Calibration of robot vision systems for flexible assembly

Oleksandr Semeniuta

Master Thesis
Master in Sustainable Manufacturing
30 ECTS
Department of Technology, Economy and Management
Gjøvik University College, 2014

Avdeling for
teknologi, økonomi og ledelse
Høgskolen i Gjøvik
Postboks 191
2802 Gjøvik

Department of Technology,
Economy and Management
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Calibration of robot vision systems for flexible assembly

Oleksandr Semeniuta

2014/06/10

Abstract

In the contemporary competitive and fast-changing economy, the manufacturing enterprises require high degree of flexibility, to timely respond to the changing demands, and automation, to cope with the requirements of speed and quality. Industrial robots play a vital role in flexible assembly systems, and the application of machine vision in conjunction with the robotized systems constitutes a promising direction in the contemporary industrial automation. It is also of a big interest to understand the role of software in provision of assembly flexibility. In the area of robot vision systems calibration, more accurate, precise and repeatable calibration procedure are required.

This master thesis applies systems approach and is divided into two main themes, namely *Software-defined assembly flexibility* and *Robot vision systems calibration*. The first theme is aimed at understanding the concept of assembly flexibility and the role of software in its provision. This topic is considered on a high magnifying level, and is investigated by the means of literature study, systems models building and discussion. It is concluded that the there is a lack of common understanding and agreed-upon taxonomy on flexibility in manufacturing. Certain flexibility types are identified as having the biggest reliance on software. The notion of software-defined assembly flexibility is proposed. Machine vision and holonic manufacturing control are identified as the main enabling technologies of software-defined assembly flexibility.

The second theme is focused on robot vision systems calibration. It is viewed on a low magnifying level, and constitutes a practical undertaking aimed at studying the processes of camera calibration, stereo vision systems calibration and hand-eye calibration, and improving the quality of the processes by maximizing their accuracy, precision and repeatability. A Python library FlexVi is developed and used for studying the calibration processes from the perspective of interactive computing and data analysis. The outcomes of camera calibration are analyzed on the basis of a large number of calibration experiments with the subsequent distribution fitting to find the most frequent values. A method for stereo vision systems calibration aimed at achieving high repeatability is developed. A method for outliers elimination for hand-eye calibration process is developed based on the analysis of the resulting precision of the object-to-base transformations measurement.

Keywords: Industrial automation, Flexible Assembly Systems, Machine vision, Camera calibration, Hand-eye calibration

Acknowledgments

I would like to thank my supervisor Prof Kristian Martinsen for all the helpful efforts during the period of this master thesis work, especially for introducing me with all the talented people with whom I had a pleasure to collaborate.

Big thanks to my colleagues at SINTEF Raufoss Manufacturing AS, most notably Sebastian Dransfeld, for his professional guidance and priceless advices, Ådne Solhaug Linnerud and Øystein Knauserud, for their friendly support during working hours, Olivier Roulet-Dubonnet, for inspiring me to think creatively and unconventionally, Geir Ringen, for his interdisciplinary ideas and advices, and my fellow student and colleague Anastasiia Moldavska, for helping me to look at the problems differently.

Special thanks to Morten Lind for helping me to gain a better understanding of the mathematical aspects underlying the calibration part of this thesis. I gladly appreciate his critical comments on my ideas and all the professional conversations, especially those in the late evenings over Google Hangouts.

Thanks to all my teachers at Gjøvik University College for all the new knowledge and ideas. During the two years of my master program they helped me to unleash my creativity, apply my knowledge, and acquire a new interdisciplinary way of thinking, so important for building the new sustainable future.

The biggest thanks to my family, especially my mother and my dear Ivanna, for their love, support, and strong belief in me.

Contents

List of Figures

List of Tables

Acronyms

API Application Programming Interface

APROX Agent-based Production-Resource-Order eXecution

CIM Computer Integrated Manufacturing

F-FAS Fully-Flexible Assembly System

FAS Flexible Assembly System

FlexVi Flexibility through vision

FMS Flexible Manufacturing System

GPL General Public License

HMS Holonic Manufacturing System

KRL KUKA Robot Language

MAS Multi-Agent System

NCE Norwegian Center of Expertise

OAC Open Architecture Control

OpenCV Open Source Computer Vision

OROCOS Open Robot Control Software

PyMoCo Python-based robot motion control

R&D Research and development

RANSAC RANdom SAmple Consensus

RMS Reconfigurable Manufacturing System

RMS Root mean square

SOA Service-Oriented Architecture

SRM SINTEF Raufoss Manufacturing AS

XML Extensible Markup Language

1 Introduction

1.1 Background

In the recent decades, the need for manufacturing and assembly systems with greater degree of flexibility and automation increased. This need is mainly stimulated by the increased competition in the globalized economy, and is especially evident in the developed countries in Europe and North America.

Automated manufacturing provides competitive advantage in cost and quality, and is a surviving factor for the countries with high labor cost. To cope with volatile global markets, the manufacturing enterprises require high degree of quality, responsiveness, agility, and flexibility (Leitão 2009). According to "A Roadmap for U.S. Robotics. From Internet to Robotics. 2013 Edition" (RoboticsVO 2013), one of the major challenges for the robotics research is the one of adaptive and reconfigurable assembly.

The problem of connecting the objectives of rising the degree of flexibility and degree of automation is that flexibility and automation are somewhat contradictory concepts. Human workers are much more flexible than any automated robotics-based system. On the other hand, human workers cannot reach the objectives imposed on the robotized systems.

Flexibility of assembly systems is traditionally achieved by utilizing ingenious mechanical solutions, such as flexibility-oriented design of grippers, handling systems, automatic tool changers etc. (Edmondson and Redford 2002). Somewhat different, but complementary to this is the approach of integrating intelligence into automated assembly systems. This idea appears to be sound when recalling the abovementioned problem of relatively low flexibility of robots (as opposed to humans). As Heping et al. 2008 note, "it is difficult for conventional industrial robots to adjust to any sort of change" (Heping et al. 2008, p. 46), and therefore there is a need for more intelligent industrial robotic systems.

Intelligence in industrial robotic systems is defined by their software part, and, when it comes to robots, machine vision is of vital importance for achieving increased intelligence and flexibility. Therefore, even though the mechanical part will always play the central role in assembly systems, the challenge is to investigate the possibilities of software part, and vision systems in particular.

In order for the robot vision system to be used, they have to be properly calibrated. The calibration process involves several stages depending on the configuration of the system. For instance, in the *eye-in-hand* configuration (camera mounted on the robot) one needs to calibrate the robot, the camera, and the hand-eye device, i.e. to relate the coordinate frames of the camera and the robot flange. In addition, since stereo vision is usually used to reconstruct the 3D positions of the found features, the stereo vision system have also to be calibrated.

Concerning the robot vision system calibration, such issues as accuracy, precision and repeatability of a big importance. The calibration process is multi-staged, and robot control is depended on a series of calibration matrices. Thus, one needs to achieve the final combined result of calibra-

tion with as little error as possible. To achieve this, the calibration methods should be analyzed and improved with the aim of achieving higher accuracy, precision and repeatability.

1.2 Problem domain

Having in mind the abovementioned considerations, the work underlining this master thesis is focused on two interconnected areas:

- Software-defined assembly flexibility with an emphasis on vision-based robot control;
- Calibration of robot vision systems with the focus on improving accuracy, precision and repeatability.

More specifically, the aims of the work are to gain deeper insights into the role of software and intelligence in flexible assembly systems, to bring about better understanding of the calibration of robot-vision systems processes, and to propose computational methods of improvement of vision systems calibration with the purpose of maximizing repeatability, precision and accuracy.

Thus, the master thesis concentrates on three subjects:

1. Understanding the nature of flexibility in manufacturing and the role of software and intelligence in flexible assembly systems.
2. Development of a software framework for research of vision systems, and particularly camera calibration, based on OpenCV library and Python tools for scientific computing.
3. Improvement of camera and hand-eye calibration processes with the focus on maximizing repeatability, precision and accuracy.

1.3 Research questions

The following research questions are formulated:

RQ1 What is the role of software in provision of assembly flexibility?

RQ2 What is the role and implication of using machine vision in flexible assembly systems?

RQ3 How can practically higher accuracy, precision and repeatability of robot vision systems calibration be achieved?

1.4 Project evolution

The practical tasks of this master thesis were performed by the author while working as a Research Assistant at SINTEF Raufoss Manufacturing AS (SRM). SRM is a part of the NCE Raufoss, a Norwegian competence center in lightweight materials and automated production. During the second year of master study, the author worked part time in the Production Technology Department at SRM in Raufoss, Norway. The author was assigned various task aimed at building the competence around open source technologies and solving the practical problems. The tacked areas comprised machine vision (camera calibration, hand-eye calibration) and industrial communication.

1.5 Thesis outline

The rest of this master thesis is organized as follows:

Chapter 2 presents high-level philosophical considerations on the scientific methodology for this thesis.

Chapter 3 presents the questions of manufacturing flexibility, flexible assembly systems, and the role of control software and machine vision in the flexible systems. The chapter concludes with building a system model of software-defined assembly flexibility.

Chapter 4 overviews the area of vision systems calibration, presents a developed software library FlexVi targeting the research of robot vision systems, includes the proposed methods for improvement of camera and hand-eye calibration and the corresponding experimental results.

Chapter 5 summarizes the work by discussing both the high-level notion of software-defined assembly flexibility and the lower level of vision systems calibration.

Chapter 6 concludes the thesis.

2 Philosophical considerations on methodology

2.1 Methodological views taxonomy

Arbnor and Bjerke 2008 propose a division of scientific methodologies into three methodological views: analytical, systems and actors. Analytical view, popular in the natural sciences and engineering is aimed at uncovering universal rules with reliance on data, thus seeking for the cause-and-effect relationships. System view is based on the presumption that systems can be studied only as the totalities, which cannot be separated. This view constitutes building the systems models and determining the finality relations that lead to desirable outcomes. Actors view is the least formalistic and is based on the presumption that the reality is socially constructed and can be understood only by taking part in the process of social construction.

2.2 Justification of the chosen methodological approach

This thesis is based on the *systems view*, in the reason for complexity and interrelationships between different systems under consideration. The justification for this choice and the *methodological approach* (methodological view in application) of this thesis are described below.

The notion of flexibility, which is aimed to be understood, is complex by its nature, because it involves both the structure of a production system and the environment producing the fluctuations in the demand, to which a flexible production system should respond. Even though the technical component of a production system is a big importance (and the main theme in this thesis), one doesn't have to diminish the role of human factor, both inside a production system and in the environment.

An assembly system is a component of the whole production system, having specific structure and requirements. This imposes special features of assembly flexibility, which has to be understood.

Software, on which a big emphasis is put in this work, constitutes an increasingly important part of any enterprise. In manufacturing, the software subsystem plays a vital role in both providing manufacturing control and fulfilling hight-level information needs of a company. Even though this work is focused on low-level control, the question of vertical integration should be also kept in mind. Recalling the notion of environment producing the demand, the software-based communication technologies such as Internet is what makes flexible production systems so important. Thus, it is evident that the role of software-based system is huge in the contemporary economy.

When it comes to assembly flexibility, this thesis is aimed understanding the role of the software subsystem in providing this flexibility. In the terms of the systems view, one needs to find the *finality relation* explaining how the result (flexibility) is influenced by the driving force (software) (Arbnor and Bjerke 2008, p. 57).

The notion of *magnifying levels* is based on the idea that "any component in a system is a

potential system of its own" (Arbnor and Bjerke 2008, p. 117). Thus, a system can be viewed on a higher level, resulting in a less detailed representation, and on a lower level, with a bigger number of details. In the case of this master thesis, two magnifying levels are considered:

Level of an assembly system The purpose is to understand the structure of an assembly system, assembly flexibility, and the role of software subsystem in providing the flexibility.

Level of robot vision system The purpose is to take a close look at the problem of calibration of robot vision systems with the aim of improving accuracy, precision and repeatability of the calibration processes.

These two topics are considered in the two subsequent chapters. Each chapter preserves its own structure including the literature overview, method description, results (in chapter 4) and discussion. The chapter following them presents a discussion combining the previously accumulated knowledge on both magnifying levels.

Figure 1 schematically depicts the scope of this thesis from the systems perspective, as it was described above.

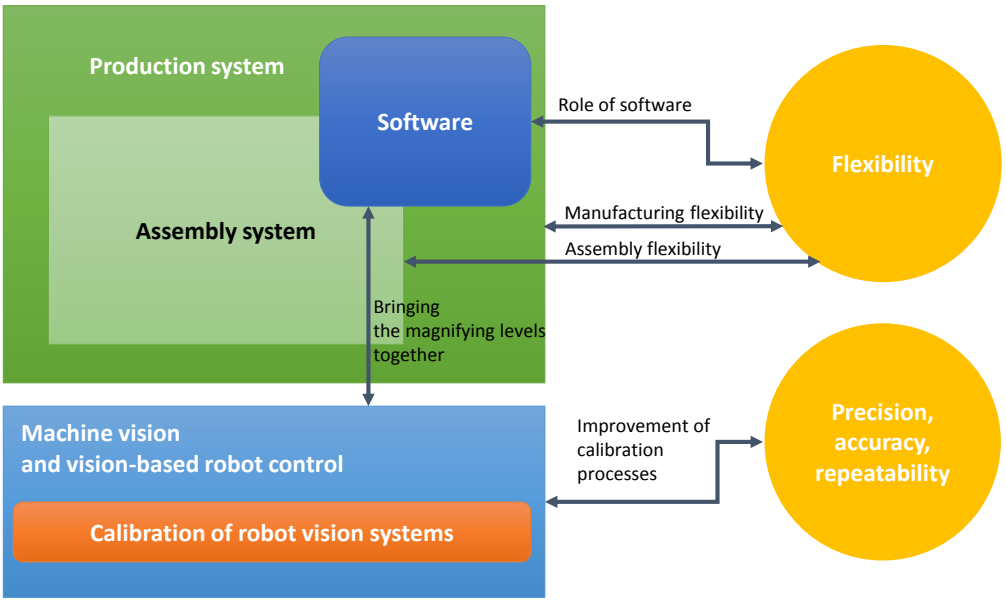


Figure 1: Scope of the thesis

3 Software-defined assembly flexibility

This chapter is aimed at investigation of the role of software in providing manufacturing flexibility. First, the nature of flexibility must be understood. Since the focus of this thesis is on assembly systems, one needs to distinguish the characteristics of flexible assembly systems and subsystems of FAS. Elaboration around the role of software and intelligence for manufacturing flexibility in general and for flexible assembly systems in particular will then be presented.

3.1 Theory and related work

3.1.1 The nature of flexibility

Upton 1994 broadly defines flexibility as "the ability to change or react with little penalty in time, effort, cost or performance" (Upton 1994, p. 73).

Many scholars present taxonomies of different types of manufacturing flexibility. Wadhwa 2012 identifies the following flexibility types:

Changeover flexibility Number of different products produced concurrently over time.

Product modularity flexibility Leveraging product variation by help of modular design.

Product mix flexibility Ability to handle multiple part families (e.g. by using automatic tool changers).

Plant layout flexibility Capability of the plant layout to react to changes.

Machine flexibility Number of operations a machine can perform, and ability to switch from one operation to another.

Part handling flexibility Ability of the material handling system to move different part types efficiently.

Production flexibility The universe of part types a manufacturing system can produce without adding major capital investments.

Operations flexibility Ability of a manufacturing process to produce a part with different process plans.

Response flexibility Ease of moving from one configuration of the manufacturing system to another.

Part routing flexibility Ability to dynamically assign parts to machines.

ElMaraghy 2006 contrasts the concepts of a Flexible Manufacturing System (FMS) and a Reconfigurable Manufacturing System (RMS): while FMSs provide generalized flexibility for the a priori anticipated variations, RMSs provide “customized flexibility on demand in a short time” (ElMaraghy 2006, p. 261).

Wiendahl et al. 2007 claim that the term "flexibility" is rather general and has to be differentiated in relation to different factory levels. The authors propose "changeability" as an umbrella term, and define it as "characteristics to accomplish early and foresighted adjustments of the factory’s structures and processes on all levels to change impulses economically" (Wiendahl et al. 2007, p. 785). The domain of changeable manufacturing is proposed to include the FMS, RMS and other related paradigms.

Bi et al. 2008 focus on four critical requirements for the contemporary manufacturing systems, namely short lead-time, more variants, low and fluctuating volumes, and low price. Then, the authors review different manufacturing systems paradigms (with FMS and RMS among them) in their relation to the abovementioned requirements. Further the author provide thorough state-of-the art analysis of the RMS paradigm, focusing on the issues of architecture design, configuration design and control design. The authors conclude by identifying the following directions for the future:

1. RMS should be supplemented with the others manufacturing paradigms, most notably lean manufacturing, in order to provide a more complete solution.

2. It is important to focus on developing a systematic design methodology for RMS that would extend the traditional non-reconfigurable methodologies and would account for balancing the trade-offs among the costs, reconfigurability and complexity.

3. It is important to explore the question of designing a configuration for RMS where complexity is high and the number of reconfigurable components is big.

4. An emphasis on control systems should be put, touching the questions of configuration design and real-time control, complexity of control systems, and developing of autonomous, distributed, scalable and self-reconfigurable control systems.

3.1.2 Flexible assembly systems

FAS requirements

Bi et al. 2008 defines a manufacturing systems as a system that transforms raw materials into products. A manufacturing system can be modeled as consisting of three main activities (Figure 2), each of which is functioned by meeting the specified requirements R by producing the products P (Bi et al. 2008, p. 969):

1. Design: defining components and assemblies based on customers’ requirements;

2. Manufacturing: producing basic parts;

3. Assembly: putting basic parts together to form the final product.

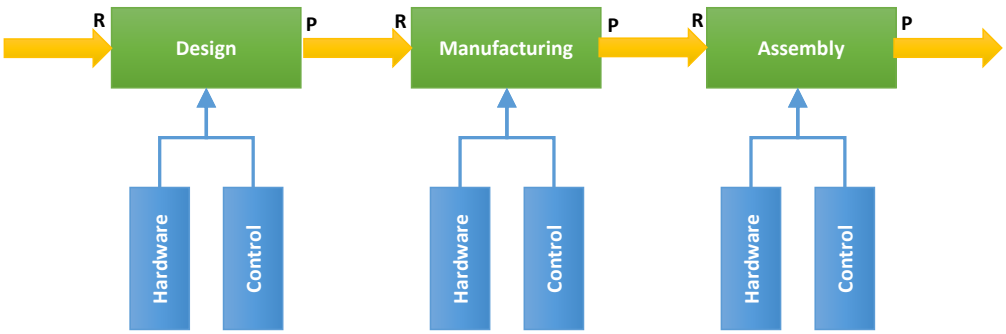


Figure 2: Manufacturing system model (adapted from Bi et al. 2008)

Thus, an assembly system can be identified as a subset of a manufacturing system. This imposes specific requirements for FAS: requirements for equipment and software comprising a FAS and for organizational structures supporting assembly processes.

Rosati et al. 2011 lists the following requirements for a flexible assembly system (Rosati et al. 2011, p. 1):

1. Flexibility: ability to handle a wide variety of part types, perform changeovers quickly and easily, process multiple parts and models simultaneously, quickly response to part design changes;

2. Compactness: limited amount of space around the perimeter of a single station assembly machine;

3. Throughput (parts/hour): the number of parts assembled in a defined period;

4. Unit direct production cost ($/part): the ratio of the hourly costs of the work cell and the average throughput per hour.

Mechanical subsystems of FAS

Edmondson and Redford 2002 examine the design, selection and integration of a number of mechanical systems to develop a generic flexible assembly system. The authors views such a system as consisting of two mechanical systems working in parallel: “the manipulator which performs the actual assembly task and the material handling equipment” (Edmondson and Redford 2002, p. 141).

Referring to Redford 1991, Edmondson and Redford 2002 list the following groups of material handling requirements:

1. Handling of pieceparts into the system;

2. Handling of palettes, fixtures and tools;

3. Removal of the completed products from the system;

4. Accommodation of operations external to the assembly cell;

5. Transportation of partially finished products to and from rework.

According to Redford 1991, handling of pieceparts should be performed by flexible small parts feeders, and a pallet system should be used for all the other handling operations.

Edmondson and Redford 2002 review the following types of flexible small part feeders:

1. Linear vibrator;

2. Robot-based system;

3. Vibratory bowl feeders with a vision system;

4. Belt feeders.

The authors concluded that the belt feeders are the best choice in the situation with many different part variants and frequent new product introduction, since they give "the lowest part presentation cost over a wide range of parts, at various production volumes and at varying production volume" (Edmondson and Redford 2002, p. 147).

Regarding the mechanical part of the manipulator, for achievement of higher degree of flexibility the following approaches are applied:

Flexible grasping Making a gripper more adaptive and universal. Achieved by utilizing deformable end-effectors, increased numbers of degrees of freedom, grasp planning, cognitive abilities, and sophisticated sensing means (Rooker et al. 2013; EUnitedRobotics 2014).

Automatic tool changers Instead of having one universal gripper, several grippers can be automatically changed by the robot (Wadhwa 2012).

Modular manipulators Designing of the manipulators from the modular components, thus allowing for quick reconfiguration of the manipulators' mechanical structure (Chen 2001)

Manual assembly in FAS

When considering FAS, Wiendahl et al. 2007 put an emphasis on possibility to upgrade or downgrade the degree of automation, i.e. to combine automated assembly and manual assembly. This type of system is called a hybrid assembly system.

Manual assembly is out of the scope of this paper. However, it worth mentioning, especially when considering human-oriented approaches to manufacturing such as *lean* (Womack and Jones 2010).

3.1.3 The role of software for assembly flexibility

Previous subsection focused on flexible assembly systems and traditional hardware requirements for FAS. This section, in turn, is aimed at exploring approaches that are more dependent on software part of the system.

Requirements for control systems

According to Bi et al. 2008, to accomplish the activities of a manufacturing system, namely design, manufacturing and assembly, both the hardware and control resources are required. The authors define control resources as those involved in information flow, whereas hardware resources are involved in process flow.

To provide reconfigurability, a control system should comply with the following requirements (Bi et al. 2008, p. 983):

- Autonomous design: a system consisting of autonomous modules able to cooperate and achieve system-level objective;
- Modularity: a system should be modularized and distributed;
- Openness: ability to update controlling components developed on heterogeneous environments (programming languages, operating systems, databases etc.);
- Scalability and upgradeability: ability to add/remove/upgrade components;
- Self-reconfigurability: ability to automatically reconfigure the control system when the hardware configuration has changed;
- Ability to identify the changes of task specifications.

The abovementioned requirements correspond to the paradigm of holonic and multi-agent manufacturing systems (discussed further).

As a contrary approach to autonomous and distributed holonic control, the Open Architecture Control (OAC) is considered. It is based on the available hierarchical structures used in Computer Integrated Manufacturing (CIM), with the addition of the requirements for change (Bi et al. 2008, p. 984).

Holonic Manufacturing Systems

Holonic Manufacturing Systems (HMS) is a system paradigm introduced in the early 90s, in which a manufacturing system is built up from a number of autonomous entities called holons, able to cooperate and be composed of other holons (Van Brussel 1994; Van Brussel et al. 1998). Morten Lind 2012 views a holonic system as possessing characteristics of both hierarchic and heterarchic system, and proposes the following definition for HMS: "A holonic control system is an dynamic organization of control entities, called holons. A holon is an agent of which a certain set of control facets must exhibit the ability to cooperate, be coordinated, and act autonomously. The precise dynamics of the modes is entirely up to the surrounding environment of holons and to the timely state and status of the local and global tasks, goals, and objectives" (Morten Lind 2012, p. 25).

An example of holons implementation, consisting of a physical part interfacing with hardware and a high-level part providing decision-making capabilities and interacting with other holons, is presented in Figure 3.

Multi-agent systems (MAS) are often considered in the literature together with HMS, as certain aspects of both system paradigms are overlapped, especially when it comes to the high-level

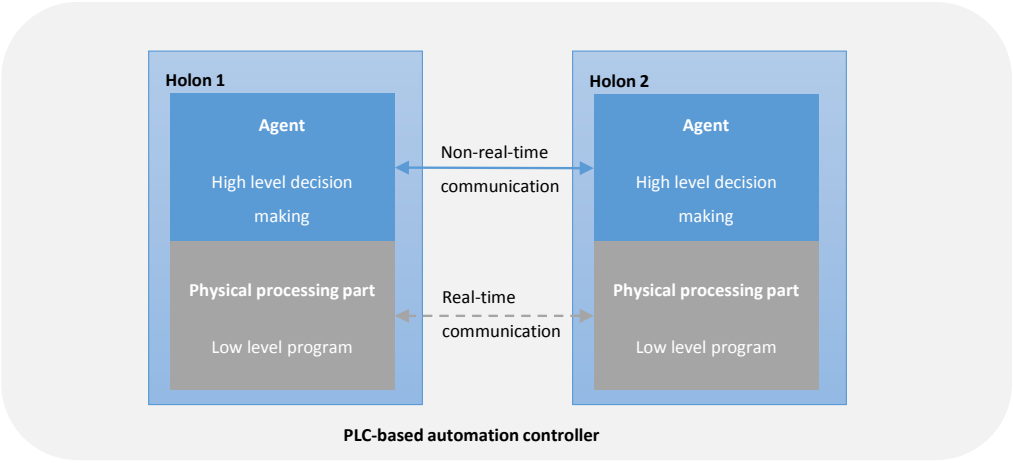


Figure 3: Example of holons implementation (adapted from Marik and Duncan McFarlane 2005)

production planning and control. It should be noted, however, that MAS is a broader area of research, targeting many areas besides manufacturing control, and some application of agents in MAS (e.g. in decision support systems) are extremely different from the application of holons in HMS.

Application of HMS on the lower level is of a big interest regarding the development of flexible and reconfigurable systems. According to Roulet-Dubonnet and Ystgaard 2011, flexibility and reconfigurability are facilitated by better separation of concerns through modularity into holons.

Roulet-Dubonnet and Ystgaard 2011 present a holonic approach to building a flexible assembly cell producing damped boring bars. The authors applied the HMS paradigm to both the architecture of the real-time control system and for the mechanical design of the cell. The developed system is based on the APROX architectural concept (Gellein and Nyen 2010) and IceHMS multi-agent middleware (Roulet-Dubonnet, M. Lind, and Skavhaug 2013).

Other examples of application of HMS to the design of flexible manufacturing are presented by Chirn and D McFarlane 2000, Maeda et al. 2003, Makris et al. 2012.

Fully flexible assembly systems

Rosati et al. 2011 introduce the concept of fully-flexible assembly systems (F-FAS), compare them to generic flexible assembly systems (FAS), and propose a quantitative method for evaluating and comparing direct production costs and convenience of introducing these two types of assembly systems.

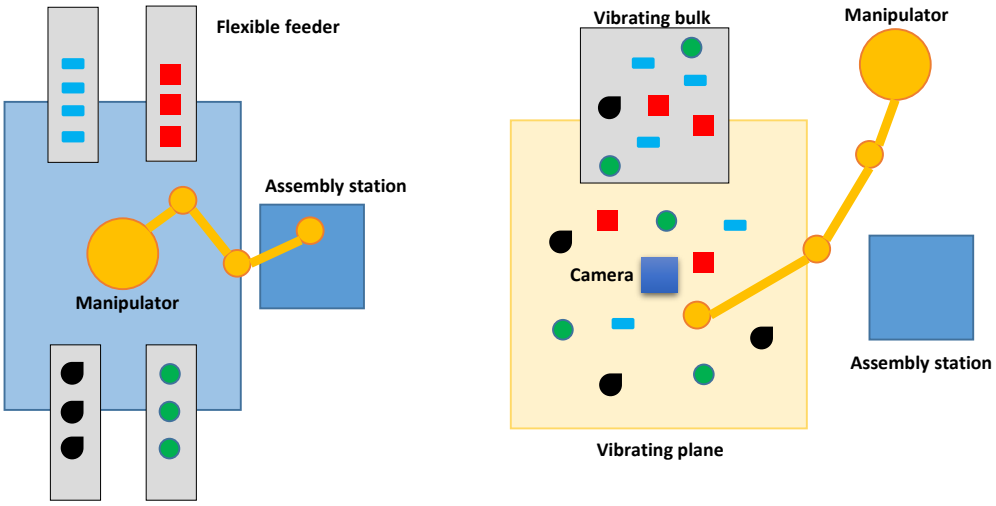
According to the authors, a generic FAS typically consists of the following components:

1. A flexible feeder subsystem: one feeder for each component;
2. One or more flexible assembly stations;
3. One programmable manipulator.

A fully-flexible assembly system (F-FAS), according to Rosati et al. 2011, is an assembly system

having the maximum degree of flexibility with respect to handling a wide variety of part types, thus “able to handle a highly mixed production order in which the size of the batch may be as small as one piece” (Rosati et al. 2011, p. 1).

To realize a F-FAS, the authors propose utilizing one fully-flexible feeder subsystem comprising a vibratory bulk, vibrating plane and a vision system. All the parts needed to fulfill the order will be placed on the vibrating plane, and the vision system will be used to identify the parts and calculate their position and orientation before picking by the manipulator. A graphical schemes depicting the concepts of FAS and F-FAS are presented if Figure 4.



(a) FAS (b) F-FAS

Figure 4: Comparision of FAS and F-FAS

To quantitatively assess and compare FAS and F-FAS, Rosati et al. 2011 proposed computing the hourly throughput and hourly direct production cost of each type of the systems. The comparison of these metrics are presented in Table 1 and Table 2.

The following parameters are involved in calculations:

- t_{pp} – the time needed for manipulator and assembly of a single component;
- t_f – the time needed for image acquisition and processing;
- N_A – the average number of components picked and assembled for each grabbed image;
- C_{plant} – the direct cost of the work cell;
- C_{robot} – the direct cost of the robot;
- C_{camera} – the direct cost of the camera;
- $C_{flexfeed}$ – the direct cost of the flexible feeder;

- h_{pb} – the number of working hours in the paying back time;
- t_{set} – the time needed for setting up each flexible feeder;
- N_c – the number of components/feeders;
- h_{res} – the average working time between two subsequent resettings of the work cell;
- $C_{h,op}$ – the hourly cost of the operator in charge for re-setting the feeders).

Table 1: Hourly throughput of FAS and F-FAS

FAS	F-FAS
$Q_{fas} = \frac{1}{t_{pp}} \frac{h_{res} - N_c t_{set}}{h_{res}}$	$Q_{ffas} = \frac{1}{t_{pp} + t_f / N_A}$

Table 2: Hourly direct production cost of FAS and F-FAS

FAS	F-FAS
$C_{fas} = \frac{C_{robot} + N_c C_{feed}}{h_{pb}} + \frac{N_c t_{set}}{h_{res} C_{h,op}}$	$C_{ffas} = \frac{C_{plant}}{h_{pb}} = \frac{C_{robot} + C_{camera} + C_{flexfeed}}{h_{pb}}$

3.2 Method

The notion of software-defined assembly flexibility, which is central to this chapter, is going to be tackled from the system perspective. Various sources presented above serve as a theoretical input to the study.

Based on the theoretical input, in the subsequent section different aspects of assembly flexibility and its reliance on software and machine vision will be discussed.

The discussion will then lead to building the system models aimed at bringing better understanding of software-defined manufacturing flexibility and building a higher level ground for the subsequent chapter on robot vision systems calibration.

3.3 Discussion

3.3.1 Problem of taxonomy

One of the most evident problems when considering manufacturing flexibility is the lack of the agreed-upon taxonomy. In this thesis the notions of *flexibility*, *reconfigurability* and *changeability* were mentioned. To the big extent, the difference between these concepts depends on the goals of the respective research groups and the contexts in which these related notions were considered.

Bi et al. 2008 provides an example of how different the understanding of RMS and FMS paradigms might be (Bi et al. 2008, p. 967):

> The confusions and controversies are often raised among the readers to understand and adopt new production paradigms. For example, at the 3rd Conference on Reconfigurable Manufacturing held at the University of Michigan during May 10–12, 2005, the attendees have had a controversy about the definition of RMS. Some insist that an RMS is an intermediate paradigm

between Mass Production and Flexible Manufacturing System (FMS), some argue that an RMS is an advanced paradigm whose flexibility must be higher than that of an FMS, and the others think it is not very meaningful to distinguish RMSs from FMSs.

Wiendahl et al. 2007 try to categorize different approaches under the umbrella term *changeability*. This, however, leads to inconsistency between the view of Wiendahl et al. 2007 and the other scholars. For example, Wiendahl et al. 2007 define *flexibility* as a tactical ability on a segment level to switch from the old family of products to the new one. By the segment level the authors mean a factory level on which the products are ready to ship. A segment, according to the authors, comprise such activities as manufacturing, assembly, buffers, quality control etc. Thus, this view considers flexibility on a higher level, which makes it contrary to the ideas of flexible manufacturing and flexible assembly on the process levels.

It is important therefore to eventually come to the common understanding of taxonomy regarding flexibility and related concepts. In this thesis, *manufacturing flexibility* is considered in a broad terms, similar to *changeability* of Wiendahl et al. 2007, but preserving the relation to the flexibility categories discussed by the other scholars (see 3.1.1). *Reconfigurability* should be regarded as an essential ability of any contemporary flexible system.

3.3.2 The model for flexible assembly

As subsection 3.1.2 show, when an assembly system is considered, it is possible to delimit three major subsystems of which it is comprised, which are mechanical, software, and human subsystems (Figure 5).

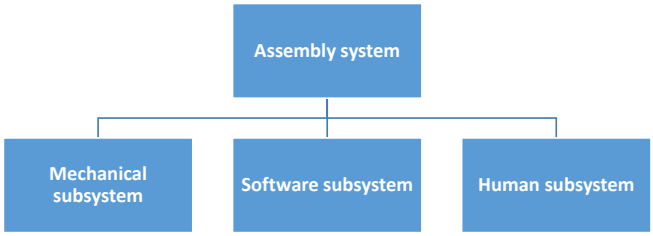


Figure 5: The subsystems of an assembly system

In subsections 3.1.2 and 3.1.3, the roles of each subsystems were presented from the theoretical perspective. To achieve flexibility by the mechanical means, one concentrates on the ingenious design of material handling equipment, robot grippers, tool changers etc. When it comes to control software, one puts an emphasis on architectural solutions providing modularity, openness, compatibility etc. (see 3.1.3). In addition, in assembly systems, vision software contributes to the flexibility level by allowing the same robots to handle different assembly tasks and the same feeders to handle different part families. Flexibility can be achieved by the means of the human subsystem by combining manual and automated assembly.

Since, as was shown in the previous paragraph, different assembly subsystems can be used to provide higher flexibility, it is proposed to introduce three respective notions:

Mechanically-defined assembly flexibility Flexibility of an assembly system imposed by the

mechanical means.

Software-defined assembly flexibility Flexibility of an assembly system imposed by the means of software.

Human-defined assembly flexibility Flexibility of an assembly system imposed by involvement of human workers.

Having in mind these three types of assembly flexibility imposed by different flexibility drivers, the question of their effective combination arises. Some discussion regarding this combination is presented below.

The approach of mechanically-defined flexibility is the most typically used, since the mechanical part of an assembly system is what directly performs the actual work. If, however, a big emphasis is put on the mechanical part, a problem mentioned by ElMaraghy 2006 may arise: a system would become too fixed to introduce some radical, not built-in a priori changes. The proposed idea of F-FAS is aimed at solving this problem by substituting the complexity of mechanical part by more sophisticated software part.

In the introduction it was mentioned that flexibility and automation are somewhat contradictory notions, because human workers are much more flexible than any robots. The human workers are therefore may be involved in the assembly process to complement the automated system and provide higher level of flexibility. This was the idea of decreasing/increasing the degree of automation, described in Wiendahl et al. 2007. Another approach can be found in th idea of *autonomation*, also known as automation with a human touch. Autonomation is a technique of stopping a line or a machine when a defect occur. After the line is stopped, the cause of the problem are investigated and corrective actions are incorporated to prevent the future occurrences of this type of defects (Ohno 1988; Monden 2011).

3.3.3 The role of software for different types of manufacturing flexibility

Recalling the categorization of manufacturing flexibility (see 3.1.1), certain flexibility types are of particular interest regarding the software-related issues. Machine flexibility, which can be seen as the dominant factor in FAS, according to Wadhwa 2012, can be implemented in a cost-efficient way by utilizing electronic and software development. Production flexibility, which is a complex measure encompassing the whole manufacturing system, depends on "the variety and the versatility of the machines that are available, the flexibility of the material handling system, and the factory's automation and control system" (Wadhwa 2012, p. 449). The effectiveness of the latter can be heavily influence by application of the holonic paradigm and the open communication system. ElMaraghy 2006 distinguishes control program flexibility as a distinct type of manufacturing flexibility. It is explained by the author as the ability of a control system to run virtually non-interrupted by the means of intelligent machines and system control software.

3.3.4 Flexibility through vision

Of a big interest is the idea of substituting the mechanically complex assembly systems by the ones with simpler mechanical structure compensated by more powerful software component. Since the vision systems are largely defined by their computational part, by integrating them into

the assembly systems, a bigger flexibility with lower mechanical complexity could be achieved. A good example of this approach is the one of F-FAS.

In FAS, traditionally, flexibility is defined by functionality of the flexible feeders. In contrast, the concept of F-FAS, is functionally based on utilizing a vision system, and therefore more dependent on the software component. This makes F-FAS mechanically simpler. The weak place of such system, however, lies in the computationally intensive image processing. If one considers hourly throughput of F-FAS, it is maximized when more components are picked and assembled per image acquisition (N_A), but, on the other hand, the bigger number of components are lying on the plane, the more image processing time t_f is needed (Rosati et al. 2011).

$$Q_{ffas} = \frac{1}{t_{pp} + t_f/N_A} \tag{3.1}$$

As a solution to the problem of big image processing time, the new image-processing algorithms are needed.

Calibration processed of robot vision systems are cornerstone for the machine vision tasks. If the calibration can be performed in automated manner with high degree of reliability, reconfigurations of the assembly systems can be done quicker and easier. Thus, the flexibility of vision-based assembly systems will be greatly improved.

3.3.5 Compatibility and open source software

Morten Lind 2012 emphasizes the problem that industrial robot control and integration suffer from that different robot manufacturers develop their own native application platform. This results in complexity of robot programming and incompatibility between the systems. Compatibility, according to Wiendahl et al. 2007 is one of the main changeability enablers, and therefore the lack of compatibility is a problem that has to be solved in order to increase the changeability potential.

Morten Lind, Schrimpf, and Ulleberg 2010 propose an open source motion framework PyMoCo, which provides the common Python API to control the robot motion. The authors refer to the open source projects having similar aims, namely Open Modular Controller and OROCOS (OROCOS 2014). As a prerequisite to PyMoCo, PyMath3D library was developed, allowing for 3D Euclidean space computations (Morten Lind and Schrimpf 2012). This library is further used in this master thesis work for representation of homogeneous transformations in the robot vision systems calibration routines.

In his PhD thesis, Schrimpf 2013 favors using open source software and software platforms suited for fast prototyping when designing flexible and reconfigurable systems.

In the area of computer vision, OpenCV library has reached a big popularity (Bradski and Kaehler 2008) and is used as a standard in many applications.

Most notably, the power of open source software is evident in the communication platforms. Roulet-Dubonnet, M. Lind, and Skavhaug 2013 list three types of software platforms for implementing distributed systems for manufacturing control, namely lower level communication middleware, multi-agent and holonic platforms, and robotic platforms. The majority of platforms, outlined by the authors, are developed as open source projects by research institutions and industrial companies.

4 Robot vision systems calibration

4.1 Theory and related work

4.1.1 Vision-based robot control and calibration

Vision-based robot control, or visual servoing, is a technique that uses data from the vision system in the servo loop to control the robot motion (Chaumette and S. Hutchinson 2006). A camera in the system can be in one of the several configurations, among which the most typical are *eye-in-hand* configuration, when the camera is mounted on the robot, and *stationery* configuration, when the camera is fixed in the workspace (Chaumette and S. Hutchinson 2006).

Geometry of three-dimensional space and rigid motions

Before proceeding to the details of robot vision systems calibration, an overview of the mathematical background of the rigid body motion will be presented.

As it will be shown later, in order to control a robot using the data from a vision system, one needs to know the coordinates of an identified by the vision object in the robot's coordinate space. The calibration procedures are used to define the unknown transformations between different coordinate frames: of the object, of the camera, and of the robot. To mathematically describe coordinate frames and transformation between them, *geometry of three-dimensional space and rigid motions* is used. It is described in details by Spong, Seth Hutchinson, and Vidyasagar 2006 and Stramigioli and Bruyninckx 2001, and is briefly presented below.

Euclidean n-space, is "the space of all n-tuples of real numbers, $(x_1, x_2, \ldots, x_n)$" (Stover and Weisstein 2014), and is denoted as $\Re^n$. Thus, $\Re^3$ represent a Cartesian three-dimensional space.

A point in $\Re^3$ space can be defined as a vector $(x_1, x_2, x_3)^T$ attached to a reference frame. A vector defined in reference frame A can be represented in reference frame B using the procedures of rotation and translation:

$$p^{(B)} = R_A^B p^{(A)} + d_A^{(B)} \tag{4.1}$$

where $p^{(A)}$ and $p^{(B)}$ are representations of the vector in coordinate frames A and B respectively, R_A^B is a 3×3 rotation matrix rotating frame A so that its axes are parallel with the respective axes of frame B, and d_A^B is a 3×1 translation vector that translates the center of frame A into the center of frame B.

A rotation matrix R_A^B is said to be a member of the rotational group $SO(3)$ that represents all the rotations about the origin in the $\Re^3$ space.

Rotation matrix and translation vector can be combined into one matrix, called a *homogeneous transformation*:

$$T_A^B = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_1 \\ r_{21} & r_{22} & r_{23} & d_2 \\ r_{31} & r_{32} & r_{33} & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

where r_{ij} are the elements of the rotation matrix, and d_i are the elements of the translation vector.

Homogeneous transformation T_A^B is said to be a member of the Lie group $SE(3)$, and allows to perform translations in the following way:

$$p^{(B)} = T_A^B p^{(A)} \tag{4.3}$$

where $p^{(A)}$ and $p^{(B)}$ are representations of the vector in coordinate frames A and B respectively, expressed in the homogeneous coordinates (4×1):

$$p = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{4.4}$$

To obtain the Cartesian coordinates (x, y, z) of a vector, the following formulas are applied:

$$x = \frac{x_1}{x_4} \tag{4.5}$$

$$y = \frac{x_2}{x_4} \tag{4.6}$$

$$z = \frac{x_3}{x_4} \tag{4.7}$$

If reference frames 0, 1, and 2 exist, and one needs to obtain a transformation from frame 2 to frame 0 (T_2^0) having T_2^1 and T_1^0, the rule of composition of transformations applies:

$$T_2^0 = T_0^1 T_2^1 \tag{4.8}$$

The calibration trio

When a system consisting of a robot and a camera rigidly mounted on the robot is considered (*eye-in-hand* configuration), in order to estimate the 3D position and orientation of a part, the following three homogeneous transformations are needed (Figure 6):

1. Transformation between the hand and the robot base;
2. Transformation between the camera and the hand;
3. Transformation between the object and the camera.

To obtain the desired transformations, the corresponding calibration procedures are applied:

1. Cartesian robot hand calibration;
2. Hand-eye calibrations (also referred to as “robot eye-to-hand calibration”);
3. Camera calibration.

Tsai and Lenz 1989 refer to these tasks as “The Calibration Trio”.

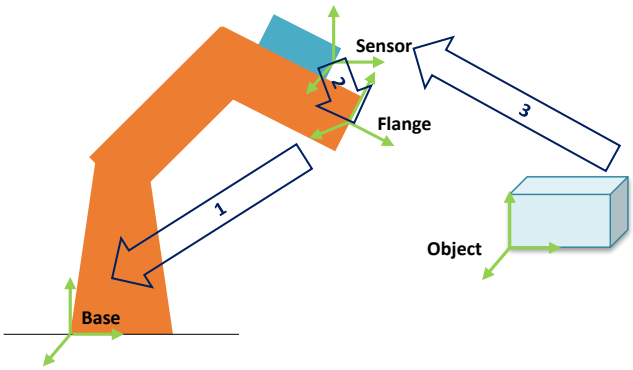


Figure 6: "The calibration trio"

4.1.2 Camera model and calibration

In a nutshell, to calibrate a camera means to computationally determine the intrinsic parameters of the camera in order to establish a correspondence between the real-world coordinates of an object and the pixel coordinates of the object's image taken by the camera. Theoretical overview of camera calibration presented below is largely based on the work of Bradski and Kaehler 2008.

Pinhole camera model

Pinhole camera model is the simplest model of a camera that consists of the image plane and the pinhole plane. The pinhole plane contains the pinhole aperture that lets through only those light rays that intersect the aperture in space. After a ray enters the pinhole aperture, it is then projected onto the image plane (Figure 7).

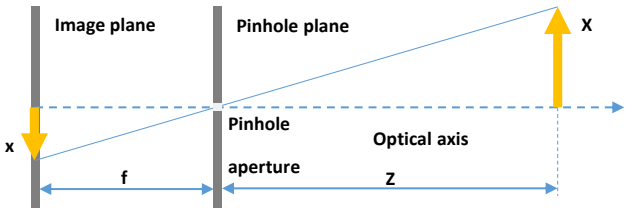


Figure 7: Pinhole camera model

On fig 7 f is the focal length of the camera, whereas Z is the distance from the camera to the object. If X is the length of the object in the real-world coordinates, then x is the object image on the image plane (Bradski and Kaehler 2008). By the rule of similar triangles ($-x/f = X/Z$), the

following relation applies:

$$x = -f\frac{X}{Z} \tag{4.9}$$

To simplify the calculations, the image plane is moved between the object and the pinhole plane (Figure 8). In this way, the negative sign is eliminated:

$$x = f\frac{X}{Z} \tag{4.10}$$

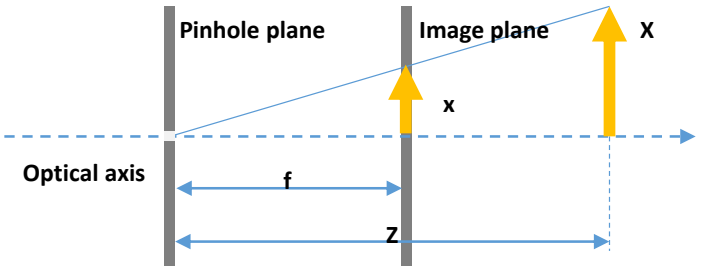


Figure 8: Image plane is "pushed" in front of the pinhole

The point of intersection between the image plane and the optical axis is called the principal point. Because in the real-world cameras it is impossible to position the imager in the way that its center ideally corresponds to the principal point, one needs to account for the displacement of the imager away from the optical axis by introducing the displacement parameters c_x and c_y. Having these considerations in mind, a point Q in the physical space with coordinates (X, Y, Z) will be projected onto the image plane at the pixel coordinates (x_{screen}, y_{screen}):

$$x_{screen} = f_x\frac{X}{Z} + c_x \tag{4.11}$$

$$y_{screen} = f_y\frac{Y}{Z} + c_y \tag{4.12}$$

Relation between the physical world coordinates and the projection screen coordinates

Projective transformation is a relation "that maps the point Q_i in the physical world with coordinates (X_i, Y_i, Z_i) to the points on the projection screen with coordinates (x_i, y_i)" (Bradski and Kaehler 2008, p. 373). By using homogenous transformations, the projective transformation from the physical world to the projection screen is performed in the following way:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{4.13}$$

The matrix M is referred to as the camera matrix.

Since in reality one needs to account for the relative rotation and translation between the reference frame of the object in the physical world and the reference frame of the projection

screen, a transformation matrix should be introduced to obtain the perspective transformation (OpenCV 2013):

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_1 \\ r_{21} & r_{22} & r_{23} & d_2 \\ r_{31} & r_{32} & r_{33} & d_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.14}$$

Lens distortion

The described above pinhole model is rather a simplification. The real cameras (as well as human eyes) use lenses to catch the light rays: this way, a camera gathers more light to be enough for rapid exposure (Bradski and Kaehler 2008).

Even though theoretically it is possible to create a lens that introduces no distortion, in practice no lens is ideal. The reasons for this, according to Bradski and Kaehler 2008, is that a mathematically ideal lens has a parabolic form, whereas for manufacturing it is easier to make spherical lenses, and it is not practically possible to mechanically align the imager and the lens. Thus, two corresponding types of distortion can be distinguished:

1. Radial distortion (the result of the lens shape), appears as “fish-eye” effect, i.e. bending of the rays farthest from the center;

2. Tangential distortion (the result of the assembly process of a camera).

To correct the distortions, the distortion coefficients are introduced: k_1, k_2, k_3 for radial and p_1, p_2 for tangential distortion (Bradski and Kaehler 2008). If (x, y) is a distorted point, the following equations are used to correct it:

$$x_{corrected}^{(radial)} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{4.15}$$

$$y_{corrected}^{(radial)} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{4.16}$$

$$x_{corrected}^{(tangential)} = x + (2p_1 y + p_2(r^2 + 2x^2)) \tag{4.17}$$

$$y_{corrected}^{(tangential)} = y + (p_1(r^2 + 2y^2) + 2p_1 x) \tag{4.18}$$

where r is a coefficient of a Taylor series ($r = 0$ in the center of the image).

The distortion coefficients are convenient to arrange into one distortion vector. In OpenCV it constitutes the following sequence of coefficients: $(k_1, k_2, p_1, p_2, k_3)$.

Camera calibration

Camera calibration process is aimed at determining the camera’s intrinsic parameters: the camera matrix and distortion coefficients.

To perform camera calibration, the camera makes a number of images of the calibration object from different aspects. A calibration object is an object on which it is easy to identify target points with known coordinates. A widely used calibration object is a chessboard pattern of

a rectangular form (Figure 9). This way, it would be possible to estimate the intrinsic parameters of the camera by knowing the real world coordinates of the target points and the corresponding pixel coordinates on the images. The mathematical details of finding intrinsic parameters are described by Zhang 2000 and Sturm and Maybank 1999.

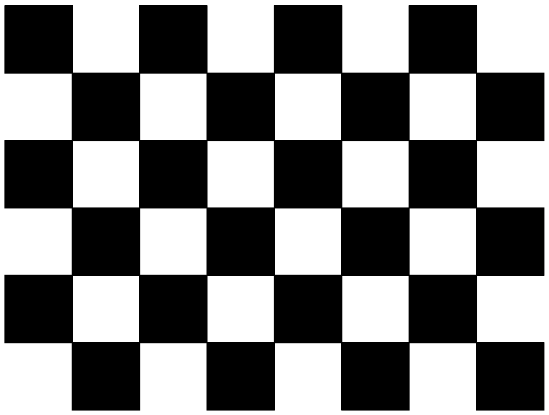


Figure 9: Chessboard calibration object

The described above calibration process solves a system of equations based on the supplied images of the calibration object. If K is the number of images and N is the number of corners on the chessboard pattern, one needs to supply $K > 1$ images of the 3×3 chessboard to the calibration function in order to account for all of the unknown intrinsic parameters. It is recommended, however, to use at least $K = 10$ images on 7×8 or larger chessboards (Bradski and Kaehler 2008, p. 388).

4.1.3 Stereo vision systems calibration

A stereo vision system is a system consisting of two or more cameras, able to assess coordinates of an object in 3D space. As Bradski and Kaehler 2008 note, "there is no reliable way to do calibration or extract 3D information without multiple images" (Bradski and Kaehler 2008, p. 405). Therefore, a stereo vision system is needed, which takes several images of the same scene at the same time, and gives the ability to reconstruct the 3D coordinates by matching the same features on different images.

Further a stereo vision system consisting of two cameras is considered. In order to conduct stereo imaging, one needs to have a system depicted in Figure 10. In the figure, f is the focal length, O_l and O_r are the centers of projection, P is the point in the real world, x_l, x_r are the coordinates of the point P on each of the imagers, Z is the depth.

In such system both cameras' imagers are positioned in the frontal parallel arrangement, allowing to find the depth Z by the rule of similar triangles (Bradski and Kaehler 2008):

$$Z = \frac{fT}{x_l - x_r} \tag{4.19}$$

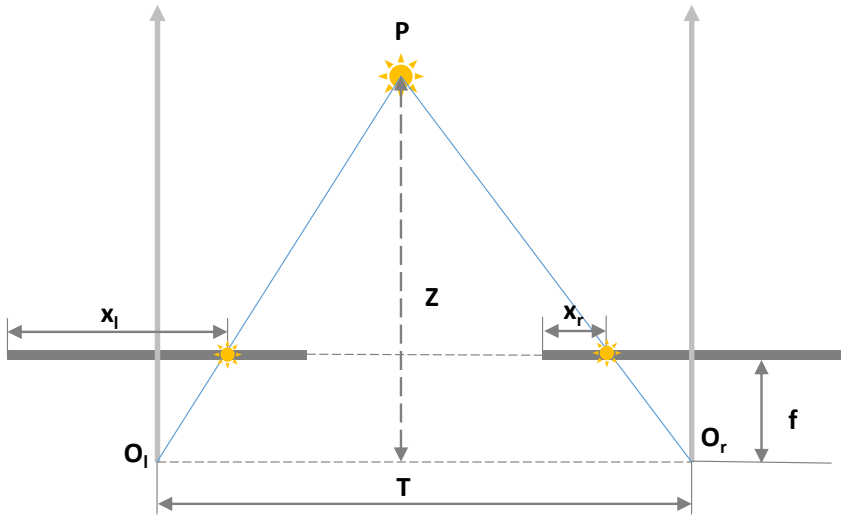


Figure 10: Stereo vision system (adapted from Bradski and Kaehler 2008)

Since it is impossible to obtain such configuration physically, it can be done mathematically by conducting the following operations:

1. Undistortion: removing radial and tangential distortion in the images;
2. Rectification: row alignment of the images.

Images can be undistorted based on the distortion coefficients obtained during camera calibration. Then, one needs to rectify, or row-align, the images, i.e. to relate the images in a way when the features on both images are situated on the same rows. Figure 11) depicts two row-aligned images of a chessboard calibration object. The chessboard corners are identified on the first image, and horizontal lines are drawn through the pixel rows containing each first corner on the first image. It is seen that the lines intersect the same corners in the second image.

To make the images row-aligned, one needs to obtain the rectification transforms. This is done in two stages:.

1. Stereo calibration process is used to determine the relationships between two image planes. It results with rotation matrix, translation vector, essential and fundamental matrices. The latter matrix contains information about intrinsic parameters of the cameras, while the former three relate the image planes of the cameras.
2. Stereo rectification process is used to determine the rectification transforms: rotation matrix for the first camera, rotation matrix for the second camera, projection matrix for the first camera, projection matrix for the second camera, and disparity-to-depth mapping matrix.

Essential matrix E and fundamental matrix F are 3×3 matrices that relate the image planes of two cameras. If p_l and p_r are the location in the physical coordinates of a point P seen on each

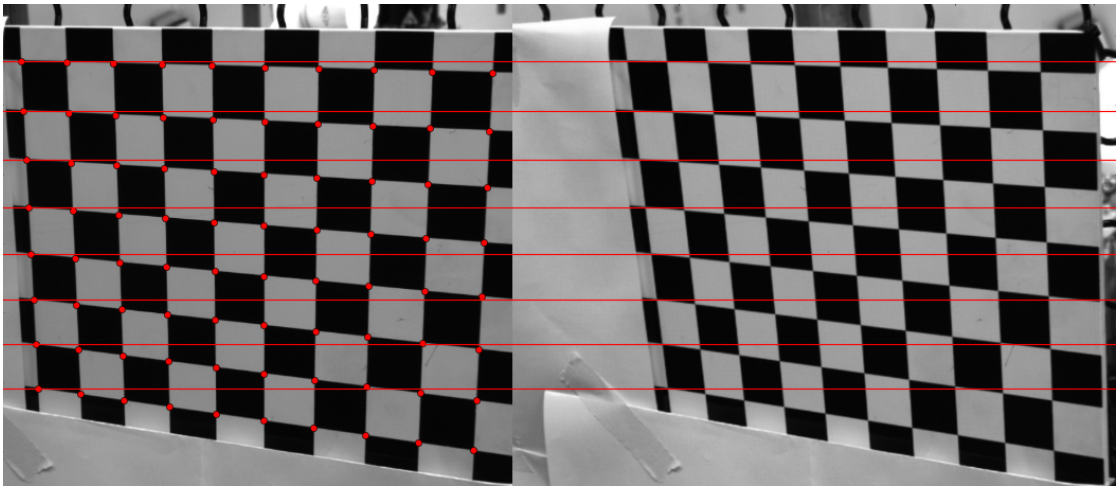


Figure 11: Example of row-alignment of the images

of the images, and q_l and q_r are the corresponding pixel values of the coordinates, the following relations apply (Bradski and Kaehler 2008):

$$p_r^T E r_l = 0 \tag{4.20}$$

$$q_r^T F q_l = 0 \tag{4.21}$$

4.1.4 Hand-eye calibration

Overview of the problem

To calibrate a hand-eye device means to find a transformation matrix between a sensor coordinate frame and the flange coordinate frame. When solving this problem, the coordinate frames of the following objects are considered (Figure 12):

1. Robot's base;
2. Sensor (camera) mounted on a robot – "eye";
3. Gripper flange – "hand";
4. Calibration object.

Let G be the transformation from the flange frame to the base frame, V – the transformation from the object frame to the sensor frame, X – the transformation from the sensor frame to the flange frame (Figure 13).

To find the desired X matrix, the robot makes a number of movements. For each of the movements, transformations G_i and V_i are recorded. G_i is supplied by the robot, and V_i is computer by a computer vision system after taking a picture of the calibration object (as extrinsic parameters of the camera).

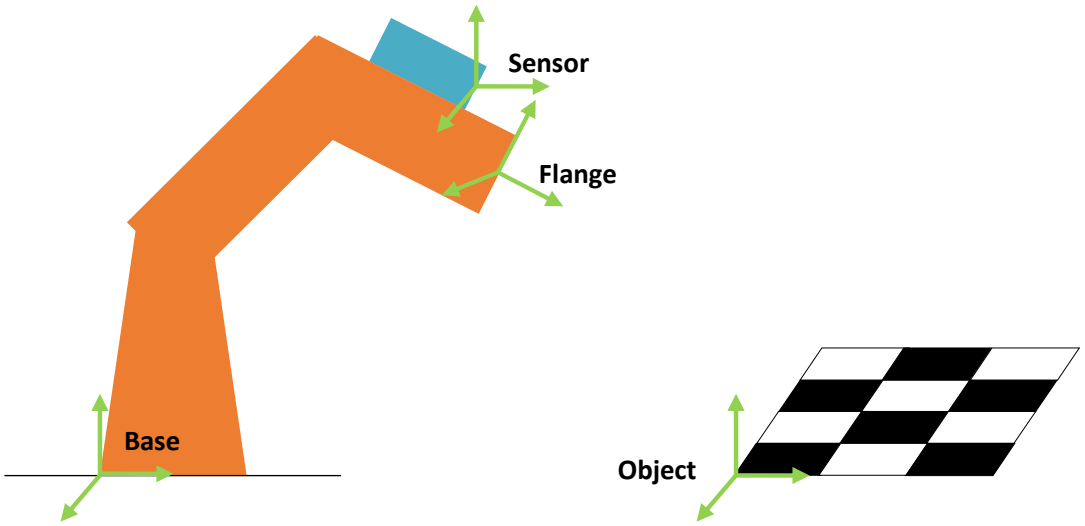


Figure 12: Coordinate frames

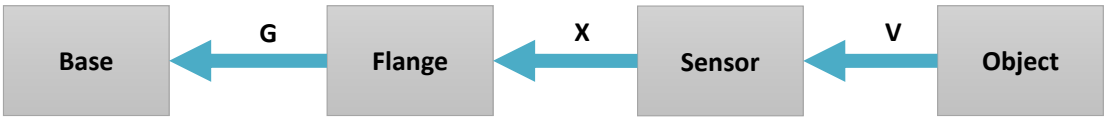


Figure 13: Transformations between the coordinate frames

To represent each move, matrices A and B are introduced, where A is the transformation from the old to the new position of the flange coordinate frame, and B is the transformation from the old to the new position of the sensor coordinate frame. If the movement from position 1 to position 2 is considered (Figure 14), A and B are calculated as follows:

$$A = {G_2}^{-1} G_1 \tag{4.22}$$

$$B = V_2 {V_1}^{-1} \tag{4.23}$$

The latter equations are derived from the following relations (see also Figure 14):

$$G_1 = G_2 A \Rightarrow A = {G_2}^{-1} G_1 \tag{4.24}$$

$$V_2 = B V_1 \Rightarrow B = V_2 {V_1}^{-1} \tag{4.25}$$

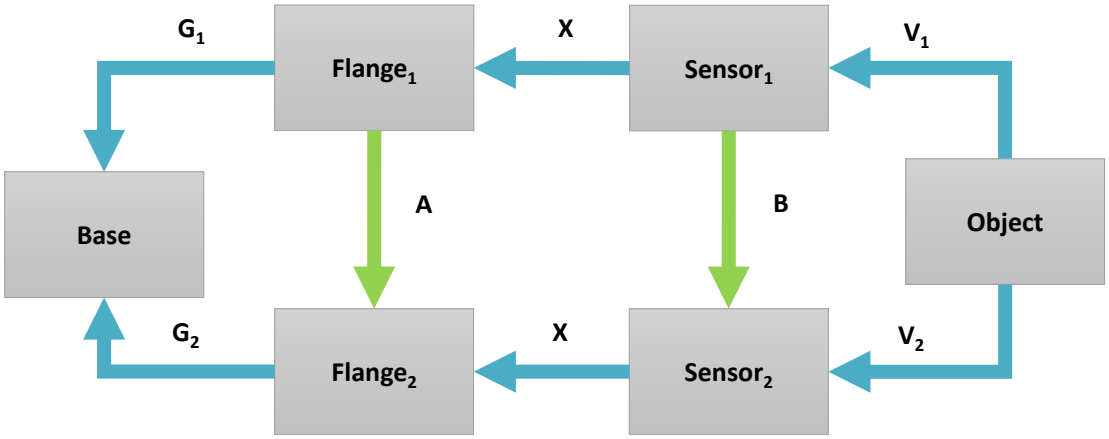


Figure 14: Transformations involved in one movement

Because the sensor is mounted on the robot, the desired X matrix remains the same for all movements of the robot. By applying the rule for composition of transformations for one move, the following equation is obtained:

$$G_1 X V_1 = G_2 X V_2 \tag{4.26}$$

It is then transformed:

$$G_2{}^{-1} G_1 X = X V_2 V_1{}^{-1} \tag{4.27}$$

$$AX = XB \tag{4.28}$$

Having a number of pose pairs (G, V), it is possible to compute a set of all possible movement pairs (A, B) as all combination of poses.

Algorithm 1 Computation of all movement pairs

```
for i = 1 to n do
    G1 = Gi, V1 = Vi
    for j = i + 1 to n do
        G2 = Gj, V2 = Vj
        A = G2^-1 G1
        B = V2 V1^-1
    end for
end for
```

Methods of solving AX=XB equation

A number of methods were developed to solve equation 4.28 and obtain the value of transformation between the object reference frame and the sensor reference frame. These methods are

reviewed by Strobl and Hirzinger 2006 and Fassi and Legnani 2005. The mathematical details of methods described by Park and Martin 1994 and Tsai and Lenz 1989 are presented below.

In both approaches, the solutions for rotational part of X is decoupled from the translational one. To do that, equation 4.28 is splitted in two parts:

$$R_A R_X = R_X R_B \tag{4.29}$$

$$R_A t_X + t_A = R_X t_B + t_X \tag{4.30}$$

The method of Park and Martin 1994 solves for the rotation al part of X matrix (R_X) first. If at least two movement pairs are available, R_X is found as:

$$R_X = \mathcal{A}\mathcal{B}^{-1} \tag{4.31}$$

where $\mathcal{A}$ and $\mathcal{B}$ are computed as follows:

$$\mathcal{A} = \begin{bmatrix} \log R_{A_1} & \log R_{A_2} & \log R_{A_1} \times \log R_{A_2} \end{bmatrix} \tag{4.32}$$

$$\mathcal{B} = \begin{bmatrix} \log R_{B_1} & \log R_{B_2} & \log R_{B_1} \times \log R_{B_2} \end{bmatrix} \tag{4.33}$$

After rotation matrix R_X is obtained, the solution to the translation vector t_X is found by solving the following systems of equations:

$$\begin{bmatrix} R_{A_1} - I \\ R_{A_2} - I \\ \cdots \\ R_{A_N} - I \end{bmatrix} t_X = \begin{bmatrix} R_X t_{B_1} - t_{A_1} \\ R_X t_{B_2} - t_{A_2} \\ \cdots \\ R_X t_{B_N} - t_{A_N} \end{bmatrix} \tag{4.34}$$

Because there is always noise present in the measurements, in order to obtain the best possible solution, the least squares fitting is performed to minimize the following error criterion:

$$q = \sum_{i=1}^{k} d(A_i X, X B_i) \tag{4.35}$$

where d is some distance metric on the Euclidean group.

The method of Tsai and Lenz 1989 provides closed form solution by performing the least squares fitting of rotations, and then translations.

In this method, those move pairs (i, j) that have the largest rotational angles between A_i and A_j (or B_i and B_j) will give the best results. Once the move pairs are chosen, the following system of equations is solved against the unknown P'_X:

$$\begin{bmatrix} Skew(P_{A_1} + P_{B_1}) \\ Skew(P_{A_2} + P_{B_2}) \\ \cdots \\ Skew(P_{A_N} + P_{B_N}) \end{bmatrix} P'_X = \begin{bmatrix} P_{B_1} - P_{A_1} \\ P_{B_2} - P_{A_2} \\ \cdots \\ P_{B_N} - P_{A_N} \end{bmatrix} \tag{4.36}$$

In equation 4.36 P_{A_i} and P_{B_i} represent the axes of rotations representing R_{A_i} and R_{B_i} respectively. After P'_X is determined, to calculate P_X, an axis of rotation specifying R_X, the following relation is applied:

$$P_X = \frac{2P'_X}{\sqrt{1 + |P'_X|^2}} \tag{4.37}$$

As in the method of Park and Martin 1994, to find the translational part of X, system of equation 4.34 is solved.

The problem of removing outliers

Both previously described methods account for noise in the measurements. They apply the "best fit" solutions, i.e. find the sensor-in-flange transformation by having a set of noisy measurements of the move pairs.

To make the results of the hand-eye calibration methods more accurate, some of the move pairs should be excluded from the calibration process as the outliers, i.e. those providing unreliable pose information (Schmidt, Vogt, and Niemann 2003).

Schmidt, Vogt, and Niemann 2003 overviews the following methods for outliers removal (Schmidt, Vogt, and Niemann 2003, p. 6):

1. Removing the movement pairs having very high change in translation of the robot arm;
2. Applying an iterative approach, in which all the movement pairs are used first, and then those are removed that lead to very high errors in comparing the hand-eye transformed robot arm poses with the data from the calibrated camera;
3. RANSAC approach with the same error measure as in the previous item.

4.1.5 Precision, accuracy and repeatability

In order to assess and improve the quality of calibration process, the notions of precision, accuracy and repeatability are explained below.

"International vocabulary of metrology" defines measurement *accuracy* as "closeness of agreement between a measured quantity value and a true quantity value of a measurand" (JCGM 2008, p. 21). The document also notes that accuracy is not a quantity expressed as a numerical value but an attribute of a measurement: a measurement is said to be more accurate if it results in a smaller measurement error. Accuracy, according to Aikens 2011, is a matter of calibration, and “can be determined only by repeatedly measuring a standard that has a known true value” (Aikens 2011, p. 294).

Measurement *precision* is defined as "closeness of agreement between indications or measured quantity values obtained by replicate measurements on the same or similar objects under specified conditions" (JCGM 2008, p. 22). As Aikens 2011 notes, precision is “the ability of a measurement process to repeat its results” (Aikens 2011, p. 292), i.e. the more precise the process, the less variability around its mean it has. Typical measures of precision are standard deviation, variance and coefficient of variation (JCGM 2008).

Figure 15 graphically depicts the notions of accuracy and precision.

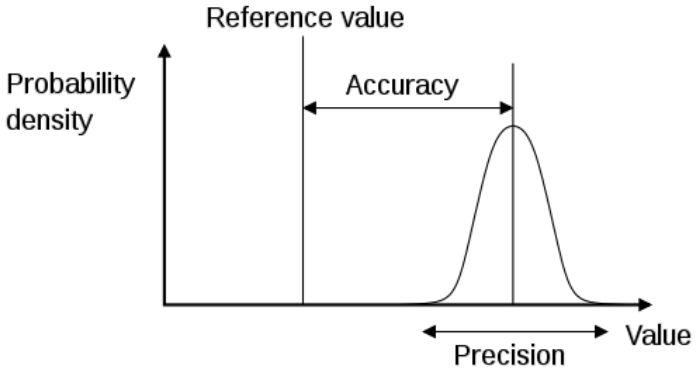


Figure 15: Accuracy and precision

Measurement *repeatability* is defined as "measurement precision under a set of repeatability conditions of measurement" (JCGM 2008, p. 24). The latter, in turn, means "condition of measurement, out of a set of conditions that includes different locations, operators, measuring systems, and replicate measurements on the same or similar objects" (JCGM 2008, p. 24).

Thus, the concept of repeatability is used when the considered measurement process is meant to yield the same results over time, and the variability between the measurements should be as little as possible. A related notion of *reproducibility* is measured as a variability between the measurements under the changed conditions (Bartlett and Frost 2008).

4.2 Method

To bring about better understanding of the vision systems, and most notably, the calibration processes, the approach was taken to develop a software library suitable for interactive computation and rapid application development. Eventually, by exploring various aspects of the calibration and adjacent processes, the new algorithms for improving the results of camera calibration and hand-eye calibration can be proposed, and the very processes can be better understood.

The developed library, dubbed FlexVi (Semeniuta 2014), was used for the abovementioned research undertakings. FlexVi is written in Python and based on a popular open source computer vision library OpenCV (OpenCV 2014). This choice was dictated by the suitability of Python programming language for the purposes of interactive computing, the vast popularity of OpenCV library for computer vision tasks, and the available Python API in OpenCV.

Other components on which FlexVi is dependent are the following Python libraries:

1. NumPy, a library providing support for arrays manipulations (NumPy 2014);
2. PyMath3D, an "Euclidean mathematics library for working with positions, vectors, orientations, rotations, reference systems, homogeneous transforms, and linear interpolations" (Morten Lind 2012, p. 174);
3. SciPy, a library of various scientific routines (SciPy 2014);
4. Pandas, a library for statistical data manipulation and analysis (Pandas 2014);

5. Matplotlib, a library for 2D visualization (Matplotlib 2014).

FlexVi development and testing was done based on Python 2.7.6 platform running on the following machines:

1. HP ProBook laptop computer running Windows 7 Professional and WinPython distribution (WinPython 2014), version 2.7.6.2.
2. VirtualBox virtual machine running Ubuntu Linux 14.04 and the standard Python distribution.

The structure of FlexVi library is presented in Table 3.

Table 3: FlexVi library structure

Package	Description
`flexvi`	The root package
`flexvi.calibration`	Vision calibration functionality
`flexvi.calibration.containers`	Classes-containers related to vision calibration
`flexvi.calibration.ti`	Modules related to identification of "true intrinsics"
`flexvi.confmanager`	Management of configuration files
`flexvi.dataanalysis`	Modules used for data analysis and statistical operations
`flexvi.handeye`	Hand-eye calibration functionality
`flexvi.handeye.hecalibrators`	Hand-eye calibrator classes
`flexvi.handeye.outliers`	Modules related to the removal of outliers from the movement data
`flexvi.opencv`	Modules providing interaction with OpenCV library
`flexvi.transform`	Modules dealing with homogeneous transformations

The implementations of hand-eye calibration methods included in the `flexvi.handeye.hecalibrators` package are the following GPL-licensed modules:

1. `parkmartin`, an implementation of the method of Park and Martin 1994 written by Morten Lind;
2. `tsailenz`, an implementation of the method of Tsai and Lenz 1989 written by Lars Tingelstad.

Image data was taken using a system of two industrial Ethernet cameras (Figure 16):

1. Prosilica GC1350 with Fujinon HF9HA-1B 9mm lens;
2. Prosilica GC1350C (color version) with Fujinon HF9HA-1B 9mm lens.

Data for both camera calibration (images of the chessboard calibration pattern) and for hand-eye calibration were gathered by Ådne Solhaug Linnerud using KUKA KR 60 HA industrial robot.

The chessboard pattern used for camera calibration has 10×8 corners to be identified, with the square size (length between two neighbor corners) of 30 mm (Figure 17).

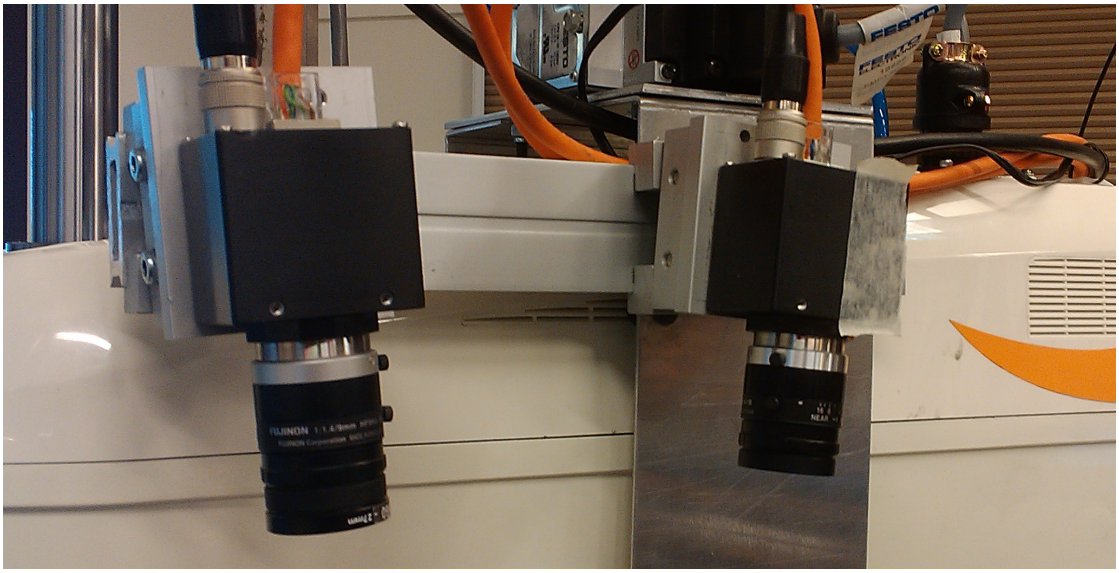


Figure 16: Prosilica GC1350 and GC1350C cameras mounted on the robot

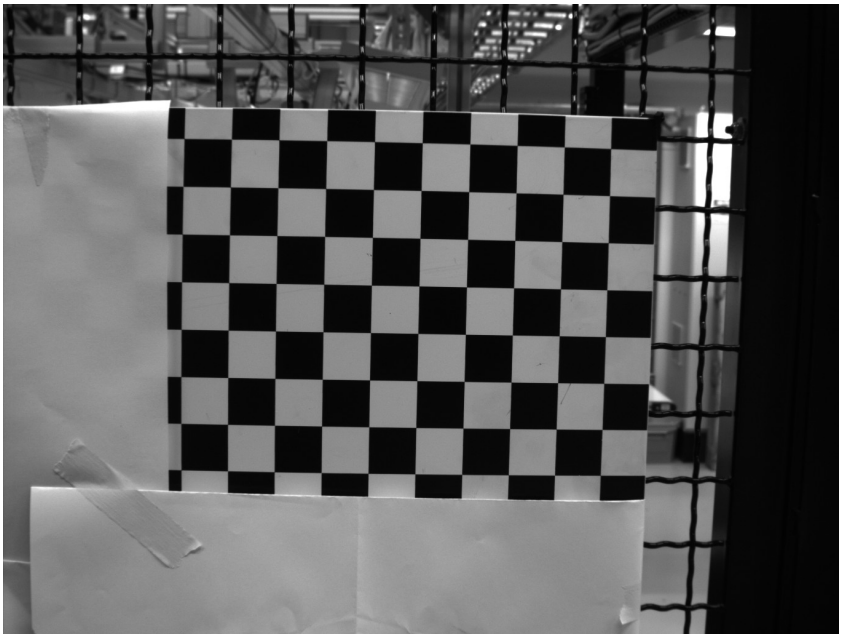


Figure 17: Chessboard calibration object

Data for hand-eye calibration (pose pairs) was acquired by the Scorpion vision software (Scorpion 2014) toolbox, developed in-house at SRM by Ådne Solhaug Linnerud. The toolbox computed the camera extrinsic parameters and fetched the corresponding robot positions from the robot controller via KUKA KRL XML interface (KUKA 2006).

In order to fetch camera extrinsic parameters, the Scorpion toolbox identifies position and

orientation of the calibration object and returns the transformation from the camera coordinate frame to the object coordinate frame.

The calibration object (Figure 18), created by Ådne Solhaug Linnerud, includes 12 black circles on a white surface. The circle in the bottom right corner has the biggest size; two circles of smaller size are located in the top right and bottom left corners; all the rest of circles are of the smallest size. This pattern allows for clear identification of the position and orientation of the calibration object in space.

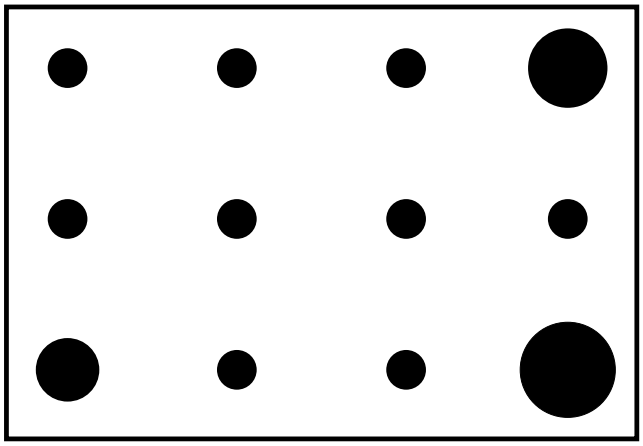


Figure 18: Calibration object for hand-eye calibration

As mentioned above, the results of the Scorpion toolbox for gathering calibration data are the transformations from the camera coordinate frame to the object coordinate frame. This type of transformation is the opposite to the one usually described in the literature (from the object to the camera), and is induced by the specifics of the used Scorpion tools. This facts leads to the minor changes in the way the movement transformations A and B are computed.

As was mentioned above, in the approach used at SRM, V matrix represents transformation in the opposite direction from the one described in the literature: from the camera coordinate frame to the object coordinate frame. If the direction of the A and B transformations are changed as well (Figure 19), it allows for using the relation of the same type for A and B calculation:

$$G_2 = G_1 A \Rightarrow A = {G_1}^{-1} G_2 \tag{4.38}$$

$$V_2 = V_1 B \Rightarrow B = {V_1}^{-1} V_2 \tag{4.39}$$

4.3 Results

4.3.1 FlexVi library

In this subsection the processes of camera calibration and stereo vision system parameterization are shown in terms of FlexVi's packages and the underlying OpenCV functions.

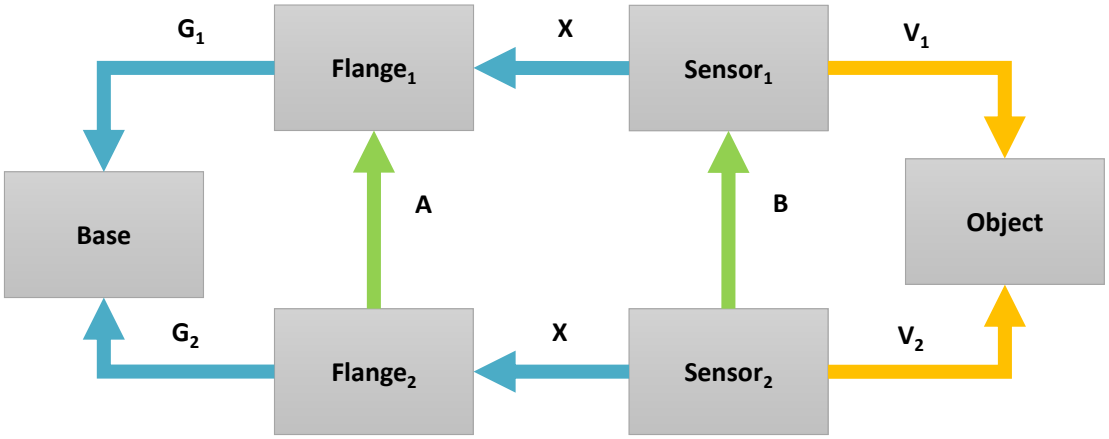


Figure 19: Transformations involved in one movement (SRM approach)

To perform camera calibration using OpenCV, the `cv2.calibrateCamera` function is applied (function names starting with `cv2` refer to the functions from the OpenCV's Python API). It uses a set of images of the chessboard pattern, taken from different angles, and returns the camera's intrinsic parameters (camera matrix and distortion coefficients) and extrinsic parameters (rotation matrix and translation vector) for each view. The function takes the following arguments:

1. `objectPoints` – a list of matrices for each chessboard view that constitute the coordinates of each chessboard corner in real-world coordinates. The center of the reference system is assumed to be in the top left corner, and Z-coordinate of each point is equal to 0. Function `flexvi.opencv.calibration.get_object_points` simplifies creation of such list of matrices.

2. `imagePoints` – a list of matrices with found chessboard corners in pixel coordinates as it is returned by `cv2.findChessboardCorners` function. The latter, applied to each view, returns a Boolean signalizing about success of finding chessboard corners, and a matrix with corners results.

3. `imageSize` – image size as $(width, height)$.

The process of camera calibration, implemented in FlexVi, is schematically depicted in Figure 20. First the images are opened and `cv2.findChessboardCorners` function is called upon each image. Images on which not all the corners were found are then filtered-out, and camera calibration is performed. The results include calibration error, camera intrinsic parameters, and extrinsic parameters for each view (image).

To parameterize a stereo vision system consisting of two cameras, two following steps are performed: stereo calibration and stereo rectification. These steps and their outputs are briefly described below.

Stereo calibration process is used to determine the relationships between two image planes. It results with rotation matrix, translation vector, essential and fundamental matrices. The latter

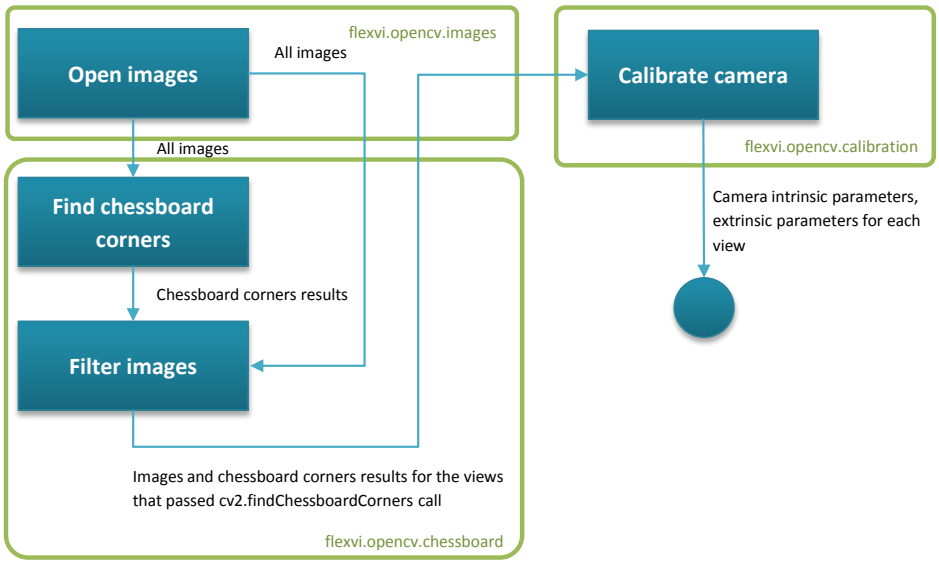


Figure 20: Camera calibration

matrix contains information about intrinsic parameters of the cameras, while the former three relate the image planes of the cameras.

To perform stereo calibration, OpenCV's function `cv2.stereoCalibrate` is used. The parameters of the function include:

1. `objectPoints` – the same object points (for chessboard pattern) as for camera calibration;
2. `imagePoints1`, `imagePoints2` – corresponding image points in the images for the first and second camera.
3. `cameraMatrix1`, `distCoeffs1`, `cameraMatrix2`, `distCoeffs2` – intrinsic parameters of the cameras, obtained during camera calibration; these parameters are optional, and if they are not provided, the corresponding intrinsic parameters will be computed inside the function.

Stereo rectification is aimed at mathematically achieving row-alignment of the images. OpenCV's function `cv2.stereoRectify` computes rectification transforms: rotation matrix for the first camera R_1, rotation matrix for the first camera R_2, projection matrix for the first camera P_1, projection matrix for the second camera P_2, and disparity-to-depth mapping matrix. Matrices P_1 and P_2 are analogous to the corresponding camera matrices, and are used in `cv2.triangulatePoints` function for reconstructing 3D coordinates of the points.

FlexVi's implementation of stereo vision system parameterization (Figure 21) is conceptually similar to the described above camera calibration (Figure 20).

After the rectification transform are obtained, the projection matrices P_1 and P_2 can be used to reconstruct real-world 3D coordinates of the points, detected in pixel coordinates. To

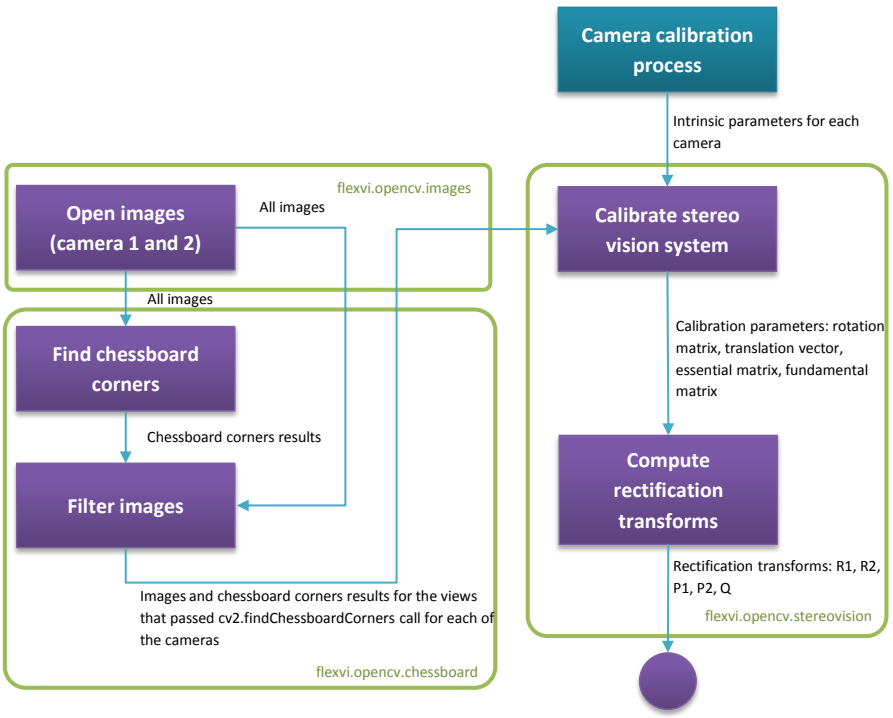


Figure 21: Stereo vision system parameterization

do that, OpenCV's function `cv2.triangulatePoints` is used. The function takes the following arguments:

1. `projMatr1`, `projMatr2` – 3×4 projection matrices P_1 and P_2, obtained after computing rectification transforms for current stereo vision system;

2. `projPoints1`, `projPoints2` – matrices containing the interest points coordinates on each of the images of the pair; the array shape for these parameters is $2 \times N$, where N is the number of points, that is, each column in the array represent a point's coordinates.

The result of this function call is $N \times 4$ matrix in which each row represent the corresponding point's homogenous coordinates (c_1, c_2, c_3, c_4). To obtain the Cartesian coordinates, the formulas 4.5 –4.7 are applied.

4.3.2 Finding the true intrinsics by distribution fitting

During investigation of the nature of camera calibration it was noticed that different sets of images used for calibration lead to different camera intrinsic parameters. The question arose which values of the calculated intrinsics are closest to the real ones.

To investigate the abovementioned question a statistical method was applied. From the set of all available chessboard images, m samples of the size n were randomly chosen. Each sample

was used to perform camera calibration, and the resulting intrinsic parameters were analyzed as the following distributions:

1. Distribution of the values of focal length f_x: $f_{x_1}...f_{x_m}$;
2. Distribution of the values of focal length f_y: $f_{y_1}...f_{y_m}$;
3. Distribution of the values of camera center c_x: $c_{x_1}...c_{x_m}$;
4. Distribution of the values of camera center c_y: $c_{y_1}...c_{y_m}$;
5. Distribution of the values of distortion coefficient k_1: $k_{1_1}...k_{1_m}$;
6. Distribution of the values of distortion coefficient k_2: $k_{2_1}...k_{2_m}$;
7. Distribution of the values of distortion coefficient p_1: $p_{1_1}...p_{2_m}$;
8. Distribution of the values of distortion coefficient p_2: $p_{2_1}...c_{2_m}$;
9. Distribution of the values of distortion coefficient k_3: $k_{3_1}...k_{3_m}$.

From the obtained distributions it is evident that each intrinsic parameter has a clear central tendency and is approximately normally distributed. To find the assumed true values of each intrinsic parameters, distribution fitting is applied to each distribution. The resulting central tendencies are taken as the true intrinsic values.

The result of the abovementioned procedure applied to the f_x distribution is depicted in Figure 22. The rest of the distributions are presented in Appendix A.

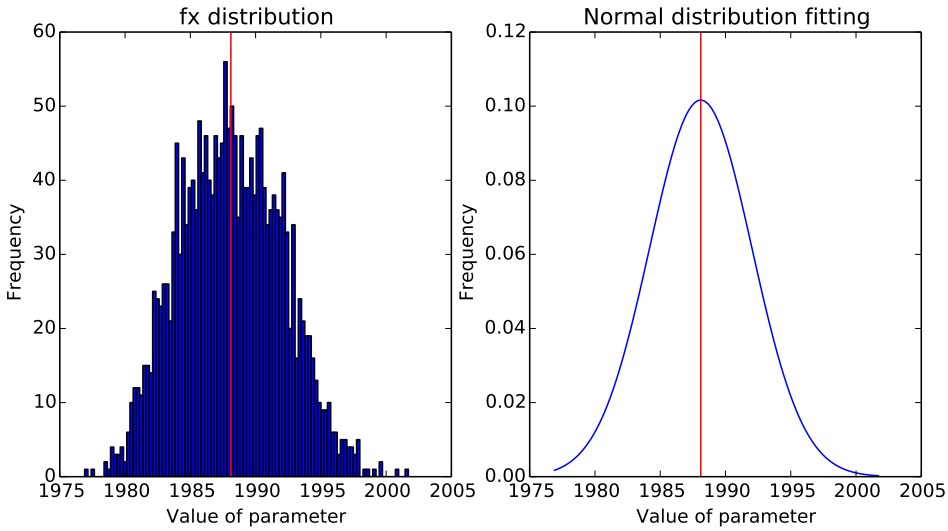


Figure 22: f_x distribution

4.3.3 Achieving repeatability of stereo calibration

In the process of stereo vision systems calibration, the intrinsic parameters fro each cameras are fed into the stereo calibration algorithm. It is naturally that one wants to feed the most accurate intrinsics. The previously described method of distribution fitting returns more accurate results only if a bigger number of samples are used. This makes this method computationally intensive and time consuming.

The method described in this subsection is aimed at conducting stereo vision system calibration with a relatively small amount of image samples used to calibrate the cameras (e.g. $m = 5$) with the focus on achieving repeatability. This method, dubbed *repeatability-oriented stereo calibration* is presented below:

1. From two given image sets (from left and right cameras) generate m samples of size n;
2. For each sample open the images as find chessboard corners;
3. Use each sample of the first camera to conduct camera calibration;
4. Compare the calibration error value (RMS) for all calibrations and choose the sample which was used to calibrate the camera with the smallest error value;
5. Use the chosen sample to calibrate the second camera and the stereo vision system.

To asses this method, repeatability-oriented stereo calibration was conducted N times, and the elements of the fundamental matrix F (the matrix relating two image plane in pixel coordinates) were compared.

$$F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \tag{4.40}$$

Table 4 shows the variances of the elements of the fundamental matrix (f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}) obtained in N runs of the calibration. The detailed results of this experiment are presented in Appendix B.

Table 4: Variances of each elements of the fundamental matrix obtained in N runs of repeatability-oriented stereo calibration

Element	Variance
f_{11}	5.937E-20
f_{12}	2.696E-16
f_{13}	9.491E-11
f_{21}	9.294E-16
f_{22}	1.243E-16
f_{23}	3.243E-07
f_{31}	1.667E-10
f_{32}	3.103E-07
f_{33}	0.000E+00

As the table 4 shows, the values of the variances are close to zero, which indicates that the prosed method is repeatable since it yields very close results in each run, regardless which images sample was used.

4.3.4 Hand-eye move pairs outliers elimination by precision assessment

The quality of each movement must be evaluated by comparing transformation matrices AX and XB. They derive from the equation $AX = XB$, and, thus, the bigger the difference between the matrices, the more chances for the particular movement pair to be considered as an outlier.

Having the abovementioned considerations in mind, the following method is proposed:

1. Hand-eye calibration is performed using one of the methods (e.g. Park and Martin 1994) with all possible movements;
2. For each of the movements, a distance between transformations AX and XB ($dist_i$) is calculated, where X is the result of hand-eye calibration using all of the movements (performed in step 1).
3. Having a threshold value $dist_{max}$, from the set of all movements only those are retained that comply to the criterion $dist_i < dist_{max}$.

The proposed methods of outliers elimination has an unresolved question: how to set the maximal threshold value ($dist_{max}$). If the taken threshold value is too small, too many movement pairs will be eliminated, which will lead to unreliable results of the hand-eye calibration. If, otherwise, the threshold is too big, some of the hypothesized outliers may stay in the set.

To find a threshold value that leads to the best results of hand-eye calibration after the outliers are eliminated, a quality evaluation method should be proposed. Here the evaluation method is based on the notion of precision (see 4.1.5).

Let T_{object}^{base} be the transformation from the calibration object to the robot base, computed by applying the rule of composition of transformations:

$$T_{object}^{base} = GXV \tag{4.41}$$

Having the set of pose pairs and the value of X matrix, the precision of the T_{object}^{base} matrix is proposed to be assessed.

Since T_{object}^{base} is a transformation matrix, it contains a rotation matrix and a translation vector. Let r_{ij} be the elements of rotation matrix, and d_i – the elements of translation vector ($i = 1,2,3; j = 1,2,3$).

$$T_{object}^{base} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & d_1 \\ r_{21} & r_{22} & r_{23} & d_2 \\ r_{31} & r_{32} & r_{33} & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.42}$$

The abovementioned assessment of precision is proposed to be conducted in regard to each individual element of the translation vector, i.e. d_1, d_2, d_3. This is done as follows:

1. For each of the pose pairs, the transformation from object to base is computed: $T_{object}^{base} = RXV$;

2. From each computed T_{object}^{base} matrix the d_1, d_2, d_3 translation components are extracted;
3. For each of the resulting samples – d_1, d_2, and d_3 – the corresponding variance is calculated: Var_{d_1}, Var_{d_2}, Var_{d_3};
4. A sum of variances is calculated, which will serve as an indicator of the quality of precision: $Var_{\sum} = \sum_{i=1}^{3} Var_{d_i}$.

The described computation scheme is depicted in Figure 23.

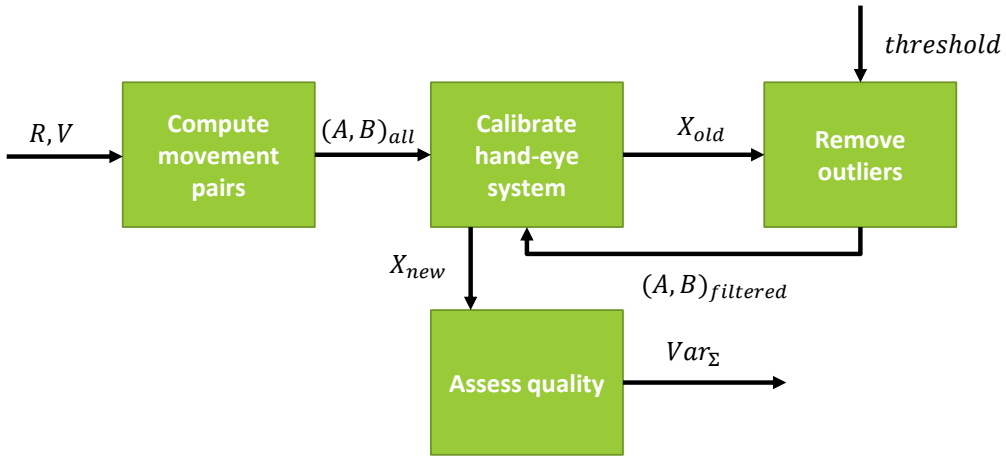


Figure 23: Algorithm of outliers elimination

To identify which threshold value gives better result of hand-eye calibration, the $Var_{\sum}$ should be calculated for calibration result before and after elimination of outliers for each of the threshold values. The threshold for which the value of $Var_{\sum}$ is minimal will be the optimal one.

The results of determining the optimal threshold value by precision assessment are presented in Figure 24.

4.4 Discussion

The work described in this chapter considered robot vision systems calibration and proposed methods for their improvement. To calibrate a robot vision system with a *eye-in-hand* configuration, one needs to perform three distinct calibration routines, namely robot calibration, camera calibration, and hand-eye calibration. Robot calibration was out of the scope of this thesis, and only calibration processes directly related to utilizing machine vision were considered. In addition, stereo vision systems calibration was included as a distinct topic.

In this work, the use of open source software was emphasized. The routines from OpenCV library were used for the tasks of finding chessboard corners, calibration a camera, calibrating a stereo vision system, and image transformation. The intention was to take the ready OpenCV's implementations of various computer vision algorithms and wrap them in a new library that would simplify the usage of the abovementioned routines and allow for conducting computational experiments and data analysis. This way, the FlexVi library was developed and used for

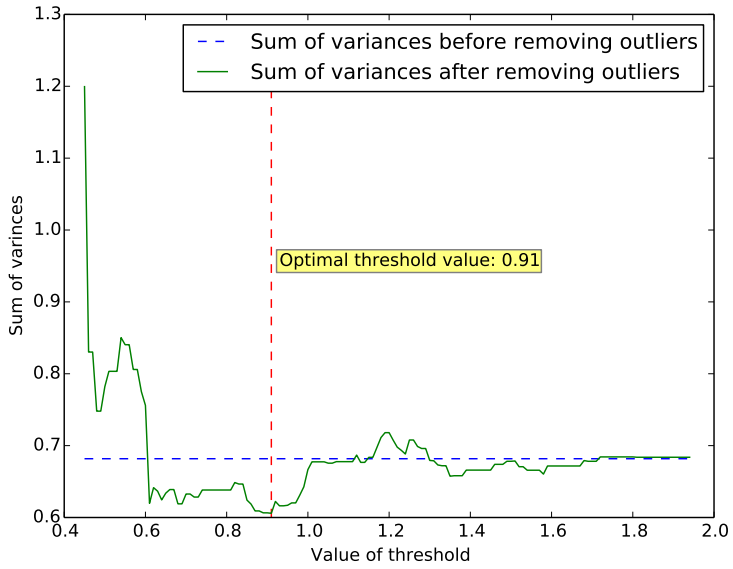


Figure 24: Results of precision assessment (sum of variances)

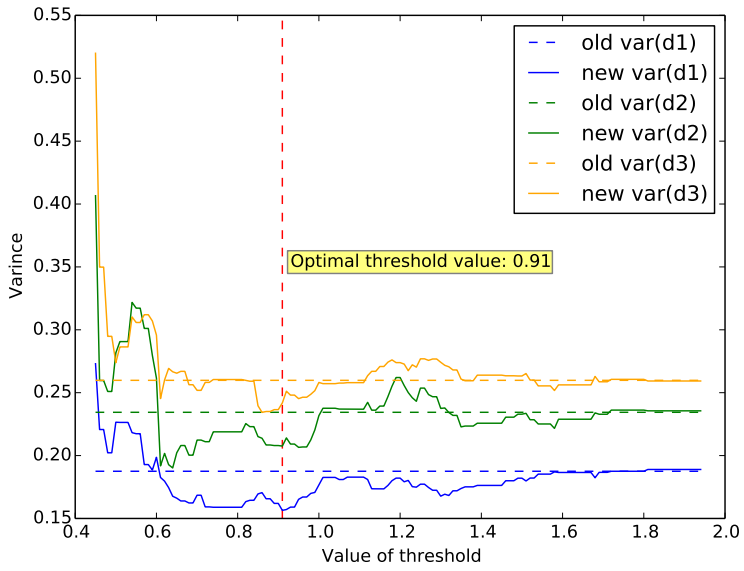


Figure 25: Results of precision assessment (translational components separated)

proposing the improvement methods for calibration processes. The functionality for hand-eye calibration was also included into FlexVi library.

To study the work of various calibration methods, a statistical approach was taken, and the

indicators of *accuracy*, *precision* an *repeatability* were considered as the goals of improvement.

With the camera calibration it was noticed that different image sets selected for the calibration yield different camera intrinsics. An experiment was conducted which involved calibration a camera using a big number of image samples with the subsequent analysis of the obtained distributions of the intrinsic parameters. It was assumed that the central tendencies of these distributions obtained by distribution fitting will be the true intrinsics values, and the bigger the number of samples used, the more *accurate* the results would be. The downside of the described process of finding the true intrinsics is that it is very time consuming if more accurate results are desirable.

Concerning stereo vision system calibration, a *repeatability*-oriented approach was taken. The proposed method is based on selecting an image set for camera calibration, from a relatively small number of sets, on the basis of the smallest calibration error. As the results show (see 4.3.3), this approach leads to the results that repeat themselves with extremely small variability.

In order to improve the results of hand-eye calibration, the procedure of outliers elimination was applied. The movement pairs (A, B) were excluded from the calibration process based on the criterion $\mathrm{dist_i} < \mathrm{dist_{max}}$, where $\mathrm{dist_i}$ is an Euclidean distance between the transformations AX and XB, and $\mathrm{dist_{max}}$ is a specified threshold value. An optimal threshold value was being chosen based on the considerations of *precision* of the measurement of object-to-base transformation, which was aimed to be maximized.

5 Discussion and further work

The work presented in this master thesis is based on the system approach. A topic of robot vision systems calibration for flexible assembly is considered at two magnifying levels: a higher level of flexible assembly and a lower level of robot vision systems calibration. Because of the difference in the systemic levels and the resulting difference in the research techniques, the work regarding these two themes was kept separated in the two respective chapters (3, 4).

This chapter is aimed at bringing the knowledge on software-defined assembly flexibility and robot vision system calibration together. This is going to be done by discussing the both themes from the systems perspective, elaborating of the adjacent topics out of the scope of this thesis, and outlining the further research challenges.

5.1 Towards a unified object-oriented framework for robot vision systems calibration

The developed library FlexVi, used for the research undertakings in the area of robot vision systems calibration, includes in its current status the functionality for camera calibration, stereo vision systems calibration, hand-eye calibration, the implementations of the methods described in this thesis, and other computer vision-related routines. The library thus has provided a higher layer on top of OpenCV and extended the area of OpenCV by the functionality related to hand-eye calibration and research-oriented computing.

Until its current state, FlexVi has been mostly developed in an ad-hoc manner. Firstly, the practical problems were solved by creating stand-alone scripts. Eventually, the often reused functionality was decomposed into packages and modules comprising the library. As the result, FlexVi, as a more formalized solution, appeared.

The challenge for the future is to further develop the library to a more unified and production-ready state. This includes the following improvements:

1. Creating a comprehensive documentation;
2. Improvement and unification of the library structure, API and data formats;
3. Establishing/enrichment of interoperability mechanisms with other software, most notably OpenCV, PyMath3D, Scorpion Vision Software, and other tools.

5.2 Increasing the share of the software subsystem in manufacturing and assembly systems

Chapter 3 resulted in distinguishing three subsystems in an assembly system, namely mechanical, software and human subsystem. Each of them can contribute to the flexibility level in their own way, and it was therefore proposed to delimit *mechanically-defined assembly flexibility*, *software-defined assembly flexibility*, and *human-defined assembly flexibility*.

In automated systems the human intervention is naturally minimized. If one considers only the mechanical and software subsystems, a previously mentioned idea is recalled: when a systems possesses a high degree of software-defined intelligence, one needs less complex mechanical part. In this regard the following question arises: how to identify and compare the shares of software part and hardware/mechanical part in a system? Obviously, it is not possible to quantify the shares in the percentage terms. However, if two or more systems are considered, one is able to compare their reliance on software and their hardware complexity.

Brody and Pureswaran 2013 propose a term *software-defined supply chain* by putting an emphasis on three emerging technologies, namely additive manufacturing, intelligent robotics and open source electronics. The authors claim that these technologies are creating manufacturing environment driven by digital data and thus transform a supply chain from being hardware-based to being software-defined. This thesis is closely connected with the area of intelligent robotics, and puts a particular emphasis on open source software. The role of other technologies mentioned by Brody and Pureswaran 2013 and the role of software in manufacturing should be studied further to make a ground for development of the brand new manufacturing systems.

5.3 Reflections on the applied computational method

The applied computational method is based on *interactive computing and data analysis*. Python programming language, in which all the computational work was performed, can be classified as multi-paradigm, i.e. allowing for simultaneous application of different programming paradigms (object-oriented, procedural, functional). In addition, it is permitted to develop at the same time both simple weekly-structured scripts and formal source code hierarchies. As was noted before, various functional parts of the FlexVi library were decomposed from simpler standalone scripts solving particular problems. Eventually, with the growth of the library, the possibilities for rapid development and testing were increasing.

For data analysis, the Pandas library was used. It allows for creating matrix-like data frames having the specified indices for rows and columns, and manipulating on them. For example, the method of outliers elimination in hand-eye calibration using precision assessment, described in 4.3.4, utilized the Pandas' functionality by manipulating the data frames of the following form:

- row indices correspond to the threshold values used for removing outliers;
- column indices correspond to the names of elements comprising a transformation matrix (or only the translational components of a transformation matrix);
- a value $V_{\mathrm{threshold,element}}$ constitutes a computed variance of an object-in-base transformation matrix element.

The applied computational method has proven its effectiveness when no hypotheses and anticipated statistical outcomes are known a-priori. It allows for exploring the available data in an interactive way, with the subsequent formulation/testing of hypotheses and solving the data-related problem computationally.

Whilst in this master thesis work interactive computing and data analysis were applied to the problem of robot vision systems calibration, there is a big potential for using this computational

method in other areas of manufacturing enterprise, especially when the available data are diverse and unstructured.

5.4 Open source software as a flexibility driver

A great deal of open source software was used in this master thesis work: computer vision library OpenCV, Euclidean mathematics library PyMath3D, GPL-licensed implementations of hand-eye calibrators, and Python libraries for scientific computing and engineering (NumPy, SciPy, Pandas, Matplotlib). The developed FlexVi library is also published as an open source project (Semeniuta 2014).

By its nature, open source software offers a greater degree of flexibility comparing to proprietary solutions. Because, by the licensing rights, the user is able to access and modify the source code, it brings awareness of the implementation and ability of quick change. It is, surely, unlikely that manufacturing companies possess a personnel whose responsibility includes patching the source code. Manufacturing companies by their nature tend to be conservative in the choice of software and therefore favor stable solutions. On the other hand, these companies can contribute to open source development in the way of reporting the problems and proposing functionality. Research and consulting companies can in turn be more actively involved in development of the open source projects.

It is important to note that in order to adopt open source products effectively, the manufacturing companies require highly-skilled personnel to support the software.

The idea of community-based development, which is applied in the majority of open source projects, is very close to the idea of open innovation (Van de Vrande et al. 2009). The latter constitutes an innovation model in which the R&D activities are performed in collaboration between the companies.

Concerning the theme of open source software in manufacturing, a big issue is of its integration with the proprietary solutions. The latter are likely to remain a larger portion among the software used by manufacturing companies. Therefore the interoperability between proprietary and open source software is a critical problem that is required for the adoption of the open source technologies.

5.5 Holonic architecture for vision systems

It was emphasized by a number of researchers that in order to achieve higher flexibility and reconfigurability, one needs a control software architecture possessing such qualities as autonomous design, modularity, compatibility, scalability and openness. It is evident that a paradigm of holonic and multi-agent systems complies to these criteria.

When it comes to the vision systems, it is of particular interest how can vision-related computing be integrated in the new holonic control systems.

Recalling the issue of effective integration of proprietary and open source software and the gained knowledge within robot vision systems calibration, the concept of *holonic-based calibration services* can be proposed. Whilst the majority of vision-related tasks can be performed using an industrial vision software such as Scorpion, the calibration tasks can be decomposed into a separated software module based on open source calibration solutions (e.g. OpenCV, FlexVi).

Communication between the vision software and calibration module can be established using an open source communication middleware, such as ZeroMQ or IceHMS. The latter, as mentioned in 3.1.3 is specially designed for developing holonic applications. Figure 26 depicts a simplified architecture of the proposed concept.

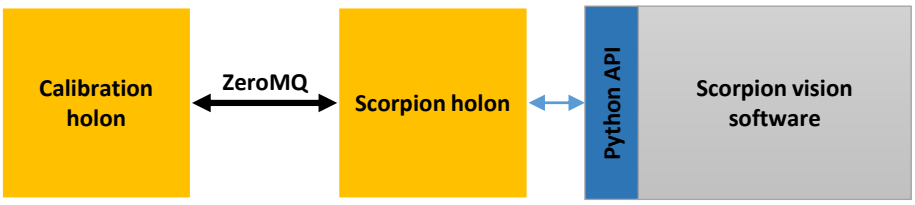


Figure 26: Architecture of a holonic-based calibration service

5.6 Seeing a big picture: other software levels

In chapter 3 an emphasis was put on flexible assembly and the software providing assembly flexibility. This is largely control software, i.e. directly involved in the control of the processes. It is important, though, to be aware of the other software levels. A generally accepted model providing categorization of the software levels is ANSI/ISA-95. The levels in this model (Figure 27) range from the lowest control to business logistics.

Colombo and Karnouskos 2014 envision the future possibilities in the industrial automation environment that can be brought up by such technologies as Service-oriented architecture (SOA) and Cloud computing. The new generation of systems, dubbed *cloud-based industrial cyber-physical systems*, can transform or complement the traditional hierarchical architecture of automation systems, as viewed by ANSI/ISA-95 model, with the flat architecture. "With the empowerment offered by modern SOA, the functionalities of each system or even device can be offered as one or more services of varying complexity, which may be hosted in the Cloud and composed by other (potentially cross-layered) services" (Colombo and Karnouskos 2014, p. 15). The authors' view of the future flat cloud-based information infrastructure is presented in Figure 28.

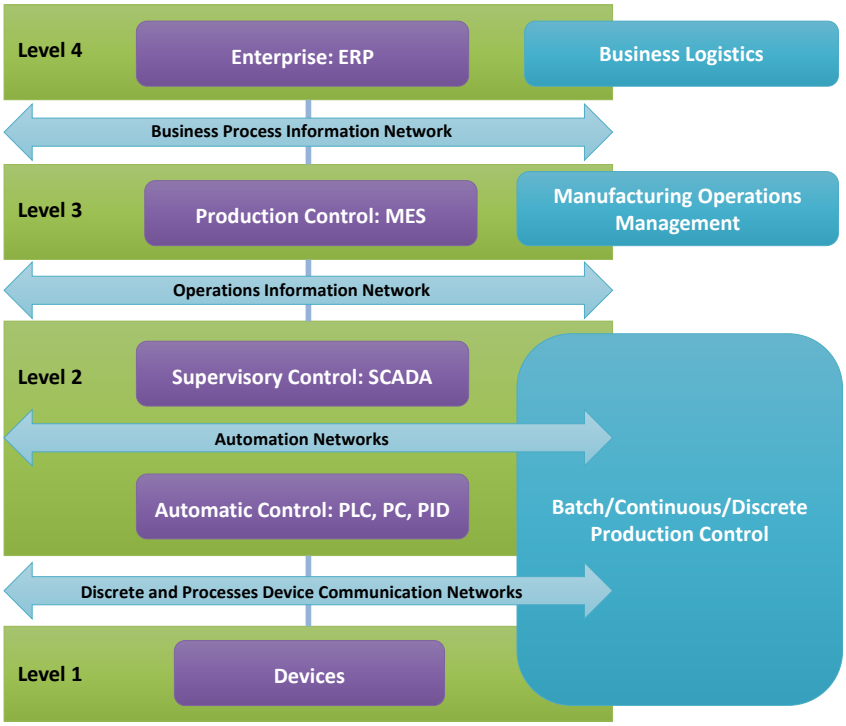


Figure 27: ANSI/ISA-95 model (adapted from Herrera, Ramos, and Lastra 2012)

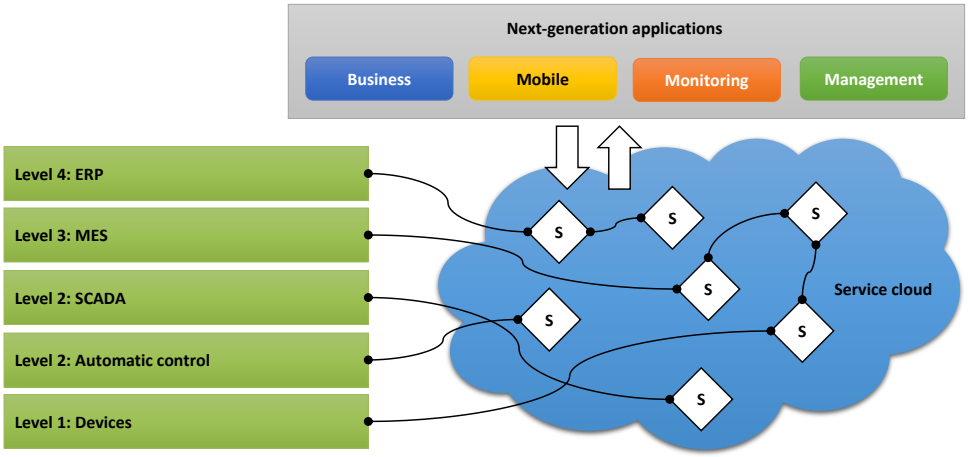


Figure 28: Flat cloud-based infrastructure (adapted from Colombo and Karnouskos 2014)

6 Conclusion

In master thesis, the notion of manufacturing flexibility was reviewed with the aim of bringing a better understanding of the concept. Then, the assembly process was reviewed from the flexibility perspective. Because the intention was to focus on the software part of an assembly system and utilization of machine vision, the latter themes were delimited from the non-software parts of FAS, namely mechanical and human subsystems. The role of software in FAS was theoretically presented, including the flexibility-oriented requirements for control systems, the paradigm of holonic manufacturing control, and the vision-defined flexibility as seen in the concept of F-FAS.

One of the most evident problems regarding flexibility in manufacturing is the one of lacking the agreed-upon taxonomy. Different but related notions of flexibility, reconfigurability, changeability etc. are discussed within the scientific community with the varying understandings. In this thesis, a term *manufacturing flexibility* was chosen in a broad perspective encompassing flexibility on various system levels of a manufacturing enterprise.

The abovemnetioned view on manufacturing flexibility is in line with the established categorization of flexibility types, to which a big number of scholars refer. In this thesis, the following types were identified as having the largest degree of reliance on software: machine flexibility, production flexibility, and program flexibility. It was concluded that the paradigm of holonic manufacturing control is well suited for provision of higher degree of manufacturing flexibility.

The assembly process was analyzed from the systems perspective: an assembly system was presented as consisting of a the mechanical subsystem, the software subsystem and the human subsystem. Each of the subsystems, when the assembly flexibility is considered, play a particular role in provision of this flexibility. Thus, the respective notions of mechanically-defined assembly flexibility, software-defined assembly flexibility, and human-defined assembly flexibility were proposed. In this thesis the emphasis was put on the following driving forces of software-defined assembly flexibility:

- Architectural solutions providing modularity, openness, compatibility etc.;
- Utilizing machine vision to allow robots and material handling equipment be more flexible.

The role of machine vision for assembly flexibility was discussed, leading to identifying the following problems in this regard:

- Big time for image processing;
- Lack of automated and reliable vision systems calibration.

The latter problem was the central in the subsequent chapter regarding calibration of robot vision systems.

In this master thesis a technique of magnifying levels was used by focusing on software-defined assembly flexibility and robot vision systems calibration separately. The latter theme was approached from a computational perspective. First, the theoretical base was overviewed and the important mathematical concepts were presented. The covered topics comprised vision-based robot control, camera calibration, stereo vision systems calibration, hand-eye calibration, and the statistical characteristics of precision, accuracy and repeatability. The practical part of the work constituted the computational improvement of robot vision systems calibration methods with the aim of maximizing the abovementioned statistical characteristics. To achieve this, a library, dubbed FlexVi, was developed and applied for interactive computing and data analysis.

The process of camera calibration was analyzed from the accuracy perspective: aiming at finding the most accurate values of the camera intrinsic parameters, a large number of calibration experiments was conducted resulting in the distributions of intrinsic parameters. The true values of the latter were computed using the distribution fitting. A repeatable method for stereo vision systems calibration was proposed based on selecting the camera calibration results having the smallest calibration error. A method of outliers elimination for hand-eye calibration was proposed based on maximizing the precision of measurements of hand-eye-transformed flange poses.

The use of open source software was emphasized in this master thesis. It was noted that open source may serve as a solution to achieving more standardized information infrastructure and improve interoperability, which is a critical factor leading to higher reconfigurability potential. The developed library FlexVi is itself based on the open source components and was published as an open source project. It was concluded that by adopting open source software into manufacturing enterprises it is possible to achieve greater flexibility. This, however, needs highly-skilled professionals and good interoperability between open source and proprietary software products.

Bibliography

Aikens, C. Harold (2011). *Quality Inspired Management. The Key to Sustainability*. Prentice Hall.

Arbnor, I. and B. Bjerke (2008). *Methodology for Creating Business Knowledge*. Sage Publications (CA). ISBN: 9781446244074.

Bartlett, J. W. and C. Frost (2008). "Reliability, repeatability and reproducibility: analysis of measurement errors in continuous variables". In: *Ultrasound in Obstetrics and Gynecology* 31.4, pp. 466–475. ISSN: 1469-0705.

Bi, ZM et al. (2008). "Reconfigurable manufacturing systems: the state of the art". In: *International Journal of Production Research* 46.4, pp. 967–992. ISSN: 0020-7543.

Bradski, G. and A. Kaehler (2008). *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media. ISBN: 9780596554040.

Brody, Paul and Veena Pureswaran (2013). *The new software-defined supply chain. Preparing for the disruptive transformation of Electronics design and manufacturing*. Report.

Chaumette, F. and S. Hutchinson (2006). "Visual servo control, Part I: Basic approaches". In: 13.4, pp. 82–90. URL: http://www.irisa.fr/lagadic/pdf/2006_ieee_ram_chaumette.pdf.

Chen, I. Ming (2001). "Rapid response manufacturing through a rapidly reconfigurable robotic workcell". In: *Robotics and Computer-Integrated Manufacturing* 17.3, pp. 199–213. ISSN: 0736-5845.

Chirn, J and D McFarlane (2000). "Application of the holonic component-based approach to the control of a robot assembly cell". In: *IEEE Conference on Robotics and Automation, San Fransisco*. Vol. 268.

Colombo, Armando Walter and Stamatis Karnouskos (2014). "Cloud-Based Industrial Cyber-Physical Systems". In: *ERCIM NEWS* 97.

Edmondson, NF and AH Redford (2002). "Generic flexible assembly system design". In: *Assembly Automation* 22.2, pp. 139–152. ISSN: 0144-5154.

ElMaraghy, Hoda A. (2006). "Flexible and reconfigurable manufacturing systems paradigms". In: *International Journal of Flexible Manufacturing Systems* 17.4, pp. 261–276. ISSN: 0920-6299.

EUnitedRobotics (2014). *Multi-Annual Roadmap for Robotics in Europe*. Web Page. Accessed: 2014-05-04. URL: http://www.eu-robotics.net/cms/upload/PDF/Multi-Annual_Roadmap_2020_Call_1_Initial_Release.pdf.

Fassi, Irene and Giovanni Legnani (2005). "Hand to sensor calibration: A geometrical interpretation of the matrix equation AX=XB". In: *J. Robot. Syst.* 22.9, pp. 497–506. ISSN: 0741-2223.

Gellein, Lars Tore and Per Aage Nyen (2010). "APROX - Agent-based Product-Resource-Order eXecution From solution concept to application development framework". In: *CATS 3rd CIRP Conference on Assembly Technologies and Systems*.

Heping, Chen et al. (2008). "Flexible assembly automation using industrial robots". In: *Technologies for Practical Robot Applications, 2008. TePRA 2008. IEEE International Conference on*, pp. 46–51.

Herrera, Vladimir Villaseñor, Axel Vidales Ramos, and José L Martínez Lastra (2012). "An agent-based system for orchestration support of web service-enabled devices in discrete manufacturing systems". In: *Journal of Intelligent Manufacturing* 23.6, pp. 2681–2702. ISSN: 0956-5515.

JCGM (2008). *International vocabulary of metrology — Basic and general concepts and associated terms (VIM)*. Web Page. Accessed: 2014-03-11. URL: http://www.bipm.org/utils/common/documents/jcgm/JCGM_200_2008.pdf.

KUKA (2006). *KUKA.Ethernet KRL XML 1.1 For KUKA System Software (KSS) 5.x*. Report.

Leitão, Paulo (2009). "Agent-based distributed manufacturing control: A state-of-the-art survey". In: *Engineering Applications of Artificial Intelligence* 22.7, pp. 979–991. ISSN: 0952-1976.

Lind, Morten (2012). "Open Real-Time Control and Emulation of Robots and Production Systems". Thesis.

Lind, Morten and Johannes Schrimpf (2012). "PyMoCo-Python-Based Robot Motion Control". In:

Lind, Morten, Johannes Schrimpf, and Thomas Ulleberg (2010). "Open Real-Time Robot Controller Framework". In: *Proceedings of the 3rd CIRP Conference on Assembly Technology and Systems - Responsive, customer demand driven, adaptive assembly*, pp. 13–18.

Maeda, Y. et al. (2003). "An easily reconfigurable robotic assembly system". In: *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*. Vol. 2, 2586–2591 vol.2. ISBN: 1050-4729.

Makris, S. et al. (2012). "Cooperating Robots for Reconfigurable Assembly Operations: Review and Challenges". In: *Procedia CIRP* 3, pp. 346–351. ISSN: 2212-8271.

Marik, Vladimir and Duncan McFarlane (2005). "Industrial adoption of agent-based technologies". In: *Intelligent Systems, IEEE* 20.1, pp. 27–35. ISSN: 1541-1672.

Matplotlib (2014). *Matplotlib: Python plotting*. Web Page. Accessed: 2014-05-04. URL: http://matplotlib.org/.

Monden, Y. (2011). *Toyota Production System: An Integrated Approach to Just-In-Time, 4th Edition*. A Productivity Press book. Taylor & Francis. ISBN: 9781439820971.

NumPy (2014). *NumPy project website*. Web Page. Accessed: 2014-05-04. URL: http://www.numpy.org/.

Ohno, T. (1988). *Toyota Production System: Beyond Large-Scale Production*. Productivity Press. Taylor & Francis. ISBN: 9780915299140.

OpenCV (2013). *Camera Calibration and 3D Reconstruction*. Web Page. Accessed: 2013-08-27. URL: http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html.

— (2014). *OpenCV project website*. Web Page. Accessed: 2014-05-04. URL: http://opencv.org/.

OROCOS (2014). *Open Robot Control Software Project*. Web Page. Accessed: 2014-05-24. URL: http://www.orocos.org/.

Pandas (2014). *pandas: Python data analysis library*. Web Page. Accessed: 2014-05-04. URL: http://pandas.pydata.org/.

Park, Frank C and Bryan J Martin (1994). "Robot sensor calibration: solving AX=XB on the Euclidean group". In: *Robotics and Automation, IEEE Transactions on* 10.5, pp. 717–721. ISSN: 1042-296X.

Redford, AH (1991). "Materials handling for general purpose assembly". In: *The Iinternational Journal Of Production Research* 29.2, pp. 229–246. ISSN: 0020-7543.

RoboticsVO (2013). *A Roadmap for U.S. Robotics. From Internet to Robotics. 2013 Edition*. Report.

Rooker, Martijn et al. (2013). "Flexible Grasping of Electronic Consumer Goods". In: *Robotics in Smart Manufacturing*. Ed. by Pedro Neto and AntónioPaulo Moreira. Vol. 371. Communications in Computer and Information Science. Springer Berlin Heidelberg. Chap. 15, pp. 158–169. ISBN: 978-3-642-39222-1.

Rosati, G. et al. (2011). "Convenience analysis and validation of a fully flexible assembly system". In: *Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on*, pp. 1–8. ISBN: 1946-0740.

Roulet-Dubonnet, O., M. Lind, and A. Skavhaug (2013). “IceHMS, a middleware for distributed control of manufacturing systems”. In: vol. 8062 LNAI, pp. 95–105.

Roulet-Dubonnet, O. and P. Ystgaard (2011). “An application of the holonic manufacturing system to a flexible assembly cell”. In: vol. 6867 LNAI, pp. 29–38.

Schmidt, Jochen, Florian Vogt, and Heinrich Niemann (2003). “Robust hand–eye calibration of an endoscopic surgery robot using dual quaternions”. In: *Pattern Recognition*. Springer, pp. 548–556. ISBN: 3540408614.

Schrimpf, Johannes (2013). “Sensor-based Real-timeControl of Industrial Robots”. PhD thesis. Norwegian University of Science and Technology, Department of Engineering Cybernetics.

SciPy (2014). *SciPy library*. Web Page. Accessed: 2014-05-04. URL: http://www.scipy.org/scipylib/index.html.

Scorpion (2014). *Scorpion Vision Software*. Web Page. Accessed: 2014-05-09. URL: http://www.scorpionvision.com/.

Semeniuta, Oleksandr (2014). *FlexVi*. Web Page. Accessed: 2014-05-22. URL: https://github.com/semeniuta/FlexVi.

Spong, Mark W, Seth Hutchinson, and Mathukumalli Vidyasagar (2006). *Robot modeling and control*. John Wiley & Sons New York. ISBN: 0471649902.

Stover, Christopher and Eric W. Weisstein (2014). *Euclidean Space*. MathWorld–A Wolfram Web Resource. Web Page. Accessed: 2014-05-13. URL: http://www.scorpionvision.com/.

Stramigioli, Stefano and Herman Bruyninckx (2001). “Geometry and screw theory for robotics”. In: *Tutorial during ICRA* 2001. URL: http://profs.sci.univr.it/~fiorini/corsi/robotica/icra2001notes.pdf.

Strobl, Klaus H and Gerd Hirzinger (2006). “Optimal hand-eye calibration”. In: *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, pp. 4647–4653. ISBN: 1424402581.

Sturm, P. F. and S. J. Maybank (1999). “On plane-based camera calibration: A general algorithm, singularities, applications”. In: *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on*. Vol. 1, 437 Vol. 1. ISBN: 1063-6919.

Tsai, Roger Y and Reimar K Lenz (1989). “A new technique for fully autonomous and efficient 3D robotics hand/eye calibration”. In: *Robotics and Automation, IEEE Transactions on* 5.3, pp. 345–358. ISSN: 1042-296X.

Upton, David (1994). “The management of manufacturing flexibility”. In: *California management review* 36.2, pp. 72–89. ISSN: 0008-1256.

Van Brussel, Hendrik (1994). “Holonic manufacturing systems the vision matching the problem”. In: *at First European Conference on Holonic Manufacturing Systems, Hannover, Germany*. Citeseer.

Van Brussel, Hendrik et al. (1998). “Reference architecture for holonic manufacturing systems: PROSA”. In: *Computers in industry* 37.3, pp. 255–274. ISSN: 0166-3615.

Van de Vrande, Vareska et al. (2009). “Open innovation in SMEs: Trends, motives and management challenges”. In: *Technovation* 29.6, pp. 423–437. ISSN: 0166-4972.

Wadhwa, Rhythm Suren (2012). “Flexibility in manufacturing automation: A living lab case study of Norwegian metalcasting SMEs”. In: *Journal of Manufacturing Systems* 31.4. ISSN: 0278-6125.

Wiendahl, H. P. et al. (2007). “Changeable Manufacturing - Classification, Design and Operation”. In: *CIRP Annals - Manufacturing Technology* 56.2, pp. 783–809. ISSN: 0007-8506.

WinPython (2014). *WinPython project website*. Web Page. Accessed: 2014-05-04. URL: http://winpython.sourceforge.net/.

Womack, J.P. and D.T. Jones (2010). *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*. Free Press. ISBN: 9781439135952.

Zhang, Zhengyou (2000). “A flexible new technique for camera calibration”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22.11, pp. 1330–1334. ISSN: 0162-8828.

Appendices

A Intrinsic parameters distributions

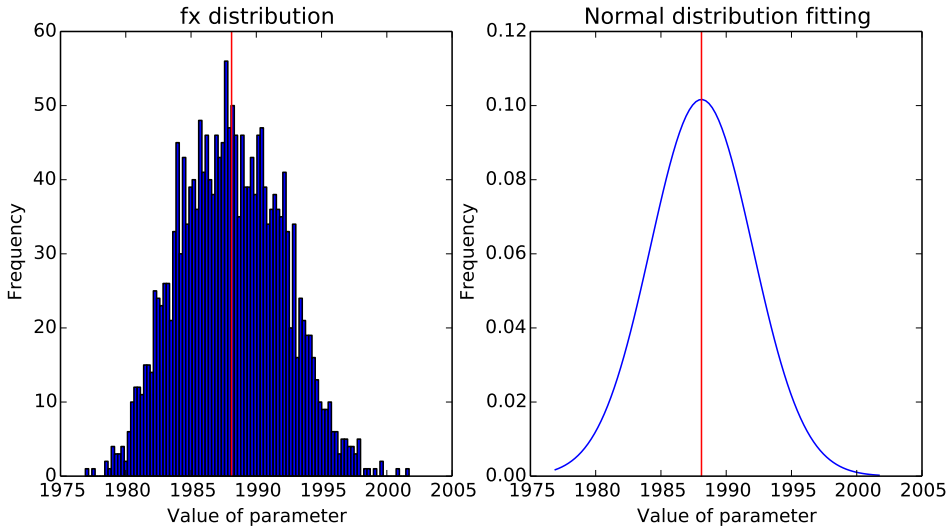


Figure 29: f_x distribution

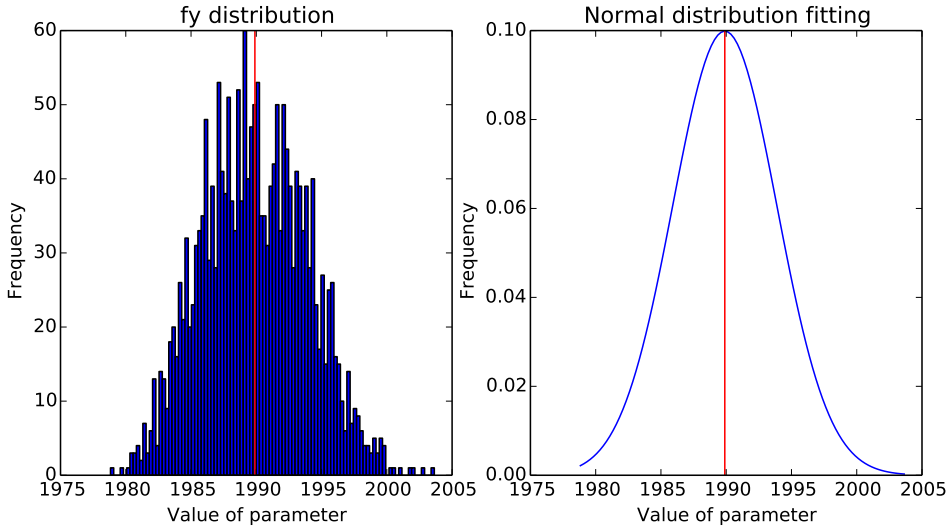


Figure 30: f_y distribution

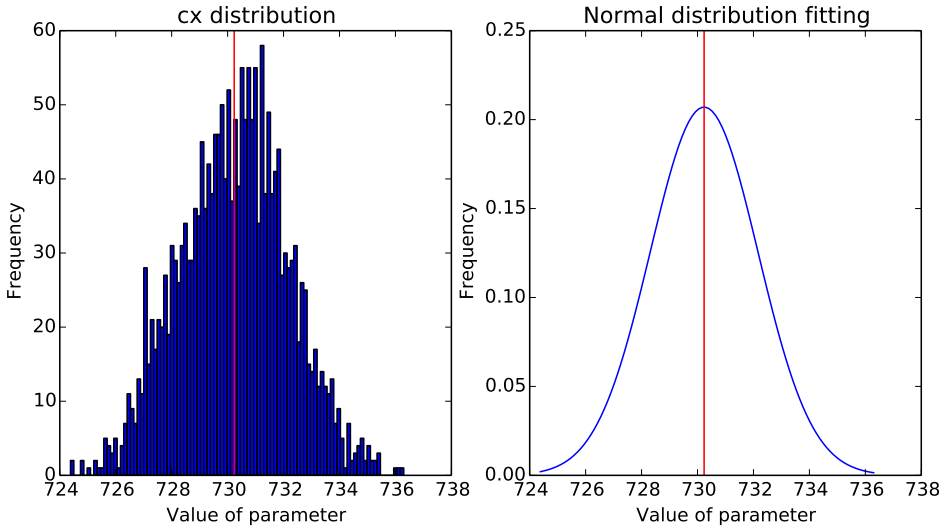


Figure 31: c_x distribution

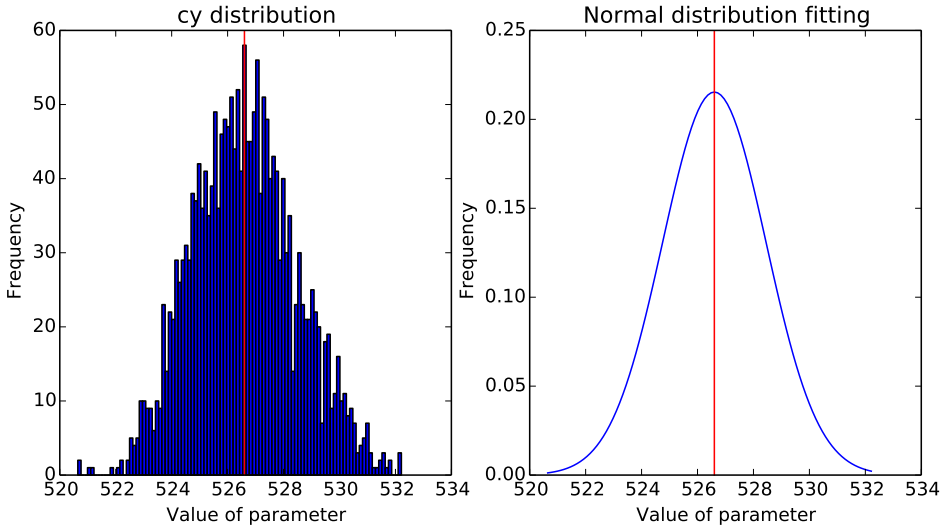


Figure 32: c_y distribution

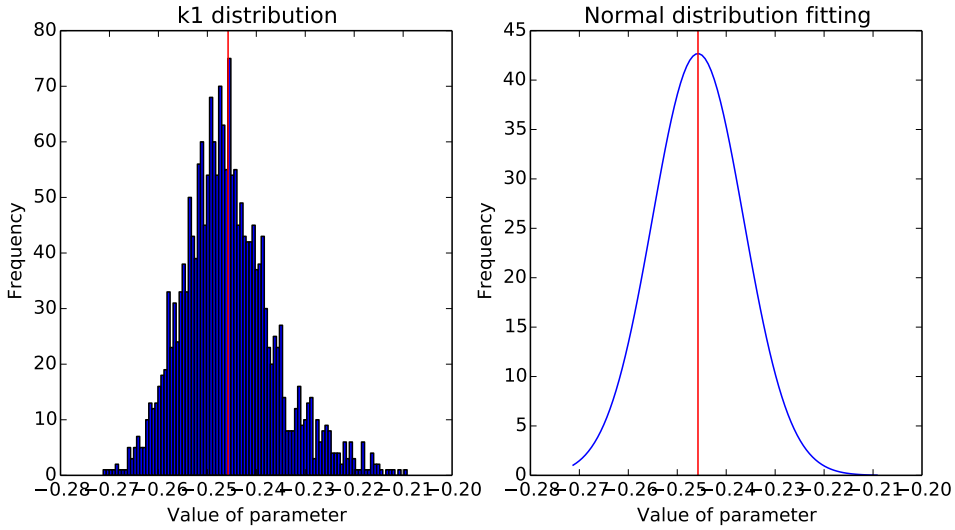


Figure 33: k_1 distribution

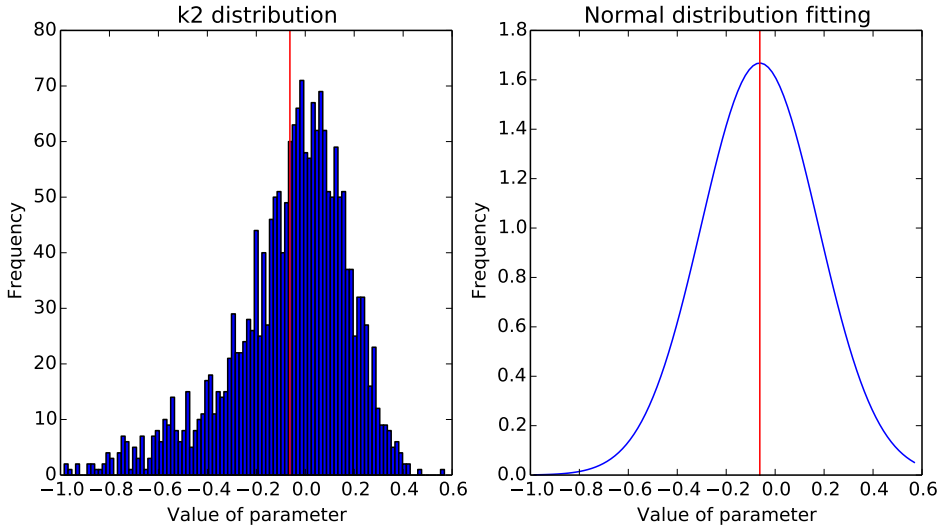


Figure 34: k_2 distribution

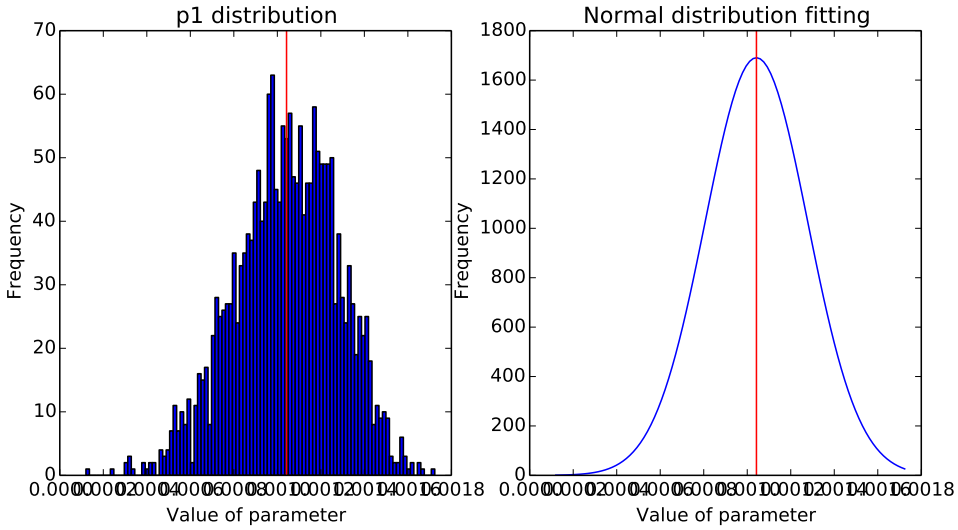


Figure 35: p_1 distribution

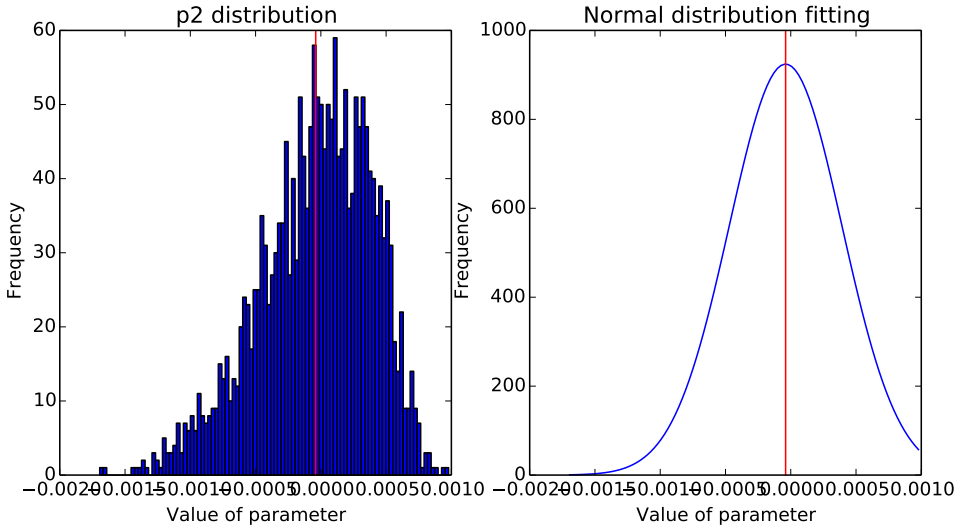


Figure 36: p_2 distribution

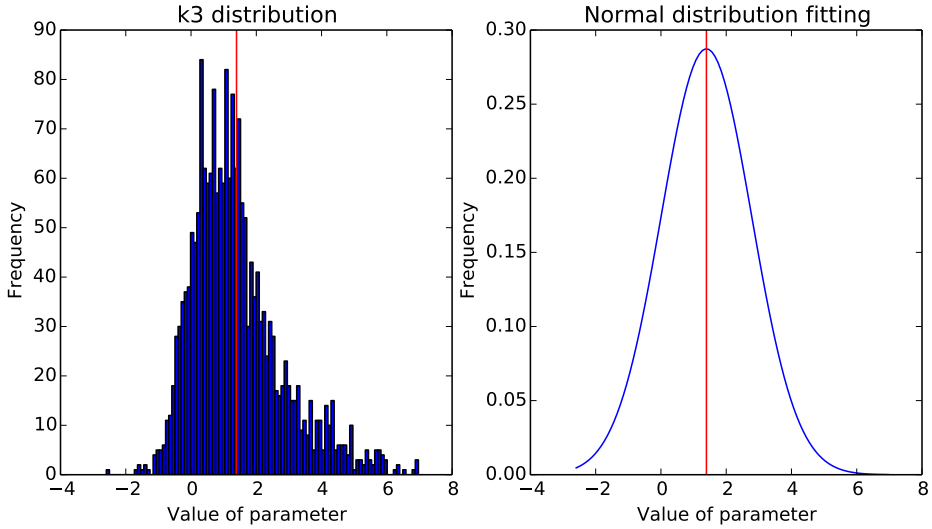


Figure 37: k_3 distribution

B Results of the runs of repeatability-oriented stereo calibration

Table 5: Results of N runs of repeatability-oriented stereo calibration

Run	f_{11}	f_{12}	f_{13}
1	2.12E-08	2.02E-06	-1.24E-03
2	2.12E-08	2.02E-06	-1.24E-03
3	2.13E-08	2.02E-06	-1.24E-03
4	2.17E-08	2.00E-06	-1.23E-03
5	2.11E-08	2.03E-06	-1.24E-03
6	2.15E-08	2.01E-06	-1.24E-03
7	2.16E-08	2.05E-06	-1.26E-03
8	2.11E-08	2.03E-06	-1.24E-03
9	2.16E-08	2.05E-06	-1.26E-03
10	2.15E-08	2.01E-06	-1.23E-03
Run	f_{21}	f_{22}	f_{23}
1	1.81E-06	8.15E-08	-2.81E-02
2	1.80E-06	7.99E-08	-2.81E-02
3	1.81E-06	8.18E-08	-2.81E-02
4	1.84E-06	9.18E-08	-2.82E-02
5	1.80E-06	7.73E-08	-2.81E-02
6	1.84E-06	8.46E-08	-2.83E-02
7	1.88E-06	5.83E-08	-2.95E-02
8	1.80E-06	7.65E-08	-2.81E-02
9	1.88E-06	5.77E-08	-2.95E-02
10	1.84E-06	8.54E-08	-2.83E-02
Run	f_{31}	f_{32}	f_{33}
1	-8.00E-04	2.54E-02	1.00E+00
2	-7.98E-04	2.53E-02	1.00E+00
3	-8.00E-04	2.54E-02	1.00E+00
4	-8.12E-04	2.54E-02	1.00E+00
5	-7.96E-04	2.54E-02	1.00E+00
6	-8.12E-04	2.55E-02	1.00E+00
7	-8.29E-04	2.67E-02	1.00E+00
8	-7.95E-04	2.54E-02	1.00E+00
9	-8.29E-04	2.67E-02	1.00E+00
10	-8.13E-04	2.55E-02	1.00E+00