global average. In the Fig 4.22, the value crosses the threshold only once at 157th frame. Until this moment, the current and average median of error is low and stable. Then, it starts significantly rising more than 100%. The high value stabilizes and goes until the end of the sequences. The trigger turns on in the 157th frame. When the extrinsic parameters are perfect, the error from perfect points should tend to zero. The median of error from precise points should be naturally small. For the first half of sequences, it does not exceed 1. During the second half, the median starts to vary and hits the values higher than 4 $\times$ of previous value.

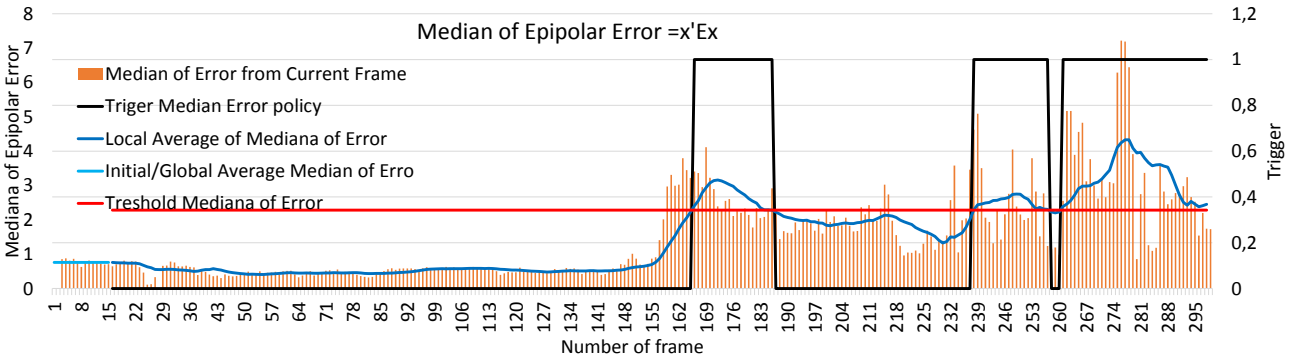


Figure 4.22: 1st SCCM policy is based on median of epipolar error of all points from current frame.

Another SCCM policy is based on the ratio sum of epipolar error from inliers to outliers. This monitored parameter does not rely on local average value, but it is constructed on the current epipolar error sum. The Fig 4.23 presents decision-making process with monitored parameters. The threshold value is set on the 20% of initial average. Two halves of the dataset are well visible due to the value of the ratio. The first half, it is more than 1, when the second half it tends to 0.

The Fig 4.24 presents the last SCCM policy in the second group of decision. The measured parameter describes the whole epipolar error sum of considered outliers points. The same average methodology, the system use to turn trigger. It computes the local average base on the last 15 frames. If, it is higher than 5 $\times$ the initial average, it turns the trigger. In the Fig 4.24, the trigger turns on three times, at the frame 158th, 239th and 295th. It is well visible that during the second part of the sequence monitored parameter is a few times higher.

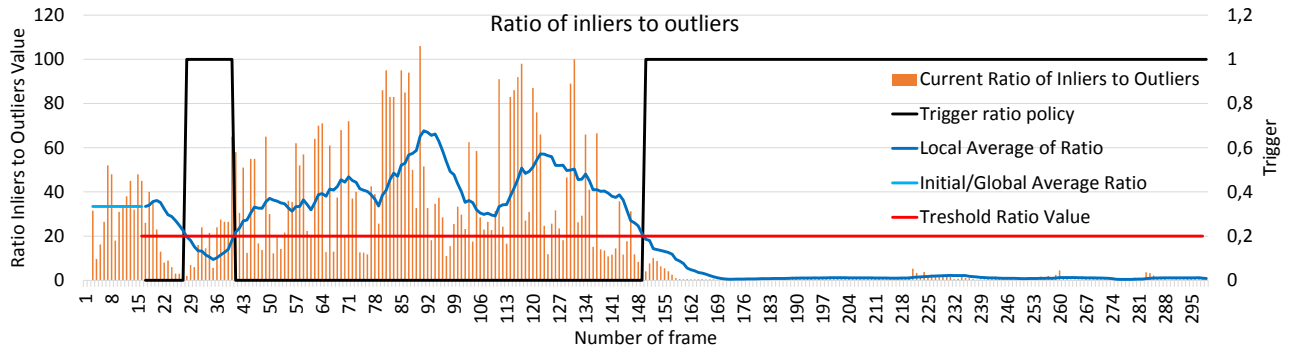


Figure 4.23: 2st SCCM policy is based on the ratio of sum of epipolar error inliers and outliers (from current frame).

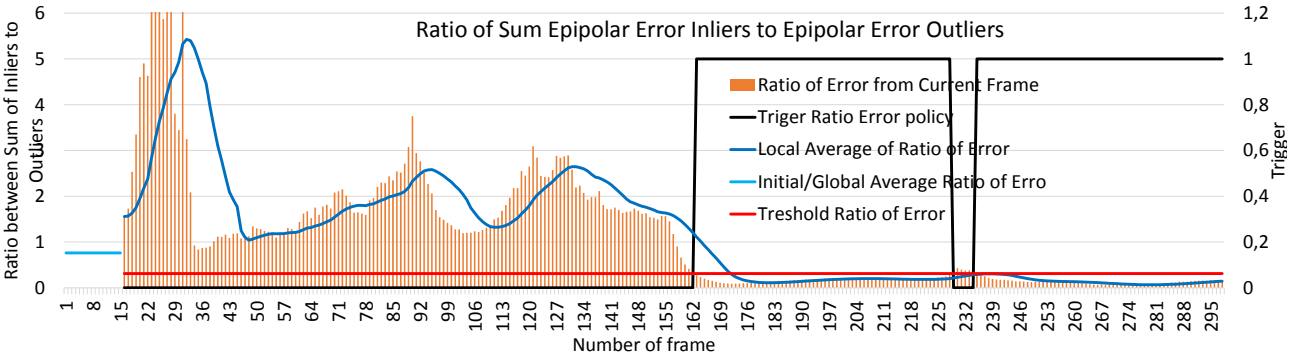


Figure 4.24: 3st SCCM policy is based on sum of the whole epipolar error calculated only from points considered as outliers.

4.3.4 Combination of SCCM policies

Each of proposed SCCM policies can give a false alarm despite the methodology used. If there is not enough data (considered points with low requirements - tracker parameters), it is impossible to take up the correct decision. Such a situation can occur when there is insufficient light quality of the scene or the camera's view obscure, etc. Then, there is not enough points in the system for a certain period. The SCCM should have the possibility to understand this kind of scenario. During this type of sequences, it is unlikely to decide whether the current parameters are correct or not. If such a period is long, there is a high probability that the environmental conditions are not sufficient to analyze whether the current calibration is precise enough.

In the following section, Fig 4.25 presents the methodology to realize this type of problem. The system monitors the ratio between numbers of not considered points to the whole stereo number points, in order to decide when policies cannot take a decision. The average methodology presents the three periods in the whole dataset, 25th- 37th, 238th-267th and 283th-289th, where the ratio of not considered points is too high compare to the initial global value. The whole system during these frames cannot decide if current parameters are good or bad.

The section 4.3.3 presents 6 different triggers, which are responsible for SCCM policies. Each three

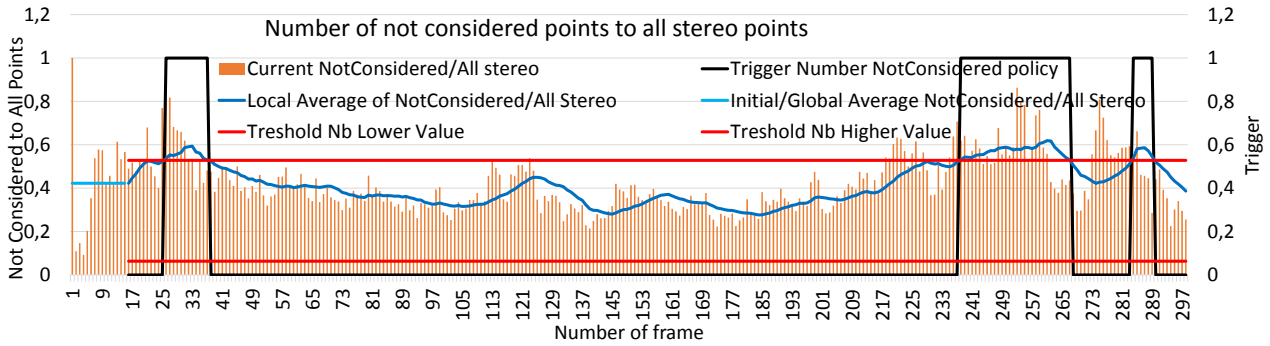


Figure 4.25: Not considered points SCCM policy is based on number of the not considered points to all stereo points.

of them create a one separate decision. First is based on the logical sum of the all three decisions from ratio parameters, it is presented in the Fig 4.26 in red color. The second relies on the similar conclusion, according to the value of epipolar error in the Fig 4.27. Both SCCM decisions at the same time must be positive, in order to pass to next decision step. The third condition is based on the not considered points. It allows knowing when the system cannot decide whether it needs or does not new extrinsic parameters. Its decision from the Fig 4.21 must be equal 0, if it is not, the camera monitoring policy provides the -1 which refer a lack of opportunities to take a calibration decision without having regard to the other conditions. If and only if these three conditions are favorably satisfied, the system receives information that needs a new calibration. If the global trigger is set once, system must be recalibrate. Even, if after a few frames, one or more policies are deactivated. Once, the high-level applications received a signal, those are waiting for new extrinsic camera parameters. The Fig 4.28 illustrates the final decisions based on the three conditions. The list 4.1 presents the three decisions with 7 different triggers.

- Condition 1 - decision and policies are based on amount of points
  - ratio between average inliers to average outliers
  - ratio between local average inliers to all stereo points from current frame $Average(nb_{inliers})/nb_{stereo}$
  - ratio between local average outliers to all stereo points from current frame $Average(nb_{outliers})/nb_{stereo}$
- Condition 2 - decision and policies are based on value of epipolar error
  - median of error of pair of points based on epipolar geometry $x'_R E x_L = e_{rror}$
  - sum of error, sum of error inliers and outliers $\sum(e_{rror})$
  - average error of one point, inliers and outliers$(\sum(e_{rror}))/nb_{inlier}$
- Condition 3 - decision and policy is based on sufficient number of point
  - ratio between local average nor considered points to all stereo points from current frame

List 4.1: List of all conditions with each trigger.

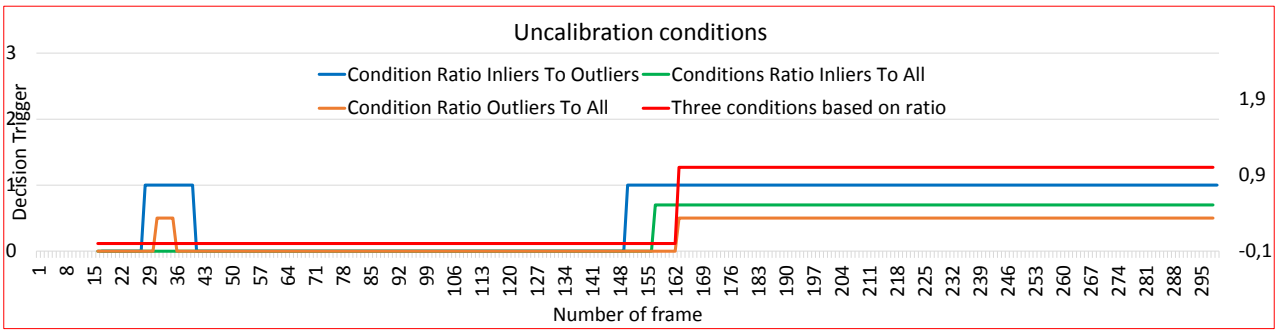


Figure 4.26: First decision based on three different ratio policies.

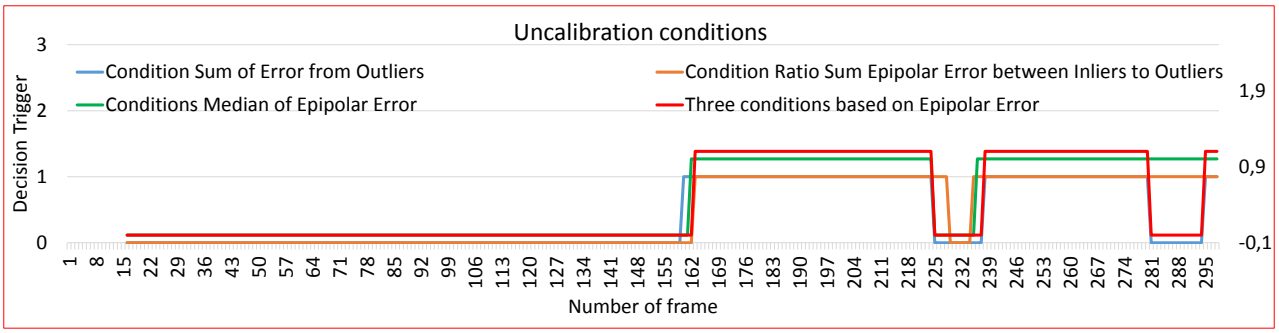


Figure 4.27: First decision based on three different epipolar error policies.

Custom dataset

The section 4.3.3 presents each policy on the custom-recorded dataset. We use the sequence where position of camera change in order to visualize and explain the selected methodology. The whole sequence has around 15 s. We record it with the speed of 20 FPS. The first phase is finished, in approximately half of this dataset. The second phase starts with the change of the right camera position. On each of Fig 4.19 4.20 4.21 4.22 4.23 and 4.24, the moment when the extrinsic camera parameter change, it is well visible and detectable thanks to the presented methodology. We show the two main decisions together on the top of Fig 4.28. At the bottom, there is a final decision, which composed from three presented conditions. If the final decision presents zero, it means that, the current parameters are correct. If it is -1, the system cannot take a decision, and when it is 1, it requires new parameters.

In this dataset, there are three different periods with not sufficient number of points (inliers and outliers). That is why, in that time some of the proposed policies triggered false alarms. On the other hand, the other suggested policies choose a correct decision during the same time. To take a final positive decision for calibration, there is need of policy consensus.

The system activates all proposed policies at 162th frame. It has a small delay compared to real moment when camera pose changed. Moment, when the right camera moved (touched) happened around the 157th-161th frame. The moment is clearly visible and can be can be detected by the

proposed strategy. Even with a not perfect dataset, the used technique works well and fast.

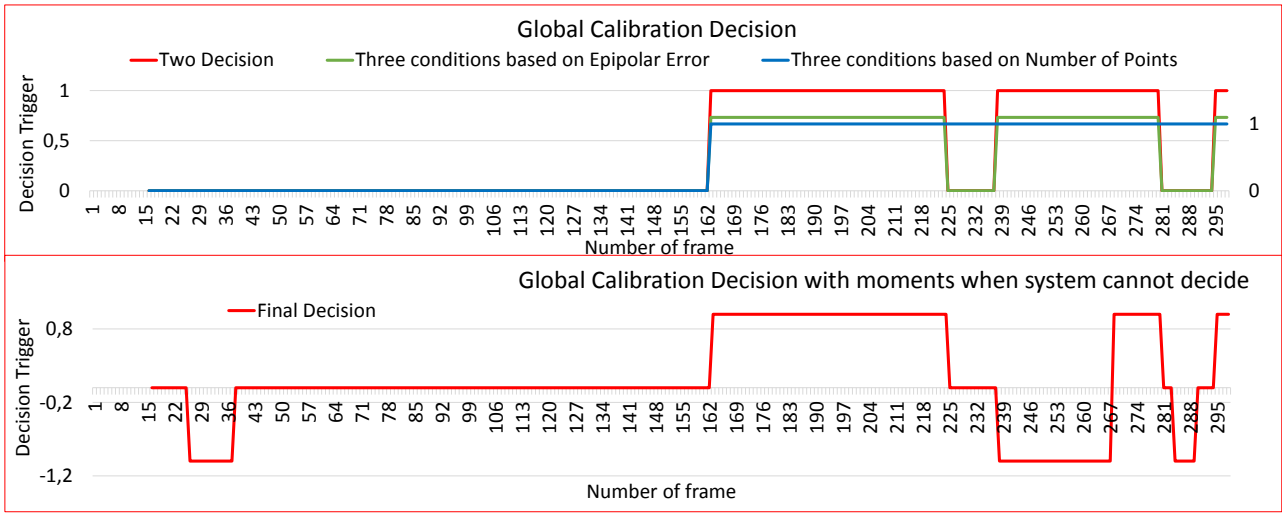


Figure 4.28: Final triggers from each condition for custom dataset, which are based on previously presented policies.

Custom dataset inversion

We realize the second validation of this methodology on the inverted dataset from previous test. The system finds the extrinsic camera parameters on the end of the sequence. Then, we apply them as the initial parameters. For this reason, the system during the first phase has wrong T, R and E. Then, for the second half of the sequence, when the camera is touched, the extrinsic parameters are correct. The Fig 4.29 illustrates the high epipolar error in the first half of the dataset, after it tends to zero. The Fig 4.30 shows that the number of outliers in the beginning is higher than at the end of the dataset. From both illustrations, it can be read that the parameters are changed somewhere between frames 157th-162th.

The system applies all policies to realize SCCM in this dataset. The Fig 4.31 presents only the median of epipolar error policy. Despite the fact that on the two graphs a decalibration moment is visible, the proposed SCCM technique does not work. Because the system must have ideal input extrinsic parameters at the beginning of the mission. That, it can verify the parameters later on. The last presented Fig 4.32 shows the decision making process based on all conditions. The system thinks that the system does not need to recalibrate although it must. Unfortunately, this is due to poor input parameters.

KITTI dataset

We preform the last test in this section on the KITTI dataset. The extrinsic parameters of the calibration are very precise and stable during whole sequences. Anyway, we use this dataset to test,

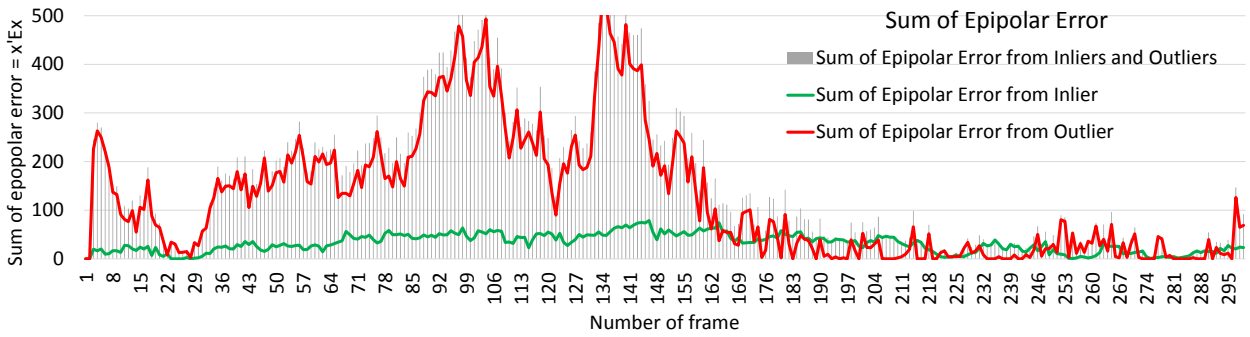


Figure 4.29: The plot shows the sum of epipolar error on inliers and outliers group from every frame of the analyzed sequence (inverted dataset).

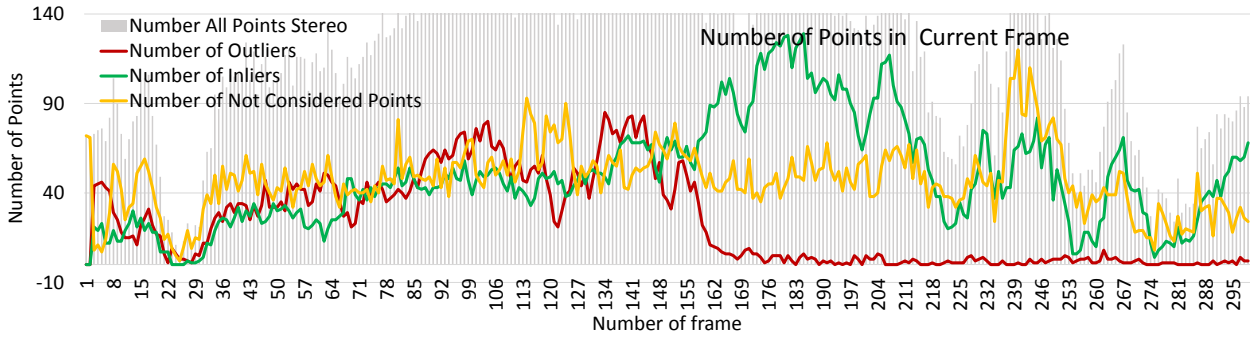


Figure 4.30: The plot shows the number of points on each group from every frame of the analyzed sequence (inverted dataset).

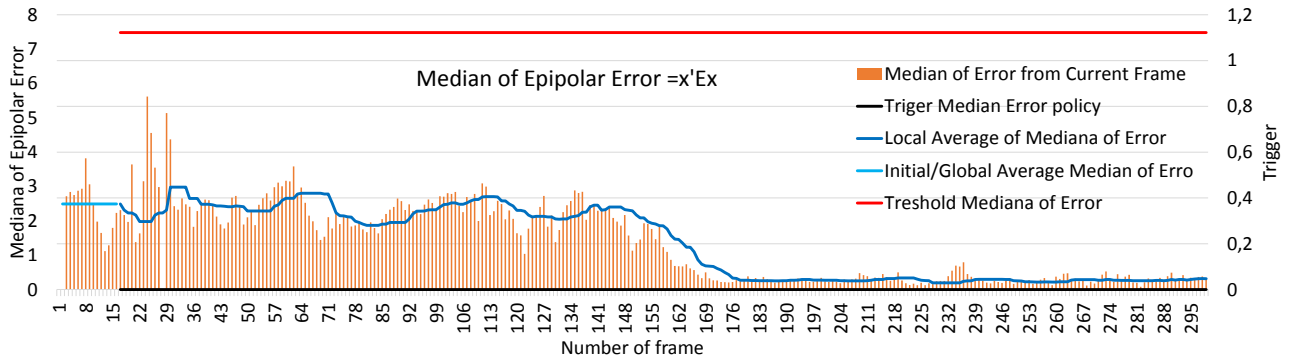


Figure 4.31: SCCM policy presents median of epipolar error of all points from current frame (inverted dataset).

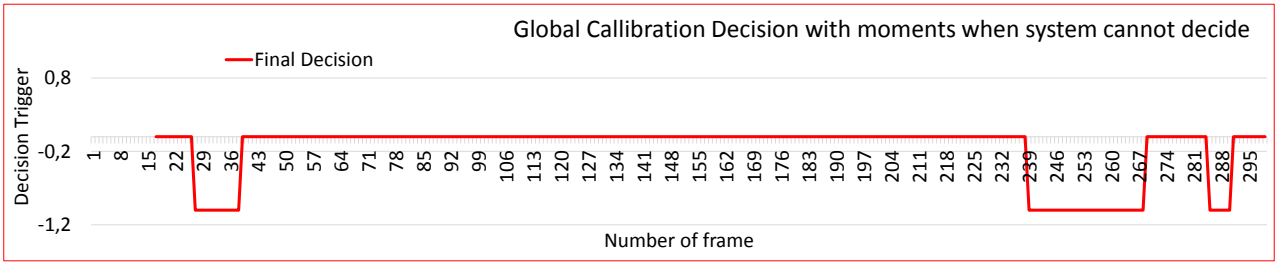


Figure 4.32: Parameters of SCCM, which is based on the parameters from the end (inverted dataset).

because it has a different numbers of points in the image. If proposed policies are not depended from the different dataset's parameters, it can show false information. In a previous test (custom dataset), tracker detects usually between 100 to 200 points, for KITTI there is more than 700 points, what it is shown in Fig 4.33. The Fig 4.34 illustrates the epipolar error of each point group. This section presents only two policies, but the final tested system applies all policies to take a final decision. The Fig 4.35 presents the one trigger, which base on the number of point. The Fig 4.36 presents the second that relies on epipolar geometry. We plot the final decision in the last graph in this section 4.37. As expected, the SCCM shows that the parameters of the camera are correct throughout the entire length of the sequences. We carry on the tests on two different scenarios provided by KITTI. In addition, it is necessary to pay attention to a much smaller scale of error, which comes from dataset, where ideal parameters and very high quality of images exist. The system should realize the point detection with greater precision compared to custom dataset.

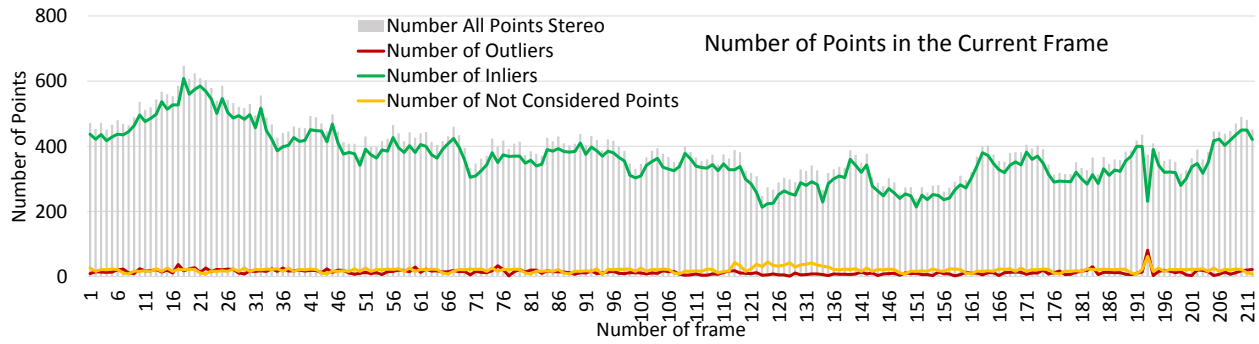


Figure 4.33: The plot shows the number of points from each group, all stereo, inliers, outliers and not considered from every frame of the analyzed sequence (KITTI dataset).

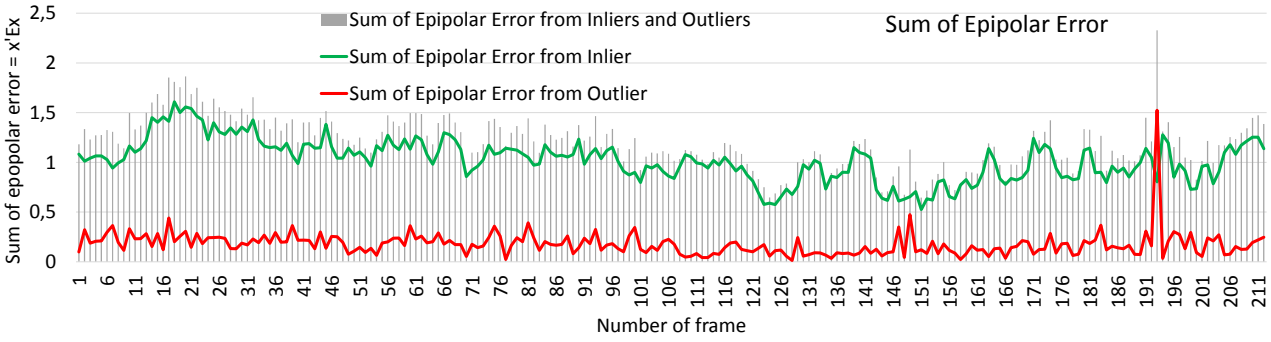


Figure 4.34: The plot shows the sum of epipolar error on inliers and outliers group from every frame of the analyzed sequence (KITTI dataset).

4.3.5 Conclusion

This part of chapter explains the characterization of SCCM. The section presents two groups with six standard monitoring policies and their description. We design the techniques to be independent from

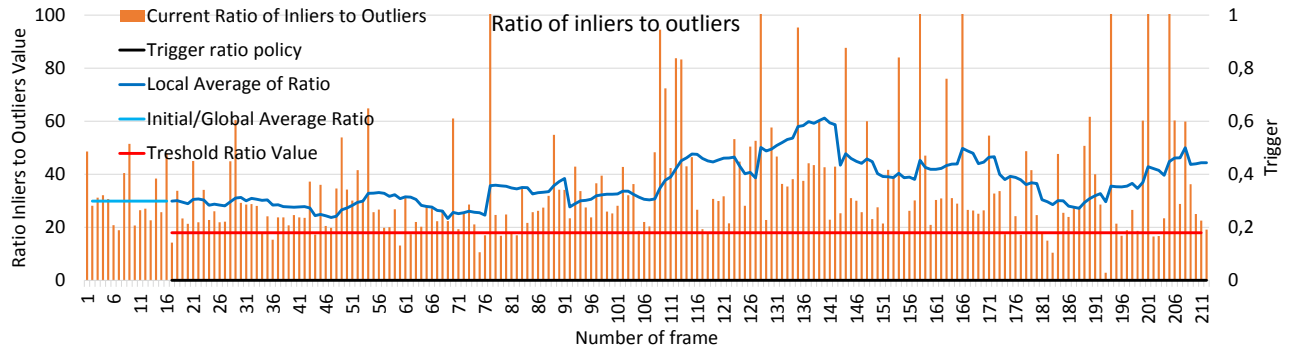


Figure 4.35: Ratio between inliers and outliers - one of the SCCM, which is based on the number of points, realized on KITTI dataset.

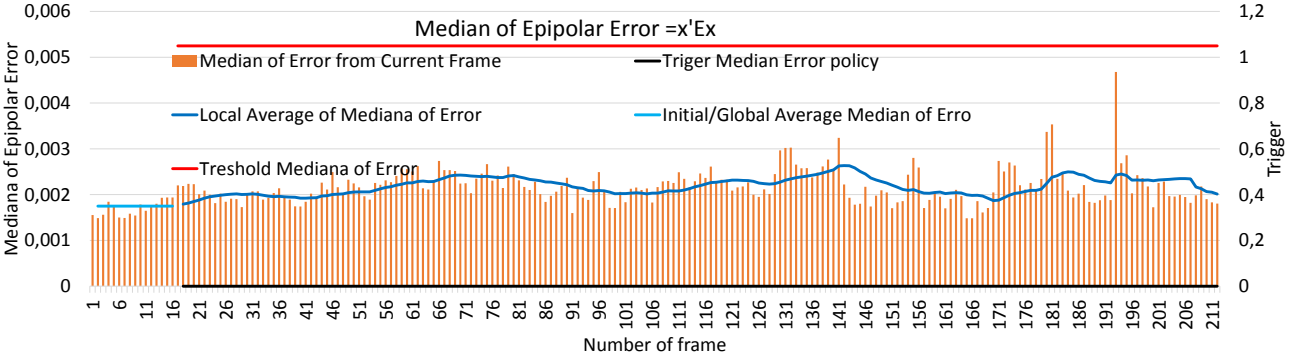


Figure 4.36: Median of epipolar error measurement - one of the SCCM policy, which is based on the epipolar error, realized on KITTI dataset.

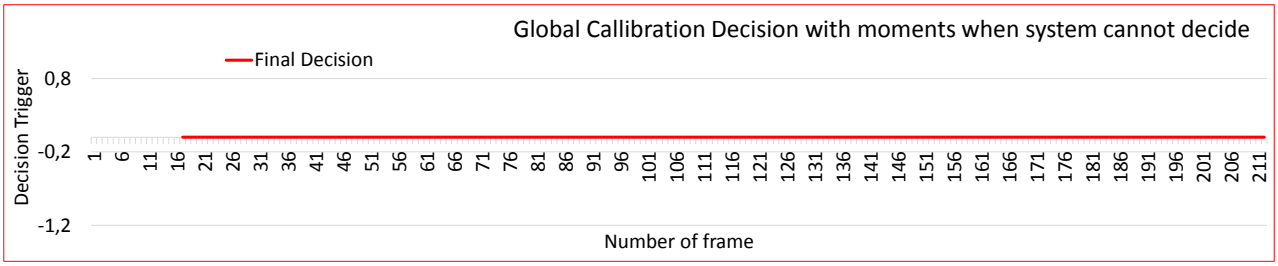
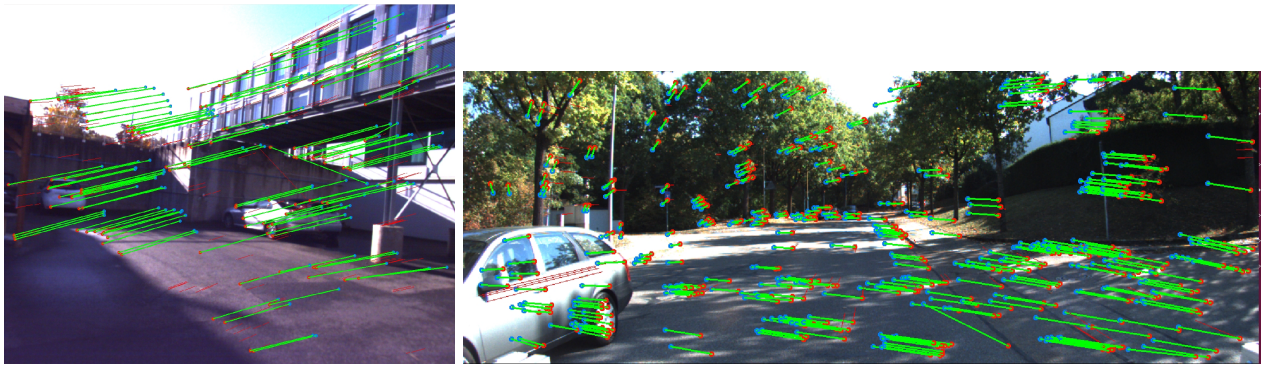


Figure 4.37: The final decision of SCCM realized on KITTI dataset.

the dataset. However, it requires the precise extrinsic input parameters. We describe and test each of policy on one particular example in this section. Moreover, we run the same tests on the KITTI sequence. This section presents and discuss the obtained results. In addition, we present the technique that limits the level of decision-making, it runs when in the system is the insufficient amount of data.

We perform the first test on the dataset recorded on the methodology from the section 4.2. It has a two phase with different camera's position. Unfortunately, the quality of dataset is not perfect. The points are not equally distributed over each part of the image. Moreover, they are not in different depth of the image. The Fig 4.38a illustrates the example of one frame from the dataset. However, the proposed monitored methodology is able to detect correct moment, when camera's pose changed.

We test the same dataset but in the reverse order during second test. The system confirms that the monitored indicators allow detecting the same moment when parameters are changed. However, the



(a) Custom dataset example. (b) KITTI dataset example.

Figure 4.38: Statistic frame with points detected from each of dataset tested.

calibration trigger does not work correctly, because the methodology requires the ideal input extrinsic parameters. The recorded dataset does not fulfill this condition. Therefore, we realized the test as we expected.

The Fig 4.38 shows the last tested dataset which one frame has perfect distribution of points, good lighting, and many structures on many different depths. This dataset is not prepared for OSCC test, because it contain only one phase, where camera are perfectly calibrated, due to the same initial parameters. SCCM technique shows the perfect stability of each parameter. The proposed policies do not detect any changes, what is an expected result.

A technique, which, due to the limited number of points in the system, prevents system from making a decision about calibration, works correctly. We run the same policy on each dataset. For first tests, in the situation when the number of stereo points clearly drops, the technique blocks the decision making process.

There is need to test proposed methodology on higher number of different dataset, to decide if proposed strategy works for various setups. So far, it works well on both datasets. It is not depending on number of detected point. On the other hand, the methodology requires the perfect extrinsic parameters at the beginning of sequences. The realized tests prove that proposed techniques have sense. Thanks to SCCM system knows when parameters are not correct, and then calculation of a new extrinsic camera calibration is possible.

4.4 Analyze of online stereo camera calibration - OSCC

We show that the SCCM from previous section 4.3.4 can detect the need for new extrinsic camera parameters in the system (that the camera's position has changed). If such situation occurs, there is a need to find new R, T and E. When the system is able to detect decalibration moment, it should start calibration procedure. In the 3nd chapter, the section 3.4.3 shows that it is impossible to obtain a high accuracy of extrinsic parameters while the 8PA uses only points from one image frame. The amount and precision of points detected only at one frame are not sufficient. That is why, to have higher number of points to disposal, the calibration procedure should start the points accumulation process (described in the section 3.5.1). However, the considered input points for future calibration appears after a new camera's position acquisition. When system detects the moment when system is uncalibrated, it removes all points from the pipeline. The old point in the map structure describes old extrinsic parameters. The Fig 4.39 presents that the accumulation allows saving all the point in the new map (structure). This example contains only points from last three frames. The stored data permits to the different filtering strategy realization. The main goal is to choose best points for the calibration to obtain new extrinsic parameters as precisely as it is possible. Once extrinsic parameters are calculated and found, the system has to confirm the parameters, restart the SCCM process until the next uncalibration phase.

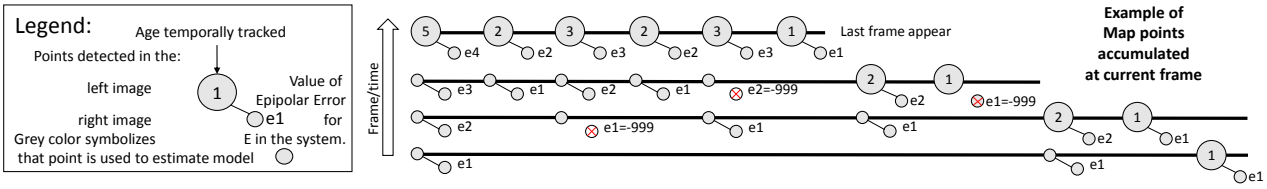


Figure 4.39: Example of standard points accumulation from last three frames. It presents how points look in the map structure. The history of points with its epipolar error is contained. While points are removed and new extrinsic parameters arrived, the error is not longer true. The only information storage in epipolar error is if point has a stereo match. The age and frame's number when point appeared is stored.

Based on structure from Fig 4.39, the different filtering functions can choose the most interesting points. The Fig 4.40 presents each techniques in graphic and text form. The system continuously tracks every point between left and right image. The data allows monitoring in how many frames it appears. System does not know the epipolar error value, because the E is not actual. The system fills the parameter with value -999 if and only if point is not stereo tracked.

4.4.1 Continuous stereo camera calibration and filtering methods verification

The continuous stereo camera calibration allows testing and choosing the best and most optimal filtering method for our pipeline. We run the calibration procedure on each frame for 40 continuous

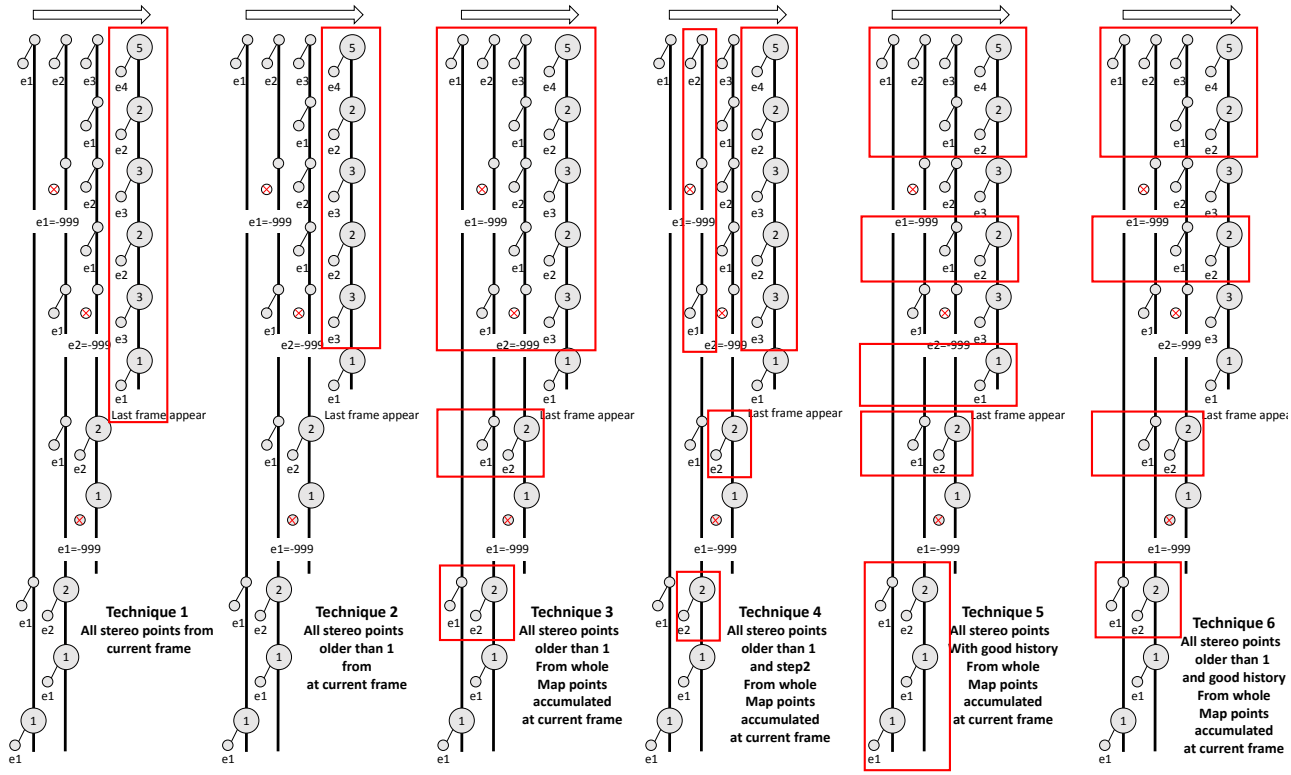


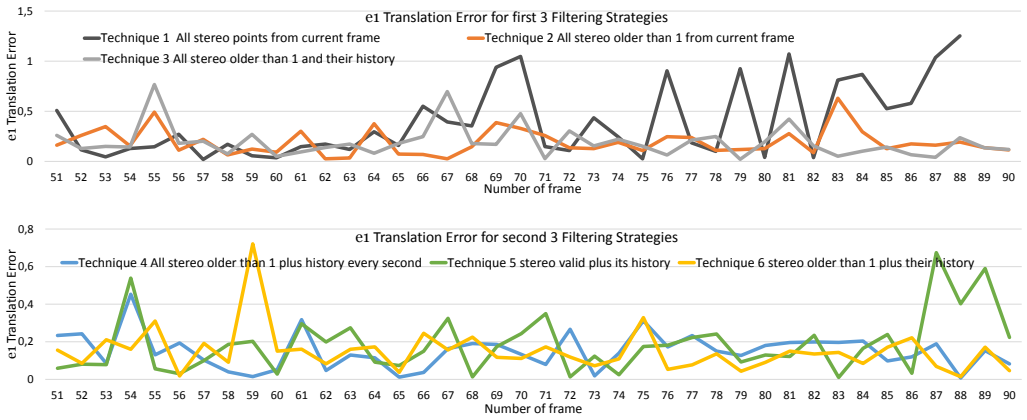
Figure 4.40: Filtering strategy points presentation.

- Technique 1 - All stereo from current frame (inliers, outliers, not considered points)
- Technique 2 - Stereo from current frame older than 1
- Technique 3 - Stereo older than 1 plus their history from the whole map point structure
- Technique 4 - All points older than 1 plus every second match of their history from the whole map point structure
- Technique 5 - Stereo if their history is good
- Technique 6 - Stereo older than 1 plus their history if and only if whole history is good

List 4.2: List of all filtering policies with description.

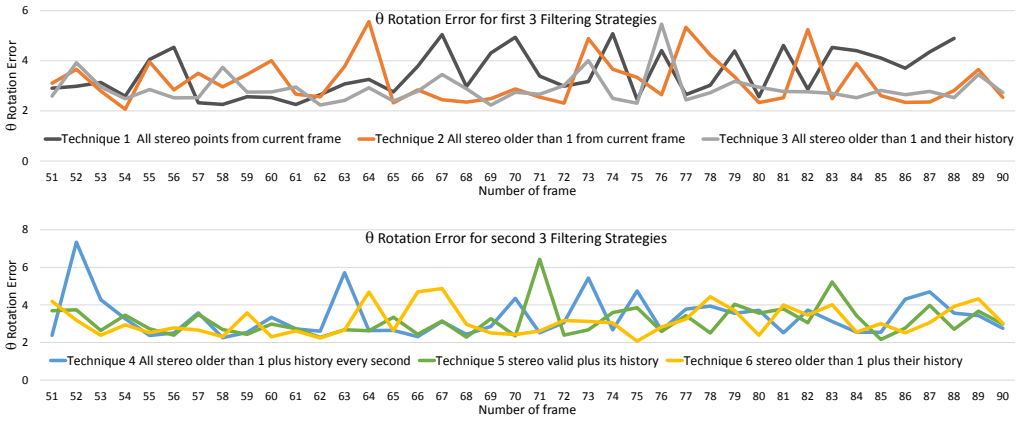
measurements in the same sequence. It starts on the 50th frame of input sequence, that the input points fill the map structure. The filtering strategy selects points according to methodology from the List 4.2. The points after filtration moves directly as the input for the 8PA. We select the preference from among methods based on comparing the obtained R and T to the parameters delivered by the traditional offline method. Two Figs 4.41 and 4.42 show the error of each strategy which is presented in a form of e_1 and θ (see section 3.3.2). Moreover, in the right part of the each figure, we present the small table with the median and average computed from all measurements. The system preforms the filtering test on custom dataset, which runs for SCCM test.

We obtain the best results in the term of T by the technique 6. The average from 40 measurements is equal 0.16. It provides all stereo points older than one with all stereo tracked history if and only if



Filtering Nb	T Average e_1	T Medians e_1
Technique 1	0,394179	0,215774
Technique 2	0,241214	0,180959
Technique 3	0,193432	0,155055
Technique 4	0,276511	0,240591
Technique 5	0,186161	0,169653
Technique 6	0,161861	0,139783

Figure 4.41: Two plots present the translation error - e_1 calculated between the T obtained in the continuous stereo camera calibration, which is based on the points provided by selected method and the offline traditional method. In first (up) there are three (1-3) strategy methods, which seems be less precise and more vary. On the second there are (4-6) another three methods, which are more stable and precise.



Filtering Nb	R Average θ	R Medians θ
Technique 1	3.182956	2.83877
Technique 2	3,486938	3,15278
Technique 3	2,876492	2,75434
Technique 4	2,974363	2,68922
Technique 5	3,155563	2,98101
Technique 6	3,141539	2,98655

Figure 4.42: Two plots present the rotation error - θ calculated between the R obtained in the continuous stereo camera calibration, which is based on the points provided by selected method and the offline traditional method. On first there are three (1-3) strategy methods which seems be less precise and more vary. On the second there are (4-6) another three methods, which are more stable and precise.

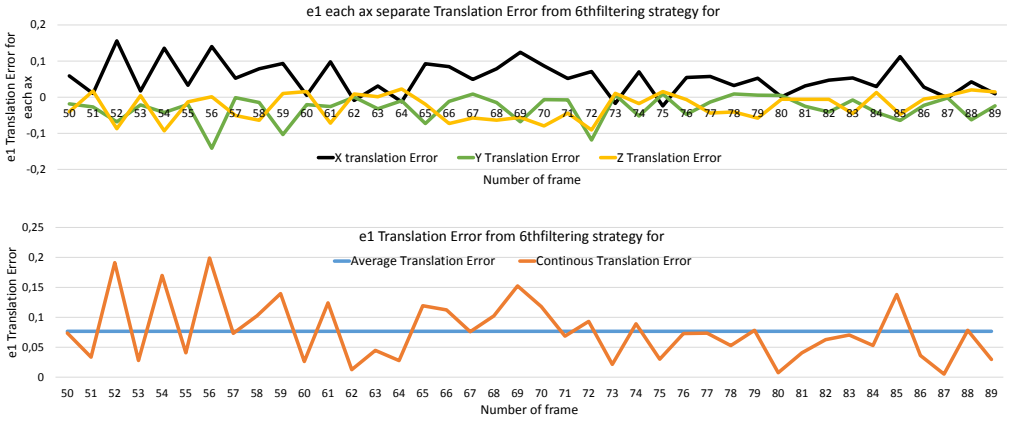
the whole history of point has stereo matches. This method should have one of the highest number of points among all methods. However, the differences between the technique 3 and 4 is negligibly small on this test. This is probably because the RANSAC can handle and remove a small number of wrong points without any problem. The R results are the same for each of the proposed filtering methods and is equal approximately 3 degrees. For the future tests, we propose to use only 6th technique to provide points for the OSCC.

Nevertheless, the obtained results are not very precise. The R and T found by online method are not the same as the parameters found by offline method. The two main facts cause it. The first, the point detector provides a position in the integer precision, when the offline method uses points

in the floating magnitude. The second fact, we noticed in the part 4.3.5. The created dataset does not provide a good enough structure to detect perfect points. The traditional method requires perfect points in term of distribution and distance from camera relatively close between 2 m to 3 m in order to calibrate offline. Therefore, the same type of point must be used to the OSCC. Unfortunately, the environment is very poor in term of structures in the used dataset. It prevents to the detection of a large number of points, from different distance to the structures in the view of the cameras. Moreover, the system has not same distribution of points in the each part of image, visible in Fig 4.38a.

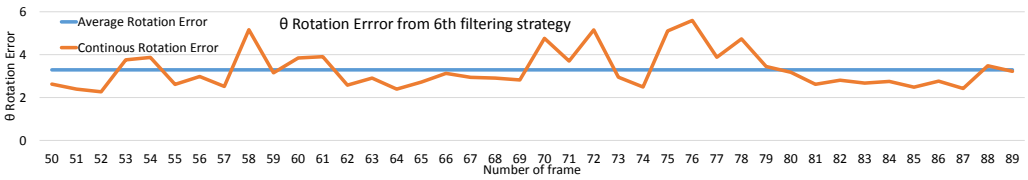
Based on this dataset, the system proves that even with not well-developed structures and not sufficient number of points in the integer precision, the algorithm is able to test and verify that the current extrinsic parameters are not precise. On the other hand, the custom dataset is not good enough to re-estimate the stereo extrinsic camera parameters, from the presented reasons.

We test the same filtering strategy in the continuous stereo camera calculation on the KITTI sequence. This dataset provides the perfect structure from each distance, which should allow computing better camera parameters, in order to validate proposed method. The four plots presents the obtained results of test. The Fig 4.44 shows the R error expressed in the θ, which is around 3. This result has the same level of precision as tests performed on previous dataset. On other hand, the test realized in this section use the perfect points and show that the θ reduces to 1.5. Fig 4.43 presents the T error e_1. For each axis, it is close to 0.07 of normalized value in the length distance, when for previous dataset it is 0.15. The points from the chessboard give the 0.02 result. The last Fig 4.45 gives the number of input points from the filtering method and number of inliers. We can compare it with the number of points from the same sequence presented in Fig 4.33, however we add history of points. That is why the number of points is almost 1.5 times higher.



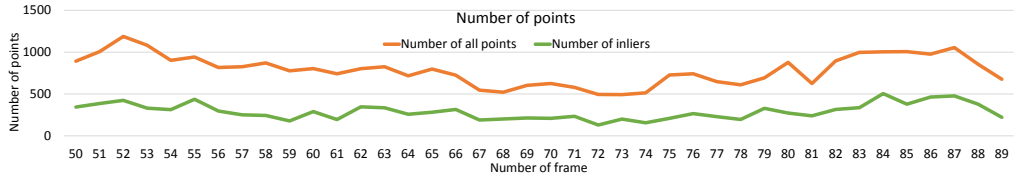
Filtering Nb	T Average e_1
Technique 6 for e_1	0,076692218
Technique 6 for X axis	0,052534208
Technique 6 for Y axis	-0,030248358
Technique 6 for Z axis	-0,025748553

Figure 4.43: Two plots present the translation error - e_1. On the top the error of each separate error is presented between the online and offline method. On the bottom the whole T error is shown.



Filtering Nb	R Average θ
Technique 6	3,291326

Figure 4.44: The plot presents the rotation error - θ between the R obtained in the continuous stereo camera calibration which is based on the points from the filtering strategy number 6. and the offline traditional method.



For Technique 6	Average number of points
All input points	787
inliers	289

Figure 4.45: The plot presents the number of input points delivered by the 6th filtering strategy and amount of the inliers used to estimate error.

4.4.2 Triggered stereo camera calibration

The previous sections show that the presented SCCM is able to check when the extrinsic parameters are not up to date. This section presents the continuous stereo camera calibration. The proposed accumulation and filtering strategies, average techniques and all SCCM policies are applied. Following section presents results of the OSCC, which activates the camera calibration, when it is required.

We realize all previously proposed methodologies on the same dataset. As for previous tests, the traditional method provides the input extrinsic parameters. At the beginning of the sequence, the SCCM procedure shall construct the initial global values with its thresholds in appropriate proportions. Then, all policies verify each condition in real time. The QoS waits when all conditions are satisfied. Next, the system can decide that it has to recalibrate, then the system delates the existing parameters and begins the data accumulation process.

In the previous tests, the system executes the continuous calibration when the point accumulation process runs during the 50 frames. In the real scenario, where FPS equals 16, the accumulation can take three seconds. For some applications, this time can be too long. For this reason, in this test, the system executes the 8PA if there is minimum 100 points in the accumulated structure and minimum 10 frames passed from the calibration trigger. When both conditions are satisfied, the algorithm starts to compute model and extrinsic parameters based on the filtered points from the map structure.

The newly estimated parameters are set in the system as soon as they are calculated. On this base, during next 15 frames, the system executes the new process of initial global values. Then, it computes and sets the new thresholds in the same proportions, as first ones. When done, the new SCCM process begins again.

Custom dataset

For the first test, the dataset created according to a methodology from the section 4.2 is used. Figs 4.46 and 4.47 presents the number of points and sum of epipolar error, which are the main parameters on each every policy, runs. Due to the other filtering techniques for former tests, the number of points and sum of error is not equal to the previous measurements from Figs 4.16 and 4.17, even though they are realized on the same sequence. Despite, all tendencies of the observed parameters occur and are similar.

It is important to notice that the scale of epipolar error has changed. It is because instead of using points expressed in pixel coordinates and F. The system computes the epipolar error on the base of normalized points and E. This reduces order of error magnitude and affects each epipolar error policy.

Following, the two plots 4.47 and 4.48 presents selected SCCM policies. We present the ratio in logarithmic scale, because the second half of the dataset has parameters on another order of magnitude. The last Fig 4.50 illustrates the decision making process.

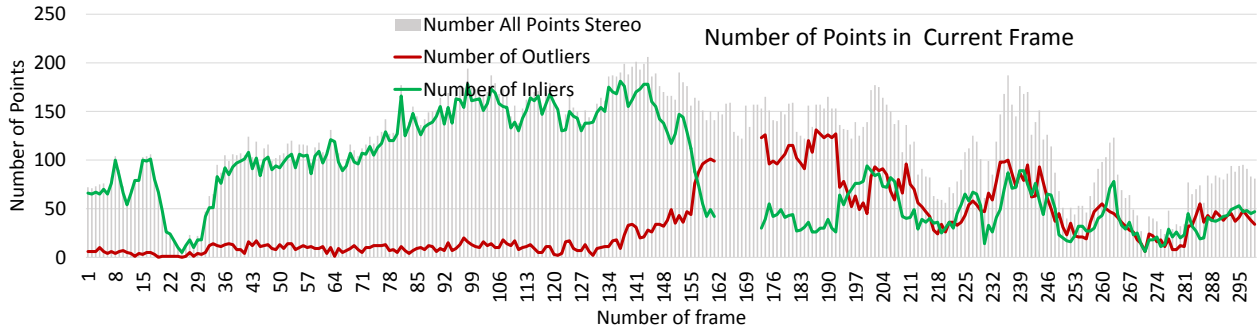


Figure 4.46: The plot shows the number of points from each group, all stereo, inliers and outliers from every frame of the analyzed sequence (on custom recorded dataset).

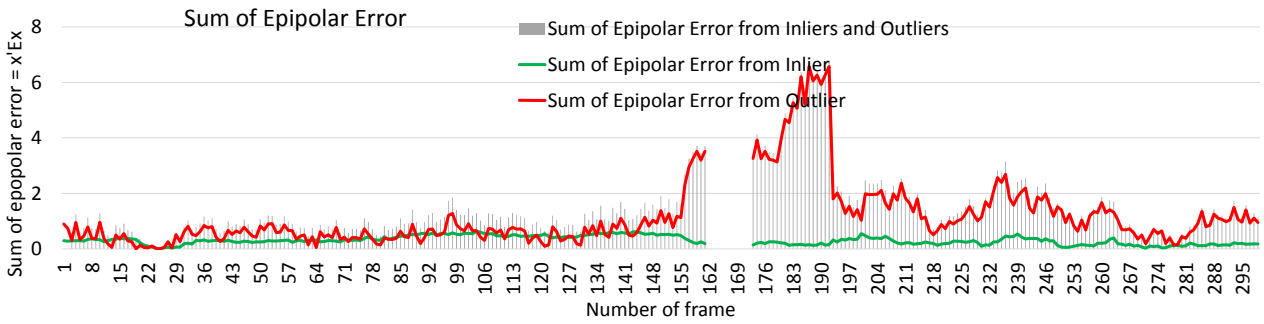


Figure 4.47: The plot shows the sum of epipolar error on inliers and outliers group from every frame of the analyzed sequence (on custom recorded dataset).

In each of figure, the moment when the camera changes poses can be detected. The frame when SCCM decided that system is decalibrated occurs at 161th frame. It is one frame earlier compared to the test from Fig 4.28. After recalibration, the measured parameters are not as high as they were

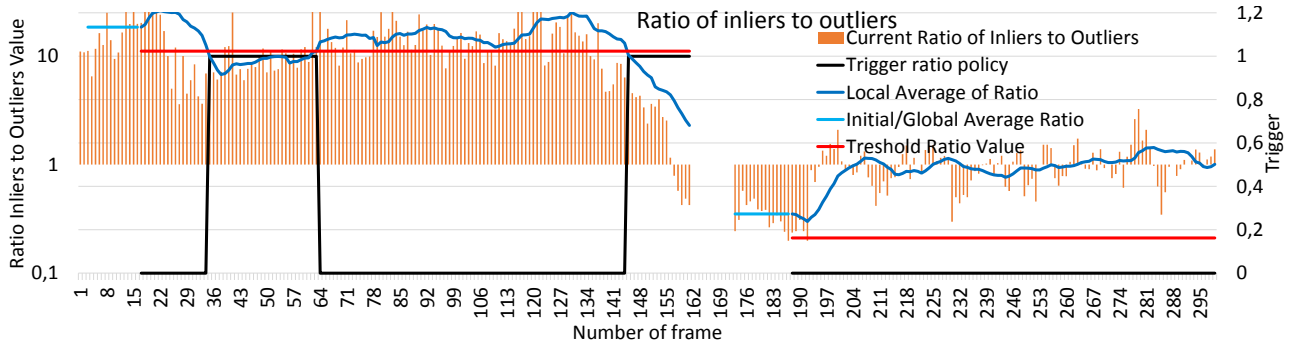


Figure 4.48: Ratio between inliers and outliers - one of the SCCM, which is based on the number of points, realized on recorded dataset.

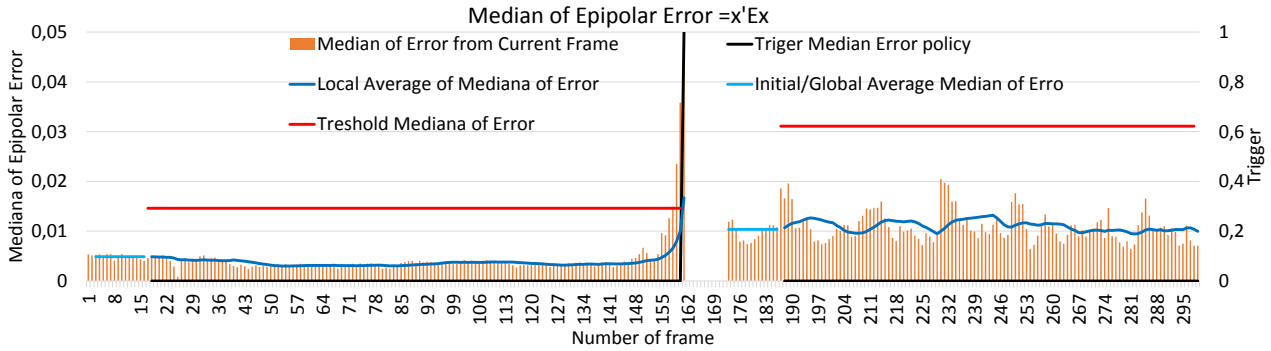


Figure 4.49: Median of epipolar error measurement - one of the SCCM policy, which is based on the epipolar error, realized on recorded dataset.

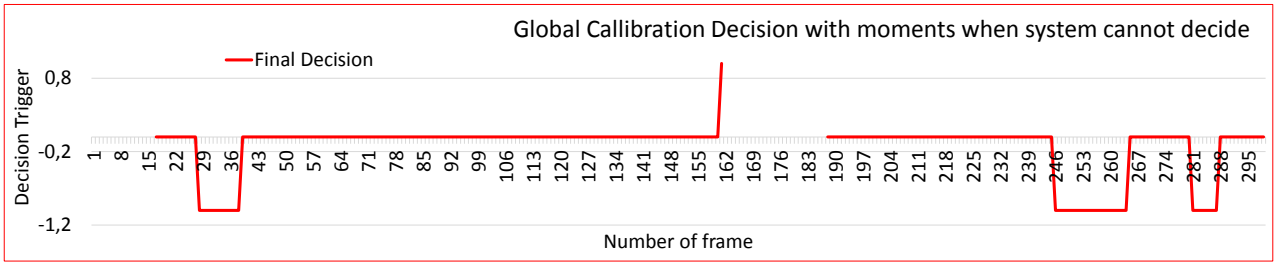


Figure 4.50: The final decision of SCCM realized on recorded dataset.

before. However, most of the new thresholds are not intersected by the average measurement values, as it is shown in Figs 4.19 and 4.22. Therefore, the system considers that it is correctly calibrated. It suggests that the calculated parameters are more accurate than the initial, offline parameters.

From the monitored parameters, it can be concluded that the second change of the position of the camera did not occur in the later part of the measurements. When camera changed position differences in term of epipolar error and number of points is huge. It is observed that this kind of variation does not appear after the system is recalibrated. Unfortunately, in spite of everything, it is difficult to decide if the system has been precisely calibrated with new parameters. The system calculates its new global parameters, assuming that it has been correctly calibrated. According to the tests performed in section 4.4.1, the results obtained are not always correct.

Kitti dataset

For the second test, the KITTI scenario, which has perfect parameters for the whole sequence, is used. In order to make this dataset usable, after 21 frames the algorithm automatically resets the existing extrinsic parameters in the system. In this manner, it simulates parameters decalibration. This decision is visible in Fig 4.51. Thus to this the process of accumulation start at 21th frame and goes until 33th. Then during next 15 frames, the new global initial values and thresholds are constructed. At 48th frame, system starts again the SCCM

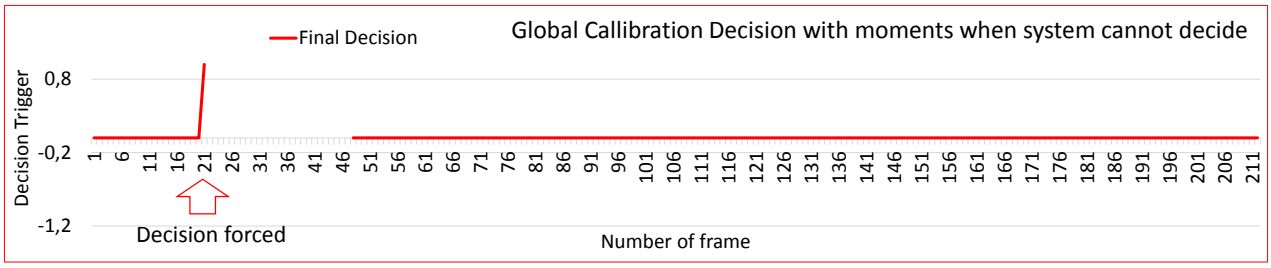


Figure 4.51: The final decision of SCCM realized on KITTI dataset.

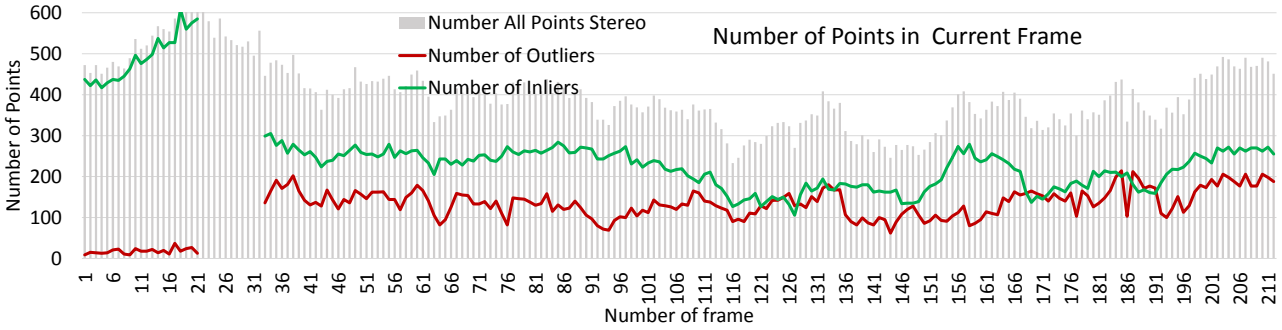


Figure 4.52: The plot shows the number of points from each group, all stereo, inliers, outliers and not considered from every frame of the analyzed sequence (KITTI dataset).

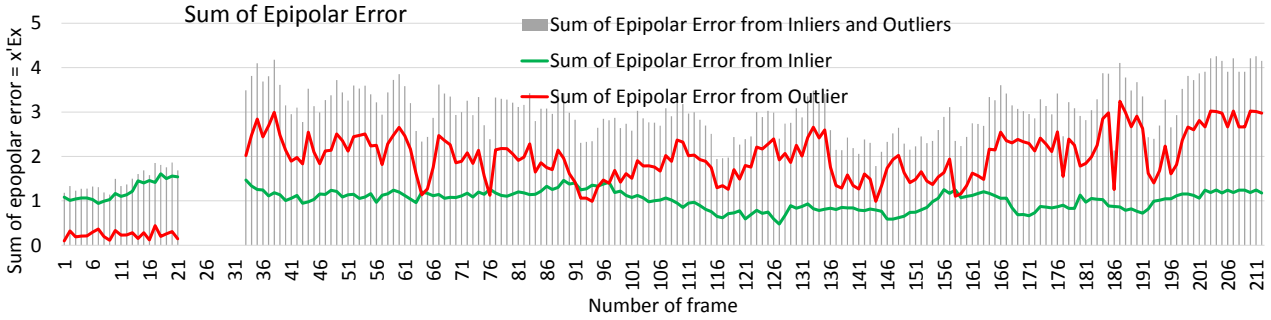


Figure 4.53: The plot shows the sum of epipolar error on inliers and outliers group from every frame of the analyzed sequence (KITTI dataset).

The Fig 4.52 illustrates the number of inliers and outliers. We can observe that the proportion between two numbers is clearly different during first 20 and last 140 frames. This plot allows concluding

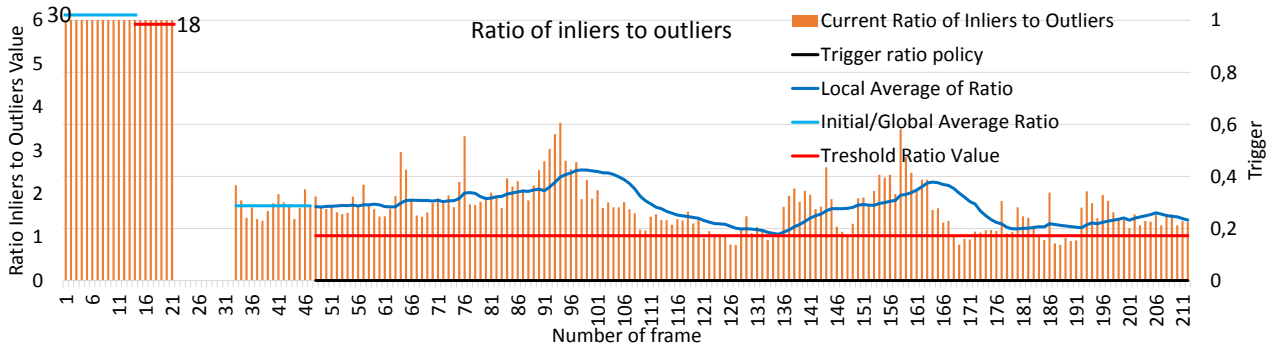


Figure 4.54: Ratio between inliers and outliers - one of the SCCM policy, which is based on the number of points, realized on KITTI dataset.

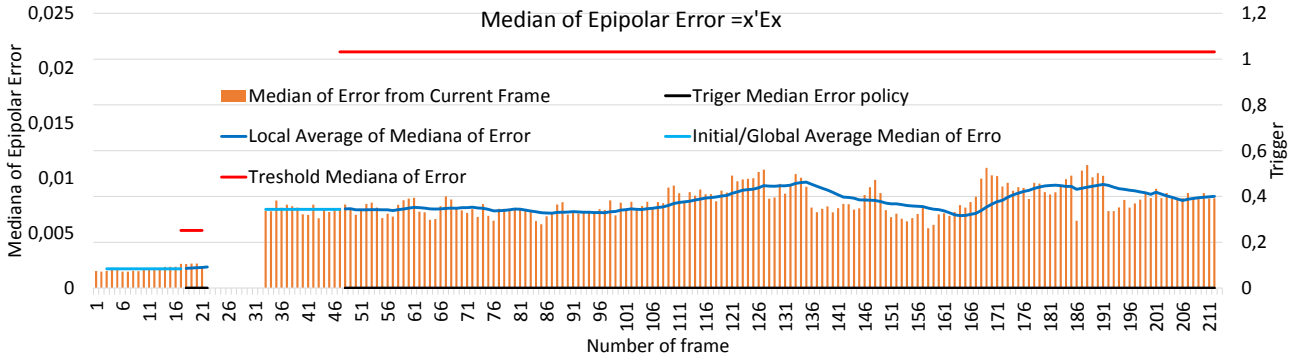


Figure 4.55: Median of epipolar error measurement - one of the SCCM policy, which is based on the epipolar error, realized on KITTI dataset.

that the calculated parameters are not as precise as those found by traditional method. We can draw the similar conclusions from the Fig 4.53, which shows the amount of epipolar error.

Two policies presented in Figs 4.54 and 4.55 show that both obtained value can turn the alarm with previous thresholds. However, the system calculates them from scratch that is why it cannot compare it directly. This work presents that the quality of parameters obtained by the selected 8PA significantly deviates from the value of parameters calculated by means of the traditional algorithm. However, the re-estimated parameters and thresholds allow considering that, the system is well calibrated.

4.4.3 Precision of the results

The selected 8PA is not the most accurate choice possible, but it has been chosen because its parameters allow testing it on an embedded systems. The results shown in the previous tests are described and some aspects to improve the precision of the results are presented in this part of the chapter.

RANSAC parameters

The section 3.3.1 presents the importance of the RANSAC's parameters. Its threshold is a key variable that de ne if current tested point is in inlier or outlier group. The first column in Table 4.3 presents the constant parameter value for the whole previous tests. This allows to almost each time confirm model before the maximal number of iterations, which means that the best model is always found. However, if the threshold value is smaller and searching model must be more precise, it can happened that algorithm does not confirm model, Then the procedure achieves the maximal number of iterations (1000 here) with possibly the best, but not confirmed model. The change the threshold parameter is made and the results are shown in the second column of Table 4.3.

We can observe the significant correction in both errors expressed in θ and e_1 for the second column. However, it does not result from precise measurements, but from the higher stability of the calculated parameters. The final model fluctuates less. Therefore, the average of errors comes out appropriately lower. With worse RANSAC threshold, it is possible to obtain the same precise of results, but as the method sometimes indicates an inaccurate model, the average result of several measurements is much higher. On the other hand, lowering the threshold parameter causes that the model is very often not conformed before the maximum number of iterations. In such case, it is important to keep this parameter relatively low so that the algorithm does not work too long, when it is not able to confirm the model. Both measurements come from the same sequence of KITTI dataset, where the 1036 input points go to 8PA. In the next part of the work, we test the selected algorithm on an embedded system processor with the lower value of the threshold.

Parameter	Measurement 1	Measurement 2
RANSAC threshold	7.8125e-07	7.8125e-08
Number of iteration	826	9220
Number of input points	1036	1036
Error of R	3.2913 θ	1.907 θ
Error of T	0.0866 e_1	0.0377 e_1

Table 4.3: The table shows the results of the 8PA when the different RANSAC parameter is used. The averages values are calculated from 10 measurements of the same sequence.

Subpixelic approach

The precision of the points is extremely important in the 8PA. The tests from previous chapter proved that calculation of the extrinsic parameters similar to the offline method is possible. However, the

input points are expressed in float form, with accuracy to three decimal places.

According to the section 4.1.2 where we describe the C-tracker used in the tested pipeline. It provides the precision of input points in the pixel integer. The position is expressed in this manner, because it is a copy of the version, which is implemented on the FPGA available in the laboratory. Precise point detection is a complex task, increasing precision can only be achieved if the point is more accurately expressed between pixels. This type of improvement should definitely increase the precision of the obtained results and increase their stability.

The extension to the float (subpixelic) form of input point must be realized in tracker for the future tests. The conclusion from obtained results is that the need to more precise points is critical and essential. It should have the high impact, especially on points detected from close distance from camera. For this reason, we propose the subpixelic extension of the pipeline. The good distribution of points in term of different distance and all part of image can increase the quality of OSCC.

Impact of calibration on the image rectification procedure

We should not study the stereo camera calibration procedure itself in isolation to the high-level applications. The system considers the calibration as the input data provider. For this reason, we present in section that some of the functions can measure the impact calibration method. The section presents the aim and goal of the rectification procedure. The following part of the chapter shows the impact of different extrinsic parameters in this process building block for high-level application. The presented figures illustrated the captured image frame, from the custom-recorded dataset. The green lines are not the epipolar line, but the horizontal lines. It allows finding the same line in the both images. However, if the image is rectified the lines should represent the epipolar lines.

We present the screens of rectification application to simplify understanding the results. In each left corner of figure, the text in the rectangle presents if the images are input or output. Moreover, in the red circles there are characteristic POI, which describes the same interesting points in both images. The Fig 4.56 shows results of the rectification, which use the parameters, found by the offline method. The extrinsic parameters are very precise and allow computing well-rectified images. In the center of the image on the black background, there is log from system, which informs that system is well calibrated and gives a current number of frame.

The second Fig 4.57 presents the same dataset but during the second half, when we touched the cameras. Thus, the image illustrates the wrong rectification. Because, there are the same initial parameters from the first half of the dataset, obtained by the traditional approach. This type of information allows us to interpret that the system is uncalibrated. The Last Fig 4.58 presents the rectification results of the KITTI dataset. We can observe that the initial parameters are correct for the rectification process.

Figure 4.56: Rectification process realized on the extrinsic parameters which are based on the traditional method - it is realized on the first part of custom dataset.

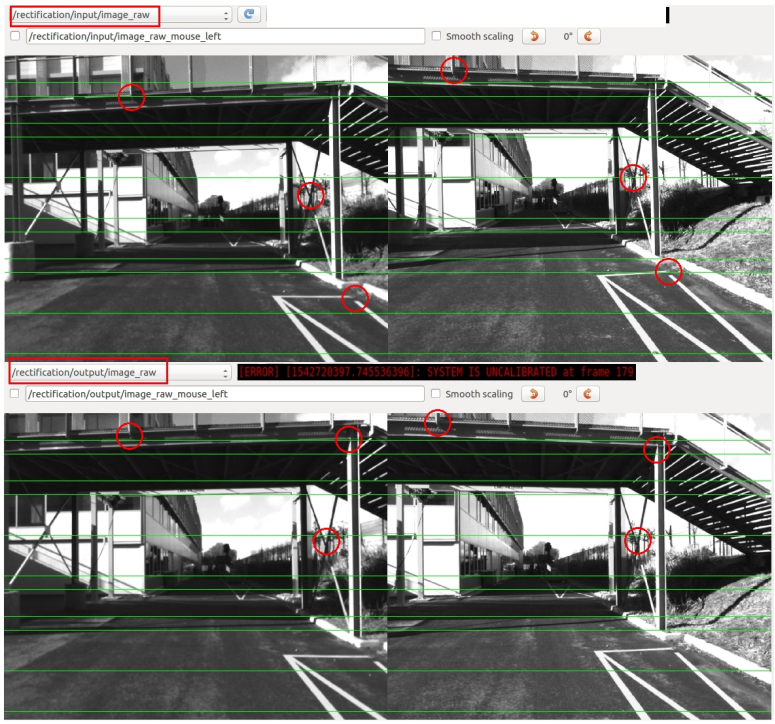


Figure 4.57: Rectification process realized on the extrinsic parameters which are based on the traditional method - it is realized on the second part of custom dataset.

Figure 4.58: Rectification process realized on the extrinsic parameters which are based on the traditional method - it is realized on KITTI dataset.

4.5 The whole approach characterization on RPi 2b

This chapter presents the detailed approach to online stereo camera calibration in real time. The previously tests show the results carried out on the PC. It is the first environment of this work according to the section 4.1. It allows concluding on many aspects as precision and realization of the calibration. It provides opportunity to transfer the entire methodology to embedded systems. In accordance with the second working environment from section 4.1.3, the system realizes the method in the same structure as for previous test. We properly adapt the code to the RPi environment. Then we execute system on ARM Cortex A7 embedded processors, which is available in the RPi. This section presents the characteristics of the time performances of each function.

The tested sequence is a fragment of a dataset recorded by us, used in the previous tests. The Fig 4.59 presents the dataset explanation in the graphic view. The system creates the map points in the first frame, and then at each time when input points arrive, the structure is expanding. We call this function – the F1 in the order to facilitate a discussion. In the flow, the second function (F2) normalizes input points by the intrinsic parameters.

Once, after first 17 frames, the averages and threshold values are constructed for each policy on the base of the last 15 frames (F3). If and only if the parameters are set the SCCM, start to work (F4). It calculates all 7 policies based on the number of points and epipolar error. The decision making function (F5) analyzes the SCCM output and decide if the current extrinsic parameters are precise. As long as they are correct, nothing changes and only those four functions work on every frame. However, if the decision is positive, the system removes the current map of points. From the next frame, the procedure builds the new structure with appropriate filtration (F6). The procedure performs the calibration (F7) when there are sufficient number of points in the new structure.

The previous last position in Table 4.4 is sum of four functions used to the whole SCCM computation, from the received input points through normalization, to calculation and decision computation (F8), while the last row is the OSCC function so the data filtering and 8PA realization. From the

analyzed sequence, in the Table 4.4, we present the average time of each functions. The whole tested dataset has 60 frames, where at 37th the decalibration is detected. The second Table 4.5 shows the details of the different calibration measurements.

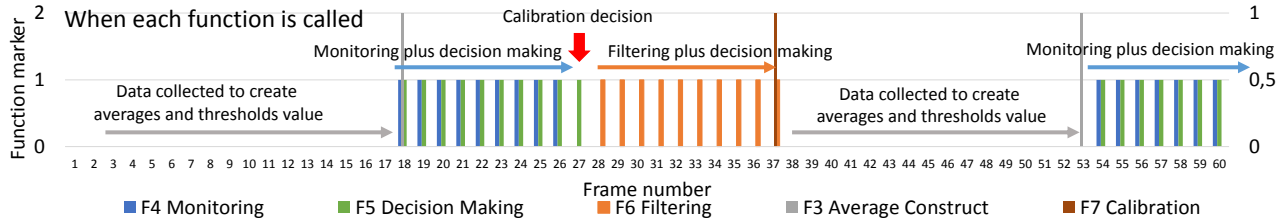


Figure 4.59: Plot presents the analyzed sequences for time characteristic. According to the number of frame different functions is executed. Function 1 which create a map structure and Function 3, which normalize data, are realized each frame.

Number - name Function	Average time	How often realize
F1 Map point structure creation	0.02417 s	Each frame
F2 New points normalization	0.023951 s	Each frame
F3 Averages and threshold constructions	2,00E-06 s	Once after first 17 frames arrived, and there are not the parameters in the system
F4 Monitoring SCCM	0,028181 s	Each frame, when the threshold and averages are constructed
F5 Decision making	2,00E-06 s	Each frame, when the SCCM works
F6 Points filtering	0,010791 s	If parameters are wrong as long as new parameters does not arrive
F7 Calibration	21.857 s	If parameters are wrong and there is enough point in the filtered structure, and passed at least 10 frames from calibration decision
F8 Whole SCCM = F1 + F2 + F4 + F5 F9 Whole OSCC = F6 + F7	0.076 s 21.867 s	Each frame when calibration are good and parameters set Once when monitoring signalized that parameters are not precise and there is enough point in the system

Table 4.4: The table shows the average time in seconds for each function that is executed on custom dataset. The right column of the table shows how often a function is performed in that sequence.

Description	test callgrind	1	2	3	4	5	Average
Time Required [s]	672.549	26.09	25.32	9.81	26.31	21.74	21.857
Error of T e_1	0,02376	0,01603	0,07063	0,02251	0,02512	0,05444	0,03775
Error of R θ	1,919	1,925	1,881	1,915	1.917	1.898	1,907
Number of Input Points	1033	1036	1036	1037	1036	1036	1036
Number of inliers	424	435	415	496	441	441	447.2
Number of Iteration	10001	10001	10001	3513	10001	8534	8410

Table 4.5: The details of different calibration measurement (six independent runs). The first column shows the monitored parameters. The second column presents parameters obtained in the Callgrind simulation, that is why the time required by first column is higher, and it is not included in average. Next columns shows measurements from one run of program. In the last column the average from 10 execution is calculated.

The calibration clearly outperforms other functions by the amount of time. It is the most demanding in term of computation function in the entire pipeline. An embedded processor such as Cortex ARM A7 requires around 21 s to calculate the extrinsic parameters. The right column in Table 4.5 provides an average time from different measurement based on the same input points. These obtained results are far from the actual real time calibration approach. During that time, many new frames arrived, for

example if there is 20 FPS, more than $20 \times 21 = 420$ frames arrived. On the other hand, the SCCM does not require many computation load and can be effectively executed in real time on an embedded system with a tested processor. Each function required by SCCM i.e. F1 + F2 + F4 + F5 needs 0.076 s.

Another important aspect of this work is to place the stereo camera calibration as a function built into the entire application pipelines. The section 3.2.3 presents the methodology of this solution. The benefits of this approach are significant. The external processing unit realizes some of the most concerning functions such as POI detection, extraction and matching. In this work, we propose and realize the low-level processing function on external processing unite. In our methodology, the stereo matched points arrive directly to the calibration block. This approach allows omitting the impact of the low-level functions on the time required by camera calibration procedure.

For the moment, we propose to execute the calibration on the same processor where we can realize the high-level application. In such situation, when a system is well calibrated, only SCCM can work with the other application such us rectification, because it does not need a many resources. On the other hand, when the system detects that the system is not calibrated any more, the high-level application does not make sense and can stop. The false data provided to the functions do not allow to proper execution of functions. Then the calibration method has 100 % of resources to perform OSCC, Table 4.5 shows that it needs approximate 22 s. Once after, the system computers new extrinsic parameters, they are in the system and the high-level application with SCCM can run again.

Finally, it is important to notice that time required to execute each function depends on the number of input points. Each function performs mathematical operations based on input data. If the size of input data increases, the number of required calculations grows. We execute the entire test on the same custom dataset that we presented in previous sections of this chapter. Thanks to the similar number of input points on each image with a similar precision, the system requires the same time to execute a whole pipeline. In the future work, we would like to test the approach on other sequences with a random number of points.

4.6 Summary and conclusion of results and realization

This 4th chapter presents the experimental phase of the previously described approach to solve the OSCC on embedded systems. The section 4.1 shows the different environments setups used during the implementation tests. The first one is a standard PC with an Intel i7-260 CPU. The second environment is the RPi 2B equipped with the embedded processors ARM Cortex A7. This processor has similar parameters to the target CPS environment. This third environment equipped with MIMOSA board, where two FPGA, Intel i5 and ARM core is available. We select this setup as the final electronic devices, because it is one of the project developed in the laboratory.

At the end of the first section, we explain the methodology for required data set. In the literature, there is not dataset, which allows for the OSCC tests. The third environment allows creating a necessary sequence to realize the future tasks. We present many details of the dataset that must be satisfied in order to run the calibration of camera parameters.

We point the well synchronization of the image streams, appropriate distribution of structures on the image, even distribution of points and elements of the scene, which allows distinguishing POI without problem, etc. Moreover, the system must see the structures at different distances from the camera, so that the majority of points is not located far from the cameras. Unfortunately, the future tests show that the custom-recorded dataset does not meet all of these restrictions. On the other hand, we use the KITTI dataset, which does not provide sequences with different camera calibration setup. However, it provides a well distributed structures in the images, which allows for precise POI detection

The second section 4.3 explains in the details the whole methodology and implementation of SCCM. The need for and the sense of the SCCM is admitted in the base of the results of two different datasets. Both cases show proper and expected results, due to realized dataset. The proposed SCCM proved that the system, which is equipped with this function, is able to detect if current extrinsic parameters are precise, thanks to the proposed SCCM policies. There are two different groups of techniques, which run on the number of points and their epipolar error. The six various tactics are able to detect if one of camera poses has changed, on the other hand, one policy is responsible for confirmation if there are sufficient number of detected point in the image. Each of SCCM policies is independent from the dataset, but requires perfect initialization parameters.

It is important to remind that proposed solution, from the point of view of the CPS, hides the SCCM and OSCC in the application pipeline. The selected algorithm is based on the simple POI, so it can take profits, because this kind of data exists into the system for other purpose.

Next section 4.4 provides analyze of the whole pipeline of OSCC. Moreover, it presents the impact of the filtering method on the precision of results and other aspects in the triggered calibration procedure. The proposed optimization of whole approach such as point accumulation and filtering can help the 8PA to select the best and most reliable input data thus increased the precision of the results. In

the test, the SCCM triggers the calibration and inject new extrinsic parameters in the system. The algorithm confirms the new parameters, but the algorithm cannot confirm how precise they are. It is caused by the fact, that algorithm implicitly consider the calculated parameter as accurate.

Results in term of precision based on sequences from the KITTI dataset are satisfactory. They allow looking with optimism at the future. However, the precision of extrinsic parameters found in custom dataset is far from the expected results. This show how important in this approach is the good dataset, where there is enough structure from different distances and well distributed in the scene. The appropriate data accumulation, filtering functions and RANSAC parameters stabilize the obtained results. The parameter precision depends on the precision of the input points. The presented approach targets the embedded systems, which have many restrictions. One of them is the imperfect detection of POI. Those are rounded to the size of pixels, thus to the numerical values of the integer. The introduction of subpixel precision for POI detection can significantly increase the expected results.

The last part of this section gives a time characterization of the second environment, where we test the whole pipeline on an embedded processor. The SCCM can realize its task during the real time. We propose to use it as an additional functionality for future models of the stereo cameras. It informs that provided data from the sensor are correct and can be trusted. It introduces a greater reliability and security into the system. On the other hand, the system knows when it needs calibration. It can stop the task at a safe moment. This work proposes an approach to recalibration from data available in the system. The method uses only on the POI provided by the cameras, so they must be precise. The selected algorithm needs about 20 s to calculate new parameters on the selected embedded processor. The obtained results are stable by appropriate real-time data accumulation and filtering functions. The results of the quality of the parameters are strongly dependent on the input dataset. We present the discussed quality of results in the previous section.

Chapter 5

Conclusion and future work

The last chapter of this manuscript concludes the whole work and present future goals. The 1st chapter presented the main context and motivation of this work. The 2nd part shows the existing stereo camera calibration methods and dataset in the literature. The 3rd chapter explains custom approach to online stereo camera calibration in the application pipeline. The 4th chapter presents the whole realization and results of the advanced calibration pipeline.

5.1 Conclusion

This manuscript presents the study of an online calibration pipeline on the embedded systems. The main goal of this work was to select the best method from existing algorithms and test it in the targeted smart glasses context with embedded system limitation.

Nowadays, there are many different stereo calibration methods. The procedures are usually adapted to specific sensor configuration, applications and environments. The most popular stereo cameras ensure that the parameters do not change. The system performs calibration once at the beginning and it guarantees that the system is working properly. These methods require well-known calibration patterns in many planes and distances from the camera. The operator must correctly prepare the procedures. Moreover, mostly all of them are realized offline.

In this work, the main hypothesis is that the camera positions can change due to various circumstances. Stereo cameras are exposed to many factors that can change their position. For this reason, the whole system must have ability to verify if the current parameters are up to date. In the situation, when they are not valid, the selected algorithm should recalculate new precise extrinsic parameters. Therefore, the method should be able to execute online during the system's mission, without the use any special calibration tool. The self-calibration methods fulfil these restrictions. However, these computer vision procedures run usually on the powerful hardware, like a PC. The context of this work enforces to realize the self-calibration method on embedded systems, which is not a common assumption in the literature.

In increasingly intelligent systems should increase their reliability, safety, security and precision data. Moreover, systems must be as self-automatic as it is possible. For this reason, the control of camera parameters and their recalibration must be possible in the systems of the future. I am convinced that the role of camera as leading sensors will force calibration processing in real time on the devices such have an embedded processor.

The calibration is never the main task of any devices. The process feeds the pipeline with the right parameters. Therefore, this work proposes the concept that the calibration method must hide into the whole application pipeline to reduce the number of computation required by calibration procedure and realize it online. We analyzed many algorithms in that context during the state of the art. We select the method, which inject and use the basic data occurring in the majority low-level computer vision processes.

Thus, we propose to use the 8PA low complex self-calibration method. The algorithm uses only simple stereo matched point of interests (POI) which exist in many computer vision applications. It allows considering that, the input points for the algorithm come with zero cost, because this data are anyway in the system. We proved that the selected 8PA has ability to calculate the precise extrinsic parameters with perfect input points. We postulated that the system could find the perfect input POI from the real scene. To achieve it, we proposed the advanced calibration pipeline with accumulation, filtering, monitoring strategy and real scale extraction. The chosen methodology and method is universal, independent from other sensors, environments and scenarios. The system should realize the proposed method in the background of application pipeline. Thus to its input points, it can be performed at any mission in the computer vision application pipeline. In this work, we proposed another important concept related to the online camera calibration pipeline, the stereo camera calibration monitoring SCCM. The approach, which verifies if current extrinsic parameters in the system are still precise enough. The proposed monitoring methodology uses several policies, which measure quality of current R and T. In order to test and verify the selected approach, a dataset where two different calibrations exist in one scenario is required. This kind of computer vision benchmark does not exist in the literature. We needed to specify and create this dataset.

Therefore, during this work, we proposed a methodology for perfect dataset. We recorded a custom dataset according to our approach. During this activity, we solve many problems related to the dataset such as a prototype device, synchronization between images, parameters of the camera settings, scene elements, reference camera calibration, etc. We realized few scenarios, when during sequences we changed the right camera pose with the programmed moment. We know the camera calibration due to the offline camera calibration at the beginning and at the end of the dataset. We successfully tested the proposed SCCM on two different custom datasets, it is able to detect at perfect moment that

system requires a new calibration. The control tactics and whole approach are independent from the dataset. They detect a false alarm and if in the system, there is sufficient number of points take a right decision. This procedure can guarantee that the system is well calibrated and the camera parameters are updated.

The system can successfully detect the moment when the extrinsic parameters changed due to custom methodology. We propose the online stereo camera calibration (OSCC) procedure in order to avoid the return of stereo cameras to the manufacturing process. The advanced 8PA selects the best stereo points detected from last frames in the system according to the proposed accumulation block and MPOI strategy. The accumulation block collects the information about points thanks to temporal and stereo tracking strategy proposed in this thesis. We proposed a filtering the POI in order to select the most stable and precise points. Thanks to this upgrades the whole advanced calibration pipeline can obtain different quality of extrinsic parameters, on the basis of the different precision of the input points.

We study the whole approach on different environments. We obtain the first results on the standard PC. Then, we characterize the whole pipeline in an embedded system - Raspberry Pi 2 Model B (RPi), where the ARM Cortex A7 processor is used. Finally, we present the whole prototype as the custom final target environment. In the final version, we test the whole online calibration pipeline, where the SCCM trigger the OSCC. It successfully performs on the targeted processor. We solve many issues with code transfer between different coding environments in order to realize these tests. We measure its performance on the final target. We prove that the SCCM needs the 76 ms to verify if current parameters are precise as it is a background task. We consider it as the real time process without any additional optimization. However, we need initially calibrate the whole system in order to run. The OSCC with the 8PA requires around 20 s to compute new extrinsic parameters on the RPi, while calibration is the background process.

The obtained results allows considering the online calibration pipeline as the one, which can handle operation in the real time. However, the computation of new parameters take a while but the traditional calibration method realized on the PC environment in Matlab or other application frequently takes more time. The approach of SCCM satisfy the real time constraints on embedded systems and run on ARM processor at near real time.

The advanced calibration approach adds the safety feedback loop in the application or cyber physical system. It guarantees that the stereo camera are precisely calibrated and provide correct data. The proposed methodology increase a reliability and precision of the system where the stereo camera exists. Thus, a new automatic functionality such as self-healing and self-adaptation for long-term missions is proposed. The subject of online calibration has not yet been carefully analyzed in the literature. This work is a prelude to reflections on the calibration of an online stereo camera on an embedded

system. In view of the results, I am convinced that the proposed approach can support and enhance the autonomy of modern systems. I am aware of the quality of the obtained parameters, which can sometimes be unstable and unsatisfactory for some applications with high functionality. However, I strongly believe that the introduction of future work, which is in the next section of this chapter, help to achieve this performance.

5.2 Future work

During this thesis, we successfully realized the whole concept of online stereo camera calibration with parameters monitoring in the application pipeline. The proposed methodology performs the task assigned to it, with several assumptions. We would like to eliminate them in future work.

The SCCM concept works properly if and only if input extrinsic parameters are precise. However, if the precision of the calculated parameters by OSCC is not excellent, and algorithm chooses new parameters not precise enough, the system fails but behaves as if it work well. Thus, we would like to improve the precision of OSCC. Despite all this, we consider the selected 8PA algorithm as poorly accurate. However, it is necessary to remember that we would like to execute this work with the whole algorithm on embedded systems, where computing power and memory are limited.

This manuscript shows that the accuracy of input points is a key to improve method's precision. According to selected prototype used in this thesis, the custom embedded tracker generates the point's position in the integer. The tracker needs the subpixelic precision, which can significantly increase the quality of results.

The realized approach extracts the scale factor from the baseline. This assumption works well for custom prototype. However, for future more universal CPS, the scale should be extracted from the scene. The work proposes a methodology to detect the scale from objects. We propose to test it in the future work.

We realized the datasets in the natural environment. Therefore, these are very complicated in the analysis of camera calibration. There are many conditions that good dataset must satisfy. We did not realize all of them during record of custom dataset. There is a lack of some elements of the scene from close distances during the recorded sequences. Moreover, the number of POI detected at each part of image is limited. In future we should realize more complete and various datasets. Then, we must perform the custom approach, in order to validate the whole calibration pipeline on different sequences.

For this reason, we used the KITTI dataset. During the whole sequence, there is only one calibration and there is no need to recalibrate. However, this sequence provides good image structures where the

system can detect points from each part of the image and from many distances to the camera. These datasets provide precise results realized by OSCC.

The assembly of the entire prototype and the implementation of a high-level application at the end of the pipeline is necessary for a complete verification of the whole concept. The application should be able to inform and verify whether the calculated parameters are accurate. It generates additional feedback, which can force OSCC of the camera on a similar principle as the SCCM. We should implement and realize everything together on the presented prototype.

Chapter 6

Appendix

6.1 Projective camera geometry

The basic pinhole, camera model. There are camera, image and the world coordinate systems. For each of them, a special parameters are related. The camera and image coordinate system are described by intrinsic or internal parameters which include a radial distortion parameters. The extrinsic, external parameters characterize the world system.

We consider the central projection of points in space onto a plane. Let the center of projection be the origin of a Euclidean coordinate system, and consider the plane $Z = f$, which is called the image plane or focal plane. Under the pinhole camera model, a point in space with coordinates $X = (X, Y, Z)^T$ is mapped to the point on the image plane where a line joining the point X to the center of projection meets the image plane. This is shown in Fig 6.1.

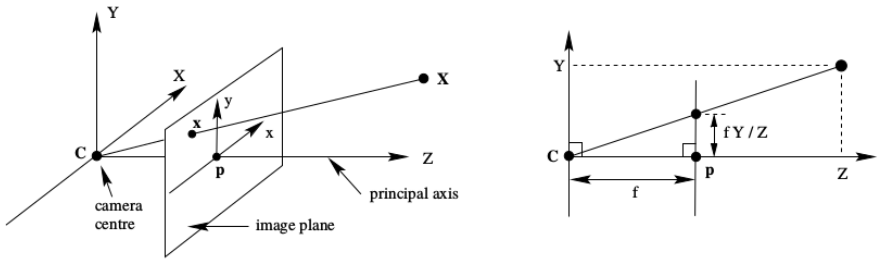


Figure 6.1: Pinhol camera geometry. C is the camera centre and p the principal point. The camera centre is here placed at the coordinate origin. Note the image plane is placed in front of the camera centre.

By similar triangles, computes that the point $(X, Y, Z)^T$ is mapped to the point $(fX/Z, fY/Z, f)^T$ on the image plane. Ignoring the final image coordinate, we see that The centre of projection is called the camera centre. It is also known as the optical centre. The line from the camera centre perpendicular to the image plane is called the principal axis or principal ray of the camera, and the point where the principal axis meets the image plane is called the principal point. The plane through the camera centre parallel to the image plane is called the principal plane of the camera.

6.1.1 Intrinsic parameters.

Central projection using homogeneous coordinates. If the world and image points are represented by homogeneous vectors, then central projection is very simply expressed as a linear mapping between their homogeneous coordinates. In particular, may be written in terms of matrix multiplication as:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} \tag{6.1}$$

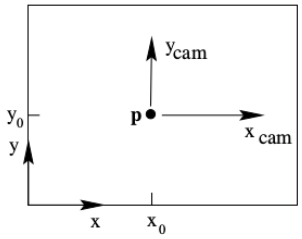


Figure 6.2: Image (x, y) and camera $(x_{cam}\ ,\ y_{cam}\ )$ coordinate systems

Principal point offset. The origin of coordinates in the image plane is at the principal point. In practice, it may not be, so that in general there is a mapping where $(p_x, p_y)^T$ are the coordinates of the principal point:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & & p_x \\ & f & & p_y \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} \tag{6.2}$$

Now writing

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \tag{6.3}$$

The matrix K is called the camera calibration matrix. In 6.3 we have written $(X, Y, Z, 1)^T$ as X_{cam} to emphasize that the camera is assumed to be located at the origin of a Euclidean coordinate system with the principal axis of the camera pointing straight down the Z -axis, and the point X_{cam}

is expressed in this coordinate system. Such a coordinate system may be called the camera coordinate frame.

Radial distortion. The assumption throughout these chapters has been that a linear model is an accurate model of the imaging process. Thus the world point, image point and optical centre are collinear, and world lines are imaged as lines and so on. For real (non-pinhole) lenses this assumption will not hold. The most important deviation is generally a radial distortion. In practice this error becomes more significant as the focal length (and price) of the lens decreases. Lens distortion takes place during the initial projection of the world onto the image plane. The actual projected point is related to the ideal point by a radial displacement. Thus, radial (lens) distortion is modelled as

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = L(\bar{r}) \begin{bmatrix} x \\ y \end{bmatrix} \tag{6.4}$$

where:

- (x, y) is the ideal image position (which obeys linear projection).
- (x_d, y_d) is the actual image position, after radial distortion.
- $\bar{r}$ is a radial distance $\sqrt{(x^2 + y^2)}$ from the center for radial distortion.
- $L(\bar{r})$ is a distortion factor, which is a function of the radius $\bar{r}$ only.

In pixel coordinates the correction is written:

$$\hat{x} = x_c + L(r)(x - x_c)\ \hat{y} = y_c + L(r)(y - y_c) \tag{6.5}$$

where (x, y) are the measured coordinates, $(\hat{x}\hat{y})$ are the corrected coordinates, and (x_c, y_c) is the centre of radial distortion, with $r^2 = (x - x_c)^2 + (y - y_c)^2$. Note, if the aspect ratio is not unity then it is necessary to correct for this when computing r. With this correction the coordinates $(\hat{x}, \hat{y})$ are related to the coordinates of the 3D world point by a linear projective camera.

$$x = K[I|0]X_{cam} \tag{6.6}$$

6.1.2 Extrinsic parameters.

These kind of parameters known also as external camera parameters describe a transformation between the unknown cameras reference frames and the known world reference frame. This is referred to translation vector between the relative position of the origins of the two cameras and rotation matrix

that can cover the corresponding axes of the two frames into alignment. In general, points in space will be expressed in terms of a different Euclidean coordinate frame, known as the world coordinate frame. The two coordinate frames are related via a rotation and a translation. See Fig 6.1 if $\bar{X}$ is an inhomogeneous 3-vector representing the coordinates of a point in the world coordinate frame, and $\bar{X}_{cam}$ represents the same point in the camera coordinate frame, then we may write $\bar{X}_{cam} = R(\bar{X} - \bar{C})$ where $\bar{C}$ represent the coordinates of the camera center in the world coordinate frame, and R is a $3x3$ rotation matrix representing the orientation of the camera coordinate frame. This equation mat be written in homogeneous coordinates as

$$X_{cam} = \begin{bmatrix} R & -R\bar{C} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} R & -R\bar{C} \\ 0 & 1 \end{bmatrix} X \tag{6.7}$$

Putting this together with 6.6 leads to the formula

$$x = KR[I| - \bar{C}]X \tag{6.8}$$

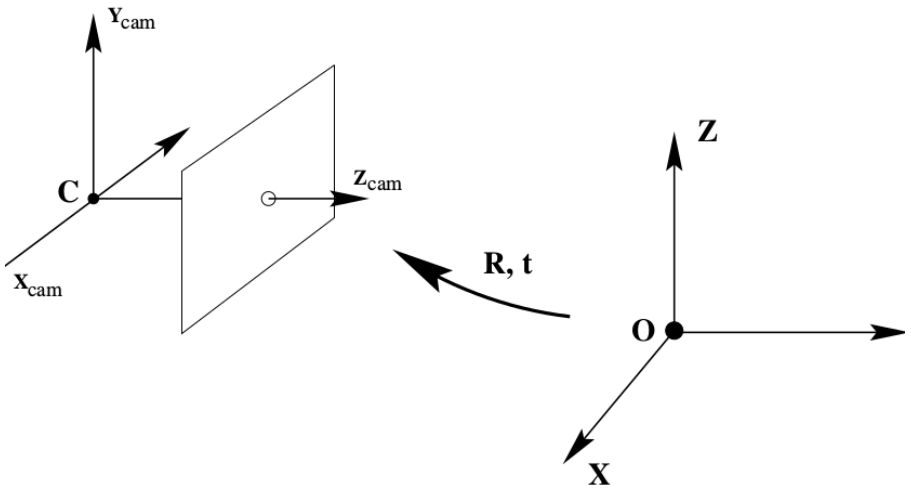


Figure 6.3: The Euclidean transformation between the world and camera coordinate frames.

where X is now in a world coordinate frame. This is the general mapping given by a pinhole camera. One sees that a general pinhole camera, $P = KR[I\| - \bar{C}]$ has 9 degrees of freedom: 3 for K (the elements f, p_x , p_y), 3 for R, and 3 for $\bar{C}$. The parameters contained in K are called the internal camera parameters, or the *internal orientation* of the camera. The parameters of R and C which relate the camera orientation and of the camera position to a world coordinate system are called the external parameters or the *exterior orientation*.

It is often convenient not to make the camera centre explicit, and instead to represent the world to

image transformation as $\bar{X}_{cam} = R\bar{X} + t$. In this case the camera matrix is simply:

$$P = K[R|t] \tag{6.9}$$

where $t = -R\bar{C}$

Camera rotation and translation for stereo cameras. In scenarios where there are two cameras, the required knowledge includes intrinsic parameters for both cameras and extrinsic parameters between the two cameras. Distances in space will be expressed in form of another Euclidean coordinate frame, known as the world coordinate frame. Two frames of coordinates are linked by rotation R and translation T. Fig 6.4 illustrates that the R and T describes pose of the right camera in the left frame camera.

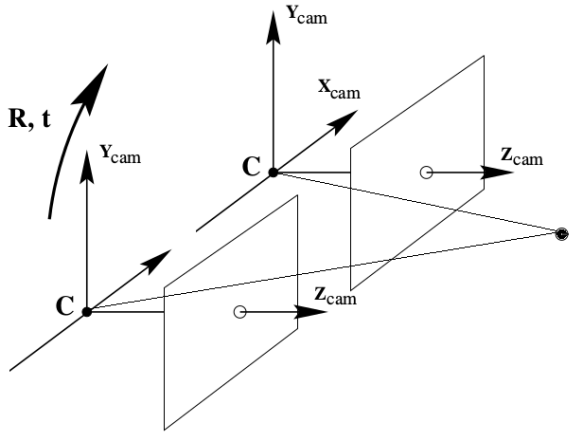


Figure 6.4: The Euclidean transformation between the one camera coordinate frame and second camera coordinate frames.

6.2 Camera projections

The drop from three-dimensional world to a two-dimensional image is a projection process in which we lose one dimension. The usual way of modelling this process is by central projection in which a ray from a point in space is drawn from a 3D world point through a fixed point in space, the centre of projection. This ray will intersect a specific plane in space chosen as the image plane. The intersection of the ray with the image plane represents the image of the point. If the 3D structure lies on a plane then there is no drop in dimension.

This model is in accord with a simple model of a camera, in which a ray of light from a point in the world passes through the lens of a camera and impinges on a film or digital device, producing an image of the point. Ignoring such effects as focus and lens thickness, a reasonable approximation is that all the rays pass through a single point, the centre of the lens.

This matrix P is known as the camera matrix. In summary, the action of a projective camera on

a point in space may be expressed in terms of a linear mapping of homogeneous coordinates as"

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = P_{3x4} \begin{bmatrix} X \\ Y \\ Z \\ T \end{bmatrix}$$

Furthermore, if all the points lie on a plane (we may choose this as the plane $Z = 0$ then the linear mapping reduces to

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = H_{3x3} \begin{bmatrix} X \\ Y \\ T \end{bmatrix}$$

which is a projective transformation.

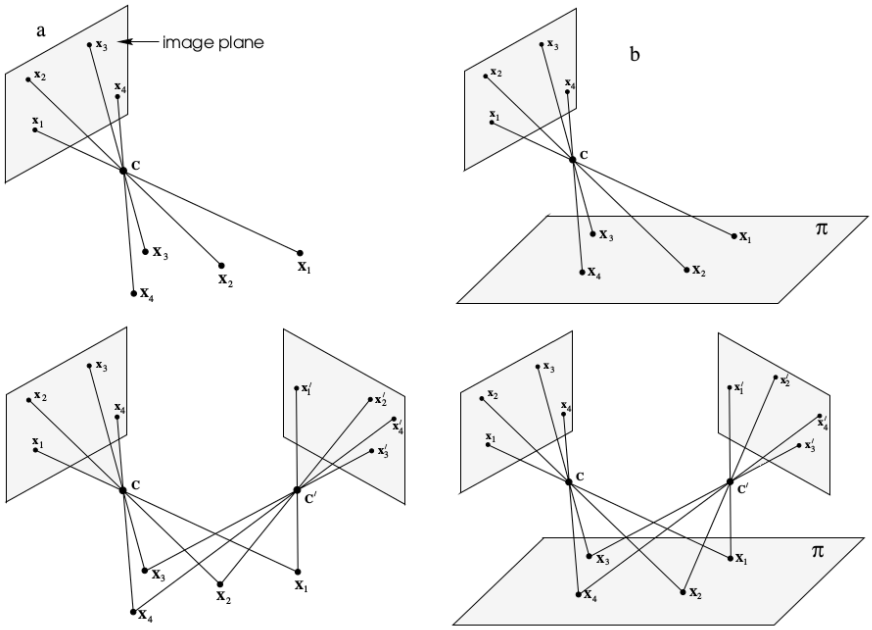


Figure 6.5: Projective transformation between the world space points X (left) or the world plane (right) to the image planes.

6.3 Epipolar geometry

The epipolar geometry between two views is essentially the geometry of the intersection of the image planes with the pencil of planes having the baseline as axis (the baseline is the line joining the camera centres). This geometry is usually motivated by considering the search for corresponding points in stereo matching. It is independent of scene structure, and only depends on the cameras internal parameters and relative pose. The F encapsulates this intrinsic geometry. It is a 3 x 3 matrix of rank 2. If a point in 3-space X is imaged as x in the first view, and x' in the second, then the image

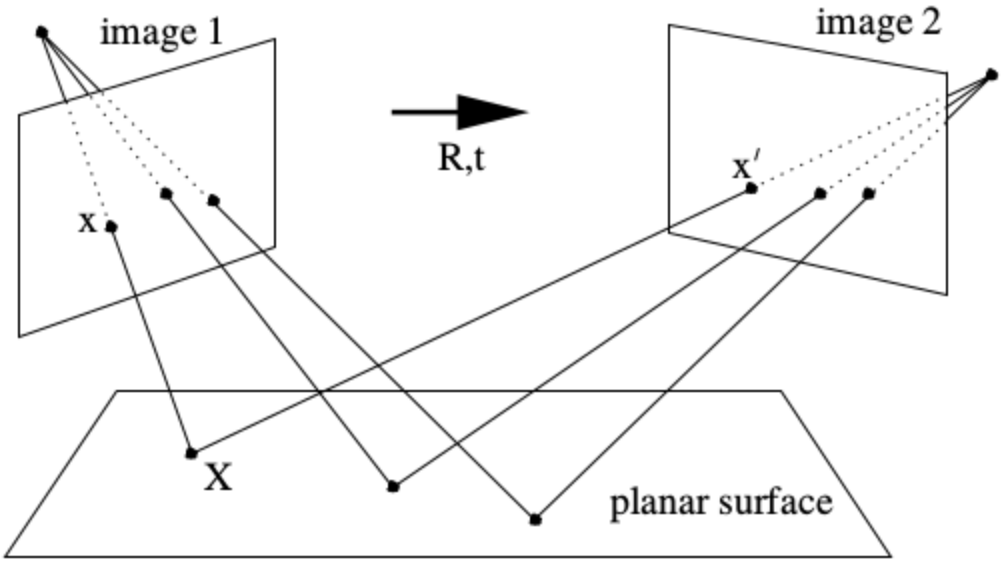


Figure 6.6: The camera centre is the essence, all the space points are coplanar.

points satisfy the relation $x'^T Fx = 0$. The F is independent of scene structure. However, it can be computed from correspondences of imaged scene points alone, without requiring knowledge of the cameras internal parameters or relative pose.

Suppose a point X in 3-space is imaged in two views, at x in the first, and x' in the second. What is the relation between the corresponding image points x and x' ? As shown in Fig 6.7 the image points x and x', space point X, and camera centers are coplanar. Denote this plane as π. Clearly, the rays back-projected from x and x' intersect at X , and the rays are coplanar, lying in π. It is this latter property that is of most significance in searching for a correspondence.

Supposing now that we know only x, we may ask how the corresponding point x' is constrained. The plane π is determined by the baseline and the ray defined by x. From above we know that the ray corresponding to the (unknown) point x' lies in π, hence the point x' lies on the line of intersection l' of π with the second image plane. This line l' is the image in the second view of the ray back-projected from x. It is the epipolar line corresponding to x. In terms of a stereo correspondence algorithm the benefit is that the search for the point corresponding to x need not cover the entire image plane but can be restricted to the line l'.

The Epipol is the point of intersection of the line joining the camera centers (the baseline) with the image plane. Equivalently, the epipole is the image in one view of the camera center of the other view. It is also the vanishing point of the baseline (translation) direction.

An epipolar plane is a plane containing the baseline. There is a one-parameter family (a pencil) of epipolar planes.

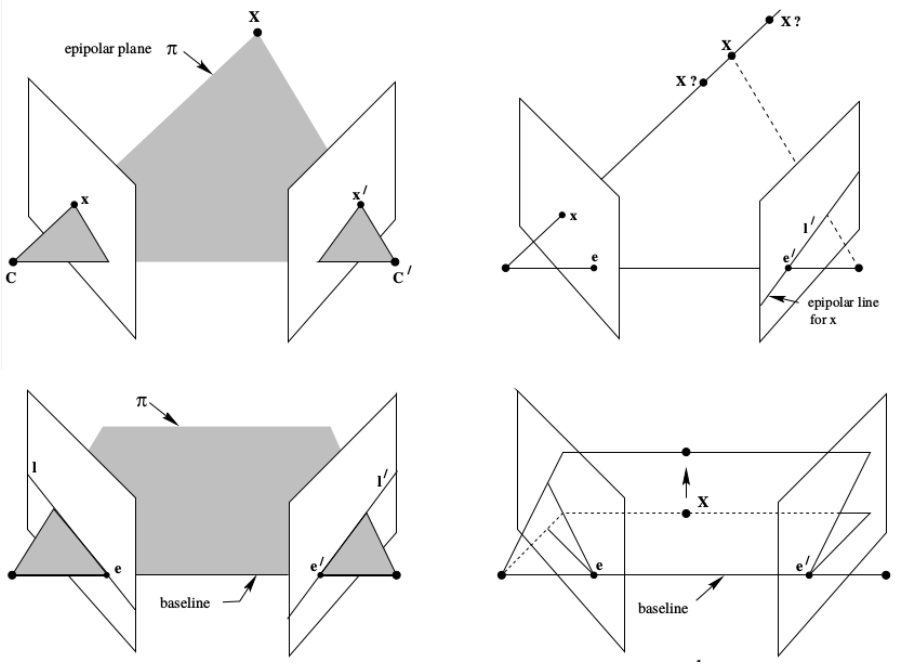


Figure 6.7: Point correspondence geometry and Epipolar geometry.

An epipolar line is the intersection of an epipolar plane with the image plane. All epipolar lines intersect at the epipole. An epipolar plane intersects the left and right image planes in epipolar lines, and defines the correspondence between the lines.

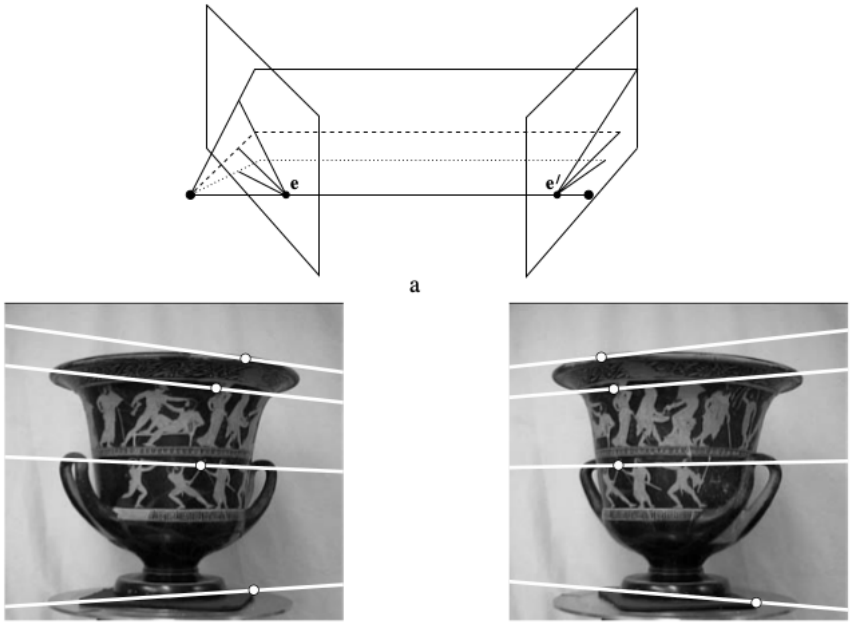


Figure 6.8: Converging cameras.

6.3.1 Properties of the F

Suppose we have two images acquired by cameras with non-coincident centres, then the F F is the unique $3x3$ rank 2 homogeneous matrix which satisfies equation 2.1 .

- Transpose: If F is the F of the pair of cameras (P, P'), then F^T is the F of the pair in the opposite order: (P', P).

- Epipolar lines: For any point x in the first image, the corresponding epipolar line is $l' = Fx$. Similarly, $l = F^T x'$ represents the epipolar line corresponding to x' in the second image

- The epipole: for any point x (other than e) the epipolar line $l' = Fx$ contains the epipole e'. Thus e' satisfies $e'^T(Fx) = (e'^T F)x = 0$ for all x. It follows that $e'^T F = 0$, i.e. e' is the left null-vector of F. Similarly $Fe = 0$, i.e. e is the right null-vector of F.

- The degree of freedome: F is a rank 2 homogeneous matrix and has seven degrees of freedom: a $3x3$ homogeneous matrix has eight independent ratios (there are nine elements, and the common scaling is not significant), however, F also satisfies the constraint det F = 0 which removes one degree of freedom.

6.3.2 Properties of the E

The E, has only five degrees of freedom: both the R matrix R and the T t have three degrees of freedom, but there is an overall scale ambiguity – like the F, the E is a homogeneousquantity. A $3x3$ matrix is an E if and only if two of its singular values are equal, and the third is zero. Once it is known, the camera matrices may be retrieved from E. It can be assumed that the first camera matrix is $P = [I|0]$. In order to compute the second camera matrix, P' , it is necessary to factor E into the product SR of a skew-symmetric matrix and R.

$$W = \begin{vmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{vmatrix} \quad Z = \begin{vmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix} \tag{6.10}$$

Suppose that the Singular Value Decomposition (SVD) of E is $U\ diag(1,1,0)V^T$. Using the notation of 6.10 there are (ignoring signs) two possible factorizations $E = SR$ as follows:

$$S = UZU^T \ R = UWV^T \tag{6.11}$$

For a given E $E = Udiag(1,1,0)V^T$, and first camera matrix $P = [I|0]$, there are four possible choices for the second camera matrix P', namely

$$P_1' = [UWU^T|u_3]\ P_2' = [UWV^T|-u_3] \qquad P_3' = [UW^TV^T|u_3]\ P_4' = [UW^TV^T|-u_3] \tag{6.12}$$

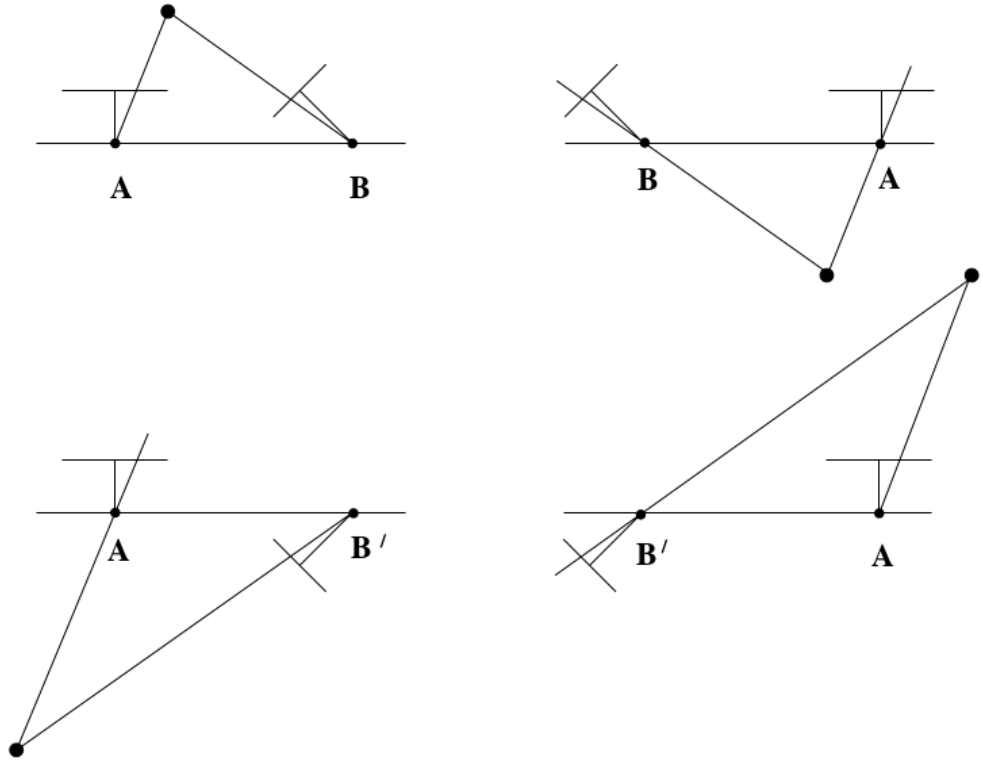


Figure 6.9: The four possible solutions for calibrated reconstruction from E.

The four solutions are illustrated in Fig above 6.9, where it is shown that a reconstructed point X will be in front of both cameras in one of these four solutions only. Thus, testing with a single point to determine if it is in front of both cameras is sufficient to decide between the four different solutions for the camera matrix P'.

Glossary

θ Error of Rotation represented in θ. 33

e_1 Error of Translation represented in e_1. 33

8PA Eight Point Algorithm. 33

ADAS Advanced Driver Assistance System. 18, 19, 21, 33

AI Artificial Intelligence. 18, 33

ARM Advanced RISC Machine, originally Acorn RISC Machine. 18, 20, 24, 26, 27, 33

ASCI Application-Specific Integrated Circuit. 24, 33

BA Bundle Adjustment. 33

CEA Commissariat a l'energie atomique et aux energies alternatives. 33

CMOS Complementary Metal-Oxide Semiconductor. 4, 9, 10, 33

CPS Cyber Physical System. 4–6, 9, 16–19, 25, 26, 28–33

CPU Central Processing Unit. 18, 25, 26, 28, 32, 33

dataset Collection of similar and related data for processing by computer. 33

DC Direct Current. 19, 33

E Essential Matrix. 33

epipolar error Epipolar Error. 33

epipolar geometry The geometry of stereo vision. 33

F Fundamental Matrix. 33

FPGA Field-Programmable Gate Array. 24, 33

FPS Frame Per Second. 33

GNSS Global Navigation Satellite System. 19, 33

GPS Global Positioning System. 6–8, 11, 19, 29, 33

GPU Graphics Processing Unit. 25, 28, 33

HPC High Performance Computing. 33

ID Identification Data. 33

IEKF Iterated Extended Kalman Filter. 33

IMU Inertial Measurement Unit. 6, 11, 19, 29, 33

inlier Point satisfies equation $x'_{left} M_{odel} x_{right}$ <threshold. 33

IOT Internet of Things. 4, 33

KF Kalman Filter. 33

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago. 33

LIDAR Light Detection and Ranging. 7, 9, 11, 29, 33

MPOI Map Points of Interest. 33

ORB Oriented FAST and Rotated BRIEF. 33

OS Operating System. 26, 33

OSCC Online Stereo Camera Calibration. 33

outlier Point satisfies equation $x'_{left} M_{odel} x_{right}$ > threshold. 33

PC Personal Computer. 25, 32, 33

POI Points of Interest. 33

QoS Quality of Service. 33

R Rotation Matrix between cameras - Stereo extrinsic parameter. 33

RAM Random-Access Memory. 33

RANSAC Random Sample Consensus. 33

RGB Red Green Blue. 33

RISC Reduced Instruction Set Computing. 26, 33

ROS Robot Operating System. 33

RPi Raspberry Pi. 9, 27, 33

RTOS Real-Time Operating System. 24, 33

SCCM Stereo Camera Calibration Monitoring. 33

SLAM Simultaneous Localization And Mapping. 30, 33

SOC System on a Chip. 24, 33

SSH Secure Shell. 33

SURF Speeded-Up Robust Features. 33

SVD Singular Value Decomposition. 33

SWAP Virtual Memory. 33

T Translation, Euclidean distance between cameras - Stereo extrinsic parameter. 33

TED Technology, Entertainment, Design. 19, 33

threshold The point that must be exceeded to begin producing a given effect or result or to elicit a response. 33

VO Visual Odometry. 33

WHO World Health Organization. 18, 33

[title = Glossaire]

Glossary

θ Error of Rotation represented in θ. 33

e_1 Error of Translation represented in e_1. 33

8PA Eight Point Algorithm. 33

ADAS Advanced Driver Assistance System. 18, 19, 21, 33

AI Artificial Intelligence. 18, 33

ARM Advanced RISC Machine, originally Acorn RISC Machine. 18, 20, 24, 26, 27, 33

ASCI Application-Specific Integrated Circuit. 24, 33

BA Bundle Adjustment. 33

CEA Commissariat a l'energie atomique et aux energies alternatives. 33

CMOS Complementary Metal-Oxide Semiconductor. 4, 9, 10, 33

CPS Cyber Physical System. 4–6, 9, 16–19, 25, 26, 28–33

CPU Central Processing Unit. 18, 25, 26, 28, 32, 33

dataset Collection of similar and related data for processing by computer. 33

DC Direct Current. 19, 33

E Essential Matrix. 33

epipolar error Epipolar Error. 33

epipolar geometry The geometry of stereo vision. 33

F Fundamental Matrix. 33

FPGA Field-Programmable Gate Array. 24, 33

FPS Frame Per Second. 33

GNSS Global Navigation Satellite System. 19, 33

GPS Global Positioning System. 6–8, 11, 19, 29, 33

GPU Graphics Processing Unit. 25, 28, 33

HPC High Performance Computing. 33

ID Identification Data. 33

IEKF Iterated Extended Kalman Filter. 33

IMU Inertial Measurement Unit. 6, 11, 19, 29, 33

inlier Point satisfies equation $x_{left}^{'} M_{odel} x_{right}$ <threshold. 33

IOT Internet of Things. 4, 33

KF Kalman Filter. 33

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago. 33

LIDAR Light Detection and Ranging. 7, 9, 11, 29, 33

MPOI Map Points of Interest. 33

ORB Oriented FAST and Rotated BRIEF. 33

OS Operating System. 26, 33

OSCC Online Stereo Camera Calibration. 33

outlier Point satisfies equation $x_{left}^{'} M_{odel} x_{right}$ > threshold. 33

PC Personal Computer. 25, 32, 33

POI Points of Interest. 33

QoS Quality of Service. 33

R Rotation Matrix between cameras - Stereo extrinsic parameter. 33

RAM Random-Access Memory. 33

RANSAC Random Sample Consensus. 33

RGB Red Green Blue. 33

RISC Reduced Instruction Set Computing. 26, 33

ROS Robot Operating System. 33

RPi Raspberry Pi. 9, 27, 33

RTOS Real-Time Operating System. 24, 33

SCCM Stereo Camera Calibration Monitoring. 33

SLAM Simultaneous Localization And Mapping. 30, 33

SOC System on a Chip. 24, 33

SSH Secure Shell. 33

SURF Speeded-Up Robust Features. 33

SVD Singular Value Decomposition. 33

SWAP Virtual Memory. 33

T Translation, Euclidean distance between cameras - Stereo extrinsic parameter. 33

TED Technology, Entertainment, Design. 19, 33

threshold The point that must be exceeded to begin producing a given effect or result or to elicit a response. 33

VO Visual Odometry. 33

WHO World Health Organization. 18, 33

Bibliography

[1] Asim qureshi "10 best gps modules for engineers and hobbyists" wonderfulengineering.com, 2015.

[2] European commission cyber-physical european roadmap & strategy, 2015.

[3] International technology roadmap for semiconductors 2.0, 2015.

[4] Kinect for windows features microsoft.com, 2015.

[5] 3d-sensor asus xtion pro asus.com, 2018.

[6] Blaxtair perimeter protection devices around machinery blaxtair.com, 2018.

[7] Fintan corrigan, 12 top lidar sensors for uavs and so many great uses dronezon.com, 2018.

[8] Stereolabs zed camera stereolabs.com, 2018.

[9] Farhan Ahammed, Javid Taheri, and Albert Zomaya. Lica: Robust localization using cluster analysis to improve gps coordinates. In *Proceedings of the First ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*, DIVANet '11, pages 39–46, New York, NY, USA, 2011. ACM.

[10] M. Alemán-Flores, L. Alvarez, L. Gomez, P. Henriquez, and L. Mazorra. Camera calibration in sport event scenarios. *Pattern Recognition*, 47(1):89 – 95, 2014.

[11] Gianluca Antonelli, Fabrizio Caccavale, Flavio Grossi, and Alessandro Marino. Simultaneous calibration of odometry and camera for a differential drive mobile robot. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5417–5422, 05 2010.

[12] Pathum Rathnayaka. Seung-Hae Baek. and Soon-Yong Park. An efficient calibration method for a stereo camera system with heterogeneous lenses using an embedded checkerboard pattern. *Journal of Sensors*, 2017.

[13] Kirk Baker. Singular value decomposition tutorial. *Singular Value Decomposition Tutorial*, 2013.

[14] Tayeb Basta. Is the fundamental matrix really independent of the scene structure? In *international journal of signal processing, image processing and pattern recognition*, volume 7, pages 149–167, 10 2014.

[15] Tayeb Basta. The eight-point algorithm is not in need of defense. In *ARPN Journal of Engineering and Applied Sciences*, volume 12, 02 2017.

[16] Mary Bellis. "who invented the odometer?". *ThoughtCo*, June 2018.

[17] Abhishek Bhattacharjee and Daniel Lustig. *Architectural and Operating System Support for Virtual Memory*. Morgan & Claypool Publishers, 2017.

[18] M. Bjorkman and J. O. Eklundh. Real-time epipolar geometry estimation of binocular stereo heads. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):425–432, Mar 2002.

[19] Randolph Blake and Hugh Wilson. Binocular vision. *Vision Research*, 51(7):754 – 770, 2011. Vision Research 50th Anniversary Issue: Part 1.

[20] Emily Blem, Jaikrishnan Menon, and Karthikeyan Sankaralingam. A detailed analysis of contemporary arm and x86 architectures. Technical report, 2013.

[21] F. Brenot, P. Fillatreau, and J. Piat. Fpga based accelerator for visual features detection. In *2015 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*, pages 1–6, June 2015.

[22] J. H. Brito, R. Angst, K. Köser, and M. Pollefeys. Radial distortion self-calibration. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1368–1375, June 2013.

[23] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.

[24] B. Caprile and V. Torre. Using vanishing points for camera calibration. *Int. J. Comput. Vision*, 4(2):127–140, May 1990.

[25] G. Carrera, A. Angeli, and A.J. Davison. Slam-based automatic extrinsic calibration of a multi-camera rig. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2652–2659, May 2011.

[26] Gerardo Carrera. *Robot SLAM and Navigation with Multi-Camera Computer Vision*. PhD thesis, IMPERIAL COLLEGE LONDON Department of Computing, March 2012.

[27] Andrea Censi, Antonio Franchi, Luca Marchionni, and Giuseppe Oriolo. Simultaneous calibration of odometry and sensor parameters for mobile robots. In *Robotics, IEEE Transactions on*, volume 29, pages 475–492, April 2013.

[28] Lei Cheng, Fuchao Wu, Zhanyi Hu, and Hung-Tat Tsui. A new approach to solving kruppa equations for camera self-calibration. In *Proceedings of the 16th International Conference on Pattern Recognition, Volume 2*, pages 308–311. IEEE Computer Society, 2002.

[29] Wojciech Chojnacki, Michael J. Brooks, Anton van den Hengel, and Darren Gawley. Revisiting hartley's normalized eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1172–1177, 2003.

[30] J. Civera, D. R. Bueno, A. J. Davison, and J. M. M. Montiel. Camera self-calibration for sequential bayesian structure from motion. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 403–408, May 2009.

[31] D.M. Cole and P.M. Newman. Using laser range data for 3d slam in outdoor environments. *IEEE International Conference on Robotics and Automation*, 2006:1556 – 1563, 06 2006.

[32] J. M. Collado, C. Hilario, A. de la Escalera, and J. M. Armingol. Self-calibration of an on-board stereo-vision system for driver assistance systems. In *2006 IEEE Intelligent Vehicles Symposium*, pages 156–162, 2006.

[33] Peter Corke, Jorge Lobo, and Jorge Dias. An introduction to inertial and visual sensing. IEEE International Conference on Robotics and Automation, 06 2007.

[34] Rachel Courtland. Transistors could stop shrinking in 2021. *spectrum.ieee.org/semiconductors*, 2016.

[35] James L Crowley and Yves Demazeau. Principles and techniques for sensor data fusion. *LIFIA (IMAG)*, 2002.

[36] T. Dang and C. Hoffmann. Stereo calibration in vehicles. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 268–273, June 2004.

[37] T. Dang, C. Hoffmann, and C. Stiller. Continuous stereo self-calibration by camera parameter tracking. *Image Processing, IEEE Transactions on*, 18(7):1536–1550, July 2009.

[38] Thao Dang, C. Hoffmann, and C. Stiller. Self-calibration for active automotive stereo vision. In *Intelligent Vehicles Symposium, 2006 IEEE*, pages 364–369, 2006.

[39] Richard Szeliski Daniel Scharstein. A taxonomy and evaluation of dense two-frame stereo correspondence algorithm. In *Stereo and Multi-Baseline Vision*, 2001.

[40] Rob Fergus David Eigen, Christian Puhrsch. Depth map prediction from a single image using a multi-scale deep network. In *Computer Vision and Pattern Recognition*, 2014.

[41] M.B. de Paula, C.R. Jung, and L.G. da Silveira. Automatic on-the-fly extrinsic camera calibration of onboard vehicular cameras. *Expert Systems with Applications*, 41(4, Part 2):1997 – 2007, 2014.

[42] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *Intl. J. of Robotics Research (IJRR)*, 25(12):1181–1204, December 2006.

[43] ARM developer. Gnu arm embedded toolchain. In *https://developer.arm.com/open-source/gnu-toolchain/gnu-rm*, 2018.

[44] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[45] Chapuis Roland Dziri Aziz, Duranton Marc. Real-time multiple objects tracking on raspberry-pi-based smart embedded camera. In *Journal of Electronic Imaging. 25. 041005. 10.1117/1.JEI.25.4.041005.*, 2016.

[46] Philipp Bender Eike Rehder, Christian Kinzig and Martin Lauer. Online stereo camera calibration from scratch. In *Online Stereo Camera Calibration From Scratch*, 2017.

[47] Wilfried Elmenreich. An introduction to sensor fusion. Research Report 47/2001, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2001.

[48] Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, and Doug Burger. Dark silicon and the end of multicore scaling. In *Proceedings of the 38th Annual International Symposium on Computer Architecture*, ISCA '11, pages 365–376, New York, NY, USA, 2011. ACM.

[49] Olivier Faugeras. *Three-dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, Cambridge, MA, USA, 1993.

[50] Yuxiang Feng, Simon Pickering, Edward Chappell, Pejman Iravani, and Christian Brace. Distance estimation by fusing radar and monocular camera with kalman filter. In *SAE Intelligent and Connected Vehicles Symposium*, USA United States, 9 2017. SAE International.

[51] M. Fleps, E. Mair, O. Ruepp, M. Suppa, and D. Burschka. Optimization based imu camera calibration. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3297–3304, Sept 2011.

[52] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. *Proceedings - IEEE International Conference on Robotics and Automation*, 2014.

[53] E. R. Fossum. Cmos image sensors: electronic camera-on-a-chip. *IEEE Transactions on Electron Devices*, 44(10):1689–1698, Oct 1997.

[54] Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry: Part ii - matching, robustness, and applications. In *IEEE Robotics & Automation Magazine - IEEE ROBOT AUTOMAT*, volume 19, pages 78–90, 06 2012.

[55] Amy French. Nvidia brings artificial intrlligence to automobiles. www.auvsi.org, 03 2018.

[56] Michał Fularz, Marek Kraft, Adam Schmidt, and Andrzej Kasiński. The architecture of an embedded smart camera for intelligent inspection and surveillance. In Roman Szewczyk, Cezary Zieliński, and Małgorzata Kaliczyńska, editors, *Progress in Automation, Robotics and Measuring Techniques*, pages 43–52, Cham, 2015. Springer International Publishing.

[57] Wei P. Cagle L. Reza T. Ball J. Gafford. Lidar and camera detection fusion in a real-time industrial multi-sensor collision avoidance system. *MDPI and ACS Style Electronics*, 2018.

[58] A. Geiger, F. Moosmann, O. Car, and B. Schuster. Automatic camera and range sensor calibration using a single shot. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3936–3943, May 2012.

[59] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[60] Arren Glover, Will Maddern, Michael Milford, and Gordon Wyeth. Fab-map + ratslam: Appearance-based slam for multiple times of day. In *ICRA*, Anchorage, USA, 2010.

[61] Lazaros Grammatikopoulos, George Karras, and Elli Petsa. An automatic approach for camera calibration from vanishing points. *{ISPRS} Journal of Photogrammetry and Remote Sensing*, 62(1):64 – 76, 2007.

[62] Ari Grobman. Smart-glasses company. Technical report, lumus, 2018.

[63] Antonio Guiducci. Camera calibration for road applications. *Computer Vision and Image Understanding*, 79(2):250 – 266, 2000.

[64] Volkan Gunes, Steffen Peter, Tony Givargis, and Frank Vahid. A survey on concepts, applications, and challenges in cyber-physical systems. 8:4242–4268, 12 2014.

[65] C.X. Guo, F.M. Mirzaei, and S.I. Roumeliotis. An analytical least-squares solution to the odometer-camera extrinsic calibration problem. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3962–3968, May 2012.

[66] Jussi Hakala, Stina Westman, Marja Salmimaa, Monika Pölönen, Toni Järvenpää, and Jukka Häkkinen. Why 3d cameras are not popular: A qualitative user study on stereoscopic photography acceptance. *3D Research*, 5(1):4, Jan 2014.

[67] Peter Hansen, Hatem Said Alismail, Peter Rander, and Brett Browning . Online continuous stereo extrinsic parameter estimation. *unknown*, June 2012.

[68] R.I. Hartley. In defence of the 8-point algorithm. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 1064–1070, Jun 1995.

[69] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[70] C. J. Hegarty and E. Chatre. Evolution of the global navigation satellitesystem (gnss). *Proceedings of the IEEE*, 96(12):1902–1917, Dec 2008.

[71] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 1106–, Washington, DC, USA, 1997. IEEE Computer Society.

[72] E. E. Hemayed. A survey of camera self-calibration. In *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.*, pages 351–357, July 2003.

[73] L. Heng, M. Burki, Gim Hee Lee, P. Furgale, R. Siegwart, and M. Pollefeys. Infrastructure-based calibration of a multi-camera rig. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4912–4919, May 2014.

[74] L. Heng, Bo Li, and M. Pollefeys. Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1793–1800, Nov 2013.

[75] Hann Woei Ho, Guido Croon, and Q Chu. Distance and velocity estimation using optical flow from a monocular camera. 10 2016.

[76] S. Hold, S. Gormer, A. Kummert, M. Meuter, and S. Muller-Schneiders. A novel approach for the online initial calibration of extrinsic parameters for a car-mounted camera. In *Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on*, pages 1–6, Oct 2009.

[77] Haosheng Huang and Georg Gartner. *A Survey of Mobile Indoor Navigation Systems*, pages 305–319. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[78] Wolfgang Hugemann. Correcting lens distortions in digital photographs. In *researchgate.net*, 08 2010.

[79] Du Q. Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, Oct 2009.

[80] Faugeras O. D. Luong Q. T. Maybank S. J. *Camera self-calibration: Theory and experiments*, chapter Camera self-calibration: Theory and experiments, pages 321–334. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992.

[81] JCGM. *International vocabulary of metrology — Basic and general concepts and associated terms (VIM)*. Joint Committee for Guides in Metrolog, 2008.

[82] D. Jiang and J. Yi. Comparison and study of classic feature point detection algorithm. In *2012 International Conference on Computer Science and Service System*, pages 2307–2309, Aug 2012.

[83] Abhishek Pandey Anirudh Kaushik Amit Kumar Jha Girish Kapse. A technological survey on autonomous home cleaning robots. *International Journal of Scientific and Research Publications (IJSRP)*, 4(4), April 2014.

[84] Jonathan Kelly, Larry H. Matthies, and Gaurav S. Sukhatme. Simultaneous mapping and stereo extrinsic parameter calibration using gps measurements. In *ICRA*, pages 279–286. IEEE, 2011.

[85] A. Khan, G. Aragon-Camarasa, L. Sun, and J. P. Siebert. On the calibration of active binocular and rgbd vision systems for dual-arm robots. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1960–1965, Dec 2016.

[86] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors (Basel, Switzerland) vol. 12,2 (2012): 1437-54.*, 2012.

[87] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. In *Proc. Eigth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, Orlando, October 2009.

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

[89] M. Knorr, W. Niehsen, and C. Stiller. Online extrinsic multi-camera calibration using ground plane induced homographies. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 236–241, June 2013.

[90] Kurt Konolige. Sparse sparse bundle adjustment. In *British Machine Vision Conference, BMVC 2010 - Proceedings*, pages 1–11, 01 2010.

[91] Shourjya Sanyal Koushik Kumar Nundy. A low cost vein detection system using integrable mobile camera devices. In *1-3. 10.1109/INDCON.2010.5712670*, 2010.

[92] A. Kuhn, L. Roth, J. M. Frahm, and H. Mayer. Improvement of extrinsic parameters from a single stereo pair. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1011–1019, March 2018.

[93] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4):387, Sep 2009.

[94] Philip Lamoureux. Numerical stability of the 8-point algorithm. Computer Vision Final Project, 2005.

[95] Edward A. Lee and Sanjit Seshia. *Introduction to Embedded Systems A Cyber Physical Systems Approach.* Published by authors, first edition edition, 2011. Solutions are available at http://chess.eecs.berkeley.edu/instructors/".

[96] K. Levenberg. A method for the solution of certain non-linear problems in least squares,". *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[97] Hongdong Li and R. Hartley. Five-point motion estimation made easy. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 1, pages 630–633, 2006.

[98] M. Li and A. I. Mourikis. 3-d motion estimation and online temporal calibration for camera-imu systems. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5709–5716, May 2013.

[99] Roland Li. Depth sensors are the key to unlocking next level computer vision applications. *Comet Labs Research Team*, 2017.

[100] David Liebowitz and Andrew Zisserman. Metric rectification for perspective images of planes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 482 – 488, 07 1998.

[101] Weikang Fang Yanjun Zhang Bo Yu Shaoshan Liu. Fpga-based orb feature extraction for real-time visual slam. *Institute of Application Specific Instruction Set Processor*, 2012.

[102] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. In Martin A. Fischler and Oscar Firschein, editors, *Readings in computer vision: issues, problems, principles, and paradigms*, chapter A Computer Algorithm for Reconstructing a Scene from Two Projections, pages 61–62. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.

[103] Manolis Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms inc/c++. levmar, 01 2004.

[104] Manolis Lourakis. A brief description of the levenberg-marquardt algorithm implemened by levmar. In *A Brief Description of the Levenberg-Marquardt Algorithm Implemented by Levmar*, volume 4, 01 2005.

[105] Manolis I.A. Lourakis and Rachid Deriche. Camera self-calibration using the kruppa equations and the svd of the fundamental matrix: The case of varying intrinsic parameters. Research Report RR-3911, INRIA, 2000.

[106] Manolis I. A. Lourkais and Antonis A. Arygros. Sba: A software package for genericsparse bundle adjustmen. *ACM Transactions on Mathematical Software, Vol. 36, No. 1, Article 2,*, 2009.

[107] T. Marita, F. Oniga, S. Nedevschi, T. Graf, and R. Schmidt. Camera calibration method for far range stereovision sensors used in vehicles. In *2006 IEEE Intelligent Vehicles Symposium*, pages 356–363, 2006.

[108] MATLAB. Image processing and computer vision. application to calibrate cameras.

[109] Brandon Bray Matt Zeller. Hololens hardware details. Technical report, Microsoft, 2018.

[110] Faessler Matthias, Fontana Flavio, Forster Christian, Mueggler Elias, Pizzoli Matia, and Scaramuzza Davide. Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4):431–450, 2015.

[111] Stephen J. Maybank and Olivier D. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[112] Jérôme Maye, Paul Furgale, and Roland Siegwart. Self-supervised calibration for robotic systems. In *IEEE Intelligent Vehicles Symposium, Proceedings*, 06 2013.

[113] mckinsey.com. Automotive revolution - perspective towards 2030. Technical report, mckinsey.com, 2016.

[114] Michael McWhertor. What's inside a kinect? kotaku.com, 04 2010.

[115] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A constant time efficient stereo slam system. In *BMVC*, 2009.

[116] M. Miksch, B. Yang, and K. Zimmermann. Automatic extrinsic camera self-calibration based on homography and epipolar geometry. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 832–839, June 2010.

[117] F. M. Mirzaei and S. I. Roumeliotis. A kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation. *IEEE Transactions on Robotics*, 24(5):1143–1156, Oct 2008.

[118] I. Miyagawa, H. Arai, and H. Koike. Simple camera calibration from a single image using five points on two orthogonal 1-d objects. *Image Processing, IEEE Transactions on*, 19(6):1528–1538, June 2010.

[119] Lionel Moisan and Bérenger Stival. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. In *International Journal of Computer Vision*, volume 57, pages 201–218, 05 2004.

[120] Gordon Moore. Moore's law. In Encyclopedia Britannica, 1965.

[121] N. Moutinho, R. Ferreira, J. Gaspar, A. Bernardino, and J. Santos-Victor. Markerless online stereo calibration for a humanoid robot. In *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, pages 454–460, Oct 2014.

[122] G. R. Mueller and H. J. Wuensche. Continuous extrinsic online calibration for stereo cameras. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 966–971, June 2016.

[123] Karsten Muhlmann, Dennis Maier, Jurgen Hesser, and Reinhard Manner. Calculating dense disparity maps from color stereo images, an efficient implementation. *International Journal of Computer Vision*, 47, 12 2002.

[124] Elon Musk. What will the future look like? TED conference, 2017.

[125] S. Nedevschi, C. Vancea, T. Marita, and T. Graf. Online extrinsic parameters calibration for stereovision systems used in far-range detection vehicle applications. *Intelligent Transportation Systems, IEEE Transactions on*, 8(4):651–660, Dec 2007.

[126] P. Nelson, W. Churchill, I. Posner, and P. Newman. From dusk till dawn: Localisation at night using artificial light sources. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 5245–5252, May 2015.

[127] National Geodetic Survey (NGS). National geodetic survey (ngs). Frequently Asked Questions FAQ icon.

[128] David Nister. An efficient solution to the five-point relative pose problem, 2004.

[129] Bob O'Donnell. Your average car is a lot more code-driven than you think. eu.usatoday.com, 06 2016.

[130] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[131] OpenCV. Open source computer vision library. https://opencv.org/.

[132] World Health Organization. Global health estimates 2016: Deaths by cause, age, sex, by country and by region, 2000-2016. Technical report, World Health Organization, May 2018.

[133] Frank. Pagel. Motion adjustment for extrinsic calibration of cameras with non-overlapping views. In *Computer and Robot Vision (CRV), 2012 Ninth Conference on*, pages 94–100, May 2012.

[134] J. Palacin, J. A. Salse, I. Valganon, and X. Clua. Building a mobile robot for a floor-cleaning operation in domestic environments. In *Proceedings of the 20th IEEE Instrumentation Technology Conference (Cat. No.03CH37412)*, volume 2, pages 1391–1396 vol.2, May 2003.

[135] Xiameng Qin, Jiaolong Yang, Wei Liang, Mingtao Pei, and Yunde Jia. Stereo camera calibration with an embedded calibration device and scene features. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, pages 2306–2310, Dec 2012.

[136] RaspberryPi. https://wiki.ros.org/rosberrypi. https://wiki.ros.org/ROSberryPi.

[137] raspberrypi. Kernel configuration. *www.raspberrypi.org*, 2015.

[138] Philipp Rauschnabel, Alexander Brem, and Young Ro. Augmented reality smart glasses: Definition, conceptual insights, and managerial importance. *Working Paper, The University of Michigan-Dearborn*, 07 2015.

[139] Antoni Rogalski. Infrared detectors: Status and trends. *Progress in Quantum Electronics*, 27:59–210, 12 2003.

[140] Haidi Ibrahim Rostam Affendi Hamzah. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016.

[141] M. Ruffo, M. Di Castro, L. Molinari, R. Losito, A. Masi, J. Kovermann, and Luis Rodrigues. New infrared time of-flight measurement sensors for robotic platform. 20th IMEKO TC4 International Symposium.

[142] Ewelina Rupnik, Mehdi Daakir, and Marc Pierrot Deseilligny. Micmac – a free, open-source solution for photogrammetry. *Open Geospatial Data, Software and Standards*, 2(1):14, Jun 2017.

[143] Ehab Salahat and Murad Qasaimeh. Recent advances in features extraction and description algorithms: A comprehensive survey. *CoRR*, abs/1703.06376, 2017.

[144] David Samper, Jorge Santolaria, Francisco Javier Brosed, Ana Cristina Majarena, and Juan José Aguilar. Analysis of tsai calibration method using two- and three-dimensional calibration objects. *Machine Vision and Applications*, 24(1):117–131, Jan 2013.

[145] A.D. Sappa, F. Dornaika, D. Ponsa, D. Geronimo, and A. Lopez. An efficient approach to onboard stereo vision system pose estimation. *Intelligent Transportation Systems, IEEE Transactions on*, 9(3):476–490, Sept 2008.

[146] A.D. Sappa, D. Geronimo, F. Dornaika, and A. Lopez. On-board camera extrinsic parameter estimation. *Electronics Letters*, 42(13):745–747, June 2006.

[147] Davide Scaramuzza. Tutorial on visual odometry. *IEEE Robotics & Automation Magazine*, 2011.

[148] Cameron Schaeffer. A comparison of keypoint descriptors in the context of pedestrian detection: Freak vs. surf vs. brisk. Stanford University CS Department.

[149] Stefan Schmerler and Robert Rimkus. Autosar — shaping the future of a global standard. *ATZelektronik worldwide*, 8(1):42–45, Feb 2013.

[150] Victor Hugo Schulz, Felipe Gustavo Bombardelli, and Eduardo Todt. A harris corner detector implementation in soc-fpga for visual slam. In Fernando Santos Osório and Rogério Sales Gonçalves, editors, *Robotics*, pages 57–71, Cham, 2016. Springer International Publishing.

[151] Amar Shan. Heterogeneous processing: A strategy for augmenting moore's law. *Linux J.*, 2006(142):7–, February 2006.

[152] Ling. Yonggen. Shen. Shaojie. High-precision online markerless stereo extrinsic calibration. *IEEE International Conference on Intelligent Robots and Systems*, 2016.

[153] Fangwu Shu, L. Toma, Werner Neddermeyer, and Jianwei Zhang. Precise online camera calibration in a robot navigating vision system. In *Mechatronics and Automation, 2005 IEEE International Conference*, volume 3, pages 1277–1282 Vol. 3, 2005.

[154] Gabe Sibley. Relative bundle adjustment. *Technical Report No. 2307/09*, 2009.

[155] Catharine Smith. Microsoft kinect: World's fastest-selling consumer electronics device. www.huffingtonpost.com, 09 2011.

[156] BOBBY YAZDANI SOULAIMAN ITANI. Smart-glasses company. Technical report, atheer, 2018.

[157] Robert Spangenberg, Tobias Langner, and Raúl Rojas. *Image Analysis: 18th Scandinavian Conference, SCIA 2013, Espoo, Finland, June 17-20, 2013. Proceedings*, chapter On-line Stereo Self-calibration through Minimization of Matching Costs, pages 545–554. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[158] Holger Steiner, Sebastian Sporrer, Andreas Kolb, and Norbert Jung. Design of an active multispectral swir camera system for skin detection and face verification. *Journal of Sensors*, 2016:1–16, 01 2016.

[159] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1100–1105, Jun 1997.

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

[161] Frédéric Sur, Nicolas Noury, and Marie-Odile Berger. Computing the uncertainty of the 8 point algorithm for fundamental matrix estimation. In Mark Everingham, Chris Needham, and Roberto Fraile, editors, *19th British Machine Vision Conference - BMVC 2008*, Electronic Proceedings of the 19th British Machine Vision Conferencee 2008 (Leeds, September 2008), page 10, Leeds, United Kingdom, September 2008. British Machine Vision Association.

[162] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. Visual slam algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1):16, Jun 2017.

[163] J. Tan, X. An, X. Xu, and H. He. Automatic extrinsic calibration for an onboard camera. In *2013 Chinese Automation Congress*, pages 340–343, Nov 2013.

[164] Jun Tan, Jian Li, Xiangjing An, and Hangen He. An interactive method for extrinsic parameter calibration of onboard camera. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 236–241, June 2011.

[165] Dengqing Tang, Tianjiang Hu, Lincheng Shen, Zhaowei Ma, and Congyu Pan. Apriltag array-aided extrinsic calibration of camera–laser multi-sensor system. *Robotics and Biomimetics*, 3(1):13, Jul 2016.

[166] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000.

[167] Philip H. S. Torr and Andrew Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000.

[168] Philip A. Tresadern and Ian D. Reid. Camera calibration from human motion. *Image Vision Comput.*, 26(6):851–862, June 2008.

[169] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 298–372, London, UK, UK, 2000. Springer-Verlag.

[170] R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 364–374, Miami, June 1986.

[171] Rao R. Tummala. Autonomous cars: Radar, lidar, stereo cameras. In *Georgia Institute of Technology*.

[172] Donald W. Marquardt. An algorithm for least square estimation of non-linear parameters. In *SIAM Journal on Applied Mathematics*, volume 11, pages 431–441, 06 1963.

[173] Chenxi Wang, Qing He, Ning Wei, Wei Liu, Chao Hu, and M. Q. H. Meng. A new calibration method used in the infrared ray environment. In *Information and Automation (ICIA), 2011 IEEE International Conference on*, pages 480–484, June 2011.

[174] Huawei Wang and Ying Sun. New stereovision self-calibration method and its application in vision guided approaching. In *Proceedings of the 5th International Conference on Intelligent Robotics and Applications - Volume Part III*, ICIRA'12, pages 532–541, Berlin, Heidelberg, 2012. Springer-Verlag.

[175] Pichao Wang, Wanqing Li, Zhimin Gao, Jing Zhang, Chang Tang, and Philip Ogunbona. Deep convolutional neural networks for action recognition using depth map sequences. 01 2015.

[176] M. Warren, D. McKinnon, and B. Upcroft. Online calibration of stereo rigs for long-term autonomy. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3692–3698, May 2013.

[177] Greg Welch and Gary Bishop. Siggraph 2001 course 8 an introduction to the kalman filter.

[178] J. Westerhoff, S. Lessmann, M. Meuter, and A. Kummert. A classification and temporal filtering based system for online extrinsic camera calibration. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1285–1290, Nov 2016.

[179] Catherine Wong. Ros camera calibration. wiki.ros.org, 2018.

[180] Anqi Xu. Api for ueye cameras by ids imaging development systems gmbh. https://wiki.ros.org/ueyecam, 2018.

[181] De Xu, Min Tan, and Gang Chen. An improved dead reckoning method for mobile robot with redundant odometry information. In *7th International Conference on Control, Automation, Robotics and Vision, 2002. ICARCV 2002.*, volume 2, pages 631–636 vol.2, Dec 2002.

[182] Donghao Xu, Qiqi Zeng, Huijing Zhao, Chunzhao Guo, K. Kidono, and Y. Kojima. Online stereovision calibration using on-road markings. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 245–252, Oct 2014.

[183] Atushi Yamashita, Toru Kaneko, Shinya Matsushita, Kenjiro T. Miura, and Suekichi Isogai. Camera calibration and 3-d measurement with an active stereo vision system for handling moving objects, 2002.

[184] Yongjie Yan, Qidan Zhu, Zhuang Lin, and Quanfu Chen. Camera calibration in binocular stereo vision of moving robot. In *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, volume 2, pages 9257–9261, 2006.

[185] Xiaochuan Yin, Xiangwei Wang, Xiaoguo Du, and Qijun Chen. Scale recovery for monocular visual odometry using depth estimated with deep convolutional neural fields. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[186] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, Dec 2015.

[187] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems Conference*, July 2014.

[188] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 666–673 vol.1, 1999.

[189] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, Nov 2000.

[190] Zhengyou Zhang. Camera calibration with one-dimensional objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):892–899, July 2004.

[191] C Zhaoxue and S Pengfei. Efficient method for camera calibration in traffic scenes. In *Electronics Letters*, volume 40, pages 368 – 369, 04 2004.