ulm university universität uulm

Ulm University | 89069 Ulm | Germany

Faculty of Engineering
and Computer Science
Institute of Media Informatics

A Novel Projector Camera System for Effortless Domestic Deployment

Master Thesis
Ulm University

Pascal Knierim
pascal.knierim@uni-ulm.de

Reviewers:
Prof. Dr. Enrico Rukzio
Prof. Dr. Michael Weber

Consultant:
Prof. Dr. Enrico Rukzio

2014

‘A Novel Projector Camera System for Effortless Domestic Deployment’
6th May 2014

Typesetting: PDF-LaTeX 2_ε
Graphics created with: TikZ
Printed by: Kommunikations- und Informationszentrum (kiz), University of Ulm

Abstract

Providing instant access to information when and wherever it is desired is one of the aims of the ubiquitous computing research area. To accomplish omnipresent visualisation of information, display technology needs to be present everywhere. As modern technology is not yet mature enough for truly ubiquitous visualisation, researchers instead emulate the rendering of information on any surface by use of a projector camera system (PROCAMS). These devices consist of a projector to render content onto any surface, and a camera to obtain user interaction.

This thesis proposes a novel PROCAMS for effortless domestic deployment. Its key features are a user-friendly interface, automatic calibration and the stand-alone character of the unit, as well as the ability to rotate along two axes, providing two degrees of freedom. As the proposed system is ceiling-mounted, it is able to transform every surface visible from this spot into a projected touch screen. Content is encapsulated into widgets which are pre-warped for a rectified presentation to the user. Rapid development of widgets is accomplished by decoupling the complexity of spatially and geometrically aware projection from the widget. Hence touch detection is also dissociated, interaction cues are injected into widgets.

The objective of this thesis is to integrate results of current research with several new techniques and small inexpensive hardware to create a small, rotatable, calibration-free projector camera system. This yields an effortlessly deployable prototype which for the first time allows the evaluation of new interaction concepts and everywhere information projection directly in users' well-known environments; their homes.

Everything not saved will be lost.

—Nintendo Quit Screen

Contents

1 Introduction

Ubiquitous computing is almost “imperceptible, but everywhere around us,” Weiser said already in 1991 [Wei91]. So far we achieve ubiquitous computing through carrying sophisticated mobile devices in our pockets that contain our whole digital life, rather than by the ubiquity of computing infrastructure. We are still dependent on the physical persistence of devices in our environment such as laptops or smart phones. To achieve ubiquitous computing, technology needs to recede into the background. Small, inexpensive processing devices distributed everywhere should provide location independent computational services. Therefore, an integration of human factors, software and hardware engineering, as well as social science needs to be accomplished.

One critical question in ubiquitous computing is, how to visualise information and enable user interaction beyond the traditional forms of smart phones, tablets, and desktop computing. Currently no technology is capable of enabling integrated information presentation on any location in a domestic environment. In order to investigate novel interaction concepts and information visualisation in a ubiquitous environment, researchers use projectors to render information onto any physical object or surface. In combination with a camera, user input can be captured. Summarised under the term projector camera system (PROCAMS), many research groups have proposed several basic approaches to facilitate everywhere projection. With the developed systems, every flat surface can be transformed into projected touch screens. However, proposed PROCAMS lack mobility, are bulky, and need large powerful workstations. They often require a complex calibration or even an instrumentation of the environment. Heretofore, systems are always located in laboratories and studies could never be conducted in a domestic environment. To the best of my knowledge no self-calibrating, rotatable, stand-alone PROCAMS was published so far.

In this thesis, a novel PROCAMS is designed, built and evaluated. The requirements are derived from preliminary interviews with potential end users. Additionally, ideas and approaches from many previously proposed systems are drawn together. Key features of the proposed systems are the user friendliness and lack of calibration tasks. There is no need for the end user to execute any calibration tasks. Furthermore, the PROCAMS is a stand-alone unit. In detail, one small box contains all hardware. The typically used large workstation is replaced by a small single board computer (SBC), integrated directly into the system. The complete system could be ceiling-mounted and is rotatable in two degrees of freedom (DOF). This allows it to render information to every spot visible from the suspension point of the unit.

The implemented software framework enables rapid application development. The complexity of spatially and geometrically aware projection, as well as the analysis of user input is completely decoupled from the application development. Touch interaction is converted into abstract events which can be handled in a generic way. The designed platform consists of a hardware and software component, which are perfectly tailored to each other to ensure smooth, efficient operation.

The objectives of this thesis are to integrate the new results of the current research in this field, combining these new outcomes with severals new techniques and small inexpensive hardware to create a small, rotatable, calibration-free projector camera system. This yields an user-friendly prototype which allows for the first time to explore how we deal with everywhere available computing and presentation of information as well as new interaction concepts directly in users' well-known environments; their homes.

1.1 Vision

In a few years provided that technology continues to progress at its current rate, we could have large interactive displays everywhere. We could then easily access information and our digital life independently of our current location. Information would be available without the need for any personal physical device. Microsoft described this vision in a concept video[1] that shows how people will get things done at work, at home, and on the go in a world full of ubiquitous displays.

Innovation and development of new technologies will allow us to equip our home with large interactive screens everywhere. For example the work surface in the kitchen, dining room table or the mirror in the bathroom to name a few. Every screen will be interactive and will provide all the information we want wherever it is desired. Digital cooking instructions right beside the oven, virtual board games or a remote for any technical device could become reality. However, this vision is not limited to our homes or workplace. Cars or public transport as well as bars and restaurants, can be equipped with this future technology and will provide a new way of interacting, sharing and grasping information.

[1] www.microsoft.com/office/vision

Exploring the design space with the help of PROCAMS already today will reveal important findings for future systems. This will lead to more mature user interfaces and interaction concepts together with an adequate visualisation when technology makes wide spread deployment possible.

1.2 Overview

This thesis starts with a brief discussion of related work. Different types of PROCAMS including fixed as well as wearable and mobile installations, are compared and discussed in chapter 2. In chapter 3 the interviews conducted for the requirements engineering are described and the findings are discussed in detail. Requirements of the PROCAMS derived from the interviews are acquired in the following chapter. In chapter 4, a generic software and hardware architecture design is presented which fulfils the requirements. In the next two chapters, the hardware construction and software framework implementation is described in particular. The built prototype was evaluated in a technical laboratory study. The setup and findings are discussed in chapter 8. Chapter 9 summarises the work and gives an outlook on upcoming challenges and future work.

The software framework and Android App, the engineering drawing of the designed hardware parts and the construction plan as well as the gathered data from the interviews including the images and a digital copy of this thesis can be downloaded at GitHub:

Source Code `https://github.com/uUlmKnierim/everywhereDisplaySoftware`
Remote App `https://github.com/uUlmKnierim/everywhereDisplayRemote`
Hardware `https://github.com/uUlmKnierim/everywhereDisplayHardware`
Thesis and Interviews `https://github.com/uUlmKnierim/master-thesis`

For the matter of archiving, the version of the thesis at submission date is tagged with *thesis* in each repository.

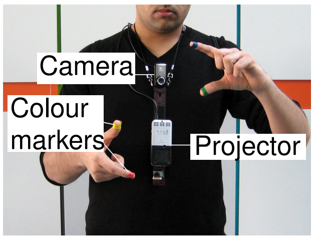
2 Related Work

Presenting content on a large surface is typically done by an ordinary projector. Over the last decade various ideas have been developed to extend the capabilities and integrate projectors into more sophisticated systems. Potential systems range from simple keystone correction over mobile projection to fully environment sensing systems which are capable of augmenting recognised objects with an overlay. The human computer interaction (HCI) community has done research on various types of user interaction, especially on augmented environment interactive projection interfaces. Following, a brief overview is given over several wearable or handheld as well as mounted systems which are correlated to this thesis. Additionally frameworks enabling rapid prototyping for mobile projector interaction techniques or for development of interactive projection based displays are examined.

2.1 Wearable and Handheld Systems

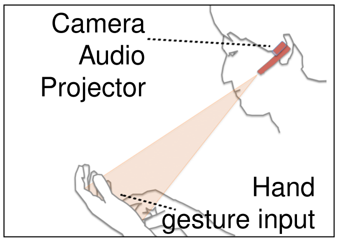
The research of wearable and handheld projector camera systems (PROCAMSs) focus in particular on mobile interaction concepts and the development of small units providing information access everywhere we go.

Such PROCAMS could be attached to different positions. For example shoulder, hip, back of the hand or the head. McFarlane and Wilder [MW09] analysed several of theses possibilities to minimise situational impairments since it can affect primary task like using the phone while driving. They developed *Interactive Dirt*, a simple PROCAMS from stock hardware under the aspect to minimise negative effects to situational awareness and enable easy interaction. The systems performance is evaluated in a field study with military background. In a semi-structured usability trial the left shoulder is discovered as the positions with best characteristic for supporting mobile teamwork. Challenges discovered in this trails are the lack of uniform surfaces in non urban environments as well as the brighter ambient lighting. McFarlane and Wilder named instant access to a large display while hands free one as one of the biggest benefits of a PROCAMS.



(a) SixthSense hardware

(b) Augmenting physical objects with SixthSense



(c) Brainy Hand

Figure 2.1: Mobile Systems

Focusing on novel freehand gestural interaction Mistry et al. [Mis+09] develop a interactive system containing a projector, a camera and coloured markers on the finger to track gestures (see Figure 2.1a). In addition interaction with different augmented physical objects (see Figure 2.1b) is possible. Popular interactive multitouch based gestures as pinch to zoom are developed. Tending a wide distribution of the system Mistry et al. distributed *SixthSense* as an open source project [Pra]. Furthermore, they planed to improve the prototype by markerless tracking, distortion free projection as well as a smaller form factor. The proposed system shows already that complex interactions are possible with average hardware however markers were necessary. Moreover distortion free rendering is not yet realised since it is very complex and computational expensive with the proposed hardware.

Another system similar to *SixthSense* which also allows interaction with physical objects is the PROCAMS by Brauer et al. [Bra+09]. Here the PROCAMS is not wearable but handheld. It is capable of augmenting physical objects with meta information. A flashlight metaphor as interaction concept which present stored metadata next to the illuminated object is used. Recognition is done via computer vision using motion and feature tracking. Compared to the *SixthSense* prototype features are matched against a database which makes separate markers redundant.

Presented and developed systems are often to bulky for everyday usage. Tamaki et al. [Tam+09] focus on this issue and propose *Brainy Hand*, a wearable hand gesture interaction device. They show that it is entirely possible to construct very small PROCAMS but with a tradeoff of flexibility in feedback and interaction. Tamaki et al.describe two different devices. A simple one containing only a small camera to estimate 3D hand gestures as input and a laser to indicate the field of view. The second version additionally includes a projector to display visual feedback. The concept of *Brainy Hand* is illustrated in Figure 2.1c. A novel 3D hand gesture estimation technique which does not require any marker was used to detect up to five gestures including point, motion and sign gestures.

With an implemented music player application the feasibility of the prototype has been shown.

Molyneaux et al. [Mol+12] present a novel environment aware projector to enable new interaction method. To enable spatial and geometrical awareness they pursue two different approaches. On the one hand, they develop a PROCAMS which is tracked by a tracking system deployed in the environment. This enables environmental aware distortion free projection as well as novel freehand user interactions du to the tracking of the user. One disadvantage of this approach is the need of a deployed tracking system. The second PROCAMS is equipped with Microsoft Kinect which enables to determine the pose in space via simultaneous localisation and mapping (SLAM) technique. This allow spatial aware projection and re-projection without any infrastructure.

Interactive Phone Call by Winkler et al. [Win+11] provides in call collaboration while making a telephone call. Therefore they proposed to attach a pico projector to a mobile phone to project content to a flat surface. This allows the user to interact and share content with one hand while holding the phone in the other. Winkler et al. use a desktop metaphor with a private and a shared area and additional extended permissions for file sharing.

OmniTouch by Harrison et al. [Har+11] is a shoulder-worn portable PROCAMS. Multi-touch interaction on planar surfaces is provided by a depth sensing camera. Harrison et al. propose that with *OmniTouch* they can provide phone like interfaces in the palm of the hand. Projecting content as well as multitouch interaction works without calibration nor instrumentation of the environment. Hence geometrical awareness is limited to the depth sensing camera data. Spatial awareness is not provided.

Winkler et al. [Win+14] present the *Ambient Mobile Pervasive Display* concept. The hardware composition of the prototype is very similar to *OmniTouch* [Har+11]. Winkler et al. refine this idea. A constantly projection in front of the use to project public information as well as a projection in the palm of the hand for private information on demand is provided. In contrast to *OmniTouch*, the build prototype has geometrical and spatial awareness due to the used sensors.

There are several HCI research groups which focus on mobile interactive projector systems. Still, there are several issues and challenges to overcome. Systems offering advanced input methods require advanced algorithm and powerful hardware which reduces battery live or even mobility. For mobility purpose pico-projectors are used, but they require room lighting to be significantly dimmed and make outdoor use fairly impossible. Others systems even need an instrumentation of the environment. But there are first approaches to solve some of these problems by more sophisticated computer-vision based techniques or by applying more sensors to the build prototypes. A lot new questions arise form the proposed systems. For example, which is the most intuitive and comfortable way to interact with such portable systems. Especially in public. Or, more technically questions, is it possible to project undistorted 3D objects on non-planar surfaces. It might be reasonably assumed that over the coming years, as the various underlying technologies get improved, mobile PROCAMS could become widely distributed.

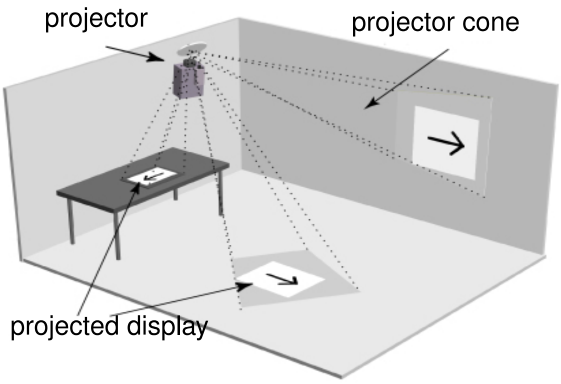
2.2 Mounted Systems

Mounted PROCAMS are systems which are basically constructed in laboratories to evaluate new interaction concepts in a domestic environment or how the physical and digital world could blend together for example by augmenting physical objets with meta information.

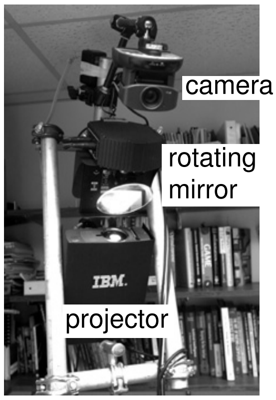
More than 20 years ago Wellner present the *DigitalDesk Calculator* [Wel91]. This is one of the first approaches to merge the physical world with the digital. He propose a fixed PROCAMS to enable touch interaction on a desk as well as concepts to transfer paper documents to electronic ones and vice versa. Touch point detection is done by simple image processing. The camera is mounted on top of the desk which makes tap detection difficult. Hence Wellner use a microphone to detect taps.

Pinhanez [Pin01] propose *Everywhere Display*, a novel PROCAMS which can project to several predefined surfaces. A rotating mirror placed over the projector enables a movable projection cone. A also movable camera is used for interaction input. A schematic setup and the prototype are illustrated in Figure 2.2. Pinhanez discus several problems and suitable solutions in his work. User interfaces can be projected distortion free, but each surface has to be calibrated by hand. New challenges accompanied by the steerable projection as the loss of focus, the loss of resolution involved by projection distance or more complex interaction analysis are identified. Oblique projection of content is for example computed by standard hardware by adapting 3D graphics rendering concepts.

Fails and Olsen [FO02] focused on a low cost solution for ubiquitous interaction. They proposed *Light Widgets* which makes use of inexpensive cameras to detect skin coloured objects in certain areas of the image to trigger events. Interfaces are manually calibrated in the still image of the camera. Since projectors are expensive Fails and Olsen provide feedback via a XWeb interaction platform which distribute it to any connected output modality like TVs, the traditional desktop or speech. This work shows that even with



(a) Everywhere display projector setup



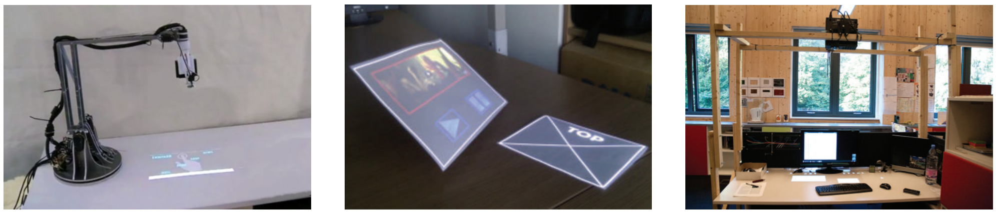
(b) Everywhere display projector setup

Figure 2.2: Everywhere display projector [Pin01]

cheap hardware ubiquitous indoor interaction is feasible however a instrumenting the environment and manual calibration is mandatory.

Andy Wilson is a principal researcher at Microsoft Research focused on depth sensing interaction techniques [WB10; Wil10] as well as wearable [Har+11] and mounted PROCAMS. In 2012 Wilson et al. proposed the *Beamatron*, the first movable projector with depth camera. The depth sensing camera allows geometrically aware projection. They managed to project information on moving objects in the physical world. Moreover, the user is tracked and can grab a digital item, carry it through space and place it somewhere else as desired. Alternatively the user can grab items via audio command while pointing at them. Here interaction is only limited by the view frustum of the depth sensing camera. In addition Wilson et al. develop a novel menu in space where the user select items depending on the height of the hand above the table.

Linder and Maes [LM10] introduce *LuminAR*, a projected augmented reality interface, which dynamically augments objects and surfaces with digital information. *LuminAR* consists of a pico-projector, a camera, and wireless computer in a compact form factor suitable to be embedded in everyday objects like a desk lamp (see Figure 2.3a). The idea based on [Und+99] but is enhanced in several ways. Linder and Maes add robotic movement to the desk lamp to enable dynamic motion capabilities. Hence interaction space and augmentation experiences are increased. With the objective of augmented reality becoming mainstream Weissmann et al. [Wei+13] publish *LENS* a Javascript framework for easy application development for the *LuminAR* platform. This framework provides a API for multitouch, marker and contour tracking, connectivity and decouples complexity from the application.



(a) *LuminAR* prototype (b) Undistorted projections to flat surface (c) Long term study environment

Figure 2.3: Mobile Systems

In contrast to the movable *LuminAR* system, Hardy and Alexander [HA12] propose a fixed hardware setup. He presents a software toolkit designed to enable rapid development of multitouch enabled interactive projection-based displays. His framework based on the .NET framework and only requires a ordinary projector, a Microsoft Kinect and a Windows workstation. All challenges involved into interactive projections are decoupled from the application which is a simple HTML5 web page. Interaction like multitouch are injected into these websites. As shown in Figure 2.3b it is possible to simultaneously project displays distortion free to diverse aligned flat surfaces. Still a manual calibration task is necessary. The toolkit capabilities and performance are evaluated. Interestingly touch accuracy, excluding extreme cases, mainly depends on the distance to the surface not on the angle.

In a long term study over one year Hardy [Har12] evaluated the impact of a PROCAMS on general office productivity. The PROCAMS depicted in Figure 2.3c is deployed above his desk and offers stylus input. Positively mentioned are the physical integration of the system, the management of notes and collaborative working on ideas as well as the logical differentiation between the projection and the real displays. On the other hand Hardy criticise the low resolution of only 20 dpi and the bright light of the projection which makes it unpleasant to read or focus on the projection for a longer period of time. A lack of multiuser support and the single input are other limitations. Occlusions are not necessarily a problem as he is sitting most of the time at the desk which does not lead to significant occlusions. Hardy outlines that the proposed interactive table is a transitional vision and no replacement for a personal computer.

Achieving ubiquitous computing Xiao et al. propose the *WorldKit* system [Xia+13]. Using a paired depth camera and projector, user interfaces can interactively placed on nearly any surface by simply rubbing the surface with the flat hand. Compared to *LuminAR* and the toolkit by Hardy none manual calibration of surfaces or setup tasks are necessary. Furthermore, the WorldKit systems implemented a *interactor* library offering a variety of interaction types like contact, contact area, number of contacts or colour and brightness. The system encapsulates these capabilities to enable easy and rapid development of novel interfaces. Xiao et al. expect that this abstractions make the new ubiquitous computing accessible for more developers.

Overall, there is an increase in proposed systems and toolkits over the last decade. Starting from Wellner idea 1991 researchers develop systems to achieve ubiquitous computing. Beginning with easy interaction realised with simple cameras and manual configuration tasks, the release of the Microsoft Kinect provides a inexpensive solution for novel interaction concepts. Systems and toolkits get enhanced from static to movable PROCAMS offering more accurate touch input and simple application development by providing simplified APIs. Nowadays PROCAMS enable ubiquitous, on demand creation of interactive interfaces without manual calibration tasks. Issues and challenges know form portable systems like power management or computational power are not crucial but there are still hardware limitations like interaction accuracy or projection resolution. Furthermore all presented systems are tied to laboratory. Non of them where actually used to conduct user studies in a domestic environment.

2.3 Commercially Available Products

To give a complete overview of the current PROCAMS evolution, commercially available products for educationally use, entertainment or advertisement are discussed in the following.

(a) *Light Touch* by Light Blue Optics[1]

(b) omiVista by OM Interactive

Figure 2.4: Commercial products

[1] http://www.engadget.com/products/light-blue-optics/light-touch

The startup Light Blue Optics [Opt] launches *Light Touch*[Per] at the Consumer Electronics Show in 2010. *Light Touch* is a standalone PROCAMS with novel laser projection technology which turns flat surface into a touch screen. Due to the laser projection the image is always in focus. The device supports multitouch via camera tracking. Simple applications like photo manager, movie player, or messaging service are presented. Sadly the projection area of the device is very limited and seems like a tablet. Interaction is heavily based on concepts known from current smart phones. However, the small standalone character of the final product which is shown in Figure 2.4a is very appealing.

OM Interactive motivated their products for the education and health sector. They sell and rent different interactive projection systems for different setup situations like wall, floor or ceiling projection. All interactive systems are powered by the *EyeClick* engine which respond to gestures or movement tracked by a infrared camera system. Gesture information are forwarded to a computer which adjusts the content. Because the gesture recognition appears to be not very accurate user interface elements are very large or the interaction concept does not require direct selection of objects. A sample template showing the omiVista [OM] is illustrated in Figure 2.4b. Other companies offering similar products as OM Interactive are GestureTek or Reactrix Systems. Later are focused in particular on out-of-home advertising, retail, and entertainment. The used technologies and scope of operation are basically the same described above.

3 Interviews

First of all, the user requirements for a projector camera system (PROCAMS) had to be determined. Therefore, structured interviews were conducted, to figure out how such a system could be employed in a domestic environment and would be used by end users. In interviews potential end users could remark their ideas and possible application scenarios. From the obtained information possible use-cases as well as the hardware and software requirements for a PROCAMS were derived. In the following the realised interviews are described and their results are discussed.

3.1 Methodology

For explaining the typical properties and abilities of a PROCAMS and for stimulating the creativity of the participants, a simple mock-up was built (see Figure 3.1). It comprises a portable battery powered LED projector PocketCinema V60 by AIPTEK contained in a cardboard box. The projector provides 50 ANSI Lumen and projects images up to 152 cm in diagonal size. On the left side, it has a continuous rotating wheel to focus the projection. On the cardboard box, several hardware components are illustrated. On the front, a depth sensing camera loosely based on a PrimeSense carmine is shown. On the left side, a LCD display with some information and beneath four buttons (select, up, down, back) are affixed. On the backside, a big power button is illustrated. The mock-up has a total size of 12.5 cm $\times$ 6.5 cm $\times$ 6 cm and weighs 195 g. It is mounted on a tripod with flexible, wrappable legs which allows the user to place or attach it almost anywhere.

The interviews took place in the participant's home. To create a relaxed atmosphere and enable innocuous speaking no audio recording was made. Notes and ideas were drafted down on a clipboard. At the beginning of an interview session some introductory words about ubiquitous computing and what the following interview is all about were narrated.

The structured interviews were separated into three parts. First, every room of the participants home was inspected. They were called upon several questions about how they

Figure 3.1: Simple mock-up for interviews

would use the PROCAMS. In particular, how they would place or mount the system, which were the typical projection surfaces, how they would interact and what content they would like to project. Furthermore, they were asked to build a potential setup with the provided mock-up. There were several pre-designed non interactive widget examples stored on the projector. Widgets are small graphical items which present a specific content to the user. Some of them are illustrated in Figure 3.2. Widgets were displayed when the participant referred to a suitable one and a realistic setup with the projected widget was constructed. Then different interaction methods were discussed. To avoid biasing the results and ideas of the participant pre-designed widgets were not presented beforehand.

The second part was a short questionnaire about the general requirements for a PROCAMS. This part aimed to summarise the basic hardware requirements and design decisions. For example if a separate screen, like illustrated at the mock-up, would be necessary or if the PROCAMS should be movable. Both, outcome of the interview and the questionnaire are discussed in detail in section 3.4.

The last part of the interview was a questionnaire which asked for demographic data and technical understanding of the participant as well as the size of the living area and the housing situation. These results are written down in the next section. The whole interview including the questionnaire took 56 minutes on average.

(a) Clock widget (b) Facebook widget (c) Weather widget

Figure 3.2: Pre-designed content

3.2 Participants and Environment

Interviews were taken in a time period of three weeks. Eighteen participants, half male and half female, took part in the interviews. Their age ranged between 22 and 58 with an average age of 29.5. Half of the participants were students with a technical background. The remaining were students of other fields (22%) or employed. Looking at the technical affinity, 15 of the participants used a smart phone for more than two years. Only one-third of them already had been used a larger touch device, like a touch table or touch notebook. More than half of participants (61%) had experience with free-space gestures interaction devices like the xBox. Inferred from technology they know and use, it is fair to say that the participants had a good technical understanding and were able to imagine what a PROCAMS is capable of.

Most of the flats the participants were living in were shared flats with a living area ranged between 27 and 104 square metre. The average size was 68 square metre. The flats had 1 to 4 rooms (average 2.05) and 78% had a corridor. In average, they lived together with one other person (Min: 0, Max: 3, Avg: 1.05). In 61% (11) of the visited flats, a common room was present. Only one participant (P-13) had a projector installed in his flat. A full table of the collected data is presented in Table A.1

3.3 Definitions

For a clearer understanding of the presented ideas and the following discussion, a few terms need to be defined first.

A projector is capable of projecting onto several flat areas in the physical space at the same time. The illuminated area is defined as *interaction space*. Within this *interaction space* content can be projected onto plane areas named *surface*. Of course, the *interaction space* can contain more than one *surface* for instance when the projector is aligned to project into a corner of a room.

Displays are capsules wherein the digital content is rendered and projected to the physical world. *Displays* are the correspondence to known visual display units as computer or television screens. However, *displays* can render content wherever a *surface* within the *interaction space* is available. Certainly, it is possible to place several *displays* onto one *surface*. The correlation between *interaction space*, *surface* and *display* is exemplified in Figure 3.3.

A *widget* is a simple graphical software component. It serves as a graphical user interface (GUI) for a specific use case as well as the logic for user interaction. Widgets are bound to *displays* which accomplish rendering of the widget.

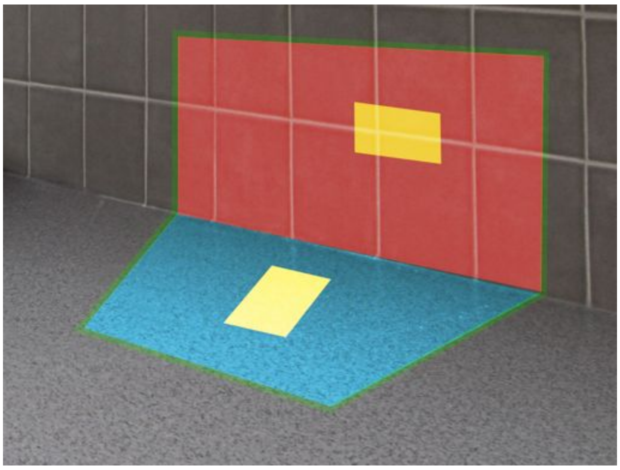


Figure 3.3: Design spaces: *interaction space*: green, *surfaces*: red and blue, *displays*: yellow

3.4 Installations and Interfaces

In the following sections, the results of the first part of the interview are presented. The main focus of attention is the arrangement of the mock-up in the room and how the user would engage in dialogue with it. Furthermore, the ideas of the participants regarding widgets or information which should be projected are discussed here.

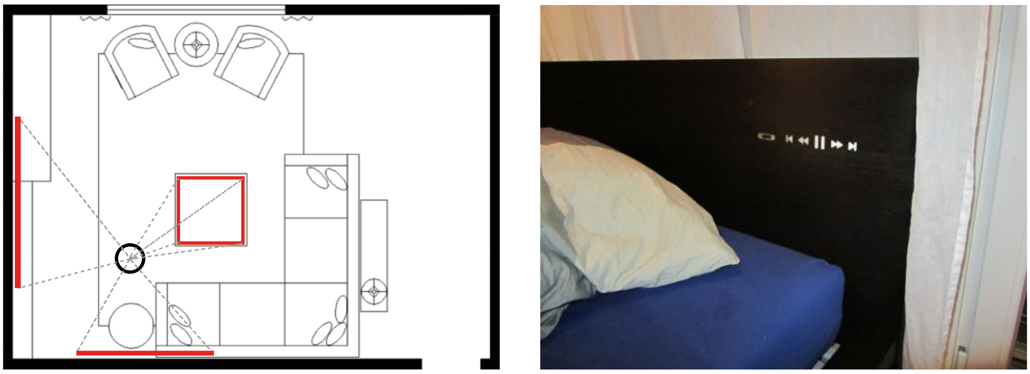
3.4.1 Interaction Spaces and Surfaces

Hereafter, the created *interaction spaces* and the resulting *surfaces* are described in detail. In the kitchen, the participants created one to three *displays* for user input and output. The most commonly used *surfaces* were the cupboard doors, the refrigerator or the wall on top of the oven. Participants avoided using the table or the worktop since they needed the space for cooking or eating. One participant (P-2) placed the PROCAMS in such a way that the *interaction space* includes the wall and the worktop at the same time. They explained that they wanted to use the wall as output display and the worktop as input since it would not be reasonable to touch the wall while cooking.

In the work room, participants used mainly two *surfaces*. Either the wall close to the desk or the desktop directly, if it was not cluttered. Two participants mentioned that the computer screen is enough as an in and output device and they do not require any extension in the working room.

In the living room one particular setup prevailed. Participants decided to mount the PROCAMS at the ceiling. From there, it was possible to render content to three different *surfaces*. On the one hand to the sofa table and from the same mounting point to two different locations on the wall. Typically to the wall on the opposite side of the sofa and one distinct wall. The described arrangement is illustrated abstracted in Figure 3.4a. One participant thought about using the sofa as a *surface* but decided against it because they is often sitting in a different posture or a different location. This would lead to occlusion of the hosted *displays* or unattainability for interaction, they said. Two participants explained that they do not need a PROCAMS in the living room since they have a large-screen television and prefer real images.

In the bedroom participants describe all in all two *interaction spaces*. First, an *interaction space* containing one *surface* in proximity of while lying in bed. For example the bedside cabinet or the wall close to the bed. Figure 3.4b shows such a setup. For improved visibility, projected content is enhanced in some the images of this chapter. Second



(a) Livingroom setup (b) Music controller at bed

Figure 3.4: Living and bedroom setups

a larger *surface* for instance the wall on the opposite of the bed or the ceiling. If they decided to use the ceiling as a *surface* the participants placed the PROCAMS close to the headboard. Even with the assumption of a rectified projection, none of the participants could imagine using parts of the mattress or blanket as a *surface*.

A few participants try to use the mirror as a *display* in the bathroom and were disappointed that they cannot see the widget due to the reflective nature of the mirror. Therefore, *displays* were created close to the mirror or next to the toilet and in the shower. Participants used always a free flat spot for placing *displays*. Some setups are shown in Figure 3.5.

(a) Bathroom setup (b) Restroom setup

Figure 3.5: Bath and restroom setup

In the corridor of the flats participants created *interaction spaces* on free parts of the walls or the doors. One participant also created an *interaction space* on the floor before his front-door to use his doormat as a *surface*.

3.4.2 Placement

While trying different setups participants always try to mount or place the PROCAMS above body height. The PROCAMS were typically mounted at the ceiling in the centre of the room. Apart from that it was placed on top of higher furniture as cupboards or shelfs. Participants achieved with this kind of placement to minimise the occlusions caused by their bodies when move around the room. Dependent on the room size and use case the distance between the PROCAMS and the *displays* were 0.7 m to 3.5 m. For example only 0.7 m at the working desk and up to 3.5 m in the living room when aligned to the large *surface* on the opposite wall of the sofa.

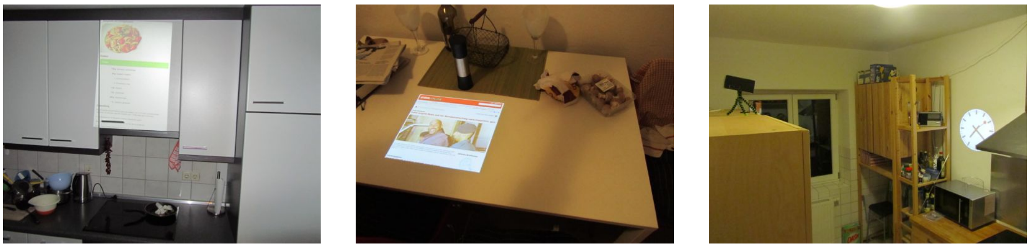
3.4.3 Widgets

Over the course of the interviews, participants propose several widgets. In the kitchen, they describe a widget presenting a digital recipe book. A typical setup using cooking instructions in the kitchen is illustrated in Figure 3.6a.

Another widget which was mentioned several times, especially in the kitchen and working room is a widget which takes track of notes. In the kitchen in particular a shopping list where items could easily be added and get synchronised with the smart phone was desired.

Many participants motivate widgets which can be categorised into social media. Especially Facebook and Twitter or pre-configured feeds based on interests were mentioned. Besides social media, a widget presenting news or dynamic information was outlined. In particular global news, weather, sporting news, the TV guide, calendar or the bus timetable were listed. A news widget rendered at the dining table is depicted in Figure 3.6b. Such a widget should also appear, in the vicinity of the user after awaken, to inform him about the latest and upcoming events.

When the PROCAMS is not actively used participants had two ideas how to use the system in an alternative way. On the one hand, they suggested a widget which shows a simple clock or even a combination of different clocks for different time zones. The clock



(a) Cooking instruction setup (b) News on dining table (c) Clock setup

Figure 3.6: Proposed setups and widgets

widget should also provide a timer and alarm function to support users while cooking or waking up. This use case is illustrated in Figure 3.6c On the other hand participants had the idea to use the PROCAMS as an ambient light source. The system could move slowly around and illuminate the room in desired fading colours.

Participants wanted to use the large *surfaces* mainly for two reasons. First they want to replace the television or the media centre with a corresponding widget. Second, especially in the corridor, the *surface* is used to render a digital image frame showing for example the latest holiday photos. In addition, some participant described an interactive music player which visualises the music on the large *display* and presents a control widget in the vicinity of the user e.g. the sofa table.

Focusing on the sofa table a widget for playing board games was described. Participants wished to combine a projected board with real tokens to play games with dynamic boards.

Other stated ideas were for instance a digital whiteboard, a widget for action items and a reminder in the working room. Or, a remote control widget for all entertainment devices as a replacement for several different remotes they own. Four participants also imagined using the PROCAMS for home automation purposes or even as video door interphone. Another idea which was mentioned regarding the desk was a widget with shortcuts which could trigger events on the PC. For example, launching the mail application or control the music playback. An interesting last idea only named once, refers to the *display* created at the doormat. A widget should appear there which shows, depending on a status, a away or welcome message.

3.4.4 Interaction

As interaction method participants brought up touch interaction in the first place. It appears to be most obvious interaction concept for interaction with widgets. Secondly they named voice interaction. Participants preferred voice interaction in the kitchen since they need their hands while cooking or they do not want to touch the surface because of dirty hands. Other known concepts mentioned were mid air gestures for scrolling or using a laser pointer to trigger buttons when they are not in the vicinity.

Two participants describe more unique interaction concepts. The first wants to use real physical objects to interact with a widget. For example, they described a widget showing a counter which increases every time a coffee cup is placed on it. The other wanted to interact via the shadow they produces when the finger is in the light beam of the projector. In particular, this would be usable in some of the bedroom setups where the PROCAMS was close to the user. Unfortunately, there were no extra novel interaction concepts elaborated. A possible reason could be the well known and daily used touch interaction.

3.5 Findings

During setting up arrangements in the different rooms or final discussion, participants expressed several ideas or ask questions which lead to the following special explanatory notes. While elaborating setups in the bathroom, four participants remark privacy concerns due to the built-in camera. They claimed not to use a PROCAMS in this room. In addition, two participants stated in the bedroom to try to minimise the number of technical devices and therefore would not use a PROCAMS there.

To switch between different *interaction spaces*, the PROCAMS should be movable and listen to voice commands or switch to the *surface* the user is tapping on. In total 67% of the participants want a movable PROCAMS. Eight of them propose a PROCAMS, which moves automatically. Four of them want to control by direct manipulation. Furthermore, five participants suggested a battery powered version of a PROCAMS. The idea behind it was to enable the possibility to take the PROCAMS to friends and share content on a large interactive *display*. Additionally, some participants described the possibility to own only one or two PROCAMS, many for financial reasons, which can seamlessly switch between rooms and always know the available *surfaces* and related *displays*. One participant described an intelligent wall plug which takes control of storing *surfaces* for the specific position of the plug and also recharges the portable PROCAMS submodule.

Another feature which was mentioned is multi user, multi *display* support. A participant explained that they wanted to use a *display* in a *surface*, for example the kitchen table, while another person can interact with another *display* hosted on the same *surface*. All participants agreed that the projected widgets need to be in in focus without any interaction. Furthermore, projections have to be light intensive to ensure visibility during daytime.

For instantiating new *displays* participants explained to use touch gestures or a smart phone as a remote. Further calibration and configuration tasks should not be forwarded to the user, participants said.

Most participants (10) were satisfied with the dimensions and plain design of the mock-up. Most of the time the PROCAMS will be mounted at the ceiling and is thus not directly in the field of vision. However, some of the participants indicated that they prefer a lighter (3), more robust (1) or more aesthetic (2) version.

3.6 Summary

Interviews with 18 participants were conducted, thereby gaining some new and interesting use cases for a PROCAMS. In particular, the wide variety of already existing things which PROCAMS can replace is remarkable. Wall paintings, television, clocks, printed bus timetables, alarm clock or a remote control to cite just a few.

Noticeable is that all participants chose only flat *surfaces* even after clarifying that it would be feasible to project without distortion to irregular surfaces. Participants created solutions to get the most of the PROCAMS by proposing a movable and straightforward mount and unmountable system.

On the negative side participants claim about privacy issues. Some of them also mentioned that depending on the situation parts of the body could occlude larger parts of the display. The overall positive interviews with various ideas lead to some sufficient requirements. Aspiration of the resulting PROCAMS is to be available to everyone in a domestic environment.

4 Requirements

The build PROCAMS is intended to be used by end users in order to investigate their behaviour with everywhere projections. The requirements for this PROCAMS are deduced from the findings of the previously accomplished interviews. They are structured in a hardware and software section.

4.1 Hardware

First of all the PROCAMS consists of two main components: a projector and a camera. The projector should be lightweight, light intensive and most important have an autofocus, which continuously focuses on the desired *surface*. The camera is necessitate to interact with the PROCAMS. To realise simple touch interaction a depth camera is reasonable [Wil10; Klo+12]. To achieve a PROCAMS which can project to different *interaction spaces* of the room, parts of the system need to be movable. At least the projector and the camera must be able to rotate in two degrees of freedom. According to 50% of the participants, alignment should be possible without any manual effort.

Besides the projector and the camera, the system needs a computational unit which handles at least pre-processing of the camera images, video output to the projector and control of the movement of the unit. Furthermore, there needs to be a wireless link for data transmission as well as a loudspeaker and microphone for audio in- and output. These are the minimum hardware requirements to fulfil the majority of use-cases mentioned in the interviews. A first schematic sketch (see Figure 4.1) illustrates how such a system could look like.

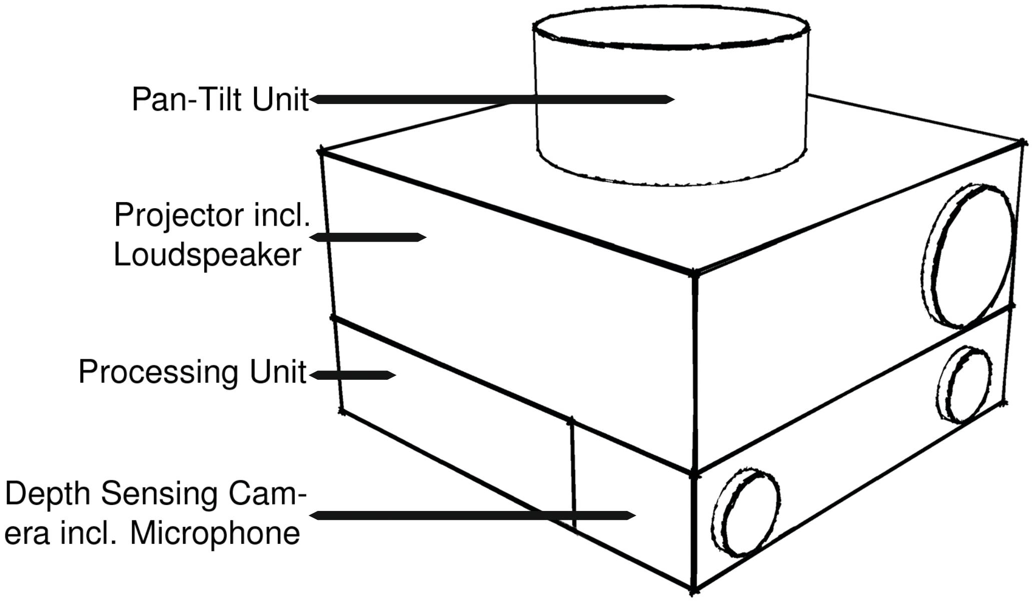


Figure 4.1: Schematic sketch of a PROCAMS

4.2 Software Framework

The scope of operation of the hardware is very similar to many other systems [HA12; Xia+13; Wil+12; Ras+05]. But for the user, an efficient working software is as important as a capable and light hardware. To enable frustration-free usability, the system must be usable with a minimum of configuration task. As far as possible configuration and calibration have to be done in the background hidden from the users perspective. For example, after creating a new *interaction space* by aligning the system to the desired space the user should not have to calibrate the system or manually set the focus. All he has to do is to place the *display* to the desired location.

The PROCAMS should be tilted and paned via remote to enable simple changes of the *interaction space*. The space the projector is illuminating. Furthermore, it should be possible to store and load the current settings, including the actual alignment of the system, the position of *displays* and therein rendered widgets. This means that the user is able to load previously stored settings and the PROCAMS automatically aligns to this *interaction space* and projects the last shown widgets to the stored positions. In addition, it should be possible to dynamically add and remove *displays* to the *surfaces* without the need to do manual adjustments for a rectified projection.

Development of widgets should be unproblematic and be deployed effortless into the framework in order to enable new rich content. Interaction with widgets should mainly be done via touch input. An alternative input method could be plain speech commands to trigger events like moving to a previously defined *interaction space*. All in all the software framework should disburden the user from unnecessary tasks and allow intuitive use of user-friendly widgets.

5 Systems Architecture

An intuitive to use PROCAMS features a tight integration of software and hardware components. Furthermore, a fine calibration and well assembled hardware are essential for proficient operation. In the following the architecture design of both, hardware and software, are described in detail.

5.1 Hardware Architecture

The proposed hardware architecture is designed for a compact stand-alone PROCAMS. The basis of the systems consists of a camera and a projector. To enable accurate touch detection a depth sensing camera will be used. A motion capture system is no alternative hence it requires an instrumentation of the environment. Such instrumentation would not meet the acquired requirements. For an easy transformation between the camera and projector coordinate systems, the camera is fixed to the projector. It is important, that the field of view (FOV) of the camera matches the FOV of the projector as closely as possible. The projector itself needs to be focus free, as known from laser projectors, or requires the possibility to set the focus electronically via control commands.

Two DOF are required to have enough variance to project to any spot visible from the mounting point of the PROCAMS. Thus two rotary actuators are utilised for panning (yaw) and tilting (pitch) of the projector camera unit. The actuator which tilts the unit is bolted directly to the projector. The other is shifted by $90°$ in a vertical plane and bolted to the first actuator. Hereby the projector camera unit is always level. A third actuator which would pivots the unit side to side (roll) is not necessary since it would not change the *interaction space* in a meaningful way.

For a stand-alone solution, the PROCAMS includes a powerful processing unit performing all calculations without any remote dependencies. To minimise cable tangle, the processing unit is placed close to the projector and camera. Hence the processing unit is also moving along with the camera and projector. The proposed hardware architecture with a single processing unit for each PROCAMS maximises scalability since no remote connections or calculations are necessary.

An alternative approach using only one powerful dedicated workstation for several PROCAMS is not considered any further. Indeed, the systems could be smaller and lighter since they do not require a processing unit, but this approach will not scale well. Broadcasting a depth sensing camera video feed for touch detection or elements of the graphical user interface would quickly overstrain the wireless link to the remote unit.

5.2 Software Architecture

Developing a PROCAMS fitting the requirements of an average end user is challenging. Technically complex tasks have to be hidden from the user. The initialisation effort has to be minimal while maintaining flexibility. A generic software architecture design for a rotatable PROCAMS providing touch input is illustrated in Figure 5.1 and described in the following.

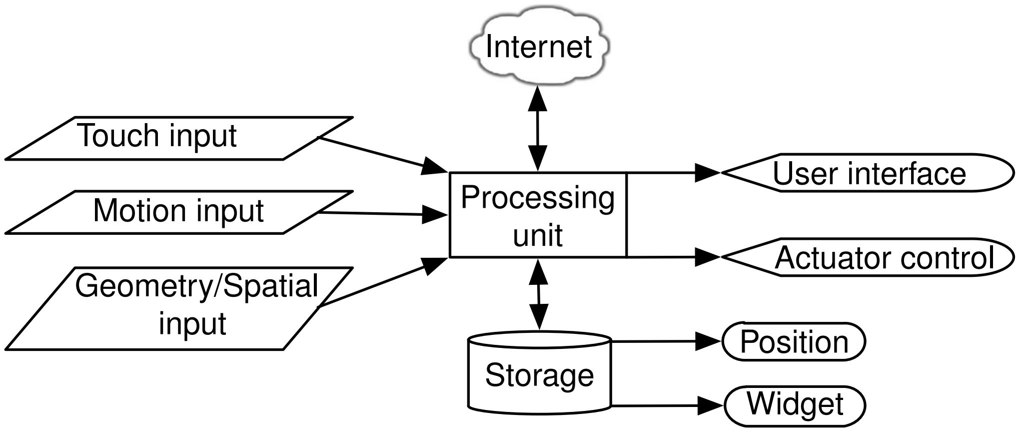


Figure 5.1: Generic projector camera system software architecture

Processing Unit The processing unit is the core piece which includes all logic of the PROCAMS. To provide the desired functionality, it requires three input sources. Touch input to control the GUI, motion input to control the pan-tilt unit and finally geometry and spatial input for rectified projection to a defined positions in space.

The main task of the processing unit is to collect all these data and performs the necessary calculations to react like the user intends. For example, touch data are reported to the processing unit in a physical world coordinate system. To trigger events on the corresponding widget, touch events have to be transformed to the GUI coordinate system regarding the geometry and spatial input. Moreover, the processing unit takes account of all *displays* and rendered widgets. To provide dynamic loading of new content or widgets the processing unit is wirelessly connected to the Internet. Furthermore, the wireless link is used for receiving remote motion input which is translated by the processing unit and then forwarded to the actuators to control the pan-tilt unit.

User Interface For the end user, the graphical user interface is the most important part of the PROCAMS. In this architecture design, the GUI consists of the actual *surfaces* within the *interaction space* the projector is projecting to. Therefore, it is dynamic in terms of spatial alignment. Within a defined *interaction space* widgets can be placed, moved or removed freely on any *surface*. Since the projector is rotatable, *surfaces* are not always planar to the projector. Hence, widgets are encapsulated in *displays* which pre-warps the content to get a rectified reproduction of the widget rendered onto the surface. Geometrically input is used by the processing unit, to calculate the needed transformation to obtain a rectified projection.

Touch Input Touch input is provided by new smart phones to interact directly by touching the screen with one or more fingers. Nowadays, it is a common and well understood input method for users. To enable touch input on ordinary non instrumented surfaces many PROCAMS [HA12; Wil+12; Xia+13] use a depth sensing camera to detect touch. Regardless of how the touch is detected, the world coordinates of the detected touch event are passed to the main processing unit. There they are further processed to trigger the desired interaction.

Motion Input Motion input is used to control the alignment of the PROCAMS. In order to define a new *interaction space* motion input needs to be provided externally via remote or touch input. Motion input data is passed to the processing unit which triggers the desired events into control commands. Alternatively, stored configuration can be loaded from the database. Control commands are then directly forward the to the actuators.

Geometry and Spatial Input Geometric and spatial input is necessary to gain geometrically and spatially awareness. Geometrically awareness enables the PROCAMS to project rectified GUI elements to planar or even nonplanar surfaces. Spatial awareness ensures that stored GUI elements are always projected to the same position in space. Geometrical and spatial input can be accomplished via tracking or sensing of the environment, or even a combination of both.

Actuator Control Actuator control commands directly power the actuator which rotates the PROCAMS or control the focus of the projector when not focus free. Commands are divided into three individual command segments, which are calculated independently by the processing unit. The tilt command segment controls the rotation of the system in the vertical plane. The range needs to be between $0°$ for horizontal alignment to $90°$ for projection onto the floor. As long as the PROCAMS is mounted at the ceiling this range is enough to project to the lower hemisphere. The pan command segment is for rotation in the horizontal plane of the PROCAMS. A range of at least $360°$ is necessary to cover the complete hemisphere. The processing unit translates the motion input or stored *interaction spaces* alignments into actuator control commands to enable projections to all parts of the space. If a projector is used which requires manual focusing there is an additional control command segment to keep the projection in focus. The processing unit is required to analyse the spatial and geometrical data input and calculate the correspondent focus actuator command.

Storage The storage is directly accessed by the processing unit to store and load content, settings and other properties. In particular widgets and positions. As already described, widgets are relatively simple and easy to use software components which are loaded into the GUI. They can vary from simple clock representation, to interactive information presentation or mini games. The other important information which is stored permanently are *positions*. Positions represent the alignment of the PROCAMS for a special use case and are separated in a pan and a tilt component. A position property additionally contains the prior projected widgets and their position within the corresponding *display*.

6 Hardware Implementation

Several projector camera system (PROCAMS) hardware designs have been proposed but often they are bulky, very expensive or need additional environmental instrumentation. To fulfil the gathered requirements from chapter 4 several technologies are evaluated. Focus of attention is on size and quality on the one hand but also the price on the other. To equip homes with multiple PROCAMSs they need to be affordable.

6.1 Actuator

Actuators are necessary in order to realise the movement of the PROCAMS as well as the automatic focus. There are basically two kinds of electric motors which suit the requirements: stepper motors and servos.

A stepper motor is a synchronous brushless electric motor. Rotations are divided into 200 to 400 equal steps. To control the motor, a micro-controller and a additional motor driver is needed. The motor can move step by step or hold the current position. There is no feedback about the current position since the stepper motor has no sensors. For a closed-loop control of the pan-tilt unit, separate sensors are needed, for example an accelerometer and a compass. Otherwise, the system has to execute a self-calibration each launch to determine the movement constraints and the mapping between steps and position.

In contrast to a stepper motor, a servo is a combination of an electro motor and a sensor, which provides position and speed feedback. The position is controlled by a pulse-width modulation (PWM) signal which can be generated by a micro-controller. The output shaft is moved depending on the difference between the commanded position and the measured position. The position is typically determined by a potentiometer. This allow up to 36000 steps per rotation. Servos are generally used in small robotics. Besides servos for rotational movement, there are also linear moving servos.

Servo motors with high quality potentiometer offer up to 36000 possible positions per rotation compared to 400 of a stepper motor. Power is only consumed as the servo rotates. A stepper motor continues to consume power while holding a commanded position. As consequence, it runs warm. Finally, stepper motors need additional hardware to control precise positioning. Considering these arguments, using servos as actuators seems to be the superior approach.

6.2 Microcontroller

A separate microcontroller is needed to command the actuators. When using a servo the microcontroller outputs a PWM to set the position of the servo. Using a stepper motor, the microcontroller commands the number of steps to the motor driver.

A wide variety of microcontrollers which can be used as a servo or stepper motor controller are available. However, embedded programming and an external programmer are necessary. The USB IO Board from Hardkernel[1] allows easy upgrading of the firmware via USB. It provides a PWM interface via a peripheral interface controller (PIC) microcontroller. Nevertheless, low level programming is required.

Boards like the IOIO-OTG[2], mainly developed by Ytai Ben-Tsvi, provide similar interfaces. The IOIO-OTG is especially designed to work with Android devices. A Java based API offers abstract and lucid programming. With such a board and the provided API neither low level programming nor an external programmer is necessary.

Another platform offering effortless microcontroller programming is Arduino [Ard]. It is an open-source prototyping platform combining software and hardware. Many boards in different sizes and versions are available to suit different use-cases. Whereas, hardware design is open-source there are also less expensive boards from third-party manufacturers obtainable. The microcontroller is programmed with a platform independent IDE based on the Processing IDE. Various Arduino libraries are linked to simplify C and C++ programming. The servo library for example, allows setting the position of the servo with one simple command. Regarding these arguments, Arduino is used as microcontroller platform.

[1] http://www.hardkernel.com/main/products/prdt_info.php?g_code=G135390529643 visited: 02.04.14

[2] http://ytai-mer.blogspot.de/2013/01/go-go-ioio-on-go.html visited: 02.04.14

6.3 Projector

Projection technologies became more sophisticated over the years. New technologies allow smaller and less power consuming devices. Projectors can be classified by light source and projection technology. Typically light sources are laser, LED, arc lamps and combinations of laser and LED. Arc lamps consume much power and become very hot. So, they require a powerful cooling system which makes them loud and bulky. Therefor, projectors with an arc lamp are not further examined.

There are three major projection technologies utilising lasers as light source [Mer]. With Laser-Beam-Steering the image is created pixel by pixel by a laser beam guided across the projection surface by using a mirror. Combining three lasers with different colours (red, green, blue) and intensity using optics, a full coloured image is created. Light Blue Optics [Opt] developed Holographic Laser Projection where a laser illuminates a hologram that diffracts the laser to create the original image. Light Blue Optics claim, that it is very power efficient and has less speckle than other laser projector technologies. A new technology combines lasers and Liquid Crystal on Silicon (LCoS) to project an image. White laser light as a combination of red, blue and green is used to illuminate the LCoS, which control the amount of reflected light by changing polarisation. Research showed [Gut+11] that with this approach more eye safety, lower power consumption and higher resolution are possible. All three technologies using laser light have the advantage that the image is always in focus. No manual refocusing is necessary. However, maximum brightness of laser projectors is 50 ANSI Lumen which requires a dimmed environment for acceptable visibility.

One of the most mature technologies using LEDs as light source is Digital Light Processing (DLP) developed by Texas Instruments [Ruk+12]. Light is emitted to tiny mirrors on a chip that directs the light. Dependent on the state, light is reflected to the surface or an absorber. Colour is generated by a rotating colour wheel splitting the light into red, blue and green. The above described technology using LCoS is also used with LEDs, but this time focusing optics are necessary. Since LEDs are not getting very hot while operating, small and energy efficient projectors with LEDs are available. While maximal brightness of tiny laser projectors is limited to 50 Lumen, LEDs projectors provide up to 550 lumen without exceeding the size of half cubic decimetre.

6.4 Depth Sensing Camera

Three distinct technologies are applicable to generate depth information [KA12]. Stereoscopic vision systems use two slightly offset cameras and compare the obtained images. On basis of the difference in location of a point in corresponding images, the depth can be calculated. Stereoscopic vision systems are very low-cost but high in software complexity. The depth accuracy is only in the cm range.

Depth sensing cameras using Structured Light (SL) pattern have also high software complexity but offer, dependent on the model, depth accuracy in µm to cm range. In comparisons to a stereoscopic vision system, one camera is replaced by a laser or led light source which creates a light pattern. This pattern is obtained by the remaining camera. To acquire depth information the same technique is used, but due to the pattern, finding corresponding points is easier and more accurate.

A relatively new technology is Time of Flight (ToF). It allows depth accuracy in the mm range. An emitter transmits a light pulse to an object. There it gets reflected and finally determined again by the receiver. The distance of an object can be calculated by measuring the time between transmit and receive of the light pulse. Available ToF cameras have only a quarter of pixel resolution compared to SL cameras. But temporal resolution with up to 160 frames per second (fps) (Argos 3D - P100) is very high.

Type	Technology	Range (m)	FOV (H,V)	Depth res. (w x h,fps)	Colour res. (w x h,fps)
Kinect XBOX	SL	0.8 - 10.0	57.0, 43.0	640x480,30	1280x960,12
Kinect Windows	SL	0.4 - 3.0	57.0, 43.0	640x480,30	1280x960,12
Kinect 2	ToF	0.8 - 4.0	70.0, 60.0	512x424,30	1920x1080,30
Carmine 1.08	SL	0.8 - 3.5	57.5, 45.0	640x480,60	640x480,60
Carmine 1.09	SL	0.35 - 1.4	57.5, 45.0	640x480,60	640x480,60
DS311 short	ToF	0.15 - 1.0	57.3, 42.0	160x120,60	640x480
DS311 long	ToF	1.5 - 4.5	57.3, 42.0	160x120,60	640x480
DS325	ToF	0.15 - 1.0	74.0, 58.0	320x240,60	1280x720
Senz3D	ToF	0.15 - 1.0	74.0, 58.0	320x240,30	1280x720

Table 6.1: Depth sensing camera overview.

In Table 6.1 a brief overview of several commercially available depth sensing cameras is given. In late 2013, Apple bought PrimeSense the manufacturer of the Carmine cameras. Ever since PrimeSense cameras are no longer available for purchase. The DepthSense311 (DS311) and DS325 are developed by SoftKinetic a Belgian company. The

Kinect and Kinect 2 is manufactured by Microsoft, but by now it is not possible to develop applications with the Kinect 2. Senz3D bases on the DepthSense311 and is made available by Creative.

SDKs and Frameworks

Hardware manufacturers distribute powerful SDKs to minimise the challenges in developing frameworks or applications which their depth sensing cameras. Moreover, there is a big community developing frameworks, offering numerous useful functionalities and services.

The Kinect SDK is used to develop Kinect-enabled applications. It includes Kinect Fusion and Kinect Interactions. With Kinect Fusion it is possible to create a 3D reconstruction of objects or the environment in real time. Kinect Interaction provides gesture recognition. iisu, an acronym for ”The Interface Is You“, is the 3D gesture recognition development framework by SoftKinetic. As well as Kinect SDK it supports full body tracking and natural gesture development and recognition. Intel Perceptual Computing SDK supports body and gesture recognition as well as facial analysis, hand and finger tracking. The SDK focuses on short range interaction in particular. Open Natural Interaction (OpenNI) is an open source SDK for middleware and application development mainly driven by PrimeSense. The framework supports voice and voice command recognition, hand gestures and full body motion tracking. The point cloud library (PCL) is an open source library which offers many algorithms for n-dimensional point could processing. In the context of depth sensing, the provided algorithms like filtering, surface reconstruction, or segmentation allow sophisticated analysis of the scenery. Table 6.2 summarises which depth sensing camera receives the support of which framework.

	OpenNI	Kinect SDK	Intel PC SDK	IISU	PCL
Kinect	●	●	○	●	●
Kinect 2	○	◐	○	○	○
Carmine	●	○	●	●	●
DepthSense	○	○	●	●	○
Senz3D	●	○	●	●	●

Table 6.2: Overview of supported SDKs.
●supported ○not supported ◐announced

6.5 Single Board Computer

The hardware design of the proposed stand-alone PROCAMS includes a processing unit which is fixed near to the projector and camera. The processing unit needs to be small and lightweight. To connect WiFi, microcontroller and the depth sensing camera, at least three USB-ports are mandatory. These requirements lead to a so called SBC, a complete computer built on a single circuit board. To handle all computations a high performance SBC is essential. An overview of currently existing SBC is given in Table 6.3. It should be noted that most of the high performance SBC utilise the ARM instruction set architecture (ISA). Deduced from mobile phone processors, SBC processors are very energy efficient, produce little heat and have similar computational power as present mobile phones. However, using the ARM architecture narrows the choice of operating systems (OSs) to Linux and Android.

NAME	CPU (GHz)	RAM (MB)	USB	SIZE (cm)	ISA	OS
ODROID-XU	4x(1.6 + 1.2)	2048	4/2	7x9.4	ARM	L, A
ODROID-X2	4x1.7	2048	6	9x9.4	ARM	L, A
ODROID-U2	4x1.7	2048	2	4.8x5.2	ARM	L, A
Origen Board	4x1.4	1024	2	11.9x11.9	ARM	L, A
Arndale Board	2x1.7	1024	3	36x24	ARM	L, A
Nitrogen6X	4x1	1024	2	11.4x7.6	ARM	L, A, W-CE
Sabre Lite	4x1	1024	2	8.25x8.25	ARM	L, A, W-CE
Via Epia-P910	4x1	512	2	10x7.2	X86	W, X, L, A

Table 6.3: Single-board computer overview. Operating system acronyms: **L**inux, **A**ndroid, OS **X**, **W**indows and **W**indows-**CE**

6.6 Used Hardware

Actuator For pan and tilt of the PROCAMS two HS-785HB by HiTEC are used. This are quarter scale servos with a torque of 132 N cm at 6 V. They are one of the stronger servos in this size segment. Based on the data sheet, a full rotation takes less than 9 s. Focus is driven by a NYA 2.9g linear long throw servo SPMSH2040L from Spektrum with a torque of 329g running at 4.2 V.

Microcontroller Actuators are controlled by an Arduino Pro Mini. This is one of the smallest Arduino boards. It is built on the ATmega168 with six PWM output pins for servo control. Arduino is used because it offers great library support for servo control as well es easy prototyping.

Projector For the proposed PROCAMS the ultra-compact LED projector ML550 by OPTOMA is used. In this projector, DLP technology by Texas Instruments combined with a LED light source is deployed. It measures only $105\,\text{mm} \times 106\,\text{mm} \times 39\,\text{mm}$ in size and weights $380\,\text{g}$. The projection distance is between $0.55\,\text{m}$ and $3.23\,\text{m}$.

Depth Sensing Camera The depth sensing cameras presented in section 6.4 can be divided into short (less than 1.4 metres) and long distance sensing cameras. Derived from the interviews, a long distance sensing camera is necessary. For deployment, the Carmine 1.08 from PrimeSense was chosen. It has a smaller form factor than the Kinect and a higher depth resolution than the DS311. Moreover, it is supported by all presented Frameworks except the Kinect SDK.

Single Board Computer As processing unit, the ODROID-XU was chosen. It is developed by Hardkernel a South Korean open source hardware company. ODROID-XU is one of the most powerful SBC available. The CPU is based on big.LITTLE architecture which features a powerful eight-core for efficient handling of multitasking. Besides it has enough ports to connect the Android Pro Mini microcontroller, the depth sensing camera as well as a WiFi-Dongle and keyboard.

6.7 Pan-Tilt Unit

The pan-tilt unit is responsible for moving the complete PROCAMS, in detail the depth sensing camera, the projector and the SBC. There are commercially available pan-tilt units e.g. the D46-17build by FLIR or the MPT1100-SS retailed by ServoCity. However, these products are heavy, large in size and expensive. Prises are around 650 to 2500 USD. Tending to an affordable lightweight stand-alone PROCAMS, an approach with servo motors looks promising. A direct drive mechanism without any reductions minimises additional gearing and bearing parts.

Each of the tilt and pan servo is grounded to a *ServoBlocks* by ServoCity. A ServoBlocks isolates the lateral load from the servo and increase the load-bearing. ServoBlocks are easily screwed together to create rigid structures actuated by servos. An image of a servo fixed to a *ServoBlocks* is illustrated in Figure 6.1a.

(a) ServoBlocks construction

(b) Pan-Tilt unit

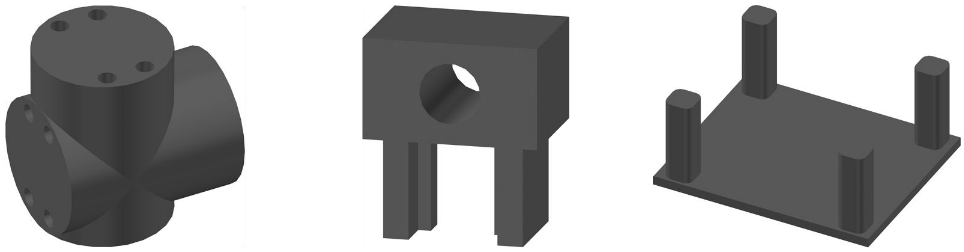
Figure 6.1: Hardwar and construction of pan-tilt unit

The pan servo is mounted overhead. To gain more open space for tilting, a 3D-printed polylactic acid (PLA) spacer (see Figure 6.2a) is connected to the hub shaft. The tilt servos is rotated by $90°$ in a vertical plane. The hub shaft is bolted to the other side of the spacer. A complete illustration of the pan-tilt unit is shown in Figure 6.1b. The servos are

Command	Code	Value	Description
Position	00	-	Returns pan and tilt value
Pan	01	Position in µs	Pan to given position
Tilt	02	Position in µs	Tilt to given position
Focus	03	Position in µs	Focus to given position
Up	04	-	Moves unit up
Down	06	-	Moves unit down
Left	05	-	Moves unit left
Right	07	-	Moves unit right

Table 6.4: Implemented commands for pan-tilt unit and focus control

controlled by the previous presented Arduino Mini Pro. The Arduino framework provides a servo library which allows simple servo control. Via serial interface the ODROID-UX SBC can send commands to the Arduino board which are translated into servo control commands. Commands are seperated into command code, and optional values. Code and values are separated by a # character. The command to pan to a specific direction for example is: `pan#position` respectively `01#1580` where position is the pulse width in µs. All supported commands are summarised in Table 6.4.



(a) Pan-Tilt unit spacer (b) Focus handle grabber (c) Focus servo mount

Figure 6.2: 3D-representations of the special designed parts

6.8 Focus

The focus of the Optoma 550ML is manually adjusted via a small lever. To realise automatic adjustment of the focus, the movement of the lever must be controlled with a servo. A linear servo is used for this task. The servo is glued to the designed servo mount shown in Figure 6.2c. This construction is then glued to the projector, left of the adjustment lever. A special piece is designed (see Figure 6.2b) which grabs the lever and is also connected via a steel wire to the servo. The complete construction is illustrated in Figure 6.4a. The servo is controlled in the same way as pan and tilt servo.

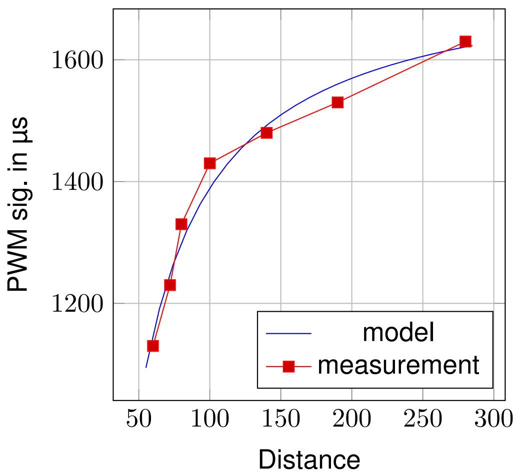


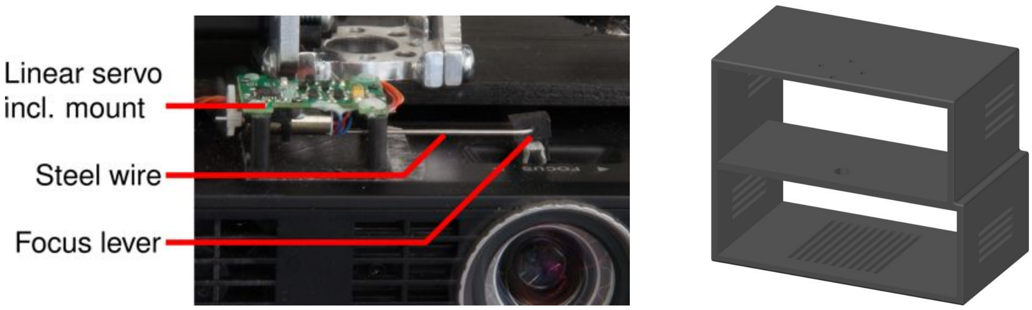
Figure 6.3: Focus-Distance model

To determine the required position of the servo for a given distance a calibration task was conducted. The projector was placed at seven different distances between 0.6 and

2.8 m perpendicular to a wall. The corresponding PWM signal that produces the visually sharpest focus was determined. These seven distance–PWM value pairs were used to determine the formula for setting the focus. Non-linear regression with two parameters was used for analysis. Analysis leads to:

$$PWM = \frac{-36076.6267}{distance} + 1750.06825$$

Where PWM is the PWM signal in µs and $distance$ is the distance between surface and projector in cm. The measured values and the calculated formula are plotted in Figure 6.3. Maximum error is less than 40µs which does not lead to projections which are out of focus.



(a) Autofocus construction (b) Projector and SBC case

Figure 6.4: Hardware construction

6.9 Construction

After assembling the pan-tilt unit and preparing the auto focus, the complete hardware of the PROCAMS is assembled and wired. To connect the projector and SBC with the pan-tilt unit a special case (see Figure 6.4b) is designed and 3D-printed. The PrimeSense depth sensing camera is glued to the bottom of the projector facing the same direction with a maximum overlap of both fields of views. The final hardware construction is illustrated in Figure 6.5.

It measures only 10.5 cm $\times$ 12.2 cm $\times$ 22.5 cm including the pan-tilt unit and weights 996 g. With a size of 10.5 cm $\times$ 12.2 cm $\times$ 9 cm excluding the protrude part of the camera and a weight of about 690 g the PROCAS itself (excluding the pan-tilt unit) is notable smaller and lighter. The complete hardware can be bought and assembled for less than 1000 USD, assuming mechanical skill and access to a 3D printer.

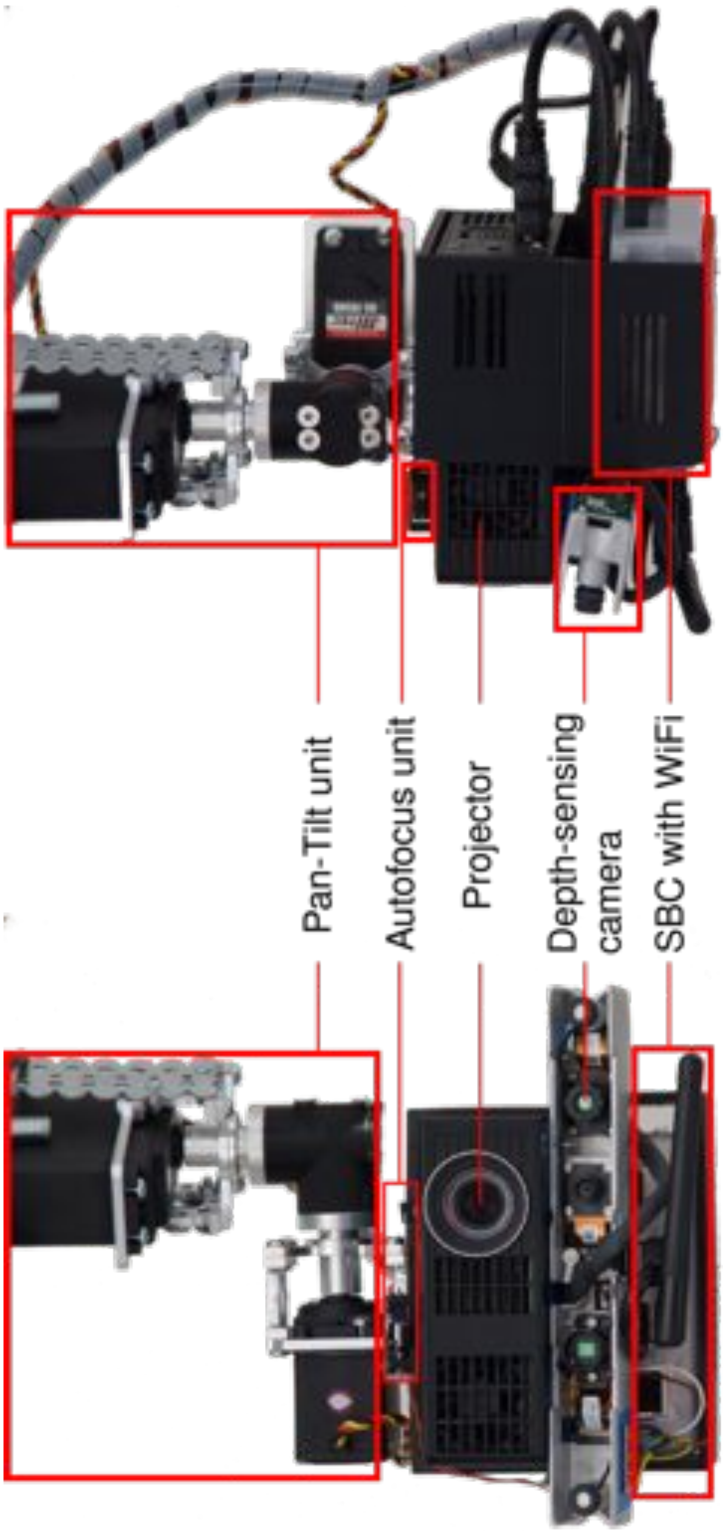


Figure 6.5: Complete construction of the stand-alone PROCAMS

7 Software Implementation

Building a stand-alone projector camera system (PROCAMS) requires a lightweight and resource saving software. Processing power provided by the ODROID-UX presented in section 6.5 can not stick with current workstations. For a responsive user interface experience, developed algorithms have to be adjusted to the available hardware resources.

As mentioned earlier the ODROID-UX builds on ARM architecture. Linux and Android are natively supported. There continues to be no Windows support for the ARM architecture for researchers. Following a guide[1], Ubuntu 12.04 was installed to this relatively new hardware. Some special changes to the video driver and kernel were necessary. Ubuntu was chosen since it is lightweight and has a great library support for different tasks. For reading RGB and depth images, OpenNI version 2.2 for ARM [Pri+] introduced in section 6.4 is used. Image processing is done with OpenCV in version 2.4.6 [Int+]. OpenCV is an open source library aiming at real time computer vision algorithms. Visualisation of widgets is accomplished with Qt (version 4.8.2) [DQ14], a library for UI development using C++ and QML.

The developed software can be separated into five packages.

Touch Detection Detecting direct user input.
Transformation Mapping of touch events to UI elements.
Pan-Tilt Unit Receiving user input and controlling pan-tilt unit.
User Interface Rendering of rectified widgets to surfaces.
Main Logic Loop handling all events and PROCAMS logic.

7.1 Interaction

Interaction with the PROCAMS is divided into three intuitive parts. First, initial alignment of the projector to create the *interaction space*. In other words where the projection should appear. Secondly placing and arranging the UI-elements namely widgets. And

[1] `http://forum.odroid.com/viewtopic.php?f=61&t=2207&p=16999#p16999`

finally, the interaction with widgets itself. The first two tasks are basically one time actions since different alignments of the PROCAMS and the arrangement of widgets are stored. This allows undemanding switching between several surfaces for different use-cases.

The PROCAMS is designed to be ceiling-mounted. Hence, direct manipulation of the alignment is not possible. A remote control to control the alignment of the PROCAMS is implemented based on suggested ideas from the interviews. The user is able to control the pan-tilt unit and in this way the alignment of the projection via their smart phone. An Android-App displaying a virtual joystick allows indirect manipulation as well as storing and loading previous configurations.

After the configuration of the location, the system executes a self-calibration task. This task is concealed from the user and takes only a short period of time. Then, the PROCAMS can be operated via touch interaction. Following interactions are provided:

Adding Widgets
: The user can add widgets by touching and holding the surface at a free spot. A menu with the available widgets appears and the user can choose one of them. The widget is then projected rectified at the previously touched position. Of course, it is possible to add more than one widget to the *interaction space*.

Moving Widgets
: The user can rearrange the widget by touching and holding (long press) the widget until a red frame appears. When moving the finger around the widget sticks it. When done, the user just presses the green check button and the red frame disappears.

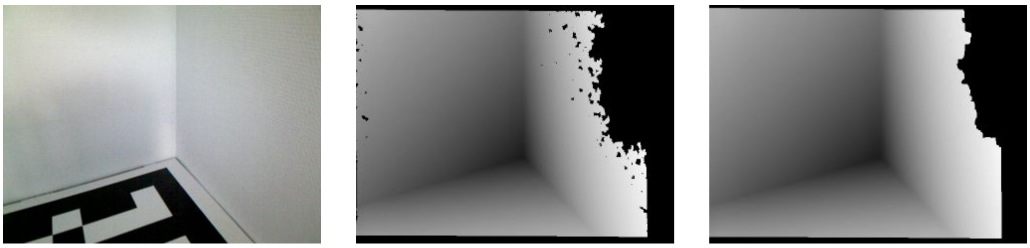
Removing Widgets
: To remove a widget the user long presses on it. When the red frame appears a trash icon on the right side of it needs to be pressed to remove the widget.

After arranging all widgets to the desired layout the user can interact with them immediately. Widgets can offer any content to the user. The current implemented framework forwards four different touch events to the widget. Implemented events are touch down, touch move, long touch and touch release. However, more sophisticated interaction like multi-touch, size and number of touches or the like [Xia+13] can be easily added to the framework.

7.2 Touch Detection

Accurate and responsive touch detection is a crucial part of this thesis. There are several frameworks [FOR; Klo+12] and projects [HA12; Xia+13] supporting finger tracking or touch detection, but all require high performance workstations. Additionally, CUDA (Compute Unified Device Architecture), a parallel computing platform by NVIDIA giving direct access to the GPU processing, or the .NET Framework by Microsoft is required to gain reasonable results. FORTH [FOR] developed tracking runs only at 2.67 fps on an Intel Core 2 Duo with 2.20GHz and 4096 MBs of RAM using a Quadro FX 1600M for example. Neither CUDA nor .Net is supported by the used SBC.

For that reasons a unique very lightweight touch detection, based on the research of Wilson [Wil10] is developed. A key feature is that touch is detected on any physical object without user driven calibration tasks. The developed touch detection can be separated into four parts. First the scenery is analysed and a spatial image, the ground truth, is generated. This obtained image is used to calculate a binary contact image while touch detection is running. The contact image is filtered and simple blob detection detects contact points. In a last step, contact points are tracked over time and transformed into interaction events which finally trigger events intended by the user.



(a) RGB image of scenery (b) Single depth image (c) Filtered ground truth

Figure 7.1: Calculating ground truth

The spatial ground truth image is generated by temporal filtering of 30 single depth images. First a dilation operation with a structuring element of 7x7 in size is executed on the depth image. For each pixel depth information is cumulated and divided by the number of depth information available. As result, a good noise reduced approximation of the scenery is calculated. Temporal and spatial filtering is necessary since the depth sensing camera provides only noticeable noisy depth information. A single depth image is shown in Figure 7.1b and the corresponding filtered ground truth in Figure 7.1c. Depth information are encoded from near (light grey) to far (dark grey). Black pixel provide no depth information because depth is out of sensing range or because of noise.

A simple assumption is made to enable fast contact image calculation. If the current depth for a pixel is less than the corresponding pixel in the ground truth (minus some offset) and is beyond greater than the ground truth minus the height of a finger, the pixel is representing a potential contact point. This coherence is illustrated in Figure 7.2.

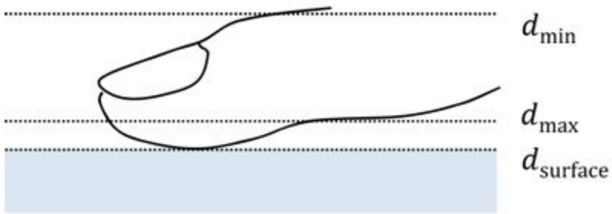


Figure 7.2: Concept of contact point calculation [Wil10]

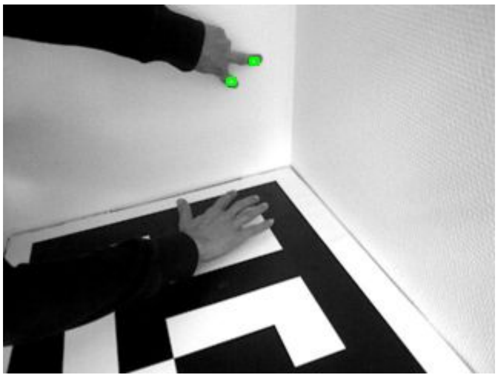
Where $d_{surface}$ is the depth in the ground truth image, d_{max} is $d_{surface} - \mathit{offset}$. The offset needs to be well chosen to minimise false positives due to noise. d_{min} is $d_{surface} - \mathit{finger}$. The finger value specifies in which height from the surface a contact is detected. Mathematical a pixel $d(x, y)$ is a potential contact point if the following formula holds:

$$d_{max}(x, y) > d(x, y) > d_{min}(x, y)$$

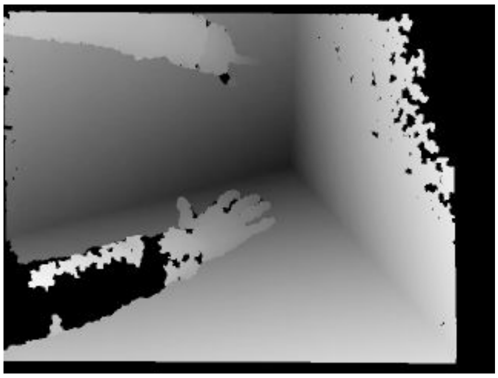
A contact point image is illustrated in Figure 7.3c. Since there are still some false positives the image needs to be filtered before a blob detection can be executed. Therefore a low-pass 3x3 box filter and subsequent thresholding is applied. For performance reasons the box filter is separated into one 1x3 respectively 3x1 filter. Separating the filter leads to faster filtering by more than 40%. A filtered contact point image is shown in Figure 7.3d On this filtered image a blob detection is executed. First the *SimpleBlobDetector* of the OpenCV framework was applied, but it proved to be too slow. Analysing one frame takes between 45 and 58 ms which is far from realtime. Hence, an alternative library namely OpenCVBlobsLib[2] is utilised. This library makes use of multi core processing and analyses an image in less than 6ms. The result of a blob detection is illustrated in Figure 7.3a and Figure 7.3e. Detected blobs are marked green in these images.

Due to the prior analysis of the scenery, contact points can be detected on any random shaped surface as long as the depth sensing camera can see it. This allows in particular simultaneous touch detection on diverse aligned surfaces.

[2] `http://opencvblobslib.github.io/opencvblobslib/`

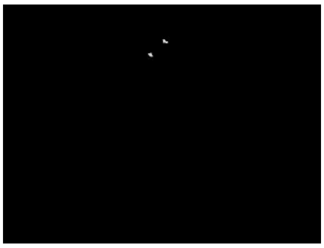


(a) Image with detected touch points

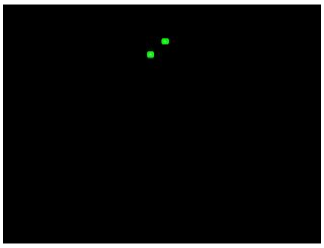


(b) Depth image for touch detection

(c) Potential contact points



(d) Filtered contact points



(e) Detected blobs

Figure 7.3: Touch detection sequence showing two touching fingers and one hovering hand.

Detected contact points are tracked over time to classify them into different touch events. As described prior events are touch down, touch move, touch release and long touch. The implemented tracker assigns the $time$ of the first appearance, $age = 0$ and $lives = 5$ to each new detected blob. In subsequent frames distances between previous blobs and the new blobs are calculated. If the distance is below a certain threshold, the age for this particular blob is increased. Additionally, the distance is cumulated and stored in $movement$. This is used for long touch detection. For all previous blobs, where no corresponding blob is found, the $lives$ is decreased. All new blobs are handled like described before. Touch events are differentiated as following:

$$event = \begin{cases} \text{touch down,} & \text{if } age = 3 \\ \text{long touch,} & \text{if } age > 3 \wedge movement < thld_{move} \wedge time_{elapsed} = thld_{time} \\ \text{move,} & \text{if } age > 3 \\ \text{touch release,} & \text{if } age \geq 3 \wedge lives = 0 \\ \text{no event,} & \text{otherwise} \end{cases}$$

Where $thld_{move}$ is the threshold for the maximum allowed distance the finger can drift whereby a long press is still detected and $time_{elapsed}$ is the time a long press takes.

The position of a touch event is detected in the depth sensing camera coordinate system. Whereas all GUI elements also known as widgets are located in the projector coordinate system. Hence, event coordinates need to be transformed to the projectors coordinate system. This transformation procedure is outlined in the next section.

7.3 Calibration and Transformations

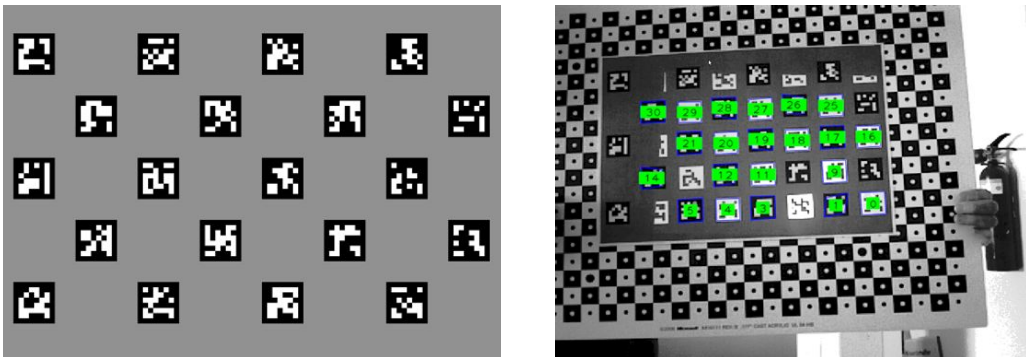
Mapping the detected touch event coordinates from the camera coordinate system to the widget, displaying the content, is necessary to enable interaction at all. Therefore, the touch event point $P_{x,y}$ detected in the camera coordinate system is transformed several times until the point is located in the same coordinate systems as the widget. In detail, the touch point is transformed from the depth sensing camera coordinate system to the colour camera coordinate system. From there to the world coordinate system which is the only three dimensional coordinate system in the transformation process. From the world coordinate system, the point can be transformed to the projectors coordinate system. There it is transformed one last time to the widgets coordinate system since it is pre-warped. This many transformations are indispensable because only the transformation matrixes between the named coordinate systems are known or can be determined due to manual one time calibration task.

7.3.1 Camera-Projector Calibration

The most challenging part is to determine the transformation between the depth sensing coordinate system and the projectors one. Xiao et al. [Xia+13] present an approach to determine the perspective transformation between a Kinect and a projector, but a special 3D-calibration target is needed. Hence, such a target was not accessible other approaches were evaluated. Simple colour-camera-projector calibration algorithms are developed by Legarda-Sáenz et al. [LS+04], Gao [Gao08] and Li et al. [Li+08] using structured light or Ashdown and Sato [AS05], Griesser and Van Gool [GVG06] and Audet and Okutomi [AO09] using optical pattern.

Using the colour camera for camera projector calibration requires to transform the touch point from the depth sensing coordinate system to the one of the colour camera. Despite that, an additional concept for calibration presented by Audet and Okutomi [AO09] was chosen. Especially because he provides a software, ProCamCalib[3], which performs a "full calibration of a camera and a projector in about 30 second" by preserving comparable good results as the preceding stated concepts.

To use the ProCamCalib software a special *framegrabber* was implemented since the software was originally developed for simple web-cameras and not for depth sensing cameras. The OpenNI framework was used to read colour frames from the PrimeSense camera. The frames were then transformed in a ProCamCalib readable format.



(a) Printed barcode pattern

(b) Calibration process

Figure 7.4: Projector camera calibration with printed and projected barcode pattern

Indeed, the calibration task itself is really simple. The printed pattern containing fiducial markers (shown in Figure 7.4a) enables the software to calculate the intrinsic parameters of the camera. Then the projector starts to project an inverted marker pattern into the spaces of the printed one which the camera tries to recognise (see Figure 7.4b) simultaneously. The intrinsic parameters of the projector can be calculated when the projected pattern is detected. Knowing this information the software takes several pictures showing the pattern in different poses and calculates the extrinsic parameters which reflect the relation between the camera and the projector. With this information touch points can be transformed from the camera coordinate system into the 3D world coordinate system and then into the projectors one. The calculated calibration file for the built PROCAMS is printed in section A.1.

[3] `http://www.ok.ctrl.titech.ac.jp/~saudet/research/procamcalib/`

7.3.2 Transformation

As mentioned, several transformations are necessary to calculate the position of the touch event represented in the widgets coordinate system. In the following, the executed transformations are described in detail.

Depth Sensing Camera - Colour Camera

The transformation between depth sensing camera and colour camera is necessary since the calibration was performed between colour camera and projector. Fortunately, the OpenNI framework in combination with the used PrimeSense camera allows a direct mapping of the depth sensing camera image to the colour camera image. This is done internally by the camera hardware so no computational performance is wasted.

Colour Camera - World

The 3×4 camera matrix describes the mapping from 3D points in world space to 2D points in a camera image. It is identified in the calibration task. Using the inverse camera matrix as well as the camera distortion coefficients which contains the intrinsic parameters allow in combination the distance from the depth sensing camera a retransformation of the touch point into the 3D world coordinate system. For matrix multiplication and rectification, OpenCV library is used.

World - Projector

Transforming the point from the 3D world coordinate system into the projector space is the inverse operation to the colour camera - world transformation using the projectors parameter. This time the projector matrix containing the alignment of the projector in the world coordinate system as well as the projectors distortion coefficients are used to calculate to perspective projection of the 3D point onto the projector image. This operation is also performed using the OpenCV library.

Projector - Widget

Since the widget is pre-warped to enable a rectified projection the touch point needs to be transformed once again. The transformation of the widget is set out in subsection 7.5.2. Since the perspective transformations matrix which is applied to the widget is known, the touch point can be transformed in the same manner.

Finally, the coordinate of the touched point is known and the interaction can be performed as designated by the widget.

7.4 Pan-Tilt Unit Control

The user can execute the initial alignment of the PROCAMS remotely. An Android App was implemented to provide basic control of the pan-tilt unit. The App UI is shown in Figure 7.5. It consists of a virtual joystick and four buttons for storing and loading surface configurations as well as starting interaction and recalibrating. Moving the joystick on the display triggers the pan-tilt unit to move the PROCAMS to the desired direction. Meanwhile, the projector is rendering a frame indicating the resulting *interaction space*

Figure 7.5: Android App providing control over the PROCAMS pan-tilt unit via joystick

The App connects via Internet to the server running on the PROCAMS. The server receives the movement commands (up, down, left, right) and forwards them via serial interface to the connected Arduino microcontroller which handles the movement of the servos. The check button triggers the self-calibration task for the touch detection and starts the system for user interaction. The recalibrate button provides the opportunity to repeat the cal-

ibration task if for example the touch de-
tection ground truth run out of sync due to unintentional movement of the PROCAMS. Pressing the store button asks the user to provide a name for the current configuration. The store command and the name are then transmitted to the server which stores name, current position of the PROCAMS as well as the position of the widgets in a database. On pressing the load button all stored positions are loaded from the server and displayed to the user for selection.

7.5 User-Interface

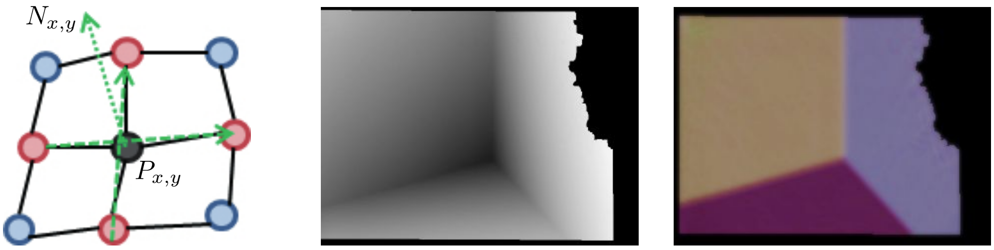
After setting up the alignment of the PROCAMS and self-calibration of the touch detection is done the user can start placing widgets. The interaction space is still indicated by a light white frame.

The projector is usually not aligned perpendicular to the surface it is projecting to. This causes geometrical distortion. Rendering a representation of the widgets without distortion onto the surfaces requires a pre-warping according to the alignment of the projector to the surface.

7.5.1 User-Interface Pre-Warping

For the proposed PROCAMS, the already available depth map, the ground truth calculated for touch detection, is used to facilitate a rectified projection. First a plane detection is executed on the depth map to find possible projection surfaces. Then four points situated on one of the detected planes, spanning a rectangle of the desired size are determined. Finally, the affine transformation which transforms the widget to the determined points is calculated and applied to render a corrected representation of the widget. These three steps will now be described in detail.

Plane Detection Plane detection is done following the concepts Yoo et al. [Yoo+13] presented in their work. To compute the surface normal of a point $P_{x,y}$ in the ground truth depth map, four points with a distance d around the point are selected. Up: $P_{x,y-d}$, Down: $P_{x,y+d}$, Left: $P_{x-d,y}$ and Right: $P_{x+d,y}$. d defines the amount of smoothing. A greater d will flatten the surface normal since it is calculated over a larger area. These four points are transformed into world space and the vectors between up-down and left-right are determined (see Figure 7.6a). The cross product of these two vectors is finally the surface normal $N_{x,y}$ of the point $P_{x,y}$.



(a) Surface normals calculation scheme [Yoo+13]

(b) Depth map of scenery

(c) Colour encoded surface normals

Figure 7.6: Plane detection

Surface normals are directly calculated after determination of the ground truth in the self-calibration task. To minimise processing time surface normals are only calculated over a grid with a distance of 4x4 pixel. For a given ground truth like illustrated in Figure 7.6b the corresponding surface normals are presented in Figure 7.6c. The length of each direction (X,Y,Z) of the normal vector $N_{x,y}$ is encoded via a corresponding colour (red,green,blue). Hence, surface normals pointing in the same direction have the same colour.

Rectangle Calculation When the user presses long at the surface to add a new widget, the position as well as the surface normal in camera-space is known. Both define a plane where the widget should appear rectified. Therefore, the plane (point and normal) is transformed into world space. Based on the normal vector, two vectors x and y each perpendicular to each other and to $N_{x,y}$ are calculated. Hence the alignment of the pan-tilt unit is known x and y can be rotated in that way, that y points down and x is horizontal in the 3D coordinate system. The calculated vectors are displayed in red and blue in Figure 7.7.

Knowing the two vectors, allows to calculate the four vertexes of the rectangle where the widget should appear. The four points can be calculated as follows:

$$
\begin{aligned}
\mathsf{P}_0 &= P_{x,y} \\
\mathsf{P}_1 &= P_{x,y} + w \times x \\
\mathsf{P}_2 &= P_{x,y} + h \times y \\
\mathsf{P}_3 &= P_{x,y} + w \times x + h \times y
\end{aligned}
$$

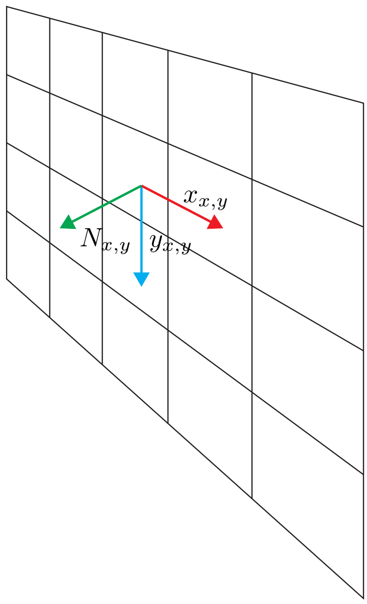


Figure 7.7: Surface normals calculation in world-space

Where $P_{x,y}$ is the point the user touched, w the width and h the height of the desired widget. These four points in the 3D world coordinate system are then transformed into the projector coordinate system in order to calculate the perspective transformation for the widget.

Projective Transformation Projective transformation allows creating perspective distortion. This is needed to change the perspective of the projected content in order to get an undistorted representation of the content. The conceptual result of a projective transformation is visualised in Figure 7.8.

Projective transformation are realised by simple matrix multiplication:

$$\begin{pmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ c_1 & c_2 & 1 \end{pmatrix} \times \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$

Where a_1 to a_4 is the rotational matrix which performs scaling and rotation, t_1 and t_2 is the translation vector, moving the content in space, and c_1 and c_2 is the projection vector. x and y are the source positions which are projected to x' and y'.

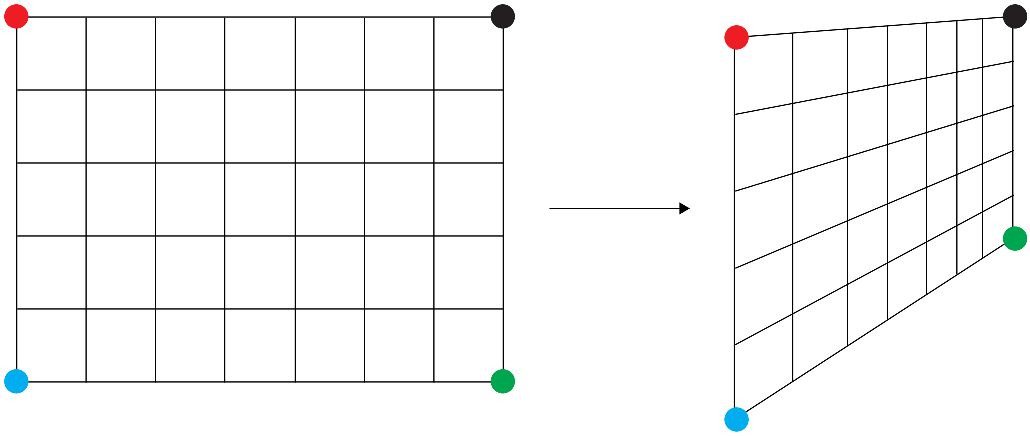
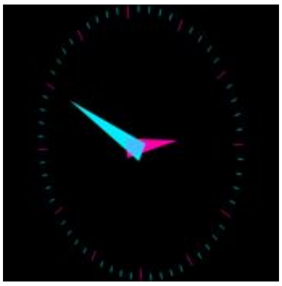


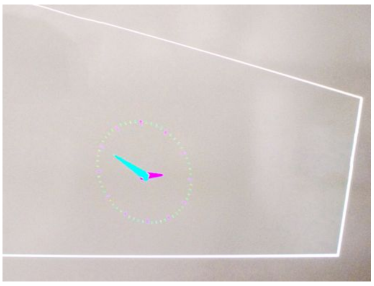
Figure 7.8: Projective transformation

With the four prior calculated and transformed points it is possible to determine the projective transformation matrix so that the vertex of the content e.g. the widget with defined vertex points at position (0,0), (1,0), (0,1) and (1,1), are transformed to the calculated ones. The transformation of an exemplary clock widget as well as the projected rectified result is shown in Figure 7.9. For calculating the transformation matrix the *QTransformation* class provided by Qt was utilised. Calculation starting from the ground truth is done fully automatic without any interaction necessary by the user.

(a) Simple clock Widget



(b) Pre-warped widget



(c) Widget projected on surface

Figure 7.9: Pre-warping of a widget

7.5.2 Widgets

The developed framework allows dynamic loading of widgets which actually displays the content to the user. All the complexity of the spatially aware projection, dynamic touch detection and movement of the PROCAMS are encapsulated and hidden from the view of the widget. This enables trouble-free and straight forward widget development.

Two different possibilities are supported to create a new widget. Developers are able to implement the provided interface depicted in Listing A.1 to create a more desktop like looking widgets. The interface inherits from *QWidget* the base class of all Qt user interfaces. The interaction logic can directly be implemented in the corresponding methods. In the *paintEvent* method, the projected representation of the widget must be implemented. The core logic of the widget can be implemented as desired, for example by following the model view controller pattern.

Alternatively, developers can implement widgets using Qt User Interface Creation Kit (Qt Quick). It uses QML to describe modern looking, fluid UIs in a declarative manner. Qt Quick is increasingly used on mobile phones or other embedded hardware. QML allows to develop the logic using Javascript or even native code. Simplicity is shown by the source code (printed in Listing A.2) for a plain widget showing a scrollable web page. In both cases, widgets can easily be developed and tested in a common desktop environment by simply deploying them as a desktop application. This allows rapid creation of rich widgets without the need of the PROCAMS nearby. In addition, there are a lot of tutorials, demos and sample widgets provided by Qt Project [DQ14] which all run directly or after just small modifications in the framework.

(a) Timetable widget (b) Google News widget (c) Qt clock widget

Figure 7.10: Some implemented widgets

Three widgets were implemented. A simple digital image frame which shows dynamically loaded images. On touching the left or right part of the image, the next or previous image is presented. The second widget is an adaptive bus time table (see Figure 7.10a) which shows the next buses leaving a desired bus stop. The last implemented widget is a news browser (see Figure 7.10b), showing the Google news website. Any other web page could be displayed by this widget as well. In addition, three widgets provided as demos by Qt Project were slightly modified to work with the proposed framework. The first one is a simple clock showing the time (see Figure 7.10c), the second one is a sophisticated image browser allowing to browse the flickr.com image library in a convincing way. The third adapted widget shows a Twitter feed. There are many more examples available offering great services. Most of them can directly deployed into the developed framework. The three exemplary chosen widgets conduce just as proof of concept.

8 Evaluation

To validate the fineness and quality of the proposed PROCAMS a technical evaluation was executed. In particular, the precision and speed of the pan-tilt unit were examined as well as the touch accuracy. Finally, overall performance of the PROCAMS is evaluated. While planning the execution of larger usability studies in a domestic environment this evaluation should give a first clue of strengths and weaknesses of the build PROCAMS.

8.1 Pan-Tilt Unit Performance

The task of the pan-tilt unit is to move the PROCAMS fast and accurate to a desired location. This two properties accuracy and pace were assessed in a laboratory study.

8.1.1 Alignment Accuracy

The accuracy approaching a previously stored position was determined by placing the PROCAMS with a distance of 1 m to a wall. The projector was displaying a red cross to indicate the centre of the projection. Then the pan-tilt unit was commanded to approach the stored position from eight defined starting points. The position where the red cross came to a standstill was marked at the wall. Starting points were up, up-right, right, right-down, down, down-left, left, and left-up. Where up and down indicates a vertical shift by 45° from the stored position. Accordingly left and right indicates a horizontal shift by 90°. The measured distances in horizontal a vertical direction between the marked and stored position lead to an angel of aberration by simple trigonometry. The stored position was approached ten times from each starting point. Thus 80 data points were obtained. A plot of the data is shown in Figure 8.1.

The average horizontal misalignment is 3.29°. For vertical alignment, the average error is 1.48°. Hence, the misalignment in an arbitrary direction is 3.74°. This accords to a shift of less than 10 cm if the *surface* is 150 cm away from the projector. A likely reason for

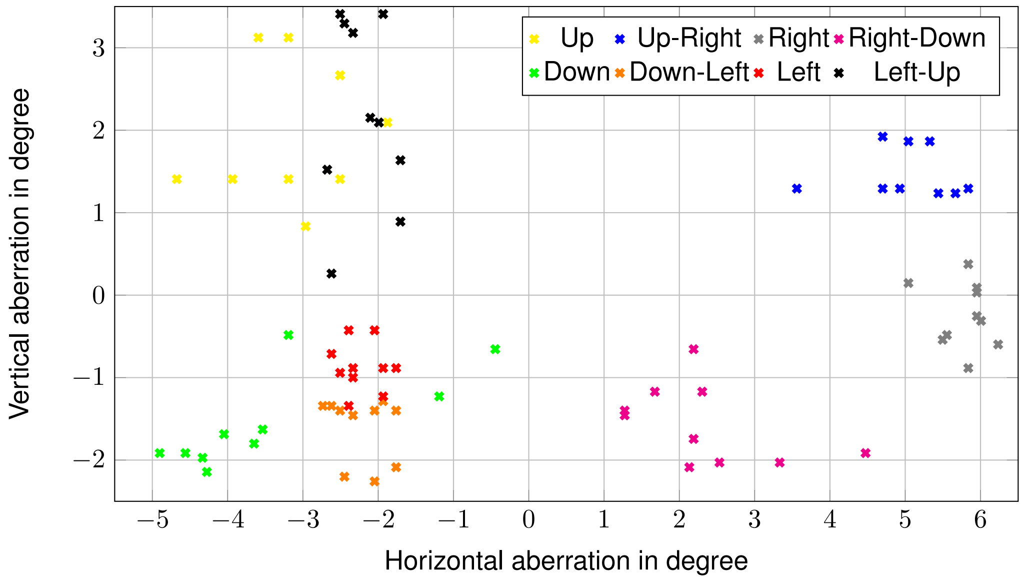


Figure 8.1: Aberration of the pan-tilt unit approaching a defined location

the smaller misalignment in the vertical direction is caused by an additionally used accelerometer to control the servo for horizontal alignment. Since, for horizontal alignment no secondary sensor is used, the alignment is not as good. Overall the alignment is fair enough to re-project a widget at almost the same location in the physical world, but is not sufficient enough to augment small tangible objects as for example a light switch.

A more accurate alignment could be achieved by two different modifications. On the one hand, a sensor providing very accurate data for horizontal and vertical alignment could be attached to provide feedback of the current alignment of the pan-tilt unit. Nonetheless, previous attempts already showed that a simple calibrated and compensated compass is not suitable for this task since the projector and servos have a strong influence on the magnetic field.

On the other hand, more powerful servos with a high resolution potentiometer could be installed. The potentiometer would lead the servo to move more precisely to the commanded position. This approach seems very promising without any major change at the pan-tilt unit.

8.1.2 Alignment Pace

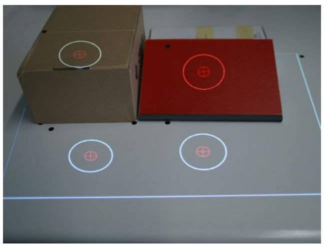
The pace of the pan-tilt unit was evaluated in a separate benchmark. Therefor, the time needed for 164° horizontal pan and a 110° tilt was measured. Each movement was repeated ten times from both directions. Since panning and tilting is performed simultaneously no combinations of tilt and pan were executed.

On average the pan-tilt unit needed 3.5 s for the horizontal pan task. For the tilt task, the unit needed 4.8 s. A reason for the slower tilt movement could be the higher force needed for tilting compared to the rotation force. Overall the PROCAMS can reach every position in less than 6 s (worst-case: move 135° vertically). This seems to be a decent time. Of course, there are faster servos available, but higher acceleration forces could damage the printed case holding the PROCAMS.

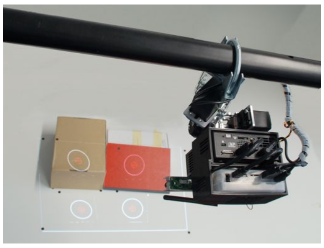
8.2 Touch Performance

Touch performance was evaluated in a basic laboratory study. The PROCAMS was mounted over a desk in a distance of 75 cm. It was tilted down 70° from horizontal, pointing at the desk illuminating an *interaction space* of 40 cm × 30 cm. The setup is shown in Figure 8.2b. Four red crosses surround by a white circle posed as target. They were distributed on three different *surfaces*. Two targets at the desk, one at the cardboard box on the left side and one on a ramp composed of a red notebook. The four targets are depicted in Figure 8.2a. In all cases, the diameter of the red cross was 18 mm.

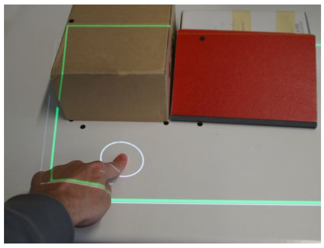
During the study participants had the task to touch the targets as accurately as possible. Participants were ordered to take as much time as needed. Forty targets were presented



(a) Four target positions for touch evaluation



(b) Evaluation setup: PROCAMS and touch targets

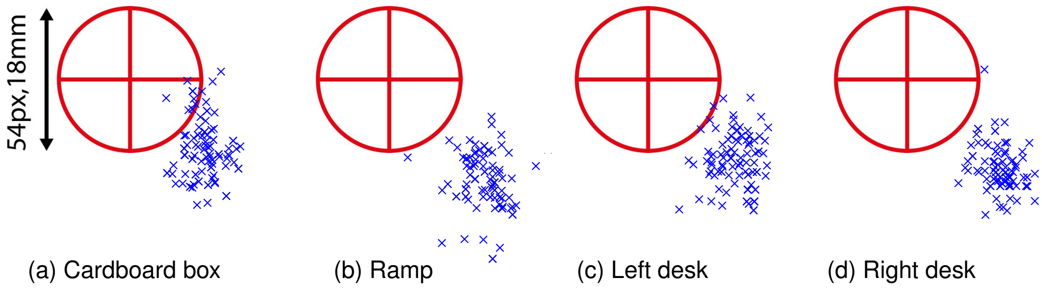


(c) Green border indicates a detected touch

Figure 8.2: Touch accuracy evaluation setup

in a counterbalanced order, one at a time. A detected touch was indicated by a green border (see Figure 8.2c). After touching the target, it disappeared and appeared at one of the three other positions. Time as well as touch position in projector and world coordinate system were monitored. From that data, the error in mm in the world coordinate system can be derived.

Ten participants between 24 and 27 years took part in this study. Hence, 400 touch events were monitored. On average participants needed 109 s to touch all 40 target. In less than 1% the touch was not detected on the first approach. This was counted manually. Distribution of monitored touch events for each target is shown in Figure 8.3. The targets are labeled as follows: cardboard box (T1), ramp (T2), left desk(T3) and right desk (T4).



(a) Cardboard box (b) Ramp (c) Left desk (d) Right desk

Figure 8.3: Touch accuracy evaluation

Collected touch data was analysed with a standard weighted-means ANOVA test (F=57.06, p<0.0001) in conjunction with a Tukey's honest significance test. The mean touch error, variance and standard deviation for the different targets is specified in Table 8.1.

In this study setup, T1 has a significantly smaller error than T2, T3 and T4 (p<0.01, HSD[.01]=1.3). This coincides with Hardy and Alexander [HA12] results, who identify that the accuracy of touch detection mainly depends on sensor distance. Target 1 was closest to the camera. The significantly worse result of T2 in comparison to T3 and T4 (p<0.01, HSD[.01]=1.3) cannot be explained by distance since it was very similar. A likely reason could be the material and colour of the notebook, but further investigations are necessary.

A mean error of less than 20 mm requires large buttons for pleasant interaction. However, the small standard deviation for all targets is remarkable. It appears that for the evaluation setup a fixed offset between executed and detected touch was present. It is necessary to figure out how and where this offset emerged and if the offset could be correct. The offset could accrue at several positions in the processing pipeline. First of all a bad calibration

Target	T1	T2	T3	T4
Mean (in mm)	14.1104	19.3232	16.5835	17.82
Variance	8.4757	12.4966	8.5013	7.5729
Std. deviation (in mm)	2.9113	3.5351	2.9157	2.7519
Significant versus	T2,T3,T4	T3,T4		

Table 8.1: Statistical data

file between the projector and camera could lead to errors. Although, this is unlikely since a new calibration file was created before the evaluation. A likely reason for non optimal detection could be the different field of views of projector and depth sensing camera. The projected image fills only an area of approximately $340\,\mathrm{px} \times 220\,\mathrm{px}$ which is roughly $1/4$ of the available resolution. To improve this issue a projector with a short throw lens or a camera with zoom lens should be used to adapt the field of views. Finally, a non-conform transformation matrix could result in the error. A source code recheck could clarify this concern.

8.2.1 Touch Pace

Even though the depth sensing camera runs at 30 fps on average only 21.8 fps are processed. In particular, processing limitations and the Qt event-queue are responsible for that loss of frames. Due to the used tracking users must press a target for at least three frames which is equivalent to $0.138\,\mathrm{s}$. In the study it was only problematic for the first one or two targets since participants know instant detection from their smart phones. After explaining to interact slightly slower participants could interact in an enjoyable way.

8.3 Overall Performance

Overall, the proposed PROCAMS performs well. Widgets are rendered with up to 30 fps. As mentioned before, touch detection runs at almost 22 fps. This performance remains also when two to three individual widgets are added to different *surfaces*. Adding more widgets will decrease the available processing power and touch detection slows down to 10 to 14 fps. This forces the user to interact in an unnatural slow way. For more synchronous rendered content alternative rendering techniques or more processing power are required.

8.4 Conclusion

To assess the quality and performance of the build PROCAMS an evaluation was conducted. Pan-tilt unit and touch detection were analysed in separate studies. The pan-tilt unit fulfils the requirements and can approach a stored position in a short space of time. However, accuracy of the servos could be more accurate. Numerous possibilities of improvement were discussed.

Touch detection accuracy is very reasonable with a small standard deviation of less than 3.5 mm. Touch accuracy is comparable to other PROCAMS requiring a manual calibration task. Enabling touch detection on arbitrary surfaces without any setup specific calibration was a tough task. Mapping of detected touch points back to the projection makes is even more prone to failure. From this point of view and considering inexpensive hardware the touch accuracy is very auspicious.

9 Conclusion

In the course of this thesis, a projector camera system was designed, built and evaluated. The built PROCAMS can be deployed in domestic environments to make research beyond the laboratory possible. Previously published PROCAMS were heavy, bulky and required manual calibration before touch interaction and rectified projection of content was possible. In contrast, in this thesis a light, stand-alone and rotatable prototype was developed which does not require any calibration to be executed by the user.

The proposed PROCAMS consists of a PrimeSense Carmine 1.09 depth sensing camera for touch input detection, a small light-intensive projector as well as a powerful single board computer, the ODROID-XU. The volume of composed parts is less than $1\,\mathrm{dm}^3$ and the whole unit weighs less than $1\,\mathrm{kg}$. All parts are attached to a small inexpensive self-built pan-tilt unit. The unit is ceiling-mounted and enables the PROCAMS to project information in any direction. As an input method, a light-weight touch detection algorithm was implemented, using images of the depth sensing camera combined with a tracking algorithm.

As part of the designing process semi-structured interviews were conducted. Eighteen potential end users were visited at their homes and realistic setups were built with a PROCAMS mock-up, capable of projecting static user interfaces. Participants used the mock-up in particular to replace already existing technical things or displays such as wall paintings, televisions, clocks, printed bus timetables or a set of remote controls. They strongly preferred to project onto flat surfaces even after clarifying that it would be possible to project without distortion to irregular surfaces.

A generic software and hardware architecture design was elaborated. Creating a configuration free, movable PROCAMS offering touch input on arbitrary surfaces was the main focus while developing a small and light system from scratch. For displaying graphical user interface elements, the Qt library was used as it supports rapid prototyping. Several widgets were implemented for different scenarios such as an image viewer, bus timetable or news browser. Due to the software architecture, already existing Qt widgets are effortlessly integrated into the system. Implementing a pre-warping technique allows projection onto surfaces from any projector alignment so that it appears correct to the observer. Therefore, a spatial model based on the depth sensing camera is calculated on demand. The movement of the PROCAMS is controlled via a remote. An Android App containing a virtual joystick enables the user to control the pan-tilt unit. Furthermore, the App allows the user to store and reload previous configurations.

Finally, a technical evaluation was conducted. The accuracy of touch input as well as the pace and fidelity of the built prototype were evaluated. The built PROCAMS provides a very reasonable touch accuracy. Considering its limited processing power, the prototype has a good overall performance and users were able use it in an enjoyable way.

9.1 Challenges

Using inexpensive hardware makes an extensive deployment possible. Unfortunately, the quality and performance of the used hardware are not particular valuable. Especially the used depth sensing camera provides only very noisy images. One big challenge was to filter the noise to enhance the image quality, as this directly correlates with improved touch accuracy and reduced false touch events. A huge hurdle was to accomplish this filtering as well as the touch detection, while keeping the time between the actual touch and reaction by the system small. The limited processing resources make this task even more challenging.

Another challenge was the development of the hardware construction due to a lack of previous experience in designing and printing parts as well as the control and proper use of actuators. It took several attempts until the design was robust and well balanced enough that the unit operates in a decent manner. Furthermore, there were many other minor and major challenges such as camera calibration or transformation of touch events, until the system finally manifested in its current accurate and usable form.

9.2 Future Work

A basic platform for everywhere information rendering and touch interaction was created. However, there are several areas which remain open to further development. On the hardware side, some components could be replaced to gain a higher performance and less noise. The servo responsible for projector focus should be replaced, as it is very loud. Furthermore, replacing the servos of the tilt-pan platform by high precision servos would accomplish a higher accuracy when approaching a commanded alignment. In a final step, the fan of the SBC could be replaced to obtain an almost silent operation of the PROCAMS.

On the software side, one should consider which input methods should be supported in planned long-term studies. Only four simple touch interactions were implemented, but the used hardware is capable of providing much more interaction possibilities. The implemented framework allows for an easy integration of new interaction or input methods. Several more touch interactions like multi-touch or swiping gestures are conceivable. Colour, size or quantity of touches as well as the expansion from surface interaction into free space interaction could be used for interesting new interaction concepts. As mentioned in the interviews, spoken commands could easily be integrated. A stereo microphone is already built-in which allows for a coarse localisation of the audio source. Another idea is to link the PROCAMS with the smart phone of the user to allow fast remote text input when speech recognition is not feasible.

The proposed PROCAMS finally allows the conduction of large long-term user studies in users' homes. Due to the inexpensive cost of production of the proposed PROCAMS, the accommodations could be equipped with several units. For the first time, it could be examined how everywhere displays are used in a domestic environment. Primarily the use of widgets and the preferred location for information placement, as well as interaction methods, could be evaluated. Another crucial aspect which could be reappraised is the social effect of everywhere projection, including its influence on privacy concerns. Furthermore, it would allow an examination of the most appropriate input method, how the movability of the system is used, and how users would interact if real world objects are augmented with information.

It is my hope that the proposed hardware and software framework can support the discovery of answers to these questions by enabling an easy deployment of the PROCAMS as well as the rapid application development for everywhere projections.

Acronyms

DLP	Digital Light Processing
DOF	degrees of freedom
fps	frames per second
FOV	field of view
GUI	graphical user interface
HCI	human computer interaction
ISA	instruction set architecture
LCoS	Liquid Crystal on Silicon
OpenNI	Open Natural Interaction
OS	operating system
PIC	peripheral interface controller
PLA	polylactic acid
PROCAMS	projector camera system
PWM	pulse-width modulation
Qt Quick	Qt User Interface Creation Kit
SBC	single board computer
SL	Structured Light
ToF	Time of Flight

Bibliography

[AO09] S Audet and M Okutomi. 'A user-friendly method to geometrically calibrate projector-camera systems'. In: *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*. 2009, pp. 47–54. DOI: 10.1109/CVPRW.2009.5204319.

[Ard] Arduino. *Arduino Language Reference*. URL: http://arduino.cc.

[AS05] M Ashdown and Y Sato. 'Steerable Projector Calibration'. In: *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*. 2005, p. 98. DOI: 10.1109/CVPR.2005.533.

[Bra+09] Robert Brauer, Lars Müller and Ilhan Aslan. 'Mobile projector-camera system based interaction technique for annotating physical information resources'. In: *Adjunct Proceedings UbiComp 2009*. 2009.

[DQ14] Digia and QtProject. *Qt Project*. 2014. URL: http://qt-project.org/ (visited on 10/01/2014).

[FO02] Jerry Alan Fails and Dan R Olsen. 'LIGHT WIDGETS'. In: *the 7th international conference*. New York, New York, USA: ACM Press, 2002, p. 63. DOI: 10.1145/502716.502729.

[FOR] Foundation for Research Technology-Hellas FORTH. *Kinect 3D Hand Tracking*. URL: http://cvrlcode.ics.forth.gr/handtracking/ (visited on 03/02/2014).

[Gao08] Wei Gao. 'Flexible method for structured light system calibration'. In: *Optical Engineering* 47.8 (2008), p. 083602. DOI: 10.1117/1.2969118.

[Gut+11] Karl M Guttag, Shawn Hurley and Bill Mei. '39.1: Distinguished Paper: Laser+ LCOS Technology Revolution'. In: *SID Symposium Digest of Technical Papers* 42.1 (2011), pp. 536–539. DOI: 10.1889/1.3621379.

[GVG06] A Griesser and L Van Gool. 'Automatic Interactive Calibration of Multi-Projector-Camera Systems'. In: *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on*. 2006, p. 8. DOI: 10.1109/CVPRW.2006.37.

[HA12] John Hardy and Jason Alexander. 'Toolkit Support for Interactive Projected Displays'. In: *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*. MUM '12. New York, New York, USA: ACM, 2012, p. 1. DOI: 10.1145/2406367.2406419.

[Har+11] Chris Harrison, Hrvoje Benko and Andrew D. Wilson. 'OmniTouch: Wearable Multitouch Interaction Everywhere'. In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. UIST '11. New York, New York, USA: ACM, 2011, pp. 441–450. DOI: 10.1145/2047196.2047255.

[Har12] John Hardy. 'Reflections: A Year Spent with an Interactive Desk'. In: *interactions* 19.6 (11/2012), p. 56. DOI: 10.1145/2377783.2377795.

[Int+] Intel, Willow Garage and Itseez. *OpenCV - Open Source Computer Vision*. URL: http://opencv.org/.

[KA12] Dong-Ik Ko and Gaurav Agarwal. *Gesture Recognition: Enabling Natural Interactions with Electronics*. 2012.

[Klo+12] Florian Klompmaker, Karsten Nebe and Alex Fast. 'dSensingNI: A Framework for Advanced Tangible Interaction Using a Depth Camera'. In: *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*. TEI '12. Kingston, Ontario, Canada: ACM, 2012, pp. 217–224. DOI: 10.1145/2148131.2148179.

[Li+08] Zhongwei Li, Yusheng Shi, Congjun Wang and Yuanyuan Wang. 'Accurate calibration method for a structured light system'. In: *Optical Engineering* 47.5 (2008).

[LM10] Natan Linder and Pattie Maes. 'LuminAR: Portable Robotic Augmented Reality Interface Design and Prototype'. In: *Adjunct Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*. UIST '10. ACM, 2010, pp. 395–396. DOI: 10.1145/1866218.1866237.

[LS+04] Ricardo Legarda-Sáenz, Thorsten Bothe and Werner P. Jüptner. 'Accurate procedure for the calibration of a structured light system'. In: *Optical Engineering* 43.2 (02/2004), pp. 464–471. DOI: 10.1117/1.1635373.

[Mer] Ron Mertens. *Introduction - What is a pico projector?* URL: http://www.picoprojector-info.com/introduction (visited on 03/04/2014).

[Mis+09] Pranav Mistry, Pattie Maes and Liyan Chang. 'WUW - Wear Ur World: A Wearable Gestural Interface'. In: *CHI '09 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '09. New York, New York, USA: ACM, 2009, pp. 4111–4116. DOI: 10.1145/1520340.1520626.

[Mol+12] David Molyneaux, Shahram Izadi, David Kim, Otmar Hilliges, Steve Hodges, Xiang Cao, Alex Butler and Hans Gellersen. 'Interactive environment-aware handheld projectors for pervasive computing spaces'. In: *Pervasive'12: Proceedings of the 10th international conference on Pervasive Computing*. Springer, 06/2012, pp. 197–215.

[MW09] Daniel C. McFarlane and Steven M. Wilder. 'Interactive Dirt: Increasing Mobile Work Performance with a Wearable Projector-camera System'. In: *Proceedings of the 11th International Conference on Ubiquitous Computing*. Ubicomp '09. New York, New York, USA: ACM, 2009, pp. 205–214. DOI: 10.1145/1620545.1620577.

[Opt] Light Blue Optics. *Light Blue Optics*. URL: http://lightblueoptics.com/.

[Per] Michelle Perkins. *CES 2010: Light Blue Optics launch "Light Touch" | Digital Media Blog*. URL: http://blogs.adobe.com/digitalmedia/2010/01/ces_2010_light_blue_optics_lau/.

[Pin01] Claudio Pinhanez. 'The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces'. In: *Proceedings of the 3rd International Conference on Ubiquitous Computing*. UbiComp '01. Springer-Verlag, 2001, pp. 315–331.

[Pra] Mistry Pranavv. *SixthSense - a wearable gestural interface (MIT Media Lab)*. URL: http://www.pranavmistry.com/projects/sixthsense/ (visited on 03/12/2014).

[Pre94] Jenny Preece. *Human-computer interaction*. Addison Wesley Publishing Company, 1994.

[Pri+] PrimeSense, Garage Willow and ASUS. *OpenNI* - Open Natural Interaction. URL: http://www.openni.org/ (visited on 10/11/2013).

[Ras+05] Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao and Clifton Forlines. 'iLamps: Geometrically Aware and Self-configuring Projectors'. In: *ACM SIGGRAPH 2005 Courses*. SIGGRAPH '05. New York, New York, USA: ACM, 2005, p. 5. DOI: 10.1145/1198555.1198715.

[Ruk+12] E. Rukzio, P. Holleis and H. Gellersen. 'Personal Projectors for Pervasive Computing'. In: *Pervasive Computing, IEEE* 11.2 (04/2012), pp. 30–37. DOI: 10.1109/MPRV.2011.17.

[Tam+09] Emi Tamaki, Takashi Miyaki and Jun Rekimoto. 'Brainy Hand: An Ear-worn Hand Gesture Interaction Device'. In: *CHI '09 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '09. New York, New York, USA: ACM, 2009, pp. 4255–4260. DOI: 10.1145/1520340.1520649.

[Und+99] John Underkoffler, Brygg Ullmer and Hiroshi Ishii. 'Emancipated Pixels: Real-world Graphics in the Luminous Room'. In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '99. New York, New York, USA: ACM, 1999, pp. 385–392. DOI: 10.1145/311535.311593.

[WB10] Andrew D. Wilson and Hrvoje Benko. 'Combining Multiple Depth Cameras and Projectors for Interactions on, Above and Between Surfaces'. In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. UIST '10. ACM Request Permissions, 10/2010, pp. 273–282. DOI: 10.1145/1866029.1866073.

[Wei+13] Ben Weissmann, Natan Linder, Rony Kubat and Pattie Maes. 'Lens: A Javascript SDK for Building Web-Based Augmented Reality Applications on the LuminAR Platform.' In: *EECScon*. Cambridge, 04/2013.

[Wei91] Mark Weiser. 'The Computer for the 21st Century'. In: *Scientific American* 265.3 (09/1991), pp. 94–104. DOI: 10.1038/scientificamerican0991-94.

[Wel91] Pierre Wellner. 'The DigitalDesk Calculator: Tangible Manipulation on a Desk Top Display'. In: *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology*. UIST '91. Hilton Head, South Carolina, USA: ACM, 1991, pp. 27–33. DOI: 10.1145/120782.120785.

[Wel93] Pierre Wellner. 'Interacting with paper on the DigitalDesk'. In: *Communications of the ACM* 36.7 (1993), pp. 87–96.

[Wil+12] Andrew Wilson, Hrvoje Benko, Shahram Izadi and Otmar Hilliges. 'Steerable Augmented Reality with the Beamatron'. In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST '12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 413–422. DOI: 10.1145/2380116.2380169.

[Wil10] Andrew D. Wilson. 'Using a Depth Camera As a Touch Sensor'. In: *ACM International Conference on Interactive Tabletops and Surfaces*. ITS '10. Saarbrücken, Germany: ACM, 2010, pp. 69–72. DOI: 10.1145/1936652.1936665.

[Win+11] Christian Winkler, Christian Reinartz, Diana Nowacka and Enrico Rukzio. 'Interactive Phone Call: Synchronous Remote Collaboration and Projected Interactive Surfaces'. In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS '11. Kobe, Japan: ACM, 2011, pp. 61–70. DOI: 10.1145/2076354.2076367.

[Win+14] Christian Winkler, Julian Seifert, David Dobbelstein and Enrico Rukzio. 'Pervasive Information Through Constant Personal Projection: The Ambient Mobile Pervasive Display (AMP-D)'. In: *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*. CHI 2014. ACM, 2014, pp. 4117–4126. DOI: 10.1145/2556288.2557365.

[Xia+13] Robert Xiao, Chris Harrison and Scott E. Hudson. 'WorldKit: Rapid and Easy Creation of Ad-hoc Interactive Applications on Everyday Surfaces'. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 879–888. DOI: 10.1145/2470654.2466113.

[Yoo+13] Hyun Woo Yoo, Woo Hyun Kim, Jeong Woo Park, Won Hyong Lee and Myung Jin Chung. 'Real-time plane detection based on depth map from Kinect'. In: (2013), pp. 1–4. DOI: 10.1109/ISR.2013.6695732.

[OM] OM Interactive Ltd. *OM Interactive: omiVista Interactive Projection*. URL: http://om-interactive.com/interactive_projection.html.

A Appendix

A.1 Camera Projector Calibration

```
Camera  PrimeSense (1280 x 960)
=========
Intrinsics
----------
camera matrix = [ 1089.4766, 0.0, 643.72626
                  0.0, 1086.0332, 527.50134
                  0.0, 0.0, 1.0 ]
distortion coefficients =
 [ 0.03148991, -0.18659991, -0.0018285413, 0.003724281 ]
reprojection RMS/max error (pixels) = 0.9886399 / 2.3826725

Extrinsics
----------
rotation = [ 1.0, 0.0, 0.0
             0.0, 1.0, 0.0
             0.0, 0.0, 1.0 ]
translation = [ 0.0
                0.0
                0.0 ]
epipolar RMS/max error (pixels) = 0.0 / 0.0

Color
-----
order = BGR
mixing matrix = null
additive light = null
normalized RMSE (intensity) = 0.0
R^2 (intensity) = 1.0
```

```
Projector  Optoma (1280 x 800)
============
Intrinsics
----------
camera matrix = [ 1969.6365, 0.0, 675.2177
                  0.0, 1956.7374, 789.2852
                  0.0, 0.0, 1.0 ]
distortion coefficients =
[ -0.0133327255, -0.10250233, -0.0037307362, 0.005952384 ]
reprojection RMS/max error (pixels) = 0.50833434 / 1.6034822

Extrinsics
----------
rotation = [ 0.99992317, 0.011167082, -0.005384094
             -0.011341835, 0.999371, -0.03359995
             0.0050054938, 0.033658434, 0.9994209 ]
translation = [ 403.57474
                -277.41336
                -198.09766 ]
epipolar RMS/max error (pixels) = 0.5259377 / 1.6386813
```

A.2 Interview Participants

Participant	Gender	Age	Job	Smart-phone	Touch Dev.	Gesture Dev.	Beamer present	Size of Flat	rooms	house-mate	common room
P1	m	26	student	y	y	y	n	72	3 + c	3	y
P2 - 1	m	26	student	n	n	y	n	55	3 + c	2	y
P3 - 2	f	26	digital media designer	y	n	y	n	55	3 + c	2	y
P4 - 1	f	31	Medical laboratory scientist	n	n	y	n	75	3 + c	2	y
P5 - 2	f	28	PHD Student	y	n	y	n	75	3 + c	2	y
P6	m	27	Student	y	y	y	n	27	1	1	n
P7 - 1	m	26	PHD Student	y	n	n	n	84	3	2	n
P8 - 2	f	31	Psychotherapist	y	n	y	n	84	3	2	n
P9	m	27	Student	y	n	n	n	77	25 + c	2	y
P10	m	25	Student	y	y	y	n	60	2+c	2	n
P11	f	22	Student	y	n	y	n	60	2 + c	2	n
P12	f	22	Student	y	n	y	n	27	1	1	n
P13	m	24	Student	y	n	y	y	68	3 + c	2	y
P14	m	26	Student	y	y	n	n	100	4 + c	4	y
P15	f	25	Student	n	n	n	n	55	3 +c	2	y
P16	f	26	Student	y	y	n	n	55	2 + c	2	n
P17	m	58	Businessman	y	n	n	n	104	4	2	y
P18	f	55	independent gentleman	y	y	n	n	104	4	2	y

Table A.1: Participants taking part in the interviews.
f: female; m: male; y: yes; n: no; c: corridor

A.3 Abstract Widget Class

```

#include <QWidget>
#include <QMouseEvent>

class ProcamsWidget : public QWidget
{
    Q_OBJECT
public:
    virtual void paintEvent(QPaintEvent *) = 0;

protected:
    virtual void touchPressEvent(QMouseEvent *event);
    virtual void touchReleaseEvent(QMouseEvent *event);
    virtual void touchMoveEvent(QMouseEvent *event);
    virtual void touchLongClickEvent(QMouseEvent *event);
};
```

Listing A.1: Abstract Widget Class

A.4 QML Widget

```
import QtQuick 1.1
import QtWebKit 1.0

Flickable {
  property bool flipped: false
  property string urlVar: "https://news.google.com/"

id:flicker
width: 750
height: 500
contentWidth: web.width
contentHeight: web.height

  WebView {
      id: web
      url: urlVar
      preferredWidth: 750
      preferredHeight: 500
      scale: 1
      smooth: false
  }
}
```

Listing A.2: QML web widget

Name: Pascal Knierim Matrikelnummer: 651625

Erklärung

Ich erkläre, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den ....

Pascal Knierim