6-MIG Project: Multi-User Interaction System for CAVE-type VR Environments

Prof. Dr.-Ing. Rainer Herpers
Dipl.-Ing. Timur Saitov, M.Sc.

Additional co-authors in alphabetic order:
A. Bochem, E. Dayangac, T. Hofhammer, K. Kent,
P. Samarin, D. Scherfgen, J. Sommer

Publisher: Dean Prof. Dr. Wolfgang Heiden

University of Applied Sciences Bonn-Rhein-Sieg,
Department of Computer Science

Sankt Augustin, Germany

August 2013

Technical Report 03-2013

Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

ISSN 1869-5272

Technical Report 03-2013

6-MIG Project:

Multi-User Interaction System for CAVE-type VR Environments

Prof. Dr.-Ing. Rainer Herpers
Dipl.-Ing. Timur Saitov, M.Sc.

August 2013

Reporting period: 01.07.2008 – 30.06.2012

This project was funded by the Federal Ministry of Education and Research (BMBF), within FHprofUnt program, grant No. 1759X08, under the title:
"6-DoF multiuserfähiges Interaktionsgerät für 3D-Projketionsumgebungen"

Project partners:

- University of New Brunswick
  Fredericton, NB, E3B 5A3, Canada
- Fachhochschule Koblenz, Standort Remagen, RheinAhrCampus
  53424 Remagen
- Institut für Arbeitsschutz (IFA) der Deutschen Gesetzlichen Unfallversicherung (DGUV)
  53757 Sankt Augustin
- Matrix Vision GmbH
  7150 Oppenweiler
- Sør-Trøndelag University College
  N-7004 Trondheim, Norway
- Virtual! Köln
  50674 Köln
- C.E.E. Media GmbH
  53113 Bonn

Disclaimer:

This Technical Report of the 6-MIG project is a summary and collection of various scientific contributions, intermediate reports and student qualification projects [19], [22], [23], [24], [25], [26], [27], [28], [30], [31], [32] with their respective authors or active contributors (in an alphabetical order): A. Bochem, E. Dayangac, R. Herpers, T. Hofhammer, K. Kent, T. Saitov, P. Samarin, D. Scherfgen, J. Sommer

Table of Contents

1 Objectives and Requirements

The objective of this research project is to develop a user-friendly and cost-effective interactive input device that allows intuitive and efficient manipulation of 3D objects (6 DoF) in virtual reality (VR) visualization environments with flat projections walls.

During this project, it was planned to develop an extended version of a laser pointer with multiple laser beams arranged in specific patterns. Using stationary cameras observing projections of these patterns from behind the screens, it is planned to develop an algorithm for reconstruction of the emitter's absolute position and orientation in space. Laser pointer concept is an intuitive way of interaction that would provide user with a familiar, mobile and efficient navigation though a 3D environment.

In order to navigate in a 3D world, it is required to know the absolute position (*x*, *y* and *z* position) and orientation (*roll*, *pitch* and *yaw* angles) of the device, a total of 6 degrees of freedom (DoF).

Ordinary laser-based pointers when captured on a flat surface with a video camera system and then processed, will only provide x and y coordinates effectively reducing available input to 2 DoF only. In order to overcome this problem, an additional set of multiple (invisible) laser pointers should be used in the pointing device. These laser pointers should be arranged in a way that the projection of their rays will form one fixed dot pattern when intersected with the flat surface of projection screens. Images of such a pattern will be captured via a real-time camera-based system and then processed using mathematical re-projection algorithms. This would allow the reconstruction of the full absolute 3D pose (6 DoF) of the input device.

Additionally, multi-user or collaborative work should be supported by the system, would allow several users to interact with a virtual environment at the same time. Possibilities to port processing algorithms into embedded processors or FPGAs will be investigated during this project as well.

2 State of the Art

2.1 Known Designs

There were no additional new designs found that utilizing approach or combination of approaches as originally intended in this project. Nevertheless, after changing the overall system design from "outside-in" to "inside-out", the project starts to cover another area of research in the field of fiducial marker systems. Additional state-of-the art search revealed at least one development project, which is related to our work:

"…a system to track position and orientation of a generic mobile device equipped with a camera using a set of variable size fiducial markers. The system provides six degrees-of-freedom (6-DOF) by tracking fiducial markers through the camera and deriving the position and orientation of the device, thus making possible the implementation of innovative and affordable 3D user interfaces. The system has been integrated in a Cave Automatic Virtual Environment (CAVE) through the use of projectors and polarizers." [1].

Despite being related to this project, there is a difference in illumination technique. In particular, a usage of polarizing filters to separate the markers' image from visualization environment. With current state of the art in polarization technology it is impossible to reach a 100% channels separation. It results in the fiducial marker system interferences with the visualization content, that appear to the user as "ghosting" effects with partially visible patterns. The effect will significantly change with a change in viewing angle or a change in user's head pose. The system we developed based on infrared technology outperforms the visual quality suggested by [1].

Sample references to the different approaches (outside-in, inside-out) can be found in the Appendix A.

2.2 Preliminary Work

As a preliminary work for the project, a first prototype of the laser-emitting device [16] was built[1]. This emitter consists of a set of five laser pointers, working in the infrared domain invisible to human eye, and a microcontroller that is capable of switching on and off each laser individually. Laser pointers were arranged on a metal plate emitting a static pattern, see Figure 2.1.

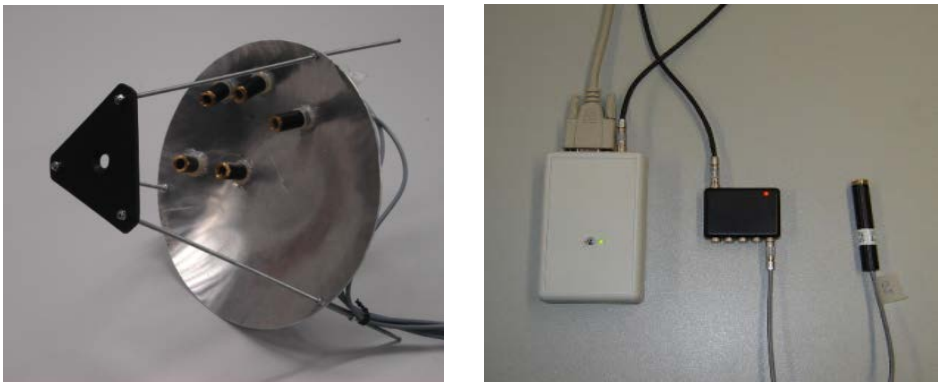


Figure 2.1: An infrared laser emitting device with 5 lasers arranged on a metal plate in order to emit a certain fixed pattern (left). Microcontroller module for five laser emitters with serial-port communication channel (right).

[1] In cooperation with the Institute for Occupational Safety and Health of the German Social Accident Insurance (Institut für Arbeitsschutz (IFA) der Deutschen Gesetzlichen Unfallversicherung (DGUV))

Additionally to the laser-emitting device, a miniature model of the immersive visualization environment "Immersion Square" has been built using the original screen material. The model improves the overall usability of the working environment (Figure 2.2). The small size allows straightforward development of new approaches, effective rapid prototyping and testing.

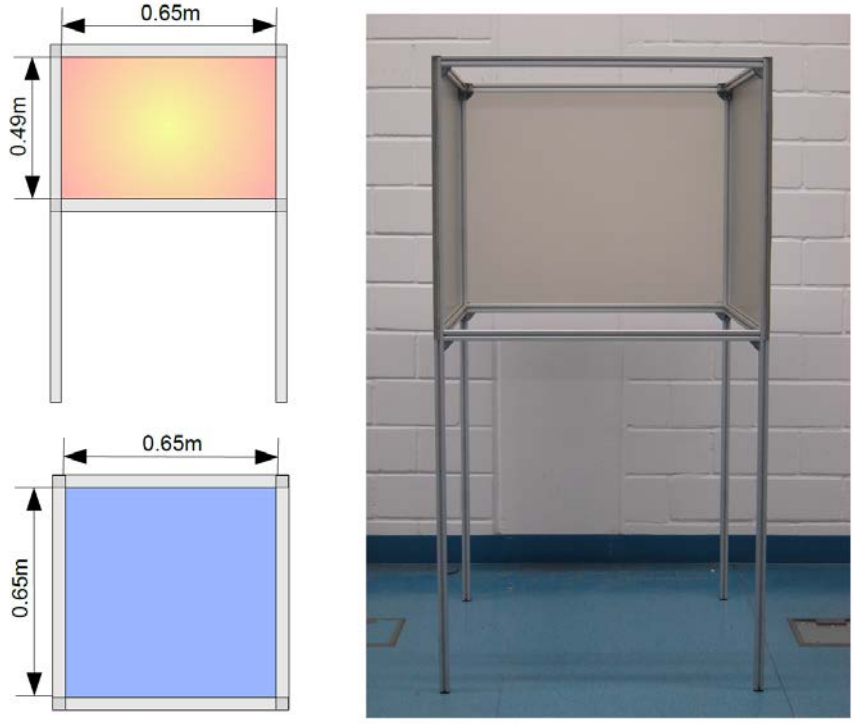


Figure 2.2: A model of the Immersion Square scaled down in size. The screens are made from the original materials allowing exactly same light distribution as in the original, large version.

This model was successfully used in several experiments throughout the entire project period. All new infrared emitters were successfully tested and evaluated using this test environment.

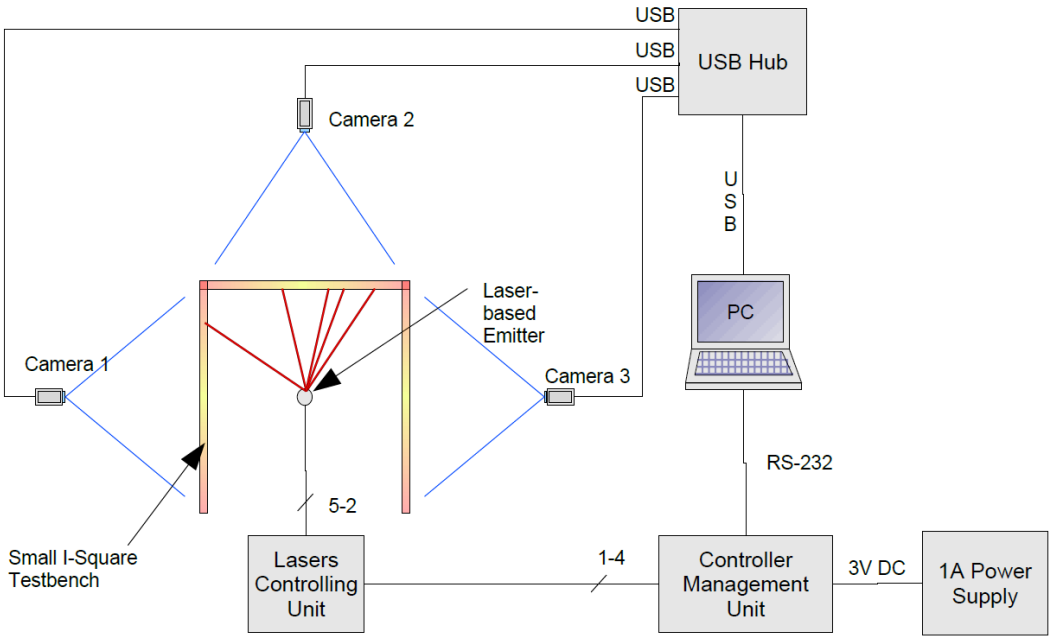


Figure 2.3: First prototype of the system built in the small Immersion Square test bench using the original 5-laser emitters design.

This setup (see Figure 2.3) was limited in several aspects. There were some major problems:

- Incapability to support multiple users simultaneously without time-based switching with its increasing complexity due to synchronization problems.
- Difficulties in physical production of the necessary high-precision dot pattern, or a beam splitter (diffractive optical element, DOE), where the price for such production compromises the entire project idea of a low-cost input device.
- High sensitivity to the dot position deviations inside the pattern, rendering resulting system output inaccurate.
- Very high power consumption levels of the laser system incompatible with a mobile, battery-driven, hand-held, pointing device.

This prototype was used during an initial stage of the project for testing and evaluating purposes of the already existing theoretical approaches. Due to its non-rigid frame, the laser pattern configuration was often influenced by external physical forces leading to severe quality problems in pose calculations. Being an experimental prototype, the overall size, weight and power consumption levels were rated as not acceptable to be a candidate for a mobile hand-held device.

2.3 Evaluation of Available Solutions for a Pattern Emitter

2.3.1 Multiple Laser Emitters

A systematic search for an appropriate pattern emitter began after the first prototype with five separate laser emitters had been built and evaluated (see Figure 2.1, Figure 2.4). It has a number of limitations that led to further investigations for other possible solutions. If used as a hand-held device, there are some of the major limitations such as low precision in laser beams orientation and a limited number of points in the pattern.

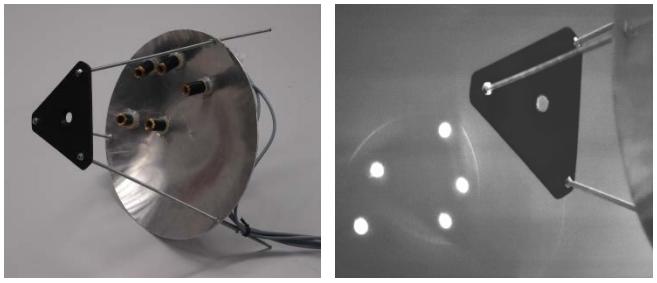


Figure 2.4: Multiple laser emitters arranged in a way that all individual beams pass through one common point.

Low Precision in Laser Beams Orientation: All laser beams had to originate from the same point due to the mathematical approach that was used. This became a nearly impossible task in reality, since each module has an arbitrary displacement of the ray relative to the casing. That makes a rigidly manufactured holder an unpractical solution and leads to high errors in position estimation. Since each laser module has its own casing and it needs to be oriented at a particular angle in 3D space, the overall design becomes large and heavy. This increases the overall special dimensions beyond the acceptable for a hand-held device. High power consumption makes it nearly impossible to produce a lightweight mobile device with a reasonable working time due to large and heavy battery modules. (The prototype design from DGUV (see Figure 2.4) was draining its batteries so fast, that it was converted to use a wall-socket power adapter).

Limited Number of Points in the Pattern: Only 5 laser modules were built in the prototype, making it already much bigger and heavier than acceptable. For the back projection approach, this device is not suitable as well, because it cannot project unique (individually identifiable) markers.

2.3.2 Laser Module with Diffraction Grating or Diffractive Optical Element

Another alternative would be a diffraction grating or diffractive optical element (DOE). Such a design would require only one laser module and a relatively small DOE in front of it. This makes the module reasonably small for a hand-held pointer. Nevertheless, a laser beam split by a DOE loses it strength and each individual point inside projected pattern contains just a small fraction of the original beam's energy (see Figure 2.5). This means that high power consumption and therefore a heavy and large battery might still cause problems. But the major problem is that diffractive optics can only produce a regular, repetitive pattern without significant loss in transmitted light energy. In addition to this, irregular power distribution across the pattern makes points located further from centre appears darker in the camera image (see Figure 2.5, right).

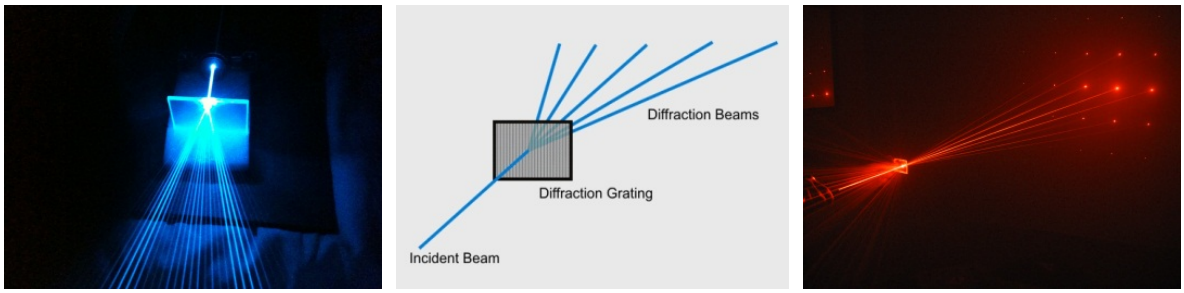


Figure 2.5 Diffraction gratings splitting laser beams.

As discussed below in Section 4.6.6, in order to solve the correspondence problem, none of the 3 points should be collinear. A regular grid pattern destroys this concept entirely. Since this approach cannot project unique (individually identifiable) markers, the problem stays the same for the back-projection approach as well.

In case of a low quantity orders another problem of high production costs occurs. In fact, the production of an infrared version of the DOE is even more expensive than the one for the visible spectrum and it is way beyond the budget of this project, which makes this solution impractical. Not being able to find a reasonable solution for a hand-held laser emitter, the entire project design turned from the original "outside-in" to a new "inside-out" approach (Section 5). This enables us to take into consideration another set of light projection systems.

2.3.3 DLP Image Projectors

An Ideal solution for a light source to be used for the "inside-out" approach would be a sufficiently powerful DLP projector with an infrared light source. There are only several such systems available on the market and so far only in research laboratories. Most advanced and highly suitable for the project are two systems: "DLP LightCommander" from Logic PD, Inc. (see Figure 2.6) and "CEL5500S" from Digital Light Innovations (see Figure 2.7).

Figure 2.6: "DLP LightCommander" from Logic PD, Inc.

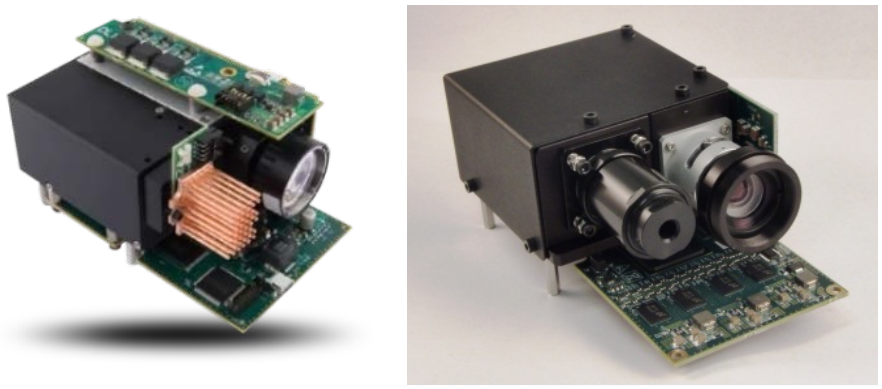


Figure 2.7: DLP based universal light source "CEL5500S" from Digital Light Innovations

Both of these projectors completely satisfy our needs for an "inside-out" approach. In fact, these projectors do have much more features and technical possibilities than what is necessary for our application: they can dynamically change the output image up to 60 times per second. This feature is not a necessity for this project; even a static infrared pattern would satisfy the requirements. The major drawback of such systems is that their prices exceed the project budget. Otherwise, these would be another valuable alternative for the project's light source.

2.3.4 An Array of Refractive Micro-Optical Elements

Being a cutting-edge technology, a LED light sources with refractive micro-optical components are a very promising alternative for the project, see [15]. These elements are able to generate images consisting of points, they can be irregular as well, but the problem is that this technology is not available for general use and it's still in the development stage (see Figure 2.8). An additional problem with this approach is a static (fixed) focal distance, allowing effective use (without blurring) of the device only in an "inside-out" configuration.

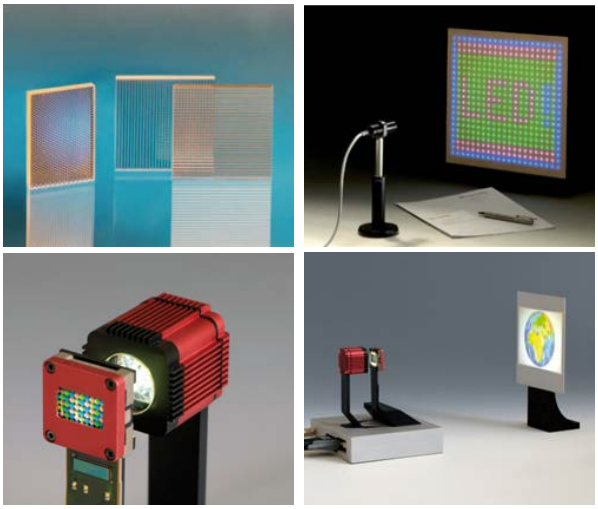


Figure 2.8: LED light sources with refractive micro-optical components (top row). Micro-optics array projector (bottom row).

A micro-optics array projector developed on similar refractive micro-optical components combined with micro-mask is able to produce static images up to VGA resolution (see Figure 2.8). Once equipped with an infrared light source, they would be another real alternative for the project's light source.

The energy loss is still quite a serious issue here, meaning that a large battery would render this device impractical. Nevertheless this solution can be successfully used in a back projection "inside-out" approach. This cutting-edge technology is still quite expensive when produced in small (prototype scale) quantities.

2.3.5 Static Micro-Mask Projector

Another alternative to a LED with micro-mask combined with micro-lens array is a micro-mask combined with a standard (ordinary scale, C-mount) lens. This product is the most suitable one that is available on the market. It satisfies all requirements of the project and does not contain any extra, unnecessary features (unlike the solutions described in Section 2.3.3 and Section 2.3.4), keeping the price for this type of technology within an affordable range. This type of projector is capable of producing a binary image with a high spatial resolution, see Figure 2.9. Since no diffractive elements are used, the light source can be equipped with an infrared light source at little expense.

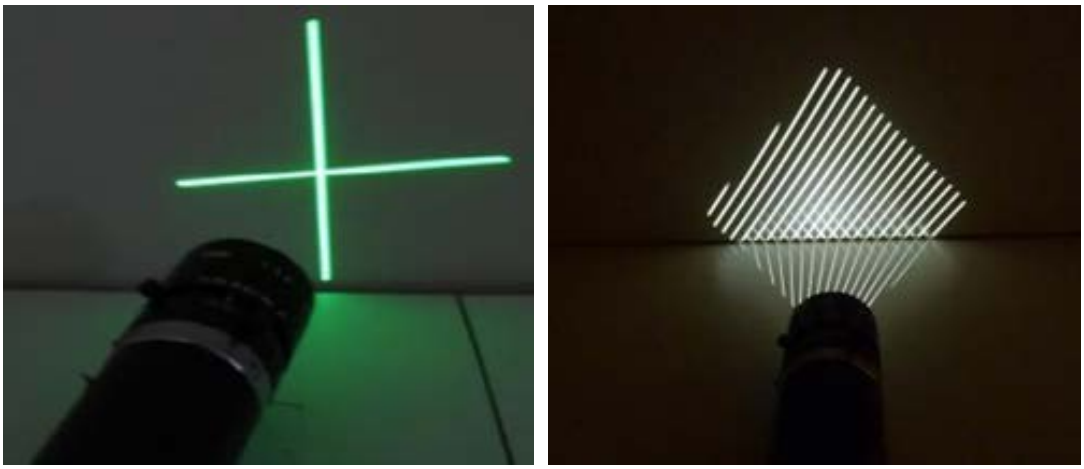


Figure 2.9: Static micro-mask projector "EFFI-Lase" from EFFILUX

This type of projection system was considered as the most inexpensive and promising one for further developments.

3 General Frameworks

In this chapter the results of two different alternative developments of software frameworks are presented. Due to the uncertainty which platform would be used in the future, both Windows-based (GigE Vision) and Linux-based (FireWire) frameworks were developed in parallel. This created future flexibility and allowed different developers to concentrate on the solution of the problem using familiar tools. Additionally, a selection of the lenses and the infra-red filters for the cameras are discussed in this section as well.

3.1 Development of a Multi-Camera Framework for GigE-Vision Protocol (Windows-based)

A highly modular GigE-Vision software framework has been developed. The basis of the framework is the idea of hierarchical management with a specific set of basic controlling and performing classes. This kind of structure improves modularity of an application and reduces the need of inter-module communications utilizing the system of decision-making within each of the modules. An organization like this also provides portability of code, for example, as in the case of user graphical interface module, where the main computational part becomes independent of the user interface.

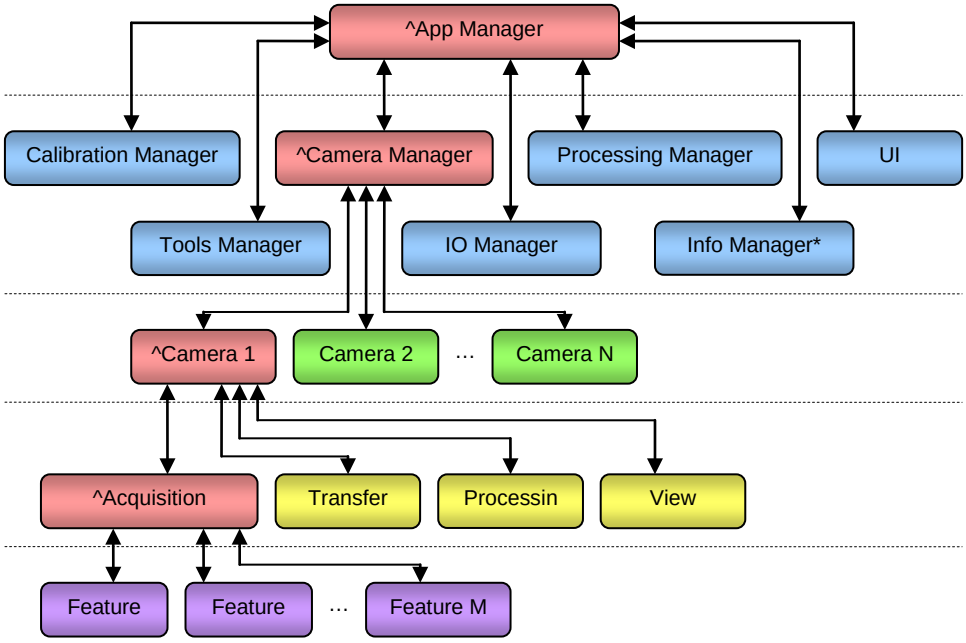


Figure 3.1: General structure diagram of the hierarchical system's management. Only one major branch is fully expanded (shown in red, with ^ symbol).

Thus, each module performs a specified task within itself, referring to the parent in the hierarchy module only in the case of events which require a decision at a more global level. A generalized structure diagram of the hierarchical approach is shown in Figure 3.1. At run-time there could be any number of cameras, any number of calibration samples, unknown number of available camera features, unknown number of tracking objects, etc. Therefore, all modules of the framework are designed to support dynamic scalability.

The root module of the system is App Manager. It has direct control over several others main control modules. At the moment there are four major controllers available: Calibration Manager, Camera Manager, Processing Manager, and UI Manager. More detailed information on the developed framework can be found in [22].

3.2 Development of a Multi-camera Framework for FireWire cameras (Linux-based)

During the project a software framework for the use of FireWire cameras under Linux has been developed. It is based on the previous development of an analogous framework on the library (libdc1394 version 1.x). This library enables FireWire cameras to be used under the Linux platform.

In January 2008 the version 2.0.0 of the library with a completely new API and a data structures has been published. In order to use the previously developed software, the acquisition module of the framework needs to be completely redesigned.

The framework developed in this project based on C++ and uses different methods of the object-oriented language. The aforementioned library is a pure C library and therefore, does not fit to the concept of the framework. Due to this reason, a wrapper has been developed which encapsulates the methods of the libdc1394 library into a C++ object. For reasons of simplicity and maintenance the wrapper is limited to the most necessary ones, leaving space for future extensions.

3.3 Lens Selection

Considering the particular application of the system as an interactive device for immersive visualization environments, in particular the Immersion Square (IS) [2], the corresponding lenses for the observing camera system have to be chosen. In order to cover the screens of the IS using one camera per screen, the horizontal angle of view has to be roughly around 60 degrees. Our cameras sponsored by Matrix Vision have sensor size of 1/2 inch (6.4mm width, 4.8mm height). The closest available focal length is 6mm and it will result in viewing angle of 56 degrees. Therefore, the minimum requirements for the lens are as follows: mount type: C-mount; focal length of maximum 6 mm; sensor size at least 1/2"; high transfer characteristics in near-infrared spectral range. Schneider-Kreuznach has a special line of lenses for infrared applications: IR Cinegon 1.8/4.8 with its perfect transfer characteristics at near-infrared spectrum and wide angle with low distortions. Nevertheless, this kind of high-precision optics is not necessary and seemed to be too expensive. The standard lenses like H612A from Pentax or DF6HA-1B from Fujinon (Figure 3.2) are considered as suitable for this project.

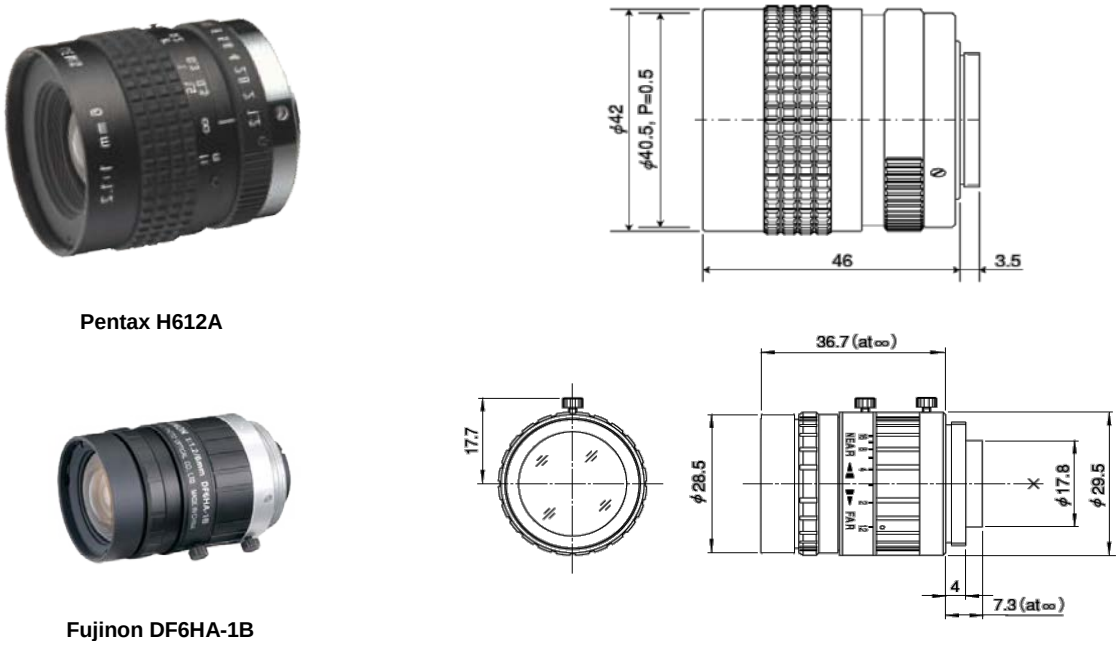


Figure 3.2: Pentax H612A and Fujinon DF6HA-1B; Blueprints units: mm.

These two lenses have almost identical optical characteristics, making both of them equally suitable. In practical comparison test Fujinon DF6HA-1B showed little less radial distortions in the image periphery. Besides that it is smaller in size and less expensive than the Pentax one. Therefore, the Fujinon lens has been selected for the current project. For more information on lens selection please refer to [22].

3.4 Near-Infrared Pass-Filter Selection

One of the project's requirements for IR illumination sources is that they have to be beyond visible spectrum, since one of the system's application areas is immersive visualization. Fulfilling that requirement will help to avoid visual interferences with projectors and prevent distractions of the system's users.

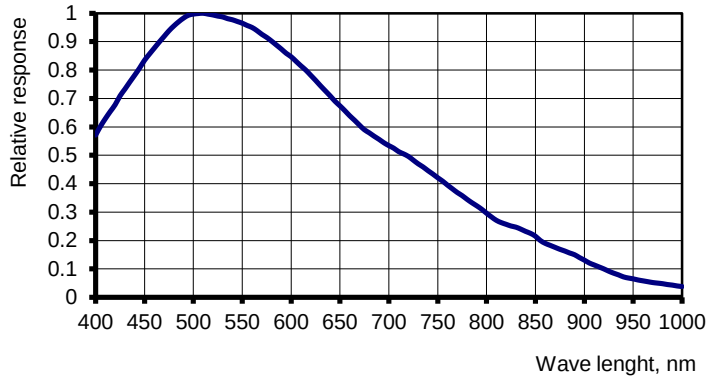


Figure 3.3: Camera's relative spectral response.

Most of the near-infrared LEDs available have a peak emission at wave length of 850 nm, 880 nm, 940 nm, 950 nm. According to the camera spectral response (Figure 3.3) it is at least three times more sensitive at 850 nm than at 950 nm. In most cases the same applies to the IR LEDs: 850 nm models are more powerful than 950 nm devices. Therefore, the decision to use base frequency of 850 nm for this development has been made.

Due to the broad variety of external diameters of the lenses, every type of a lens will require a special IR filter. The idea to install an IR filter behind the lens seemed very attractive, thus making it independent of the lens installed. This kind of configuration will ease the lens exchange procedure, but poses an additional requirement for the IR filter selection. Therefore, the minimal requirement for a near-IR filter is to have an internal mount (C-mount). There are three such internal filters available from Schneider-Kreuznach Company's "B+W" division. Their spectral transmission characteristics are shown in Figure 3.4.

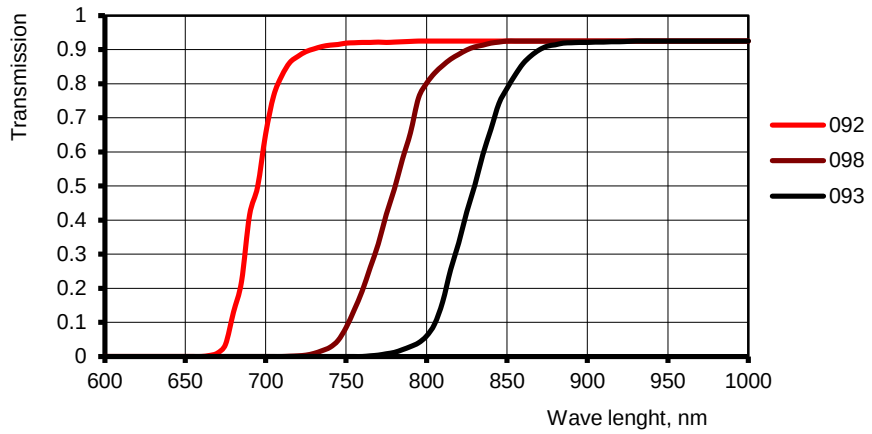


Figure 3.4: Transmission characteristics of Schneider-Kreuznach B+W Near-IR filter 092, 098 and 093 for internal C-Mount.

Since we are targeting LEDs with a wave length of 850 nm, in order to find out which filter is most suitable for such an application and how well it will perform, a combined graph of filters transmission characteristics and camera relative response has been calculated (Figure 3.5).

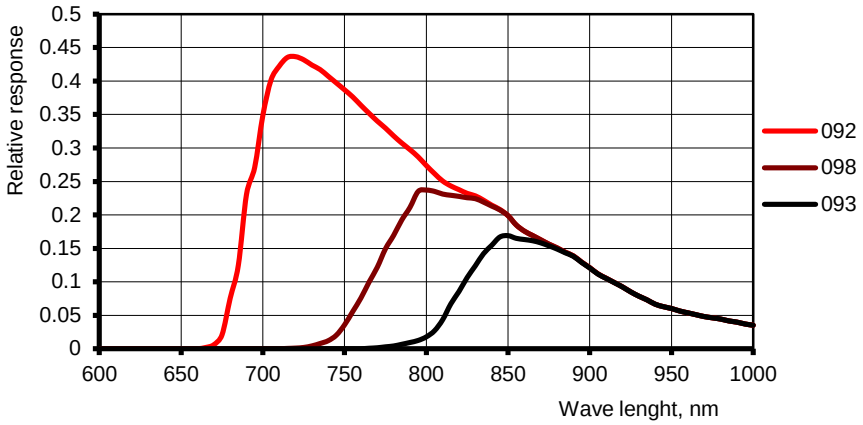


Figure 3.5: Relative responses the camera equipped with Schneider-Kreuznach B+W filter type 092, 098 and 093.

The peak of the relative response of the camera equipped with the Schneider-Kreuznach B+W 093 C-mount filter shows a best match for 850 nm. Therefore this filter type has been selected for the current project. For more information on filter selection please refer to [22].

4 Outside-In Approach

In this chapter the results of several sub-projects related to the outside-in approach are presented. This includes a Camera with an Embedded CPU [24], an external FPGA using analog video signal [23] and an external FPGA using digital CCD camera [26], a new temporary solution for the pattern emitter, a 3D system's emulator [25], a method for dots pattern optimization and a solution for the outside-in approach [19]. A method for combining the data from an inertial measurement unit with the data from our outside-in approach is presented here as well [28].

4.1 BLOB Detection on-board of the Camera with an Embedded CPU

In this subproject a Connected Component Labelling (CCL) approach, using an intelligent camera with an embedded CPU, has been implemented. This was done to save network bandwidth, as only the coordinates of the found connected components are sent through the network. Another benefit of running CCL directly on the camera is the saving of processing power on the host PC, although this is not as critical as saving the network bandwidth. The focus of this subproject was mostly directed towards optimization, as an application running on an embedded system need to be optimized as much as possible due to limited resources, like processing power and memory. The algorithms developed were tested in comparison to a standard PC, in order to evaluate the advantage of running the algorithms directly on an embedded CPU on the camera.

Laser ray projections on the visualization surface create a distinctive pattern of points. These point's centre positions relative to the screen need to be found. An algorithm for a real-time point detection and calculation of centre coordinates was developed for the intelligent camera mvBlueCOUGAR G-p120a from Matrix Vision GmbH with built-in CPU.

Figure 4.1 shows an example of a pattern build with eight LEDs as a test sample and the output of the algorithm (positions of the centre points). There was a series of experiments conducted in order to compare the performance of the camera-based solution to a reference solution that runs on a standard PC (see Figure 4.2)

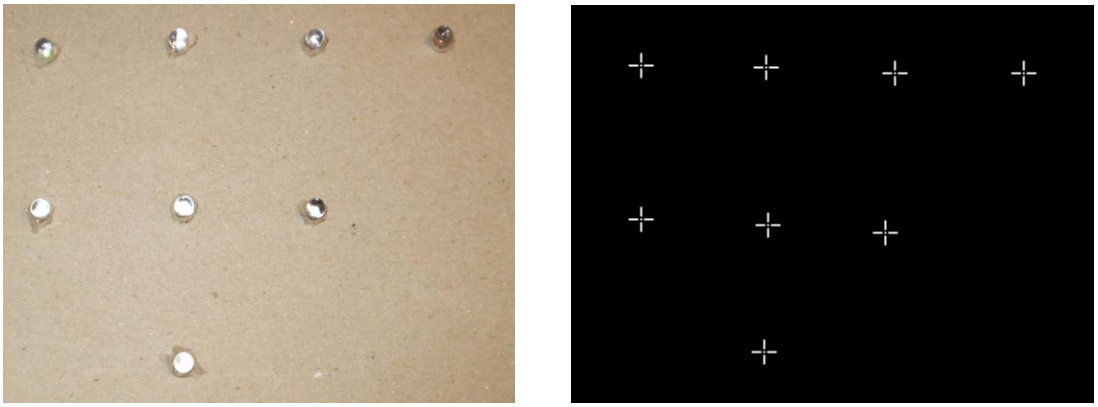


Figure 4.1: Test pattern of 8 infrared LEDs (left) and the coordinates of detected points, taken from another camera position (right).

The results show that with the use of a single camera, no performance gain can be achieved; despite the fact that no image information has to be transmitted over a network from the camera to the PC. The camera's CPU can access the image locally and therefore, all processing is done on the camera. That reduces transmission from the entire image down to the coordinates of the detected points only. The main advantage of the algorithm is that utilization of the available network bandwidth decreases and processing capacity increases with the use of more cameras or cameras with a higher resolution.

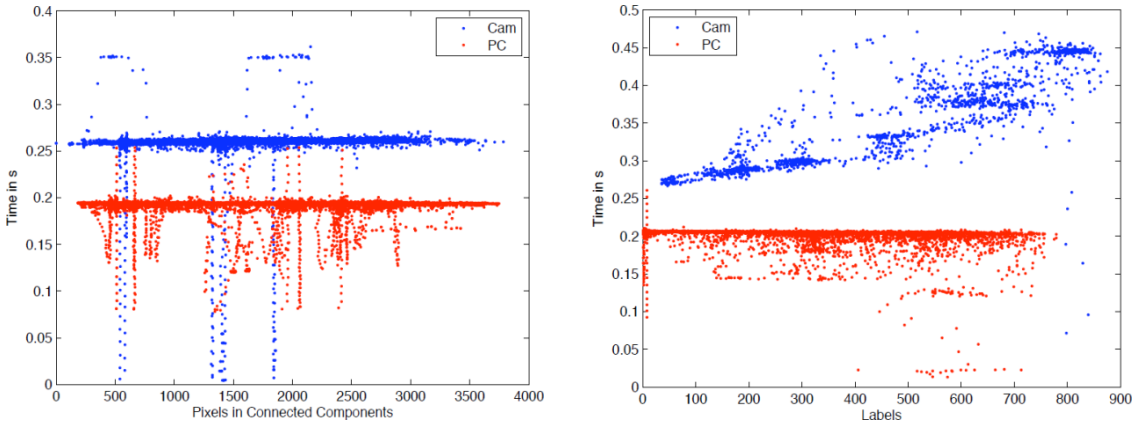


Figure 4.2: Performance charts: On average the intelligent camera with on-board CPU (blue) requires 50% more time than the algorithm running on a standard PC (red).

On average, the camera can run the algorithms close to 26 ms when tracking up to 450 points created by several infrared LEDs. This is done at a speed of about 38 frames per second. The same algorithms running on a PC is close to 20 ms running at a speed of 50 frames per second.

Due to the relatively large delay in the detection of the intelligent camera, this approach wasn't continued within this project. Instead, a shift of the focus towards an FPGA based approach was made (see Section 4.2, 4.8, and 4.9). More detailed information on the embedded approach can be found in [24].

4.2 Acceleration of BLOB Detection using an External FPGA

This subproject presents the implementation and evaluation of point detection algorithms on a Field Programmable Gate Array (FPGA). The task of detecting laser points or Binary Large OBjects (BLOBs) in a continuous video stream was addressed.

4.2.1 System Design

The FPGA hardware design has been developed for the "Altera DE2 development and education" board to detect BLOBs in continuous video streams. This board uses a Cyclone II FPGA that contains 33,215 logic elements (LE). It is equipped with several communication technologies and I/O interfaces. The different modules of the hardware design have been implemented in Verilog. For the module which controls the AD-converter and receives the digitized video signal, a demonstration project from Altera has been reused and modified for the application purpose. The input video stream had to be provided in NTSC format and was converted into RGB format with a resolution of 640x480. The connection and process flow between the different modules are shown in Figure 4.3.

The "Pixel detection" module performs the threshold check with the received data for identifying relevant image data. In the "Adjacency proof" module, the pixels are sorted into data containers. The hardware design uses a structure of registers, allocated in the FPGA, to keep the attribute data for each BLOB. Because of the usage of internal registers, the amount of containers to store BLOB data for the current frame had to be fixed.

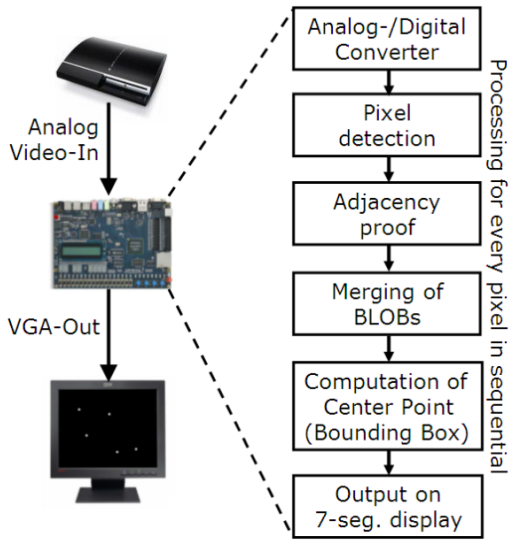


Figure 4.3: System design for a BLOB detection approach on an FPGA.

The developed hardware design can detect up to five BLOBs in a single frame. To increase the system reliability the module "Merging of BLOBs" performs an additional adjacency check for all detected BLOBs after the end of the frame is reached. This late merging eliminates data for a single BLOB that has been labelled different and stores it into one container. The module "Computation of Centre Point" uses the BLOB attributes, stored in the container structure, and computes the centre points. The results are shown on the seven-segment display. In addition, the video stream that is feed into the system on the AV-input is displayed on the VGA output of the board. The transition from an analog-video input to a digital RGB output signal is part of the reused demonstration program from Altera.

4.2.2 Evaluation and Results

For the validation of the BLOB detection approach the design has been tested by simulation and by execution on the target platform. The employed development environment supports functional and timing simulation for the hardware design using Waveform files. The functionality from the pixel detection to the centre point computation has been simulated. The Waveform file contained data for six frame lines with a line size of ten pixels. The simulation of the design showed correct results. It was planned, for the platform evaluation of the BLOB detection approach, to compare the computed results with predetermined ground truth values. This would have allowed an automated precision evaluation. Since the serial communication module could not be realized, the validation of precision has been made for static images only. The results during the first evaluation period showed a systematic error. All computed centre points from the FPGA design showed a constant offset of minus three pixels on the X-axis and plus four pixels on the Y-axis (Figure 4.4). This error showed up for all applied image material.

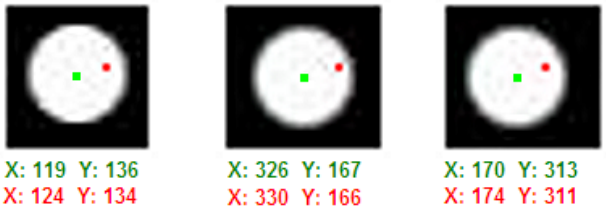


Figure 4.4: Detection of a centre point of a round-shaped BLOB showed a systematic error in the X and the Y axes.

The green/top coordinate values belong to the green dot in the BLOB, while the red/bottom coordinates describe the position of the red dot. The green dots are the so called "ground truth" values and are expected as the correct result. The red results are the outcome of the BLOB detection system that shows

the offset error. For the estimation of the ground-truth values a bounding box approach on a standard PC architecture has been applied. The correctness of those results has been validated by hand.

For providing the image material on the AV-input a standard DVD player has been allied for evaluation purposes. This DVD player provided the video signal in NTSC format, which means the input image, was scaled down. The AD-converter of the FPGA design transformed the video stream into RGB format with 640x480 pixels. This scaling back and forth was one part of the reason for the offset error. All images had a different resolution between 640x480 and 800x600. This was the main problem for the offset in the centre point results. By using input images in the final RGB resolution of 640x480, the offset error could be reduced to one pixel. That was the best precision that could have been reached, applying the two scaling processes of the image material.

For the evaluation of the system's performance, the number of processed frames per second has been counted. The maximum performance of the BLOB detection system was restricted by the processing speed of the analog-digital-converter that pre-processes the video stream from the AV input interface. In this design the observed performance reached 64 frames per second. The hardware design required approximately 25 % of the FGPA resources with the functionality of processing a single video stream. The modular design of this approach allows the configuration to process three video streams in parallel, which requires 66 % of the FPGA resources. More detailed information about this subproject can be found in [23].

4.3 Acceleration of BLOB Detection using a CCD Camera and an External FPGA

As the FPGA development continued, a considerable improvement of previous implementation [23] in processing performance and precision has been achieved. Different algorithms for detecting points more precisely have been implemented. In addition, the set of input devices for acquiring image data has been extended by a digital CCD camera.

The development of a BLOB detection system was conducted on an Altera DE2 development and education board with a Cyclone II FPGA. It detects binary spatially extended objects in image sequences and computes their centre points. Two different sources have been applied to provide image sequences for the processing. First, an analog composite video input, which can be attached to any compatible video device. Second, a live stream, generated by a five-megapixel CCD camera attached to the DE2 board. The results are transmitted via serial interface of the DE2 board to a PC for evaluation and further processing. Major limitations in this approach are the number of points that can be detected simultaneously and the serial interface to the PC. The application developed during the project is able to detect up to five points simultaneously.

Figure 4.5 shows the schematic design of the FPGA-based platform. The image sequences can be acquired on two different devices. The analog video input allows connection to any device with a video output, like a DVD player. The analog video input is used for evaluation with known and annotated image material. The CCD camera has been applied for performance evaluation purposes with real-time data, since the camera allows running the system with higher frame rates. Each input device provides image that requires pre-processing. The BLOB detection has been designed to work with image sequences in the RGB format.

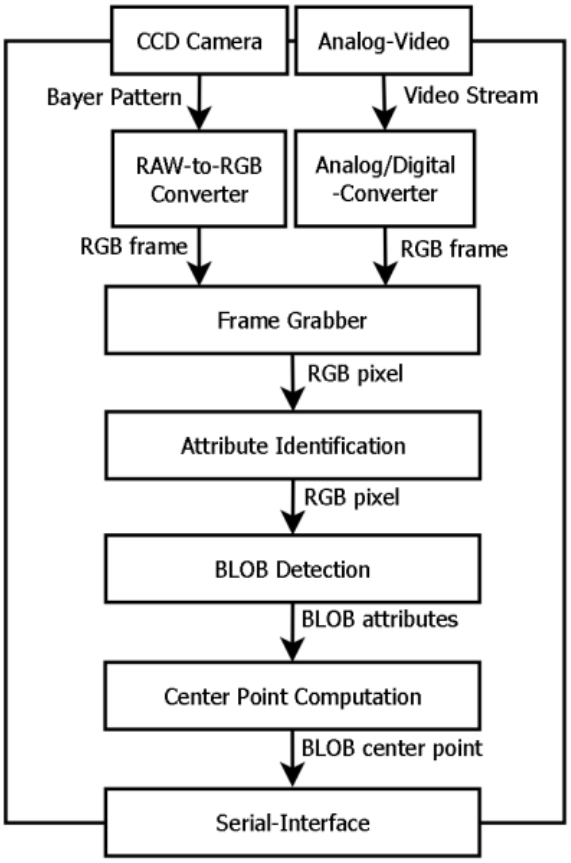


Figure 4.5: Schematic design of the FPGA-based detection system architecture.

The input image sequence from the analog video input is in YCbCr format with a resolution of 640x480 pixels. The input image sequence from the CCD camera is in RGB format. According to the Bayer pattern of the sensor, the frame rows need to be merged. A frame from a sensor with a resolution of 1280x960 results in an image of 640x480 pixels. In the system, every frame is processed only once.

4.3.1 Serial Interface

To observe the system process and evaluate its results, a communication module using the serial interface of the target FPGA board has been developed. The serial interface allows the minimum design effort with respect to protocol overhead and resource allocation on the FPGA. The serial module reads the information about the BLOB result from the FIFO and transmits it to the RS232 controller. The result has to be split into four one byte blocks to be processed by the RS232 controller. The serial interface module operates independently from the other system modules and sends results as long as the FIFO with BLOB results is not empty.

4.3.2 CCD Camera

The performance evaluation of the first BLOB detection approach developed in previous work [23] was restricted to the performance of the Analog-/Digital-Converter on the DE2 board. The video input signal could not be converted as fast as the BLOB detection module would have been able to process it. In the current approach the CCD camera D5M (Figure 4.6) from Terasic Technologies has been attached to the DE2 development board in order to provide a faster image acquisition.

The system design uses the input data from the camera for two different processing tasks. Based on the demonstration design the acquired image data is displayed on the VGA output of the DE2 board. To allow

the additional use of the image data in the BLOB detection module the RGB pixel values have been stored into a dual-clocked FIFO. The BLOB detection module reads the FIFO and searches for BLOBs in the image data.

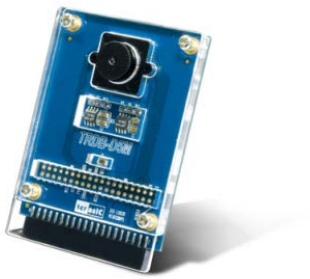


Figure 4.6: Five megapixel CCD camera D5M from Terasic Technologies.

The result of the BLOB detection computation is the computed centre point, which is stored in another dual-clocked FIFO. The modular architecture allows switching between the two designed BLOB detection approaches: "bounding box" and "centre-of-mass".

4.3.3 BLOBs Detection

For the detection of the BLOBs the first problem to be solved is the identification of relevant pixels. A pixel is considered to be relevant if its colour or brightness value exceeds a specified threshold value. This threshold value can be a static parameter or a computed average value, based on the pixel values of previous frames. The adjacency condition is the second important step in BLOB detection. The two most common definitions for adjacency are known as four pixel neighbourhood and eight pixels neighbourhood. Figure 4.7 is showing the two ways of labelling pixels to describe adjacency. On the left image the four pixel neighbourhood is applied and four BLOBs are detected. The detection applies the adjacency check only on the horizontal and vertical axis. On the right image it is demonstrated that the same pixels are labelled as only two BLOBs. For the eight pixel neighbourhood the diagonal axis is taken into account as well [23], [26].

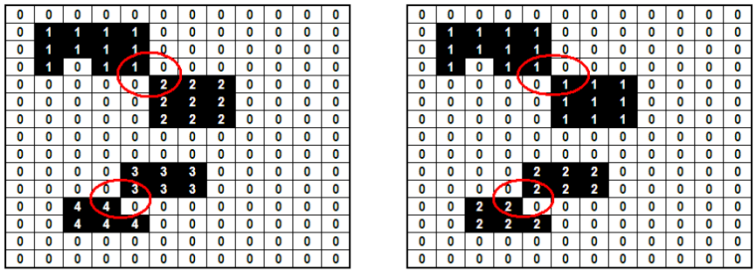


Figure 4.7: Variation in BLOB labelling in case of four (left) and eight (right) pixel neighbourhood.

The objective of the BLOB detection approach is to determine the centre point of the BLOBs in the current frame. With respect to the application area, this subproject describes BLOBs as a set of white and light grey pixels while the background pixels are black. This comes from the setup for the image acquisition where infrared cameras will be applied to track laser dots on a plain projection surface. The application of infrared cameras can help to avoid unwanted features of the immersive visualization environment in the image. The image material acquired by the camera expected to be similar to the synthesised examples in Figure 4.8.

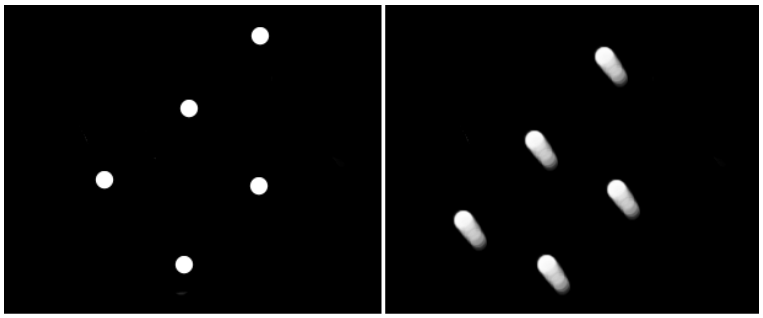


Figure 4.8: Examples of an ideal BLOBs shape (left) and motion blurred one in case of acceleration/deceleration during exposure time (right).

For the computation of the centre point the bounding box approach provides not enough information about the BLOB to reach precise enough results. The centre coordinates are strongly affected by the pixels at the BLOB's edge. This effect becomes even stronger for BLOBs in motion. With reference to the light emitting device for the Immersion Square environment, the angle between the light beams and the projection surface changes the shape of the BLOBs and increases the amount of pixels with less intensity. In addition, the movement of the device by the user will cause some motion blur. These effects will increase the flickering of the pixels at the BLOB's edge and might cause shifts of the computed centre point. In Figure 4.9 some examples are given for inaccurate results based on the bounding box approach.

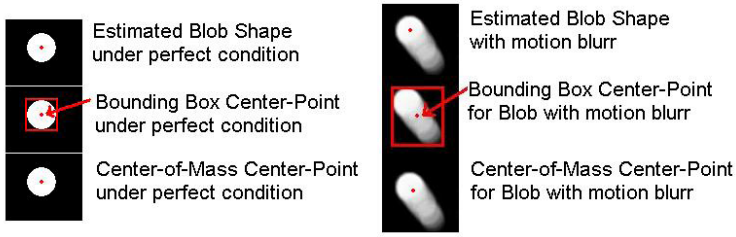


Figure 4.9: Examples of results based on the bounding box approach. Results of BLOBs centre point detection can vary significantly depending on the approach and shutter integration time with respect to different motion patterns.

The estimation of the BLOB's centre-point is sufficiently accurate if the BLOB does not show a blurring effect. As shown in the Figure 4.9 (right column, middle image), the bounding box computation shows a higher error than the centre-of-mass approach (Figure 4.9, right column, bottom image) for BLOBs in motion. With the centre-of-mass approach the coordinates of all pixels of the detected BLOB are taken into account for the computation of the centre point. The algorithm combines the coordinate values of the detected pixels as a weighted sum and calculates an averaged centre coordinate.

0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0
0	2	1	1	1	1	1	0
0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0
0	2	2	2	2	2	2	0
0	0	2	2	2	2	0	0
0	0	0	2	2	0	0	0
0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0
0	2	1/2	1/2	1/2	1/2	1/2	0
0	0	1/2	1/2	1/2	1/2	0	0
0	0	0	1/2	1/2	0	0	0
0	0	0	0	0	0	0	0

Figure 4.10: Example of BLOB shape requires additional adjacency check.

As described in [23] the sequential processing of a frame requires an additional post-processing check for adjacency of the BLOBs itself. Dependent on the BLOB's shape or orientation the detection might separate pixels of one BLOB into two different BLOBs. As demonstrated in Figure 4.10 the pixels of the same BLOB have been identified as two individual BLOBs in the first two examples or, like shown in the third example, pixels are labelled twice.

4.3.4 Evaluation and Results

For the computation of the BLOBs centre point the bounding box and the centre-of-mass based methods showed the comparable results for the clear BLOBs with ideal circular shape. If a BLOB is not showing any blur effect, the applied method for computing the centre point has no major influence on the precision. This applies for all selected threshold values. The bounding box and centre-of-mass computations showed different results for images showing BLOBs with blur effects. The centre-of-mass results turned out to be closer to the expected BLOB's centre point. Results for one particular example are shown in Figure 4.11.

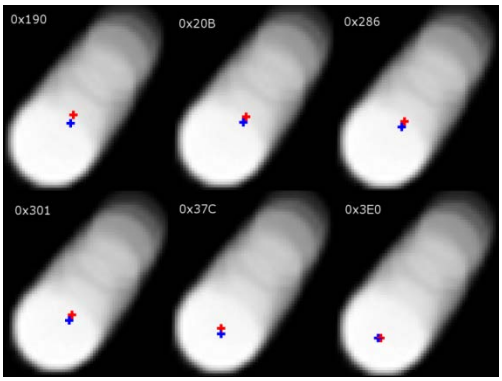


Figure 4.11: Computation of centre points for the BLOBs with motion blur using different threshold values. The centre-of-mass approach is shown in blue colour; the bounding box approach is shown in red colour.

Centre point computation with the centre-of-mass shows higher precision for BLOBs with blur effects, compared to the bounding box method. With the applied visualization on the VGA output the frame rate of the BLOB detection was restricted to 12 frames per second. Without visualisation the computation reached 46 frames per second. The threshold values were adapted to the applied image sequences. Therefore, the estimation of the value range is a configuration requirement prior to BLOB computations. For any threshold values above or below the specified range, the system is not able to accurately detect BLOBs.

The system performance has been evaluated on a fixed environment setup. Reported values about performance and resource allocation are given in Table 4.1.

Table 4.1: Resource allocation and benchmark results

	With monitor output		**Without monitor output**	
	Bounding box	**Centre-of-mass**	**Bounding box**	**Centre-of-mass**
Speed (fps.)	12	12	46	46
Camera clock (MHz)	25	25	96	96
System clock (MHz)	40	50	125	125
Max. system clock (MHz)	72	65	140	189
Allocated resources on the FPGA				
Logic elements	7 850	14 430	5 884	13 311
Memory Bits	147 664	27 3616	113 364	239 316
Registers	2 260	2 871	1 510	2 078

The “clock” performance results refer to the obtained frame rate during the benchmarking. The "Camera clock" is the particular clock rate that is used to read out the CCD sensor. The "System clock" is the clock rate for the BLOB detection module during the performance evaluation and the "Max. System clock" describes the maximum possible clock rate for the implemented design.

For the evaluation with monitor output a faster frame rate would have been possible in theory. But it would have required time consuming configuration of the camera settings and the VGA controller module. For monitoring the image data while performing the BLOB detection the VGA module has to run synchronized with the image capturing module.

The applied CCD camera has been used for the evaluation of faster frame rates. It will not be used in the target application for the active tracking device of the 6-MIG project, because of its missing ability to record infrared light. For this reason it was reasonable not to put more effort into the integration of the CCD camera than was required for the performance evaluations.

The CCD camera itself has an average clock rate of 48 frames per second for the applied configuration parameters. While both BLOB detection approaches would have been able to perform on faster frame rates, the estimation of the maximum performance was again restricted by the input source. The same problem about slow input sources existed in [23] as well, and did not allow evaluation the system for maximum performance. The resource allocation shows that centre-of-mass requires about twice as many logic elements and memory bits than bounding box approach. For more technical details please refer to [26].

4.4 Pattern Emitter using a Hand-held Micro-Projector

An earlier prototype of a laser emitter (Section 2.2 and 2.3.1) consisting of five laser modules. Unfortunately, multiple issues with this design were discovered during further project development:

- difficult adjustment of the laser's common intersection point;
- insufficient structural rigidity, leading to a quick self-de-adjustment;
- unacceptable size for a hand-held mobile unit;
- limited (in our case to only 5) number of laser point projections.

While unable to find an acceptable solution for this problem, an alternative temporary prototype was a necessity for the continuation of the project. At this stage a hand-held micro-projector was investigated as an alternative to a desired pattern emitter. Being a digitally controlled projector, it is able to generate any image up to 30 times per second within its maximum resolution. That mean, generating an image similar to the laser dots can be realized with little effort. The only difference would be a miniature scale of the entire setup (see Section 2.2) due to output power restrictions of the projector.

The working principle of the micro projector based on the DLP technology was selected, as it allows the light source inside the projector be replaced from visible to infrared. According to the maximum allowed power dissipation of the DLP unit, the illumination power of this projector would not be sufficient for the final full-scale design even after modification.

Therefore, a decision to leave it in the visual domain was supported by the findings, that during a testing phase it is much easier to work in the visible range.

In fact, shortly after we implemented our hand-held projector, another research group published their project [3] where they used the same model of micro-projector for a development of their collaborative entertainment system. They modified the light source of the projector from visible to infrared and combined the projector with an inertia measurement unit and a camera.

4.5 Visual 3D Emulator of a 3-Wall CAVE with a Laser-based Pattern Emitter

For the evaluation of the developed algorithms it is necessary to have reliable ground truth data (e.g. position and orientation of the device at every moment of a test sequence) for comparison with reconstructed data and to determine the latency (the delay between input and output) of the system. Such ground truth data is rather difficult to obtain in real world experiments, because its precision has to be considerably higher than that of the system itself to generate trustworthy results.

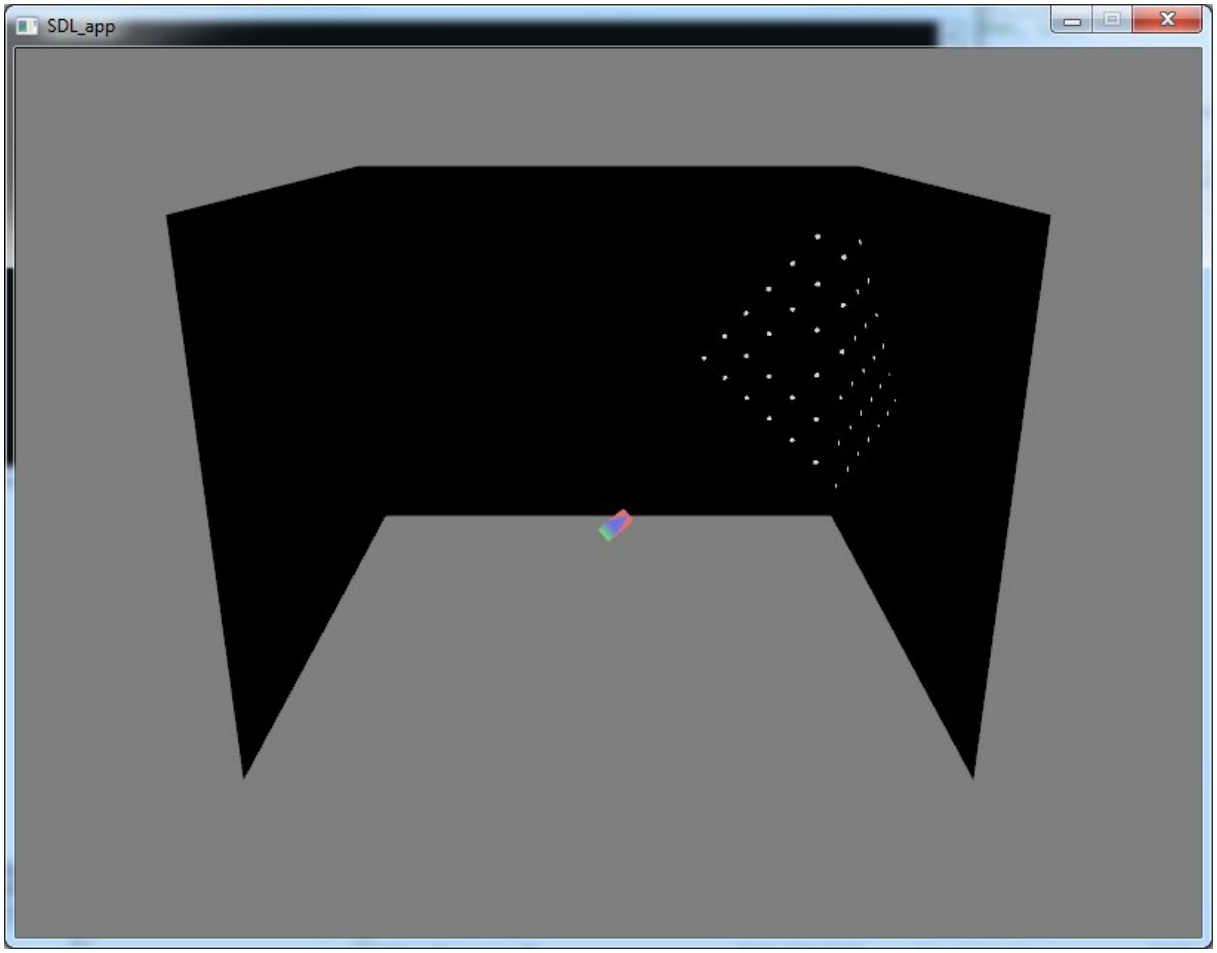


Figure 4.12: A 3D view of the simulator software. As shown in this figure, a virtual pattern projector generates a projected pattern on the centre and the right virtual screens of the simulator. Real-time pose controls of the projector provided via 6DoF desktop mouse controller (Logitech 3D-Connexion).

For that reason a simulation program has been developed within the scope of a student project [25]. The program allows modifications of the position and orientation of a virtual input device, which projects a custom pattern of laser beams on freely configurable virtual screens (see Figure 4.12, Figure 4.13) using a 6-DoF-mouse. The pattern is manually configurable. For example, the shape of the individual laser beams can be changed (e.g. to represent squares or triangles instead of dots) or some of them can be completely skipped to create gaps in the pattern. In addition, the projected point of the laser beam will be distorted towards ellipse, based on the angle between the ray and the screen surface, making the simulation more realistic. Subsequently a video stream for each screen is computed. Each stream contains typical camera-related effects like pixel noise, motion blur, radial distortion and others.

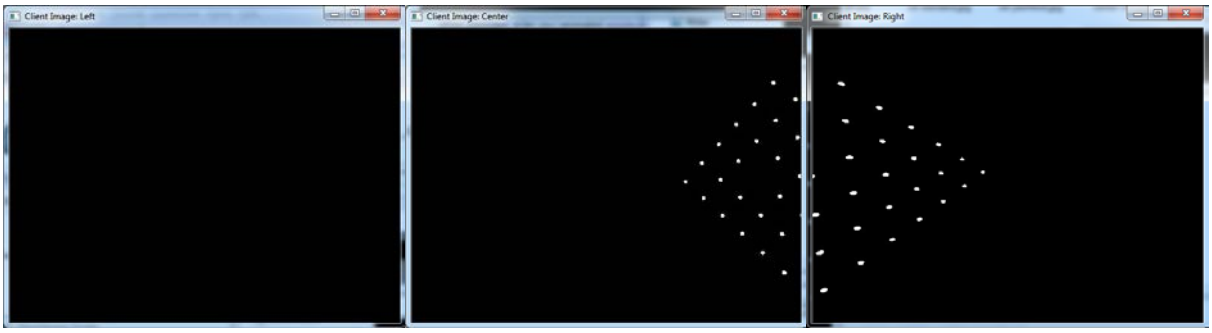


Figure 4.13: Three parallel output video streams include user-adjustable noise levels and perspectively-correct distorted projections of the simulated laser beams on the left, centre and right virtual screens respectively)

With the help of this simulator it is possible to evaluate developed algorithms efficient and fully automated. Every change in position and orientation of the virtual device can be recorded and later replayed for additional tests, so every experiment can be executed again under the reconstructed conditions. At every time of the experiment the position and orientation is known and can be compared to the reconstructed data of the tested algorithm. More details about this simulator can be found in [25].

4.6 Development of a Method for Calculating Position and Orientation of an Input Device

At this point of the project all developed solutions for sub-problems (points detection, points correspondence problem, pose calculations, correction using gyroscopes and accelerometers) are combined. As a result, software for calculating position and orientation of the device has been developed. It is capable of use different sources of the input data, such as live stream from camera, output of the developed 3D simulator [25] or a pre-recorded video file.

4.6.1 Algorithm Overview

The proposed algorithm can be divided into a calibration part and a runtime part. Calibration has to be performed only after repositioning the cameras, screens or changing the laser ray configuration.

During the run-time part, the laser spots have to be detected in the input images and camera distortions have to be accounted for. Then, for each laser spot, a 3D position in the global coordinate system is computed, using information about the position and orientation of the projection screens and their geometric relation to the cameras. The correspondence solving step then establishes a mapping from the set of observed laser spots to the input device's rays. After that, the device's position and orientation are determined using Levenberg-Marquardt iterative minimization [4]. Finally, this information is combined with the relative motion data delivered by the inertial measurement unit. Figure 4.14 shows an overview of the run-time part of the algorithm.

The algorithm presented here relies on a proper calibration of the cameras, the screens and the emitter device. In the following sections, we describe those calibration procedures. Calibration has to be recomputed again only after a change in the geometric configuration.

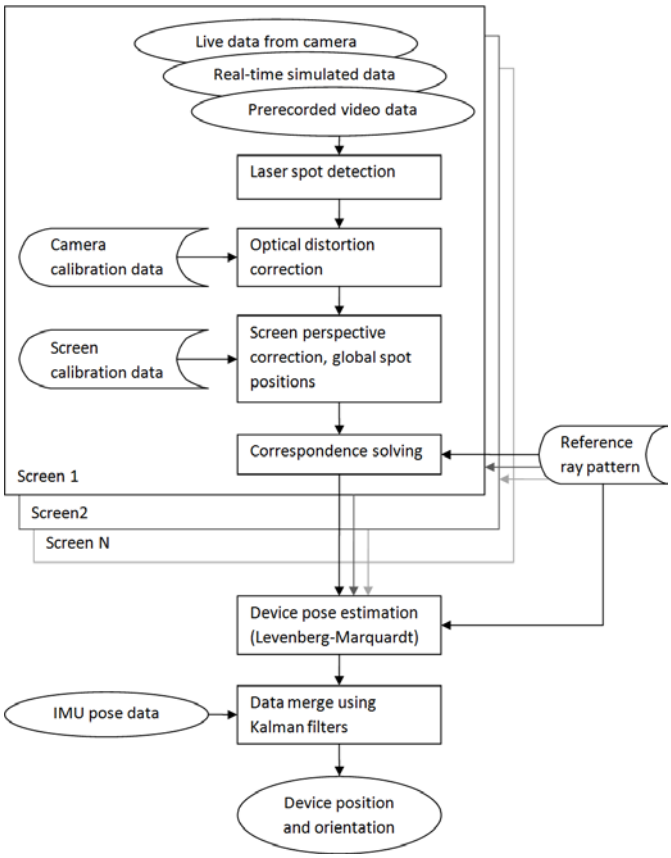


Figure 4.14: Overview of the algorithm. Image acquisition, spot detection and correspondence solving are performed for each screen individually (in parallel). The results are then combined for pose estimation.

In the current state of development, we operate on simulated or recorded video data (see Figure 4.14) only. A real-time simulation program generates simulated camera images. It allows moving a virtual input device with a configurable laser ray pattern. The advantage of this approach is that ground-truth data on the device's position and orientation is available, which facilitates evaluation of precision.

The device's position and orientation can be determined in real-time (roughly 150 frames per second with 512×512 images, using an Intel Core i7 860 processor). Solving the correspondence problem using p^2-invariants has proven to be reliable due to the voting table and fast due to the use of a kd-tree. Even with artificial noise introduced to the detected laser spot positions, most of the spots can be correctly mapped to the reference pattern. Wrong correspondences have been observed rarely. In most ambiguous cases, the algorithm labels the laser spots in question as "Unknown".

However, detailed evaluations on the precision of the laser spot detection and the pose reconstruction using the Levenberg-Marquardt algorithm have yet to be carried out.

The merging of optical and inertial data has been evaluated using an optical tracking system. First results indicate that combining both sources leads to more robust and reliable rotational data. However, determining the device's position, as opposed to orientation, using only inertial data is still imprecise due to noisy sensor readings and uncertainty in gravity vector separation procedure. Therefore, for position estimation, we rely on the detectability of a sufficient number of laser spots.

4.6.2 Camera Calibration

Every real (non-ideal) camera introduces some geometric distortions to the images it delivers, such as barrel distortion or pincushion distortion. This effect becomes apparent especially with wide-angle lenses.

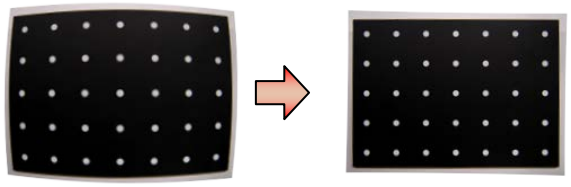


Figure 4.15: Radial distortion correction

In applications requiring high precision, such as the one presented here, those distortions have to be accounted for (see Figure 4.15). We determine the distortion properties of each camera by applying a standard procedure. This involves taking images of a grid pattern of retro-reflective markers with known physical dimensions from several perspectives. Then the parameters of a distortion model can be found. After the calibration procedure, the distortion parameters are stored and later used at runtime to determine the "undistorted" position of every detected laser spot in the image, that is, the position at which it would be seen by an ideal camera. This calibration procedure has to be repeated only after changes in the optical system of the camera.

4.6.3 Screen Calibration

Each camera observes one projection screen, which has to be entirely covered by its field of view. However, the cameras cannot be installed at an optimal position, being faced perpendicularly towards the centre of the screen, because they would interfere with the image projection system in case of a back projection, or with the user's working space in case of a front projection.

Therefore, in real-world environment, the cameras have to be installed observing the projection screens in a perspectively distorted way (Figure 4.16). In order to map 2D image pixel coordinates to global 3D coordinates on the screen surface, a homography transformation has to be computed. In our system prototype, this is achieved by using permanently installed retro-reflective markers in all corners of the projection screens (outside of the projection area) illuminated by coaxial infrared lighting sources mounted on the cameras (see [22] for detailed information about the light sources).

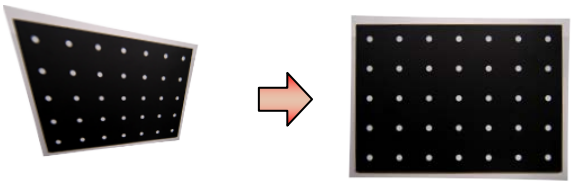


Figure 4.16: Perspective distortion correction (Homography transformation)

Determining these corner points' 2D image coordinates (using the same laser spot detection procedure as described in Section 4.6.2) and relating them to the points' real-world positions, the homography transformation parameters can be determined. This procedure has to be applied only after repositioning the cameras relative to the projection screens.

4.6.4 Proposed Input Device Calibration

In order to determine the relative orientation of each laser beam, a special calibration procedure has to be performed once (assuming that their relative configuration will remain fixed). The input device calibration procedure can be described as follows:

The device is fixed to a one-axis linear positioning system with rail-mounted movable carriage. Its axis is oriented perpendicularly to the projection screen.

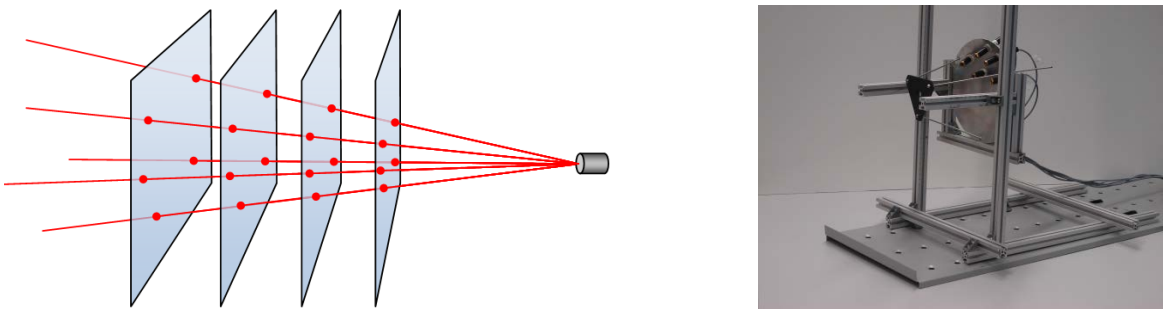


Figure 4.17: Virtual planes intersecting rays during the calibration of the input device (left). Early stage prototype of a linear motion platform allowing fixed steps only (right).

The device should be fixed to the carriage with the rays facing the projection screen in such a way that all laser beams' projections are visible on the screen. Then the device is moved stepwise towards the screen with a known offset, and images of the projection patterns are taken. The data obtained this way can be thought of as moving a plane through the laser rays of a stationary emitter and recording their intersection points, see Figure 4.17. Fitting lines through each individual laser spot using a least-squares technique allows the measurement of the ray directions and reconstruction of the 3D shape of the ray bundle.

4.6.5 Run-Time Laser Spot Detection

In each frame taken by a camera, the laser spots have to be detected. Their positions have to be determined as precisely as possible, as this data serves as the most important input to all following steps in the algorithm. The laser spot detection is implemented as follows: we sample the camera image according to a grid at every nth pixel horizontally and vertically (assuming that a laser spot always has a certain minimum size, not every pixel has to be checked). If the pixel's intensity exceeds a certain threshold t0, it is assumed to be part of a laser spot. Starting from this pixel, we collect all connected pixels whose intensities exceed a threshold t1 (which could be lower than t0) using a flood-fill search. In the next step, the connected region of pixels is checked against several criteria in order to decide if it is safe enough to assume that it represents a laser spot. Such criteria are its geometric shape and its size. In order to compute the centre point of the spot, we use a weighted average (each pixel contributes according to its intensity). This allows determining the positions of the laser spots with sub-pixel precision.

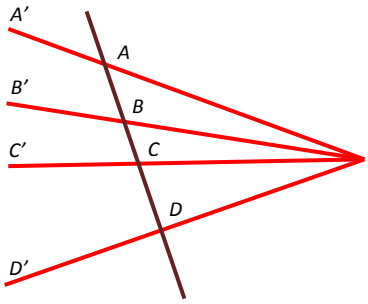
After the detection process, the image-space positions of all points are known. Those positions have to be transformed to the global reference frame. First, the distortion introduced by the camera lenses has to be corrected using the camera calibration data. The transformation to 3D coordinates follows using the screen calibration data.

The laser spot detection and transformation process is independent for each screen/camera. It is, therefore, designed and implemented to be executed in parallel on a multi-core processor. Additionally, we investigated the possibility of performing this process on dedicated FPGA hardware in order to improve scalability of the system by reducing the amount of data transfer between cameras and the PC and reduce the CPU workload [17].

4.6.6 Correspondence Solving

After the laser spots have been detected in the camera images and have been transformed to the global coordinate system, a mapping from the detected spots to the rays in the reference model has to be established. That is, for as many detected spots as possible, we need to find the corresponding ray that projected it from the emitter device.

Mathematically, there exist n! mappings from a set of n objects (detected laser spots) to another set of n objects (emitted rays). Even for only n = 10 laser beams, there are already more than three million mappings, which means that a brute-force approach of trying all possible mappings would be prohibitively slow.



$$\frac{|AC|/|BC|}{|AD|/|BD|} = \frac{|A'C'|/|B'C'|}{|A'D'|/|B'D'|}$$

Figure 4.18: An example of a cross-ratio as a projective invariant.

In order to keep the amount of computation within a feasible limit, we make use of the mathematical concept of p^2-invariants (perspective- and permutation-invariant) [5]. In n-dimensional projective space, a function taking n + 3 points as its input is called p^2-invariant if its result values remain constant regardless of any permutation of the input points (change of order) and regardless of applying any perspective transformation to all of them. In our particular case, since we operate in two-dimensional projective space, a p_2-invariant function can be defined for five two-dimensional points. It computes a five-dimensional vector of real values (the invariant vector). In the following, we describe the idea of solving the correspondence problem using p^2-invariants.

As a pre-processing step, we project the rays from the reference pattern onto any arbitrary plane, giving us 2D reference point coordinates. We then compute and store the invariant vectors for all possible five-point sets. For n points, there are $n! / (5! \times (n - 5)!)$ different five-point sets. With n = 20 points, this equals to 15504.

At runtime, for every screen we also compute and store the invariant vectors for all possible five-point sets of the respective detected points in image space. The underlying idea is that the screen position of the laser spots is just a perspectively transformed version of the corresponding 2D reference point set. For each five-point set on the screen, we find the five-point set in the reference pattern that has the most similar invariant vector. If the vectors are similar enough, we can assume that the five points from the screen correspond to the five points from the reference pattern. Due to another property of the p^2-invariant function, it is even possible to determine which point within the one five-point set corresponds to which point in the other five-point set.

In theory, there would always be a matching five-point set with the exact same invariant vector. However, in practice, the p^2-invariant function is very sensitive to small deviations in the point positions, which are inevitable due to camera noise, non-perfect calibration and floating point computations. As a result, we have to allow a certain amount of deviation. This means that for some five-point sets, we will obtain more than one five-point set from the reference pattern with almost the same spatial configuration. Only selecting the single best five-point set would mean ignoring all the others, although their chance of being the correct match is only slightly lower. Therefore, we follow the recommendations in [5] by using a voting table. Every potential match casts a vote into that table. By analysing the table after iterating over all five-point sets, we can determine for each laser spot the reference ray that it most likely corresponds to. In cases where this decision is very close, that is if one correspondence received only slightly more votes than another one, we output "unknown" as the result for that point's correspondence.

This conservative approach has shown to deliver more stable results than always selecting the best match, since errors in this stage can lead to errors in the pose estimation stage later. However, we try to keep these ambiguous situations to a minimum by using specially optimized laser patterns (see Section 4.6.8).

In order to achieve real-time results, the implementation of the described correspondence solving algorithm has to be optimized for speed. First, each five-point set can be processed in parallel (computing the p^2-invariant vector and searching for potential matches), since these tasks are independent from each other. However, the most critical part is finding potential matches from the reference pattern for a given five-point set from the screen. If the precomputed invariant vectors from the reference pattern are simply stored in a list or an array, the entire array has to be searched each time. This means that the time complexity is O(m2), where m is the number of five-point sets. Instead of using a list or an array, we store the precomputed invariant vectors in a kd-tree [6]. This is a spatial data structure that hierarchically organizes its items in such a way that a range search (searching for items in a neighbourhood around given coordinates) can be performed in time O(log n), n being the number of items in the kd-tree. With a range search, we can thus determine potential matches quickly and reduce the overall time complexity to O(m × log m).

4.6.7 Pose Reconstruction

Given the 3D positions of the laser spots on the screen surfaces and their correspondences to the reference pattern, an estimation of the global rotation and translation of the emitter device in 3D space can be computed. For this, the Levenberg-Marquardt algorithm is used to minimize back-projection errors. For an estimated pose of the emitter device, we compute these errors by virtually shooting laser rays from that pose, according to the laser ray pattern, and determining where they intersect with one of the screen surfaces. We use the distances between each simulated intersection point and the respective observed intersection point as the error values to be minimized by the Levenberg-Marquardt algorithm, see Figure 4.19.

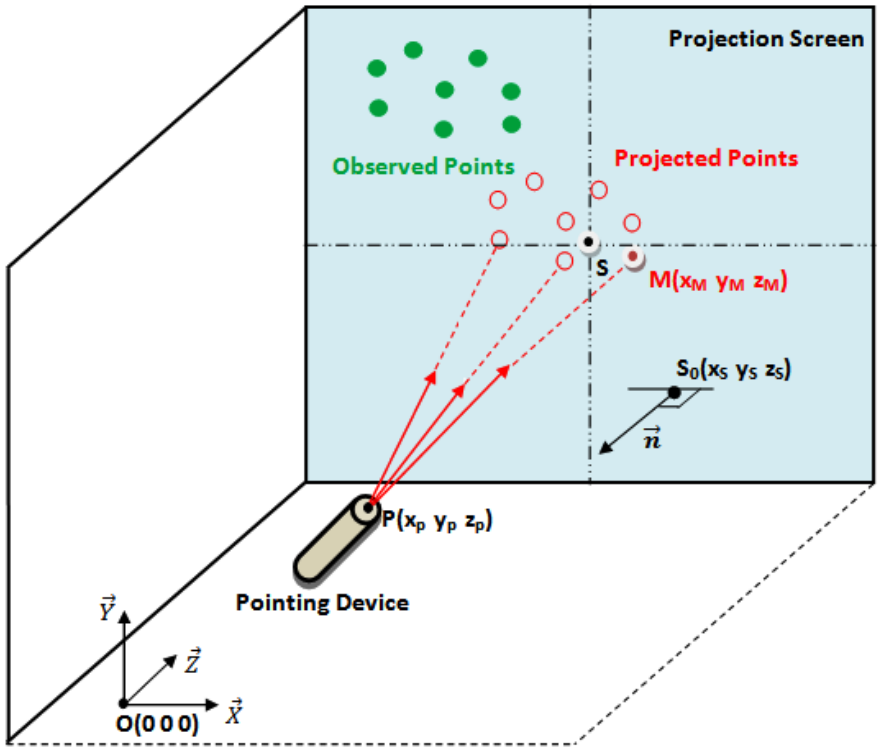


Figure 4.19: 3D-pose reconstruction of a hand-held laser emitter using Levenberg-Marquardt minimization algorithm

Any error in the correspondence mapping leads to an incorrect pose estimation result. However, a wrongly mapped point will typically show the biggest remaining error values after the Levenberg-Marquardt minimization. Hence, we iteratively remove points with errors above a certain threshold and apply the minimization again, until either all remaining errors are within the acceptable range or the minimum number of points is reached. In the latter case, we cannot safely determine the device's pose from the optical data alone and have to rely on the other sensor sources to deal with missing data. As such a complementary sensor system, an inertial measurement unit (IMU) has been integrated into the input device.

4.6.8 Generating Optimized Laser Patterns

As previously mentioned in Section 4.6.6, there are often ambiguous situations during the correspondence solving stage. This happens when for one five-point set of laser spots on one screen, there are several potential five-point sets from the reference pattern with very similar invariant vectors.

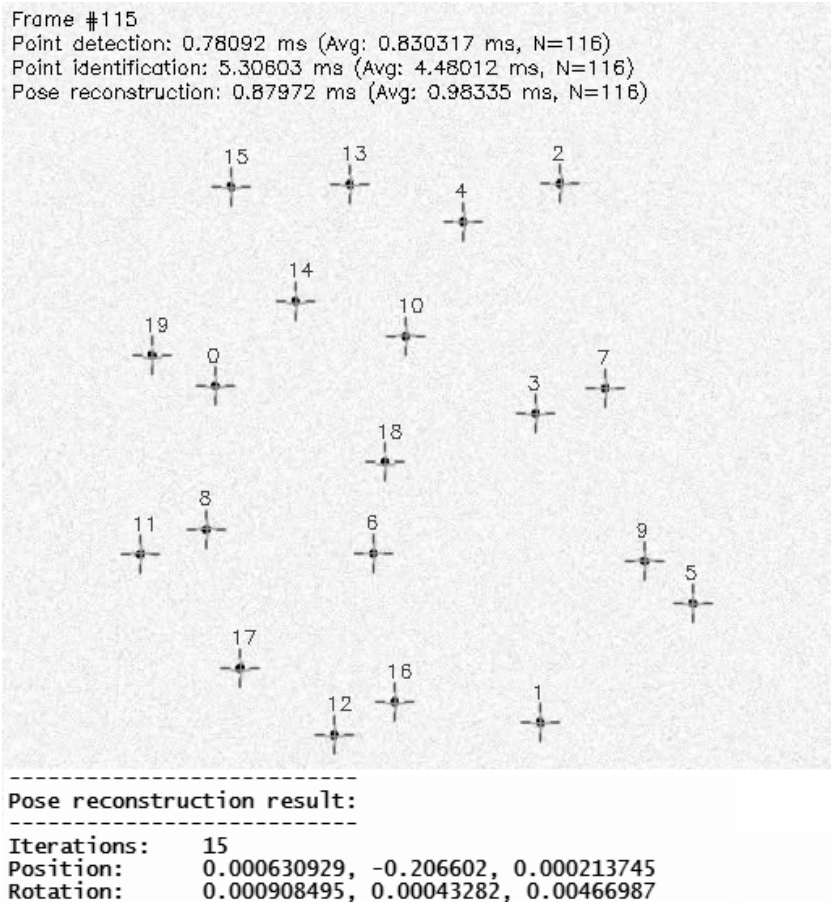


Figure 4.20: Detecting 20 points in a simulated camera image (including artificial sensor noise), solving for correspondences and reconstructing the device pose.

The frequency of this occurring can be reduced by designing the laser pattern in such a way that the number of five-point set pairs with almost identical p^2-invariant vectors is minimized. Also, no two laser beams should be spatially too close to each other, since this may result in "merging" laser spots. An additional criterion is that no three points should be collinear. Collinear points lead to numerical stability problems in the p^2-invariant vectors computation. The *simulated annealing algorithm* [7] was applied to create optimized patterns. Essentially, this is a method of "guided" random pattern creation. A set of optimized patterns were successfully created, one of them being shown in Figure 4.20. More information on the method can be found in [19].

4.7 Combining Optical and Inertial Measurement Data

The vision-based system cannot handle pose estimation when too many laser rays are projected outside the camera-monitored area, or when the pattern emitter is moving too fast. These cases of temporal incapability of a vision-based system to provide position and orientation of the device can be supported by an additional subsystem with gyroscopes and accelerometers (or inertial measurement unit, IMU) that can improve the output signal of the system to some extent. The inertial sensor is an attractive alternative for updating the motion information in situations when too many laser spots cannot be reliably detected or identified due to off-screen situations, extreme angles to the screen's surface or a very high rotational velocity, which leads to blur problems in the optical system.

The inertial sensors used are light and small with a high output data rate. They require significantly less computational power compared to the optical system. Some of the drawbacks of inertial sensors are a relatively low signal to noise ratio and only relative as opposed to absolute position estimation, which leads to "drifting" problems and error accumulation over time. In general, it is efficient to use the IMU as a short-term relative dead-reckoning system with a periodic update from the absolute optical system.

Therefore, a complementary subsystem based on an inertial measurement unit (IMU) providing an additional source of orientation data has been developed. This subsystem provides 3DoF of rotational data (2DoF of absolute, 1DoF of dead-reckoning) and is able to successfully cope with temporary outages from vision-based optical system and provides a short term tracking solution on its own.

The developed prototype provides inertial sensor data integration into the 6-MIG visual system output. It provides data acquisition from a gyroscope and an accelerometer, sensor fusion algorithms, and a graphical representation of the results in the 6-MIG simulator developed in [25]. The IMU data has been integrated into the simulator to graphically demonstrate measurement of rotations.

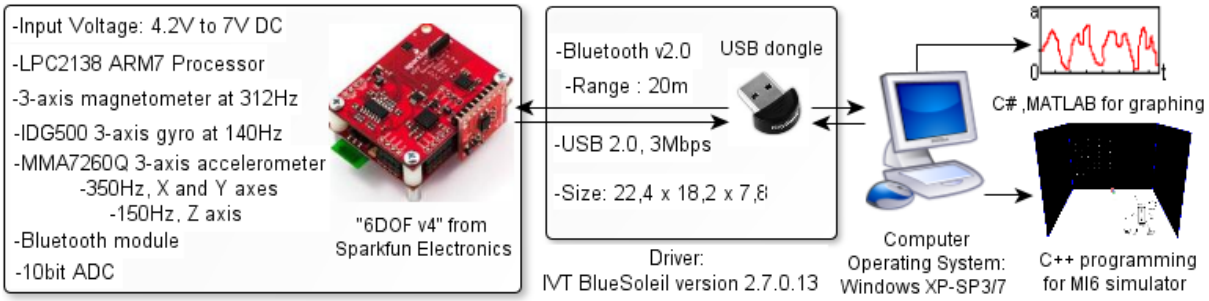


Figure 4.21: Hardware overview of an IMU-based complementary pose estimation subsystem.

The hardware configuration of the subsystem is shown in the Figure 4.21. In the prototype system, the IMU module "6DOF v4" from SparkFun Electronics with Bluetooth communication is used. It contains a 3-axis accelerometer MMA7260Q from Freescale Semiconductor and a 3-axis gyroscope IDG-500 from InvenSense Inc., see Figure 4.21 (left). For more detailed information on the realization of communication channels please refer to [28].

In the current implementation, the data is retrieved from the sensors over Bluetooth at 200 Hz. After applying an additional fault detection procedure (Figure 4.22, left) the low frequency component of signals in the data is filtered using the 4th order Runge-Kutta method (Figure 4.22, right).

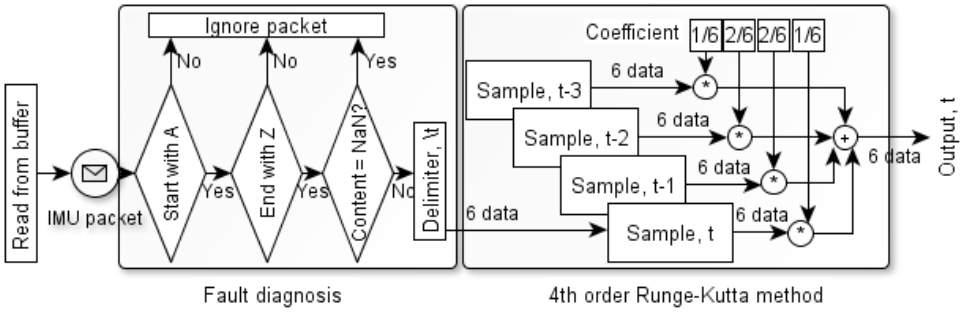


Figure 4.22: Faulty IMU data rejection (left) and IMU data low frequency filtering (right)

Two independent approaches for sensor fusion have been developed and evaluated. The first one is a classical sensor fusion approach using Kalman filters [8]. Kalman filters are known to be able to handle noisy measurements and combine the data delivered by gyroscopes and accelerometers. Additionally, they have been used to combine data between the vision-based optical system and the inertial subsystem. Our implementation partially compensates for the gyroscope drifting error and can successfully cope with motion acceleration.

The second approach, a quaternion-based sensor fusion algorithm, has been developed using spherical linear interpolation (SLERP) to support the laser-based system. As showed in Figure 4.23, as long as there is no motion, SLERP is used to combine gyroscope and accelerometer. In order to detect motion, the actual length of the accelerometer vector is checked to be close to one, representing gravity alone without any motion component. In addition, SLERP is used to integrate rotation detected by the optical system into others, whenever available. The interpolation coefficients were empirically set after a set of conducted experiments.

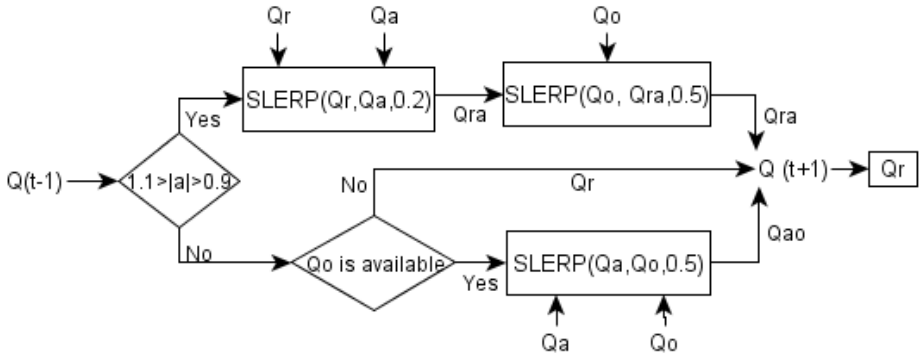


Figure 4.23: Sensor fusion algorithm based on quaternions and SLERP (spherical linear interpolation). Q_a – data from an accelerometer, Q_r – data from a gyroscope, Q_o – data from the optical system, |a| – motion indicator.

This approach enables inertial sensor data integration into the 6-MIG optical system output to improve the robustness and precision when measuring rotations. It allows combining angular rate and acceleration to enhance the tracking of the 3D orientation of the device (see Figure 4.24).

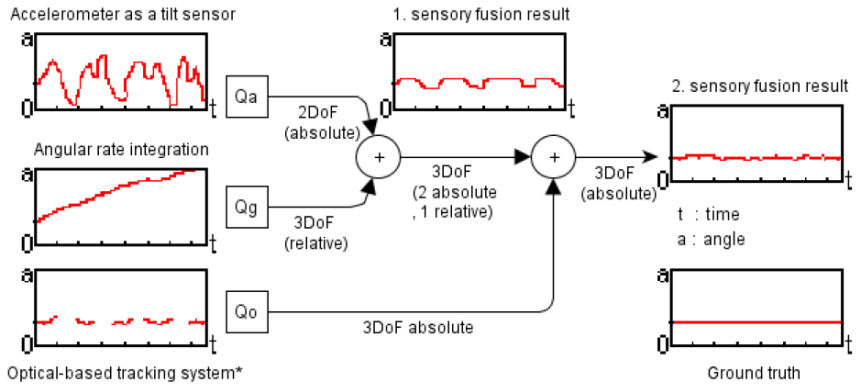


Figure 4.24: Graphical representation of sensor fusion of an IMU and a camera-based subsystem. Q_a – data from an accelerometer, Q_g – data from a gyroscope, Q_o – data from the optical system.

Due to the unavailability of the laser-based optical system at the moment of the implementation of the IMU subsystem, the external optical tracking system (OptiTrack from Natural Point) has been used to substitute the missing tracking data. The current prototype tracks orientation continuously with or without the optical tracking system.

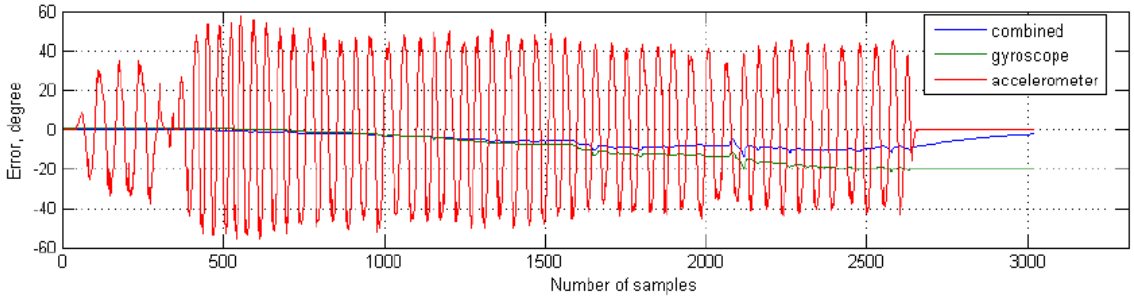


Figure 4.25: An example of the gyroscope drift correction algorithm in action. Series of strong input pulses generate an undesired drift which is successfully corrected by the algorithm towards the end of the graph.

In conclusion, combining acceleration and angular rate yields an improved performance. Evaluations showed that acceleration data is noisier than the others and drifting is reduced significantly when the unit is stationary. Figure 4.25 shows the effect of motion acceleration on the rotational data output. More detailed information about combining optical and IMU data can be found in [28].

4.8 Further Development of an FPGA Solution

The development and validation of a detection and tracking system for BLOBs on an Altera Cyclone II FPGA has been implemented. This subsystem supports different input devices for the image acquisition and can perform detection and tracking up to eight BLOBs in parallel. Additional modules for compressing the image data based on run-length encoding and sub-pixel precision for the computed BLOB centre points have been designed. For the comparison of the FPGA approach for BLOB tracking, a similar implementation in software using a multi-threaded approach has been realized. The subsystem can transmit the detection or tracking results on two available communication interfaces, USB and RS232. The analysis of the hardware solution showed a similar precision for the BLOB detection and tracking as the software approach. One major problem is the large increase of the allocated resources when extending the system to process more BLOBs. With one of the target platforms, the Altera DE2-70 board, the BLOB detection could be extended to process up to thirty BLOBs.

4.8.1 System Design

The general design of this subsystem is based upon the approach described in Section 4.3. It supports input devices as a composite analog video signal and a CCD camera connected directly into extension slots of the board. In addition, the design has been extended to use a customized industrial Gigabit Ethernet camera with a CMOS sensor.

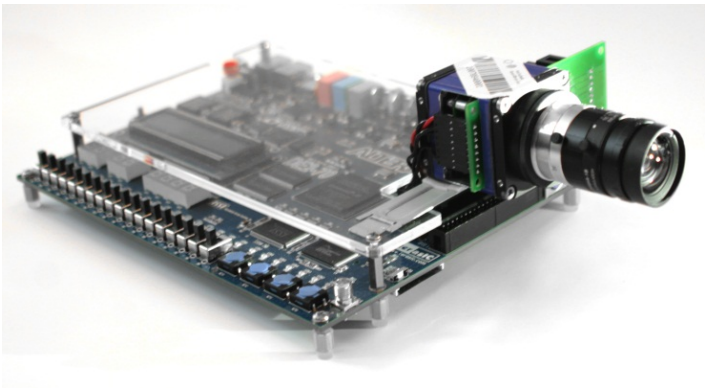


Figure 4.26: An FPGA development board with a specially-build GigE-Vision camera attached.

A special camera prototype for a direct connection to an FPGA board has been built[2], see Figure 4.26. It is based on the mvBlueCOUGAR-X 100, a camera with a CMOS sensor that delivers 10-bit greyscale images with a resolution of 752x480 pixels and allows frame rates of 117 frames per second. It is configurable over an Ethernet connection. It is connected to the DE2-70 board over one of the two GPIO expansion headers provided by the board. Based on the GigE Vision protocol this camera features an additional data output enabling image acquisition directly from the development FPGA-board. This configuration has been used for several other research subtasks as well [20], [21], [27], [31]. This prototype was used in the final, completely functional implementation of the 6-MIG prototype.

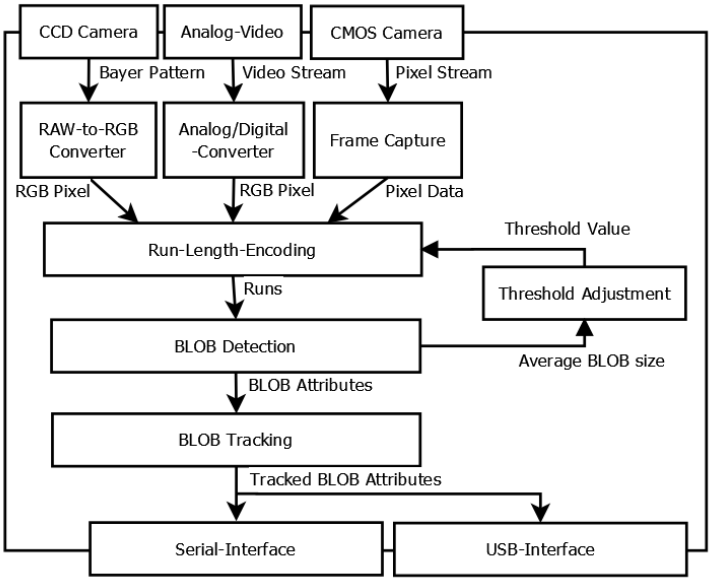


Figure 4.27: Schematic overview of the FPGA subsystem architecture, including the tracking approach.

[2] In cooperation with Matrix Vision GmbH. The fruitful cooperation and support by Matrix Vision need to be particularly acknowledged.

Figure 4.27 gives an overview of the modular design that has been extended by the additional processing parts of this subproject. For more accuracy in the estimated BLOB centre points, the system has been extended by a four digit sub-pixel precision module. In order to increase the flexibility of the system, an additional USB communication module for the transmission of the detection and tracking results has been designed.

4.8.2 Tracking in Hardware

The design approach for the tracking task in hardware uses the Euclidean distance between the BLOBs centre points of two consecutive frames to compute the tracking parameters. The capabilities of an FPGA enabled the parallel computation of the Euclidean distance between one new BLOB and the set of old BLOBs. Other steps, such as the identification of the closest new BLOB, have to be done in a sequence to avoid multiple assignments of the same BLOB ID. For the control of the process flow, the algorithm has been divided into smaller steps that have no data dependencies. The processing of those different steps is driven by a state machine design. Figure 4.28 gives a simplified representation of the state machine. Some states, such as "write results to output FIFO" consist of several sub-steps.

The additional tracking procedure to match new BLOBs by a list of expected neighbour BLOBs if they could not be matched by Euclidean distance has not been designed within the scope of this project.

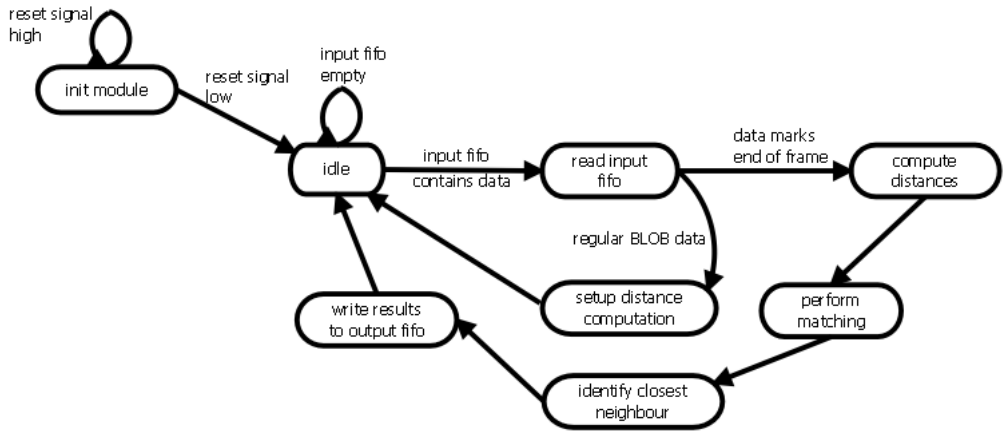


Figure 4.28: Tracking task in hardware module. State machine design.

The tracking has been designed in Verilog using IP cores and block-design files to realize sub-modules. The several steps for computing the Euclidean distance have been pipelined in a module, shown in Figure 4.29. Each block represents an IP core that performs a mathematical operation, such as multiplication, addition or square root. With respect to the detection module the tracking works for six to eight BLOBs.

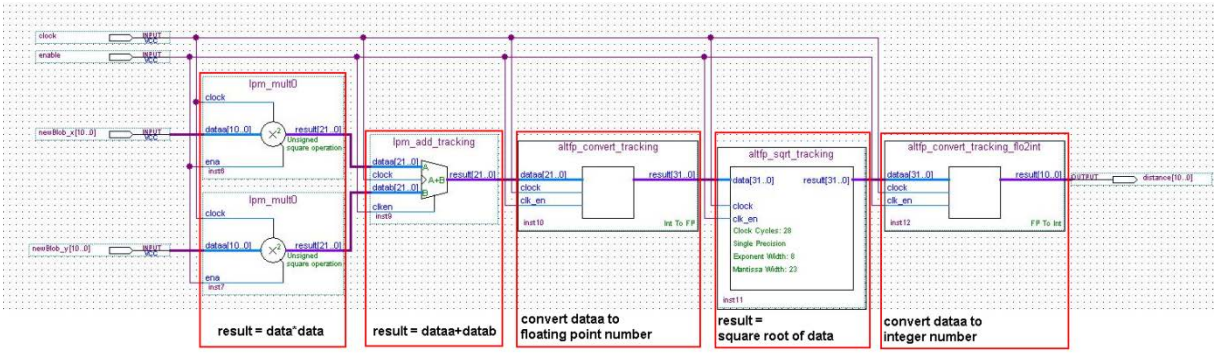


Figure 4.29: Pipelined computation of Euclidean distance.

For the visualization of the tracking results the BLOB data is transmitted via RS232 to a connected computer. The received data is written into a log file and visualized in an OpenCV based C++ program for evaluation purposes.

4.8.3 Variable Threshold Adjustment

An identification of the pixels that might belong to a BLOB in a processed frame works based on a threshold value check. In the hardware design of the previous approach [23] and [26] (Section 4.2 and 4.3) the configuration of this value was done by the user using the I/O interface of the target platform. In the current approach the brightness of the spots from the light emitting device can vary. Several factors will have an impact on the BLOBs appearance, such as the angle between the light source and the projection surface, the distance between the light source and the projection surface or the speed with which a user moves a light source in the environment. This might cause changes in the BLOB's shape or brightness.

For a continuous update of the threshold value, the module will receive an updated averaged BLOB size of the last frame processed. The automatic adjustment of the reference values shall be performed once the user has finished the calibration of the system after start-up.

The module for the automatic threshold adjustment has two processing modes, “setup” and “running”. Using the switches of the DE2 board, the user can switch between both modes during run-time. The reference values for the size of the X-/Y-axis, the threshold value for the BLOB detection as well as minimum and maximum values for the threshold are initialized with fixed values at start-up. The module receives the averaged values for the BLOB's size from the BLOB detection module after each frame. If the size of the BLOB is not within the tolerance level, the threshold value is changed accordingly. The automatic adjustment of the threshold value only works in “running” mode. For setting a new initial threshold value, the user has to activate the “setup” mode. The computed threshold value is transmitted to the BLOB detection module and updated every clock cycle.

4.8.4 Pre-Processing Run Length Encoding

For the application of the Run-Length-Encoding (RLE) as a pre-processing task in the BLOB detection system, an analysis of the algorithm has been performed. It was analysed for data dependencies to parallelize parts of the process. The analysis identified five conditions that had to be checked for every pixel to be processed to achieve a reliable RLE of the image data.

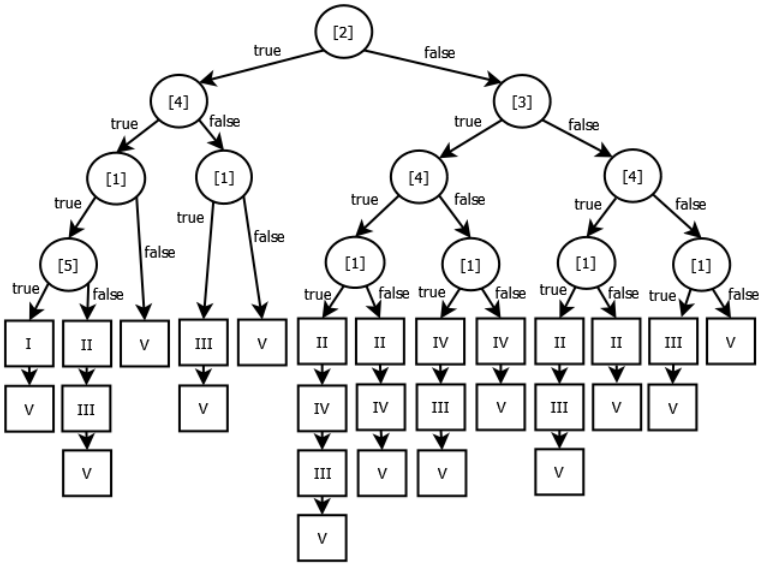


Figure 4.30: Decision tree for RLE conditions and processing steps.

Based on these five conditions a decision tree has been created, to identify patterns and dependencies in the processing flow (Figure 4.30). Different combinations of the conditions could be grouped together, since they require the same processing steps for a correct RLE encoding of the pixel data. This allowed an optimization of the processing steps for the implementation in hardware. The RLE processing has been separated into five steps that are executed in different order, dependent on the condition check results.

The implementation of the Run-Length-Encoding was done according to the optimized decision tree for conditional checks and processing steps (Figure 4.30). The conditional checks are performed in a single step in parallel. Based on the outcome the matching RLE processing step is executed in the following step. Each combination of results for the conditional check is mutually exclusive. This allows the implementation of all processing steps in parallel.

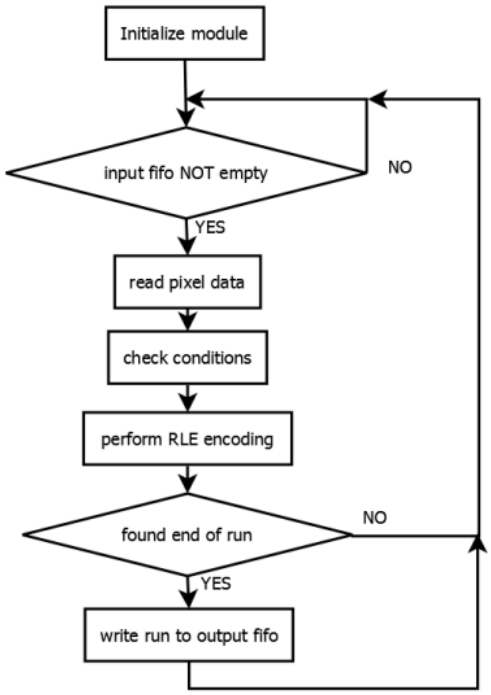


Figure 4.31: Simplified representation of the functional process in the RLE module.

Once a complete run is detected, the run's data is sent to the output FIFO before processing the next pixel data. Figure 4.31 gives an idea of the functional steps inside the module for RLE encoding of the BLOB data.

4.8.5 Sub-Pixel Precision

The previous BLOB detection approach (see Section 4.3.3) was based on integer values for all kind of data values in the system. With the applied computation methods for estimating the BLOBs centre points, the results usually showed a rounding error. To solve this problem the design has been extended with sub-pixel precision computations for all values that are in relation to the BLOB detection results.

The declaration for the register sizes and the data channels between the modules in the BLOB detection approach are hard coded. In addition, all values are defined as integers. This caused a rounding error for the computation of the average brightness of the BLOBs and for the centre point coordinates as well. A declaration of floating point or fixed point values is not supported in HDLs right away. The handling has to be realized by developer or by applying IP cores from the Altera IP core library.

The extension of the system with sub-pixel precision has been combined with the centralized configuration of the hardware design. This allowed a variable adjustment for the different hardware

modules in a single settings file. The sub-pixel precision algorithms use a fixed-point arithmetic. For this approach the implementation has been extended for fixed-point configuration with four decimal places.

4.8.6 USB Communication Module

In the previous approach the BLOB detection results had been transmitted via a serial interface (see Section 4.2.1). The bandwidth was sufficient for the existing approach, since the results did not contain very comprehensive information. Larger amount of data, resulted from sub-pixel precision computations and a system with a more common interface for higher flexibility, motivated the design of a communication module using an USB interface of the DE2 board. This also supports the extension of the BLOB detection and tracking results, which are transmitted from the target platform to the connected host PC.

A module for the USB communication was implemented to work with the same input data as the serial communication module. For a shorter development cycle it was decided to build the module based upon an existing example module out of the DE2 demonstration sources from Altera. In the modified version the BLOB detection and tracking results are transmitted to the connected host-PC with one byte per message. The communication requires driver software on the host-PC to receive the system results. It was integrated into the software for receiving and visualizing the BLOB detection results on the host-PC.

4.8.7 Verification and Validation of the System

Pre-Processing Run-Length-Encoding: Written test bench files for a functional simulation of the RLE based pre-processing contained several different cases to cover the described conditions (see Section 4.8.4). Figure 4.33 shows the representation of the test case data that has been used for the verification. For the verification of correctness, the output was compared to manually determined results. The results of the functional simulation matched with the expected values. This did show the logical correctness of the design.

In the next step the execution of the hardware design on the target platform has been evaluated. For the verification of correctness of the RLE pre-processing steps on the DE2 development board, the previously applied test data has been reused. By integrating the test data into a separate module of the hardware design, this module provided data which allowed the evaluation of the overall system's output.

It could be verified that the resulting data matched the expected values for the integrated test data. For the RLE pre-processing module a maximum clock rate of 145 MHz has been measured. This allows the system to process an input image of 640x480 pixels with up to 117 frames per second.

For a second test procedure with real-time image data, a D5M camera has been used as an input device. The DE2 board has been placed in the test model that is shown in Figure 4.32.

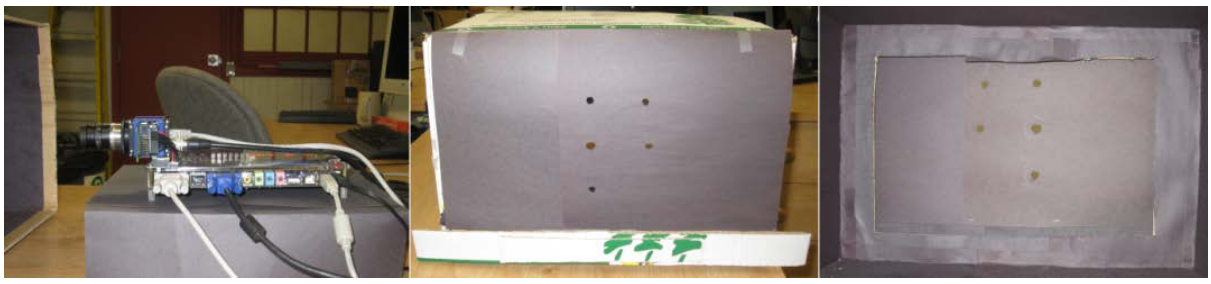


Figure 4.32: Test setup for threshold adjustment validation. Camera and the FPGA development board (left). Test pattern mask applied to the back side of the box, view from outside (middle) and view from inside (right).

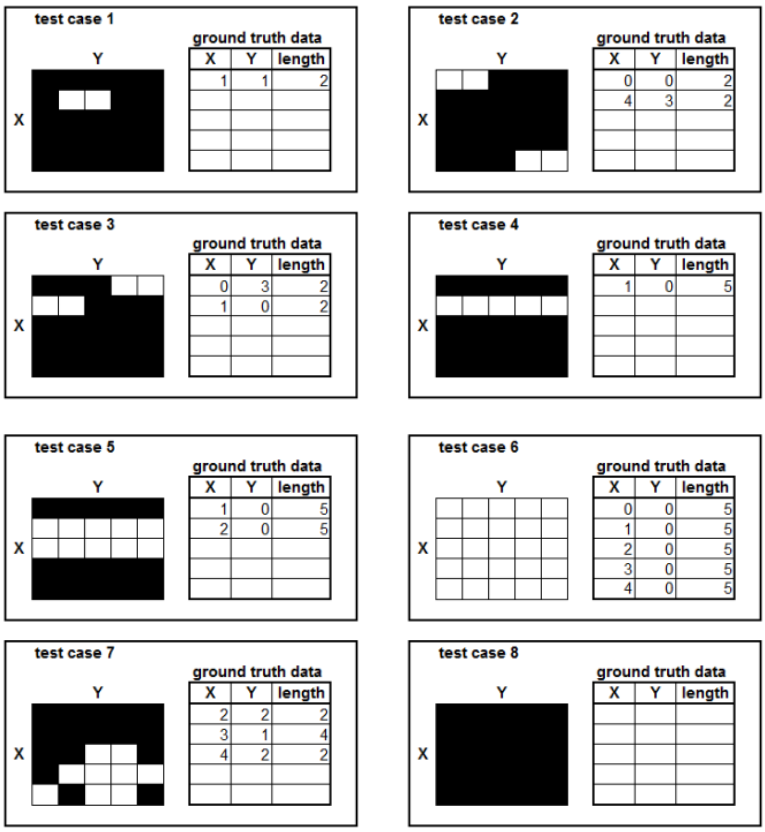


Figure 4.33: Test cases for verification of RLE pre-processing module

Evaluation of the Variable Threshold Adjustment: An execution of the module has been evaluated on the hardware platform without using specialized test bench modules to create input data. The input data has been acquired with the D5M camera which is attached to the target board.

Figure 4.34 depicts how the input data is used to set the reference values for the BLOB's size. The diameter of all BLOB's on the X- and Y-axis is used to estimate an average size for all BLOBs. This reference value will remain fixed once the system is set to "running" mode. The threshold value for the BLOB detection will be incremented or decremented by one according to the change of the BLOB size.

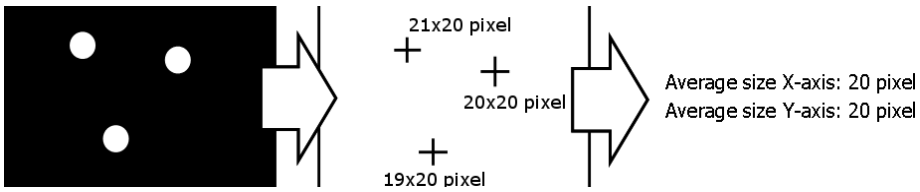


Figure 4.34: BLOB size averaging process for threshold adjustment.

For validation, the system is configured during runtime while the camera remained in a fixed distance to the model. Figure 4.32 gives an idea of the setup which has been used. Once the configuration was complete, the system was set into "running" mode to observe the threshold adjustment. Moving the model, that is showing the white BLOBs on black background, back and forth did change the BLOB size. This did allow observing the change of the threshold value until the average diameter for the BLOBs of the current image matched the reference value of the configuration procedure.

The validation of the threshold adjustment has been performed with different initial configurations. It could be verified that the threshold value changed accordingly to the change of the BLOB size. The threshold value remained in the defined minimum and maximum values.

4.8.8 Evaluation of Tracking in Hardware

The analysis of the resource requirements gives an idea of the applicability of the detection (Figure 4.35, left) and tracking (Figure 4.35, right) on the FPGA board. However, with an increasing number of BLOBs the maximum performance of the tracking module decreases much faster than the detection module. Figure 4.35 shows the relation between the amount of BLOBs that can be processed by the system (horizontal axis) and the maximum performance in MHz (vertical axis). It also shows the impact of the sub-pixel precision on the system.

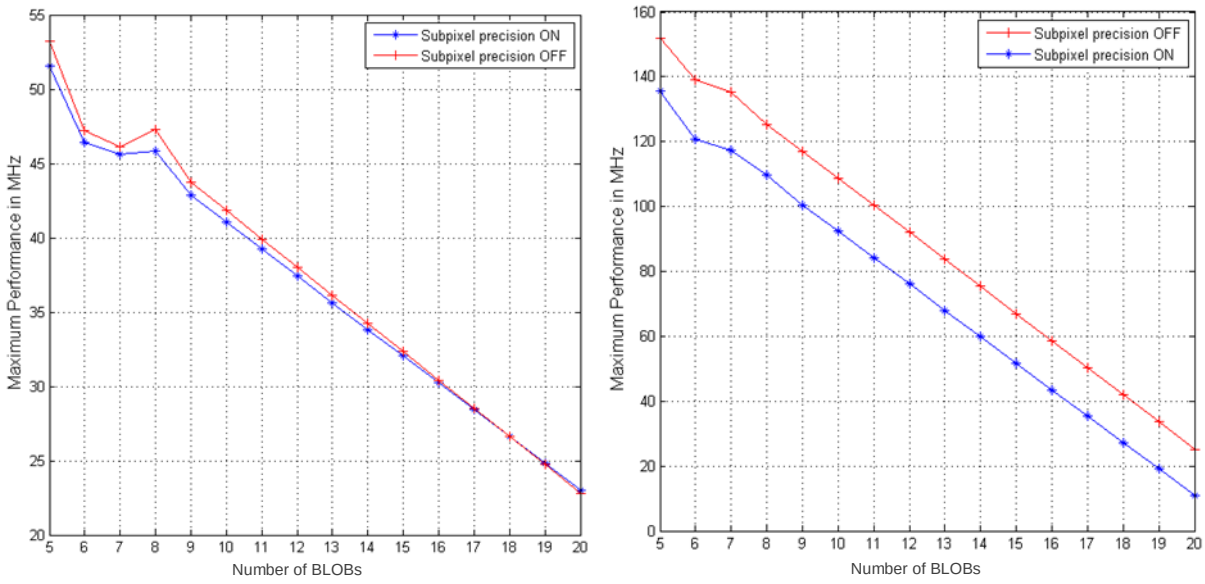


Figure 4.35: Performance for BLOB detection algorithm (left) and BLOB tracking algorithm (right) with (blue) and without (red) sub-pixel precision.

The difference in the maximum performance for the BLOB detection with and without sub-pixel precision is relatively small compared to the difference with and without tracking module. However, in both cases the clock rate is still high enough to process the frames at the rates provided by the input devices.

One problem of the hardware design is the evolution of the resource requirements for the detection and tracking modules with an increasing number of BLOBs. In Figure 4.36 left it can be seen that the resource requirements for the BLOB detection, using the centre-of-mass approach for the centre point estimation is already too high for the available target platform DE2-70 for the module implemented for fourteen BLOBs. Even with the bounding box based approach the maximum number of BLOBs which can be computed in parallel on the DE2-70 board would be approximately thirty.

For the tracking module, the shortage of resources becomes even more prominent, since the tracking module cannot be used without having the detection module included. The tracking module has to support as many BLOBs as the detection module. In Figure 4.36 right, it can be seen that the design is running out of space for DE2-70 target platform with eight BLOBs for the centre-of-mass method and twelve for the bounding box approach even without a sub-pixel precision.

Because of the resource requirements for the test bench files in hardware, the number of test samples had been kept minimal. Additional experiments that are implemented in hardware will affect the design and the performance values. Therefore, the test bench files have not been used for comprehensive performance evaluations.

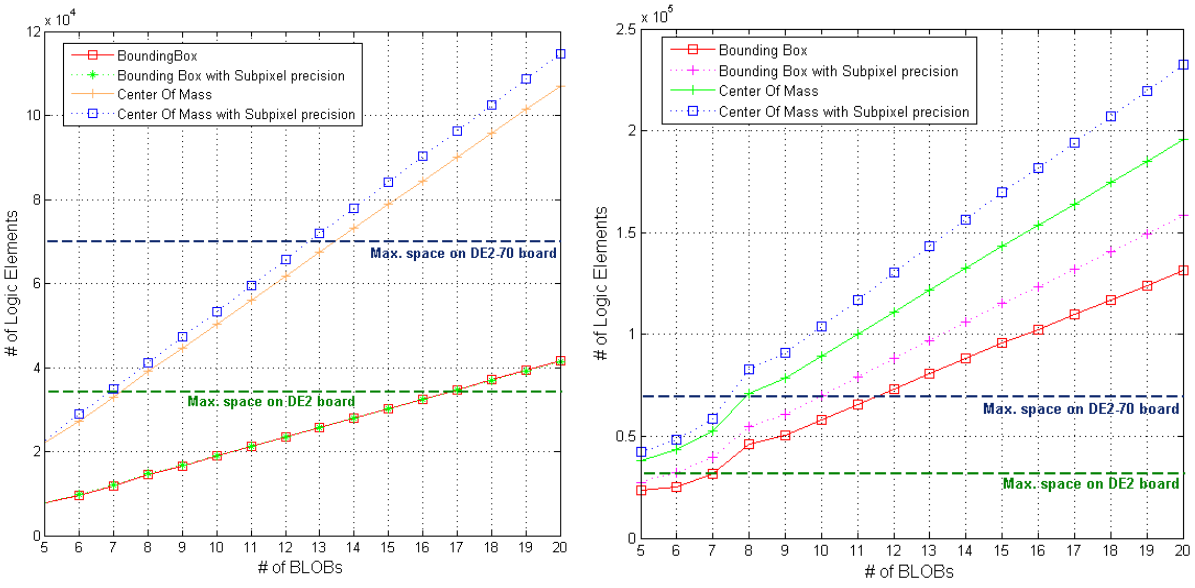


Figure 4.36: Resource Allocation for BLOB detection algorithm (left) and BLOB tracking algorithm (right) with increasing number of BLOBs.

For a better validation of this approach, it would be useful to have the ability to feed synthetic test data into the system. This would allow the computation of test data with ground truth values that can be used for a more precise evaluation. Having such an interface would solve a problem of changing validation results and resource requirements that are caused by integrating test benches into the hardware design. Another design implemented at a later stage of the project (see Section 4.10) is dealing with this problem by utilizing a secondary FPGA system that functions solely as a test-bench module. This allows execution of comprehensive evaluations strategies without any impact on the target design.

Additional and more detailed information on the FPGA-based approach described in this Section 4.8, can be found in [27].

4.9 Data Traffic Reduction using FPGA-based Image Pre-processing for BLOBs Detection

The FPGA-based approach for BLOB detection that has been developed during early phases of the project has limitations in number of points being tracked and a low tolerance to noisy data input (see [27], [20]). With increasing number of detected points the previously developed communication method via serial port has shown to become a bottleneck. A faster interface such as Ethernet communication level was considered to be implemented. Therefore a new approach in BLOB detection on FPGA and faster communication techniques has been developed. There were two ways developed to pre-process and transfer image data to a host PC.

The first method splits the task of BLOB detection between FPGA and PC. The FPGA separates foreground pixels from the background pixels and sends them to the computer, while the computer calculates the centres of the light spots.

The second method performs single pass BLOB detection on FPGA and sends the centres of the light spots to a host PC. A custom protocol has been developed for communication over Ethernet between the FPGA and the PC. Evaluation results show that, depending on the image, both methods contain a trade-off between frame rate, precision and flexibility. A co-design solution is more flexible, but suffers from a low frame rate when the amount of foreground pixels is high. A pure FPGA solution allows high frame rates, but is less flexible and less precise than the co-design solution.

4.9.1 Shared BLOB Detection on FPGA and PC

One drawback of BLOB detection performed purely on an FPGA is the lack of flexibility in a way how features of the objects are extracted. A fundamental insight of this approach is that instead of processing images on the FPGA, it is possible to perform only coarse-grained processing on the FPGA and to send important parts of the images to the computer for further processing.

Figure 4.37 shows a high level view of the approach. Camera images arrive in a sequential scan order on the FPGA. A small part of the image is stored in a frame buffer for a later look up. First, foreground pixels are separated from the background pixels by using simple thresholding. Pixels whose intensities exceed the threshold are forwarded to a run length encoder that packs neighbouring foreground pixels into intervals. The intervals are inserted into the interval table, where each line corresponds to a line of the camera image. After receiving a complete line, the interval table is used to look up the intensities of each foreground pixel in the interval. Subsequently, the intensities are sent to the computer according to a custom data transfer protocol. The PC performs BLOB detection on the received foreground pixels and extracts features from every detected BLOB.

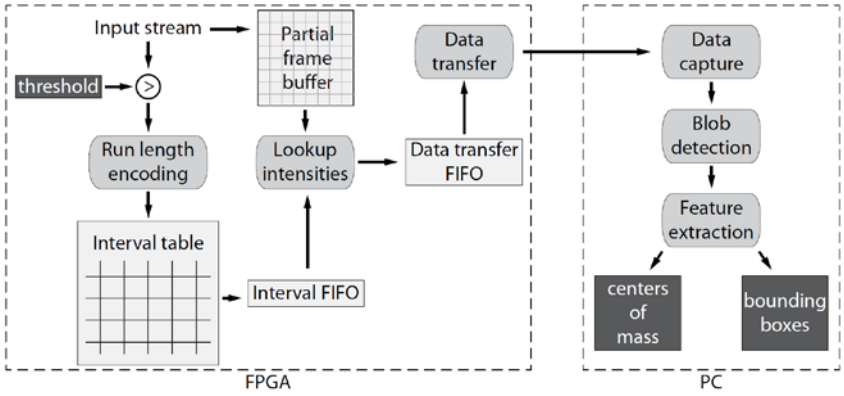


Figure 4.37: Identification of regions of interest – a high-level design overview. The camera image is partially stored in a frame buffer that is realized in the block RAM. The interval table is stored in the registers.

A straightforward approach to BLOB detection on the computer has been realized. Its underlying idea is to examine every pixel that exceeds a threshold and determine whether any of its 8 neighbours also exceed the threshold. Such pixels are grouped together. After all thresholded data of the frame are received, the centres of mass are calculated by computing the average coordinate weighted by their corresponding intensity values. This is calculated for each group of neighbouring pixels. The resulting centres of mass are visualized on the screen, which allowed evaluation of the data received from the FPGA. This approach is not optimized to take advantage of the structure of incoming data and was only used for visualization and evaluation purposes. However, such optimizations are planned as future works.

4.9.2 Extended Pixel Thresholding with Spatial Margin

Regions of interest are identified by applying a variant of thresholding. In classical thresholding, all pixels whose intensities exceed a predefined threshold are retained, and all other pixels discarded. However, in some applications, simple thresholding can result in a cut-off of important pixels on the edge of a BLOB, which might compromise the precision of subsequent feature extraction. Figure 4.38 (top left) shows a typical light spot produced by a laser on a wall (top). The lower part of Figure 4.38 plots the intensities of pixels obtained from the middle of the light spot.

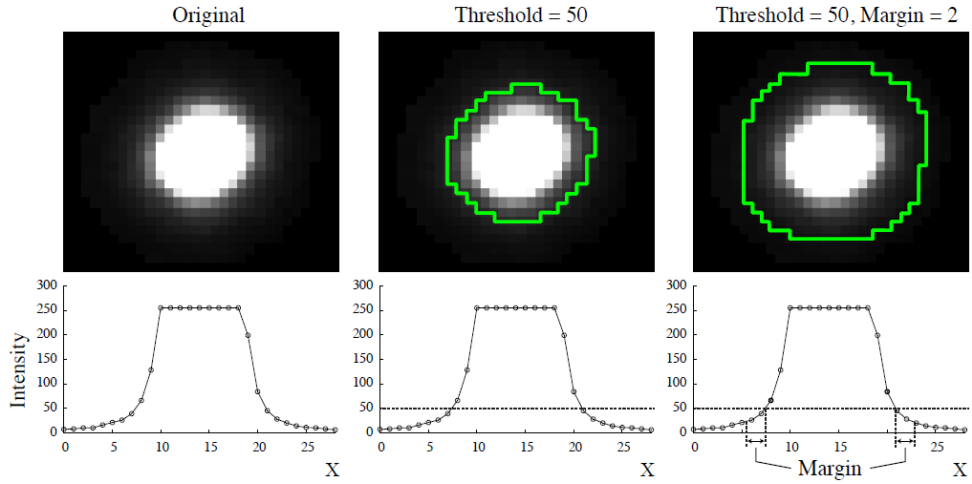


Figure 4.38: Two variants of thresholding used for identification of regions of interest. On the left is the original image. In the centre, simple thresholding (intensity value 50) is applied. On the right image the border (generated by the application of simple thresholding) is extended by a particular margin. The border is marked in green. Pixels inside the border comprise the foreground. The upper subfigures show the original image and overlaid borders. The lower subfigures show intensities of pixels obtained from the line that cuts through the centre of the light spot along the X-axis.

Simple thresholding, as shown in the middle of Figure 4.38, with a threshold of 50, corresponds to drawing a line parallel to the X-axis and discarding all pixels that are below the line. If the threshold is set too high, pixels that are important for subsequent feature extraction will be cut off. However, if the threshold is set to a lower value, too many unimportant pixels might be marked as foreground and negatively impact subsequent feature extraction. To overcome this problem, an additional margin extends the border obtained from applying simple thresholding. While the threshold dissects the image along the intensity axis, adding a margin extends the border along the X and Y-axes. The lower right part of Figure 4.38 shows the result of applying thresholding and margining on the intensities along the X axis. More detailed information about this technique can be found in [21].

4.9.3 Development of an Interval Table to Obtain Region Extension with Fixed Margins

To implement thresholding with a fixed margin on an FPGA, an auxiliary abstraction called interval table has been developed. An interval table consists of several rows, each containing a fixed number of intervals. An interval is defined by its left and right coordinates (inclusively). Adjacent foreground pixels in a single line are grouped together into intervals. Rows of an interval table correspond to lines in an image. All rows are sorted by their left coordinates. If a line contains no foreground pixels, no entries will be written into the interval table. The size of the interval table is defined by the margin of M pixels and the number of intervals N. The margin defines the depth of the table, whereas the number of intervals defines its width. The table must be maintained in a way that no intervals overlap, which can be done by merging overlapping intervals.

When the end of the line is reached, the whole table is shifted up, and the most upper interval list is ready to be sent. Since the information about the foreground pixels in a line becomes apparent only M lines after their encounter, at least M lines of the image have to be stored in the frame buffer. Pixel values of all the intervals in the most upper list can be extracted by knowing *left* and *right* and the current Y-coordinate. This information is enough to calculate an address in the frame buffer, where the corresponding pixel values can be found.

4.9.4 Single Pass Connected Component Labelling

In the beginning of this subproject, the objective was to perform BLOB identification on an FPGA and to send the dimensions of the bounding boxes and the pixel values enclosed by the bounding boxes. The approach has been developed until it was noted that no external memory controller was available for storing the whole image in an external memory of the DE2-70 board. At that moment the focus of the development was shifted to a method that does not require an external memory, which resulted in the implementation presented in the Section 4.9.1 - 4.9.3. A more elegant and concise approach is presented here, that was put on hold in the course of this project due to discovered hardware limitations. This approach is similar to the single-pass connected component labelling presented in [9]. The basic idea is to label each pixel in the frame with a number based 8-connectivity principle. The labels are only used to extract features of objects on the fly and are discarded as soon as the end of the frame is reached.

Figure 4.39 illustrates the approach to single-pass connected component labelling that was developed in this subproject. At the core of the approach is a finite state machine that checks a long list of conditions in order to decide how to label foreground pixels.

Foreground pixels are determined by comparing their pixel values with a predefined threshold. If the pixel value exceeds the threshold, then it is a foreground pixel, otherwise, it is a background pixel. The source of labels is provided by a stack that is filled with decreasing numbers before BLOB detection is started. Whenever the main FSM needs a fresh label, it obtains it from the stack and pops the label, and in this way makes the stack ready for the next read.

Upon the assignment of a new label, two events take place at once: 1) the label is saved as active in the active labels array, which is done by using the label as address and writing a 1 under that address; 2) the label is provided to the run-length encoder FSM, that saves the run that just ended together with the provided label in a FIFO. The run (or interval) that is stored in the FIFO will be read in the next line when its boundaries are reached. This allows the labelling FSM to find out whether a foreground pixel is adjacent to the run in the previous line. The FSM compares the left coordinate of the interval with a currently active pixel, and sends request to de-queue the FIFO in case the coordinate is larger.

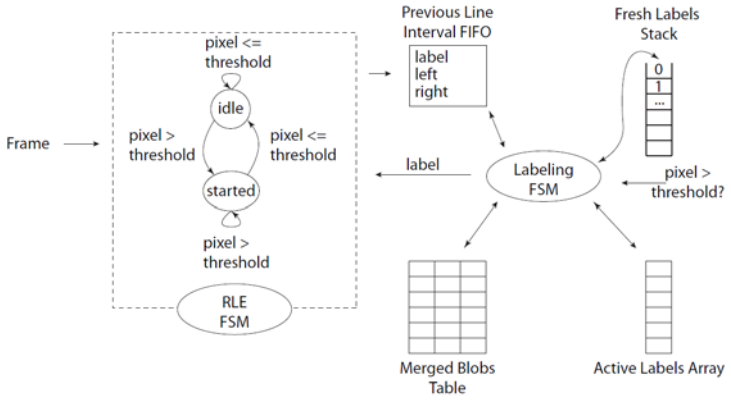


Figure 4.39: Connected component labelling

In some situations, it is necessary to merge two existing objects, because the current run connects them with each other. In such cases, a table that maps labels to other labels is used to store information about merges. It might happen that a run is a neighbour to several runs. In this case, all runs are merged, and for every merged run, an entry is noted in its corresponding merge table. When merging a BLOB that has been merged in the same line already, the merge table is looked up first, and the correct BLOB is read from the merger table. If both BLOBs have been merged, one of them will be chosen to contain the new merged BLOB. [10] Presented an approach where the merge table is kept in block RAM of the FPGA. Block RAM introduces a delay of one clock cycle for every read or write. Another constraint is that the

content of the memory cannot be accessed all at once, and must be accessed in sequential manner. Keeping the merger table in block RAM does not allow us to check two entries at once, which is resolved by processing all merges during the horizontal blanking interval of the camera. If the merge table is kept in the lookup tables of the FPGA, it is possible to randomly access the entries in the table without one clock cycle penalty, which is one of the differences of this approach to the approach developed in [10].

4.9.5 BLOBs Detection and BLOBs Analysis on an FPGA

Connected component labelling labels every foreground pixel with the number of the object to which it belongs. After labelling every foreground pixel, the objects can be analysed and their features extracted. In order to find the camera pose, it is necessary to know the coordinates of the BLOBs' centres. For this reason, the only feature that needs to be extracted is the centre of mass of each BLOB. It is possible to find the centre of gravity on the fly, while performing object labelling. This approach has the advantage that the image does not need to be stored in a memory for a second pass analysis. The centre of mass can be computed by summing up the coordinates of each pixel multiplied by their respective pixel values and normalized by the accumulated pixel values in the end. As discussed in [11], it is possible to compute additional features on the fly, however, in this project only the centres of mass are considered.

The synthesis tools for FPGA do not support division other than by numbers that are a power of 2 because it is too complex to be placed on an FPGA. This means that the circuit for performing division has to be either developed manually or included into the design as an intellectual property (IP). Altera provides a division IP, where the user can set the bit-width of input and output signals. Only two such circuits are needed, since at any point in the X/Y coordinates, only one BLOB can be completed two lines before. The required bit-width of input and outputs of the circuit can be calculated by assuming the worst case. The division IP core provided by Altera allows the user to select the bit width of the input signals and the output latency. For 64 bits output (43 bits for representation of mantissa and 11 bits for the power) it introduces at least 10 clock cycles delay, however, the division can be pipelined. For more detailed information see [31].

4.10 Camera Emulator Using an External (Additional) FPGA Board

In order to obtain ground-truth data necessary for the evaluation of the designed system, an additional camera emulator has been developed. This approach allows overcoming additional impacts of the test-related components. This problem occurred during different evaluation design mentioned in 4.8.8. It utilizes an additional external FPGA board, in our case an Altera DE2-70. It was planned to connect the original board with the MV camera to a second DE2-70 board. However, the design was changed and now the board with the camera is used to provide real and synthetic images to the second board, where the identification of regions of interest takes place. In the initial approach to emulate the camera, camera timings were captured with the Signal Tap Logic Analyser. Then, a controller was written to behave in the same way. However, this approach turned out to be inflexible when the evaluation was started. The problem is that there is no flexible way to change the frame rate to a desired value at run-time. Hence, the emulator has been redesigned to use the synchronization timings from the original camera. This allows the user to change the frame rate of the camera over the provided control panel, so that the synthetic images can be sent with the same timings as of the real camera. This approach is more flexible and allows for selecting any frame rate that the connected camera can provide. In addition, by flipping a switch, the user can choose between the original camera and synthetic images stored on the same board.

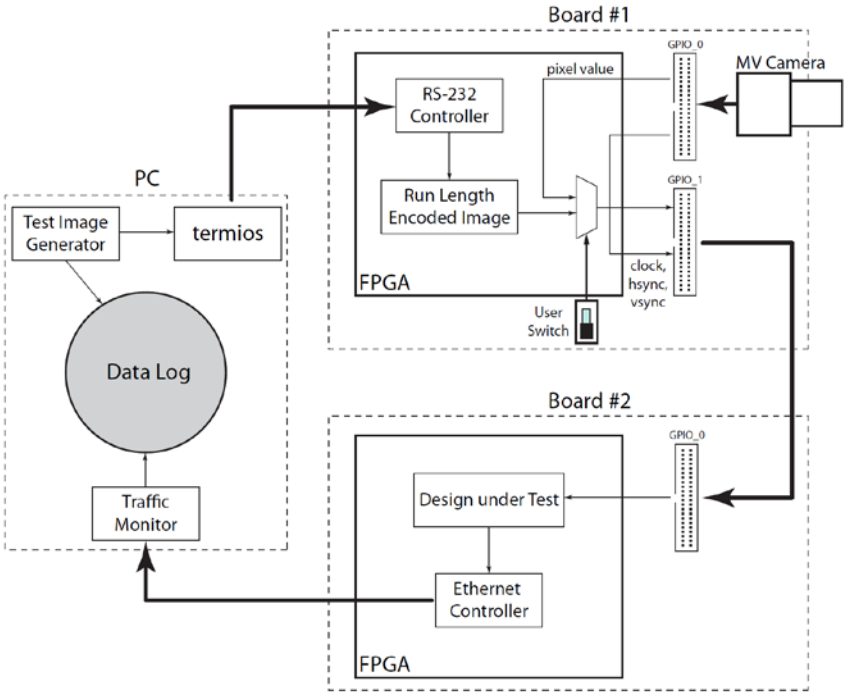


Figure 4.40: Camera emulation using an external FPGA board.

In a real-world application, such a scenario introduces an additional delay between the camera and the FPGA on which the image processing takes place. This is due to taking an indirect route of the camera data to the second board by going first to the FPGA of the board 1, then over the GPIO cable and then from the GPIO pins to the FPGA on the board 2. However, the delay is insignificantly small and plays no direct role in the scalability evaluation of the approach developed in this project.

4.10.1 Test Images Generator

In order to evaluate the approaches developed in this subproject systematically, the amount of foreground pixels must be controllable. However, it is difficult to control this amount when using the camera, because it requires camera calibration and exact positioning. These problems can be avoided if we use synthetic images instead. Synthetic images enable control over the amount of foreground pixels. For this purpose, a test image generator has been developed to generate test images with varying BLOB numbers and sizes.

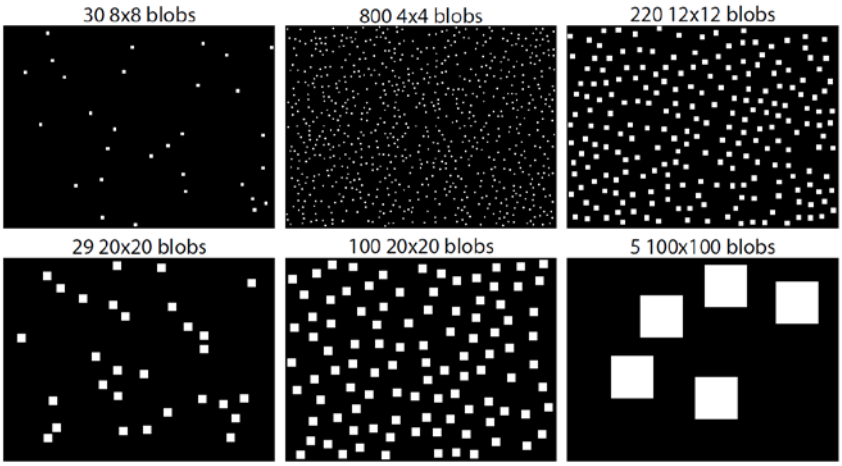


Figure 4.41: Sample images of the test images generator. All images are 8-bit greyscale with a resolution of 640x480.

The generator is written as a command line tool that accepts two arguments: size of BLOBs and the number of BLOBs. It tries to generate an image with these parameters and, upon successful generation, saves the image in PNG format under a unique name on the hard disk. In addition, it copies the name of generated image into the clipboard. In the next step, the main program compresses the image by using run-length encoding and sends the image to the DE2-70 board over the serial port. Figure 4.41 shows some sample test images generated for evaluation.

For more detailed information about the FPGA-based data traffic reduction approach described in Section 4.9 and the evaluation system based on camera emulator using an external FPGA board described in Section 4.10 please refer to [31].

5 Inside-Out Approach

In contrast to the previous chapter, this chapter presents the results of the project developments based on an inside-out approach with a fiducial marker system.

As it was discovered during early stages of the project an outside-in approach is limited by a number of hardware and software issues. Some of these problems were discussed in Section 2.2. In the outside-in configuration (according to the original project plan) of the system, the hand-held device consists of a laser beam emitter with splitter, 3DoF IMU and a power supply. This setup implies a set of cameras statically fixed behind the projection screens, one camera per projection surface (see Figure 5.1).

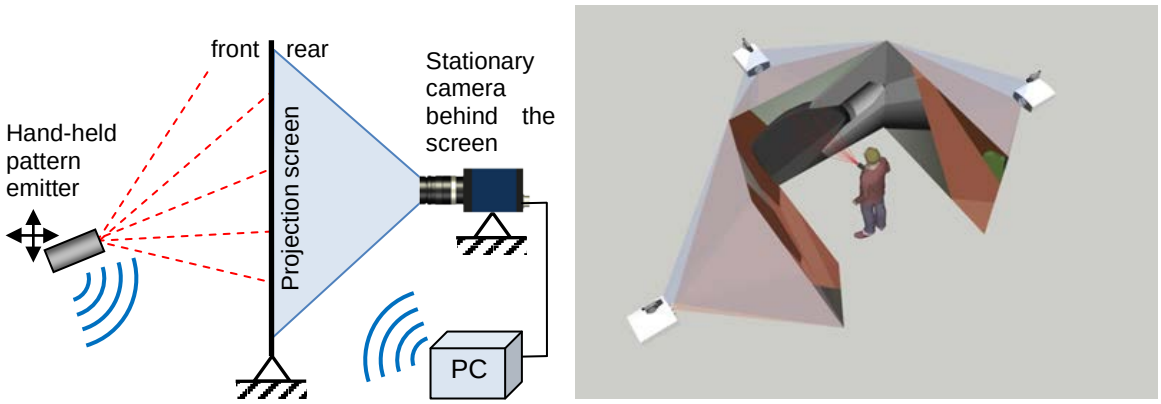


Figure 5.1: Outside-in configuration. Cameras are stationary and placed next to each projector (left, projector is not shown). The hand-held input device consists of a laser pointing device and an IMU with radio communication channel (left). Setup model of the virtual environment with 3 projection screens (right).

Taking into consideration all drawbacks of this setup, another approach was developed and investigated during the costs-neutral project extension. It relies on the so-called "inside-out" system configuration. This setup implies a hand-held device consisting of a camera (with or without on-board processing capabilities), a radio data transceiver for video data or pose coordinates transmission, an IMU and a power supply. The fixed part of the system consists of powerful pattern-projection modules (no limitations in power consumption due to stationary setup). Each projector projects an invisible (infrared) pattern to each of the visualization screens (see Figure 5.2).

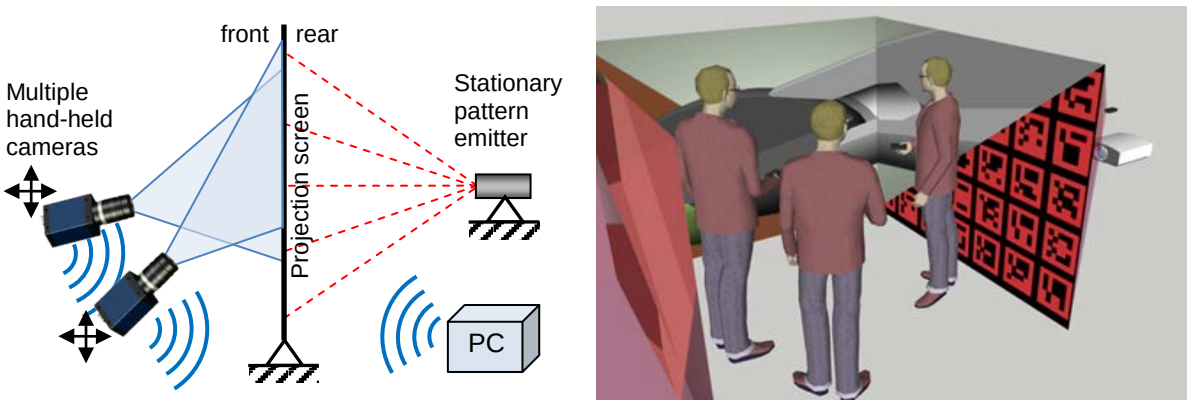


Figure 5.2: Inside-out configuration. Infrared pattern emitters are stationary and placed next to each projector (left, projector is not shown). The hand-held input device consists of a camera, an IMU and a radio communication channel (left). Setup model of a virtual environment with 3 projection screens (right).

Results from the early project designs (Section 4.6) showed problems with the identification of each individual marker inside the pattern. Since this information is required for the 3D pose reconstruction, the proper identification of the markers is a crucial step in the pose determination.

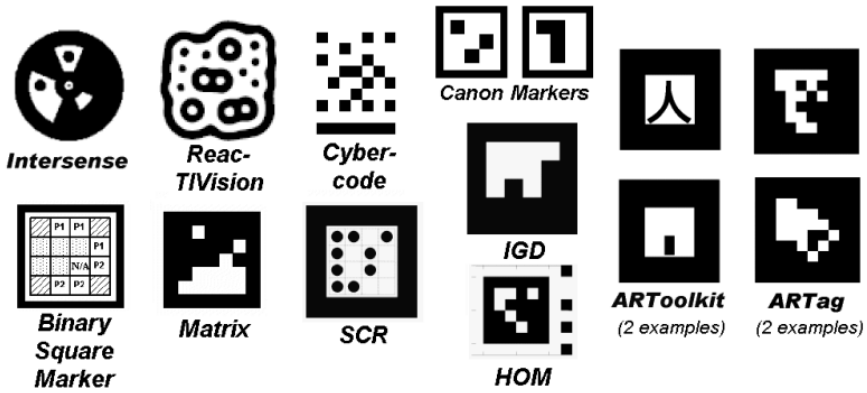


Figure 5.3: Fiducial markers used by different tracking systems.

Based on the evaluation of test implementations the decision to use one of the well-developed fiducial marker systems was made. Among many available systems (see Figure 5.3), the one, called "ArUco" [12], was selected. It is similar to the discontinued "ArTag" system with robust error correction capabilities. For more detailed information, please refer to [13], [14].

This new approach has its own limitations. In particular, the amount of video data that has to be transmitted over radio channels is considerably high. As the number of simultaneously operated input devices increases, the radio transmission bandwidth might become a limiting factor. Therefore, a smart approach in data traffic reduction using an on-board FPGA similar to the one described in Section 4.9 should be investigated.

5.1 Realization of the Selected Hardware Design for the Pattern Emitter

As discussed in Section 2.3, the pattern emitter based on the micro-mask projection principle was selected from a number of known designs as the one that fulfils the requirements of the project (see Section 2.3.5). A working prototype of a special version with an infrared light source using a static micro-mask approach has been built (see Figure 5.4). With its spectral output peaking at 850 nm, the projected pattern is not visible to the human eye and, therefore, doesn't produce any interference with virtual reality visualization. The projector has the following specifications:

- Wavelength: 850 nm
- Luminance: 1,45 mW/cm^2
- Electrical power consumption: 10 W
- Mask dimensions: 10.5 x 10.5 mm
- Mask resolution: 2200 x 2200 pixels

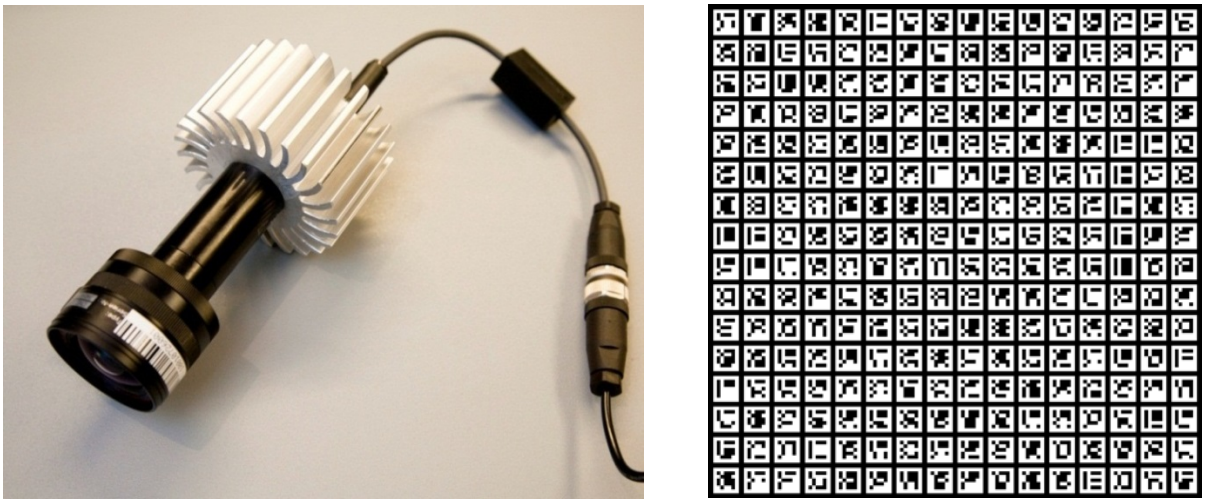


Figure 5.4: Left: Final design of the infrared pattern emitter based on the micro-mask approach. A large cooling heat-sink for the power LED and a C-mount lens are attached. Right: Fixed infrared pattern used in the emitter.

Designed to be fixed behind the projection screens, static focal distance, size and power consumption of the module do not pose any significant limitations for the project. The large heat-sink provides effective air cooling for continuous maintenance-free operation.

5.2 Acceleration of the Fiducial Marker Detection using an FPGA

5.2.1 System design

This section presents an approach for computing camera pose with the processing shared between FPGA and PC from the images of fiducial markers being projected with the developed static near-infrared projector (see Section 5.1). The task is divided between two systems as follows: the FPGA converts greyscale camera images into binary format and transfers them to a PC that computes the camera pose. The approach is evaluated in terms of processing time and camera pose error. A 3D simulator is used to generate a large number of test images in order to evaluate the proposed system. Synthetic images generated in the simulator undergo a set of transformations in order to make the evaluation closer to images obtained from a real camera. Several environmental conditions are simulated, e.g., blur with a Gaussian kernel of different sizes, and two types of noise. The results of evaluation show that the system can process camera images in real time.

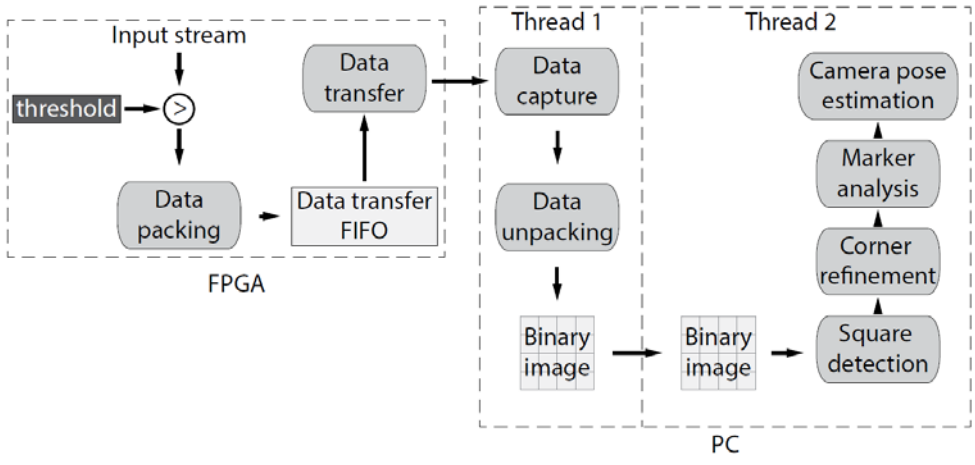


Figure 5.5: System design of the shared approach.

The system design is illustrated in Figure 5.5. On the FPGA, greyscale images captured by the on-board camera are converted into binary images by using a threshold. The resulting binary image is divided into a fixed number of packets, which are transferred to a PC one by one. The PC captures the data and converts back into a binary image. In the next step, the camera pose estimation methods are applied onto the binary image, and the camera pose estimated.

Thresholding is performed by comparing the intensities of all pixels in the camera image to a fixed threshold. Thresholding can be done on the fly without the need of a frame buffer.

5.2.2 Data Packing and Transfer

The amount of data to be sent to PC is known beforehand, so that it can be divided into a fixed number of parts. A 640 x 480 binary camera consists of 640 x 480 = 307200 bits and is divided into 75 equal intervals of 4096 bits each. The intervals are numbered from 0 to 74, and this number is sent together with the image bits to the PC. Numbering the packets in this way has the advantage that if the PC stops capturing data for a while, and the packets are not numbered, then there is no way to reconstruct the binary image once the capturing process has been started again. The number is converted into image coordinates, so that the place of every received bit is identified uniquely.

The bits of the binary image are grouped together into words of 16 bits and written into a FIFO that is used for data transfer by the Ethernet controller. Before packing 4096 bits of an interval together, the interval number is written into the data transfer FIFO. 16 bits are allocated for sending the interval number.

Data is transferred from FPGA to PC by using the Ethernet protocol. First, the 6 x 8 bits of the destination MAC address is written, then the MAC address of the receiver. Subsequently, image data and the interval numbers are transferred. Thus each Ethernet packet consists of 2 x 6 x 8 bits allocated for the MAC addresses, 16 bits for sending the interval number, and 4096 bits for sending the binary image data. Some additional data is added by the hardware, such as the 8 bytes preamble, and 4 bytes of checksum to detect errors in the Ethernet frame. Altogether, each packet is:

2 x 6 x 8 + 16 + 4096 + 8 x 8 + 4 x 8 = 4304 bits or 538 bytes.

Assuming that the camera captures images at 100 frames per second, the expected traffic of this approach is:

4304 x 75 x 100 = 32‘280‘000 bits per second, or ~32 Mbits/s.

5.2.3 Data Capture and Camera Pose Estimation on PC

On the PC side, the images are captured by using the Berkeley packet filter, which allows low-level access to the network interface. To detect packets sent from the FPGA, the sender and receiver MAC addresses of all packets are inspected. Packets with the right addresses are considered for further processing. The data is unpacked by extracting the interval number and the 4096 image data bits.

To allow the utilization of multicore processors, data capture process and the camera pose estimation run in different POSIX threads. To communicate between the threads, shared data guarded by a mutex is used. Upon reception of a complete binary image, a shared status variable is set to high, so that the camera pose estimation thread can copy the binary image into its own memory space and start the process of camera pose estimation. The image processing pipeline for detection ArUco markers on the PC is provided by the ArUco library that is written in C++ on top of the OpenCV library. More information about fiducial marker detection using an FPGA can be found in [32].

5.3 Ground Truth Evaluation of a Vision-based 6DoF Input Device Using a Robotic Arm

A framework for ground truth based evaluation of a vision-based 6DoF input device using a robotic arm was developed and realized.

The system consists of a wired CCD camera as the only sensor and multiple unique markers. The markers are placed into a video image and projected within a scene using a projector. All testing procedures were conducted in the visible spectrum (as opposed to near-infrared one) using available projectors inside the Immersion Square visualization system [2]. That allowed rapid prototype development with a fast error correction cycle. A visible marker system is easy to inspect for possible errors. Nevertheless, the system still needs to be tested with a new infrared pattern emitter.

The framework for evaluation of our camera-based tracking system utilizes a visualization system as well as 5DoF computer-controlled robotic arm and an industrial optical tracking system. The robot was used for systematic camera displacements and rotations, while the optical tracking system was tracking the pose of the robot base. Both contributed to provide ground truth data of the 3D pose of the device to be evaluated.

The framework is capable of collecting static image sequences at pre-programmed 3D positions. The data analysis part is done manually afterwards, which is one of the limitations of the system so far. Due to the robotic arm that cannot be controlled dynamically (only point-to-point motion control is available), the motion pattern is limited and collected images are static. Besides evaluation of our implemented marker-based tracking system, the professional optical tracking system ("OptiTrack") has been partially evaluated as well.

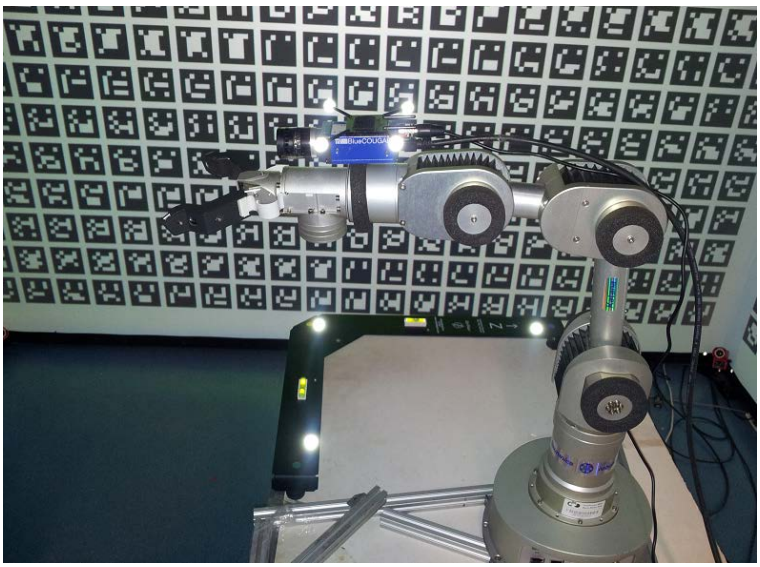


Figure 5.6: System evaluation using the robotic arm "Katana" and the optical motion tracking system "OptiTrack". The red small camera in the right part of the image is one of the eight "OptiTrack" cameras. The bright spots in the image are retro-reflective markers used as a reference and tracking bodies for "OptiTrack". The fiducial marker system "ArUco" is visible in the background being projected onto the visualization surfaces. The blue rectangle in the top part of the image is an "mvBlueCOUGAR" camera from Matrix Vision used as an input device being evaluated.

The evaluation framework consists of three major systems: an optical tracking system, a robotic arm and a camera system (visual tracking system under test) mounted on a robotic arm's gripper, see Figure 5.6.

The approach used in this subproject is that the robotic arm's gripper will carry the camera system attached with markers. The optical tracking system will track these markers to calculate the pose of the inside-out camera system. The inside-out camera system will use the projected images on the three

screens with patterns on them to estimate the camera pose. Movement of the robotic arm will provide the necessary ground truth data.

The robotic arm will move the camera attached to its gripper in point-to-point trajectories in the arm's workspace. Pose information will be available from three sources: the optical tracking system, the camera system and from the robotic arm's internal encoders. Differential errors between the two pose estimates reported by the robotic arm will be used to calculate accuracy, since absolute pose values are difficult to obtain without introducing large (inacceptable) errors.

The robotic arm's workspace is limited because of limited degrees of freedom and its overall small size relative to the visualization environment. The position of the base of the arm, taken from the optical tracking system is used for visualization purposes and does not participate in error calculations. This measurement is only done to get a rough idea about the position and orientation of the base of the arm relative to the visualization system and it is also used for error comparison between several 3D point clouds.

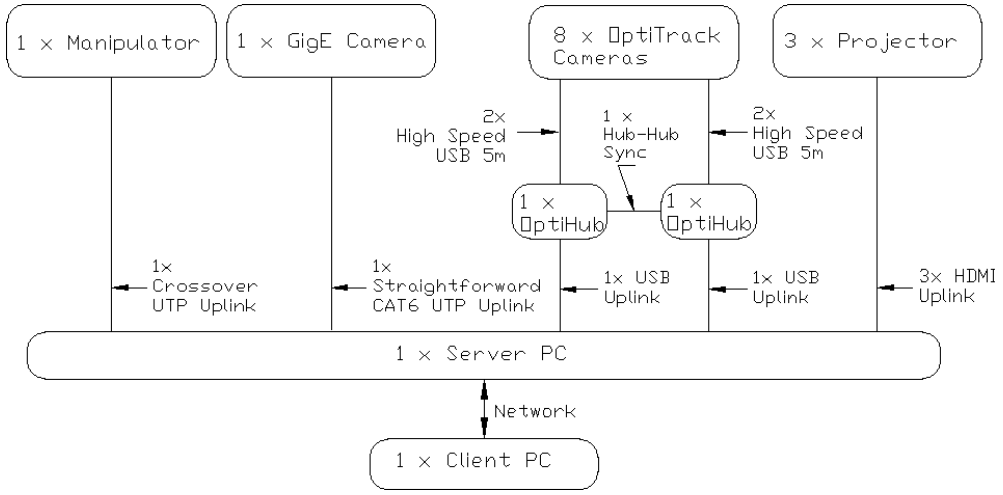


Figure 5.7: Evaluation system's overview.

All testing and evaluation experiments were conducted within the full scale CAVE-type visualization environment "Immersion Square". The system was equipped with a PC, OptiTrack cameras and projectors. The configuration of the hardware is shown in Figure 5.7.

5.3.1 Software and Overall System Architecture

This section describes the software and system architecture of the framework based on a component-based software design model. The architecture of the framework has been designed to reduce the complexity and provide systematic and synchronized data acquisition from the hardware. The system consists of the components shown in Figure 5.8.

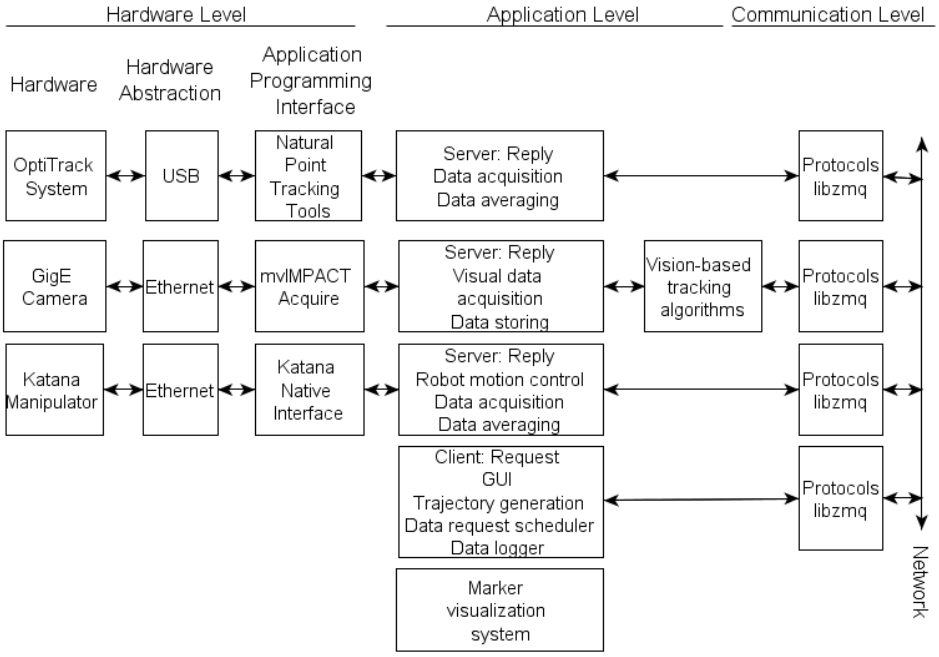


Figure 5.8: Evaluation framework architecture.

The components are categorized into three levels: hardware level, application level and communication level. Hardware level components are provided by the manufacturers. The components in the application and communication level are implemented using the C++ programming language under Visual Studio. The C# programming language is used for marker visualization system.

An overview of the coordinate systems used in the project, sequences of transformation steps necessary to obtain final transformation matrices from the camera's local coordinate frame to the global one and transformations from the tool's local coordinate frame to the global one are shown in the Figure 5.9.

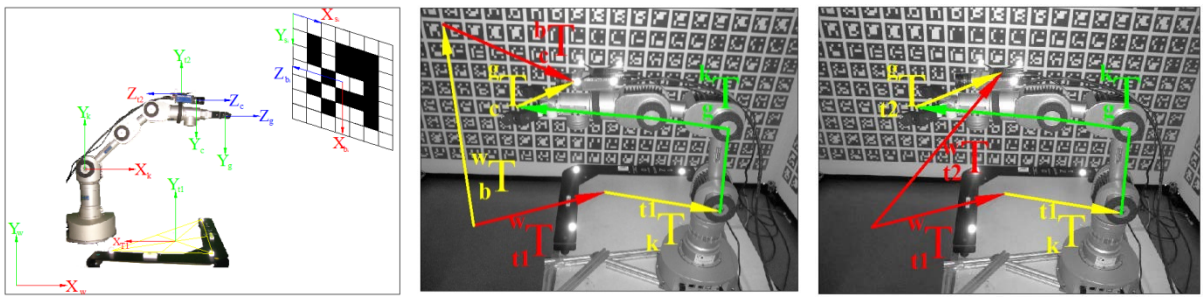


Figure 5.9: An overview of the system's coordinate frames (left). Transformations from the camera's local coordinate frame to global coordinate frame (centre). Transformations from tool's local coordinate frame to global coordinate frame (right).

5.3.2 Robot Motion Control and Sampling Strategy

The manipulator's workspace was relatively small compared to the available space in the Immersion Square Environment (Figure 5.10), and it was necessary to locate the robot at different locations in the Immersion Square in order to execute the motion pattern (Figure 5.11, Figure 5.12). The optical tracking system was useful to relate the movements made by the robot.

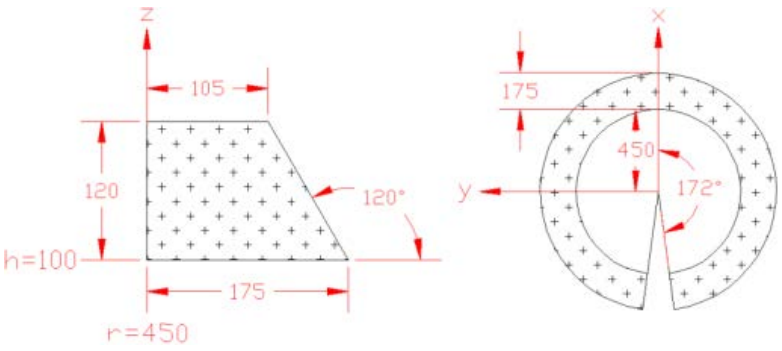


Figure 5.10: The robot's working volume configuration.

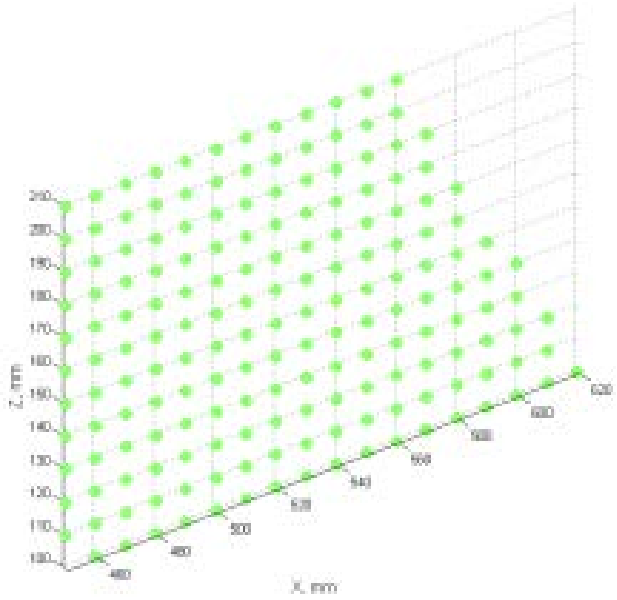


Figure 5.11: Sample scan within the XZ-plane, step: 10mm.

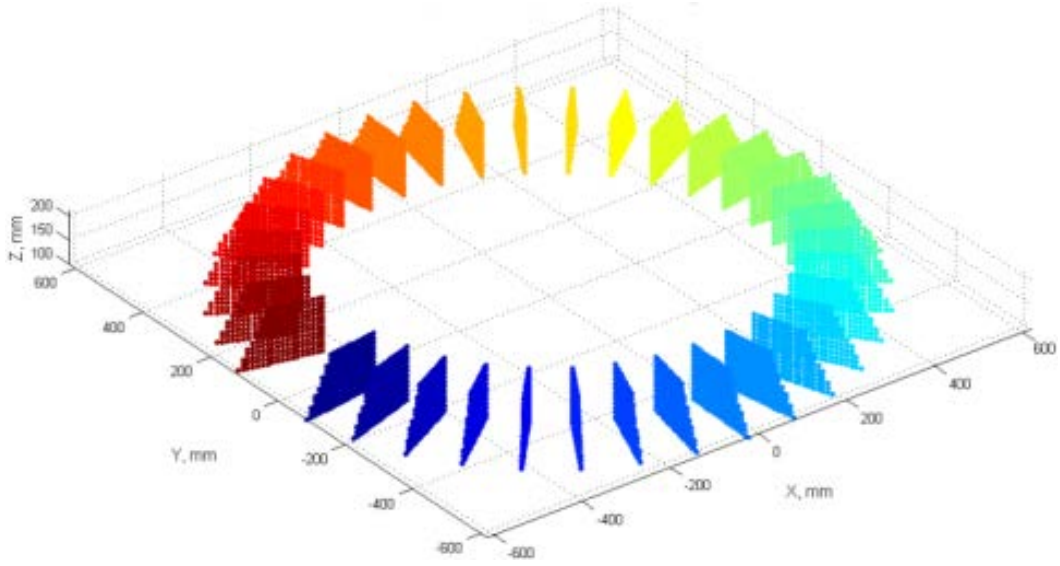


Figure 5.12: Sample scan, steps: 10 mm, 10 deg. Angle change is colour-coded (right).

5.3.3 Experimental Results of the Visual Tracking System Evaluation

The objective of the experimental environment described above is to evaluate implemented inside-out visual tracking system with respect to angle and distance. For this purpose, a single screen is enough to simplify the evaluation process. Camera position and orientation relative to the gripper were required in order to relate static images taken from the camera with the motion pattern made by the computer-controlled robotic arm. The existing "camera-hand-eye" calibration methods were not useful for the implemented setup due to the constraints of the robotic arm. It was observed that the results were worse than human hand precision, having ±15 mm under good conditions. Therefore, differential data analysis is preferred rather than absolute data analysis. Considering the uncertainty of the hand-eye-calibration and the physical constraints of the robotic-arm, two motion patters have been used to acquire ground truth.

Differential positional error estimation: The robotic arm makes 1 mm incremental sequences of movement of the camera in a single line nearly perpendicular to the screen. In each increment, the differential error is calculated and analysed. This procedure is repeated for 13 locations in the Immersion Square. Figure 5.13 (left) shows the datasets from 1 to 13 for different locations of the camera in the virtual environment. The error distribution is shown in Figure 5.13 (right) for a combination of the datasets. Execution time is around 60ms and distribution of it is given in the Figure 5.14 (right). It is observed that the execution time is independent of camera distance from the screen.

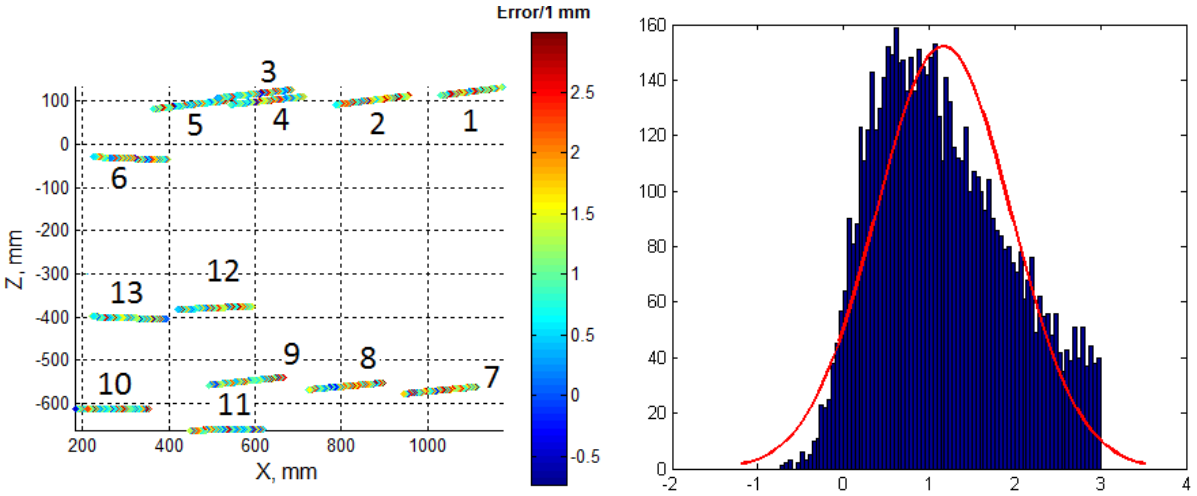


Figure 5.13: Differential position error position/error in mm (colour coded, left). Differential error distribution, samples / mm (right).

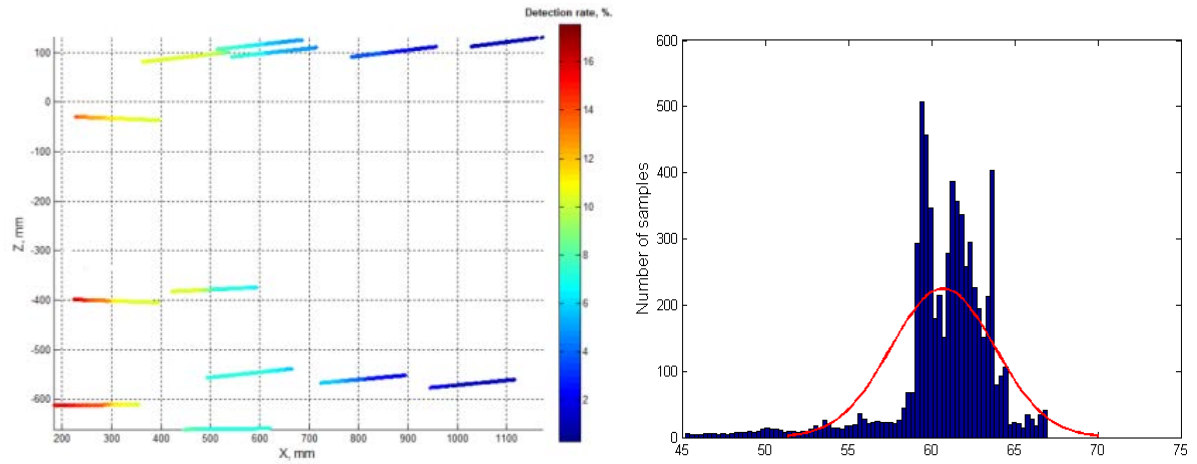


Figure 5.14: Rate of samples with no position detection results in % (colour-coded, left). Execution time distribution in ms (right).

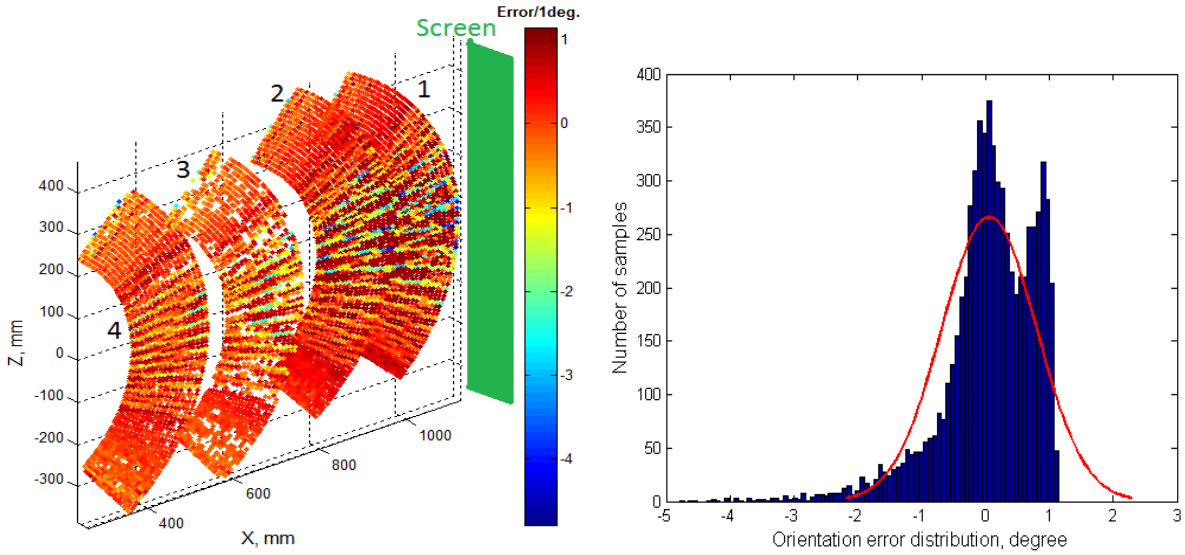


Figure 5.15: Differential rotational error position/error in degrees (colour coded, left). Differential error distribution, samples / degree (right).

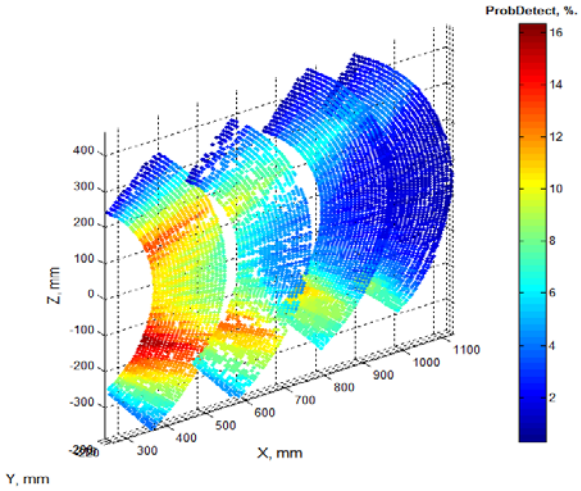


Figure 5.16: Rate of samples with no result in orientation or "missed" samples in % (colour coded).

Differential orientation error estimation: The robotic arm makes rotational sequences of the movement with the camera along multiple axes forming a plane parallel to the ground. The rotational resolution was chosen at 1°. The robotic arm is scanned from -60° to +60° and similar data were analysed.

The overall error distribution is shown in Figure 5.15 (right) having mean error of 0.05° and variance of 0.72°. Figure 5.15 right, shows the probability of non-detection rate which increases as the camera moves away from the screen.

The implemented setup provides several advantages: The manipulator is affordable compared to the coordinate measurement machines. Displaying textures is rather easy using the projector systems. The robot and the marker visualization system can be automated for dataset design while displaying the textures continuously. Nevertheless, the robotic arm is not capable of following a trajectory linearly. Moving speed and acceleration of the motion cannot be controlled properly. Therefore, this framework should not be applied for the evaluation of tracking systems based on inertial sensors. However, it is useful for the evaluation of tracking systems based on fiducial markers. More detailed information about this subproject can be found in [30].

6 Conclusions and Future Work

It has been shown that a number of methodological non-trivial sub-problems related to the general objective of the project could be solved on a scientific level during the course of the project [16] - [21], [22] - [32]. The developed tracking approaches [16], [18], [22], [27] are of general use as well as the blob detection methods applying highly parallel FPGA technology [17], [20], [23], [24], [26], [31]. The application of a parallel processing architecture for the blob detection ensured a high throughput of data, which is needed for real-time processing of multiple video streams without causing latencies higher than acceptable for dynamic interactions (few milliseconds).

A sensor data fusion approach has been developed and realized for a reliable 3D rotation estimation using an IMU and optical system data. Moreover, a quaternion-based sensor fusion algorithm has been developed using spherical linear interpolation [28] which has shown promising results.

The outside-in approach led to a numerous scientific results, but was rejected as the final solution for the project due to multiple limitations (see Section 5). The most significant limitation of this approach was an increase in computational load (and as result decrease in the system's performance) with an increase in image noise level and increase in undesired deviations of the dots in the pattern (Section 4.6.8). Despite being very efficient and stable in the simulated environment, in real-world scenarios, these shortcomings led to a degraded performance of the correspondence matching sub-system (Section 4.6.6). This caused latency, which rendered the entire system too slow and unresponsive, since end-to-end latencies of the entire interaction device were not kept within necessary ranges to ensure intuitive usability of the system and flawless interaction process.

These problems combined with difficulties in building an efficient and convenient pattern projector (see Section 2.3) led to a new, creative solution: an inside-out approach. As a result an inside-out configuration of the system (see Section 5) was implemented and evaluated.

A completely functional prototype fulfilling the initial projects' requirements was developed and evaluated shortly after the project's end [32]. This final and working solution utilizes the inside-out approach with fiducial markers (Section 5), a back-projected infra-red pattern using IR static projector (Section 2.3.5, Section 5.1). On the user-side, a camera integrated into the FPGA board (Section 4.8.1) was used. A system was able to acquire a complete 6 DoF pose of the input device under control of the hybrid software/hardware design located on the FPGA board and on the standard PC (Section 5.2).

In order to evaluate the system's precision and accuracy a framework for the ground truth evaluation of vision-based 6-DoF input devices using a robotic arm has been established [30]. Further developments in this area are still in progress and show the sustainability of the project [21], [32].

7 References and Publications

[1] C. Celozzi, G. Paravati, A. Sanna and F. Lamberti, "A 6-DOF Artag-based tracking system," *IEEE Transactions on Consumer Electronics,* vol. 56, no. 1, pp. 203-210, February 2010.

[2] R. Herpers, F. Hetmann and A. Hau, "The Immersion Square – A mobile platform for immersive visualizations," in *Aktuelle Methoden der Laser- und Medizinphysik, 2. Remagener Physiktage*, Remagen, Germany, 2005.

[3] D. Willis, "SideBySide: Ad-hoc Multi-user Interaction," in *UIST'11*, Santa Barbara, CA, USA, October 2011.

[4] K. Levenberg, "A method for the solution of certain problems in least squares," *Quart. Applied Math,* vol. 2., pp. 164-168, 1944.

[5] P. Meer, S. Ramakrishna and R. Lenz, "Correspondence of Coplanar Features Through P2-Invariant Representations," in *Proceedings of the Second Joint European–US Workshop on Applications of Invariance in Computer Vision*, London, UK, 1993.

[6] J. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM,* vol. 18, no. 9, p. 509–517, 1975.

[7] S. Kirkpatrick, C. Gelatt and M. Vecchi, ""Optimization by Simulated Annealing," *Science New Series,* vol. 220, no. (4598), p. 671–680, 1983.

[8] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transaction of the ASME, Journal of Basic Engineering,* p. 35–45, 1960.

[9] C. Johnston and D. Bailey, "FPGA implementation of a single pass connected components algorithm," in *Electronic Design, Test and Applications, 4th IEEE International Symposium on*, Jan. 2008.

[10] D. Bailey and C. Johnston, "Single pass connected components analysis," in *in Proceedings of Image and Vision Computing New Zealand*, December 2007.

[11] D. Bailey, Blob Detection and Labelling, Wiley-IEEE Press, 2011.

[12] R. Munoz-Salinas, "ArUco: a minimal library for Augmented Reality applications based on OpenCV," 2012, <http://www.uco.es/investiga/grupos/ava/node/26>.

[13] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol.2, pp. 590- 596 vol. 2*, June 2005.

[14] M. Fiala, "Designing Highly Reliable Fiducial Markers," in *IEEE Trans. Pattern Anal. Mach. Intell. 32, 7, 1317-1324*, July 2010.

[15] M. Sieler, P. Schreiber, P. Dannberg, A. Bräuer and A. Tünnermann, "Ultraslim fixed pattern projectors with inherent homogenization of illumination," *Appl. Opt.,* vol. 51, pp. 64-74, 2012.

[16] M. Vieth, R. Herpers, M. Huelke and D. Chabbra, "A Low-Cost Laser Based 6 DoF Head Tracker for Usability Application Studies in Virtual Environments," in *5th INTUITION International Conference*, Turin, Italy, October 2008.

[17] A. Bochem, K. Kent and R. Herpers, "Hardware Acceleration of BLOB Detection for Image Processing," in *The Third International Conference on Advances in Circuits, Electronics and Micro-electronics*, Venice, Italy, July 2010.

[18] D. Scherfgen, R. Herpers and T. Saitov, "Single Webcam 3D Tracking for Video Games," in *Future Play*, Vancouver, Canada, 2010.

[19] D. Scherfgen, T. Saitov, R. Herpers and E. Dayangac, "An optical laser-based user interaction system for cave-type virtual reality environments," in *4th Russian-German Workshop "Innovation Information Technologies: Theory and Practice"*, Ufa, Russia, April 2011.

[20] A. Bochem, K. Kent and R. Herpers, "FPGA based Real-Time Object Detection Approach with Validation of Precision and Performance," in *Proceedings IEEE International Symposium on Rapid System Prototyping*, Karlsruhe, Germany, May 2011.

[21] P. Samarin, T. Saitov, R. Herpers and K. Kent, "Double Thresholding in Hardware to Enable Flexible Blob Detection for a Vision System with Limited Bandwidth," in *The Fourth International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies (HEART2013)*, Edinburgh, Scotland, 2013.

[22] T. Saitov, "Visual Multi-Camera Tracking System for Low-Cost Immersive Visualization Environment," Master's thesis, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, February 2009.

[23] A. Bochem, R. Herpers and K. Kent, "Acceleration of Blob Detection within Images in Hardware," Technical report 09-197, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, December 2009, <http://www.cs.unb.ca/tech-reports/documents/TR09_197.pdf>.

[24] J. Sommer, "Connected Component Labelling on an Intelligent Camera with Embedded CPU," R&D project report, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, January 2010.

[25] T. Hofhammer, "Simulation of a Hand-held Projector for generation of Points Pattern Projection in Immersion Square," Bachelor Thesis, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, 2010.

[26] A. Bochem, R. Herpers and K. Kent, "Acceleration of Blob Detection in a Video Stream using Hardware," Technical report 10-201, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, May 2010, <http://www.cs.unb.ca/tech-reports/documents/TR_10_201.pdf>.

[27] A. Bochem, "Active Tracking with Accelerated Image Processing in Hardware," Master Thesis, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, December 2010.

[28] E. Dayangac, "Inertial Measurement Unit Data Acquisition, Analysis and Merge into MI6 Optical System Output," R&D2 project report, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, March 2011.

[29] F. Rouatbi, "6DoF Detection for Virtual Environment Interactions Using Laser Pointer Devices and Cameras," R&D project report, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, March 2011.

[30] E. Dayangac, "Vision-based 6DoF Input Device Development with Ground Truth Evaluation," Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, April 2012.

[31] P. Samarin, R. Herpers, K. Kent and T. Saitov, "Evaluation of data transfer from FPGA to PC: Increasing frame rate by BLOB detection," Technical report 12-217, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, June 2012.

[32] P. Samarin, "Using fiducial markers for precise camera pose calculation on FPGAs," Master's thesis, Department of Computer Science, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, June 2013.

Appendix A: State of the Art

A.1: "Outside-in" Approach

Six Degrees of Freedom:

1. Vorozcovs, A. Hogue, W. Stuerzlinger, The Hedgehog: A Novel Optical Tracking Method for Spatially Immersive Displays, In Proc. IEEE Virtual Reality 2005, 83-89.
2. C. Wienss, I. Nikitin, G. Goebbels, K. Troche, M. Göbel, L. Nikitina, and S. Müller. Sceptre: an infrared laser tracking system for virtual environments. In VRST '06: Proceedings of the ACM symposium on Virtual reality software and technology, pages 45–50, New York, NY, USA, 2006. ACM Press.
3. Latoschik M.E., Bomberg E. Augmenting a Laser Pointer with a Diffraction Grating for Monoscopic 6DOF Detection. Journal of Virtual Reality and Broadcasting, Volume 4, no. 14, 2007

Four Degrees of Freedom:

4. Matveyev S., Göbel M., Frolov P. Laser Pointer Interaction with Hand Tremor Elimination, The proceedings of HCI International, Greece,"Human-Computer Interaction: Theory and Practice" Vol.2, pp. 736-740., 2003.
5. S. V. Matveyev and M. Göbel. Direct interaction based on a two point Laser pointer technique. In SIGGRAPH '03: ACM SIGGRAPH 2003 Sketches & Applications, pages 1–1, New York, NY, USA, 2003. ACM Press.
   S. V. Matveyev and M. Göbel. The optical tweezers: multiple-point Interaction technique. In VRST '03: Proceedings of the ACM symposium on Virtual reality software and technology, pages 184–187, New York, NY, USA, 2003. ACM Press.

Two Degrees of Freedom:

6. D. Cavens, F. Vogt, S. Fels, and M. Meitner. Interacting with the big screen: pointers to ponder. In Proceedings of ACM CHI 2002 Conference on Human Factors in Computing Systems, volume 2 of Interactive Posters, pages 678–679, 2002.
7. K. Cheng and K. Pulo. Direct interaction with large-scale display systems using infrared laser tracking devices. In CRPITS '24: Proceedings of the Australian symposium on Information visualisation, pages 67–74, 2003.
8. B. A. Ahlborn, D. Thompson, O. Kreylos, B. Hamann, and O. G. Staadt. A practical system for laser pointer interaction on large displays. In Proceedings of the ACM symposium on Virtual reality software and technology, VRST '05, pages 106–109, New York, NY, USA, 2005. ACM.
9. X. Bi, Y. Shi, X. Chen, and P. Xiang. uPen: laser-based, personalized, multi-user interaction on large displays. In MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, pages 1049–1050, New York, NY, USA, 2005. ACM Press

A.2: "Inside-out" Approach

1. C. Celozzi, G. Paravati, A. Sanna and F. Lamberti, "A 6-DOF ArTag-based tracking system," IEEE Transactions on Consumer Electronics, vol. 56, no. 1, pp. 203-210, February 2010.

A.3: Other Related Approaches

2. A. Masselli, "Low-Cost 3D Controller with Intuitive Handling", Eberhard-Karls-Universität Tübingen, Wilhelm-Schickard-Institut für Informatik, Tübingen, Germany <http://www.uni-tuebingen.de/uploads/media/EKUT-0239_Tech_Offer_02.pdf>