TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Mensch-Maschine-Kommunikation

Stochastic Optimisation Methods and Pattern Search Algorithms for Augmented Reality Videoconferencing

Nicolas H. Lehment

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	apl. Prof. Dr.-Ing. Walter Stechele
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. habil. Gerhard Rigoll
	2. Prof. Bernard Merialdo, EURECOM / Frankreich

Die Dissertation wurde am 16.04.2015 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 16.11.2015 angenommen.

Abstract

This dissertation introduces a novel system for collaborative telepresence based on the mutual integration of two users' surroundings into a consensus reality. Other than in classical videoconferences or immersive telepresence systems, there is no separation between the users' environments. Instead, the users have the impression of standing in their own room and see their conversation partner rendered through a head mounted display, as if they shared the same physical space.

The following pages describe the overall system architecture required to achieve this effect and provide an examination of the two core challenges arising from this interaction scenario. In order to integrate two differently shaped rooms into a shared environment, the position of the users and the layout of the floorspace must be aligned carefully. At first, the human pose tracking is considered in detail. This tracking of posture can be understood as a high dimensional optimisation problem in a stochastic framework. The central problem of approximating the observation likelihood of a given pose is discussed in detail. Furthermore, the integration of the resulting approximation function into an Annealing Particle Filter is described and evaluated extensively.

Once the users' poses and positions are known, their environment can be merged into a shared consensus reality. This leads to a second optimisation problem. Since the participating rooms can have very different layouts, discontinuities between the two spaces can destroy the illusion of co-presence. The problem is expressed through a series of energy functions, which can be approached as a maximisation problem. The design of these energy terms is discussed in detail and a thorough examination of their characteristics is given.

Common to both topics is the expression of otherwise intractable problems in a global optimisation framework and their central role in driving the envisioned telepresence system. While the creation of the consensus reality sets the foundation for projecting two rooms into a common workspace, the human pose tracking then drives the interaction with virtual content. The goal is to provide a channel for communication over distance which incorporates not only visual and auditory cues, but allows the users to interact naturally by sharing the same physical space - even if they are on different continents.

Zusammenfassung

Diese Dissertation stellt ein neuartiges System für die kollaborative Telepräsenz vor. Hierbei werden die Räume der beiden Teilnehmer in eine gemeinsame Konsens-Realität zusammengeführt. Im Unterschied zu herkömmlichen Videokonferenzsystemen oder immersiven Telepräsenzsystemen wird dabei keine statische Trennung zwischen den teilnehmenden Umgebungen vorgenommen. Die Benutzer des Systems haben daher stets den Eindruck, in ihrem eigenen Raum zu verbleiben. Der Gesprächspartner wird jeweils über ein Head Mounted Display dargestellt und natürlich in die Umgebung eingefügt. Dabei teilen sich beide Teilnehmer einen gemeinsamen virtuellen Raum, der im Hintergrund die beiden örtlich getrennten Lokalitäten in einen Arbeitsraum zusammenfasst.

In den folgenden Kapiteln werden die theoretischen Grundlagen sowie die Gesamtarchitektur des Systems beschrieben. Aus der Konzeption des Systems ergeben sich zwei zentrale Herausforderungen, die im weiteren Verlauf dieser Dissertation genauer untersucht werden. Die Kenntnis der Körperhaltung der Benutzer ist zentral für die Interaktion mit dem System und die Konstruktion der Konsens-Realität. Die Erfassung der Haltung kann dabei als ein hochdimensionales Optimierungsproblem betrachtet werden. Zur Lösung bieten sich ein stochastisches Trackingverfahren an, der Annealing Particle Filter. Seine Funktionsweise und Adaption auf die Problemstellung wird ausführlich beschrieben und in einer Reihe von Experimenten untersucht. Dabei wird besonderes Augenmerk auf die Approximation der Beobachtungswahrscheinlichkeit gelegt.

Auf Basis der bekannten Benutzerpositionen kann im Folgenden die Konsens-Realität konstruiert werden. Hier ergibt sich ein weiteres Optimierungsproblem. Falls nämlich die Räume nicht sorgfältig aufeinander abgebildet werden, können Brüche zwischen der Raumgeometrie die Illusion der Kopräsenz rasch zerstören. Daher wird die Geometrie und die Position der Benutzer in eine Reihe von Energiefunktionen überführt, die mittels globaler Optimierungsverfahren gelöst werden. Die Details der Problemformulierung und die Lösungsansätze werden ausführlich diskutiert. Eine Reihe von Experimenten illustriert dabei Charakteristiken der einzelnen Terme und untersucht die Eignung verschiedener Optimierungsverfahren.

Beide Schwerpunkte haben gemein, dass analytisch nicht lösbare Probleme mittels globaler Optimierungsverfahren betrachtet werden. Ebenso sind beide Themen von zentraler Bedeutung für die Realisierung einer Videokonferenz in der Konsens-Realität. Während die Berechnung der Konsens-Realität die beiden Räume in ein gemeinsames Bezugs-

system überführt, stellt die Haltungserkennung die Schnittstelle für die Interaktion mit virtuellen Objekten in diesem gemeinsamen Raum dar.

Auf Basis dieser Grundlagen soll eine natürliche und nahtlose Integration von virtuellen Inhalten in Unterhaltungen ermöglicht werden - selbst wenn die Gesprächspartner auf verschiedenen Kontinenten stehen.

Acknowledgments

This dissertation is the product of nearly five years of research at the Institute for Human-Machine-Communication of the Technische Universität München. Over this time, many people have contributed to my studies and experiments in countless ways.

First and foremost among them stands Prof. Rigoll as my doctoral advisor and first examiner. While his vision and advice guided the direction of my research, he also afforded me the necessary liberty to develop own ideas and follow my curiosity. I am deeply grateful for his trust in me and his great support.

During my studies, I enjoyed the company, advice and help of my colleagues. Among these, I owe special thanks to Mohammadreza Babaee, Michael Dorr, Daniel Merget, Simon Schenk and Philipp Tiefenbacher for their help in proofreading early drafts of this dissertation. During my time at the institute, I was privileged to share my office with Daniel Merget and Moritz Kaiser. They both contributed to my work with ideas, insights and their great sense of humour. I am indebted to Dejan Arsić for introducing me to the arcane art of scientific writing and his guidance during my early days at the institute. I also would like to thank the non-scientific staff for their administrative and technical support.

It is unfortunate that I will never get the chance to thank personally the anonymous reviewers who contributed their time, experience and insight to my publications. Their collective advice helped me develop ideas with more clarity than I had thought possible. Their habit of pointing out just the most uncomfortable weaknesses eventually led me to asking myself just the right questions.

Over the five years of my work on this dissertation, many people have accompanied me on my path. I owe thanks to all of them, both for lessons sweet and bitter. Focussing on the sweet lessons, I especially want to thank my parents Christiane and Harmen Lehment, my sister Eva Lehment and my brother Henrik Lehment for their love and support.

Munich, April 2015

Contents

Chapter 1

Introduction

The speed and fidelity at which information can be exchanged between individuals is one of the decisive factors in humanity's history. Whether in the government of states, the waging of wars or in the conduct of trade, efficient and effective communication over distance was the key to success. However, the latency inherent to letters, dispatches and packages often forbade the close coordination required for successful collaboration on complex projects. Before the invention of wire-bound and wireless electronic communication, closer coordination and collaboration could only be achieved by placing people into the same room or building. Most people can achieve efficient coordination only when instantaneous communication is possible, when a question or proposal is met with a response straight away. This need for direct consultation led to the centralisation of people into trading hubs and administrative centres. To this day, meetings and conferences are indispensable whenever organisation and consultation between multiple parties are desired.

Even with the advent of the telephone and videoconferencing, these tools still leave much to be desired. On a telephone, many essential communication cues like gesturing, facial expression etc. are simply lost. Videoconferences solve some of these problems, but suffer from other constraints, such as the artificial boundary introduced by the screen. So what would a perfect remote collaboration tool look like? In the best case, it would simply look just like a normal conversation with both people standing in the same room. They communicate with gestures, voice and facial cues. As one colleague shows the model of a new building, the other collaborator steps around the desk to get a better look at the object. Their posture, bearing and interpersonal distance are all part of the communication process and are exchanged in real time without conscious input from the user. The entire experience should be as seamless as possible, without the window analogies of videoconferencing or the artificial worlds of immersive virtual reality (VR) environments.

The scenario described in the preceding paragraph is at first glance quite similar to telepresence like it is researched by many different groups around the globe: One person puts on an head mounted display (HMD) and immediately the senses of vision and hearing are immersed into another location for a meeting. On the other side of the conversation, the visitor is rendered to the colleague via a screen or some other device. At that point, the question central to this dissertation arises: What if *both* participants have the impression of remaining in their own office? Is it possible to have a telepresence system without being transported to a remote location? The goal would be to create a *consensus reality*

in which both users meet, encompassing both rooms and affording a full integration of all major communication channels. This consensus reality (CR) would extend the concept of augmented reality (AR) to include two spatially separate locations with two or more participants.

The integration of two rooms with different layouts and furniture into a common reference system poses a complex challenge. To get an intuition of the problem, consider the following example: Two users are conducting a meeting using the consensus reality videoconferencing system. User A has a spacious corner office with plenty of uncluttered floorspace. As she talks, she likes to pace the room. Her conversation partner, user B has a smaller office cluttered with bookshelves and desks. He can walk only a few steps before bumping into a wall or a piece of furniture. So what happens as these two different rooms are introduced into the same reference frame? Without adjustments, user A would continue to pace her room as usual, seeing her conversation partner standing near to her desk. As it is her habit, she talks with him and walks around her office. User B meanwhile would watch user A walk around his small office, occasionally disappearing into shelves and walking through his furniture. The different size and layout of the rooms clearly interferes with his sense of co-presence.

It is therefore important to take both the different room sizes and the users' position and heading within these rooms into account. Using the concepts developed in the following chapters, the system could thus determine a mapping of both rooms which minimises discrepancies between the layouts. Thus, the users interact in a consensus reality, which is created from features common to both users' environment. As the meeting proceeds, a pose tracking system would keep track of users' positions and heading directions. These data can then be used to warn users as they are about to leave the consensus reality area, avoiding collision with remote obstacles.

In summary, two questions are central to the concept of the consensus reality:

- How can we determine the body posture of the users?
- How can we calculate the shape of the consensus reality spanning both rooms?

Both the problems of pose tracking and consensus reality computation pose complex optimisation problems. As the mathematical descriptions of the human body and the consensus reality do not permit an analytical solution, more specialised approaches are required. A part of the dissertation will therefore be dedicated to a general overview over the most relevant and promising methods applicable to this challenge. During the discussion of the problems, there will be frequent references to these techniques, their implementation and their performance. Thus, there are two ways to read this dissertation: Either as a guideline towards designing and implementing a consensus reality system or as a study in the application of optimisation techniques to stationary and non-stationary optimisation problems.

In Chapter 2, the general concept of the consensus reality videoconferencing system is discussed in detail. At first, an introductory literature overview establishes the context for this work. Within this context a first definition of the consensus reality is given. The following sections identify the basic functional components of such a system. For each of the components, possible implementations based on existing research and commercially

available systems are discussed. Finally, the chapter closes with a discussion of some yet unresolved technological challenges.

Chapter 3 gives an overview over the applied optimisation methods. These techniques are indispensable for solving the problems of body pose tracking and consensus reality computation in the following chapters. The focus lies on global heuristic methods, such as *pattern search* methods, *simulated annealing (SA)* and *particle filters*. Since human pose tracking is a non-stationary problem, the extension of the particle filter in conjunction with simulated annealing to the annealing particle filter (APF) is considered in detail.

Once the theoretical tools are discussed, Chapter 4 shows how to apply the APF to the problem of human pose tracking when using point clouds as input. Especially the observation likelihood approximation is discussed in detail, as most differences to previous silhouette based approaches are found here. In addition, modifications to the scattering and resampling mechanisms of the APF are presented and examined.

In Chapter 5, we turn to the central problem of consensus reality (CR) videoconferencing. Here the computation and automatic alignment of the consensus reality are described in detail. The process uses 3D models of both rooms to generate 2D floor maps, onto which it overlays camera observation cones and user positions. An optimisation stage attempts to find an alignment between the two rooms which maximises an energy function. This energy function encodes the various requirements on a consensus reality meeting and is described extensively.

The dissertation closes with Chapter 6, which gives a summary of the main contributions and findings. Furthermore, a number of open questions and future directions of inquiry are highlighted. The chapter finishes with a tentative roadmap for the full implementation and future development of CR videoconferencing.

Chapter 2

Augmented Reality Videoconferencing

2.1 Introduction

Remote collaboration has grown into a vast field of research in the last decades. This chapter introduces the context from which the concept of augmented reality (AR) videoconferencing has emerged and introduces a novel AR videoconferencing system. We will start with core concepts of remote collaboration on documents and from there follow the development over virtual reality teleconferencing towards current mixed reality collaboration systems. Those readers already familiar with the terminology and contributions in the field of collaborative AR are advised to proceed to Section 2.4 where the usage scenario envisioned for this dissertation is introduced. Following the usage scenario, the single components making up the conferencing system are introduced and discussed. Their descriptions are paired with practical advice on their implementation. This chapter serves to set the context, scope and terminology for the problems discussed in the following chapters.

2.2 Remote Collaboration in Mixed Reality

The idea of a videoconference is central to the topic of this work. Before diving into the modes of remote collaboration, a definition of this central term is warranted. A good starting point are the definitions given by Mühlbach *et al.* [MBP95]. Their paper defines the two terms of videoconferencing and telepresence thus:

> **Definition of *Videoconferencing*** by Mühlbach *et al.*, 1995
> Videoconferencing is defined as a mode of telecommunication with at least transmission of sound and picture. In contrast to telephony, videoconferencing includes the transmission of visual communication cues, especially non-verbal behaviour.

> **Definition of *Telepresence*** by Mühlbach *et al.*, 1995
> Telepresence is defined as the degree to which participants in a telecommunication scenario get the impression of sharing a space with conversation partners at a remote physical site.

Mühlbach *et al.* further distinguish between spatial presence and communicative presence. This dissertation focuses especially on the spatial presence aspect, as the methods shown here integrate the environments of two spatially separate persons. The communicative aspect itself is left for future research.

2.2.1 Computer supported cooperative work (CSCW)

Before videoconferencing became an established tool, computer users would already use data networks to exchange research results, letters and business data. Sharing and cooperatively editing digital documents and files is therefore the oldest and consequently most mature form of digital remote collaboration. Nowadays, most research in this area is conducted under the broad label of *groupware* or computer supported cooperative work (CSCW). This term encompasses research into file sharing, versioning, team communication and general user interaction [Gru94].

CSCW and groupware are already familiar concepts to many users of current office software packages such as *Microsoft Office*, *Google Documents* or *Pages for iOS*. These and others routinely include facilities for version handling, reviewing and even simultaneous editing of files. Such tools are best suited for collaboration on text or generic data files in most applications.

A general taxonomy for computer supported cooperative work (CSCW) systems was proposed by Johansen [Joh88]. In his book, he identifies time and presence as the major attributes of different groupware systems. "Time" is relevant in the sense of synchronous work on the same matter, as opposed to modifications performed at different times. "Presence" describes the spatial co-location or separation of users. An example of a system facilitating collaboration by spatially separate users at the same time would be chat rooms or videoconferences. Thus, the formal focus of CSCW extends beyond the word processors and spreadsheet tools familiar from office settings.

Although still connected to its technical background, the CSCW community has gradually shifted away from hardware and implementation issues. Instead, they consider questions of user interaction and group dynamics. Typical examples would be Dourish and Bellotti's examination of user awareness in shared work spaces in a desktop environment [DB92], Nardi *et al.*'s examination of instant messaging in workplace settings [NWB00] or Teasly *et al.*'s inquiry into the effects of co-location on productivity [Tea+00]. Most contributions in CSCW focus on desktop workstations, since these are the established tools in enterprise environments. Although the classical definition of CSCW encompasses videoconferencing and remote collaboration, the relevant context for this dissertation is therefore found in the fields of AR and remote collaboration research.

2.2.2 Immersive Collaboration

Text and spreadsheet collaboration benefits strongly from workstation-bound approaches developed by the CSCW community. However, other types of data are not easily handled on 2D displays, such as computer aided design (CAD) models. While for a text document it is simple to refer to single paragraphs or lines, collaboration on 3D data requires more refined reference systems.

Figure 2.1: Previous devices used in immersive VR collaboration. From left to right: *Blue-C* by Gross *et al.* [Gro+03], Kurillo *et al.*'s powerwall approach [KB13]; at bottom: *Grimage* by Petit *et al.* [Pet+10]

Humans are accustomed to inhabiting the same physical space as the person with whom they collaborate. Egocentric directions are our most common frame of reference for objects and environments. This habit leads to conflicts when two persons are discussing the same object without knowledge of each other's position relative to it. It may be natural to say "we need to shift the routing of those cables over there to the left". Unfortunately, such a statement is incomprehensible if the opposite does not know what the conversation partner is looking at. The lack of common reference points prevents the users from associating the same objects intuitively.

In a study conducted in 2004, Gergle *et al.* [GKF04] show how actions can effectively aid the communication process in a shared visual workspace. Even though their research was limited to a non-immersive desktop setup, they conclude that "[their] results highlight the importance of making it clear that people know precisely what remote collaborators can see in a shared workspace. It is not enough to simply allow others to see what is going on, but rather, mutual understanding of what is available to one another is needed." [GKF04].

Making sure both users are standing in the same space and are aware of each other's position and perspective can solve this problem. In the context of remote collaboration,

such awareness can be achieved in a shared virtual reality capable of rendering both the content and the representations of the collaborators. A notable example of such a concept was realised by a group at the ETH Zürich in the *Blue-C* project [Gro+03]. Based on a modified cave automatic virtual environment (CAVE) system [Cru+92], they added a 3D video capturing setup capable of digitising the user within the CAVE. The resulting avatar is then exchanged with a remote location running the same hardware. Thus, the users see each as realistic presences in the virtual workspace and can use gestures, speech etc. freely. The drawback of this approach are the bulky hardware[1] and the high cost associated with running two CAVEs. Consequently, even the research group managed to build only a single device, using a mock-up client to simulate the other side of the conversation.

Current research tries to make these techniques more accessible by including cost-conscious devices, even allowing combinations of different device classes. A good example is found in research done by Kurillo, Bajcsy *et al.* at the University of California [Kur+08; KB13]. Their scenario presumes only a *powerwall* or large screen for displaying the VR scene and a fixed camera array for image acquisition. The stereoscopic camera array captures a dynamic 3D model of the users from 12 different directions, compresses the data and transmits it to the remote conversation partner. Here the model is re-constructed and embedded into the virtual scene.

Petit *et al.* at the INRIA Grenoble [Pet+10; Pet+09] follow a similar approach in their *Grimage* project, but leave greater flexibility in the choice of devices. Instead of presuming identical hardware setups for all conversation partners, they include both users with a full immersive suite (such as a CAVE) and users using only a laptop with a webcam into the same virtual environment.

These advances pave the way for a more intuitive remote collaboration on 3D content in purely virtual settings. In the next section, we shall see how the lessons and techniques developed for these purely virtual spaces can be applied to mixed reality collaboration scenarios.

2.2.3 Augmented Reality Collaboration

As outlined in the previous section, users rely on knowledge of each other's perspective when discussing physical layouts. This challenge is not limited to collaboration on purely virtual data, but also becomes obvious when trying to collaborate on real objects with only one person physically present. Here we enter the field of augmented reality (AR), where virtual data is rendered to interactively enrich the physical environment of users. For the remainder of the dissertation, we will use the term of AR as defined by Azuma [Azu97].

> **Definition of *Augmented Reality*** by Ronald Azuma, 1997
> Augmented Reality is defined as any system that combines real and virtual elements. The system is interactive in real time and is registered in three dimensions.

[1] Even 10 years later, modern setups by Papadopoulos *et al.* [Pap+14] or Febretti *et al.* [Feb+13], require large laboratory spaces in excess of 30 m^2.

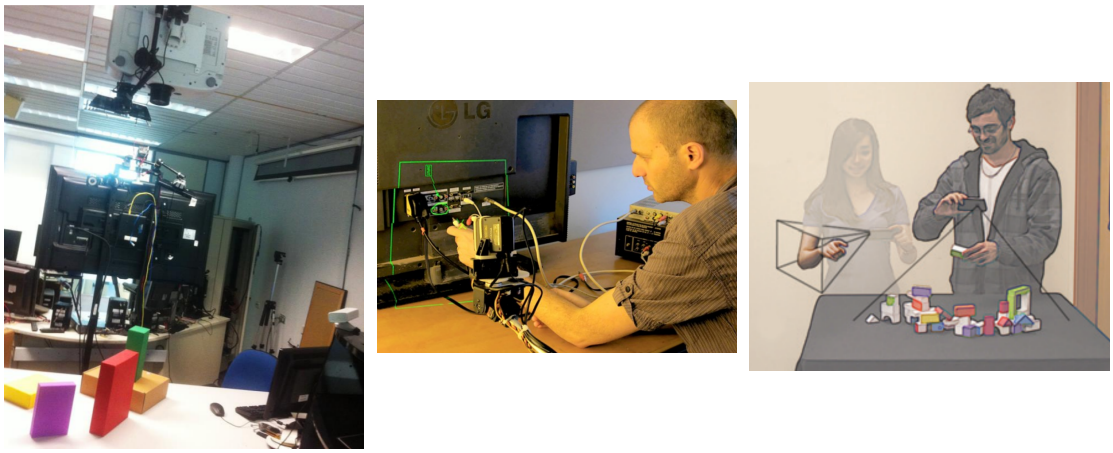


Figure 2.2: Previous devices used in remote mixed reality collaboration. From left to right: *RemoteFusion* by Adcock *et al.* [AAT13], *TeleAdvisor* by Gurevich *et al.* [Gur+12], *BeThere* by Sodhi *et al.* [Sod+13].

In practice, most AR systems focus heavily on visual content. Typically, HMDs and visual tracking systems are employed in order to render virtual content objects into a user's view of the physical world. While Azuma uses the distinction between "real" and "virtual" objects, it would be more apt to use the juxtaposition between "physical" and "virtual" objects[2]. The registration of the system is performed relative to a physical frame of reference. In the context of videoconferencing, these systems can be used to integrate virtual representations of remote persons and objects into a local scene.

Assembly and maintenance tasks often serve as examples in research literature on collaborative AR. Both are situations where the physical layout of the workspace, parts and tools are intuitively described in egocentric terms. These tasks benefit strongly from visual cues and illustrate the benefits of remote experts. The following systems employ these as their application scenarios.

The *RemoteFusion* system presented by Adcock *et al.* [AAT13] demonstrates all the basic concepts of such remote support systems. In their application scenario, a local worker is trying to solve a problem (stacking boxes) with guidance from a remote expert. The workspace is observed by a depth camera system mounted above the table surface. Mounted close to the camera there is a beamer pointed down at the table. The remote expert can observe the scene on a flat multi-touch display, using depth data gathered from the camera for a three dimensional reconstruction. Within this reconstruction, the expert can navigate freely in order to consider the problem from different perspectives. As the expert provides spoken guidance, the multi-touch interface allows drawing of annotations directly into the scene. These annotations are then shown to the local user as projections using the beamer.

While Adcock *et al.*'s system provides the remote expert with a greater sense of the work environment by allowing a free choice of perspective, the ceiling-mounted hardware

[2] Azuma's terms get easily lost in disputes on whether the virtual is actually real. The terms "physical" and "virtual" are more descriptive and avoid such quasi-philosophical debates.

makes the system unsuitable for unprepared environments. Gurevich *et al.* tackle this problem by using a more compact hardware solution [Gur+12]. Their *TeleAdvisor* device is a rather compact robotic arm with a camera/projector head in place of an end effector. Functionally similar to the *RemoteAdvisor*, it also provides the remote expert with a view of the scene while projecting annotations via the beamer. Due to the lack of a depth camera, the expert cannot change his point of view apart from the range afforded by moving the robotic arm. In addition, there is a risk of the robotic arm interfering with the local worker. Nevertheless, the system prototype is far more portable than the bulky *RemoteAdvisor* setup.

The next step in portability is presented in systems like the *BeThere* devices proposed by Sodhi *et al.* [Sod+13]. Instead of using any fixed device for capturing the environment, their work depends entirely on handheld tablet PCs. At the time of presentation, depth sensors were still limited to the form factor provided by either the Microsoft Kinect or the Asus XTion. Using current device-integrated depth sensing cameras[3], their setup would be even more compact than that shown in their paper. Using the depth sensor as data sources for a simultaneous localization and mapping (SLAM) approach, they build dynamic maps of the workspace which are continuously streamed to the remote expert's device. The expert can thus use the tablet PC as a window into the workspace, add annotations and navigate within the model. The local user can use the tablet similarly to see the expert's annotations and current field of view. Thus, the users can agree on what they are looking at and use each other's current view as a reference for directions and questions.

These tools present a first step forward from simple videoconferencing in the connection of remote spaces. However, the telepresence is mostly uni-lateral and focussed on an object present in only one of the two rooms. The environment of the remote expert is dismissed and not integrated into the conversation.

2.3 Towards an Augmented Reality Videoconferencing System

In the previous sections, approaches to collaboration ranging from text documents over CAD files and virtual scenes up to real objects were discussed. However, this list excludes a wide range of intangible foci of collaboration, especially the exchange of ideas, negotiations and discussions. Although there is a long tradition of discussing ideas and concepts in letters, this is often found cumbersome for situations were both sides prefer a fast exchange of opinions and positions[4]. If a direct meeting is not possible, people often resort to telephone calls or videoconferences. These help colleagues coordinate quickly, but are also frequently used to discuss more immaterial concepts, such as business plans, strategies or personal matters.

[3] At the time of writing, Intel is presenting the RealSense devices to a larger audience. Similar systems are also being prepared by NVidia and Google.

[4] It should however not be overlooked that exchanges of letters foster a different and often more diversified type of argumentation and thinking than a direct conversation. Complex arguments can be developed with more care and detail than possible in most conversations. The goal of this dissertation is therefore not to declare the age of letters to be over, but rather to widen the choices.

While far from replacing the telephone, videoconferencing has found its own place in remote communication and is often used in both private and business settings[5]. Research around screen-to-screen videoconferences has mostly turned to either implementation details (continuity of gaze, encoding etc.) or social matters (in the wider context of CSCW). Meanwhile there are a number of attempts to extend remote presence away from displays and into the rooms of the participants.

A group around Prince, Billinghurst and Kato [Pri+02; Bil+02] contributed one of the earliest works which uses AR to achieve such an integration. Coming from an earlier mock-up in which portraits of conference partners were superimposed over markers, their 2002 papers used a video-see-through HMD to show live videostreams of users anchored to physical markers. The videostreams shown embedded into the markers were flat 2D representations, since affordable depth sensing cameras were not yet available. Nevertheless, this posed the first step in moving conversation partners away from fixed screens and displays.

Only two years later, Barakonyi *et al.* [BFS04] attempted to enrich conventional videoconferences by rendering application data directly into the videostream. In their guiding scenario, a user could hold up a marker to the camera during the conversation. The software would then render the output of a program running on his computer onto this marker, e.g. an animated Matlab plot or a video clip. The virtual content thus becomes attached to a physical object the user can hold, manipulate and interact with. For rendering, their system still resorted to a conventional videoconferencing setup with a flat display acting as a window between the two rooms.

The window analogy innate to videoconferencing was extended in 2011 by Maimone *et al.* [MF11]. While keeping the display as the focus of the interaction, they placed a number of Kinect cameras in both users' rooms and generated a dynamic 3D model of the two rooms and the users sitting in them. During the session, the rendering shifts depending on a user's position in front of the screen. This shift mimics the motion parallax people experience in everyday life when looking through a window: If the user moves to the left, the field of view through the window shifts to the right while things moving behind the left frame of the window are lost to sight. Such adaptive rendering gives the remote scene a greater plasticity and a sense of depth triggered by the shifting of perspective. Additionally, the system allowed for the inclusion of AR content for display and interaction.

The approaches described so far are still either limited to the display of a conversation partner on a fixed screen or at least with a reduced sense of depth. There are attempts to create greater mobility by employing more elaborate hardware for rendering and interacting with the conversation partner. A number of mobile telepresence platforms are already available commercially at the time of writing[6]. These are essentially remote-controlled robots fitted with cameras, a screen, a microphone and speakers. They are controlled through a web-browser interface, allowing the remote user to navigate office spaces and

[5]Some of the systems have become so commonplace that the service providers have turned synonymous to the service they provide. As an example, the expression "to skype" has become a fixture in contemporary conversation.

[6]Some typical vendors are Suitable Technologies, Double Robotics, iRobot, Anybots, Vgo, Mantarobot, Romotive etc.

homes. Video and audio streams show the surroundings and enable interaction with other people physically present. In most models, a screen and camera are fitted at the height of a typical human face. The screen shows the remote user's face, making the telepresence robot act like a mobile videoconferencing unit. Voice is transmitted using speakers and microphones build into the chassis of the robot.

Figure 2.3: Devices used in extended remote collaboration scenarios. From left to right: First AR teleconferencing system by Prince *et al.* [Pri+02], telepresence robot "QB" by *Anybots*, *TeleHuman* by Kim *et al.* [Kim+12].

In the context of AR videoconferencing, the *TeleHuman* system presented by Kim *et al.* [Kim+12] shows a more elaborate rendering of a remote user. Other than the small face screens of telepresence robots, they enable a full rendering of the participants body and thus communicate complex gestures and posture. Similar to the fully immersive telepresence systems described in the previous section, the remote user is scanned by a number of depth cameras from different angles. Instead of using the resulting 3D model as an avatar in a VR environment, it is rendered to a cylindrical display. The remote user is shown perspectively correct without the aid of HMDs, providing important depth cues such as stereoscopic rendering and motion parallax. The system requires a free space of about $5 \times 5\ m^2$ for the rendering equipment and camera system.

Finally, Maimone *et al.* [Mai+13] extended their windowed videoconference by adding an HMD. As there are 3D models of both users' spaces available, users can choose to see only the conversation partner or decide to see also parts of his remote surroundings. This selective display leads to a blending of remote and local spaces. For this reason, their system relies on a fixed layout of furniture in the participating rooms. In the current implementation, it provides only one of the users with a full immersion in the remote location, while the other can only see the conversation partner as a 3D presence.

In summary, these systems are a good choice if one user wants to be present in a remote space for conversation or social interaction. A selection is also shown in Figure 2.3. As videoconferencing moves away from stationary screens, there is the potential for more spontaneous and natural interaction. The long-term goal of research into AR videoconferencing is the seamless and natural integration of distant conversation partners into each other's environment, leaving barriers such as displays etc. behind. This vision brings us to the usage scenario discussed in the next section.

2.4 Definition of Consensus Reality Videoconferencing

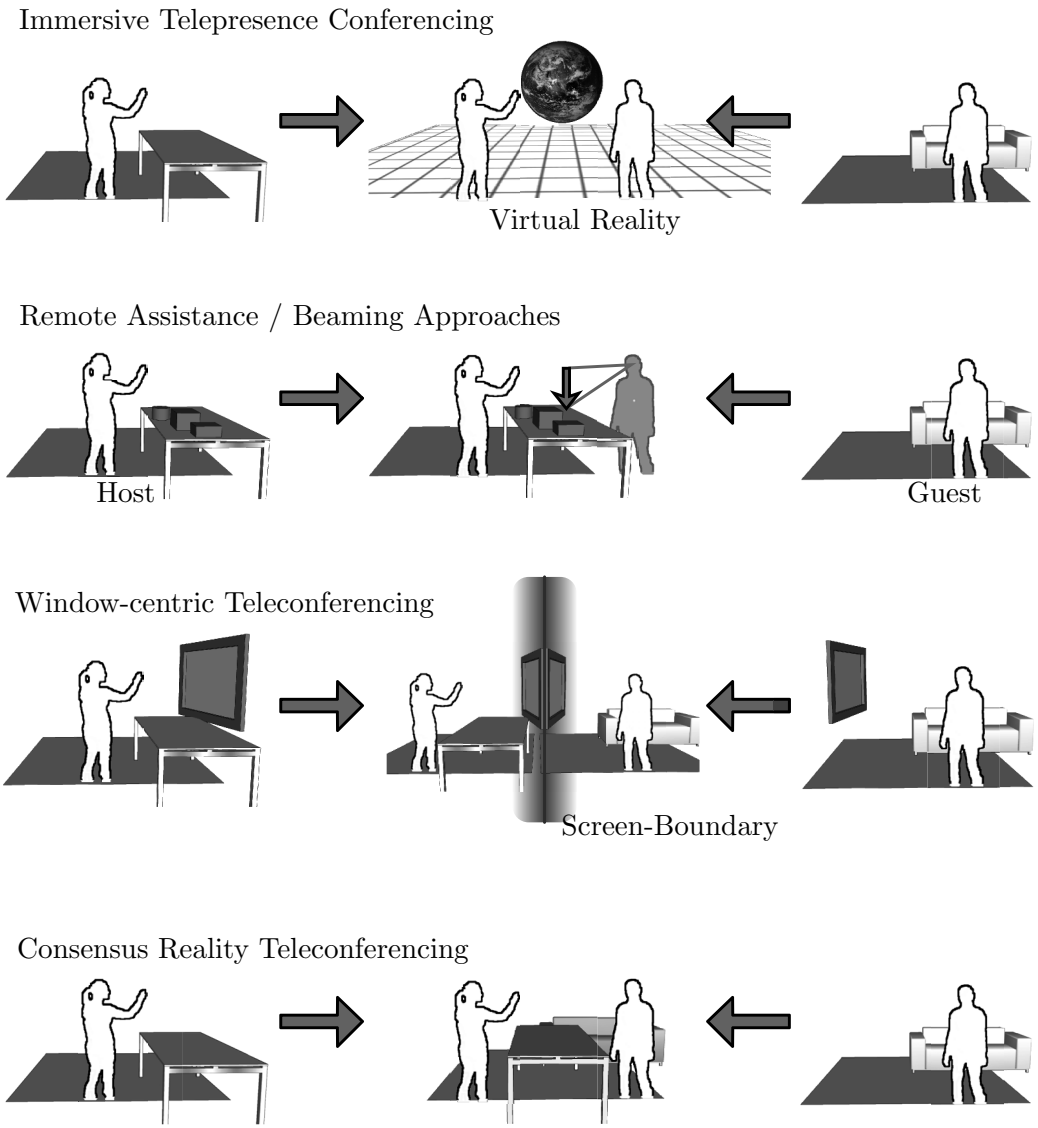


Figure 2.4: Comparing the consensus reality approach to other methods. On the outside the participating users are shown, the scene in the middle column shows the combination of the participating environs into the collaboration scenario.

The mutual inclusion of remote avatars into the physical surroundings of both users is a usage scenario not previously discussed in literature. Thus, user A sees user B as an augmented, life-sized presence in her room. Simultaneously, user B sees user A rendered into his own surroundings. This mutual presence extends common videoconferencing systems to include the physical surroundings of the conversation partners while avoiding the uni-directionality of remote telepresence systems. Both conversation partners remain in

their current space and perceive their opposite as a guest entering their room, as shown in Figures 2.4 and 2.5. The first description of such a system was given by me at the 2014 IEEE international symposium for mixed and augmented reality (ISMAR) [LMR14].

While there were previous attempts at introducing AR elements to videoconferencing, these always constrained the spatial integration either by giving preference to one room (e.g. remote support scenarios) or by defining fixed boundaries (e.g. interaction on identical table surfaces).

When integrating two non-uniform rooms into a shared frame of reference, there are bound to be discrepancies between the rooms. We can expect differently shaped rooms, unfavourable placement of furniture and many more differences. While pre-defined workspaces accessible to both conversation partners avoid the problem of discrepancies, they require both conversation partners to prepare rooms with identical floorplan and furniture. Outside of large companies, this is usually not feasible.

For a meaningful collaboration within the same reference frame, it is therefore necessary to identify those elements of the rooms on which both users can agree. This agreement would include all aspects of the rooms which are similar or even identical for both, e.g. floorspace which is unobstructed for both participants. The outline of such a consensus reality (CR) are illustrated in Figure 2.6, with hatched regions showing areas and objects on which there is a consensus. In the following, a technical definition of consensus reality as used in this dissertation is given.

Definition of a *Consensus Reality*:
A consensus reality encompasses two spatially separate rooms or environments. In order to be part of a specific consensus reality, an object present in one room must have a corresponding object with similar properties at the same position in the other room. The consensus reality is therefore defined by the physical layout of the two rooms relative to a fixed frame of reference and the relative alignment of these frames of reference. It contains all objects which are common to both rooms.

The consensus reality approach aims to support social interaction and conversation in remote collaboration tasks. It especially complements approaches placing both users into a immersive collaborative VR setting. Instead of directly plunging into an unfamiliar VR setting, both users can instead start their meeting in a AR videoconference, retaining their familiar surroundings. At this stage, the focus would lie on the social aspect of the meeting: Introductions are made, progress might be discussed etc. There is no need for immersive elements, since only the conversation partner is of interest at this stage. Only after this initial phase of the meeting is over, the users would switch into the immersive VR mode in order to work with 3D content.

During this meeting, a previously established consensus reality is used to define common workspaces and guide the users around remote obstacles, preventing them from stepping into their conversation partner's furniture and walls.

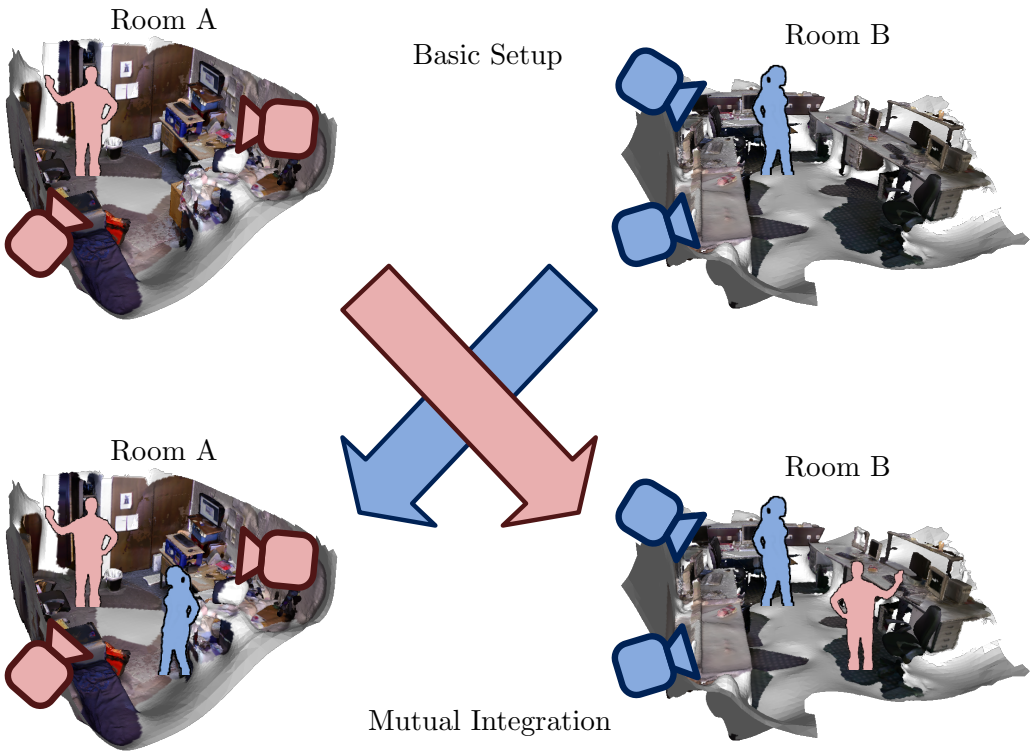


Figure 2.5: Schematic consensus reality videoconferencing system.

2.5 Required Functionality and Components

The system outlined above requires a number of functional components. The following four functions comprise the most important elements needed to build a working CR videoconferencing system.

1. Transmission of user representations (visual and voice data)

2. Rendering of users into each other's scene

3. User interface for interacting with the system and virtual content

4. Session management, content management and consensus reality generation

In the following sections, these basic functionalities and their possible implementation will be discussed in greater detail. Figure 2.7 illustrates the interaction between these components and their place in the overall system. The figure shows the system in its simplest implementation on a single computer. In practice, dedicated devices could provide some functions in order to maximise performance. For instance, a belt-worn computer might drive the HMD, using data exchanged with the session server over a wireless connection.

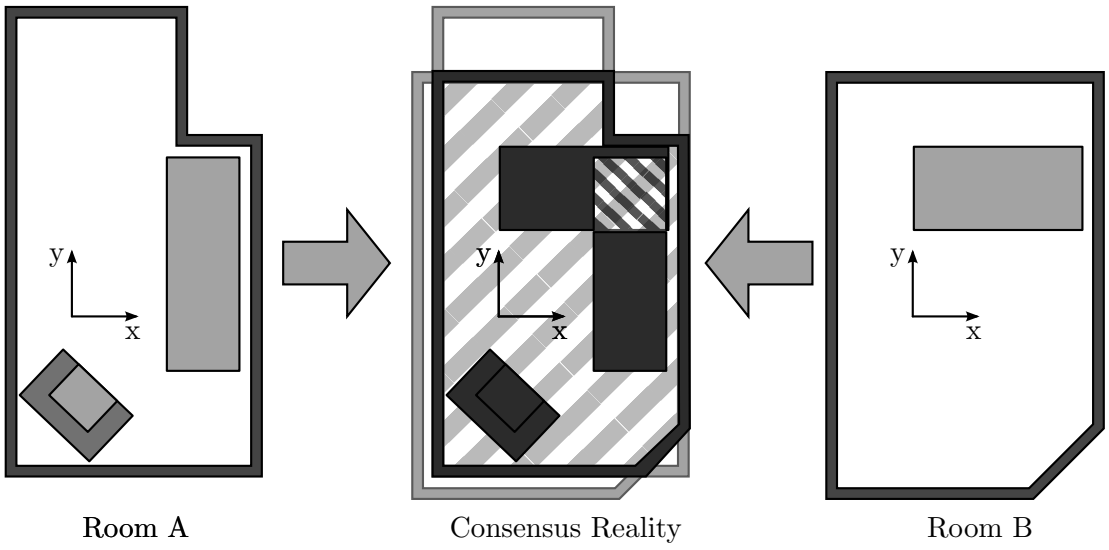


Figure 2.6: Consensus reality for two rooms without additional alignment. Hatched regions show the consensus reality, while the dark elements lie outside of the consensus. Stepping through such an element would effectively look like stepping into an object to the conversation partner.

2.5.1 Avatar Acquisition and Transmission

The users' avatars should be able to convey the full range of human emotion and expression perceivable to hearing and sight. To this end, it is important to convey language, pose, facial expression and gestural cues. The transmission of voice is a well-studied and largely solved problem, so this dissertation shall instead focus on the visual cues informing our everyday interactions.

Among the most important visual cues in social interaction are the general shape and aspect of the body, posture, facial expressions and gesturing. A single monocular camera can capture these and reproduce the image on a suitably large screen. However, as the goal is to render the person as a free-moving presence into a remote room, all data needs to be recorded, transmitted and rendered for a 3D model. Otherwise, the remote avatar would remain as flat as a cardboard cut-out to the conversation partner. Without 3D data, all rendering is reduced to the one fixed perspective from which the camera observed the original person.

In current literature there are two solutions to this problem. For once, it is possible to use a pre-recorded avatar of the person and actuate it by applying the current body pose of the user. Consequently, these approaches are often referred to as "puppeteering". The avatar however needs to be pre-build ahead of time and is not able to express facial expressions and microgestures. Microgestures are small movements of the body, which are not easily detected using body pose tracking, e.g. a slight shrug or complex movements of the hands and fingers. These subtle movements are hard to capture by human pose tracking systems. On the other hand, there are low requirements on bandwidth, as only

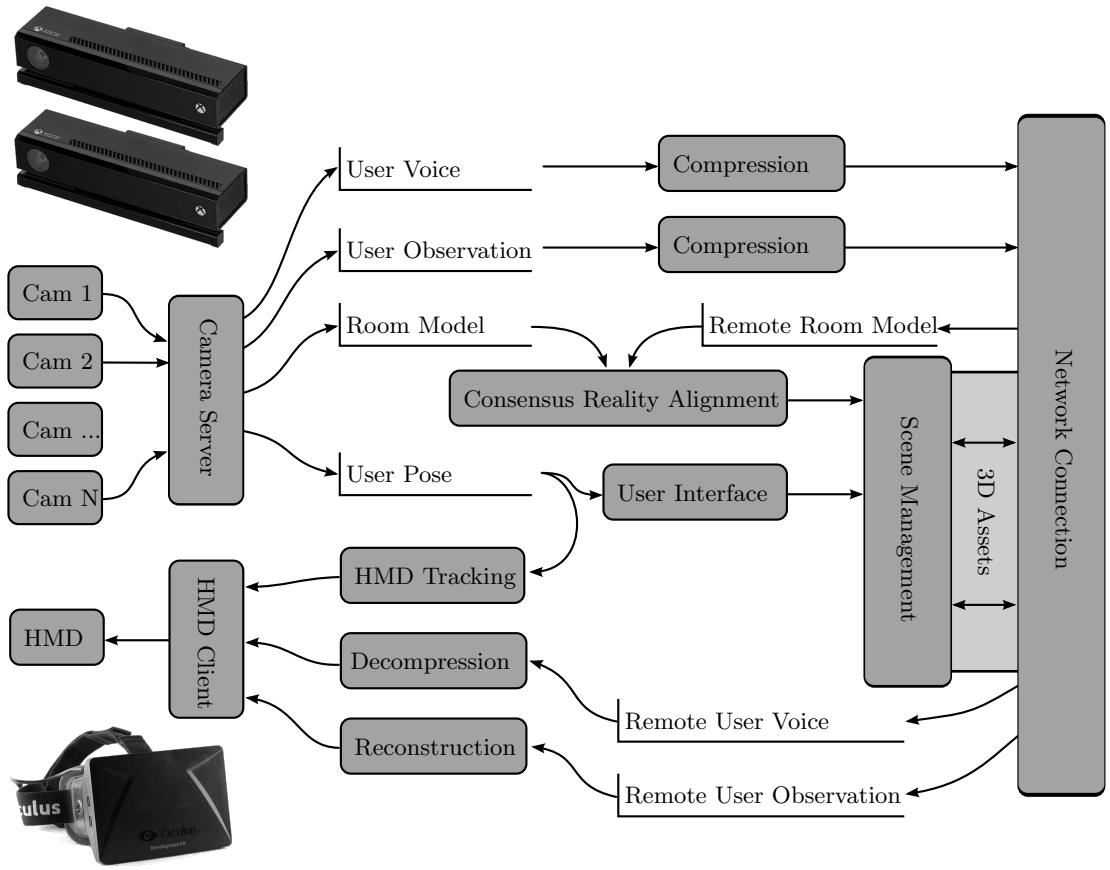


Figure 2.7: Interaction between the system components for a generic dual user consensus reality conferencing system. Only components for one participant are shown, the setup is mirrored on the other side of the conversation. Consensus reality alignment is performed only on one side, results are sent to the conversation partner over the network connection.

compact skeleton pose data is transmitted. The camera system can also be kept rather simple, in the most basic case a single pose tracking camera is sufficient to drive the remote avatar.

The second solution does not try to build complex models and instead simply transmits the data recorded from the camera system. As the data contain depth and colour information, these can be used to construct point cloud models on the fly. These point clouds are able to express both facial expressions and micro-gestures freely, at the cost of providing lower detail than the pre-build polygon models can supply. The quality of these dynamically built point clouds depends on the resolution of the camera system used, which might pose problems for older cameras such as the Kinect v1 if not placed close to the user. Newer camera systems such as the Kinect v2 should be preferred.

By necessity, the camera system for these approaches is more complex. Full coverage of the observed user from all angles is required. To provide this coverage, multiple cameras are spread throughout the room. The system thus bears similarity to the setup proposed by Schönauer and Kaufmann [SK13] and Martínez-Zarzuela *et al.* [Mar+14], but extends the scope beyond pure motion tracking. The system discussed here assumes the use of multiple Kinect cameras, providing red-green-blue-depth (RGB-D) data. Calibration is achieved either using a marker-based approach, e.g. using the ARtoolkit software [WS07] or a checkerboard marker as shown by Berger *et al.* [Ber+11]. Alternatively, Svoboda *et al.* [SMP05] demonstrated a setup with self-calibrating cameras. Agethen, Otto and Rukzio [Age15] take up the concept of self-calibrating camera arrays in their current work. The results are to be published in the coming months after the print of this dissertation[7]. Their approach uses data from a human pose tracking system to calibrate the cameras relative to the user's skeleton, yielding a very fast and user-friendly system initialisation.

The hardware used by Agethen *et al.* is already well suited to rapid setup and installation: A number of self-contained camera units can be placed freely in the room. The camera units consist of a depth-sensing RGB-D camera module, a small computer, a power supply and a wireless local area network (WLAN) adapter. Once placed and switched on, they connect wirelessly to a local control server and start streaming observation data. As the system is self-calibrating, little user interaction is required beyond placing the cameras, plugging them into a power supply and walking through the room for about 10 seconds. This approach is perfect for setting up ad-hoc conferencing systems, where the user just scatters some cameras throughout a room without spending a long time on calibration and setup tasks.

While affording expressive communication of a user's visual presence, the direct streaming of the observed user comes at the price of high bandwidth requirements. A typical observation frame can easily contain more than 100.000 individual points. If these are encoded using a basic *XYZRGB* format at 32-bit for each value, a single frame yields more than $2.8MB$ of data. Streaming over a public network at more than 25 frames per second (fps) becomes infeasible at this point.

Previously published approaches to AR videoconferencing usually either avoid this problem entirely by using only a network mock-up [Mai+12] or sending via a high-bandwidth LAN connection [Kim+12]. This substitution is obviously not realistic for most applications outside of a lab setting. A possible remedy would be the reduction of the entire point cloud using Octree compression techniques as demonstrated by Kammerl *et al.* [Kam+12]. Thus, first a local server gathers the data from all cameras in the local room, uses the extrinsic calibration parameters to reconstruct a 3D point cloud, compresses the aforementioned cloud and then sends it to the remote conversation partner. Here the cloud is again decompressed and rendered to the user.

A recent study conducted by D. Salesski [Sal14], a student working under my supervision, examined the bandwidth requirements of the state-of-the art approach presented by Kammerl *et al.* [Kam+12]. Salesski's study used point clouds generated from a single depth camera which were then processed by Kammerl *et al.*'s double buffering Octree compression algorithm. He reported bit rates between $47^{\text{Mbit}}/_{s}$ for low quality and up to

[7] I was supervising Agethen's Master's thesis in that field and appear as a co-author.

620$^{\text{Mbit}}/_{s}$ if 30 fps were transmitted. The point cloud compression at the current state of research is therefore not a suitable solution for the transmission problem in AR videoconferencing. However, the field of real-time point cloud compression is still very young. It is reasonable to expect more powerful compression algorithms to emerge in the future.

For the time being, there is a wealth of research on compression of 2D image data. Since the original data provided by the camera system is provided in just that format, it makes sense to apply these existing techniques to the transmission problem. Current depth sensing cameras such as the Kinect or the Xtion provided data in two image streams: A red-green-blue (RGB) stream provides data from a regular camera, usually uncompressed in a raw data stream. A second data stream carries the depth data, also formatted as a 2D array of data. Both streams can be cast into standard image formats which are easily compressed either as a sequence of images or even into a videostreaming format[8]. This compressed stream is sent over the network at reasonable bit rates. Salesski reports bit rates in the region of $1 - 9^{\text{Mbit}}/_{s}$ at a pixel-wise signal-to-noise ratio (PSNR) from $30.0 - 32.4$dB for the colour image stream using H.264 video compression and $2.9^{\text{Mbit}}/_{s}$ with a PSNR of 59.2dB for the depth stream using JPEG encoding. Thus point cloud transmission for a single camera can be realised reliably with around $4^{\text{Mbit}}/_{s}$, an order of magnitude less than the bandwidth required using Kammerl's compression scheme. This value can be further reduced by transmitting only image regions containing the user. Such a selective transmission is implemented easily with available foreground extraction algorithms, e.g. Hofmann *et al.*'s pixel-based adaptive segmenter (PBAS) algorithm [HTR12].

After the conversation partner receives the compressed data stream, the known intrinsic and extrinsic camera parameters are used to reconstruct the point cloud locally. As data streams from several cameras are broadcast, the receiving computer merges multiple views into a shared reference system. After an initial alignment based on the known extrinsic camera parameters, a secondary iterative closest point (ICP) [RL01] pass results in a full user representation[9]. This merged data provides a 360° representation of the remote user.

Although the inherent granularity of the point cloud does lead to lower visual quality than a high-definition avatar, at the same time it avoids the *uncanny valley*[10] problem encountered by these avatars [BSL13]. Another advantage lies in the display of real-time facial expressions and micro-gestures which are hard to replicate in avatar based videoconferencing approaches.

Transmission of audio data also falls into the scope of avatar transmission. Since there are a number of existing solutions freely and commercially available, there will be no extensive discussion of the methods at this point. Interested readers are advised to consult real-time transport protocol (RTP) implementations such as H.323 in the *OpenH323*

[8] There is a wide choice of tools for this task. For research purposes, most practitioners use the *ffmpeg* library.

[9] Though ICP can be computationally expensive, the procedure only refines the camera calibration and does not need to be performed in real-time for the entire data stream. In most cases, a single initial calibration should suffice.

[10] The uncanny valley effect desribes the sense of unease some observers report when seeing something closely resembling a real human, but not quite achieving full realism. The term was coined by M. Mori in an 1970 essay [MMK12].

project or the session initiation protocol (SIP) as implemented in *OpenSIPS* for more information.

2.5.2 Rendering

There are no strong requirements on the render engine used. It should be able to render point clouds, receive tracking data from the HMD unit and render the perspective-adjusted content (user and AR objects) to the HMD. At the time of writing, there are a number of commercial and free engines available which can achieve this easily[11]. The advantage of using an existing engine lies in the great number of available modules, accelerating the development process. Nevertheless, all functions are also implementable using more basic tools such as the *OpenGL* libraries directly. The implementations tested for this dissertation were realised in *3DVIA Studio Pro*, an integrated content development platform available by Dassault Systems. The platform offers compatibility to the CAVE system installed at the institute for human-machine-communication, which facilitates the implementation of an early demonstrator.

For an AR videoconference, at least the following elements will need to be rendered: The conversation partner (point cloud), virtual objects (polygon models) and menu structures (items, lists, navigation elements). In addition, the engine should support procedural generation of new geometry objects in order to visualize the consensus reality to the participants. Thus, as a user walks close to a region containing an obstacle in the conversation partner's room, this obstacle can be rendered to that participant. This visualisation helps users to remain within the boundaries of the consensus reality.

In case of projective AR systems, the engine should offer full support for custom graphics shaders in order to account for geometric distortion based on the environment.

2.5.3 Session Management, Content Management and Consensus Reality Generation

The various functionalities need to be integrated in a central application which coordinates network transmission, rendering, content management and the various other functions needed for a remote collaboration scenario. Once rendering, content management and all other functions are integrated, the central application continuously controls the flow of information between the participating users. All these tasks are within the scope of current game and content development engines, which makes them well suited for rapid development and prototyping of an AR videoconferencing scenario. The term "game engine" should not be misconstrued at this point. While the focus of marketing lies on development of games, all major engines offer an integrated suite of render engine, physics engine, network management, content scripting and further software development kit (SDK) hooks for integrating own software modules.

Besides transmission of user point clouds and their rendering, the central engine needs to manage additional AR content used in the collaboration scenario. Typically, these would be 3D renderings of CAD data, shapes symbolising associated data or menu structures and

[11] Obvious choices would be the *Unity* engine, the *Unreal* engine or the *CryEngine*. All three offer integration of at least the *Oculus Rift* HMD, network capability and content management.

items. As an AR object is introduced into the conversation by one user, the object and its properties must be propagated to the conversation partner over the network connection. Any subsequent manipulations must be communicated as well. These tasks are usually covered by the network management of the governing engine, simplifying the development process.

For a consensus reality scenario the physical layouts of the rooms need to be taken into account. Besides its other functions, the central application therefore drives the automated generation of the consensus reality and integrates the camera system. This process is described in great detail in Chapter 5. Since the procedure presumes the existence of a 3D room model, the same geometry data can also be used in the context of the AR content interaction. Knowing where physical walls and boundaries are located, the engine can integrate these into the interaction mechanics. Thus virtual objects can be subjected to simulated gravity, making them fall down, rest on table surfaces and generally behaving like physical objects. Izadi *et al.* [Iza+11] previously demonstrated such an integration of virtual and real objects in the *KinectFusion* system and a number of subsequent demonstrators since. Subjecting virtual objects to physical constraints and effects such as gravity and collisions with real objects helps in making the AR space feel more natural and immersive to the user. It also allows the interaction design to follow natural metaphors, such as "dropping" a model into the workspace of the conversation partner.

An important part of the illusion of co-presence lies in the integration of the consensus reality boundaries. The application therefore monitors the users' position within the consensus reality and uses subtle cues to inform users when they are about to overstep the boundaries of the shared space. Thus the engine warns users when they are about to step into their partner's desk or are about to disappear through one of the walls present in the other room.

2.5.4 Interaction Design

The goal of the system is to provide a intuitive and natural user experience both in interaction with the conversation partner and in the interaction with the system and virtual content. As the display system does not use the window analogy employed by current videoconferencing systems, the users are free to move within their rooms. This mobility leads to the question of how they are to interact with the system without carrying a separate keyboard or touchscreen. Fortunately, there are already a several possible solutions for intuitive interaction.

For simple system commands, a speech recognition module would be the obvious solution. Research in this field has matured to the point of yielding commercially available SDKs. These tools provide robust speech command recognition and are already employed extensively in smartphone-centric applications. The integration of "Google Now" into the applications running on a smartphone serves as an excellent example for such SDKs. Simple tasks like placing a call, adding an entry to a calendar etc. are available as intuitive speech triggered commands. The user can also add further parameters to the command, such as time and date for a calendar entry or the content to be placed in a new note.

Research conducted in the context of this dissertation concentrates on pose-based interaction. This choice is motivated by the specific locality of AR content objects, meaning

that these objects are associated with a fixed position and volume. While it would be possible to trigger a speech command in order to manipulate a virtual object, it is often more intuitive to interact with such localised objects with touch and gestures. As an example, we consider a CAD model placed into the shared collaboration space. A user can select the object for manipulation by placing a hand within the space occupied by the model. As the system detects the proximity of the user's hand and the interaction frame of the object, a number of control icons appear around the object. When touched, these icons can either directly manipulate the object, e.g. turning it around an axis, or expand into a menu structure making further options available.

Not all interactions are necessarily bound to an existing object and approachable via language. Especially when introducing new objects into the scene, it makes sense to show the user a palette of possible selections from which she grasps the desired object and drags it into the scene. Such menus are by nature not bound to an object, but are user-centric. They must be easy to reach for the user from the current position, intuitive to navigate and interact with. Again, this is a scenario where the inclusion of pose and gesture provides a natural interface.

K. Erhardt and C. Schäfer studied the integration of such gesture and pose driven interfaces in the course of their master's theses [Sch12; Erh12][12]. Their work contains more information on the integration and design of such systems. A joint poster including some of Erhardt's work won the "Best Poster Award" at the 2012 ISMAR [LER12]. It discusses especially the integration of AR content objects into a pose driven AR videoconference and explores interaction by virtual touch.

Work done by P. Tiefenbacher, a colleague at the institute for human-machine-communication, explores the interaction with AR objects in even greater depth. His recent papers on touch interaction and object manipulation provide a good overview over methods applicable to our usage scenario [TPR14; Tie+14].

2.5.5 Overall System Integration

The systems described in the previous sections requires a specialised hardware setup on both sides of the conversation. In the following, this setup is examined in more detail for one of the two participating rooms. Note that since the setup is identical for both participants it suffices to describe one room in detail.

Each room has several cameras observing the interaction space. These cameras provide colour, depth and optionally pose tracking data of the user to a central server. The intrinsic and extrinsic calibration parameters of these cameras must be known for reconstruction of the observed scene as a 3D point cloud. Since the scene is reconstructed only after transmission of image data to the respective remote destination, the server sends calibration data along during the initialisation phase. At least three cameras should be available for a small room, covering the entire intended collaboration space in order to ensure observability of the users. In case Kinect cameras are used, the microphones can also be used for gathering audio data. Otherwise, a separate microphone / speaker system is required for sound data.

[12]Their theses were supervised and guided by me. They conducted their work at the institute for human-machine-communication.

In addition to the fixed camera system, it is advisable to have an additional mobile depth camera available. This mobile unit easily fills holes the room model. Such holes appear for regions not observable by the fixed camera system, especially when interacting in cluttered environments or working with less than four cameras. The user can then use the mobile camera to fill holes in the room model.

A central sever gathers camera and audio data, usually also acting as the session host and running the conference engine. Ideally, the camera management is integrated directly with the engine so as to provide the room model for consensus reality building and model interaction. On the server, the data are compressed and sent over the wide area network (WAN) to the conversation partner. Conversely, data sent from the remote location is received and reconstructed into a 3D model of the conversation partner. To this end, the session management engine decodes the single RGB-D streams and converts them into 3D point clouds relative to the observing cameras. Using the provided extrinsic calibration data and the alignment data from the consensus reality generation process, these single point clouds are transformed into the common consensus coordinate space and then handed over to the render engine.

The render engine takes the current scene and the user position within the room as input and computes the current stereoscopic view for each frame. Depending on the display technology, additional shaders are run over the view output to account for the display geometry and mode of the HMD[13]. The HMD chosen should be lightweight, with robust tracking capabilities and a large field of view (FOV). As there are already a number of calibrated cameras scattered throughout the room, these modules can provide additional head tracking data. In fact, the approach implemented by Agethen *et al.* described in Section 2.5.1 was designed for precisely such a scenario and requires no further adaptation. This dissertation focusses on HMDs as a display technology for the conversation. However, most of the methods presented here are not limited by the mode of display and may be used in conjunction with different approaches, e.g. projective methods.

2.6 Technological Challenges

While most of the actual infrastructure of the AR videoconference is easily implemented using existing libraries, hardware and engines, challenges arise from the special conditions under which these elements are brought together.

In the area of pose tracking, the combination of multiple depth cameras for driving interaction with AR content explores a previously little examined application scenario. While there are plenty of multi-camera human pose tracking approaches, these were commonly specified for high quality motion capture in cinematic and game production. Most of them are not real-time capable, have high computational cost or other disadvantages. Meanwhile previous work on real-time capable interaction systems commonly relies on single camera systems and body part detection rather than tracking. In Chapter 4 a human pose tracking based on segmented point clouds observed by one or more cameras is dis-

[13]Especially the difference in technologies between video-see-through and optical-see-through requires adjustments to the render process.

cussed in greater detail. This approach fills the gap between the two previously mentioned application areas.

The integration of multiple user rooms into a consensus AR space is a novel problem not previously discussed in literature (with the exception of own contributions at the 2014 ISMAR [LMR14]). While there are approaches which work in a pre-defined collaboration space, this space is usually limited to a uncluttered surface or prioritizes one room over the other. However, there are at the time of writing no previous consensus space approaches that treat both spaces with equal importance. A possible solution to this problem is described and studied in Chapter 5.

2.7 Summary of Chapter

The preceding chapter gave a brief overview over previous work in remote collaboration and its relevance to the field of AR. Drawing on previous contributions, a new usage scenario was identified which combines elements of telepresence, AR rendering and remote collaboration. The resulting approach lets users see each other as lifelike representations standing in each other's room. Meanwhile, both users retain the impression of their own surroundings, which are enriched with shared virtual content visible to both parties. The focus on this system lies on a natural integration of remote conversation partners into each other's surroundings.

The hardware necessary for this meeting consists of a tracked HMD for each user, microphone and speakers as well as a depth-sensing camera system distributed in both participants' rooms. Otherwise, no further instrumentation is required. On both sides, a server handles rendering, scene and content management together with transfer of user data.

After a description of the basic functionalities covered by the system and a brief discussion on implementation, the technological and methodological challenges arising from this system design are identified. The following chapters will discuss these challenges in greater detail.

Chapter 3

Optimisation Methods for AR Videoconferencing

3.1 Introduction

The system outlined in Chapter 2 provides a rich collection of optimisation challenges: Where does a user place cameras for the best possible coverage of the room? How do the cameras then discern the actual pose of the user moving in front of them? How can the mapping of the participating rooms be shifted to provide optimal conditions for conducting a meeting? Virtually all kinds of optimisation problems arise in this complex system, many of which lie outside of the scope of conventional, linear minimisation problems. This chapter is intended to introduce a number of optimisation strategies most relevant to the problems of human pose tracking and automatic room alignment. Readers already familiar with optimisation methods may wish to proceed directly to the following chapter.

The most basic form of a static optimisation problem can be expressed as finding a solution $\hat{\mathbf{x}} \in \mathbb{R}^n$ which minimises a function $f(\mathbf{x})$ respecting a number of equality constraints $\mathbf{c}(\mathbf{x})$ and inequality constraints $\mathbf{h}(\mathbf{x})$.

$$\hat{\mathbf{x}} = \text{argmin}_{\mathbf{x}} f(\mathbf{x}) \quad \mathbf{x} \in \mathbb{R}^n \quad f : \mathbb{R}^n \to \mathbb{R} \tag{3.1}$$

subject to

$$\mathbf{c}(\mathbf{x}) = \mathbf{0} \quad \mathbf{c}(\mathbf{x}) \in \mathbb{R}^m \quad m < n \tag{3.2}$$

$$\mathbf{h}(\mathbf{x}) \leq \mathbf{0} \quad \mathbf{h}(\mathbf{x}) \in \mathbb{R}^l \tag{3.3}$$

This formulation poses a generic non-linear optimisation problem as identified by Kuhn [Kuh82]. A wide range of specialised approaches is available for tackling different types of optimisation problems. These approaches commonly exploit the structure of the problem in order to achieve a simplification or reduction of the solution space. Especially for problems with a given analytical structure, there are solvers readily available which are able to exploit properties such as convexity, linearity etc. These solvers belong to classes such as *linear programming*, *quadratic programming*, *quadratically constrained quadratic programming*, *non-linear programming* and a wide range of approaches tailored to even more specific problem structures [BSS13; PLB12; BV09].

However, there are many more problems for which no analytical structure is available or which would not be solvable in finite time using precise methods, e.g. non-deterministic polynomial-time hard (NP-hard) problems. This dissertation examines two of them in detail due to their relevance for AR videoconferencing: Human pose tracking and automatic collaborative room alignment. There is no closed and comprehensive mathematical framework which could describe these problems. In fact, there are not even objective ideal solutions available: The alignment and layout of rooms are judged by humans subjectively. Human pose tracking is by necessity an exercise in abstraction, since the precise alignment of all 206 bones in the moving human body is not measurable using today's technology. Lacking a precise mathematical description of the problem, only approximations to an ideal solution are possible.

Typically, such challenges are tackled using approaches from the family of global heuristic optimisation [BR03; GK03]. A heuristic is a user-defined objective function which is queried for a possible solution and returns a score reflecting the quality or "fitness" of the proposed solution. The solution space described by these functions is generally not differentiable, not smooth and not convex. Starting from the 1950s, a whole field of research has evolved which tries to tackle the approximations, simplifications and complexities of these "non-precise" optimisations. Commenting in the 1970s, F. Glover remarks that "[a]lgorithms are conceived in analytic purity in the high citadels of academic research, heuristics are midwifed by expediency in the dark corners of the practitioner's lair" [Glo77]. He however goes on to observe that "[t]he heuristic approach, robust and boisterous, may have special advantages in terrain too rugged or varied for algorithms."

These comments reflect the dissent in the optimisation community of those days, in which proponents of mathematical purity scoffed at the lack of convergence guarantees attributed to heuristic optimisation. Even today, such debates are still ongoing. Writing in 2013, K. Sörensen complains in a highly readable article: "Since a few decades, every year has seen the publication of several papers claiming to present a novel method for optimisation, based on a metaphor of a process that is often seemingly completely unrelated to optimisation" [Sör13]. Certainly, there is a host of over-enthusiastic and under-reflected contributions plaguing the field of heuristic optimisation. Nevertheless, in the absence of comprehensive mathematical models of human cognition or human pose, approximations are the only available means to approach the problems discussed in the following chapters.

Notwithstanding the controversy, today the field of heuristic optimisation encompasses a number of well studied and proven approaches which are widely applied to a wide range of otherwise unwieldy optimisation problems. Among these are *genetic algorithms*, *pattern search* and *simulated annealing*, to name just a few. While none of these are guaranteed to deliver an exact optimum for general applications, they can be expected to find a good approximation in finite time.

The remainder of the chapter will first introduce three important variants of pattern search, then describe the simulated annealing algorithm and finally close with a description of the *annealing particle filter (APF)*.

3.2 Pattern Search

Pattern search and direct search methods are unconstrained optimisation techniques which do not require derivatives of the problem considered [HJ61; KLT03]. Instead of an exact analytical examination of the mathematical problem space, they approach the optimisation as a search problem. First concepts were presented as early as the 1950s. At first, their introduction was slowed by concerns in the mathematical community about their convergence behaviour [LTT00].

The Nelder–Mead method [NM65] from 1965 serves as a typical example for such methods. Conceptually, it elaborates on the evolutionary search proposed by Box *et al.* [Box57] and is also known as the *downhill simplex method*. Despite its age, it serves as a good illustration of the basic concepts of a direct search method.

Assuming an n-dimensional solution space, the method considers a polytope with $n+1$ vertices containing possible solutions. This special case of a polytope is known as a *simplex*, hence the alternative name of the downhill simplex method. A single solution in the n-dimensional solution space is denoted $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$. An objective function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ serves as a heuristic. The vertices of the simplex are $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1}\}$. Algorithm 1 then details the four basic modes of exploration employed by the Nelder-Mead method: Reflection, Expansion, Contraction and Reduction of the simplex encompassing a given solution space. The former three operations aim to change the simplex by modifying the single vertex yielding the highest heuristic function return values. The goal is to shift this single vertex into a configuration leading to lower function values, progressing the convergence towards a global minimum. The other vertices meanwhile remain unchanged. Only the fourth operation, reduction, acts on all $n + 1$ vertices. Thus, for each iteration the heuristic function needs to be queried only once.

Pattern search approaches follow the core concept of the Nelder-Mead method in iteratively querying the solution space until a likely minimum is found. However, they do not employ a polytope of possible solutions over the search space. Instead, pattern search methods attempt to systematically explore the solution space through a series of alterations of a single solution $\mathbf{x}$ according to a specific strategy. Typical strategies used today are the generalized pattern search (GPS) algorithm, [Tor97; AD02], the generating set search (GSS) algorithm [KLT06] and the mesh adaptive search (MADS) algorithm [AD06].

3.2.1 Generalized pattern search

The various forms of the GPS algorithm were described in detail by Torczon *et al.* [Tor97] and shown by Audet *et al.* [AD02] to converge under certain conditions when applying linear constraints. There are also a number of reports of GPS performing well for non-smooth and discontinuous problems [Boo+99; Cho+00], making it a possible candidate for the problems examined in this dissertation.

The general approach of GPS is to sample the solution space around the current iterate solution $\mathbf{x}_k$ at a number of nearby trial points. The basic components required are a generating matrix $\mathbf{C}_k$, a basic matrix $\mathbf{B}$, an algorithm selecting the new step $\mathbf{d}_k$ and further algorithms for updating step size Δ_k and generating matrix $\mathbf{C}_k$. The basic

Algorithm 1 Nelder-Mead Method [NM65]

procedure FINDMINIMUM($\mathcal{X}$)
  $\alpha = 1.0$ ▷ Literature value [NM65]
  $\gamma = 2.0$ ▷ Literature value [NM65]
  $\rho = -0.5$ ▷ Literature value [NM65]
  $\sigma = 0.5$ ▷ Literature value [NM65]
  while $\sqrt{\left(\sum_{\forall \mathbf{x}_i \in \mathcal{X}} f(\mathbf{x}_i) - \bar{f}(\mathbf{x})\right)/n} > 10^{-8}$ **do**
    Sort $f(\mathbf{x}_1) < f(\mathbf{x}_2) < \ldots < f(\mathbf{x}_{n+1})$
    Find the centroid $\mathbf{x}_0$, for solutions $\mathbf{x}_1$ to $\mathbf{x}_N$
    Reflect the weakest point $\mathbf{x}_{n+1}$: $\mathbf{x}_r = \mathbf{x}_0 + \alpha(\mathbf{x}_0 - \mathbf{x}_{n+1})$
    if $\mathbf{x}_1 \leq \mathbf{x}_r < \mathbf{x}_n$ **then** ▷ Reflection of weakest vertex
      Replace $\mathbf{x}_{n+1}$ with $\mathbf{x}_r$
      Return to first step of while-loop
    end if
    if $f(\mathbf{x}_r) < f(\mathbf{x}_1)$ **then** ▷ Expansion of weakest vertex
      $\mathbf{x}_e = \mathbf{x}_0 + \gamma(\mathbf{x}_0 - \mathbf{x}_{n+1})$
      if $f(\mathbf{x}_e) < f(\mathbf{x}_r)$ **then**
        Replace $\mathbf{x}_{n+1}$ with $\mathbf{x}_r e$
        Return to first step of while-loop
      else
        Replace $\mathbf{x}_{n+1}$ with $\mathbf{x}_r$
        Return to first step of while-loop
      end if
    end if
    if $f(\mathbf{x}_r) \geq f(\mathbf{x}_n)$ **then** ▷ Contraction of weakest vertex
      $\mathbf{x}_c = \mathbf{x}_0 + \rho(\mathbf{x}_0 - \mathbf{x}_{n+1})$
      if $f(\mathbf{x}_c) < f(\mathbf{x}_{n+1})$ **then**
        Replace $\mathbf{x}_{n+1}$ with $\mathbf{x}_c$
        Return to first step of while-loop
      end if
    end if
    for all $\mathbf{x}_i \in \mathcal{X}$ **do** ▷ Reduction of all simplex vertices
      $\mathbf{x}_i = \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1)$
      Return to first step of while-loop
    end for
  end while
  Return $\mathbf{x}_1$ as result
end procedure

matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ and the generating matrix $\mathbf{C}_k \in \mathbb{Z}^{n \times n}$ define the pattern $\mathbf{P}_k = \mathbf{B}\mathbf{C}_k$ at iteration k. The family of methods draws its name from this pattern.

New trial steps $\mathbf{d}_k^i$ are generated from the rows of the pattern $\mathbf{P}_k$ in combination with the scalar step size Δ_k. The pattern therefore controls the direction of the step, while the step size determines the length.

$$\left[\mathbf{d}_k^1, \mathbf{d}_k^2, \ldots, \mathbf{d}_k^n\right]^T = \Delta_k \mathbf{P}_k \tag{3.4}$$

$$= \Delta_k \mathbf{B}\mathbf{C}_k \tag{3.5}$$

Commonly, GPS approaches use a fixed basic matrix $\mathbf{B}$ alternating only one component for each step (i.e. an identity matrix). The generating matrix modifies the basic patterns encoded in the basic matrix, yielding new stepping directions. These steps are then applied to the current iterate $\mathbf{x}_k$ in order to generate candidate moves $\mathbf{x}_k^i = \mathbf{x}_k + \mathbf{d}_k^i$. The *ExploratoryMoves* algorithm selects from these possible steps the most promising step for the next iteration by polling the heuristic function for minimum values:

$$\mathbf{d}_k = \operatorname{argmin}_i \left(f\left(\mathbf{x}_k + \mathbf{d}_k^i\right)\right) \tag{3.6}$$

While conceptually simple, a number of different implementations of the *ExploratoryMoves* algorithm exist: Some perform a complete polling of all n possible steps, others use a greedy approach and return with the first step performing better than the current iterate. Torczon *et al.* [Tor97] collect different strategies to the exploration and re-computation of the variables from four established methods. Algorithm 2 shows the general flow shared by all of these methods.

Common to all methods derived from GPS is the differentiation between SEARCH and POLL steps. Searching takes place when the steps around the current iterate yield a better solution. In that case, the step is executed and the new iteration starts with $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta_k \mathbf{d}_k$. If on the other hand all of the steps fail to find a better solution, the current iterate is retained $\mathbf{x}_{k+1} = \mathbf{x}_k$ and only the step size is reduced for a more local exploration in the next iteration. Therefore the step size is also used for detecting convergence, as it keeps shrinking as the algorithm converges on a specific solution.

For a better understanding of the complete pattern search approach, in the following the method presented by Hooke and Jeeves [HJ61] is examined in detail. While they are better known for their popularisation of "direct search", they have also examined pattern search as a specific strategy. In order to accelerate the convergence of the algorithm, their method uses knowledge gained in earlier steps in order to guide the stepping pattern.

Their exploration strategy is outlined in Algorithm 3. In cases were the last step brought an improvement in the heuristic function, the step is simply repeated once more. Only then the surroundings of the new iterate is examined by polling steps generated from the generating matrix $\mathbf{C}_k$ and basic matrix $\mathbf{B}$. In cases where the last step brought no improvement on the heuristic function, the surrounding solution space of the current iterate is examined more closely.

In Hooke and Jeeves' approach, the generating matrix $\mathbf{C}_k$ is partially changed to reflect the previous step. In the case of n decision variables, the first 3^n columns contain fixed

Algorithm 2 Generalized pattern search [Tor97]

procedure GENERALISEDPATTERNSEARCHMETHOD
  Initial solution $\mathbf{x}_0 \in \mathbb{R}^n$
  Initial step width $\Delta_0 > 0$
  while not converged **and** $k <$ MaximumIterations **do**
    Calculate $f(\mathbf{x}_k)$
    Find step $\mathbf{d}_k$ using an *ExploratoryMoves* algorithm
    Calculate $\rho_k = f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k)$
    if $\rho_k > 0$ **then** ▷ Next iteration in SEARCH mode
      Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$
    else ▷ Next iteration in POLL mode
      Set $\mathbf{x}_{k+1} = \mathbf{x}_k$
    end if
    Update Δ_k and $\mathbf{C}_k$
  end while
end procedure

steps along only a single decision variable. The following 3^n columns are set in each iteration depending on the current iterate $\mathbf{x}_k$:

$$\forall \mathbf{c}_k^i \in \mathbf{C}_k \quad (\text{if } i > 3^n)$$

$$\mathbf{c}_{k+1}^i = \mathbf{c}_k^i + \left(\frac{\mathbf{d}_k}{\Delta_k} - \mathbf{x}_k \right) \tag{3.7}$$

$$\mathbf{C}_k = \begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 & 0 & & & \\ 0 & 0 & 1 & -1 & \dots & 0 & 0 & & & \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \mathbf{c}_k^{3^n+1} & \dots & \mathbf{c}_k^{2\times 3^n} \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & & & \\ 0 & 0 & 0 & 0 & \dots & 1 & -1 & & & \end{bmatrix} \tag{3.8}$$

At initialisation, the relevant columns are set to mirror exactly the first 3^n columns. Thus, after each iteration the previous step guides further exploration.

The basic matrix $\mathbf{B}$ is usually chosen to be the identity matrix. However, given knowledge of the problem model may inform the design of this matrix. Especially for cases were variables are known to differ by orders of magnitude, steps of single decision variables can be scaled easily by setting appropriate values in $\mathbf{B}$.

$$\mathbf{B} = \begin{bmatrix} \beta_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \beta_n \end{bmatrix} \tag{3.9}$$

Finally, the step width is recomputed for each iteration as outlined in Algorithm 4. The goal of this update is to achieve a difference between function values greater than zero:

Algorithm 3 Exploration Strategy GPS [Tor97; HJ61]

procedure ExploratoryMove($\mathbf{x}_k, \mathbf{x}_{k-1}, \Delta_k, \mathbf{B}, \mathbf{C}_k, \rho_{k-1}$)
  $\rho_k = \rho_{k-1}$
  if $\rho_k > 0$ **then** ▷ SEARCH: duplicate the previous step
    $\{\mathbf{d}_k^1, \mathbf{d}_k^2, \ldots, \mathbf{d}_k^n\} = \Delta_k \mathbf{B}\mathbf{x}_{k-1} + \Delta_k \mathbf{B}\mathbf{C}_k$
    $\mathbf{d}_k = \text{argmin}_{\mathbf{d}_k^i} \left(f(\mathbf{x}_k + \mathbf{d}_k^i)\right)$ ▷ Find the best step $\mathbf{d}_k^i$
    $\rho_k = f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k)$
  end if
  if $\rho_k \leq 0$ **then** ▷ POLL: Remain at the last position, explore around it
    $\{\mathbf{d}_k^1, \mathbf{d}_k^2, \ldots, \mathbf{d}_k^n\} = \Delta_k \mathbf{B}\mathbf{C}_k$
    $\mathbf{d}_k = \text{argmin}_{\mathbf{d}_k^i} \left(f(\mathbf{x}_k + \mathbf{d}_k^i)\right)$ ▷ Find the best step $\mathbf{d}_k^i$
    $\rho_k = f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k)$
  end if
  Return $\mathbf{d}_k$ and ρ_k
end procedure

$\rho_k f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k) > 0$. Such a result corresponds to a successful iteration, i.e. the heuristic returns lower values for the new iterate. In cases where the new value is lower, the current step size is retained — obviously the solution is moving towards a minimum. If however the new value is not as high as for the previous step, a smaller step size is used for the following iteration — the proposed step would have led away from a minimum. Thus overstepping of minima will reduce the step size and enable a more finely grained search.

Algorithm 4 GPS Step Width Update Algorithm [Tor97; HJ61]

procedure StepWidthUpdate(ρ_k)
  $\theta = 0.5$ ▷ literature value [Tor97]
  $\lambda = 1.0$ ▷ literature value [Tor97]
  if $\rho_k \leq 0$ **then** ▷ POLL: Search vicinity
    $\Delta_{k+1} = \theta \Delta_k$
  end if
  if $\rho_k > 0$ **then** ▷ SEARCH: Search the wider permissible solution space
    $\Delta_{k+1} = \lambda \Delta_k$
  end if
end procedure

3.2.2 Generating set search

Expanding on the previously introduced GPS algorithms, the generating set search (GSS) methods follow the same approach and use the same patterns of exploration. However, upon encountering linear constraints of the form $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ and $\mathbf{C}\mathbf{x} = \mathbf{g}$ in the vicinity of the current iterate $\mathbf{x}_k$, GSS algorithms switch to alternative patterns. These are intended to speed up convergence by avoiding polling of constrained regions of the solution space.

With the introduction of explicit constraints, the solution space gets reduced to a polyhedron $\Omega = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ of feasible solutions. Obviously, not all linear constraints are relevant at each point of the solution space. In a first step, only those constraints which are reachable from the current iterate $\mathbf{x}_k$ are identified. This limitation leads to a set $\mathcal{C}_k$ of relevant constraints. In the following equations, $\mathbf{a}_i$ denotes a single inequality constraint from the matrix $\mathbf{A}$ and $\mathbf{c}_i$ a single equality constraint taken from the matrix $\mathbf{C}$. The scalars g_i and b_i correspond to single entries from $\mathbf{g}_i$ and $\mathbf{b}_i$ from the constraining equations. Finally, the terms $\epsilon_{\max}$ and $\beta_{\max}$ are the threshold values for the boundary distance ϵ and step width modifier β.

$$\epsilon_k = \begin{cases} \epsilon_{\max} & \text{if} \quad \epsilon_{\max} < \beta_{\max}\Delta_k \\ \beta_{\max}\Delta_k & \text{else} \end{cases} \tag{3.10}$$

$$\mathcal{C}_k^I = \left\{ \mathbf{a}_i \;\middle|\; \frac{\mathbf{a}_i^{\mathrm{T}}\mathbf{x}_k - b_i}{\|\mathbf{a}_i\|} \leq \epsilon_k \;,\; i \in \{1, \ldots, n\} \right\} \tag{3.11}$$

$$\mathcal{C}_k^E = \left\{ \mathbf{c}_i \;\middle|\; \mathbf{c}_i^{\mathrm{T}}\mathbf{x}_k = g_i \;,\; i \in \{1, \ldots, n\} \right\} \tag{3.12}$$

$$\mathcal{C}_k = \mathcal{C}_k^I \cup \mathcal{C}_k^E \tag{3.13}$$

In the GPS algorithm by Hooke and Jeeves [HJ61], half of the columns of the generating matrix $\mathbf{C}_k$ contained a fixed set of step patterns while the other half was generated at each iteration from the previous step. In contrast, Kolda *et al.* [KLT07] vary the entire generating matrix $\mathbf{C}_k$ using the relevant constraint set $\mathcal{C}_k$ as a starting point.

The linear constraints $\mathbf{a}_i$ which are included in the relevant set $\mathcal{C}_k^I$ correspond to the outward pointing normals of the constraining polyhedron Ω. Consequently, these normals lend themselves to the definition of infeasible search directions. In the vicinity of the boundary, it would make little sense to do an additional step in a direction which is pointing directly across the boundary. Therefore, these normals are steps which the algorithm does not need to explore further.

Kolda *et al.* use sets of cones in order to express these constraints. In this context, a cone $\mathcal{K}$ is defined as a set of solutions which is closed for scalar multiplication. Thus, if a vector $\mathbf{x}$ is contained in a cone $\mathcal{K}$, then for all $\alpha > 0$ the scaled vector is also contained in the cone: $\alpha\mathbf{x} \in \mathcal{K}$. A cone is thus finitely generated from a set of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_r$:

$$\mathcal{K} = \left\{ \alpha_1\mathbf{x}_1 + \ldots + \alpha_r\mathbf{x}_r \;\middle|\; \alpha_1, \ldots, \alpha_r \geq 0 \right\} \tag{3.14}$$

The nearby linear constraints $\{\mathbf{a}_i | i \in \mathcal{C}_k\} \cup \{\mathbf{0}\}$ now generate the *ϵ-normal cone* $\mathcal{K}_N$ and the *ϵ-tangent cone* $\mathcal{K}_T$. The normal vectors span the ϵ-normal cone, which thus contains infeasible stepping directions. On the other hand, the ϵ-tangent is also the polar of the ϵ-normal cone, therefore containing feasible directions for further search:

$$\mathcal{K}_T = \mathcal{K}_N^\circ = \left\{ \mathbf{x} \;\middle|\; \mathbf{x}^T\mathbf{x}_\mathcal{K} \leq 0 \quad \forall \mathbf{x}_\mathcal{K} \in \mathcal{K}_N \right\} \tag{3.15}$$

Due to the polarity with the ϵ-normal cone, any vector contained within $\mathcal{K}_T$ and no greater than ϵ can be used to perform a step without overstepping the boundaries of Ω.

Lewis, Sherperd and Torczon [LST07] describe a possible implementation of a suitable vector selection. Five different cases must be considered:

1. There may be no constraints applicable.
2. Only bound constraints on the decision variables apply.
3. Equality constraints apply to the decision variables.
4. General linear constraints apply.
5. General linear constraints apply which might be degenerate cases.

Case 1 can be solved trivially by using the unit coordinate vectors $\pm\mathbf{e}_i \in \mathbb{R}^n$, $i = 1, \ldots, n$ as a basis for $\mathbf{C}_k$. This choice is analogous to the first 3^n columns of the generating matrix used by Hooke and Jeeves for GPS as shown in Equation 3.8.

If only bound constraints apply (case 2), the approach is nearly the same. In order to avoid exceeding a boundary, solely unit coordinate vectors not included in the ϵ-normal cone are used.

Case 3 corresponds to a linear subspace of Ω. The generating vectors of the ϵ-tangent cone therefore constitute the spanning vectors of this subspace. From these vectors, an orthonormal base $\mathbf{\Omega}_Z$ is computed. The vectors $\pm\mathbf{e}_i^Z \in \mathbb{R}^n$ $i = 1, \ldots, n_Z$ spanning this subspace $\mathbf{\Omega}_Z$ then serve as the basis for the generating matrix $\mathbf{C}_k$. This approach is the dimension-reduced analogue to case 1.

General linear constraints are considered in case 4. In the first step, the inadmissible subspace $\mathbf{\Omega}_E$ is generated from the vectors $\mathbf{c}_i$ of the equal constraint set $\mathcal{C}_k^E$. The basis $\mathbf{Z}_E$ constitutes the orthogonal of this subspace. As explained previously, the vectors $\mathbf{a}_i$ are the normals of the boundary hyperplanes. If we map them to the new base $\mathbf{Z}_E$ and the resulting set of vectors $\left\{\mathbf{Z}_E^T\mathbf{a}_i \;\middle|\; \mathbf{a}_i \in \mathcal{C}_k^I\right\}$ is linearly independent, these vectors are used as the columns of a new matrix $\mathbf{Q}$.

Computing the right inverse $\mathbf{R}$ of this new matrix $\mathbf{Q}^T$ as a pseudoinverse yields a set of column vectors. These vectors are orthogonal to the lineality space of the ϵ-tangent cone $\mathcal{K}_T$. The lineality space encompasses the actual boundaries of the cone. Searching too close to its volume would therefore only yield steps in the proximity of the infeasible solution space. Meanwhile, choosing vectors orthogonal to it guarantees the exploration of regions that are not too close to the boundaries. In addition, a second matrix $\mathbf{R}_{\text{Null}}$ spans the entire nullspace of $\mathbf{Q}^T$ using an orthonormal base. This method results in a gridded exploration of this admissible solution subspace, even close to the boundary hyperplanes. As this process considers a subspace, the single components of the decision variable are not modified independently, but rather mapped to the subspace and then varied.

The columns of $-\mathbf{Z}_E\mathbf{R}$ and $\mathbf{Z}_E\mathbf{R}_{\text{Null}}$ compose a set of possible search directions:

$$\mathbf{C}_k = [-\mathbf{Z}_E\mathbf{R}\ ,\ \mathbf{Z}_E\mathbf{R}_{\text{Null}}] \tag{3.16}$$

Finally, there are cases where the vectors $\mathbf{a}_i$ contained in the inequality set $\mathcal{C}_k^I$ form a degenerate set. These cases are detected by testing the base-transformed inequality set $\left\{\mathbf{Z}_E^T\mathbf{a}_i \;\middle|\; \mathbf{a}_i \in \mathcal{C}_k^I\right\}$ for linear dependencies. If a dependence is found, a set of vectors for

building $\mathbf{C}_k$ can be calculated from the extremal vertices and rays of the constricting polyhedron Ω. Kolda *et al.* use the *double description method* by Fukuda and Prodon [FP96] for this computation.

In a final step, the outward facing vectors $\mathbf{a}_i \in \mathcal{C}_k^I$ are added to the generating matrix $\mathbf{C}_k$. If the equality set $\mathcal{C}_k^E$ is not empty, these vectors are projected into the corresponding nullspace spanned by the equality constraints first. While this adds search directions which point straight at the boundary hyperplanes, this approach was shown to provide faster convergence for scenarios with extrema close to the boundary by Kolda *et al.* [KLT07].

The price for this additional measure is an additional adaption of step sizes Δ_k. Algorithm 5 shows this adaptation procedure. Instead of using a global value for all steps, each step can be varied in length to avoid leaving the boundaries of the constraining polytope Ω. Meanwhile, the step width update at the end of each iteration remains identical to the GPS method shown in Algorithm 4.

Algorithm 5 GSS Step Width Adaptation Algorithm [KLT07]

procedure StepWidthAdaption($\Delta_k, \mathbf{x}_k, \left\{\mathbf{d}_k^{(1)}, \ldots, \mathbf{d}_k^{(n)}\right\}$)
  for all $i = 1, \ldots, n$ **do**
    Choose the maximum $\tilde{\Delta}_k^{(i)} \in [0, \Delta_k]$ so that $\mathbf{x}_k + \tilde{\Delta}_k^{(i)} \mathbf{d}_k^{(i)} \in \Omega$
  end for
  return $\left\{\tilde{\Delta}_k^{(1)}, \ldots, \tilde{\Delta}_k^{(n)}\right\}$
end procedure

Apart from the differences in computing $\mathbf{C}_k$ and the specifically adapted step widths, the GSS algorithm is nearly identical with the GPS algorithm. For iterates removed far from any boundary hyperplanes, it is even identical. As it is to be expected for this active field of research, there are a number of modifications and adaptations by various groups. Notable for its widespread use after the inclusion in the *Matlab Global Optimisation Toolbox* is the combination of a local GSS search with the more global Lagrangian approach described by Kolda *et al.* [KLT06]. Their method is informed by earlier research, notably by Conn *et al.* [Con+96], which used a Lagrangian framework for constrained optimisation. In this context, the GSS search is used to approximate the gradient around a given point, which is then used to update the Lagrange multiplier estimates by a first-order Hestenes-Powell rule [Hes69; Pow69].

3.2.3 Mesh adaptive search

The mesh adaptive search (MADS) algorithm was proposed by Audet and Dennis in 2006 [AD06]. It is based on the GPS algorithm and shares the general SEARCH-POLL iteration structure. They re-formulate the exploration of possible solutions in the frame based notation used by Coope and Price [CP00]. The frame based notation operates on a mesh which subdivides the solution space into a grid of possible solutions. The distance of the single mesh nodes controls the resolution of the exploration in the SEARCH and POLL stage of the GPS algorithm. It is expressed as the step width $\Delta_k = \Delta_k^M$. The current region on

which the optimisation focuses for a single iteration is called the frame $\mathcal{P}_k$. This frame is a subset of the mesh $\mathcal{M}_k$ explored up to that iteration:

$$\mathcal{M}_k = \bigcup_{\mathbf{x}_i \in \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}} \left\{ \mathbf{x}_i + \Delta_i^M \mathbf{d} \;\middle|\; \forall \mathbf{d} \in \mathbf{C}_i \right\} \tag{3.17}$$

$$\mathcal{P}_k = \left\{ \mathbf{x}_k + \Delta_k^M \mathbf{d} \;\middle|\; \forall \mathbf{d} \in \mathbf{C}_k \right\} \subset \mathcal{M}_k \tag{3.18}$$

Audet and Dennis aim to improve the speed of convergence by decoupling the size of the frame from the resolution of the mesh. Thus, within a given frame, a number of steps of different length might be explored. For the POLL stage, they introduce the new poll size parameter $\Delta_k^P \geq \Delta_k^M$ which is typically set to

$$\Delta_k^P = n\sqrt{\Delta_k^M} \tag{3.19}$$

As the size of the polling frame shrinks, the number of mesh nodes begins to grow, yielding a larger set of possible explorative steps $\mathcal{D}_k$. In contrast, the GPS algorithm only allows steps which have the same width as the size of the current frame, effectively setting $\Delta_k = \Delta_k^M = \Delta_k^P$.

3.2.3.1 Lower triangle mesh adaptive search

In order to facilitate the generation of suitable search directions $\mathbf{C}_k$ in $\mathcal{D}_k$, Audet and Dennis use a stochastic approach in their first examination of the MADS framework, called lower triangle mesh adaptive search (LTMADS). First, a non-singular lower triangle matrix $\mathbf{B}$ with random integer values is generated. The rows of this matrix are then permuted randomly and completed to a positive basis in the solution space $\mathbb{R}^n$. Algorithm 6 shows this process in more detail.

Other than for the previous algorithms, the resulting generating matrix $\mathbf{C}_k$ is composed of random steps. Nevertheless, as the algorithm converges, the set of search directions is also provably dense in the solution space.

3.2.3.2 Orthogonal mesh adaptive search

In a subsequent article co-authored by Audet and Dennis, Abramson *et al.* [Abr+09] demonstrate a deterministic implementation of the MADS framework, called orthogonal mesh adaptive search (ORTHOMADS). The article shows how the general MADS framework can be used in conjunction with a deterministic step generation method. Thus, optimisations performed become repeatable and simultaneously the stepping directions can be optimised for more evenly spaced explorative steps in the current frame $\mathcal{P}_k$. Meanwhile, the variable resolutions used for determining mesh size Δ_k^M and polling frame size Δ_k^P ensure nearly the same convergence behaviour as for the LTMADS algorithm.

The computation of stepping directions begins with a single direction drawn from the pseudo-random Halton sequence [Hal60]. This sequence yields vectors $\mathbf{u}_t \in [0, 1]^n$. The process starts with the integer Halton sequence index $t_k \in \mathbb{N}$, which takes its initial value

Algorithm 6 LTMADS: Random step generation matrix [AD06]

procedure CALCULATERANDOMBOUNDSTEPS(Δ_k^M)
  $l = -\log_4(\Delta_k^M)$
  $\mathbf{b}(l) = \mathbf{0} \in \mathbb{R}^n$ ▷ initialize empty vector
  Pick random index $\bar{a} \in \mathcal{N} = \{1, \ldots, n\}$
  $\mathbf{b}_{\bar{a}}(l) = \pm 2^n$ ▷ Set sign randomly
  for all $i \in \mathcal{N} \setminus \{\bar{a}\}$ **do**
    $\mathbf{b}_i(l) = \texttt{random}() \in \left\{-2^n + 1, -2^n + 2, \ldots, 2^n - 1\right\}$
  end for
  Create basis $\mathbf{L}$ as a lower triangular $(n-1) \times (n-1)$ matrix in $\mathbb{R}^{n-1}$
  Set diagonal terms of $\mathbf{L}$ randomly to $\left\{-2^n, 2^n\right\}$
  Set lower triangle of $\mathbf{L}$ randomly to $\left\{-2^n + 1, -2^n + 2, \ldots, 2^n - 1\right\}$
  Randomly permute rows of $\mathbf{L}$
  Insert zero row at index $\bar{a} \rightarrow \mathbf{L}_\circ$
  $\mathbf{B} = [\mathbf{L}_\circ, \mathbf{b}]$ ▷ Matrix of size $(n) \times (n+1)$
  Basis composition: $\mathbf{C}_k = [\mathbf{B}, \mathbf{b}_{\min}]$ with $\mathbf{b}_{i,\min} = -\sum_{j \in \{1,\ldots,n\}} \mathbf{B}_{ij}$
end procedure

as $t_0 = 20$. An integer counter l_k is retained during the whole optimisation procedure and serves as a storage variable for the selection of the Halton index.

$$l_k = \begin{cases} l_{k-1} + 1 & \text{after successful iterations} \\ l_{k-1} - 1 & \text{after unsuccessful iterations} \\ 0 & \text{for } k = 0 \end{cases} \tag{3.20}$$

$$t_k = \begin{cases} l_k + n + 1 & \Delta_k^P \text{ is smallest step so far} \\ \max_j \left(t_j \mid j \in \{1, 2, \ldots, k-1\}\right) & \text{else} \end{cases} \tag{3.21}$$

The new index t_k determines which vector $\mathbf{u}_{t_k}$ of the Halton sequence will be used in the current iteration. This choice of parameters ensures that every time the POLL step width parameter Δ_k^P reaches a new minimum, the search direction used to create $\mathbf{C}_k$ will move further down the Halton sequence. As the mesh size $\Delta_k^M \leq \Delta_k^P$ converges on zero, the set of Halton directions $\{\mathbf{u}_{t_k}\}_{\forall k}$ considered so far will grow increasingly dense on the unit hyper-sphere, a necessary convergence condition for all MADS algorithms.

So far, the algorithm has only drawn a single direction from the Halton sequence. Before this sequence can be used to generate stepping directions, it must be scaled and rounded to fit into the MADS framework. This operation results in the adjusted Halton direction $\mathbf{q}_k \in \mathbb{Z}^n$. For the computation, a scaling parameter α and the unit vector $\mathbf{e} \in \mathbb{R}^n$ with all elements equal to one are used.

$$\mathbf{q}_k(\alpha) = \text{round}\left(\alpha \frac{2\mathbf{u}_{t_k} - \mathbf{e}}{\|2\mathbf{u}_{t_k} - \mathbf{e}\|}\right) \tag{3.22}$$

$$\alpha_k = \text{argmax}\left(\|\mathbf{q}_k(\alpha)\| \;\middle|\; \|\mathbf{q}_k(\alpha)\| \overset{!}{\leq} 2^{l_k/2}\right) \tag{3.23}$$

$$\mathbf{q}_k = \mathbf{q}_k(\alpha_k) \tag{3.24}$$

The resulting adjusted Halton vector $\mathbf{q}_k \in \mathbb{Z}^n$ has a norm close to $2^{l_k/2}$, thus fitting into the step width scheme used in the general MADS framework.

Finally, the Householder transform [Hou58] uses this single vector to generate a orthogonal integer basis $\mathbf{H}$ in the solution space $\mathbb{R}^n$.

$$\mathbf{v} = \frac{\mathbf{q}_k}{\|\mathbf{q}_k\|} \tag{3.25}$$

$$\mathbf{H} = \|\mathbf{q}_k\|^2 \left(\mathbf{I}_n - 2\mathbf{v}\mathbf{v}^{\mathrm{T}}\right) \tag{3.26}$$

As shown by Abramson *et al.*, this procedure will create a dense set of directions. The set is used both in positive and negative directions:

$$\mathbf{C}_k = [\mathbf{H}_k, -\mathbf{H}_k] \tag{3.27}$$

The advantage of ORTHOMADS compared to LTMADS is twofold. Firstly, the deterministic approach generates repeatable optimisation operations without the need for probabilistic proofs of convergence. Secondly, the search directions are guaranteed to be mutually orthogonal for a more evenly spaced exploration of the solution space. This property helps to speed up convergence.

3.2.3.3 Summary on MADS algorithms

Regardless of the step generator used to find $\mathbf{C}_k$, the theoretical advantage of MADS over GPS is an overall faster convergence. In addition, it supports higher search resolution and allows for a more graceful handling of oracular constraints (i.e. the heuristic function can act as a barrier function simply by returning high values for undesired solutions).

3.3 Simulated Annealing

In 1983 S. Kirkpatrick *et al.* [KGV83; Kir84] proposed the SA optimisation framework. Statistical mechanics, a sub-field of condensed matter physics, provide the theoretical background for their framework. This discipline examines the aggregate properties of large numbers of atoms in different states. As these properties arise from very large sample sizes (typically 10^{23} $1/cm^3$ atoms), at thermal equilibrium the observed properties must be the statistically most probable for the system.

Kirkpatrick *et al.* connect the statistical nature of such problems with the observation that the discrete states of individual atoms produce these properties. In their entirety,

these collections of atoms therefore express a combinatorial problem. Finding the optimal combination of states over all atoms thus solves a global energy minimisation problem. In their original paper, they use the example of magnetic spin direction and the resulting energy between atoms with different primary spin directions to illustrate this idea. They observe that in metallurgy the process of annealing, i.e. the gradual cooling of materials, is used to achieve such energy-optimised alignments in materials. In the context of solid materials, this effect is observable in crystalline solids or metals, where slower cooling leads to less material defects and larger volumes of uniform alignment.

As these processes lead to the observable minimisation of energy in scenarios with an extremely high number of decision variables, Kirkpatrick *et al.* argue that similar approaches should also work for theoretical optimisation problems. They differ from previous iterative approaches to optimisation by allowing not only steps that reduce the energy. Adapting the Metropolis procedure [Met+53] to combinatorial optimisation introduces a small chance of performing steps which actually increase the system energy. The chance of performing such non-minimising steps is then coupled to the *temperature* of the system. The further the system has "cooled down", the less likely the simulated annealing is to select iterates which increase energy. The effect of these upwards steps is to escape local minima in early iterations, ensuring convergence on the actual global minimum.

For each iteration, a small change is applied to each decision variable making up the current iterate $\mathbf{x} \in \mathbb{R}^n$. After the alteration, the overall change in energy ΔE is computed. If the energy increases, i.e. $\Delta E > 0$, the probability that the new configuration is accepted is computed as $P(\Delta E)$ from the change in energy, the constant k_B[1] and the current temperature T.

$$P(\Delta E) = \exp\left(\frac{-\Delta E}{k_B T}\right) \tag{3.28}$$

Therefore, the lower the temperature T drops, the smaller the probability of accepting a positive change in energy becomes. The term $P(\Delta E)$ is tested against a number randomly drawn from the uniform interval $(0, 1)$. If the number is smaller than $P(\Delta E)$, the new pose is accepted. In case the new configuration of decision variables $\mathbf{x}_{\text{new}}$ is rejected, the process returns to the last accepted configuration and a different set of slight alterations are applied. These steps are repeated until either a better solution with $\Delta E(\mathbf{x}_{\text{new}}) < 0$ is found or the a new alignment with $\Delta E(\mathbf{x}_{\text{new}}) > 0$ gets randomly accepted. Algorithm 7 shows the general outline of the simulated annealing method. For a given temperature T the Metropolis algorithm is applied until the system achieves a state of equilibrium. The SA algorithm then lowers the temperature T gradually and again applies the Metropolis algorithm. The optimisation concludes once no further improvements are apparent after lowering the temperature. The speed at which the temperature is decreased is a vital parameter of the process — too fast a decrease would lead to insufficient exploration of the state space, while too slow a decrease would lead to a slow performance. As the temperature falls over the course of the optimisation, the physical annealing process employed in generating metals and crystals with high structural uniformity is emulated.

[1] In the original Metropolis procedure, this value is set to the Boltzmann constant.

Algorithm 7 Simulated Annealing: General Algorithm [BM95]

procedure FINDGLOBALOPTIMIM($E(\mathbf{x})$, AnnealingSchedule, N_σ)
  Choose $T_0 > 0$
  Choose $\mathbf{x}_0$, let $\mathbf{x}_{\text{current}} = \mathbf{x}_0$
  repeat
    Reset $N_s = 0$
    repeat
      Generate permutation of $\Delta\mathbf{x}$
      Calculate $\Delta E = E(\mathbf{x}_{\text{current}} + \Delta\mathbf{x}) - E(\mathbf{x}_{\text{current}})$
      $N_{\text{rand}} = \texttt{uniform}(0, 1)$ ▷ Generate random number
      if $P(\Delta E) \leq N_{\text{rand}}$ **then**
        Set $\mathbf{x}_{\text{current}} = \mathbf{x}_{\text{current}} + \Delta\mathbf{x}$
        Increment $N_s = N_s + 1$
      end if
    until $\Delta E < \sigma$ for the last N_σ iterations
    Choose ρ from Annealing Schedule
    $T = \rho T$ ▷ Lower temperature
  until $N_s = 0$
end procedure

Kirkpatrick *et al.* observe that this approach not only avoids local extrema, but also shows inherent coarse-to-fine traits when applied to combinatorial problems. Their work has since then sparked a host of consecutive research applying the method of simulated annealing to a wide range of problems, especially those classified as non-deterministic polynomial-time complete (NP-complete). Nevertheless, it should be noted that while probabilistic bounds of convergence can be specified [GKR94], there is at the time of writing no proof for convergence on the global optimum in finite time.

A number of independent groups has since adapted the simulated annealing method to various continuous optimisation problems. A 1994 paper by Brooks and Morgan [BM95] presents a good summary of the differences between the purely combinatorial SA approach and the continuous methods. They argue that computers work on discrete values by design. This fact re-frames the continuous problem as an exploration of possible combinations of decision variables, where each variable is assigned a high number of potential states. These states in turn simply refer to discrete values the decision variable might take.

Brookes and Morgen also lay out the general challenge of parameter and problem dependence for SA algorithms. For instance, the threshold N_σ sets the limit for iterations the algorithm proceeds without any improvement. After this number of iterations has passed without progress, the procedure assumes that it has reached an equilibrium and lowers the temperature. Therefore, the threshold N_σ should be set to a large value in order to avoid pre-mature cooling of the system. At the same time, a high value means that the optimisation necessarily runs through more iterations, affecting convergence time. A similar trade-off must be made in the cooling schedule itself. As the cooling factor ρ is raised, the system cools down more slowly. While this increases the robustness of convergence on the global optimum, it also means that more steps must be taken to find

it. In the same vein, a high initial temperature T_0 will ensure that each point in the parameter space has a chance of being queried. Simultaneously, it also means that more time will be spent "cooling down" from this high temperature for convergence.

So while the algorithm is attractive especially for complex problems and high dimensions, it can be cumbersome to fine-tune to a problem and may be ineffective in comparison to other solver methods. Nevertheless, its ability to climb out of local optima makes it a very powerful tool for the problems we shall explore in Chapters 4 and 5.

3.4 Annealing Particle Filter

The previous optimisation approaches are tailored to solving time-stationary problems. However, tracking problems encountered in AR are by definition non-stationary and require consideration of preceding data. For example, the solution of the human pose tracking problem at any observation frame is highly dependant on previous results. Therefore, an extension of the optimisation methods to non-stationary processes is warranted in order to tackle such challenges.

Many well-studied algorithms from the class of Markov chain Monte Carlo (MCMC) methods are available for estimating the probability distributions for stationary processes. The Metropolis-Hastings algorithm [Met+53] or the Gibbs sampling [GG84] are both typical examples for these methods. They are employed for reconstructing distributions which are only partially or indirectly observable. Such probability distributions, often termed probability density function (PDF), give the likelihood of a random variable taking a specific value.

The MCMC methods solve optimisation problems by considering the system as a probabilistic model which connects observations and the underlying system state. For energy minimisation problems, the observation thus takes the role of the energy. The system state is represented by the decision variable for which we optimise. By finding the global maximum of the PDF, the corresponding system state yields the values of the decision variable that lead to the minimal system energy.

The critical step lies in moving from stationary MCMC scenarios to sequential Monte Carlo (SMC) problems, where the solution of a given problem at a specific frame of observation is connected to the solution of a previous observation. Sequential Monte Carlo (SMC) methods are therefore well suited for problems tracking the development of a given system over time, e.g. user tracking. The method is related to the *Kalman filter*, but performs better on non-linear systems and non-Gaussian noise models at the expense of higher computational costs.

The general form of problems approached by SMC can be stated as the attempt to find a solution $\hat{\mathbf{x}}_t \in \mathbb{R}^n$ which maximises the posterior probability $P(\mathbf{x}_t|\mathbf{Z}_t, \mathcal{X}_t)$. The posterior probability gives the probability of having the hypothesis $\mathbf{x}_t$ as the underlying state of a system given the current observations $\mathbf{Z}_t$ and the preceding states $\mathcal{X}_t$.

$$\hat{\mathbf{x}}_t = \operatorname{argmax}\left(P\left(\mathbf{x}_t|\mathbf{Z}_t, \mathcal{X}_t\right)\right) \tag{3.29}$$

with

$$\mathcal{X}_t = \{\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \ldots, \mathbf{x}_{t-T}\} \tag{3.30}$$

Interpreting the process as a Markov chain with $P(\mathbf{x}_t|\mathcal{X}_t) = P(\mathbf{x}_t|\mathbf{x}_{t-1})$ simplifies this expression.

$$\hat{\mathbf{x}}_t = \operatorname{argmax}\left(P(\mathbf{x}_t|\mathbf{Z}_t, \mathbf{x}_{t-1})\right) \tag{3.31}$$

Applying Bayes' theorem to the posterior probability and assuming independence between current observation and previous states $P(\mathbf{Z}_t \wedge \mathbf{x}_{t-1}) = P(\mathbf{Z}_t)\, P(\mathbf{x}_{t-1})$, the posterior can be written as follows:

$$P(\mathbf{x}_t|\mathbf{Z}_t, \mathbf{x}_{t-1}) = \frac{P(\mathbf{Z}_t, \mathbf{x}_{t-1}|\mathbf{x}_t)\ P(\mathbf{x}_t)}{P(\mathbf{Z}_t)\ P(\mathbf{x}_{t-1})} \tag{3.32}$$

It is a common assumption that observations are independent from the dynamics of decision variable.[2]

$$P(\mathbf{Z}_t, \mathbf{x}_{t-1}|\mathbf{x}_t) = P(\mathbf{x}_{t-1}|\mathbf{x}_t)\ P(\mathbf{Z}_t|\mathbf{x}_t) \tag{3.33}$$

This assumption splits the problem of the conditional probability into two sub-problems: A stochastic "motion" model $P(\mathbf{x}_{t-1}|\mathbf{x}_t)$ of the system dynamics[3] and an observation model $P(\mathbf{Z}_t|\mathbf{x}_t)$. When assuming an uniform distribution of the priors, the expression becomes simpler. Note that k is simply a normalising constant in the following equation.

$$P(\mathbf{x}_t|\mathbf{Z}_t, \mathcal{X}_t) = k\, P(\mathbf{x}_{t-1}|\mathbf{x}_t)\ P(\mathbf{Z}_t|\mathbf{x}_t) \tag{3.34}$$

For most problems, it is hard or even impossible to gain a complete observation model $P(\mathbf{Z}_t|\mathbf{x}_t)$. The prior probabilities $P(\mathbf{Z}_t)$, $P(\mathbf{x}_t)$ and $P(\mathbf{x}_{t-1})$ are usually not available as well. Even if such models were feasible, evaluating them in their entirety would be costly. For this reason, particle filters usually leave the strict Bayesian framework and turn to a simpler weighting function $w(\mathbf{x}_t, \mathbf{Z}_t) \propto P(\mathbf{Z}_t|\mathbf{x}_t)$. A reduction of the system dynamics model $P(\mathbf{x}_t|\mathbf{x}_{t-1})$ to a uniform distribution is another common simplification. Putting the simplified expression into the problem stated in Equation 3.31 leads to a re-statement of the optimisation problem that can be solved by considering only the observation likelihood model.

$$P(\mathbf{x}_t|\mathbf{Z}_t, \mathcal{X}_t) \propto w(\mathbf{x}_t, \mathbf{Z}_t) \tag{3.35}$$

$$\hat{\mathbf{x}}_t = \operatorname{argmax}\left(w(\mathbf{x}_t, \mathbf{Z}_t)\right) \tag{3.36}$$

The weighting function, an approximation of the observation likelihood, thus becomes the key to solving such optimisation problems. Please note that other than for the previous methods, SMC is expressed as an energy maximisation problem. Instead of minimising an energy term, here we are searching for the highest likelihood in the PDF.

Using these weighting functions, SMC methods can estimate posterior densities in the state space of non-stationary processes. The general idea lies in the parallel consideration of a number of hypotheses $\mathbf{p}_t^{(n)} \equiv \mathbf{x}_t^{(n)} \in \mathbb{R}^n$ independently, effectively implementing

[2] This assumption applies to most optical tracking tasks. Care should be taken in other scenarios.

[3] Approaches based on learned models build this term from training data

the Bayersian recursion equations [GSS93]. This parallel and independent testing of hypotheses has also lead to the term *particle filters*. Each hypothesis is treated as a particle which is tested against the observed data. In order to conduct these tests, likelihood approximation functions as defined in Equation 3.35 are used. Their use is necessitated by the impossibility of knowing the *real* likelihood of any observation in dependence to an hypothetical system state.

The sum of all current hypotheses, or particles, is defined as the set $\mathcal{S}_t$. For each particle $\mathbf{p}_t^{(n)}$, there is an associated weighting value $w_t^{(n)}$ returned by the weighting function defined in Equation 3.35. The particles and their weights are therefore associated in the combined set $\hat{\mathcal{S}}$.

$$\mathcal{S}_t = \left\{ \mathbf{p}_t^{(1)}, \mathbf{p}_t^{(2)}, \ldots, \mathbf{p}_t^{(N_p)} \right\} \tag{3.37}$$

$$w_t^{(n)} = w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right) \tag{3.38}$$

$$\hat{\mathcal{S}}_t = \left\{ \mathbf{p}_t^{(1)}, w_t^{(1)}, \mathbf{p}_t^{(2)}, w_t^{(2)}, \ldots, \mathbf{p}_t^{(N_p)}, w_t^{(N_p)} \right\} \tag{3.39}$$

$$\tag{3.40}$$

The basic steps with which the particles are processed within a generic particle filter are

- **Scattering** the old particle set $\mathcal{S}_{t-1}$ to $\mathcal{S}_t$
- **Weighting** to determine $w_t^{(n)}$ for each $\mathbf{p}_t^{(n)}$, resulting in $\hat{\mathcal{S}}_t$
- **Resampling** $\hat{\mathcal{S}}_t$ into a new, unweighted particle set $\mathcal{S}_{t+1}$

The *condensation algorithm* is a commonly used implementation of this particle filter procedure. This algorithm was proposed by Isard and Blake [IB98] as a tool for tracking the silhouettes of persons in video sequences. In this context, the time-dependency becomes obvious for simple tracking scenarios: The position of a user at time t is directly dependent on the position in the preceding moment $t-1$, provided that observations are made at a sufficiently high rate.

Their process is illustrated in Figure 3.1. Adding multivariate Gaussian noise $\mathcal{N}(\mathbf{0}, \Sigma_{\mathrm{IF}})$ with zero mean and a covariance of Σ_{IF} scatters the initial set of particles $\mathcal{S}_{t-1}$. The covariance Σ_{IF} controls the amount of scattering for each decision variable and should be tailored to the allowed range of each variable. The constant k_{IF} controls the amount of scattering introduced between two steps in time[4].

$$\forall \mathbf{p}_{t-1}^{(n)} \in \mathcal{S}_{t-1}: \quad \mathbf{p}_t^{(n)} = \mathbf{p}_{t-1}^{(n)} + k_{\mathrm{IF}} \mathcal{N}(\mathbf{0}, \Sigma_{\mathrm{IF}}) \tag{3.41}$$

After scattering, the weights $w_t^{(n)}$ are computed for each particle

$$\forall \mathbf{p}_t^{(n)} \in \mathcal{S}_t: \quad w_t^{(n)} = w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right) \tag{3.42}$$

[4] In image processing, the time steps are determined by the intervals between new frames coming from a camera.

Then in a final step the weighted particles are resampled to a new set $\mathcal{S}_{t+1}$. The likelihood of a particle being included in the new set is directly proportional to its weight relative to all other particles. Therefore, particles with high weights, i.e. a high observation likelihood, have a higher chance of populating the new particle set. This preference acts as a gradual maximisation of particle weights over the entire set of hypotheses as the tracking progresses. The tendency of the particles to converge at the maxima of the solution space is counteracted by the scattering shown in Equation 3.41. Scattering thus leads to a continuous exploration of the solution space, preventing erroneous convergence on local extrema.

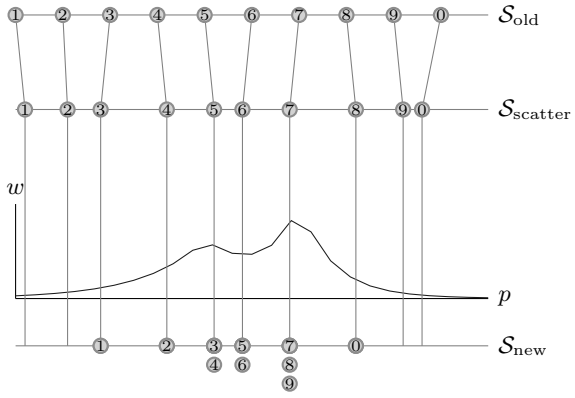


Figure 3.1: Generic Particle Filter illustrating the three essential steps of scattering, weighting and resampling over a simplified function

A generic particle filter needs a high number of particles in order to provide sufficient coverage of the solution space. In addition, there is always a chance that it becomes trapped in local maxima if not adjusted carefully. In order to improve exploration of the solution space and reduce the number of particles needed, Deutscher *et al.* [DBR00; DR05] proposed the annealing particle filter (APF). Their design performs several scatter-weight-resample cycles with varying detail for each observation frame, whereas the classical condensation algorithm only conducts a single pass.

In order to achieve a coarse-to-fine exploration of the solution space, Deutscher *et al.*'s approach combines the particle-based condensation algorithm with the idea of annealing familiar from the SA algorithm introduced in Section 3.3. A set of particles is used to repeatedly sample the solution space for each observation, while the weighting function and resampling process are subjected to a slow cooldown schedule. The basic approach is shown in Algorithm 8.

The APF approach introduces a number of changes to the condensation algorithm. Most obvious are the additional iterations within a single observation frame. Acting over these new cycles is a gradual cooling process applied to the weights generated from single particles. Instead of using the original observation likelihood function (as seen in Equation 3.42), each annealing step m uses a modified weight $w_{t,m}^{(n)}$.

Algorithm 8 Algorithm of the annealing particle filter with N_{APF} annealing steps

set $\mathcal{S}_{t,0} = \mathcal{S}_{t-1,N_{\mathrm{APF}}}$
for all $k = \{1, \ldots, N_{\mathrm{APF}}\}$ **do**
  scatter $\mathcal{S}_{t,m-1}$ to get $\mathcal{S}_{t,m}$
  weight $\mathcal{S}_{t,m}$ to get $\hat{\mathcal{S}}_{t,m}$
  resample $\hat{\mathcal{S}}_{t,m}$ to prepare for next iteration
end for
calculate $\hat{\mathcal{S}}_t^{\mathrm{opt.}}$ from particles of last set $\mathcal{S}_{t,N_{\mathrm{APF}}}$

$$w_{t,m}^{(n)} = w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)^{k_{\mathrm{ann}}} \tag{3.43}$$

The exponent k_{ann} directly controls the contrast of the weighting function. In publications, it is often referred to as the temperature T_m of the annealing process. Initialising k_{ann} with low values at the beginning of each annealing cycle, the weighting function is smoothed. Over the course of the annealing process, the value of k_{ann} is gradually raised according to a pre-defined annealing schedule, leading to more pronounced extrema in the weighting function. The weighting function starts as a rather flat hyperplane over the solution space. The particles move over this hyperplane and gradually converge on the emerging extrema as the values for k_{ann} are increased. Deutscher *et al.* [DR05] determine the current value for k_{ann} by performing a gradient descent search over the particle survival rate α between iterations using the survival rate diagnostic function $d_{\mathrm{surv}}\left(k_{\mathrm{ann}}\right)$.

$$\alpha_k = \frac{d_{\mathrm{surv}}\left(k_{\mathrm{ann}}\right)}{N_p} \tag{3.44}$$

The intended survival rates are normally chosen to encourage diverse populations for the first iterations. The diversity ensures a thorough exploration of the solution space. Later iterations enforce a lower survival rate, effectively culling particles with low likelihood scores and leading to convergence of the particles on the extrema.

The scattering of particles between the annealing iterations is performed by adding Gaussian noise $\mathcal{N}(\mathbf{0}, \Sigma_{\mathrm{ann}})$. Other than the constant noise used between observation frames (seen in Equation 3.41), the covariance Σ_{ann} is computed from the particles scattered over the solution space at the annealing step $m - 1$.

$$\forall \mathbf{p}_{t,k-1}^{(n)} \in \mathcal{S}_{t,k-1}: \quad \mathbf{p}_{t,m}^{(n)} = \mathbf{p}_{t,m-1}^{(n)} + k_{\mathrm{ann}}\mathcal{N}(\mathbf{0}, \Sigma_{\mathrm{ann}}) \tag{3.45}$$

This adaptive scattering acts as a soft partitioning of the solution space, encouraging convergence on decision variables where a good solution has been found and furthering exploration for yet ambiguous decision variables.

The inter-frame steps meanwhile aim for wide scattering and exploration of different alternative poses, while conserving pose information from the previous time step. This scattering between observations echoes the previously mentioned motion model $P\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right)$. While no such model is provided explicitly in most APF implementations, the normal scattering acts as a Gaussian motion model which takes the last pose and uses it to derive a number of alternative poses for the next observation frame.

The overall function of the APF is illustrated in Figure 3.2. Figure 3.3 shows a comparison with the generic particle filter. In comparison with a general particle filter, such as the sequential importance resampling particle filter (SIRPF) [GSS93], the APF shows a better performance especially for high dimensional solution spaces. This property recommends it for applications such as human pose tracking, where the various degrees of freedom in the human skeleton lead to decision variables with more than 30 DoF easily.

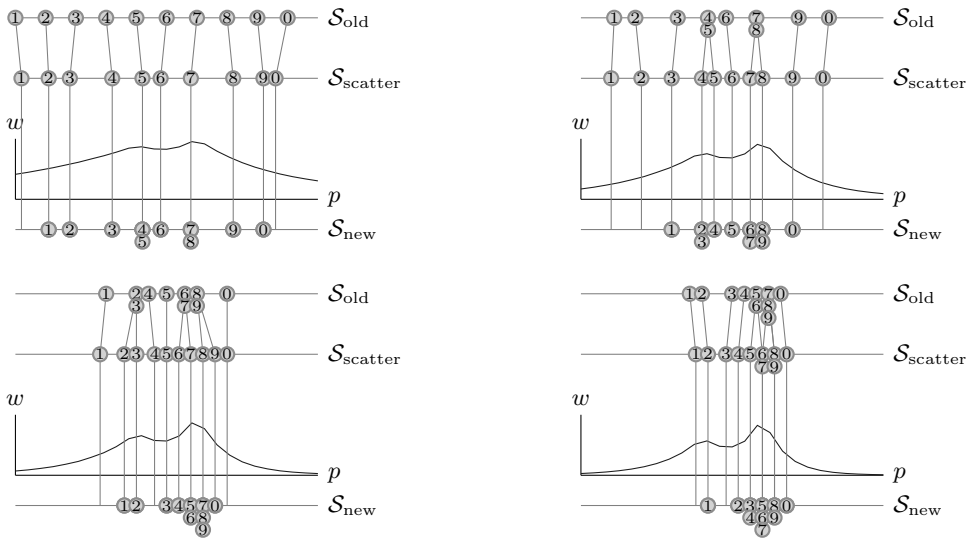


Figure 3.2: Idealised Annealing Particle Filter over four annealing steps, from left to right, top to bottom

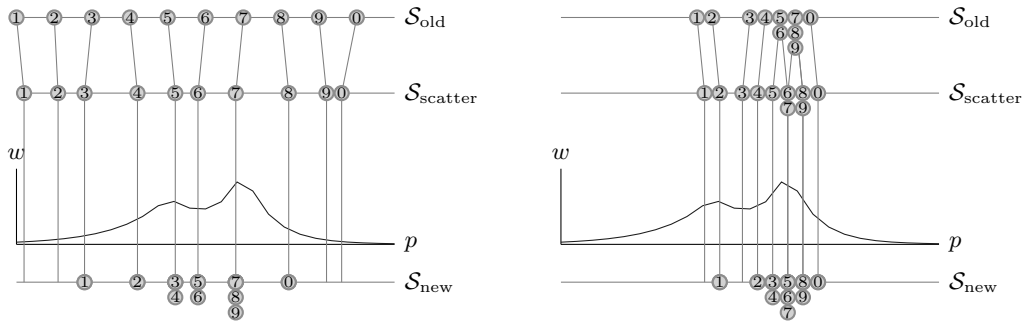


Figure 3.3: Direct comparison between a generic particle filter (left) and an annealing particle filter (right, shown after last step)

3.5 Summary of Chapter

This chapter introduced general concepts of heuristic search and global optimisation. A number of popular algorithms were discussed in detail. The family of pattern search algorithms and the simulated annealing algorithm appear as possible approaches to time-stationary problems. The concept of annealing extends to sequences of observations in the annealing particle filter (APF). The list of approaches to global optimisation presented here is by no means exhaustive — many more algorithms such as *genetic algorithms*, *particle swarm optimisation* etc. were not included. The selection was guided by both general interest — pattern search approaches serve as de-facto standards — as well as pragmatism — the APF method is a well established approach to human pose tracking.

The chapter explained the general approach for each of the selected methods. Where suitable, differences between algorithms were highlighted. The discussion includes major characteristics, advantages and shortcomings as described in literature.

In the following chapters, the application of global optimisation to two problems from the field of augmented reality videoconferences is studied. Both human pose tracking and automatic consensus reality generation are problems without an analytical solution. In the case of human pose, even an experimentally established ground truth is only an approximation of the mostly unobservable skeleton. While it is possible to fix markers to the surface of the skin, the actual state, position and shape of joints and bones can only be approximated with current technology. Even more abstract is the problem of consensus reality building. The functions guiding the optimisation are conventions derived at least partially from observations on human psychology and perception of space. An analytical solution is entirely impossible, while even the establishment of a fixed ground truth is problematic at best.

In the following, we shall see how these problems can nevertheless be approached: Specially designed functions act as approximations of the underlying, unobservable systems. In the case of human pose tracking, this system is the human skeleton. The algorithm tries to guess at the most likely pose from a sequence of observations. In the case of the consensus space generation, the system encompasses the user's perception and innate preferences concerning social interaction. The algorithm therefore uses mathematical abstractions of these preferences to find a satisfactory alignment of the participating rooms.

Chapter 4

Human Pose Tracking

4.1 Introduction

The videoconferencing system tracks the user's position and posture for a number of different tasks. Hand poses serve as the input for interaction with AR content and the menu structure. Tracking the poses of further joints of the human body also provides information on current general gaze direction, gesturing and even emotional state of the user. In addition, the user's position within the room is one of the parameters used for optimising consensus space computation (discussed in Chapter 5).

Human pose tracking is a challenging problem due to the high dimensionality of the human posture space and the complex shape of the human body. Humans exhibit a wide range of shapes and sizes. The problem of guessing the current pose of the body algorithmically is complicated by additional layers of cloth, often times loose fitting and with their own dynamics. Frequent self-occlusions by other parts of the body pose another challenge, especially for non-frontal observations.

Attempts at solving this complex problem have created their own active field of research over the last decades. An article by Poppe [Pop07] provides a good summary of research directions up to 2007. Up to that time, most approaches fell either into the group of monocular pose tracking or used a multi-camera approach. Monocular pose tracking tries to reconstruct a user's pose using data provided by a single RGB camera. These approaches suffer heavily from phenomena like self-occlusion, excessive computational complexity and pose ambiguity. The problem of monocular pose tracking is to date not solved conclusively, although a number of specialised systems show promise in constrained scenarios and with simplified models.

Multicamera methods tackle the problem of self-occlusion and pose ambiguities by observing the user from several directions at once. Such techniques are used successfully for markerless motion capture and employed for high-fidelity motion extraction. The high quality comes at the price of high computational cost, something a responsive videoconferencing system should try to avoid.

In recent years, these two approaches have been joined by depth image based pose tracking systems. Although a number of older publications on this matter exist (most using stereo-cameras for generating depth images), such systems suffered initially from high

costs, lacking real-time capability and other problems. This changed dramatically with the introduction of Microsoft's Kinect system in 2010[1]. While the hardware itself was already a great step forward in terms of quality and price, the pose tracking system demonstrated the power of machine learning approaches in computer vision systems. Developed by a collaboration of well-known researchers like Shotton, Izadi, Newcombe *et al.* [Sho+11], the human pose tracking shipped with the Kinect combined stochastic models of human pose with a depth-driven body part classification algorithm. Subsequent publications by the same research group have since elaborated on the methods used, introducing techniques such as joint regression [Gir+11] and the Vitruvian manifold classification [Tay+12]. In the wake of these publications, several teams focussed on refining the body part classification using machine learning approaches. The focus on these methods marks a wider shift from generative human pose tracking to discriminative methods. The *generative methods* use a body model in order to create pose hypothesis which are matched against the actual observation. In contrast, *discriminative methods* apply pattern recognition algorithms to the observed data in order to find likely positions of body parts.

Most current approaches combine such discriminative body part classifiers with stochastic generative models of motion and observation. At that point, the methods set out in Section 3.4 become applicable to the problem of human pose tracking. As shown in Equation 3.35, the observation likelihood lies at the heart of these stochastic methods. It effectively provides a measure of confidence that a given pose hypothesis is supported by the actual, observed data. In previous research, such functions are derived from edge or feature matching between a deformable body model and the current observation, as shown by Isard *et al.* [IB98], Azad *et al.* [AAD08] and Deutscher *et al.* [DR05]. Azad *et al.* [AAD08] used edge cues supported by separate hand and head trackers, an approach shared with Bernier *et al.* [BCB08], who considered a combination of 3D contour points and an additional hand tracker. Meanwhile Darby *et al.* [DLC08] performed tracking only on the 3D contour points without additional body part tracking, a simplistic approach also reflected in the work done by Fontmarty *et al.* [FLD07]. These approaches date before the introduction of the Kinect and therefore focus on stereo camera or multi-camera scenarios.

Once affordable and comparatively precise depth sensing cameras became available, the focus of research shifted towards weighting functions evaluating depth images. In consequence, more recent approaches to pose tracking work directly on depth images and forego the silhouette extraction characteristic of older methods. Early examples of such methods can be found in the publications by Zhu *et al.* [ZF09] or Ganapathi *et al.* [Gan+10]. These compare a simplified body model directly with the observed data, supported by an additional key-point detection.

The various methods mentioned above perform their likelihood approximation based on depth and colour cues. In the meantime, tools like the point cloud library (PCL) built by Rusu *et al.* [RC11] have paved the way towards an evaluation of pose hypotheses directly in 3D space.

This approach significantly reduces the need for reprojecting complex mesh models onto the image plane, as done by Ganapathi and Zhu. A further advantage of a point

[1] Prices for depth cameras fell by as much as 3 orders of magnitude, while resolution and framerate were improved.

cloud based approximation lies in the inherent extensibility to multiple data sources. In the AR videoconferencing system, four to eight depth sensors are observing the room. The collected observations are combined easily into a single point cloud which is then handed over to the observation likelihood function. This function can than efficiently derive the likelihood for an arbitrary number of observation sources. On the other hand, image-centric approaches would need multiple re-projections into the image plane of all observing cameras in order to achieve true multi-sensor fusion, making multi camera fusion computationally expensive.

In this chapter, a weighting function for computing observation likelihood from 3D point clouds will be introduced. The function is described and evaluated on real-world datasets. Subsequently, an APF is designed to use the approximation function for human pose tracking on depth data provided by a Kinect camera. It should be noted that the likelihood approximation derived in this chapter is applicable in any stochastic solver structure, potentially even in time-stationary approaches for single frame pose recognition.

With regard to the evaluation of the likelihood approximation function, it is interesting to note that previous evaluation of such functions done by Fontmarty *et al.* [FLD09] and Lichtenauer *et al.* [LRH04] are not directly applicable to data gathered by depth sensing cameras. While most publications on human pose tracking present extensive descriptions of inference methods and keypoint detectors, the design of the underlying body model and the associated likelihood approximation function receive relatively little attention to date. There is a good overview provided by Sigal *et al.* [SBB10] covering approaches up to 2010. Following publications on body part detection by Plagemann *et al.* [Pla+10] as well as Shotton *et al.* [Sho+11], research instead tended to concentrate on machine learning approaches to pose recognition or improvements on the APF structure. Likelihood approximation functions meanwhile still rely on silhouette, 2.5D gradients and edge features [HG12]. Multi-camera approaches such as proposed by Schönauer and Kaufmann [SK13] merge the 2D tracking data of several camera units, but do not perform a tracking on the full 3D scene. While there is a recent article by Martínez-Zarzuela *et al.* [Mar+14] demonstrating a merging of point clouds from various cameras on a single server, even their approach still draws on the Kinect pose tracker for user pose tracking.

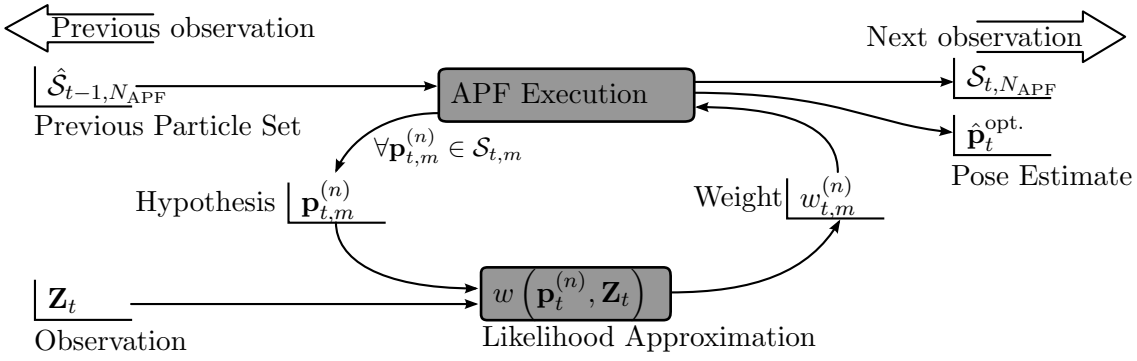


Figure 4.1: Integration of the likelihood approximation into the APF process.

The remainder of the chapter is structured as follows: Section 4.2 describes the likelihood approximation function and the underlying body model in detail. This function is then integrated into an APF framework in Section 4.3. The interaction between the tracking framework and the likelihood approximation is also shown in Figure 4.1. The goal of this integration is to track human body posture over a sequence of frames captured by a depth sensitive camera system. Section 4.4 describes the evaluation of both the tracking framework and the underlying likelihood evaluation function. The chapter closes with a summary of the framework and its significance in the context of AR videoconferencing.

4.2 Point Cloud Based Likelihood Approximation

4.2.1 Mathematical Background

As shown in detail in Section 3.4, stochastic approaches working on series of observations attempt to find the best pose hypothesis $\hat{\mathbf{p}}_t \in \mathbb{R}^n$ given a number of previous states $\mathcal{X}_t$ and a current observation $\mathbf{Z}_t$. As shown in Equation 3.35, the problem of finding a precise posterior probability for such previous states and current observations becomes tractable by introducing an approximating observation likelihood function $w(\mathbf{p}_t, \mathbf{Z}_t) \propto P(\mathbf{Z}_t|\mathbf{p}_t)$.

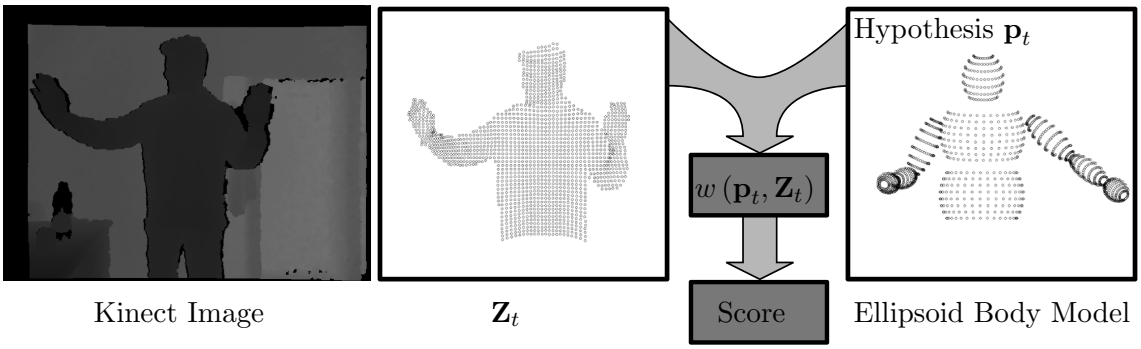


Figure 4.2: General outline of the likelihood approximation process.

The general process flow for computing the proposed likelihood approximation function is illustrated in Figure 4.2. Using depth data coming from a Kinect camera, the point cloud of the user is extracted. For an arbitrary pose hypothesis $\mathbf{p}$[2], a point cloud is computed using the approach proposed in Section 4.2.2. An approximation function then computes a likelihood score as described in Section 4.2.3 below.

4.2.2 Deformable Body Model

The observation likelihood can be approximated by comparing a simplified body model of the user with the actual observations provided by the camera system. As stated above, the approach presented here does not rely on re-projections into the 2D image plane, but instead considers the point clouds in a 3D coordinate system. The weighting function uses ellipsoid geometric primitives arranged in a hierarchical skeleton structure to generate a

[2]In this context, this pose hypothesis is a single particle of the current APF iteration.

corresponding 3D model of the human body. The single ellipsoids connect to each other by joints with varying degrees of freedom. As an hinge joint, an elbow or knee possesses only one DoF. More complex joints such as shoulder or hip are approximated using three DoF. Since ellipsoids are simply the 3-dimensional equivalent of an ellipse, they are very easy to generate and manipulate. For the remainder of the chapter, ellipsoids are denoted as e and the set of ellipsoids making up a skeleton is given as $\mathcal{E}$.

The surface of each ellipsoid is populated by equally spaced reference points $\mathbf{r}$. The set of reference points $\mathcal{R}$ contains all surface points of a given hypothesis. Each pose hypothesis specifies a certain arrangement of limbs, encoded into a vector $\mathbf{p}_t^{(n)}$. The indices of the vector supply both the step index t and the hypothesis index $n = \{1, \ldots, N_p\}$ in the current particle set $\mathcal{S}_t$. The body model uses the hypothesis to translate and rotate its limbs into the requested configuration. Together with the skeleton, the ellipsoids and their surface points also move into specific positions. The pose vector $\mathbf{p}_t^{(n)}$ encodes rotational joints as *quaternions*. The position of the root node[3] of the skeleton is defined relative to the coordinate system origin. Starting from the root node, translations and rotations of limbs and reference points are applied by traversing the skeleton hierarchy, setting each limb relative to the previous element.

Once the ellipsoid body elements are arranged to match a pose $\mathbf{p}_t^{(n)}$, the visibility of each reference point $\mathbf{r} \in \mathcal{R}\left(\mathbf{p}_t^{(n)}\right)$ to the camera system is calculated. Simultaneously, the algorithm detects collisions between single ellipsoids e_i by using the reference points for sampling intersection with other ellipsoids $e_j \in \mathcal{E}$.

The following process is repeated for all cameras observing the scene. The reference point $\mathbf{r}_i$ belongs to the ellipsoid e_j and is given in the observing camera coordinate system. At first, the normalised direction from the camera to the reference point is calculated as $\mathbf{u}_{norm}$, using the relative translation $\mathbf{t}_e$, scaling k_e and rotation $\mathbf{q}_e$ of the ellipsoid. The following process calculates the line of sight $\mathbf{u}_{line}$ from the camera to the reference point:

$$\forall \mathbf{r}_i \in \mathcal{R}$$
$$\forall e_j \in \mathcal{E}$$
$$\mathbf{u} = -\mathbf{r}_i \tag{4.1}$$
$$\mathbf{u}_{norm} = \frac{\left(\mathbf{q}_e^{-1}\mathbf{u}\mathbf{q}_e\right)^{\mathrm{T}}}{\|\mathbf{u}\|} k_e \tag{4.2}$$
$$\mathbf{u}_{line} = \left(\mathbf{q}_e^{-1}\left(\mathbf{r}_i - \mathbf{t}_e\right)\mathbf{q}_e\right)^{\mathrm{T}} k_e \tag{4.3}$$

This operation transforms the ellipsoid and the reference points into a unity sphere system for fast distance and intersection calculations . The simplified problem is equivalent to an intersection check between the line $\mathbf{u}_{line}$ and a sphere. Figure 4.3 visualizes a simplified example in a 2D space. In the following equations, the binary flag f_{coll} signals a collision between the reference point and a body element, while f_{occl} signals mere occlusion:

[3]Here the pelvis node acts as the root node.

$$d = \left(\mathbf{u}_{norm}^{\mathrm{T}}\mathbf{u}_{line}\right)^2 - \mathbf{u}_{line}^{\mathrm{T}}\mathbf{u}_{line} - 1 \tag{4.4}$$

$$f_{\mathrm{occl}} = \begin{cases} 1, & \text{if } d > 0 \cap -\mathbf{u}_{norm}^{\mathrm{T}}\mathbf{u}_{line} + \sqrt{d} > 0 \\ 0, & \text{else} \end{cases} \tag{4.5}$$

$$f_{\mathrm{coll}} = \begin{cases} 1, & \text{if } \sqrt{\mathbf{u}_{line}^{\mathrm{T}}\mathbf{u}_{line}} < 1 \cap e_j \notin \mathcal{E}_{\mathrm{Neighbors}} \\ 0, & \text{else} \end{cases} \tag{4.6}$$

Performing both checks on f_{occl} and f_{coll} using the same unity sphere model achieves significant reductions in computational effort, especially compared to polygon-based body models. Polygon-based models require a check between each model point and each polygon for occlusion. A subsequent second pass, again over all polygons, detects intersections. Meanwhile, the ellipsoid model only requires a single check for each body element, greatly reducing computational overhead. In both cases, the occlusion check needs to be repeated for each camera.

A reference point is used in the likelihood approximation only if it is visible to at least one camera. This leads to the set of observable reference points $\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)$ for a given pose hypothesis.

While the body model must provide a reference point cloud for comparing hypothesis and observation, the model should also provide a plausibility check on the pose itself. For example, if a hypothesis proposes an over-stretched elbow, this unnatural pose should result in a very low likelihood score. In order to realize such constraints on quaternion joint rotation, the limb positions are restricted on a conic subspace. The spherical joint approach suggested by Wilhelms *et al.* [WG01] provides a convenient framework for such restrictions. During the initial model generation, their algorithm sets the permissible limb rotation cone and twist for each joint. The poses of each joint are then efficiently tested against these limits during run-time. Joints exceeding the limits are adjusted to remain within the boundaries. The change is then reflected in an adjustment to the original hypothesis $\mathbf{p}_t^{(n)}$. This update ensures that the hypothesis populating the solver are all plausible and legal solutions.

4.2.3 Likelihood Approximation Function

The approximation function presented here computes the observation likelihood by finding nearest-neighbour pairs between the non-occluded 3D model points $\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)$ computed for a pose $\mathbf{p}_t^{(n)}$ and the 3D data points $\mathcal{G}\left(\mathbf{Z}_t\right)$ extracted from the observed scene $\mathbf{Z}_t$. When the observed pose and the sampled pose are very similar, we can expect points of $\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)$ to be very close to the points of $\mathcal{G}\left(\mathbf{Z}_t\right)$.

At first glance it may seem sufficient to simply find the closest model point for each data point and consider the mean distance. However, this approach ignores all regions of model points which have no nearby data point. Such isolated points indicate a badly fitting hypothesis and should therefore lead to low scores. On the other hand, the same problem arises if the scoring uses only the closest data point for each model point. This

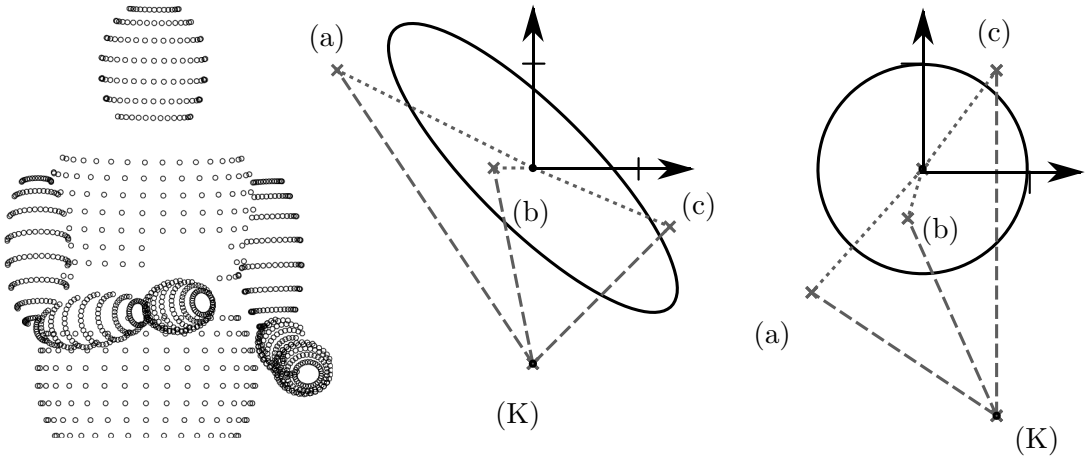


Figure 4.3: (Left) View of a dense version of the ellipsoid upper body body model. Only unoccluded points rendered. (Middle) 3 Points relative to an ellipse: (a) is visible to (K), (b) collides with the ellipse, (c) is occluded by the ellipse. (Right) Same situation after transformation to unity sphere system.

scoring would ignore data. Consequently, both distances between the closest data point for each model point and, vice versa, distances between the closest model point for each data point are relevant to the likelihood approximation $w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$. The approximation uses exponential functions on the minimum distances between the model points $\mathbf{r}$ and the observation points $\mathbf{g}$. The choice of exponential functions stems from a series of experiments shown later in Section 4.4.2.1. The constants k_1 and k_2 control the steepness of the approximated likelihood function (typically $k_1 = 20$, $k_2 = 2$). In the following equations, the variables $N_\mathbf{g}$ and N_e denote the number of visible points per data or model cluster. As these numbers differ between clusters, the equations use indices for attribution:

$$w_{\mathrm{r2o}}^{\mathrm{basic}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \prod_{e=1}^{N_e} \frac{1}{N_\mathbf{r}^{(e)}} \sum_{\mathbf{r} \in \mathcal{R}^{(e)}} \exp\left(-k_1 \mathrm{min}_\mathbf{g} \|\mathbf{r} - \mathbf{g}\|\right) \tag{4.7}$$

$$w_{\mathrm{r2o}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \exp\left(k_2\left(1.0 - w_{\mathrm{r2o}}^{\mathrm{basic}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right)\right)\right) \tag{4.8}$$

$$w_{\mathrm{o2r}}^{\mathrm{basic}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \prod_{k=1}^{K} \frac{1}{N_\mathbf{g}^{(k)}} \sum_{\mathbf{g} \in \mathcal{G}^{(k)}} \exp\left(-k_1 \mathrm{min}_\mathbf{r} \|\mathbf{r} - \mathbf{g}\|\right) \tag{4.9}$$

$$w_{\mathrm{o2r}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \exp\left(k_2\left(1.0 - w_{\mathrm{o2r}}^{\mathrm{basic}}\left(\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right)\right)\right) \tag{4.10}$$

$$w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right) = w_{\mathrm{r2o}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) \times w_{\mathrm{o2r}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) \tag{4.11}$$

In order to increase the impact of local discrepancies, Equations 4.7 and 4.9 employ segmentation of the point clouds. High minimum distances in small but significant regions, such as hands, thus lead to low likelihood values for the whole pose. The segmentation of the model points for function $w_{\mathrm{r2o}}^{\mathrm{basic}}(\ldots)$ is performed either by the number N_e of body elements making up the model or by k-means clustering. Section 4.4.2.3 provides an evaluation of both approaches. When grouping by body element, segments with less than 3 visible points receive a generic partial score of 1. This assignment allows the weighting function to treat unobservable limbs as weighting neutral — the importance of this choice will be shown experimentally in Section 4.4.2.6. The algorithm uses k-means clustering on the observed data points in function $w_{\mathrm{o2r}}^{\mathrm{basic}}(\ldots)$, since no previous mapping to body parts is known at this stage. Section 4.4.2.2 contains a thorough discussion on the effect of the various variables. The design of the chosen likelihood approximation function is derived in Section 4.4.2.1 from experimental data.

Self-collisions between limbs are penalised by lowering the likelihood approximation score. The number N_C of self-collisions observed when evaluating Equation 4.6 serves as measure for this penalty. In case the count of self-collisions remains below $N_{\mathrm{Threshold}}$, no penalty is applied. If there are more, the final score is lowered by the following formula:

$$w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)_{\mathrm{Final}} = \begin{cases} w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right) & N_C < N_{\mathrm{Threshold}} \\ \dfrac{w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)}{N_C - N_{\mathrm{Threshold}}} & N_C \geq N_{\mathrm{Threshold}} \end{cases} \tag{4.12}$$

Figure 4.4 introduces an example on how accuracy is improved by segmentation. Shown is a simplified example with an observed upper body and a score of $w_s = 1$ for sufficiently matched points. The torso and the right arm fit the observation perfectly, generating a score of 1 for each matched model point. The arm on the left is posed incorrectly, thus gaining a score of $w_s = 0$ for each associated reference point. Using no segmentation, the score would be

$$\frac{1}{N_{\mathbf{g}}} \sum_{\mathbf{r} \in \mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)} w_s = \frac{280}{340} = 0.823 \ ,$$

with w_s representing the simplified model point score. Although the pose does not fit the observation very well, it would still get a high observational likelihood. A segmentation by body element, similar to Equation 4.7, first computes the scores for each single limb and then multiplies a partial term to determine the final score. For our example, there are several limbs which receive a partial score of zero. In consequence, the entire score product returns a likelihood of zero:

$$\prod_{e=1}^{\mathcal{E}} \frac{1}{N_{\mathbf{g}}} \sum_{\mathbf{r} \in \mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)} w_s = 0.0 \ . \tag{4.13}$$

The segmented approximation thus prevents the marginalisation of more salient regions — the arm — by larger, less salient regions — the torso.

Since the assignment of data points to the real body elements is not known at this point, k-means clustering can be used to segment point clouds recorded by the depth camera. As Figure 4.4 shows, significant regions such as outstretched hands are normally assigned to a separate cluster. Thus the segmentation supports likelihood approximation not only for the reference points $\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}_t^{(n)}\right)$ but also for the data points $\mathcal{G}\left(\mathbf{Z}_t\right)$.

It is important to note that this approximation $P\left(\mathbf{Z}_t|\mathbf{p}_t^{(n)}\right) \propto w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)^{\text{Final}}$ works without colour or texture cues, making this approach suitable for all scenarios where only pure depth data is available. Since the likelihood approximation uses point cloud data as observation input, the integration of multiple depth cameras is straightforward and requires no modification of the algorithm.

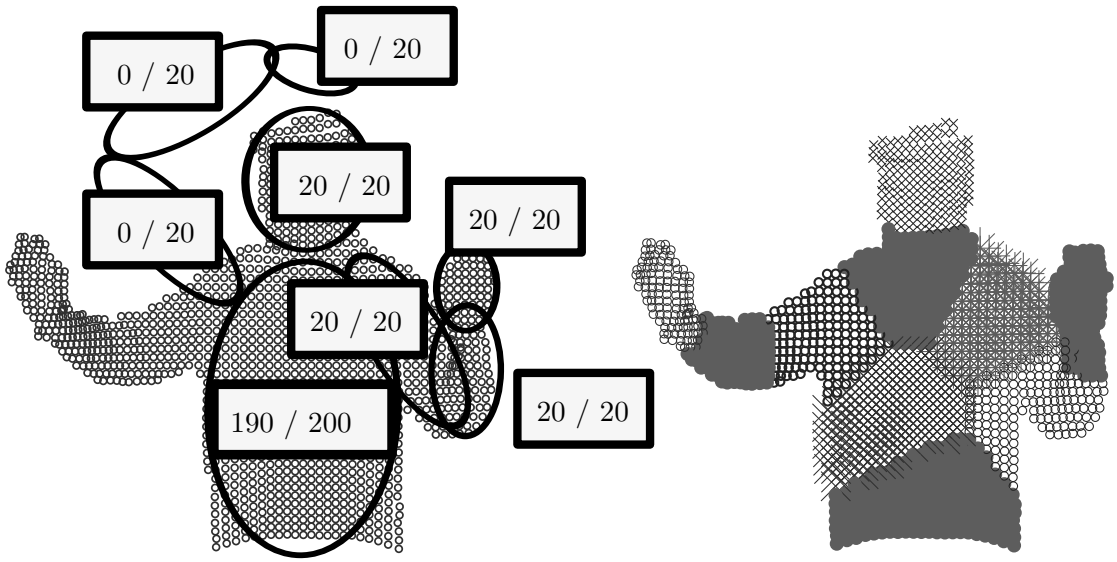


Figure 4.4: (Left) Simplified illustration of a segmented likelihood approximation. The torso is fitted correctly by the ellipsoids, while the arm hypothesis does not fit the data. Matched model points return 1, otherwise 0. Detailed discussion in Section 4.4.2.3. (Right) View of a k-means segmented data point cloud using $k = 10$. Different shadings denote the cluster assignment of the surface points.

4.2.4 Implementation of the Likelihood Approximation

Using the likelihood approximation function described in the previous section, the stochastic solver can proceed to calculate the approximate likelihood $w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$ of $P\left(\mathbf{Z}_t|\mathbf{p}_t^{(n)}\right)$. The following steps are performed for the likelihood approximation of a single pose hypothesis $\mathbf{p}_t^{(n)}$ using exhaustive minimum distance search:

1. The data point cloud is computed from depth images captured by the cameras. Smoothing and sampling clean noise from the data and optionally reduce the data set. This process yields $N_{\mathbf{g}} = \|\mathcal{G}\left(\mathbf{Z}_t\right)\|$ data points.

2. K-means clustering segments the data points into local groups.
3. The body model is used to create a point cloud based on the pose hypothesis $\mathbf{p}_t^{(n)}$ resulting in $N_\mathbf{r} = \|\mathcal{R}_\mathrm{v}\left(\mathbf{p}_t^{(n)}\right)\|$ model points. The reference points can be segmented either by limb assignment, i.e. from which ellipsoid they were generated, or by k-means clustering.
4. For each point in the data set, we calculate the distance to each point in the reference set. This leads to a $N_\mathbf{g} \times N_\mathbf{r}$ distance matrix.
5. The closest data point $\mathbf{g}$ to each reference point $\mathbf{r}$ (i.e. $\min_\mathbf{r}\|\mathbf{r}-\mathbf{g}\|$) is found.
6. We determine the closest reference point $\mathbf{r}$ to each data point $\mathbf{g}$ (i.e. $\min_\mathbf{g}\|\mathbf{r}-\mathbf{g}\|$).
7. The segmented scores $w_\mathrm{r2o}(\ldots)$ and $w_\mathrm{o2r}(\ldots)$ are computed from Equations 4.9 and 4.7. Equation 4.12 yields the final approximation score $w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$.
8. In case of integration into an APF: Adjust weight with temperature exponent k_ann $w_{t,m}^{(n)} = w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)^{k_\mathrm{ann}}$

As sequential stochastic solvers often require hundreds of hypothesis evaluations for a single frame of sensor data, the parallel computing capabilities of modern graphic processing units (GPUs) greatly reduce computation time. Especially the body model and the distance calculations are perfect candidates for GPU based processing, as they apply the same basic set of operations on a large set of data while requiring only limited memory access[4]. Each pose hypothesis can be evaluated by itself without any need for communication between the threads evaluating different poses. An own article published in the International Journal of Computer Vision [LKR13] provides a more detailed discussion of implementation issues.

4.3 Integration into an Annealing Particle Filter

So far, the previous Section 4.2 showed a practical way of approximating the observation likelihood function $P\left(\mathbf{p}_t^{(n)}|\mathbf{Z}_t, \mathcal{X}_t\right) \propto w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$. By itself, the approximation could be included in any stochastic solver framework. This section will show how it is embedded into the greater APF framework and what additional steps are necessary for performing human pose tracking.

Human pose tracking requires three components: A source of data, such as the camera system, a tracking framework, in this case a stochastic approach, and finally a way to output the data and make it available to other processes, e.g. a gesture recognition framework. The data source in this scenario is an array of one or more RGB-D cameras from which the point clouds are extracted and projected into a common coordinate system.

[4]In literature, such problems are frequently classified as "embarrassingly parallel", as they pose no challenge to a skilled programmer. I was able to build a real-time capable implementation with virtually no previous experience in parallel programming, proving the aptness of the term.

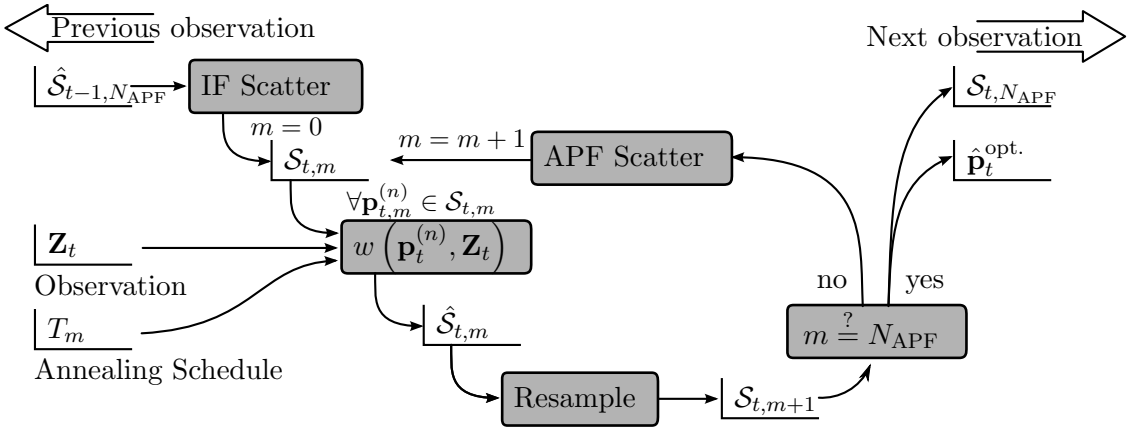


Figure 4.5: Overall workflow for the APF process.

Section 2.5.1 explained previously the setup used for this dissertation in detail. At this point it is sufficient to remember that the output of the camera system consists of the fused point cloud $\mathcal{G}_t$ and the camera calibration data. On the other side of the process, virtual reality peripheral network (VRPN) protocol [Tay+01] broadcasts the output from the human pose tracking to the rest of the system. This protocol allows for convenient and reliable integration into a range of existing AR and VR systems.

As described in Section 3.4, the basic APF algorithm consists of 3 essential steps: Scattering, weighting and resampling. Figure 4.5 illustrates the flow of data between these processes. The observation likelihood function performs the weighting step, leaving the implementation of scattering and resampling for a more thorough description.

4.3.1 Resampling of Particles

The weighted particle set $\hat{\mathcal{S}}_{t,m}$ at timestep t, annealing step m, consists of pairs of pose hypotheses $\mathbf{p}_{t,m}^{(n)}$ and their associated weight $w_{t,m}^{(n)}$:

$$\hat{\mathcal{S}}_{t,m} = \left\{ \left(\mathbf{p}_{t,m}^{(1)}, w_{t,m}^{(1)}\right) \dots \left(\mathbf{p}_{t,m}^{(N_p)}, w_{t,m}^{(N_p)}\right) \right\} \tag{4.14}$$

.

The particle $\mathbf{r}_{t,m}^{(n)}$ itself contains the encoded joint angles and basic transformations, while the weight, $w_{t,m}^{(n)}$, is computed from the observation likelihood approximation function $w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$. Stochastic universal sampling (SUS) [Bak87] performs the resampling of the particle sets. SUS produces a new set of particles from a weighted set with low bias towards successful particles. Its basic operation is given in Algorithm 9. The likelihood of individual particles appearing in the new set is directly proportional to their weight in the old set. In the context of human pose tracking, this rule translates to poses with higher observation likelihoods having a higher chance of being passed from iteration to iteration. Consequently the particle set accumulates pose hypothesis which fit the observation, while bad hypotheses are discarded gradually.

In order to avoid premature convergence on local extrema, the set should always retain a few more unlikely poses. These "weak" hypotheses act as seeds for further exploration as new observations come in, enabling the tracker to pursue multiple possible solutions simultaneously even as the particle set converges. The advantage of SUS over other methods such as roulette-wheel selection (RWS) is the low bias, which ensures that the particle set is not entirely saturated with highly probably pose hypotheses.

Algorithm 9 Stochastic universal sampling (SUS) [Bak87], drawing N_{samples} samples from a weighted particle set

Normalize weights such that $\sum w^{(n)} = 1$
Build cumulative weights $w^{(n)} = w^{(n)} + w^{(n-1)}$, $w^{(N_{\text{samples}})} = 1$
Build random first pointer $p_0 = r \,/\, N_{\text{samples}}$ with $r = [0, 1]$
Set selection index $i = 1$
for N_{samples} **do**
  Advance pointer $p_n = p_{n-1} + 1 \,/\, N_{\text{samples}}$
  while $w^{(i)} \geq p_n$ **do**
    Increment selection index $i = i + 1$
  end while
  Add $\mathbf{p}^{(i)}$ to new particle set $\mathcal{S}_{\text{new}}$
end for

4.3.2 Scattering the New Particle Sets

The SUS method described in the previous section resamples the particle set between each iteration of the annealing process and for each new frame of observations. After the new particle set is selected, scattering the decision variables in the hypotheses encourages a thorough exploration of the available solution space.

A generic APF tracker employs two different modes of scattering for annealing iterations and new observation frames.

Adding noise $\mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}})$ to the decision variables scatters the newly sampled set $\hat{\mathcal{S}}_{t,m}$ between the iterations of the annealing process. The noise $\mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}})$ is covariant with the parameters in the particle set. This scattering leads to the new unweighted set $\mathcal{S}_{t,m+1}$:

$$\forall \mathbf{p}_{t,m}^{(n)} \in \hat{\mathcal{S}}_{t,m} : \quad \mathbf{p}_{t,m+1}^{(n)} = \mathbf{p}_{t,m}^{(n)} + k_{\text{ann}} \mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}}) \tag{4.15}$$

As the amount of noise is correlated to the distribution of pose variables in the particle set, this approach acts as a soft partitioning of the solution space. Joint poses on which large parts of the particle set already agree are not altered greatly. On the other hand, the scattering applies more changes to joints for which there are still many diverging hypotheses remaining. Since hypotheses on these joints have not converged yet, the algorithm probes their solution subspace aggressively. So for each iteration stage, some joints of the pose space may have converged already, while other joints are still being explored actively.

For transitions between observation frames, the scattering also replaces the unknown motion model $P\left(\mathbf{p}_t | \mathbf{p}_{t-1}\right)$. The scattering acts as an implicit motion model by applying

specialised scatter patterns to different parts of a pose. Thus, the particle filter structure itself generates possible new limb configurations on the assumption of Gaussian motion distributions. This process was previously described in Equation 3.41. Since a new observation is available, the particle set should be scattered thoroughly from its current, converged state. Applying bounded white noise $\mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$ to the previous set $\hat{\mathcal{S}}_{t,N_{\text{APF}}}$ generates the new particle set $\mathcal{S}_{t+1,0}$:

$$\forall \mathbf{p}^{(n)}_{t,N_{\text{APF}}} \in \hat{\mathcal{S}}_{t,N_{\text{APF}}} : \quad \mathbf{p}^{(n)}_{t+1,0} = \mathbf{p}^{(n)}_{t,N_{\text{APF}}} + k_{\text{IF}}\mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}}) \tag{4.16}$$

For the new particles, the algorithm performs a plausibility check over all new joint limits to ensure that this method generates only valid hypotheses.

Besides these two standard modes of scattering, the following specialised techniques create smaller subsets of the particles for more efficient exploration. These methods specifically support certain aspects of human pose tracking. In practice, each of these methods performs its own resampling of the old particle set. For example, between two iterations of the annealing process, three new particle sets might be sampled: The combined new set is expected to contain N_p particles. One subset has $N_{\text{Set 1}} = 0.8N_p$ particles, the next $N_{\text{Set 2}} = 0.1N_p$ and the third $N_{\text{Set 3}} = 0.1N_p$. The first subset is then scattered as described above, the second subset has just random noise added and the third subset is populated with commonly observed poses. Subsequently, these three subsets are merged into a single particle set for the next iteration. The following paragraphs describe the methods used in this dissertation.

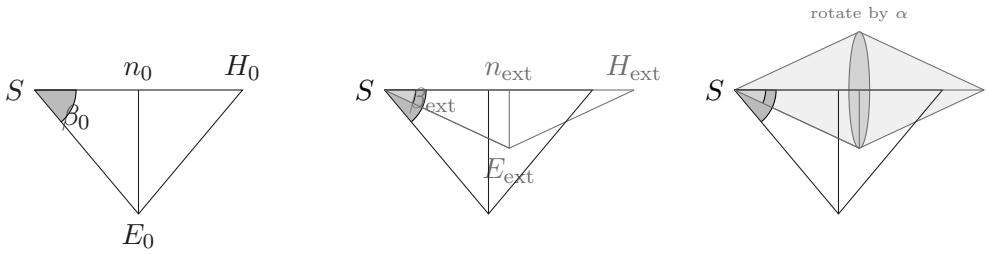
Algorithm 10 Resampling and scattering between timesteps

$S_{t+1,\text{norm}} = \texttt{SUS}(S_{t,M}) + \mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$
$S_{t+1,\text{inverse}} = \texttt{InverseKinematics}(s^{\text{optimal}}_{t,M}) + 0.1\,\mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$
$S_{t+1,\text{static}} = \texttt{StaticPoses}(S_{t,M}) + 0.1\,\mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$
$S_{t+1,0} = [S_{t+1,\text{norm}}, S_{t+1,\text{inverse}}, S_{t+1,\text{static}}]$

Algorithm 11 Resampling and scattering between annealing steps

$S_{t+1,\text{norm}} = \texttt{SUS}(S_{t,M}) + \mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}})$
$S_{t+1,\text{crossover}} = \texttt{Crossover}(S_{t,M}) + \mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}})$
$S_{t+1,\text{random}} = \texttt{SUS}(S_{t,M}) + \mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$
$S_{t+1,0} = [S_{t+1,\text{norm}}, S_{t+1,\text{crossover}}, S_{t+1,\text{random}}]$

Algorithm 12 Calculation of shoulder angles after elbow extension and rotation by α, all vectors in shoulder reference system



$$\mathbf{a} = \mathbf{v}_{\text{Elbow}} - \mathbf{v}_{\text{Shoulder}}$$
$$\mathbf{b} = \mathbf{v}_{\text{Hand}} - \mathbf{v}_{\text{Elbow}}$$
$$\mathbf{c} = \mathbf{v}_{\text{Hand}} - \mathbf{v}_{\text{Shoulder}}$$
$$\beta = \arccos\left(\frac{|\mathbf{a}|^2 + |\mathbf{c}|^2 - |\mathbf{b}|^2}{2\ |\mathbf{a}|\ |\mathbf{c}|}\right)$$
$$\mathbf{m} = |\mathbf{a}|\cos(\beta)\frac{\mathbf{c}}{|\mathbf{c}|}$$
$$\mathbf{n}_0 = \mathbf{a} - \mathbf{m}$$
$$\beta_{\text{ext}} = 0.6\,\beta$$
$$\mathbf{m}_{\text{ext}} = |\mathbf{a}|\cos(\beta_{\text{ext}})\frac{\mathbf{c}}{|\mathbf{c}|}$$
$$\mathbf{n}_{\text{ext}} = |\mathbf{a}|\sin(\beta_{\text{ext}})\frac{\mathbf{n}_0}{|\mathbf{n}_0|}$$
$$\mathbf{a}_{\text{ext}} = \mathbf{n}_{\text{ext}} + \mathbf{m}_{\text{ext}}$$
$$\mathbf{c}_{\text{ext}} = \sqrt{|\mathbf{b}|^2 - |\mathbf{n}_{\text{ext}}|^2}\,\frac{\mathbf{c}}{|\mathbf{c}|}$$
$$\mathbf{b}_{\text{ext}} = \mathbf{c}_{\text{ext}} + \mathbf{a}_{\text{ext}}$$
$$\gamma_{\text{ext}} = \arccos\left(\frac{|\mathbf{a}|^2 + |\mathbf{b}|^2 - |\mathbf{c}_{\text{ext}}|^2}{2\ |\mathbf{a}|\ |\mathbf{b}|}\right)$$
$$\phi_{\text{Elbow}} = \gamma_{\text{ext}} - \pi$$
$$\mathbf{q}_0 = \texttt{QuaternionFromAxisAngle}(\mathbf{c}_{\text{ext}},\ \alpha)$$
$$\mathbf{n} = \mathbf{q}_0\,\mathbf{n}_{\text{ext}}\,\mathbf{q}_0^{-1}$$
$$\mathbf{h}_1 = (0, 0, 1)^{\text{T}}$$
$$\mathbf{h}_2 = \frac{\mathbf{a}_{\text{ext}}}{|\mathbf{a}_{\text{ext}}|}$$
$$\delta_1 = \texttt{VecOnVecRoundAxis}(\mathbf{h}_1,\ \mathbf{h}_2,\ \mathbf{h}_1 \times \mathbf{h}_2)$$
$$\mathbf{q}_1 = \texttt{QuaternionFromAxisAngle}(\mathbf{h}_1 \times \mathbf{h}_2,\ \delta_1)$$
$$\mathbf{q}_2 = \texttt{QuaternionFromEuler}(\phi_{\text{Elbow}},\ 0,\ 0)$$
$$\mathbf{q}_3 = \mathbf{q}_1\,\mathbf{q}_2$$
$$\mathbf{h}_3 = \mathbf{q}_3\,\mathbf{h}_1\,\mathbf{q}_3^{-1}$$
$$\delta_2 = \texttt{VecOnVecRoundAxis}(\mathbf{h}_3,\ \mathbf{b},\ \mathbf{a}_{\text{ext}})$$
$$\mathbf{q}_4 = \texttt{QuaternionFromEuler}(0,\ 0,\ \delta_2)$$
$$\mathbf{q}_5 = \mathbf{q}_1\,\mathbf{q}_4$$
$$(\phi_{\text{S, x}},\ \phi_{\text{S, y}},\ \phi_{\text{S, z}}) = \texttt{QuaternionToEuler}(\mathbf{q}_5)$$

Crossover Permutation: Like the SUS process, the crossover operator is an element borrowed from *genetic algorithms* and popularised by Deutscher *et al.* [DR05]. Thinking of a particle filter in the terms of an evolutionary process, the scattering would correspond to mutation and the weighting to the fitness calculation. The crossover operator can then be thought of as the mating between two individuals of the population: Two individuals are selected with regard to their fitness, interchange elements of their DNA (which would be the particle information) and generate two new individuals. Scattering then mutates these new individuals. The procedure consists of five distinct steps:

1. Select $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ by SUS from $\hat{\mathcal{S}}$
2. Randomly select two indices $i < j \in [1, \ldots, N_{\text{DoF}}]$
3. Generate $\mathbf{p}^{(1)} = \left(p_0^{(1)}, \ldots, p_{i-1}^{(1)}, p_i^{(2)}, \ldots, p_{j-1}^{(2)}, p_j^{(1)}, \ldots, p_{N_{\text{DoF}}}^{(1)}\right)$
4. Generate $\mathbf{p}^{(2)} = \left(p_0^{(2)}, \ldots, p_{i-1}^{(2)}, p_i^{(1)}, \ldots, p_{j-1}^{(1)}, p_j^{(2)}, \ldots, p_{N_{\text{DoF}}}^{(2)}\right)$
5. Apply regular scattering by adding $\mathcal{N}(\mathbf{0}, \Sigma_{\text{APF}})$ to both new particles

Inverse Kinematics: The regular APF approach is well suited for fitting complex functions with independent variables. However, the human body consists of a number of kinematic chains, which lead to interdependencies between variables. As an example, the rotation of the shoulder determines the position of the following elbow and hand joints. In many tracking situations, pose changes are small and limited to the arms for most observations. So for example, if only the hands moved since the last observation, random changes to joint orientations throughout the entire body are supremely inefficient. Instead an inverted kinematic chain can be used to inject a number of specifically modified particles into the resampling step, exploring alternative poses of the arms.

Usually, such approaches rely on individual tracking of the hands as anchor points for the inverse kinematics[5]. However, in the absence of a dedicated body part detection, we can instead utilize the wrist position from the previous pose estimate. While this is less accurate it eliminates the need for dedicated hand detectors and consequently speeds up processing. The last 'optimal' estimate serves as the starting point for this scattering step. The elbow is rotated out of its current 'optimal' position and the resulting shoulder and elbow angles $\phi_{\text{S, x}}$, $\phi_{\text{S, y}}$, $\phi_{\text{S, z}}$, ϕ_{Elbow} are used to build a new particle (see Algorithm 12). Without anchor points, such as separately tracked hands, the APF can get stuck in high elbow angles. Such failures are caused by high weighting scores which are returned when the lower arm gets mapped onto the observed point cloud of the actual upper arm. To counteract this behaviour, the inverse kinematic pose generator extends the arms slightly for the new hypotheses $\mathbf{p}_{t+1,0}^{(n)}$.

Static Particles: Static particles consist of pre-recorded generic poses, aligned to the last stable pose estimate. For the upper body, such basic poses might consist of arms forward, arms outstretched to the sides or hanging down in all possible permutations. These typical poses are crucial for initialisation of the tracking and accelerate recovery

[5] A good example is found in the work done by Azad *et al.* [AAD08]

after tracking errors. For instance, the particle filter might follow a wrong pose hypothesis and converge on a bad pose estimate. However, when the user assumes one of the pre-recorded poses, the associated particle is highly likely to return high weight scores, thus populating the particle set and crowding the incorrect hypotheses out.

Randomising: To counteract premature convergence on local optima, about 10% of randomised particles are inserted. These are pose hypotheses drawn by SUS from the previous population and then scattered by applying strong random noise $\mathcal{N}(\mathbf{0}, \Sigma_{\mathrm{IF}})$ to the joint angles (while retaining the root node translation). This addition of noise allows for a more thorough exploration of the configuration space even as the regular particle set is converging. The insertion of randomised particles also helps in recovering the particle set from failed tracking.

4.3.3 Calculating the Current Result

At the end of each annealing loop, the majority of hypotheses in the last particle set should have converged on a small region of the solution space. However, there are most likely a number of hypotheses with smaller observation likelihood present as well. The simplest approach would force convergence for the entire particle set, e.g. by modifying particle survival rates between iterations. Unfortunately, such forced convergence would impact the APF's capability of pursuing several pose hypotheses at once. This ability is crucial in dealing with ambiguous poses, where the observation allows multiple interpretations. Therefore, only the highest scoring particles inform the final pose estimate, instead of forcing the entire particle set to converge. In practice, using the top 10% scoring particles yields good results. Within this subset $\hat{\mathcal{S}}^{\mathrm{Top10}}_{t,N_{\mathrm{APF}}}$ the single decision variables are simply averaged. In the case of joint rotations, Markley *et al.*'s [Mar+07] eigenvalue approach calculates the mean quaternions. The average over the 3D translation values yields the mean root node translation.

A VRPN server then broadcasts the resulting pose estimate $\hat{\mathbf{p}}_t$ for use by other elements of the AR videoconferencing system.

4.3.4 Performance using CUDA

In order to evaluate the potential for real-time tracking applications, the proposed observation likelihood function was implemented in compute unified device architecture (CUDA). The APF framework described in the previous section used this implementation to perform human pose tracking. In a lab setting, a single Kinect camera gathered sequences of a user sitting at a table, recorded with 30 fps. The tracking algorithm evaluated 300 hypothesis (i.e. particles) with 288 surface points each against an average of 550 observed points in less than 25 ms on a 2.66 GHz Intel Core2Quad CPU with 4 GB RAM and a NVIDIA Geforce GTX 275 graphics card. Please note that this implementation used brute-force to find nearest neighbours. More efficient implementations based on octrees or kd-trees for neighbourhood search should enable even faster evaluation [Cay11].

The tracker itself performed well, tracking hand and arm movements of varying complexity, as shown in Figure 4.6. Using a full body model in combination with a stereo camera also yields promising results, as shown in Figure 4.7. Note that these sequences

were tested without any additional body part recognition, i.e. informed only by the unlabelled point cloud data. The results underscore the potential of the point cloud driven APF approach for real-time performance.

Figure 4.6: Scenes from upper body tracking. The skeleton was redrawn for better display in monochrome print.

Figure 4.7: Scene from full body tracking. The skeleton was redrawn for better display in monochrome print.

4.4 Evaluation of the Human Pose Tracking

This section presents experiments evaluating the overall APF human pose tracking system and the observation likelihood approximation. The experiments with the APF tracker consider a scenario with a full human body model. The influence of the resampling and scattering parameters are of special interest in these tests, since these are critical to the implicit user motion model. Further experiments quantify the effects of varying the particle

number and annealing steps on tracking performance. These parameters are important to the computational overhead of the approach.

The proposed observation likelihood approximation function is generally applicable to other stochastic tracking approaches, such as the Sigal *et al.*'s graphical model approach [Sig+04]. A series of experiments explains the rationale behind the function design and shows the effect of different parameter settings on approximation quality.

The following section will therefore start with an examination of the entire tracking system on a number of different motion sequences before turning to the evaluation of the weighting function.

4.4.1 Evaluation of APF Parameter Settings

The first goal is an examination of the impact of single parameters on the overall tracker performance. Hence, we can decide how many particles are needed for robust tracking, how many annealing steps are appropriate etc. A labelled dataset serves as a reference for the quantitative analysis of the impact of parametrisation on the APF pose tracking. These sequences are repeatedly presented to the pose tracking algorithm described in Section 4.3 for systematically varied parameter settings.

The image sets used for testing cover a wide range of possible interaction scenarios. The recorded sequences start with very simple gestures, like lifting an arm or waving, and then cover a range of increasingly complex poses. Altogether, the reference set contains twenty different movement sequences from two persons. Both users repeat each motion sequence four times. All sequences show the person standing, initially facing a single camera in a neutral pose and then performing a gesture or assuming a pose. The following poses and gestures are used:

1. Raise left arm up towards shoulder height, pointing towards camera
2. Wave with left hand
3. Wave with left hand, right hand rests on hip
4. Place both hands before chest
5. Lift right leg towards camera, bend knee
6. Lift an imaginary box from the left to the right
7. Lift an imaginary box from the left to the right and back
8. Lift an imaginary box from the left (shoulder height) and drag it in front of torso
9. Both hands grab an imaginary box in front of the person, drag it towards the chest
10. Both hands grab an imaginary box in front of the person, drag it towards the chest and turn it over
11. Left hand touches head, right hand rests on hip
12. Left hand touches head, right hand rests on hip, slight kick with left foot
13. Take a bow towards the camera

14. Lean back, inspect the ceiling
15. Step to the left and back
16. Balance on left leg, spreading arms
17. Step onto a chair (sideways)
18. Sit down on the edge of the table
19. Duck to the left, rise up again, duck to the right, rise up, duck to the left
20. Lean on the door frame, arm stretched

With four repetitions of each sequence, this resulted in 80 sequences for each user. Lacking a reference pose tracking system, manually placed labels of reference points in the 3D point clouds serve as a ground truth dataset. In every 8th frame of each sequence, the 15 basic joints of the skeleton are labelled by hand. These reference points are stored as 3D coordinates and provide the basis for a relative position error.

Since the position errors between the reference points and the calculated joint positions are based on manual labelling, they can only be used for relative comparisons. It is not possible to state absolute precision, since the actual position of the joints relative to the camera is not known. Therefore, all mean distance errors within a parameter set were normalised to a range from 0.0 to 1.0, with 1.0 corresponding to the highest average error for a single body element. In addition, the average error of hands and elbows in comparison to the rest of the body are recorded. These limb errors are especially interesting for human-computer-interaction, since the arms are central to gesture based interaction. Since the limbs are relatively small in comparison to the rest of the human body, they are also prone to marginalisation effects in the likelihood approximation stage. Therefore, plots show the effect of settings on the relative position error both for the arms and for the remainder of the body.

Table 4.1 gives the default settings of the following experiments. In the individual experiments, only one of these parameters is varied at a time. Together with the relative position errors, we can determine the effect of variations of a single parameter on the position errors of the pose tracker.

4.4.1.1 Number of Particles and Annealing Steps

The expected real-time performance depends greatly on the number of particles and annealing steps: Any reduction in particles and annealing steps translates directly to fewer function evaluations. The number of particles is varied from 200 up to 350. Beyond 350 particles, the memory requirements exceed available GPU memory and performance drops below real-time requirements. Table 4.2 shows the results. The drop in relative position error between 250 and 300 particles is striking, whereas only marginal improvements are seen beyond 300 particles. A particle set for this type of problem should therefore contain about 300 particles, providing a balance between precision and speed.

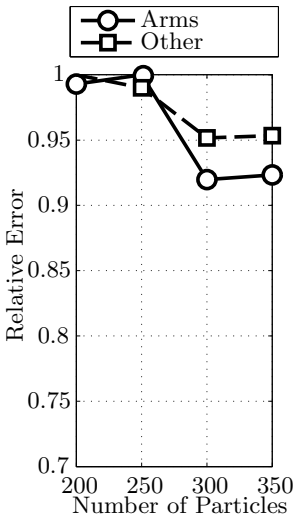
Table 4.3 shows a similar observation for the annealing steps: Increasing the annealing steps to up to 10 steps leads to a marked decrease in the average relative error. Although further improvements are possible when selecting 14 annealing steps, this would be too

Table 4.1: Basic settings for the qualitative and quantitative testing of APF tracker

General			
Number of particles N_p	300	Annealing steps N_{APF}	10
Steepness constant k_1	-20	Steepness constant k_2	-2
Model point clusters	9 (upper body) 15 (full body)	Data point clusters k	10
IF Resampling			
Scattering constant k_{IF}	0.1		
Normal	80%	Static particles	10%
Inverse Kinematics	10%		
APF Resampling			
Scattering constant k_{ann}	0.45		
Crossover	90%	Randomised scattering	10 %

Table 4.2: Relative error based on number of particles used, with 1.0 being the maximum error in each row. Mean error ranges in meters: $\Delta e_{Limbs} = 0.0099$, $\Delta e_{Other} = 0.0054$.

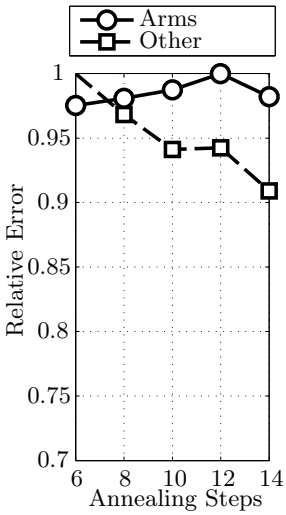
Particle Number	200	250	300	350
Arms				
Right Hand	0.9656	1.0000	0.9063	0.9134
Right Lower Arm	0.9728	1.0000	0.9347	0.9351
Right Upper Arm	0.9833	1.0000	0.9567	0.8955
Left Hand	1.0000	0.9799	0.8810	0.8806
Left Lower Arm	1.0000	0.9567	0.9031	0.9101
Left Upper Arm	1.0000	0.9572	0.9286	0.8967
Other				
Head	1.0000	0.9825	0.9754	0.9450
Upper Body	1.0000	0.9970	0.9828	0.9680
Right Foot	1.0000	0.9667	0.8825	0.9356
Right Lower Leg	0.9811	1.0000	0.8943	0.9284
Right Upper Leg	0.9900	1.0000	0.9595	0.9881
Left Foot	1.0000	0.9662	0.9564	0.9673
Left Lower Leg	1.0000	0.9724	0.9201	0.9195
Left Upper Leg	1.0000	0.9818	0.9619	0.9509
Lower Torso	0.9971	1.0000	0.9795	0.9922



costly in terms of computational time. Ten steps are a reasonable compromise between computational cost and relative position error. It is interesting to note that the relative error for hands and elbows rises with the number of annealing steps. The most likely cause is a pronounced convergence not only for correct poses, but for erroneous poses as well.

Table 4.3: Relative error based on number of annealing steps, with 1.0 being the maximum error in each row. Mean error ranges in meters: $\Delta e_{Limbs} = 0.0202$, $\Delta e_{Other} = 0.0103$.

Annealing Steps	6	8	10	12	14
Arms					
Right Hand	0.8961	0.9109	0.9636	1.0000	0.9797
Right Lower Arm	0.9033	0.9191	0.9712	0.9743	1.0000
Right Upper Arm	0.9639	1.0000	0.9813	0.9682	0.9388
Left Hand	1.0000	0.9973	0.9229	0.9232	0.8730
Left Lower Arm	1.0000	0.9814	0.9266	0.9251	0.8818
Left Upper Arm	1.0000	0.9708	0.9236	0.9282	0.8459
Other					
Head	1.0000	0.9915	0.9355	0.9401	0.8624
Upper Body	1.0000	0.9854	0.9789	0.9707	0.9682
Right Foot	1.0000	0.9825	0.9532	0.9627	0.9529
Right Lower Leg	1.0000	0.9246	0.8751	0.8885	0.8532
Right Upper Leg	1.0000	0.9416	0.9303	0.9379	0.8932
Left Foot	1.0000	0.9587	0.9479	0.9369	0.9538
Left Lower Leg	1.0000	0.9373	0.8772	0.8732	0.8724
Left Upper Leg	1.0000	0.9663	0.9414	0.9436	0.8837
Lower Torso	1.0000	0.9551	0.9374	0.9425	0.8961



4.4.1.2 Influence of the Inverse Kinematics

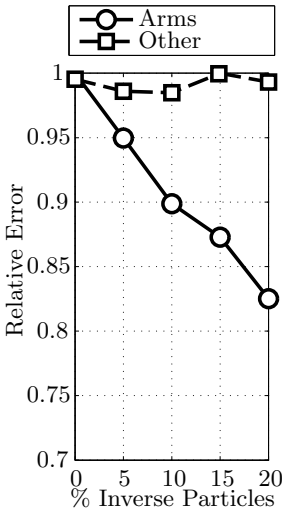
The addition of particles generated by inverse kinematics has a measurable effect on the quality of limb fitting. As shown in Table 4.4, the relative limb error drops significantly with increased injection of modified particles. At first glance, we might suspect that the higher number of stable particles being artificially introduced into the particle set causes the drop. However, the lack of improvement on the rest of the body shows that this is not the case. If the introduction of particles derived from the optimal pose were to affect the tracking of the body in general, it would result in a significant drop for all other body parts as well. As shown in the data, the body tracking remains virtually unaffected by the insertion of modified particles (the mean error varies only by $\Delta e_{Other} = 0.0016$ m). Hence, the modified particles affect only the placement of the arms and lead to an improvement in tracking precision for these.

4.4.1.3 Scattering & Weighting

The scattering coefficients play an important role in the processes of exploration and convergence. Examining the results in Table 4.5 for different inter-frame scattering coefficients k_{IF} with otherwise identical settings shows a marked improvement for lower settings. This effect shows the conservation of a strong majority of particles which were scattered only

Table 4.4: Relative error based on percentage of particles modified by inverse kinematics (0% - 20%), with 1.0 being the maximum error in each row. Mean error ranges in meters: $\Delta e_{Limbs} = 0.0219$, $\Delta e_{Other} = 0.0016$.

Injected Particles	0%	5%	10%	15%	20%
Arms					
Right Hand	1.0000	0.9570	0.8681	0.8130	0.7475
Right Lower Arm	1.0000	0.9292	0.8774	0.8412	0.8068
Right Upper Arm	1.0000	0.9477	0.9576	0.9463	0.9448
Left Hand	1.0000	0.9620	0.9533	0.9761	0.9255
Left Lower Arm	1.0000	0.9588	0.9355	0.9292	0.8960
Left Upper Arm	1.0000	0.9681	0.9667	0.9727	0.9555
Other					
Head	0.9792	0.9755	0.9861	1.0000	0.9951
Upper Body	0.9889	0.9981	0.9927	0.9991	1.0000
Right Foot	0.9818	0.9789	0.9688	1.0000	0.9894
Right Lower Leg	0.9906	0.9664	0.9645	1.0000	0.9973
Right Upper Leg	0.9977	1.0000	0.9871	0.9886	0.9827
Left Foot	0.9661	0.9722	0.9721	1.0000	0.9913
Left Lower Leg	0.9759	0.9626	0.9593	1.0000	0.9998
Left Upper Leg	0.9801	0.9707	0.9809	1.0000	0.9839
Lower Torso	0.9929	0.9892	0.9885	1.0000	0.9913



slightly around the previous optimum. On the other hand, exceedingly low scatter distances impede a thorough exploration of the pose space in the case of faster movements. A balance must therefore be struck between convergence and flexibility, leading to the choice of $k_{\mathrm{IF}} = 0.1$.

Table 4.6 shows clearly the importance of the scattering coefficients for the APF performance. It is striking that the scattering heavily affects the placement of limbs, as the arms, while being nearly irrelevant to the placement of the body, as shown by the mean error ranges: The mean error for the arm varies by 2 cm in contrast to 0.2 cm for the rest of the body. As before for the inter-frame transitions, lower scattering favours convergence. Below a certain threshold, the benefit of convergence is offset by unsatisfactory exploration of the pose space, resulting in convergence on local maxima and a rising positioning error. Setting the scattering to $k_{\mathrm{ann}} = 0.45$ offers the best compromise.

4.4.2 Evaluation of the Likelihood Approximation

Since the observation likelihood approximation is applicable to any stochastic tracking approach requiring a weighting function, the following experiments examine its properties in separation from any tracking framework. The approximation is in itself time-stationary, i.e. the values are computed independently from previous observations.

Instead of motion sequences, the evaluation uses single poses which cover the range of human upper body motion. The approximation function is tested both on synthetic and on real-world point clouds. The synthetic poses have the advantage of providing a precisely known ground truth, while the real-world observations ensure that results are applicable to data gathered by real depth cameras.

Table 4.5: Relative error based on values of the IF frame scattering constant, with 1.0 being the maximum error in each row. Mean error ranges in meters: $\Delta e_{Limbs} = 0.0268$, $\Delta e_{Other} = 0.0113$.

IF Scattering	0.05	0.075	0.1	0.125	0.15
Arms					
Right Hand	0.7027	0.7750	0.8340	0.9041	1.0000
Right Lower Arm	0.8334	0.8697	0.9169	0.9757	1.0000
Right Upper Arm	0.8499	0.8827	0.9350	0.9799	1.0000
Left Hand	0.8248	0.7985	0.8857	0.8831	1.0000
Left Lower Arm	0.8564	0.8442	0.8864	0.9014	1.0000
Left Upper Arm	0.8196	0.8400	0.8863	0.9220	1.0000
Other					
Head	0.9325	0.9474	0.9546	0.9669	1.0000
Upper Body	0.9141	0.9251	0.9594	0.9901	1.0000
Right Foot	0.8913	0.9050	0.9216	0.9684	1.0000
Right Lower Leg	0.8785	0.8998	0.9129	0.9617	1.0000
Right Upper Leg	0.9479	0.9623	0.9644	0.9813	1.0000
Left Foot	0.8264	0.8697	0.9081	0.9687	1.0000
Left Lower Leg	0.8392	0.8537	0.8922	0.9592	1.0000
Left Upper Leg	0.9515	0.9610	0.9680	0.9911	1.0000
Lower Torso	0.9742	0.9828	0.9778	0.9871	1.0000

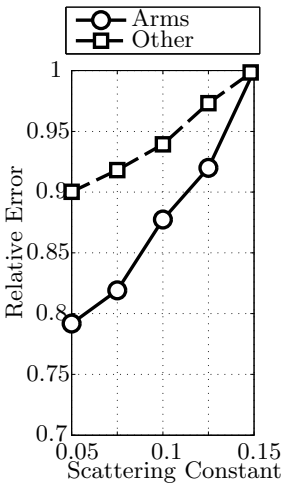


Table 4.6: Relative error based on values of the APF scattering constant, with 1.0 being the maximum error in each row. Mean error ranges in meters: $\Delta e_{Limbs} = 0.0202$, $\Delta e_{Other} = 0.0040$.

APF Scattering	0.4	0.45	0.5	0.55	0.6
Arms					
Right Hand	0.8031	0.7376	0.8563	0.9288	1.0000
Right Lower Arm	0.9186	0.8867	0.9306	0.9571	1.0000
Right Upper Arm	0.9479	0.9435	0.9667	0.9664	1.0000
Left Hand	0.7744	0.8579	0.8348	0.8785	1.0000
Left Lower Arm	0.8966	0.9460	0.9156	0.9415	1.0000
Left Upper Arm	0.9587	0.9524	0.9564	0.9714	1.0000
Other					
Head	1.0000	0.9909	0.9832	0.9755	0.9861
Upper Body	0.9761	0.9707	0.9731	0.9764	1.0000
Right Foot	0.9198	0.9321	0.9171	0.9846	1.0000
Right Lower Leg	0.9738	0.9628	0.9350	1.0000	0.9992
Right Upper Leg	1.0000	0.9999	0.9823	0.9857	0.9800
Left Foot	0.8953	0.9061	0.9117	0.9422	1.0000
Left Lower Leg	0.9269	0.9251	0.9018	0.9282	1.0000
Left Upper Leg	0.9915	1.0000	0.9737	0.9641	0.9721
Lower Torso	0.9984	1.0000	0.9840	0.9812	0.9817

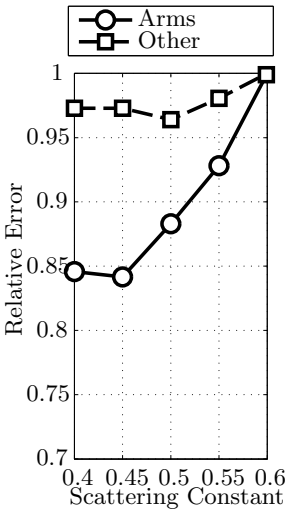


Table 4.7: Basic settings for the evaluation of the likelihood approximation. The influence of the various parameters is discussed in depth in Section 4.4.2.2.

k_1	20	k_2	2
Camera Depth Noise	±2.5 cm	Camera Raster Size	2.5 cm x 2.5 cm
Test Poses (synthetic)	20	Test Poses (real)	15
Samples per Test Pose	800	Self-collisions $N_{\text{Threshold}}$	5 (if active)

An individual data set consists of one reference pose and 800 sample poses, corresponding to one observation and 800 hypotheses. For the 20 synthetic test sets, a densely meshed upper body model rendered the reference poses to point clouds. Additive white noise emulated camera imperfections. Each of the synthetic reference clouds consists of an average of 500 data points with ideal joint and limb positions stored as ground truth. The 15 real-world point clouds were recorded both with a PointGrey BumblebeeXB3 and a Microsoft Kinect from depth-data only and stored with a hand-labelled ground truth.

Starting from the original reference pose $\mathbf{p}_{\text{ref}}$, uniform noise of varying amplitude (from 0.05 to 0.5 radian) is added to the joint variables, generating varied sample poses $\mathcal{S}_{\text{Sample}}$. These sample sets are created both for the synthetic and the real reference poses. Using the poses in these sets, the lower definition body model described in Section 4.2.2 renders the 800 sample point clouds corresponding to each reference pose. Joint and limb positions are then stored together with the corresponding sample point clouds.

The reference point clouds, both synthetic and real, act as the observations $\mathbf{Z}_i$ in this evaluation. The 800 sample poses take the place of the pose hypotheses or, for APF approaches, particles $\mathcal{S}_i = \left\{\mathbf{p}^{(1)}, \ldots, \mathbf{p}^{(800)}\right\}$. we can establish the correspondence between estimated likelihood and limb position error from the approximation $P\left(\mathbf{Z}|\mathbf{p}^{(n)}\right) \propto w\left(\mathbf{p}^{(n)}, \mathbf{Z}\right)$ found in Equation 4.12 and the average limb position error e_{Sample} for each sample. The total error e_{Sample} of a single sample pose is calculated as the mean standard deviation between all reference and sample limb positions ($\mathbf{x}_{\text{e, Sample}}$ and $\mathbf{x}_{\text{e, ref}}$)

$$e_{\text{Sample}} = \frac{1}{N_e} \sum_{e=1}^{N_e} \sqrt{(\mathbf{x}_{e,\text{ref}} - \mathbf{x}_{e,\text{Sample}})(\mathbf{x}_{e,\text{ref}} - \mathbf{x}_{e,\text{Sample}})^{\text{T}}} \,. \tag{4.17}$$

The mean likelihood is computed for bins of samples with similar e_{Sample}. A 2D plot can thus show the mean likelihood over given mean position errors. A normalisation is not strictly necessary, but helps in visualising the influences of various parameter settings. Table 4.7 shows the default parameters used in the following experiments. These tests explore the influence of various segmentation and collision detection approaches and parameters on the observation likelihood function

A separate synthetic data set examines the occlusion and ambiguity handling in a reduced 2D joint space. In this set, the derived samples are varied only in one shoulder angle and the connected elbow joint. This reduction allows a detailed analysis of occlusion and ambiguity handling in a constrained case.

Again, note that the function is evaluated outside of a tracker framework. This testbench approach focusses solely on the approximation of the likelihood function while excluding other effects originating in the APF tracker structure.

4.4.2.1 Comparison of Approximation Functions

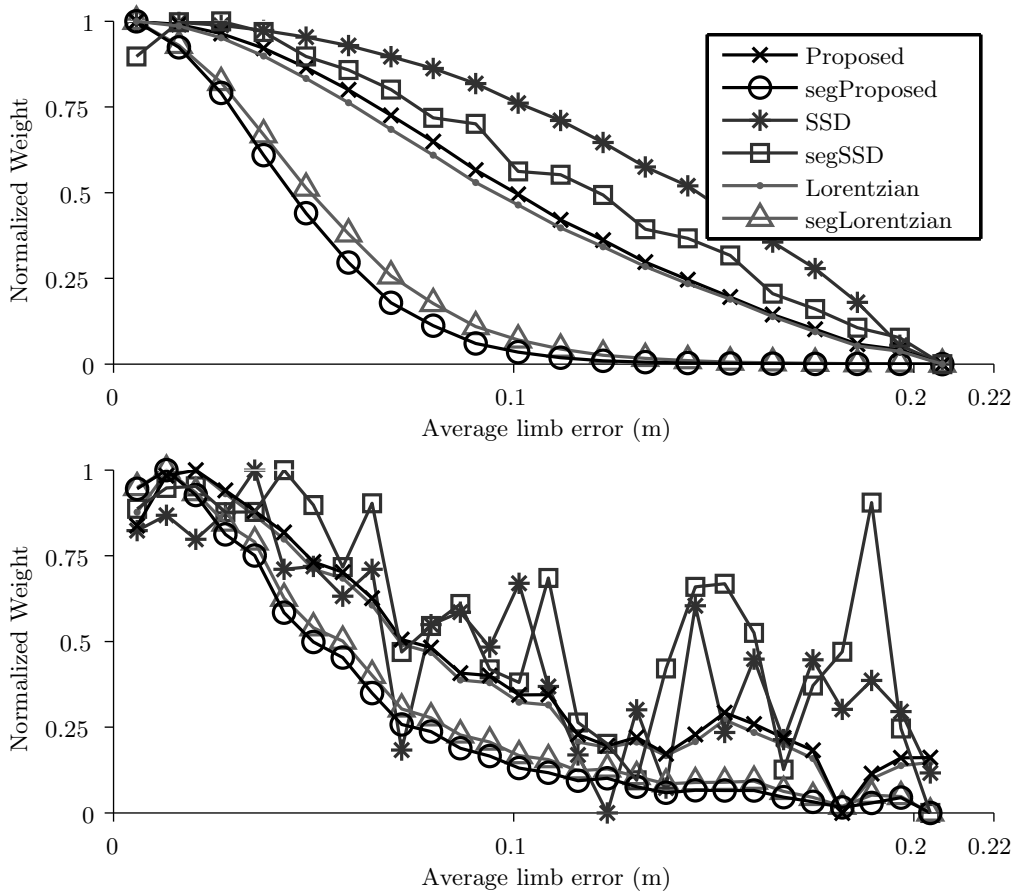


Figure 4.8: Comparison of the proposed approach (using configuration (5)), a Lorentzian function ($\delta = 0.3$) and SSD, both unsegmented and segmented. The upper plot shows the normalised weight on the synthetic dataset, the right plot uses the dataset recorded with a Kinect camera. All curves are normalised to the range $[0, 1]$ to facilitate comparison.

The likelihood approximation described in Section 4.2 is the result of incremental refinements to the point cloud matching employed by ICP algorithms. Figure 4.8 summarizes the results obtained with the different variations of the approximation function. The following equations thus trace the development of the final observation likelihood function over various adaptations of well-known algorithms and functions.

The first version of the likelihood approximation combines simple, unsegmented SSD terms, which are commonly used in iterative closest point (ICP) fitting [RL01]. Since

there is no prior knowledge on correspondences between data and model points available, a nearest-neighbour search associates a model point to each data point and vice versa. This association necessitates two terms. The first term $w_{\mathrm{SSD1}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right)$ matches the closest data point to to each element of the model point cloud. The second term $w_{\mathrm{SSD2}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right)$ repeats this process in the other direction, associating a model point to each element of the data point cloud. The model point cloud contains $N_{\mathbf{r}}$ points, whereas the data point cloud consists of $N_{\mathbf{g}}$ points. These two terms combine into the SSD likelihood approximation $w_{\mathrm{SSD}}(\mathbf{p}, \mathbf{Z}_t)$:

$$w_{\mathrm{SSD1}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \frac{1}{N_{\mathbf{r}}} \sum_{\mathbf{r} \in \mathcal{R}} \left(\min_{\mathbf{g}} \|\mathbf{r} - \mathbf{g}\|\right)^2 \tag{4.18}$$

$$w_{\mathrm{SSD2}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \frac{1}{N_{\mathbf{g}}} \sum_{\mathbf{g} \in \mathcal{G}} \left(\min_{\mathbf{r}} \|\mathbf{r} - \mathbf{g}\|\right)^2 \tag{4.19}$$

$$w_{\mathrm{SSD}}(\mathbf{p}, \mathbf{Z}_t) = w_{\mathrm{SSD1}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \times w_{\mathrm{SSD2}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \tag{4.20}$$

Unfortunately, this approximation returns high scores even for large pose errors, as shown by the "SSD" curve in Figure 4.8. In addition, the weights exceed $[0, 1]$ range. In a first refinement, the SSD approach is modified to allow for normalisation and segmentation. In this case, the segmentation of the sample points is based on their parent ellipsoid in the body model. The observed data points are clustered by a k-means approach, yielding K clusters. The new approximation function $w_{\mathrm{sSSD}}(\mathbf{p}^{(n)}, \mathbf{Z}_t)$ is computed as follows:

$$w_{\mathrm{sSSD1}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \prod_{e=1}^{N_e} \frac{1}{N_{\mathbf{r}}^{(e)}} \sum_{\mathbf{r} \in \mathcal{R}^{(e)}} \left(\min\left(1, \min_{\mathbf{g}} \|\mathbf{r} - \mathbf{g}\|\right)\right)^2 \tag{4.21}$$

$$w_{\mathrm{sSSD2}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \prod_{k=1}^{K} \frac{1}{N_{\mathbf{g}}^{(k)}} \sum_{\mathbf{g} \in \mathcal{G}^{(k)}} \left(\min\left(1, \min_{\mathbf{r}} \|\mathbf{r} - \mathbf{g}\|\right)\right)^2 \tag{4.22}$$

$$w_{\mathrm{sSSD}}(\mathbf{p}^{(n)}, \mathbf{Z}_t) = w_{\mathrm{sSSD1}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \times w_{\mathrm{sSSD2}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \tag{4.23}$$

The resulting plot of score over error ("segSSD" in Figure 4.8) is still quite imprecise and the added roughness impedes convergence in stochastic tracking frameworks. A second test of the SSD and segmented SSD approach using real data recorded with a Kinect camera underscores the lack of precision. More robust variants of the SSD exhibit the same problems, namely the logistics function [Col+77], the Fair function [Fai74] and the Talwar function [Hub64][6]. Applying these functions leads to nearly the same results as for the basic SSD based approximation discussed here. A more distinctly peaked base function on the other hand should deliver more precise results, i.e. an increase in pose error leads to an exponential decrease in approximated observation likelihood. Modifying

[6]The plotted results show the same characteristics as the SSD approach and were therefore excluded from Figure 4.8.

Equations 4.21 and 4.22, the SSD function is replaced by a mean of Lorentzian functions. Also known as Cauchy-Lorentz functions [Lor15], these terms use the variable δ as a parameter controlling the peak width

$$w_{\text{Lor1}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \prod_{e=1}^{N_e} \frac{1}{N_{\mathbf{r}}^{(e)}} \sum_{\mathbf{r} \in \mathcal{R}^{(e)}} \frac{\delta}{2\pi(0.25\delta^2 + \min_{\mathbf{g}} \|\mathbf{r} - \mathbf{g}\|^2)} \tag{4.24}$$

$$w_{\text{Lor2}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) = \prod_{k=1}^{K} \frac{1}{N_{\mathbf{g}}^{(k)}} \sum_{\mathbf{g} \in \mathcal{G}^{(k)}} \frac{\delta}{2\pi(0.25\delta^2 + \min_{\mathbf{r}} \|\mathbf{r} - \mathbf{g}\|^2)} \tag{4.25}$$

$$w_{\text{Lor}}(\mathbf{p}, \mathbf{Z}_t) = w_{\text{Lor1}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \times w_{\text{Lor2}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right) \tag{4.26}$$

The resulting approximation $w_{\text{Lor}}(\mathbf{p}, \mathbf{Z}_t)$ produces a smoother and more distinct score-error plot ("segLorentzian" in Figure 4.8) both for the synthetic and real dataset. Evaluating the Lorentzian function in this form requires one addition, two multiplications and one division for each single point. Using an exponential function instead, the overhead shrinks to one evaluation of the exponential function and a single multiplication. Since the effort of finding the closest corresponding point remains the same for all techniques discussed here, we can ignore it for the sake of comparison. This modification leads to the proposed method introduced in Section 4.2.3:

$$w_{\text{r2o}}^{\text{basic}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \prod_{e=1}^{N_e} \frac{1}{N_{\mathbf{r}}^{(e)}} \sum_{\mathbf{r} \in \mathcal{R}^{(e)}} \exp\left(-k_1 \min_{\mathbf{g}} \|\mathbf{r} - \mathbf{g}\|\right) \tag{4.27}$$

$$w_{\text{r2o}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \exp\left(k_2\left(1.0 - w_{\text{r2o}}^{\text{basic}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right)\right)\right) \tag{4.28}$$

$$w_{\text{o2r}}^{\text{basic}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \prod_{k=1}^{K} \frac{1}{N_{\mathbf{g}}^{(k)}} \sum_{\mathbf{g} \in \mathcal{G}^{(k)}} \exp\left(-k_1 \min_{\mathbf{r}} \|\mathbf{r} - \mathbf{g}\|\right) \tag{4.29}$$

$$w_{\text{o2r}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) = \exp\left(k_2\left(1.0 - w_{\text{o2r}}^{\text{basic}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right)\right)\right) \tag{4.30}$$

$$w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right) = w_{\text{r2o}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) \times w_{\text{o2r}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}_t^{(n)}\right), \mathcal{G}\left(\mathbf{Z}_t\right)\right) \tag{4.31}$$

The additional exponential functions in the Equations 4.28 and 4.30 are used to control the width of the final likelihood approximation distribution. As they are evaluated only once, they do not add significantly to the computational complexity. The segmented final likelihood approximation is shown as "segProposed" in Figure 4.8. The plot characteristics are essentially the same as for the segmented Lorentzian approximation, but require less operations for computation.

4.4.2.2 Influence of Parametrisation

The final likelihood function exposes several parameters for the tuning of precision and of processing speed. In the following section, the impact of changes on these parameter values are explored. The experiments assume configuration (5) from Table 4.7 as the basis for all parameters. For the following test, only one parameter is altered at a time.

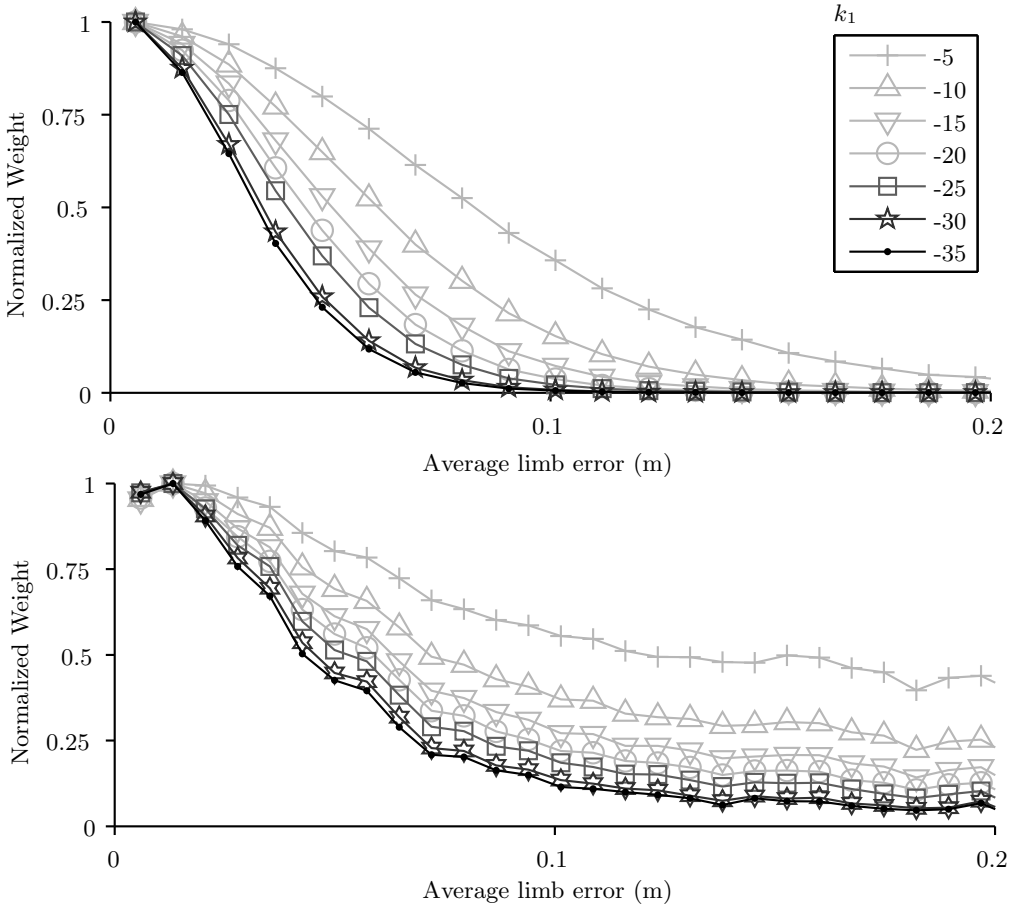


Figure 4.9: Likelihood approximation for various values of k_1 for both synthetic (top) and Kinect (bottom) data.

The parameters k_1 and k_2 lie at the heart of the likelihood approximation, shaping the steepness of the score-error response in the Equations 4.27, 4.28, 4.29 and 4.30. The Figures 4.9 and 4.10 show the influence of these two parameters on the approximation precision. The plotted curves showing the score over error are quite similar in shape, since both are based on the exponential function. The secondary exponential function in Equations 4.28 and 4.30 might thus appear superfluous at first glance. However, it is often advisable to set k_1 to gain a gentle likelihood curve for the partial weights and then refine their product in a second step by shaping the final curve more steeply by an appropriate value of k_2. Thus, the impact of a single badly fitting body part can be reduced, while

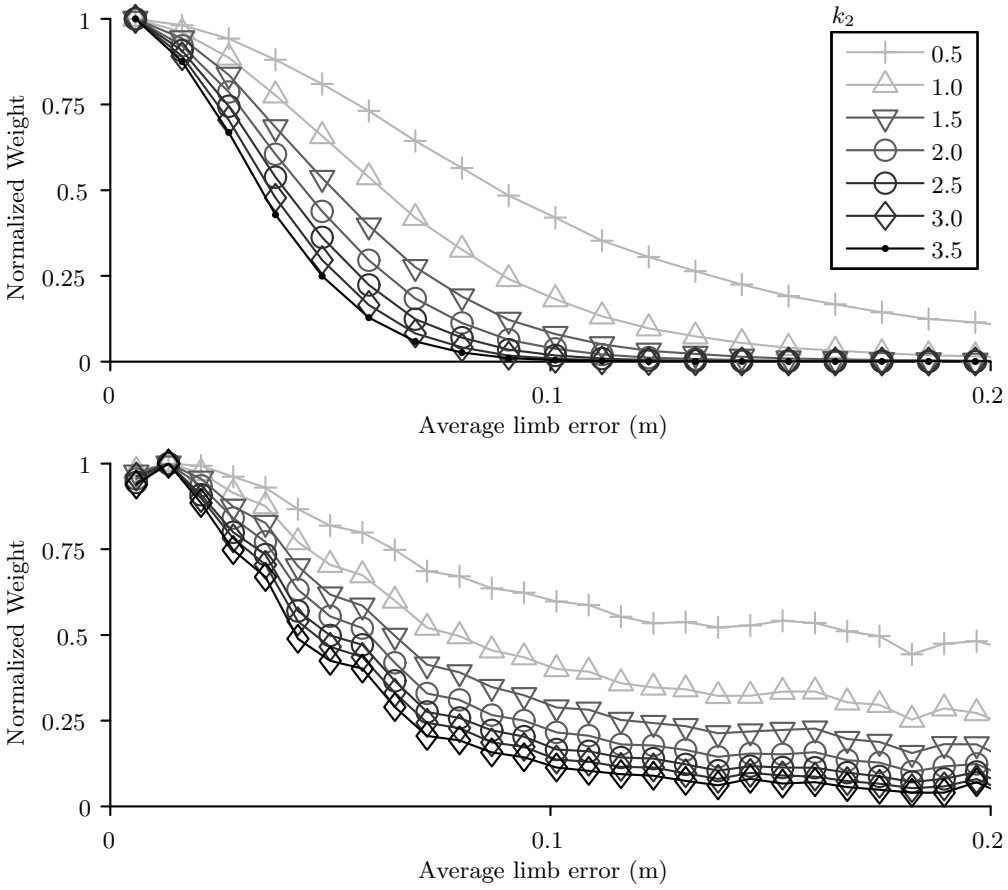


Figure 4.10: Likelihood approximation for various values of k_2 for both synthetic (top) and Kinect (bottom) data.

several badly fitted body parts still lead to low likelihoods. If this is no concern, the likelihood approximation may be adapted to skip the secondary exponential function:

$$w_{\mathrm{r2o}}^{\mathrm{basic}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right),\mathcal{G}\left(\mathbf{Z}\right)\right)=\prod_{e=1}^{N_e}\frac{1}{N_{\mathbf{r}}^{(e)}}\sum_{\mathbf{r}\,\in\,\mathcal{R}^{(e)}}\exp\left(-k_1\min_{\mathbf{g}}\left\|\mathbf{r}-\mathbf{g}\right\|\right) \tag{4.32}$$

$$w_{\mathrm{o2r}}^{\mathrm{basic}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right),\mathcal{G}\left(\mathbf{Z}\right)\right)=\prod_{k=1}^{K}\frac{1}{N_{\mathbf{g}}^{(k)}}\sum_{\mathbf{g}\,\in\,\mathcal{G}^{(k)}}\exp\left(-k_1\min_{\mathbf{r}}\left\|\mathbf{r}-\mathbf{g}\right\|\right) \tag{4.33}$$

$$w\left(\mathbf{p}_t^{(n)},\mathbf{Z}_t\right)=w_{\mathrm{r2o}}^{\mathrm{basic}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right),\mathcal{G}\left(\mathbf{Z}\right)\right)\times w_{\mathrm{o2r}}^{\mathrm{basic}}\left(\mathcal{R}_{\mathrm{v}}\left(\mathbf{p}^{(n)}\right),\mathcal{G}\left(\mathbf{Z}\right)\right) \tag{4.34}$$

Another important consideration is the number of clusters of observed data points. Figure 4.11 shows the effect of various cluster numbers for the observation data, while the clustering of model points is kept constant (cluster association by body element). It is

interesting to note that after the clustering of data points is introduced, which leads to the significant improvement discussed in Section 4.4.2.3, the effect of increasing the number of clusters is not very pronounced.

The complexity of the k-means clustering is linear to the k number of clusters, i.e. $\mathcal{O}(k)$. Therefore, it is advisable to find a balance between the number of clusters and desired precision. In this case, $k = 10$ serves as a good compromise. Note that for larger numbers of data points, i.e. when using a more detailed body model, the optimal number of clusters changes.

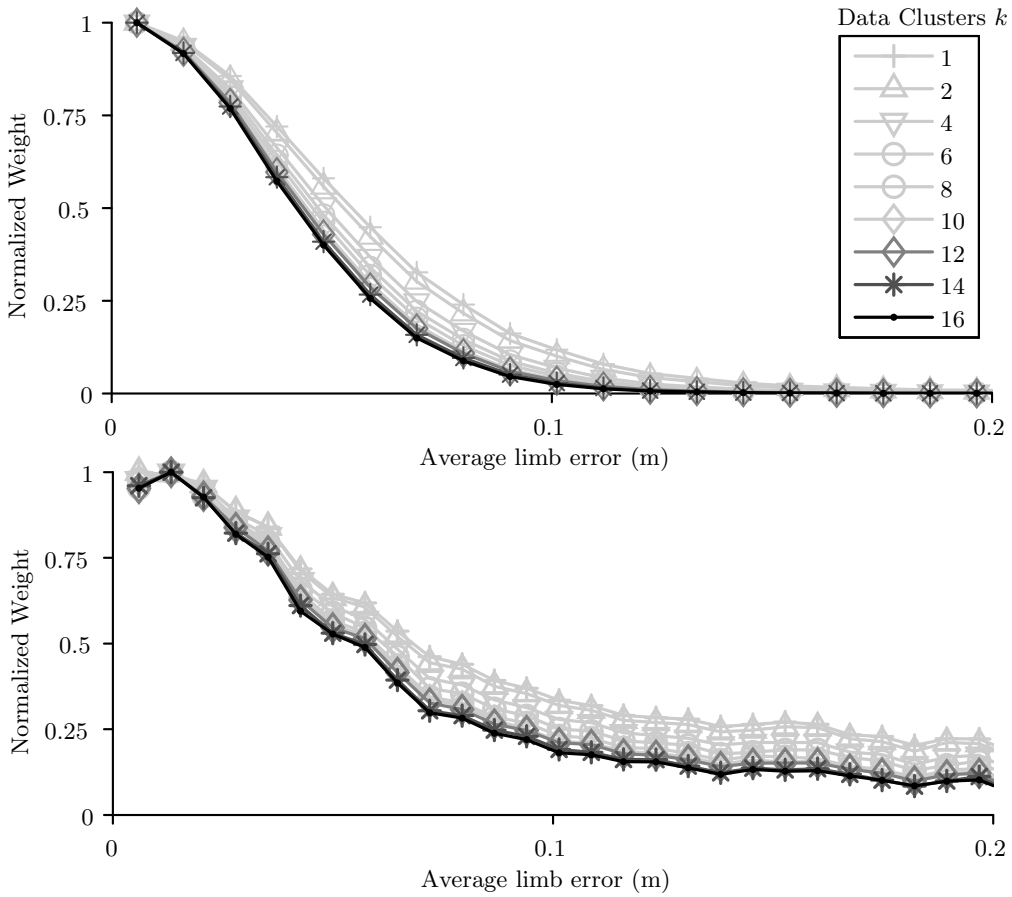


Figure 4.11: Likelihood approximation for various numbers k of data point clusters for both synthetic (top) and Kinect (bottom) data. The model points are statically segmented by body part.

Normally, the model points are assigned to clusters based on the body part they belong to. In case a k-means clustering is used instead, Figure 4.12 shows the effect of increasing the number of model point clusters. As previously observed for observation data point clustering, increases in the number of model point clusters lead only to small increments in precision. Again, a balance between computational complexity and precision is required. As the clustering by association to body parts produces better results at

negligible computational cost, this approach should be preferred. Nevertheless, model point clustering can still be used for scenarios where no prior body part association is known.

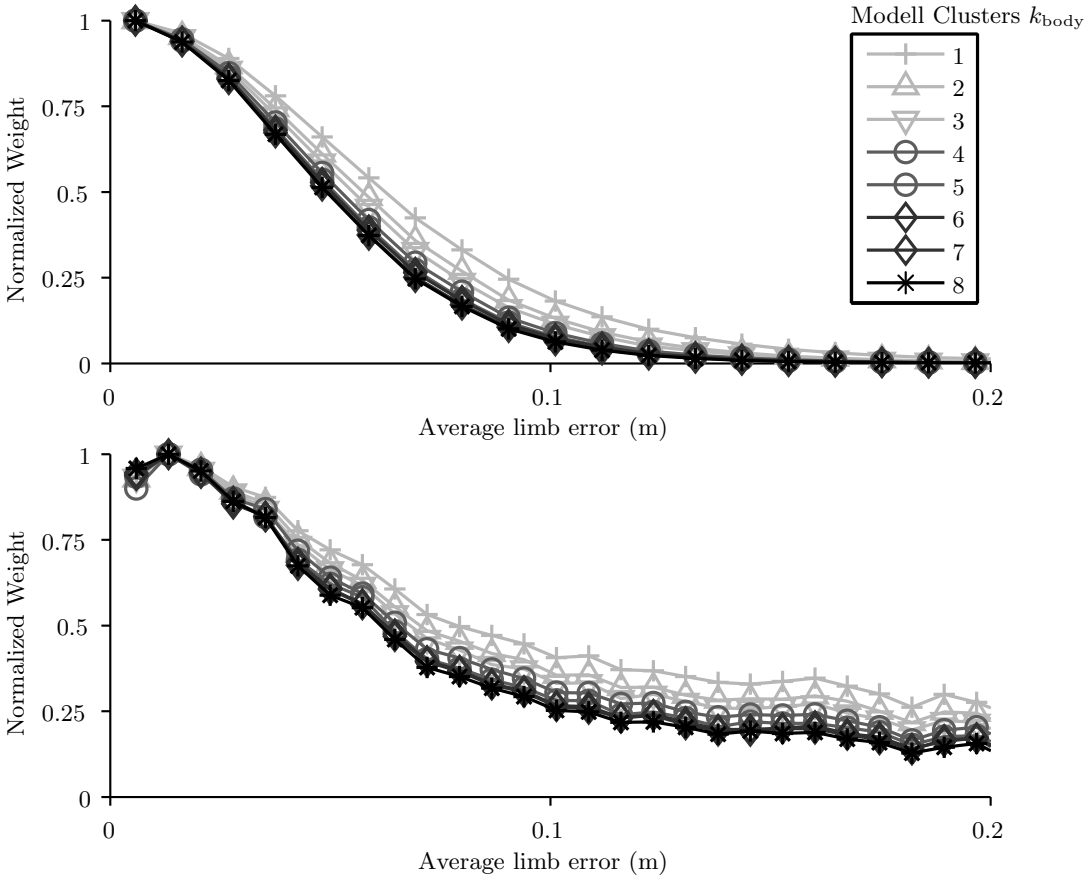


Figure 4.12: Likelihood approximation for various numbers k_{body} of body point clusters for both synthetic (top) and Kinect (bottom) data in a scenario with k-means clustered model points. The data points are segmented by k-means clustering with $k = 10$.

4.4.2.3 Evaluation of Different Clustering Strategies

Figure 4.13 presents the normalised average weights $w\left(\mathbf{p}_t^{(n)}, \mathbf{Z}_t\right)$ over position error for different approximation configurations. The configurations represent various clustering strategies and are summarised in Table 4.8. It is obvious that a simple distance measure, as shown in curve (1), does not give sufficient precision or robustness. This effect is mostly due to the negligible influence of errors in smaller regions, such as hands, on the overall score.

Curve (2) shows the effect of clustering the data points by k-means previous to the computation of $w_{\text{o2r}}\left(\mathcal{R}_{\text{v}}\left(\mathbf{p}^{(n)}\right), \mathcal{G}\left(\mathbf{Z}\right)\right)$. The decline of likelihood values over rising error

Table 4.8: Settings for the different configurations, numbering of configurations as shown in Figure 4.13. Configurations (4) & (6) are used with disabled self-collision checks. Configurations (5) & (7) apply the self-collision handling.

Configuration	Variable	Value
(1)	Data Clusters	1
	Model Clusters	1
(2)	Data Clusters	10 (k-means)
	Model Clusters	1
(3)	Data Clusters	1 (k-means)
	Model Clusters	9 (by element)
(4) & (5)	Data Clusters	10 (k-means)
	Model Clusters	9 (by element)
(6) & (7)	Data Clusters	10 (k-means)
	Model Clusters	5 (k-means)

is more pronounced than in configuration (1) but still shows a lack of robustness especially for larger position errors. Similarly, curve (3) illustrates the effect of clustering the model points by body element, such as a hand or a forearm. The improvement in accuracy is not as distinct as for configuration (2), but still notable.

Configuration (4) combines clustering of the model points by body element and clustering of data by k-means clustering. The decline of likelihood over rising error is even more greater than in configuration (2) and (3), suggesting higher precision. While the strong gradient indicates a high standard deviation, we can expect the higher precision (synthetic: 13.9% lower weight at $0.1\,m$ error) to offset the slightly lower robustness. Configuration (5) is essentially a variation of configuration (4), including the collision penalty in the calculation of the likelihood. Since in this test scenario only few collisions are possible, the effect is not readily visible.

The curve plotting results for configuration (6) shows a similar shape as the curve for configuration (4), with k-means clustering used instead of the partitioning by body element. 5 clusters were found to yield the best results. Despite a similar increase of precision (synthetic: 10.5% lower weight at $0.1\,m$ error compared to configuration (2)), this method has the disadvantage of requiring k-means clustering for every single pose sample. As clustering by body elements shows a better precision and lower computational effort, it appears preferable to k-means clustering. For the sake of completeness, curve (7) shows the influence of additional collision penalties. As before for configuration (5), the low potential for self-collisions prevents any significant differences to become visible.

4.4.2.4 Verification of Real Camera Performance versus Synthetic Data

Considering Figure 4.14, the real-world data essentially follows the performance shown in the synthetic scenarios. Note that the weighting does not drop towards zero as quickly as in the synthetic dataset. This lower decrease can be traced to greater differences between the body model and the real, observed person. Especially loose clothing on the real person

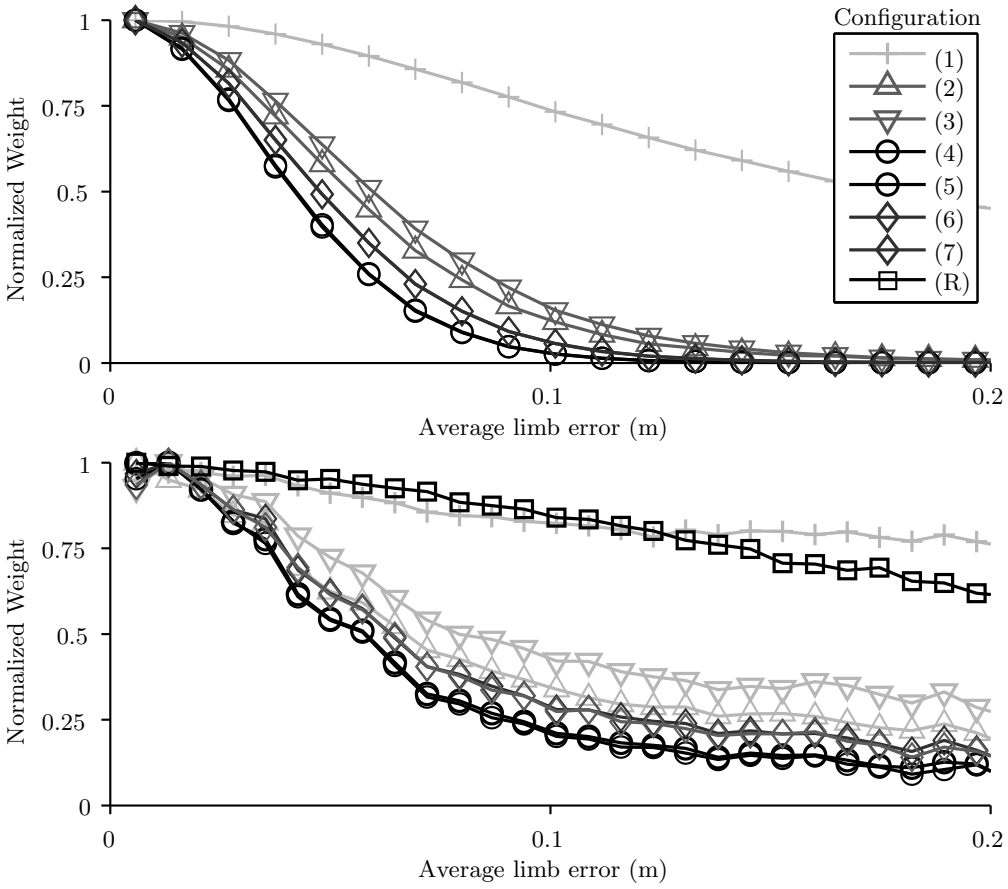


Figure 4.13: Average error for various likelihood approximations over position error for depth data from a synthetic source (top) and Kinect (bottom). The following configurations were used (summarised in Table 4.7): (1) No clustering or collision check at all, (2) k-means clustering of data points, (3) clustering of model points by body element, (4) clustering both model points and data points, (5) same as (4) with collision penalties, (6) clustering of model points by k-means, (7) same as (6) with collision penalties. (R) shows the results obtained from the reference approach (taken from [Gan+10]).

and the more slender shape of the ellipsoid body model result in a mismatch between the observed point cloud and the model. As the limbs of the body model can assume slightly differing poses within the observed point cloud, larger deviations from the reference pose become possible, adding up to higher average errors. This imprecision can be balanced by performing an automatic adaptation of the ellipsoid model within an initialisation stage (e.g. using a pre-known initialisation pose).

Comparing the two camera systems directly, no significant differences in accuracy or convergence appear. It is however interesting to note that the Kinect camera system yields a slightly smoother approximation than the BumbleebeeXB3. Since both depth images

were sampled to the same resolution, this smoother error-score plot might reflect the lower noise level of depth images obtained with a structured lighting approach.

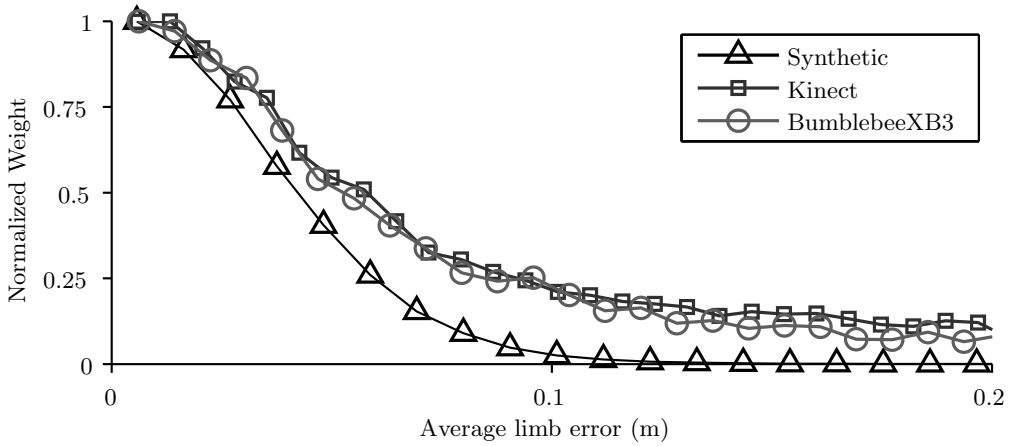


Figure 4.14: Comparison of average error for synthetic and real camera data, using configuration (5).

4.4.2.5 Comparison to a Reference Technique

The approximation function described in Section 4.2 is also compared to the approximation methods proposed by Ganapathi *et al.* [Gan+10]. While there are a number of more recent methods, Ganapathi's approach is a commonly used reference and thus serves as a good point for comparison.

A Matlab implementation of their observation likelihood approximation is tested on the Kinect dataset and compared to the approach described in the sections above. The results are normalised to the range $[0, 1]$ and are given as a reference curve (R) in Figure 4.13.

The synthetic sample depth images required by Ganapathi's method are generated using Autodesk 3ds Max 2011 and 3DVIA Virtools 5.0 for posing and rendering. The sample poses and original depth data are identical to the ones used by the proposed method. The generative polygon body model is adjusted to approximate the shape of the observed person.

The direct comparison between the approach presented in this dissertation and the approximation used in Ganapathi's tracker shows the benefits of segmentation: As shown in Figure 4.13, the reference approach (R) is more robust and precise than the unsegmented approach shown by curve (1). Once partial scores (2) or clustered evidence (3) are introduced, the segmented approximation yields more precise results. This improvement is expected, as Ganapathi *et al.*'s likelihood approximation does not differentiate between large areas of high uniformity, like the chest, and smaller areas which are more crucial to a precise fitting, e.g. the arms. However, it should be noted that their approach allows for higher framerates on current hardware. As that algorithm is based on standard graphics operations, many steps can be computed efficiently using built-in hardware acceleration.

4.4.2.6 Collision Detection in Ambiguity Handling

The effect of collision detection becomes apparent studying Figure 4.15. This scenario studies a synthetic point cloud with both arms held out in parallel in front of the body. Only the rotation of the shoulder and the angle of the subsequent elbow change, leading to a simplified 2D problem suitable for illustration in print. The brightness in these plots corresponds with the likelihood score for a given alignment of the two angles.

Without collision detection, it is easy to place one arm to collide with the other. The left plot (without collision penalties) shows a ridge (i.e. high likelihood scores) around joint angles placing the right upper arm and hand near or within the left arm. Although the likelihood is visibly reduced for these poses, the short distances between model points of the right arm and data points of the left arm still yield high partial scores. In the right plot, the collision detection enacts a penalty for any pose placing the right arm within the volume of the left arm (formally expressed in Equation 4.12), resulting in a distinct trough for angles placing the right arm in an illegal position. Although the trough does not totally eliminate the ridge and a small local maximum, it generates a more distinct gradient towards the global maximum. This trough in likelihood scores is equivalent to an implicit penalty on collisions between limbs.

4.4.2.7 Clustering and Self-Occlusions

In Figure 4.16 only one shoulder angle and the elbow angle of the nearly obscured right arm are varied. This scenario regards a synthetic model with one arm hidden behind the torse. No observations are available for that arm. As in the previous example, the reduction to a 2 DoF problem allows for an illustrative visualisation in print.

The left plot shows the likelihood scores when no model point clustering is performed. Any pose bringing many unoccluded model points close to the clustered data points leads to higher average likelihood scores. In consequence, there is a ridge of higher scores for shoulder angles bending the arm forwards, towards the visible data points. This resolution is obviously wrong, as there are no observations indicating the arm being visible. However, the proximity of model points to observed points grants higher similarity scores. In turn, the correct occluded pose gets lower scores than the incorrect poses placing the arm closer to observed points. Without clustering of the model points by body elements, the averaging of the likelihood score in Equation 4.27 leads to a diminished influence of evidence proximity to model points.

Using configurations (4) and (5) from Table 4.8, the algorithm sets partial scores for occluded body parts to “1” (see Section 4.2.3). Thus, scores for poses placing the arm behind the body are not negatively influenced. Simultaneously, badly placed single body elements receive lower partial likelihood scores. This decrease in scores offsets the gains achieved from placing a few model points closer to the observed data points. The large range of likely angles for the single shoulder and elbow joints illustrates this effect. None of the poses in the high scoring pose range contradict the observation. So even as the actual pose remains undetermined, there is a distinct range in which it can be expected with high confidence. The model point clustering thereby ensures the correct handling of occlusions by treating unobservable body parts as weighting-neutral. In the absence of

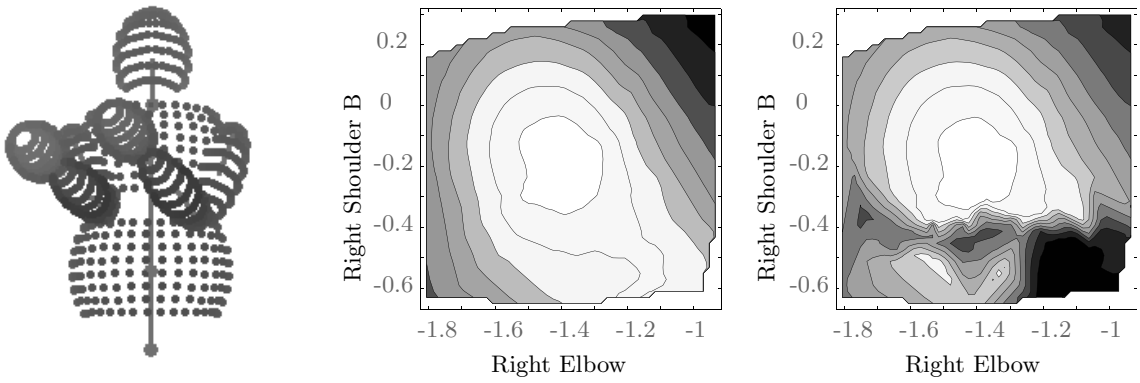


Figure 4.15: (left) reference pose with two arms in close proximity, (middle) likelihood over joint angles without collision penalty, (right) likelihood over joint angles with collision penalty. Brighter areas indicate higher likelihood.

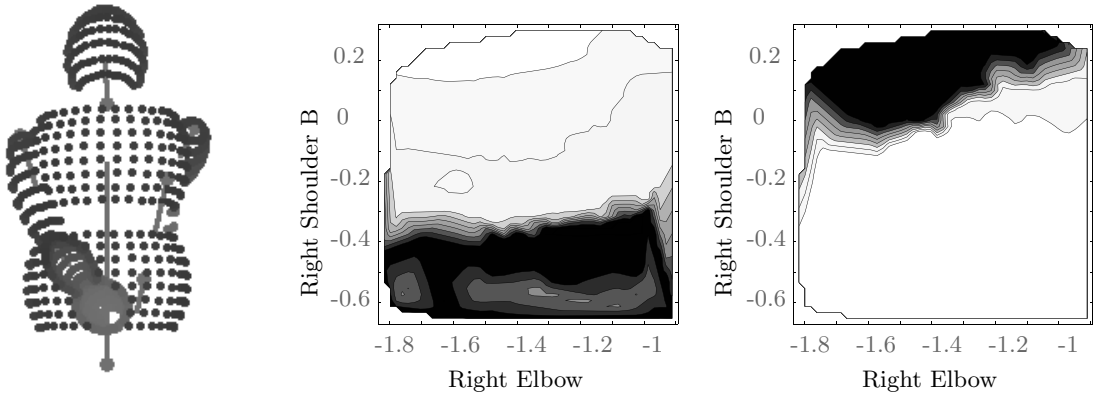


Figure 4.16: (left) reference pose with the articulated right arm nearly obscured, (middle) likelihood considering only minimal point distances with clustered data points, (right) likelihood after clustering of both reference and model points. Brighter areas indicate higher likelihood.

further data all poses not contradicting the observation have an equal likelihood. This rule is founded on the separation between observation likelihoods and the motion model as specified in Section 4.2.1.

4.5 Summary of Chapter

This chapter introduced a purely point cloud based human pose tracking framework. The APF approach was extended to work directly on 3D data gathered by one or more depth-

sensing cameras and perform pose estimation in real-time. A GPU based implementation is able to process 3D data at about 40 fps using standard hardware. Since the pose tracking is performed on 3D point clouds, it is easy to fuse data coming from different cameras. The approach is therefore especially suitable for VR and AR scenarios with a distributed camera system.

A number of modifications to the scattering and resampling mechanisms of conventional APF trackers reduces the number of particles necessary for tracking by injecting hypotheses tailored to human motion patterns. Especially the extending inverse kinematics for elbow and knee joints help in generating plausible alternative poses.

The observation likelihood approximation has shown promising performance under test-bench conditions for both synthetic and real-world datasets. The focus on using solely 3D data as evidence allows for likelihood estimation on a wide range of sensors, while the exclusion of colour cues makes the system lighting-independent. The proposed approach compares favourably against the approximation function used by a reference system with regards to accuracy, if not speed. It is also interesting to note the large impact of segmentation and clustering on approximation accuracy (see Section 4.4.2.3). Giving smaller, more salient regions equal weight as larger, less salient regions boosts approximation accuracy and reduces marginalisation effects.

The ellipsoid body model allows for fast sample generation. The seamless integration of the occlusion handling by model point clustering produces good results even for nearly obscured limbs. Thus the occlusion handling prevents misplacements and constrains regions of likelihood to comply with available observations. The collision penalties have been shown to detect and effectively suppress impossible poses, although a smoother likelihood gradient in the border state-space would be desirable. The proposed observation likelihood approximation is adaptable to nearly all current stochastic tracking approaches, presuming the availability of 3D point clouds. With only minimal modifications, the same method is applicable to calibrated multi-camera scenarios.

Integration of the likelihood approximation into the APF framework leads to a human pose tracker. This tracker is capable of following a wide range of human motion in real-time without the need for a dedicated human body part detector, as employed by other teams [AAD08; Sho+11; Pla+10; Gan+10].

The reconstructed user pose serves as input for a number of secondary modules in the context of the AR videoconference. Hand, head and upper body positions are crucial input parameters for gesture-based interaction schemes. Furthermore, user position and gaze direction are relevant parameters for the consensus reality computation. In the following chapter, the integration of the user position and heading into the consensus reality computation will be described in detail.

Chapter 5

Consensus Reality

5.1 Introduction

As laid out in Chapter 2, this dissertation aims to generate a consensus reality spanning two rooms. The previous chapters described the general architecture and the pose tracking of the users interacting in this shared space. In this chapter we turn to the question of how the spatial qualities of the consensus are defined.

Considering the treatment of space in VR and AR remote collaboration, there are to date three distinct approaches to be found in literature. Firstly, many systems disregard the space surrounding the users entirely. These are typically immersive VR systems such as a CAVE or HMD solution [Gro+03; KB13]. The shared virtual workspace supersedes the actual physical surroundings of the participants and acts as the sole basis for interaction and communication.

The second approach encompasses unilateral collaboration systems aimed at supporting a user performing specific tasks, e.g. repairing a device [AAT13; Gur+12; Sod+13; Oye+13]. These systems define one of the participant's rooms as the primary space. The remote advisor can observe this space and give advice, add annotations or manipulate virtual content. The advisor's surroundings are usually not included in the interaction.

Last of the three categories, there are window-constrained videoconferencing approaches. These are already commonly used in everyday applications, such as *Skype*, *Google Hangouts* and similar systems. Although many of these are limited to the transmission of audio and video, recently a number of papers have proposed the inclusion of the physical surroundings into the interaction [MF11; LER12]. Yet while the interaction encompasses the physical surroundings of both users, the display still acts as a clear separation between the users.

In summary, there are currently three major approaches: Disregarding all physical surroundings, electing one room as the primary interaction space or using a window analogy to separate the participating spaces. Only recently a paper by Maimone *et al.* [Mai+13] hinted at a possible fourth approach. Their concept of a HMD-based teleconferencing setup for AR assumes that two participating rooms share the same virtual space, so that furniture from one room can appear in the other room as a virtual object.

Taken further, this leads to the concept of a consensus AR collaboration space spanning two or more rooms. In contrast to immersive systems, the surroundings are not

disregarded, but instead actively integrated into the composition and rendering of the scene. Thus, users can be assured that their surroundings share essential features with those of their conversation partners. They can include common objects into their interaction, e.g. table surfaces. Disparities can also be marked, such as furniture present only in one of the rooms.

This visualisation is especially important if we consider that both users are rendered into their conversation partner's room. The different layouts of these rooms pose a significant challenge to the illusion of co-presence. For purpose of illustration, let us assume that one user has a small office, while the conversation partner is standing in a larger meeting room. As the participant in the spacious meeting room paces back and forth during the conversation, the corresponding remote avatar would appear to step through the walls of the much smaller office of his partner. Obviously, such breaks in the illusion of cohabitation of space should be avoided where possible.

By identifying common features and disparities arising from mapping two physical rooms into a shared coordinate system, an algorithm can create maps of the CR. These maps are then rendered into the conversation partners' views, allowing them to avoid obstacles present in the other room.

The origins of the coordinate systems describing our physical surroundings are placed freely. Adjustments to their alignment can lead to virtual consensus spaces with desired properties, e.g. a maximum of shared, free floorspace. In practice, this leads to an optimisation problem in which one of the rooms is translated and rotated against the other. A specialised energy function rewards certain characteristics of the CR and informs the optimisation process.

In the following chapter, this optimisation procedure will be considered in detail. Figure 5.1 shows the overall processing chain. In Section 5.2, the mapping process of two rooms into a common virtual coordinate systems is described. Section 5.3 then presents an energy function aiming to optimise certain aspects of the alignment. The actual solution of the resulting optimisation problem is examined in Section 5.5. Finally, Section 5.9 closes the chapter and summarizes the approach.

5.2 Mapping of a Consensus Reality

5.2.1 Prerequisites

The mapping stage relies on pre-existing scans of the participating rooms. Such scans are commonly obtained using off-the-shelf software, e.g. products such as *Scanekt*, *ReconstructMe* and others. Conceptually, these tools follow the *KinectFusion* volumetric reconstruction approach pioneered by Izadi *et al.* [Iza+11]. Based on previous ICP approaches, their solution substitutes the usual fusing of point clouds with a volumetric surface reconstruction. A signed distance function evaluated on the GPU helps integrating new observations into the existing space of observations. In combination with commercially available RGB-D sensors such as Microsoft's Kinect, the reconstruction achieves a high accuracy for spaces of up to $5 \times 5 \times 5\text{m}^3$ volume. As the process builds a volumetric representation of the space on the GPU, the amount of available memory limits the size of the model.

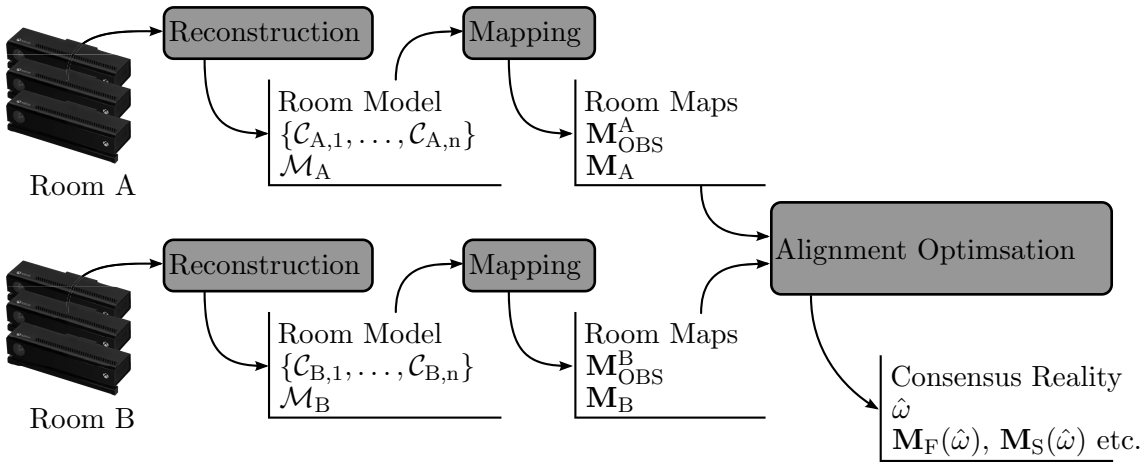


Figure 5.1: Overall workflow for the consensus reality alignment process. Section 5.2 describes the reconstruction and mapping stages. Section 5.3 then elaborates on the optimisation procedure. The variables shown will be explained in detail in the following chapters.

There are a number of recent improvements, most of which aim to overcome the constraints on the volume which can be scanned [Sal+13; New+11]. Despite the prevalence of volumetric approaches, there is also promising research on more conventional SLAM approaches. Notable are especially contributions by Steinbrücker *et al.* [Ste+13] which demonstrate the integration of large spaces into a coherent model[1].

In the context of the AR videoconference discussed in this dissertation, the primary source for geometric data of the rooms is the fixed camera system also used for observing the users. An additional, mobile camera might be used to fill in holes of the resulting mesh.

The rooms are assumed to have even and uninterrupted floor surfaces. There is only one level floor in each room, i.e. stairs, sloped surfaces and ramps are not permissible. The results of the 3D scans are triangulated meshes with points ordered and indexed as connected vertices. Although the mesh in its entirety does not need to be complete or closed, the floor surface intended for use in the interaction must be free of holes. In practice, the application of a screened Poisson resurfacing [KH13] followed by quadric edge collapsing [GH97] to a desired volumetric resolution will yield meshes suitable for the steps described in the following sections.

5.2.2 Mapping the Room Geometry

Once the scanned meshes of both rooms are prepared, the first stage of the analysis process generates maps of the layout for each single room. These maps contain information on

[1]It should be remarked that I have supervised a number of theses following related approaches, namely those by Drexl [Dre12a; Dre12b] and Schäfer [Sch12].

static properties of the rooms, such as placement of furniture, camera positions and shape of the floor.

With most scanning approaches, the origins of the local coordinate systems are placed arbitrarily in space. The resulting reference frames of room A and room B are designated as $\mathcal{F}_\mathrm{A}$ and $\mathcal{F}_\mathrm{B}$ respectively. The following operations assume that the origin is located in the floor plane of the respective rooms. If the scanning method used does not automatically place the origin accordingly, a reference marker may be used to find the transformation matrix $T_\mathrm{Floor}^\mathrm{Origin}$ between the arbitrary origin and a point in the floor plane [KB99].

In the following, the two room meshes are designated as $\mathcal{M}_\mathrm{A}$ and $\mathcal{M}_\mathrm{B}$. All maps show the meshes projected onto the floor plane of their original rooms. As most buildings and rooms follow a 2D floorplan layout, this reduction is admissible for most indoor rooms. The meshes $\mathcal{M}_\mathrm{i}$ are made up of individual polygons $\mathcal{V}_k$ which in turn each consist of three vertices $\mathbf{v} = (v_x, v_y, v_z)$ in a counter-clockwise arrangement (indicating the surface normal):

$$\mathcal{M}_\mathrm{i} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_n\} \tag{5.1}$$

$$\mathcal{V}_k = \left\{ \begin{pmatrix} v_{1,\mathrm{x}} \\ v_{1,\mathrm{y}} \\ v_{1,\mathrm{z}} \end{pmatrix}, \begin{pmatrix} v_{2,\mathrm{x}} \\ v_{2,\mathrm{y}} \\ v_{2,\mathrm{z}} \end{pmatrix}, \begin{pmatrix} v_{3,\mathrm{x}} \\ v_{3,\mathrm{y}} \\ v_{3,\mathrm{z}} \end{pmatrix} \right\} \tag{5.2}$$

In the following, these meshes will be refined into two primary maps of each room. First, the map $\mathbf{M}_\mathrm{i}$ contains a top-down view of room i. The pixels of the map show the height of each object above the floor, with walls set to a generic high value. A second map $\mathbf{M}_\mathrm{OBS}^\mathrm{i}$ assigns an observability score to each pixel of the previous map $\mathbf{M}_\mathrm{i}$.

A first step regards only the sub-meshes $\mathcal{M}_\mathrm{furn}^\mathrm{i}$ which might be part of furniture. These sub-meshes $\mathcal{M}_\mathrm{furn}^\mathrm{i}$ contain only polygons $\mathcal{V}$ located above the floor plane [2] and entirely below the ceiling [3]. An orthographic projection renders the vertices $\mathbf{v}_j$ contained in $\mathcal{M}_\mathrm{furn}^\mathrm{i}$ to the 2D floor plane.

The floor plane map of space occupied by furniture $\mathbf{M}_\mathrm{furn}^\mathrm{i}$ is defined as a discrete 2D array of fixed size. The array covers an area of $s_x \times s_y\, m^2$. Using the pixel-per-meter ratios c_x and c_y, this leads to arrays of the size $|\mathbf{M}|_x \times |\mathbf{M}|_y = c_x s_x \times c_y s_y$.

The absolute transformation between the floor plane and the mesh origin is given by $T_\mathrm{Floor}^\mathrm{Origin}$. The function `triangle` (A, B, C, t) draws a filled triangle $A \Rightarrow B \Rightarrow C \Rightarrow A$ into a specific floor plane map using the *half-space function* or a similar rendering function. The filled region is set to the value t. The following procedure draws each polygon to the floor plane:

[2] The z-component of all vertices is higher than a fixed offset: $\exists j \in \{1,2,3\} \,:\, V_{j,\mathrm{z}} > c_\mathrm{floor}$ with typically $c_\mathrm{floor} = 10\,\mathrm{cm}$.

[3] The z-component of all vertices is lower than a fixed offset: $\forall j \in \{1,2,3\} \,:\, V_{j,\mathrm{z}} < c_\mathrm{ceil}$ with typically $c_\mathrm{ceil} = 2\,\mathrm{m}$

$$
\begin{aligned}
&\forall \mathcal{V}_k \in \mathcal{M}^{\mathrm{i}}_{\mathrm{furn}}: \\
&\quad \forall \mathbf{v}_{kj} \in \mathcal{V}_k: \\
&\qquad \hat{\mathbf{v}}_{kj} = T^{\mathrm{Origin}}_{\mathrm{Floor}} \mathbf{v}_{kj} && (5.3)\\
&\qquad \hat{v}_{kj,\mathrm{x}} = \lfloor c_x \cdot \hat{\mathbf{v}}_{kj,x} + 0.5 \cdot s_x \rfloor && (5.4)\\
&\qquad \hat{v}_{kj,\mathrm{y}} = \lfloor c_y \cdot \hat{\mathbf{v}}_{kj,y} + 0.5 \cdot s_y \rfloor && (5.5)\\
&\qquad t_k = \tfrac{\hat{\mathbf{v}}_{k1,\mathrm{z}} + \hat{\mathbf{v}}_{k2,\mathrm{z}} + \hat{\mathbf{v}}_{k3,\mathrm{z}}}{3} && (5.6)\\
&\texttt{triangle}\,(\hat{v}_{k1}, \hat{v}_{k2}, \hat{v}_{k3}, t_k) \Rightarrow \mathbf{M}^{\mathrm{i,k}}_{\mathrm{furn}} && (5.7)
\end{aligned}
$$

This procedure results in a 2D map array $\mathbf{M}^{\mathrm{i,k}}_{\mathrm{furn}}$ for every single polygon. In this map, the area containing the polygon is set to the height of its centre-point. For situations in which several polygons are projected into the same floorspace, it is sufficient to consider only the highest surface from the floor. The final map of furniture is therefore computed by finding the maximum projected value for each pixel $\mathbf{p}(x, y)$:

$$
\forall \mathbf{p}(x,y) \in \mathbf{M}^{\mathrm{i}}_{\mathrm{furn}}: \quad \mathbf{M}^{\mathrm{i}}_{\mathrm{furn}}(\mathbf{p}) = \max_{\forall \mathcal{V}_k \in \mathcal{M}^{\mathrm{i}}_{\mathrm{furn}}} (\mathbf{M}^{\mathrm{i,k}}_{\mathrm{furn}}(\mathbf{p})) \tag{5.8}
$$

The previous steps result in a map showing the height of all pieces of furniture within a room. For a full map of the room, wall and floor features must be distinguished as well. Most room scanning approaches yield only a mesh for the surface of walls, but not the obstructed space behind the wall. A separate step is therefore needed to generate a map of space within walls. A modified version of the previous furniture mapping approach finds regions in the floor plane which do not have any polygons mapped onto them. As there is no data for these regions, they are assumed to be hidden behind walls. The previous mapping operation is thus repeated for all meshes $\mathcal{M}_{\mathrm{i}}$, without regard to their height above the floor plane. The new map $\mathbf{M}^{\mathrm{i}}_{\neg\mathrm{walls}}$ shows all regions for which observation data is available, regardless of other properties. Inversion of the 2D array $\mathbf{M}^{\mathrm{i}}_{\neg\mathrm{walls}}$ by using a bitwise thresholding on each pixel $\mathbf{p}(x, y)$ results in a map $\mathbf{M}^{\mathrm{i}}_{\mathrm{walls}}$:

$$
\forall \mathbf{p}(x,y) \in \mathbf{M}^{\mathrm{i}}_{\neg\mathrm{walls}}: \quad \mathbf{M}^{\mathrm{i}}_{\mathrm{walls}}(\mathbf{p}) = \begin{cases} c_{\mathrm{ceil}} & \text{if } \mathbf{M}^{\mathrm{i}}_{\neg\mathrm{walls}}(\mathbf{p}) = 0 \\ 0.0\,m & \text{else} \end{cases} \tag{5.9}
$$

This map contains generic high values for all spaces not included in the original observation and serves as a map of the surrounding walls. The combination of both maps results in a full floor plan map $\mathbf{M}_{\mathrm{i}}$ of a single room:

$$
\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{i}}: \quad \mathbf{M}_{\mathrm{i}}(\mathbf{p}) = \begin{cases} \mathbf{M}^{\mathrm{i}}_{\mathrm{walls}}(\mathbf{p}) & \text{if } \mathbf{M}^{\mathrm{i}}_{\mathrm{walls}}(\mathbf{p}) > \mathbf{M}^{\mathrm{i}}_{\mathrm{furn}}(\mathbf{p}) \\ \mathbf{M}^{\mathrm{i}}_{\mathrm{furn}}(\mathbf{p}) & \text{else} \end{cases} \tag{5.10}
$$

In addition to the geometric layout, the observability of space is a major factor for videoconferencing applications. In this context, observability denotes the availability of

visual data for a user standing at a specific spot in a room. As an example, the intersection of four camera views would be highly observable, while a corner invisible to the cameras would be unobservable. Since the videoconference requires visual data of the participants, unobservable regions are marked in a map $\mathbf{M}^{\mathrm{i}}_{\mathrm{OBS}}$ and later discarded during the CR generation. For a static arrangement of cameras, observability is a property of a room itself. The known camera positions and orientations cast view cones onto the floor plane. These view cones then yield observability maps for each room. As before, the maps are computed on a discretised floor plane.

The process starts by iterating through all available cameras $\{\mathcal{C}_{\mathrm{i,1}}, \mathcal{C}_{\mathrm{i,2}}, \ldots, \mathcal{C}_{\mathrm{i,n}}\}$. The cameras are calibrated relative to a given world coordinate system with a known transformation $T^{\mathrm{Origin}}_{\mathrm{Camera}}$ between origin and camera. As in the geometry mapping, the transformation $T^{\mathrm{Origin}}_{\mathrm{Floor}}$ provides adjustments for possible offsets between the original calibration frame and the floor reference frame. The camera view cone will usually intersect the floor plane as a parabola or ellipse. In this dissertation, the approach detailed by Schneider and Eberley [SE03, pp. 563 sqq.] maps the cone to a plane[4]. The procedure renders all view cones to a floor plane map $\mathbf{M}_{\mathrm{i,Cameras}}$ of the same size and orientation as the joint floor plan maps $\mathbf{M}_{\mathrm{i}}$. RGB-D cameras have a minimum distance c_{minDepth} and maximum distance c_{maxDepth} for which they can provide depth data. Therefore, the cone mapping is followed by a second step in which distance-clipped regions are marked for exclusion from the observability map. A thresholding step culls map pixels outside the observable range based on the distance $d^{\mathbf{p}}_{\mathrm{camera}}$ of each floor element to the camera origin. The following sequence of functions maps the view cones to the floor plane:

$$\forall \mathcal{C}_{\mathrm{k}} \in \{\mathcal{C}_{\mathrm{i,1}}, \mathcal{C}_{\mathrm{i,2}}, \ldots, \mathcal{C}_{\mathrm{i,n}}\}:$$

$$\forall \mathbf{p}(x, y) \in \mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}:$$

$$\mathbf{x}_{\mathbf{p}} = \begin{pmatrix} \left(x - 0.5\ |\mathbf{M}|_x\right) / c_x \\ \left(y - 0.5\ |\mathbf{M}|_y\right) / c_y \\ 0 \end{pmatrix} \tag{5.11}$$

$$\mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}(\mathbf{p}) = \begin{cases} 1 & \text{if } \texttt{WithinCone}(\mathbf{x}_{\mathbf{p}}, \mathcal{C}_{\mathrm{k}}, \mathcal{F}_{\mathrm{i,\ floor}}) \text{ as in [SE03]} \\ 0 & \text{else} \end{cases} \tag{5.12}$$

$$d^{\mathbf{p}}_{\mathcal{C}_{\mathrm{k}}} = \|\mathbf{x}_{\mathbf{p}} - \mathbf{x}_{\mathcal{C}_{\mathrm{k}},\mathrm{Origin}}\|_2 \tag{5.13}$$

$$\mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}(\mathbf{p}) = \begin{cases} \mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}(\mathbf{p}) & \text{if } c_{\mathrm{minDepth}} \leq d^{\mathbf{p}}_{\mathcal{C}_{\mathrm{k}}} < c_{\mathrm{maxDepth}} \\ 0 & \text{else} \end{cases} \tag{5.14}$$

After this procedure, the maps $\mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}$ show the view cones for each camera[5]. These observability maps $\mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}$ are then consolidated into a single map $\mathbf{M}^{\mathrm{i}}_{\mathrm{view}}$. The combined

[4] The basic problem of finding the section between a conic and a plane is among the oldest topics discussed by mathematicians with the oldest surviving works by Apollonius of Perga around 200 B.C. [Apo66]. Proper attribution becomes difficult with so old a field.

[5] Effects like occlusion cast by furniture are not considered. A ray-tracing approach could be used to improve on the observability mapping.

map yields the final observability score map $\mathbf{M}^{\mathrm{i}}_{\mathrm{OBS}}$ by considering the minimum number c_{minViews} and desired number c_{desViews} of cameras observing the separate space volumes:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}^{\mathrm{i}}_{\mathrm{view}}:$$

$$\mathbf{M}^{\mathrm{i}}_{\mathrm{view}}(\mathbf{p}) = \sum_{k=0}^{n} \mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}(\mathbf{p}) \tag{5.15}$$

$$\mathbf{M}^{\mathrm{i}}_{\mathrm{OBS}}(\mathbf{p}) = \begin{cases} 1 & \text{if } \mathbf{M}^{\mathrm{i}}_{\mathrm{view}}(\mathbf{p}) > c_{\mathrm{desViews}} \\ \dfrac{\mathbf{M}^{\mathrm{i}}_{\mathrm{view}}(\mathbf{p}) - c_{\mathrm{minViews}}}{c_{\mathrm{desViews}} - c_{\mathrm{minViews}} + 1} & \text{if } c_{\mathrm{minViews}} \leq \mathbf{M}^{\mathrm{i}}_{\mathrm{view}}(\mathbf{p}) < c_{\mathrm{desViews}} \\ 0 & \text{if } \mathbf{M}^{\mathrm{i}}_{\mathrm{view}}(\mathbf{p}) < c_{\mathrm{minViews}} \end{cases} \tag{5.16}$$

After repeating this procedure for both rooms A and B, there are two equally sized occupancy maps $\mathbf{M}_{\mathrm{A}}$ and $\mathbf{M}_{\mathrm{B}}$ as well as two observability maps $\mathbf{M}^{\mathrm{A}}_{\mathrm{OBS}}$ and $\mathbf{M}^{\mathrm{B}}_{\mathrm{OBS}}$. Comparing these maps yields information on the topology and observability of the consensus reality created by the AR videoconference.

5.2.3 Consolidation into a Common Coordinate System

When placing both rooms into the same reference system, we can shift the relative alignment of the two rooms. So far, all mapping steps used a local reference frame located in the floor plane. When merging the two rooms into the same reference system, an optional transform $T^{\mathrm{A}}_{\mathrm{B}}$ shifts the alignment of the two reference frames. This transformation can be expressed as offset and rotation $\omega = \{x, y, \theta_z\}$ in the 2D floor plane. After the alignment is applied to the map $\mathbf{M}_{\mathrm{B}}$ of room B, the new map is denoted as $\mathbf{M}^{\omega}_{\mathrm{B}}$.

The overlay of the aligned maps defines a consensus reality spanning the two rooms. Note that this is just one of many possible consensus realities — each alignment ω results in a different CR. The alignment procedure discussed in the following Section 5.3 will use optimisation methods for finding the best possible consensus reality. The remainder of this section describes the creation of a single consensus reality based on a given alignment ω. Four different types of space make up this consensus reality, derived from the room meshes $\mathcal{M}_{\mathrm{A}}$ and $\mathcal{M}_{\mathrm{B}}$ and their consecutive mapping:

1. Space unobstructed both in room A and B, i.e. common free space.
2. Space obstructed both in room A and B, i.e. common obstacles.
3. Space obstructed in room A, but unobstructed in room B.
4. Space obstructed in room B, but unobstructed in room A.

When considering the space with common obstacles, there are two possible combinations:

1. Space with different types of obstacles in the two rooms
2. Space containing pieces of furniture of similar height

The last point becomes interesting for cases where the furniture is of similar height, e.g. for tables present on both sides of the conversation. Here we can find surfaces which might be integrated into the actual interaction. As an example, these surfaces could be used for placing virtual objects or as collaborative workspaces.

The map $\mathbf{M}_\mathrm{F}$ of common open floorspace combines the occupancy maps $\mathbf{M}_\mathrm{A}$ and $\mathbf{M}_\mathrm{B}^\omega$ in the consensus reality. This process assumes that all elements lower than $c_\mathrm{floor} = 0.1\,\mathrm{m}$ belong to the floor. The overall process is visualised in Figures 5.2 and 5.3.

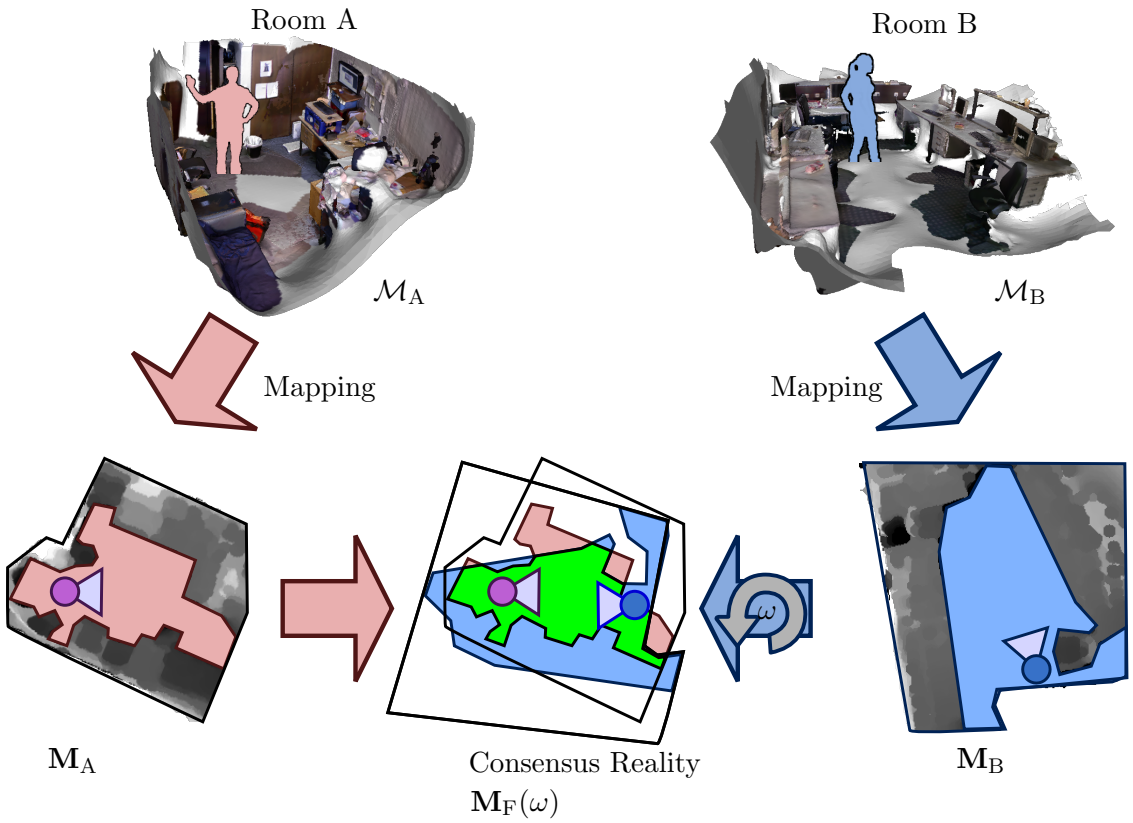


Figure 5.2: Basic approach for consensus space computation: Two rooms are mapped to the 2D floor plane, users' positions (shown in blue and purple) are remapped accordingly. The transformation ω is applied to the room B. Both maps are then overlaid and the energy terms are computed. For instance, the area mapped in green shows common free floorspace for a given pose used for computing $E_\mathrm{free}(\omega)$.

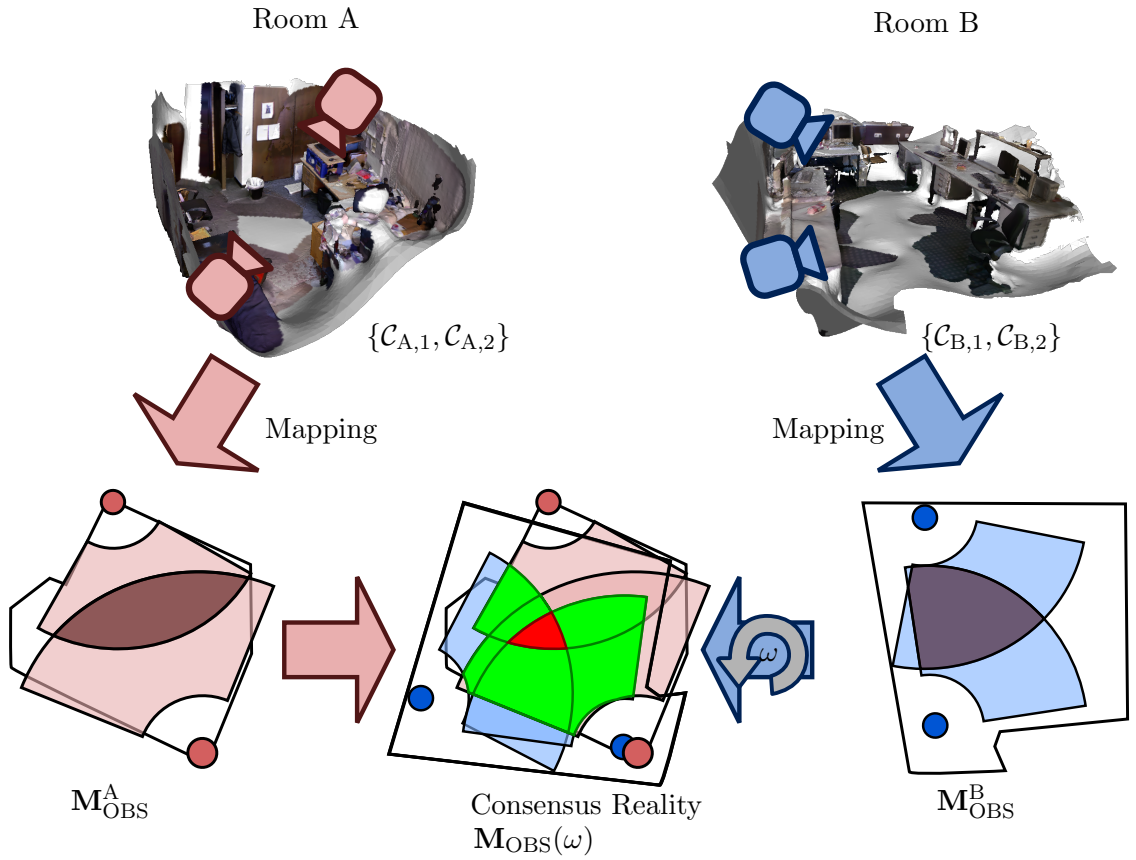


Figure 5.3: Exemplary calculation of mutual observability score $\mathbf{M}_{\text{OBS}}$ for two scenes with two cameras each. Only the region marked in red permits user tracking and recording by two cameras in both rooms. Regions marked in green are visible by at least one camera in each scene. The red region is visible by both cameras in each scene, providing best observability

The following comparisons lead to the map shared free floorspace $\mathbf{M}_{\text{F}}(\omega)$:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\text{A}} \wedge \mathbf{M}^{\omega}_{\text{B}}:$$

$$\mathbf{M}_{\text{OR}}(\mathbf{p},\omega) = \begin{cases} \mathbf{M}_{\text{A}}(\mathbf{p}) & \text{if } \mathbf{M}_{\text{A}}(\mathbf{p}) > \mathbf{M}^{\omega}_{\text{B}}(\mathbf{p}) \\ \mathbf{M}^{\omega}_{\text{B}}(\mathbf{p}) & \text{else} \end{cases} \tag{5.17}$$

$$\mathbf{M}_{\text{F}}(\mathbf{p},\omega) = \begin{cases} 1 & \text{if } \mathbf{M}_{\text{OR}}(\mathbf{p},\omega) < c_{\text{floor}} \\ 0 & \text{else} \end{cases} \tag{5.18}$$

This binary map shows the free space available to both participants. Within the regions marked “1”, there are no obstacles on either side of the conversation.

Similarly, the room maps contain all information needed in order to identify obstacles unique to one room. The maps of unilateral obstacles $\mathbf{M}_{\mathrm{AO}}$ and $\mathbf{M}_{\mathrm{BO}}$ are found by applying boolean operators as follows:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{A}}:$$

$$\mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega) = \begin{cases} \mathbf{M}_{\mathrm{A}}(\mathbf{p}) & \text{if } \mathbf{M}_{\mathrm{A}}(\mathbf{p}) \leq \mathbf{M}_{\mathrm{B}}^{\omega}(\mathbf{p}) \\ \mathbf{M}_{\mathrm{B}}^{\omega}(\mathbf{p}) & \text{else} \end{cases} \tag{5.19}$$

$$\mathbf{M}_{\mathrm{AO}}(\mathbf{p},\omega) = (\neg \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega)) \wedge \mathbf{M}_{\mathrm{A}}(\mathbf{p}) \tag{5.20}$$

$$\mathbf{M}_{\mathrm{BO}}(\mathbf{p},\omega) = (\neg \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega)) \wedge \mathbf{M}_{\mathrm{B}}^{\omega}(\mathbf{p}) \tag{5.21}$$

Processing the boolean AND map $\mathbf{M}_{\mathrm{AND}}$ together with a map $\mathbf{M}_{\mathrm{HeightDiff}}$ of obstacle height differences yields two new maps. The first is a map $\mathbf{M}_{\mathrm{CO}}$ of obstacles common to both rooms, the second is a map $\mathbf{M}_{\mathrm{S}}$ of consensus surfaces. Typically the procedure would consider surfaces with less than $c_{\mathrm{diff}} = 0.05\,\mathrm{m}$ difference in height to be a potential shared workspace. Common obstacles are all pieces of furniture of different height that are less than $c_{\mathrm{max}} = 2.0\,\mathrm{m}$ high. The following procedures populate the two new maps:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{A}}:$$

$$\mathbf{M}_{\mathrm{HeightDiff}}(\mathbf{p},\omega) = \|\mathbf{M}_{\mathrm{A}}(\mathbf{p}) - \mathbf{M}_{\mathrm{B}}^{\omega}(\mathbf{p})\| \tag{5.22}$$

$$\mathbf{M}_{\mathrm{CO}}(\mathbf{p},\omega) = \begin{cases} \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega) & \text{if } \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega) \leq c_{\mathrm{max}} \\ 0 & \text{else} \end{cases} \tag{5.23}$$

$$\mathbf{M}_{\mathrm{S}}(\mathbf{p},\omega) = \begin{cases} \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega) & \text{if } \mathbf{M}_{\mathrm{HeightDiff}}(\mathbf{p},\omega) \leq c_{\mathrm{diff}} \wedge \mathbf{M}_{\mathrm{AND}}(\mathbf{p},\omega) \leq c_{\mathrm{max}} \\ 0 & \text{else} \end{cases} \tag{5.24}$$

For the workspace and the free floorspace, the observability is an important factor. It is of no use to have a large common workspace if there are no cameras providing user data for these regions. So in a first step, the common observability $\mathbf{M}_{\mathrm{OBS}}$ of the consensus space is computed from the room specific maps $\mathbf{M}_{\mathrm{OBS}}^{\mathrm{A}}$ and $\mathbf{M}_{\mathrm{OBS}}^{\mathrm{A}}$. The lower view score of both rooms determines the score for each floor element in order to ensure mutual observability. This assignment guarantees that for every map element with a value higher than zero, there is at least the minimum required number of cameras c_{minViews} directed at that space on both sides. This yields the observability-weighted maps of free floorspace $\mathbf{M}_{\mathrm{FOBS}}$ and common work surfaces $\mathbf{M}_{\mathrm{SOBS}}$:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{OBS}}:$$

$$\mathbf{M}_{\mathrm{OBS}}(\mathbf{p},\omega) = \min\left(\mathbf{M}_{\mathrm{OBS}}^{\mathrm{A}}(\mathbf{p}), \mathbf{M}_{\mathrm{OBS}}^{\mathrm{B}}(\mathbf{p},\omega)\right) \tag{5.25}$$

$$\mathbf{M}_{\mathrm{FOBS}}(\mathbf{p},\omega) = \mathbf{M}_{\mathrm{OBS}}(\mathbf{p},\omega) \times \mathbf{M}_{\mathrm{F}}(\mathbf{p},\omega) \tag{5.26}$$

$$\mathbf{M}_{\mathrm{SOBS}}(\mathbf{p},\omega) = \mathbf{M}_{\mathrm{OBS}}(\mathbf{p},\omega) \times \mathbf{M}_{\mathrm{S}}(\mathbf{p},\omega) \tag{5.27}$$

In summary, there are 5 different maps describing the consensus AR space.

- $\mathbf{M}_{\text{FOBS}}$ shows the consensus free space available to both conversation partners.
- $\mathbf{M}_{\text{AO}}$ shows obstacles present only in room A, but not in room B.
- $\mathbf{M}_{\text{BO}}$ shows obstacles present only in room B, but not in room A.
- $\mathbf{M}_{\text{CO}}$ shows obstacles present in both rooms, but not of equal height.
- $\mathbf{M}_{\text{SOBS}}$ shows obstacles present in both rooms and of equal height, e.g. table tops present in both rooms.

These maps are valid for a specific alignment of the two rooms, denoted T_{B}^{A} for the 3D transformation or ω for the 2D maps. The next section shows the optimisation process for determining the best possible alignment for a given pair of rooms.

5.3 Automatic Alignment for AR Videoconferencing

5.3.1 Goals and Constraints

The alignment of two rooms poses one of the biggest challenges when creating a consensus reference space for use in an AR videoconference. In a simplistic implementation, the two existing reference frames could serve as fixed anchor points. Thus both rooms would be mapped into the same common reference frame without any adjustments. This approach is actually quite common in current research, e.g. in Maimone *et al.*'s conferencing approach [Mai+13]: The operators place the reference frames as visual markers into their work spaces, taking care to find the correct alignment manually. Such approaches are usually sufficient to ensure an unobstructed shared work space for experimental platforms in a laboratory setting. However, once collaboration is taking place in unprepared, cluttered settings, this manual adjustment quickly becomes cumbersome.

Using the maps derived in the previous section, it is possible to express the alignment as an optimisation problem. As there are no directly measurable quantities describing the quality of a given alignment ω, it is necessary to first identify desired properties and then translate these into quantifiable values.

For this dissertation, the application scenario is defined as a conversation between two users in physically separate office or home environments (see also Section 2.4). This scenario leads to a number of socially motivated goals and constraints.

- There should be enough shared space for moving around.
- If possible, there should be a shared work surface.
- Users should never start the conversation in space occupied by furniture or walls.
- Users should never start the conversation standing too close together.
- If possible, the meeting should start with the users facing each other.
- If possible, walls and furniture between the two rooms should be aligned, i.e. walls should run parallel, desks should stand at straight angles etc.

These goals and constraints can be expressed as functions returning specific values for a given alignment of rooms. As most optimisation approaches operate by minimising or maximising energy functions, the goals and constraints are formulated as terms returning high values for favoured configurations and low values for unsatisfactory placements.

5.3.2 Formulating an Energy Function

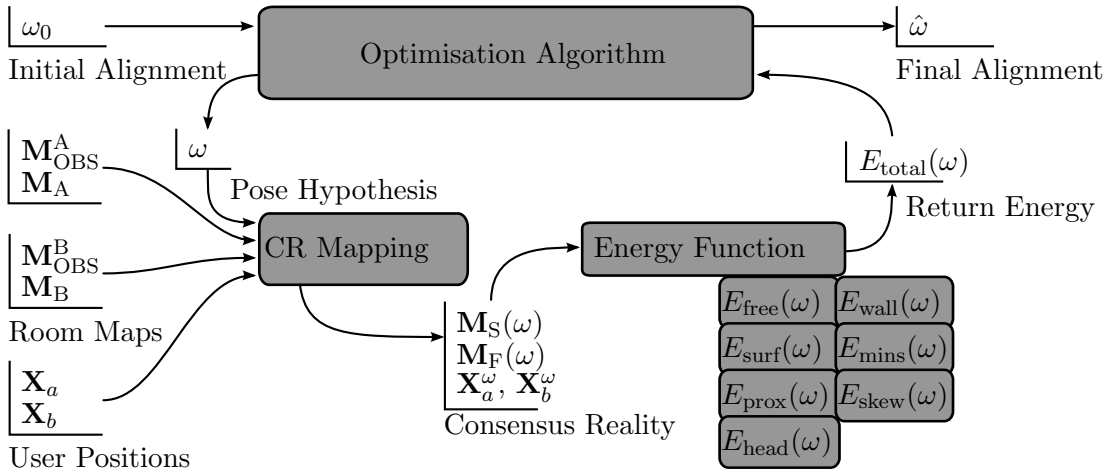


Figure 5.4: Integration of the energy function into an arbitrary optimisation framework.

Individual energy terms represent the goals and constraints formulated in the previous section. Since the optimisation methods used here are based on stochastic approaches searching for the highest likelihood, this section performs optimisation by finding the maximum of the function.

- $E_{\mathrm{free}}(\omega)$ returns high values for large unobstructed floorspace. It is weighted by the factor α_{free}.
- $E_{\mathrm{surf}}(\omega)$ returns high values for large common work surfaces. These are however not necessarily uninterrupted and can be distributed over the room. It is weighted by the factor α_{surf}.
- $E_{\mathrm{prox}}(\omega)$ returns high values if users are not standing too close together. It is weighted by the factor α_{prox}.
- $E_{\mathrm{head}}(\omega)$ returns high values if the users are facing each other. It is weighted by the factor α_{head}.
- $E_{\mathrm{wall}}(\omega)$ returns high values if the users are not standing in an obstacle, such as a wall or furniture. It is weighted by the factor α_{wall}.
- $E_{\mathrm{mins}}(\omega)$ returns high values for large, connected work surfaces, e.g. a large table present in both rooms. It is weighted by the factor α_{mins}.

- $E_{\text{skew}}(\omega)$ returns high values if walls and pieces of furniture are aligned in straight angles or in parallel. It is weighted by the factor α_{skew}.

The basic structure of the energy function is then expressed as a weighted sum of these partial energy terms. In addition, there is a single binary term α_E which can be used to signal specific "knock-out" criteria, such as initial user placement in walls. The following equation combines all the partial energy terms into the overall alignment energy $E_{\text{total}}(\omega)$:

$$\begin{aligned} E_{\text{total}}(\omega) = \alpha_E(&\alpha_{\text{free}}E_{\text{free}}(\omega) + \alpha_{\text{prox}}E_{\text{prox}}(\omega) + \alpha_{\text{head}}E_{\text{head}}(\omega) + \alpha_{\text{wall}}E_{\text{wall}}(\omega) \\ &+ \alpha_{\text{surf}}E_{\text{surf}}(\omega) + \alpha_{\text{mins}}E_{\text{mins}}(\omega) + \alpha_{\text{skew}}E_{\text{skew}}(\omega)) \end{aligned} \tag{5.28}$$

The partial terms each describe only a single aspect of the alignment. As a common convention, each term returns a value in the range between "0" and "1". Here "0" signifies a total mismatch and "1" denotes an optimal alignment.

5.3.3 3 DoF Optimisation

Section 5.2.3 introduced the alignment ω. This parameter controls the relative mapping of the two coordinate system origin frames to each other. For the 3 DoF problem, the alignment contains therefore the following parameters:

$$\begin{aligned} \omega_3 = (x, y, \theta) \ &\in \ \mathbb{R}^3 \\ \theta &\in [-180°, 180°] \end{aligned} \tag{5.29}$$

The x and y parameters control the translation of room B within the floor plane and are given in meters. The θ parameter denotes the rotation of room B around its centre. The positions of the users remain fixed relative to their respective room reference system, denoted as $\mathbf{X}_a$ for user A and $\mathbf{X}_b$ for user B. As the alignment is applied to the reference frame of room B, the position of user B relative to room A is referred to as $\mathbf{X}_b^{\omega}$. This 3 DoF approach is suitable for scenarios where the users want to start a meeting as quickly as possible.

5.3.4 9 DoF Optimisation

For refined meeting scenarios, it makes sense to take more time and set up the meeting carefully. In this case, the user positions are not taken as given, but instead the algorithm tries to suggest optimal positions for each user to start the meeting. This additional suggestion step leads to a 9 DoF problem, where the alignment variable contains not only the alignment of reference frames, but also both user positions and orientations. These are denoted with the sub-indices "A" and "B" respectively:

$$\begin{aligned} \omega_9 = (x, y, \theta, x_{\text{A}}, y_{\text{A}}, \theta_{\text{A}}, x_{\text{B}}, y_{\text{B}}, \theta_{\text{B}}) \ &\in \ \mathbb{R}^9 \\ \theta, \theta_{\text{A}}, \theta_{\text{B}} &\in [-180°, 180°] \end{aligned} \tag{5.30}$$

The parameters follow the same naming conventions as for the 3 DoF problem. Again x and y determine translation, while θ governs rotations. The first three parameters control the alignment of room B relative to the origin of room A. The following values give the user placements, also relative to the reference system anchored to room A. Within the consensus reference system, the suggested user positions are henceforth denoted as $\mathbf{X}_a^\omega$ and $\mathbf{X}_b^\omega$. Other than in the 3 DoF case described in Section 5.3.4, these are independent of the geometric alignment of the map origins. Thus, the room geometry alignment can be optimised independently from the user placement.

5.3.5 Illustrative Example

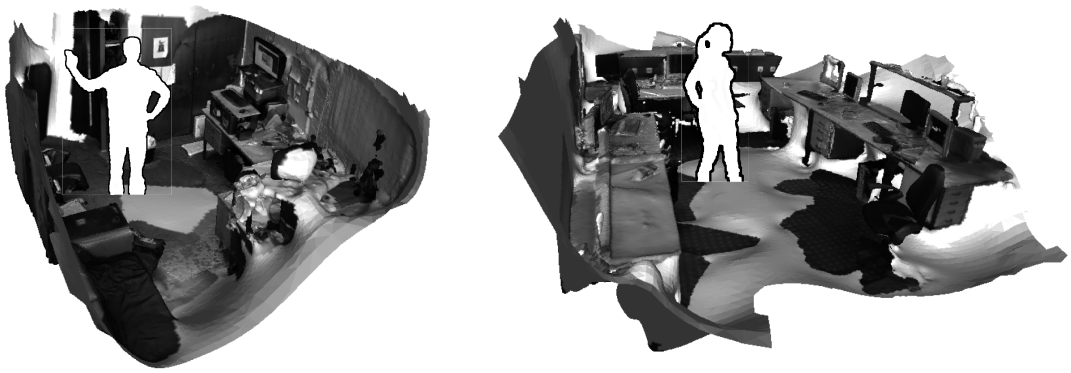


Figure 5.5: The two rooms used for the illustrative example throughout this section.

In order to illustrate the impact of the single energy terms, a simple conversation scenario between two rooms is considered. The rooms are scans of existing office spaces at the Institute for Human-Machine-Communication at the Technische Universität München. Using the "ReconstructMe" software, these offices were converted into mesh models stored in the commonly used "PLY" data format [Tur98; HK14]. These meshes were resurfaced by a screened Poisson approach [KH13] and then downsampled to the desired resolution via quadric edge collapsing [GH97].

In the example scenario, both users stand near a wall facing the middle of the rooms. Four standard Kinect cameras (version 1) observe each room, distributed evenly as to ensure good coverage of the available floorspace. The scene is illustrated in Figure 5.5. Over the course of the next few pages, this arrangement will illustrate the characteristics of different energy terms with reference to a single usage scenario. In the associated energy distribution plots, alignment parameters are varied over a range of -2 to 2 meters for the translation along the x and y axes. The rotation is altered over the entire range from $-180°$ to $+180°$. The room origins are each placed near the centre of the unobstructed floorspace.

5.3.6 The Free Floorspace Term

The partial energy term $E_{\text{free}}(\omega)$ returns high values for a maximum of shared and observable floorspace. The floor map $\mathbf{M}_{\text{FOBS}}(\omega)$ generated previously does not take the actual suitability of space for interaction into account. So there may be parts of the floor which are unobstructed, but in fact too small to use effectively. Typical examples would be small gaps between a table and the wall, or narrow passages between pieces of furniture. These should be discarded.

A morphological erosion operation on the map of observable free space effectively crops out such regions [HSZ87; Ser86]. A circular erosion kernel $\mathbf{m}_{\bigcirc}$ with a diameter of $2 \times c_{\text{UserSize}} = 1\text{m}$ is applied to the entire map. The diameter corresponds to approximately twice the average male shoulder width, as reported in recent studies [WSF11]. The resulting map $\mathbf{M}^{\text{erode}}_{\text{FOBS}}(\omega)$ will therefore only show regions which are at least large enough that two conversation partners could stand closely together:

$$\mathbf{M}^{\text{erode}}_{\text{FOBS}}(\omega) = \mathbf{M}_{\text{FOBS}}(\omega) \ominus \mathbf{m}_{\bigcirc} \tag{5.31}$$

The previous multiplication with the observability map (see Equation 5.26) has already set the map values of all unobservable regions to zero. Thus the map $\mathbf{M}^{\text{erode}}_{\text{FOBS}}(\omega)$ contains values larger than zero only for regions which are accessible and observable for both conversation partners. The energy term rewards alignments which provide a maximum of shared and observable floorspace. Since the energy term is expected to return a value ranging between zero and one, a normalising factor c_{freeNorm} contains the total weighted floor elements of the smaller room. It is therefore impossible to find any alignment which could return a higher total element-wise sum on $\mathbf{M}^{\text{erode}}_{\text{FOBS}}(\omega)$. $E_{\text{free}}(\omega)$ is calculated as follows:

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\text{A}}:$$

$$\mathbf{M}^{\text{A}}_{\text{floor}}(\mathbf{p}) = \begin{cases} 1 & \text{if } \mathbf{M}_{\text{A}}(\mathbf{p}) < c_{\text{floor}} \\ 0 & \text{else} \end{cases} \tag{5.32}$$

$$\forall \mathbf{p}(x,y) \in \mathbf{M}^{\omega}_{\text{B}}:$$

$$\mathbf{M}^{\text{B}}_{\text{floor}}(\mathbf{p},\omega) = \begin{cases} 1 & \text{if } \mathbf{M}^{\omega}_{\text{B}}(\mathbf{p}) < c_{\text{floor}} \\ 0 & \text{else} \end{cases} \tag{5.33}$$

use binary maps as follows:

$$c^{\text{A}}_{\text{freeNorm}} = \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}^{\text{A}}_{\text{floor}}} \mathbf{M}^{\text{A}}_{\text{floor}}(\mathbf{p}) \mathbf{M}^{\text{A}}_{\text{OBS}}(\mathbf{p}) \tag{5.34}$$

$$c^{\text{B}}_{\text{freeNorm}} = \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}^{\text{B}}_{\text{floor}}} \mathbf{M}^{\text{B}}_{\text{floor}}(\mathbf{p},\omega) \mathbf{M}^{\text{B}}_{\text{OBS}}(\mathbf{p},\omega) \tag{5.35}$$

$$c_{\text{freeNorm}} = \min\left(c^{\text{A}}_{\text{freeNorm}}, c^{\text{B}}_{\text{freeNorm}}\right) \tag{5.36}$$

$$E_{\text{free}}(\omega) = \frac{1}{c_{\text{freeNorm}}} \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}^{\text{erode}}_{\text{FOBS}}} \mathbf{M}^{\text{erode}}_{\text{FOBS}}(\mathbf{p},\omega) \tag{5.37}$$

As this term is computed for alignments in the example scenario, the visualisation in Figure 5.6 shows a strong central energy distribution. Due to the rather compact shape of the free floor in both rooms, the distribution is not very sensitive to rotation. On the other hand, the translation in the x and y directions has a strong impact on the shared free floor. In consequence there is a clear decline of energy values in the horizontal plane. As the distribution is plotted into a coordinate system with the rotation along the vertical axis, it takes on the shape of a rather broad and slightly twisting column.

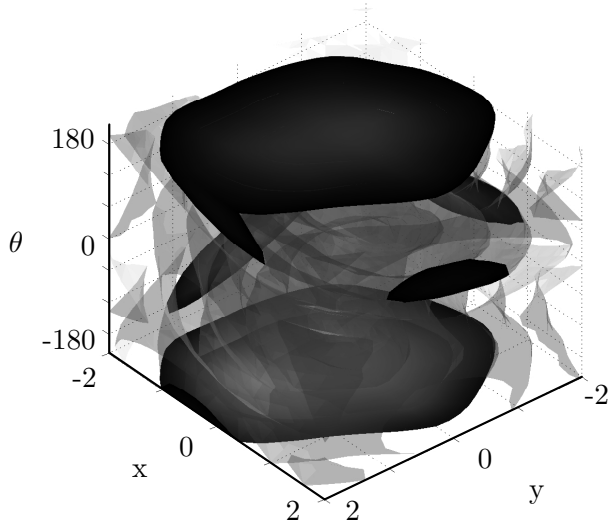


Figure 5.6: Rendering of the shared floorspace energy term applied to the scene introduced in Section 5.3.5.

5.3.7 The User Proximity Term

The term $E_{\text{prox}}(\omega)$ penalizes close initial user positions in order to prevent initialising users in the same space. This requirement is founded on inherent social norms common to most contemporary cultures: The distance at which two persons are comfortably interacting is always subject to the level of trust and intimacy between them. Scientific exploration of this topic was pioneered by Edward T. Hall [Hal63], who coined the term "proxemics" for the use of space between two or more persons. The space surrounding a person is understood to be divided into "reaction bubbles", which are subject to social norms and variable over cultures [Hal69; AVD92]. Most importantly in the context of AR presence, the violation of these norms should be avoided in order to ensure a pleasant conversation scenario. The proximity energy term therefore yields low scores for any alignment placing the users closer together than a pre-defined distance d_{min}. Hall's work [Hal69, pp. 113–125] gives the following typical distances for western cultures:

- Personal distance: $0.45\,\text{m} - 1.2\,\text{m}$
- Social distance: $1.21\,\text{m} - 3.6\,\text{m}$
- Public distance: $> 3.6\,\text{m}$

The familiarity of the users determines the distance d_{min} at which the energy term begins to drop off notably. The following are typical values for this distance depending on conversation scenario:

$$d_{\text{min}} = \begin{cases} 1.0\,\text{m} & \text{personal distance,} \\ 2.5\,\text{m} & \text{social distance,} \\ 5.0\,\text{m} & \text{public distance.} \end{cases} \tag{5.38}$$

A second constant $d_{\text{min, limit}}$ defines the absolute closest permitted distances. When an alignment places users closer together than this value, it is automatically assigned an total energy of zero by setting $\alpha_E = 0$.

$$d_{\text{min, limit}} = \begin{cases} 0.45\,\text{m} & \text{personal distance,} \\ 1.21\,\text{m} & \text{social distance,} \\ 3.6\,\text{m} & \text{public distance.} \end{cases} \tag{5.39}$$

In preparation, users' positions $\mathbf{X}_a^\omega$ and $\mathbf{X}_b^\omega$ are mapped to the 2D plane of room A. In the case of a simple 3 DoF problem, let $\mathbf{X}_a^\omega = \mathbf{X}_a$. The decay factor λ_{dmin} ensures that a proximity of $d_{\text{B}}^{\text{A}}(\omega) = d_{\text{min}}$ between users returns a score of $E_{\text{prox}}^{\text{min}}(\omega) = 0.95$, i.e. 95% of the maximum possible score for this term:

$$d_{\text{B}}^{\text{A}}(\omega) = \|\mathbf{X}_b^\omega - \mathbf{X}_a^\omega\|_{\text{euclid.}} \tag{5.40}$$

$$\lambda_{\text{dmin}} = -\log(95\%)/(d_{\text{min, limit}} - d_{\text{min}})^2 \tag{5.41}$$

$$E_{\text{prox}}^{\text{min}}(\omega) = \begin{cases} 1 - \exp\left(-\lambda_{\text{dmin}} \cdot (d_{\text{B}}^{\text{A}}(\omega) - d_{\text{min}})^2\right) & \text{if } d_{\text{B}}^{\text{A}}(\omega) > d_{\text{min}} \\ 0 \quad \text{and} \quad \alpha_E(\omega) = 0 & \text{else} \end{cases} \tag{5.42}$$

While placing users too close together may lead to social irritations, the opposite might happen as well. For scenarios with very large admissible interaction spaces, e.g. large meeting rooms, there is an additional upper limit to the distance between users.

Analogously to the minimum limit, the decay factor λ_{dmax} ensures that a distance $d_{\text{B}}^{\text{A}}(\omega) = d_{\text{max}}$ will return a score of $E_{\text{prox}}^{\text{max}}(\omega) = 0.95$. The upper threshold can be implemented simply by adding a second pass over the previously found score $E_{\text{prox}}^{\text{min}}(\omega)$:

$$\lambda_{\text{dmax}} = -\log(5\%)/(d_{\text{max, limit}} - d_{\text{max}})^2 \tag{5.43}$$

$$E_{\text{prox}}^{\text{max}}(\omega) = \begin{cases} E_{\text{prox}}^{\text{min}}(\omega) - \exp\left(-\lambda_{\text{dmax}} \cdot (d_{\text{B}}^{\text{A}}(\omega) - d_{\text{max}})^2\right) & \text{if } d_{\text{B}}^{\text{A}}(\omega) \leq d_{\text{max}} \\ 0 \quad \text{and} \quad \alpha_E(\omega) = 0 & \text{else} \end{cases} \tag{5.44}$$

$$E_{\text{prox}}(\omega) = \begin{cases} E_{\text{prox}}^{\text{max}}(\omega) & \text{if } E_{\text{prox}}^{\text{max}}(\omega) \geq 0 \\ 0 & \text{else} \end{cases} \tag{5.45}$$

In effect, the energy term shows a bandpass characteristic returning high scores for distances in the permissible range and minimums scores for any other distance.

This characteristic becomes visible as the term is applied to the example scenario: Poses placing the users too close together are shown as a connected series of minima. For the illustration in Figure 5.7, the values of minimum and maximum permissible distances were chosen to be rather close. In the outer regions, the penalty for placing users too far apart is visible as a band of minima enclosing the desired distances. The twisting structure stems from the non-centric initial placement of the users. The region of minima at the core of this structure indicates alignments placing users too close together.

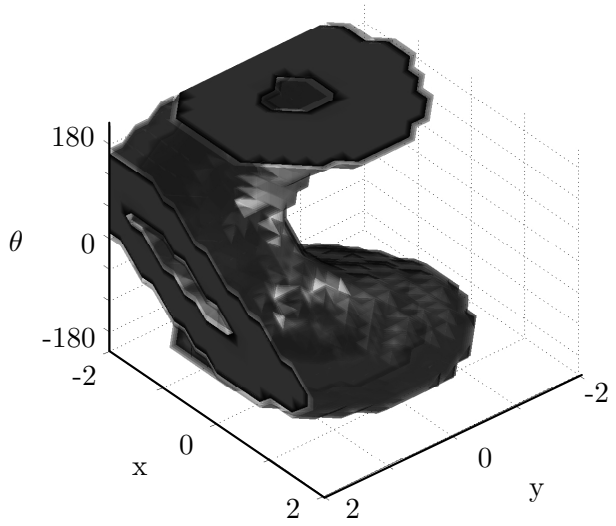


Figure 5.7: Rendering of the user proximity energy term applied to the scene introduced in Section 5.3.5.

5.3.8 The User Heading Term

In addition to research on inter-personal distances, Hall describes a practical experiment run by Sommer and himself in which they studied the effects of seating arrangements on conversations [SR58; Hal69, pp. 108 sqq.]. Hall uses the experiment to demonstrate the concepts of "sociopetal" and "sociofugal" spaces. The former denotes spatial arrangements conductive to communication and exchange, while the latter discourage interaction. In his earlier publication, "A System for the Notation of Proxemic Behavior", he supplies a specific notation for the "sociopetal-sociofugal" axis, which encodes the orientation of two persons into nine distinct relative poses [Hal63]. This notation is shown in Figure 5.8. Since then, the concept was primarily taken up in the field of architecture, environmental design research and by discourse theorists.

In the context of a meeting scenario, the initial relative orientation of users is of great importance. Initialising one person facing away from the other will most likely transport specific and unintended social cues to the person on whom the back is turned. Therefore the partial energy term $E_{\text{head}}(\omega)$ will yield high scores for alignments in which both users face each other.

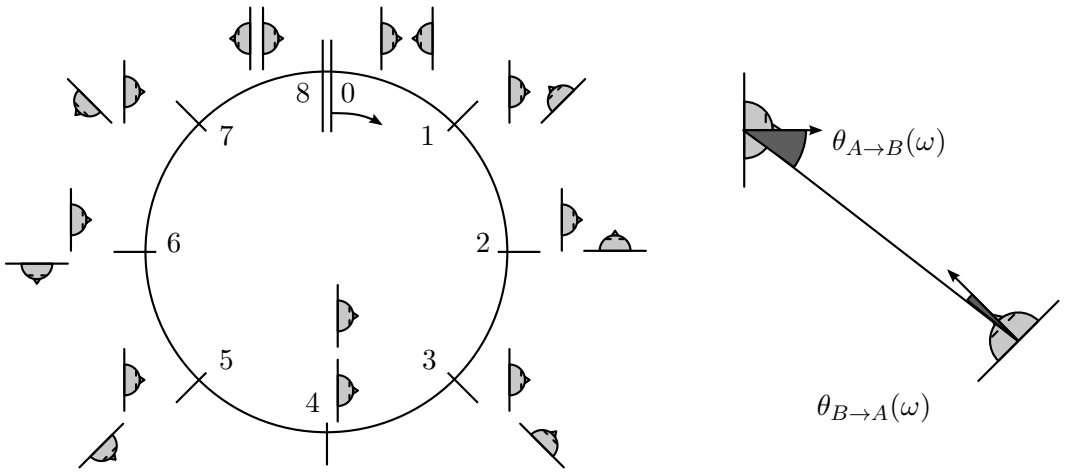


Figure 5.8: (Right) The sociopetal-sociofugal axes as defined by Hall [Hal69, pp. 108 sqq.]. Shown are the view axes of two persons. Pose 0 starts with both persons facing each other, each further step corresponds to another 45° turn. (Left) Geometric layout for computing the angles in Equations 5.49 and 5.50.

The term increases for small angles between each user's heading and the connecting vector to the other conversation partner's position. The users' 2D headings relative to the floor map of room A are denoted as $\mathbf{D}_a^\omega$ and $\mathbf{D}_b^\omega$. For the 3 DoF problem, let $\mathbf{D}_a^\omega = \mathbf{D}_a$. These headings relate to the orientation of the upper body, which is calculated from the cross-product between the shoulder axis $\tilde{\mathbf{x}}^{\mathrm{i}}_{\mathrm{shoulder}}$ and the spine axis $\tilde{\mathbf{x}}^{\mathrm{i}}_{\mathrm{spine}}$ of a given user i:

$$\forall i \in \{\text{User A}, \text{User B}\}: \\ \mathbf{D}_i = \tilde{\mathbf{x}}^{\mathrm{i}}_{\mathrm{shoulder}} \times \tilde{\mathbf{x}}^{\mathrm{i}}_{\mathrm{spine}} \tag{5.46}$$

The decay factor of the exponential is set to $\lambda_{\mathrm{head}} = -\log(5\%)/(90°)^2$. A relative angle of 90° therefore yields a partial score of $E^A_{\mathrm{head}}(\omega) = 0.05$. The relative angles from which each participant observes the other are computed separately and then added with equal weighting of each partial score. The following sequence of calculations finds the heading energy term $E_{\mathrm{head}}(\omega)$:

$$\theta_{A\to B}(\omega) = \cos^{-1}\left((\mathbf{X}_b^\omega - \mathbf{X}_a^\omega)\cdot \mathbf{D}_a^\omega\right) \tag{5.47}$$
$$\theta_{B\to A}(\omega) = \cos^{-1}\left((\mathbf{X}_a^\omega - \mathbf{X}_b^\omega)\cdot \mathbf{D}_b^\omega\right) \tag{5.48}$$
$$E^A_{\mathrm{head}}(\omega) = \exp\left(-\lambda_{\mathrm{head}}\theta_{A\to B}(\omega)^2\right) \tag{5.49}$$
$$E^B_{\mathrm{head}}(\omega) = \exp\left(-\lambda_{\mathrm{head}}\theta_{B\to A}(\omega)^2\right) \tag{5.50}$$
$$E_{\mathrm{head}}(\omega) = 0.5\cdot E^A_{\mathrm{head}}(\omega) + 0.5\cdot E^B_{\mathrm{head}}(\omega) \tag{5.51}$$

This combination of terms allows to find perfect alignments, i.e. both users standing face to face, but also rewards non-optimal solutions, such as only one user facing the other.

In cases where it is imperative to have both users face each other, the equation 5.51 can be adapted as follows:

$$E_{\text{head}}(\omega) = \begin{cases} E^A_{\text{head}}(\omega) & \text{if } E^A_{\text{head}}(\omega) \leq E^B_{\text{head}}(\omega) \\ E^B_{\text{head}}(\omega) & \text{else} \end{cases} \tag{5.52}$$

The lower user-specific score thus dominates the heading energy term and enforces a penalty if even one user is facing away from the conversation partner. This behaviour corresponds to a continuous version of the "sociofugal-sociopetal" axis encoding by Hall [Hal63].

Figure 5.9 illustrates the energy term analyses of the example scene for both variants. When using the mutual term from Equation 5.51, the energy distribution takes the shape of spiralling structure with a central maximum. The maximum appears at the optimal alignment, when both users are directly facing each other. As the rooms and respectively the users are shifted further against each other, there are alignments for which the user in room B is still facing user A. These alignments result in the spiralling band of intermediate energy scores. If the exclusive term from Equation 5.52 is used, only a small cone of alignments permits both users to stand facing each other. While there are a number of permissible translations within the floor plane, the limitation is especially pronounced along the rotation axis. Altering the rotation effectively will lead to one user turning away from the other, something this term is expressively designed to discourage. Therefore there is only a narrow scope for rotation, while simple translations in the floor plane are less constrained, as long as they do not lead to users standing back to back.

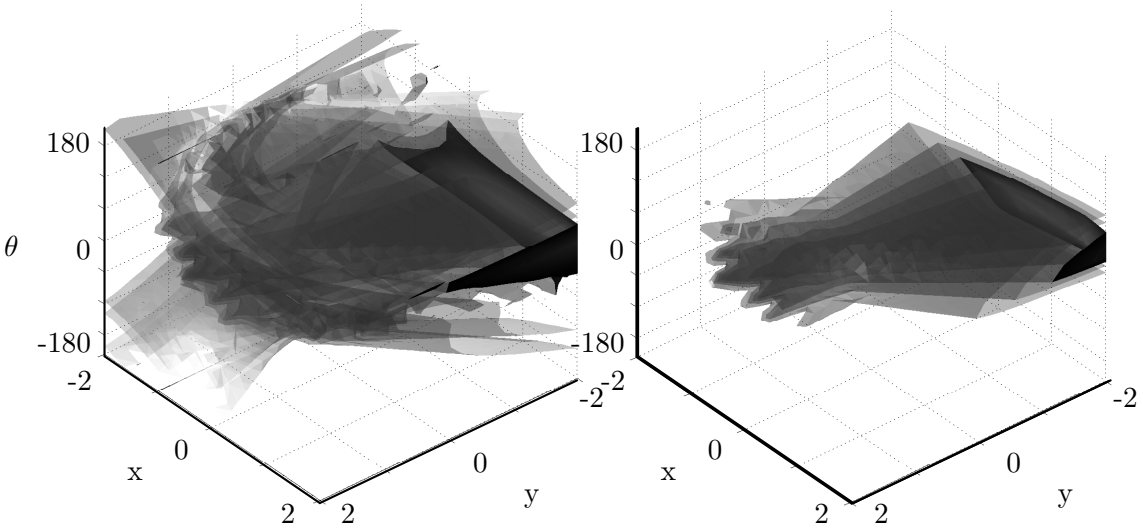


Figure 5.9: Rendering of the user heading energy term applied to the scene introduced in Section 5.3.5. (Left) Using the mean energy term from Equation 5.51, (right) using the "sociofugal-sociopetal" energy term defined in Equation 5.52.

5.3.9 The Obstacle Collision Term

In order to penalise alignments placing one or both users within obstacles, the $E_{\text{wall}}(\omega)$ term returns high values only if both users are within the free, observable consensus floorspace. Each user is considered to occupy a certain volume of the map, defined as the sub-maps $\mathbf{M}^{\text{A}}_{\text{vic}}$ and $\mathbf{M}^{\text{B}}_{\text{vic}}$. For ease of computation, the area occupied by a user is approximated as a square of $c_{\text{UserSize}} \times c_{\text{UserSize}}\ m^2$. These squares are overlaid over the map of free, observable floorspace $\mathbf{M}_{\text{FOBS}}$ and the number of non-zero map pixels are counted. If more than half of the sub-map overlaps with the free and observable space, the user is considered to be clear of obstacles. The energy term $E_{\text{wall}}(\omega)$ returns a high value of "1" only if both users are standing in unobstructed space:

$$\mathbf{M}^{\text{A}}_{\text{vic}} \subset \mathbf{M}_{\text{FOBS}} \qquad \textit{subset centered around User A}$$

$$\mathbf{M}^{\text{B}}_{\text{vic}} \subset \mathbf{M}_{\text{FOBS}} \qquad \textit{subset centered around User B}$$

$$A^{\text{A}}_{\text{free}} = \sum_{\forall \mathbf{p}(x,y)\, \in\, \mathbf{M}^{\text{A}}_{\text{vic}}} \lceil \mathbf{M}_{\text{FOBS}}(\mathbf{p}) \rceil \tag{5.53}$$

$$A^{\text{B}}_{\text{free}} = \sum_{\forall \mathbf{p}(x,y)\, \in\, \mathbf{M}^{\text{B}}_{\text{vic}}} \lceil \mathbf{M}_{\text{FOBS}}(\mathbf{p}) \rceil \tag{5.54}$$

$$E_{\text{wall}}(\omega) = \begin{cases} 1 & \text{if } 0.5\|\mathbf{M}^{\text{A}}_{\text{vic}}\| \leq A^{\text{A}}_{\text{free}} \ \wedge\ 0.5\|\mathbf{M}^{\text{B}}_{\text{vic}}\| \leq A^{\text{B}}_{\text{free}} \\ 0 & \text{else} \end{cases} \tag{5.55}$$

The binary characteristics of this energy term become clearly visible in the illustrative example shown in Figure 5.10. Any alignment placing a user within a wall will immediately lead to a low return value. As the centres of the free floorspaces are not centred within the maps, the term produces a slightly twisting shape which follows the rotational alignment.

5.3.10 The Common Work Surface Term

The term $E_{\text{surf}}(\omega)$ returns high values for large common surfaces such as tables present in both rooms. The computation is not limited to planar surfaces, but compares the height maps of both rooms for overlapping regions of similar height. The process is nearly identical to the free floorspace term $E_{\text{free}}(\omega)$, except that the map $\mathbf{M}_{\text{SOBS}}$ is used instead of $\mathbf{M}^{\text{erode}}_{\text{FOBS}}$:

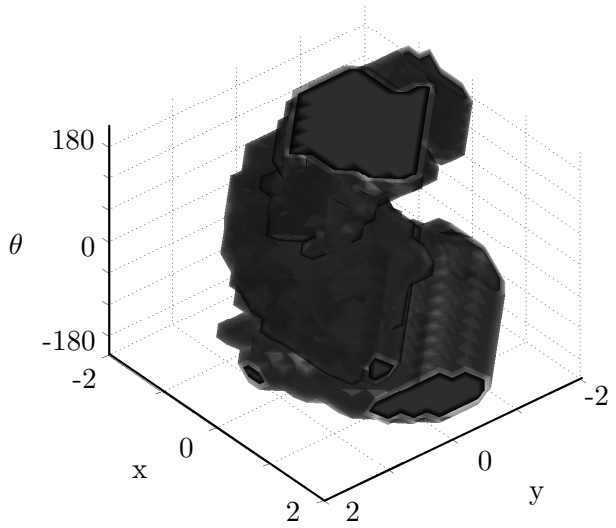


Figure 5.10: Rendering of the collision penalty term applied to the scene introduced in Section 5.3.5.

$$\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{A}}:$$

$$\mathbf{M}^{\mathrm{A}}_{\mathrm{Surface}}(\mathbf{p}) = \begin{cases} 1 & \text{if } \mathbf{M}_{\mathrm{A}}(\mathbf{p}) \geq c_{\mathrm{floor}} \wedge \mathbf{M}_{\mathrm{A}}(\mathbf{p}) < c_{\mathrm{ceil}} \\ 0 & \text{else} \end{cases} \tag{5.56}$$

$$\forall \mathbf{p}(x,y) \in \mathbf{M}^{\omega}_{\mathrm{B}}:$$

$$\mathbf{M}^{\mathrm{B}}_{\mathrm{Surface}}(\mathbf{p},\omega) = \begin{cases} 1 & \text{if } \mathbf{M}^{\omega}_{\mathrm{B}}(\mathbf{p}) \geq c_{\mathrm{floor}} \wedge \mathbf{M}^{\omega}_{\mathrm{B}}(\mathbf{p}) < c_{\mathrm{ceil}} \\ 0 & \text{else} \end{cases} \tag{5.57}$$

use binary maps as follows:

$$c^{\mathrm{A}}_{\mathrm{freeNorm}} = \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}^{\mathrm{A}}_{\mathrm{Surface}}} \mathbf{M}^{\mathrm{A}}_{\mathrm{Surface}}(\mathbf{p}) \mathbf{M}^{\mathrm{A}}_{\mathrm{OBS}}(\mathbf{p}) \tag{5.58}$$

$$c^{\mathrm{B}}_{\mathrm{freeNorm}} = \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}^{\mathrm{B}}_{\mathrm{Surface}}} \mathbf{M}^{\mathrm{B}}_{\mathrm{Surface}}(\mathbf{p},\omega) \mathbf{M}^{\mathrm{B}}_{\mathrm{OBS}}(\mathbf{p},\omega) \tag{5.59}$$

$$c_{\mathrm{freeNorm}} = \min\left(c^{\mathrm{A}}_{\mathrm{freeNorm}}, c^{\mathrm{B}}_{\mathrm{freeNorm}}\right) \tag{5.60}$$

$$E_{\mathrm{surf}}(\omega) = \frac{1}{c_{\mathrm{freeNorm}}} \sum_{\forall \mathbf{p}(x,y) \in \mathbf{M}_{\mathrm{SOBS}}} \mathbf{M}_{\mathrm{SOBS}}(\mathbf{p},\omega) \tag{5.61}$$

When the $E_{\mathrm{surf}}(\omega)$ term is computed for alignments of the example scenario, it becomes apparent that the resulting distribution is far more complex than that generated by the floorspace term. While there is a centrally located free floorspace in both rooms, the work surfaces are found on tables scattered along the walls. These surfaces are not very large, not continuously connected and irregularly shaped. As Figure 5.11 shows, a multitude of weakly bounded local maxima appears as a result.

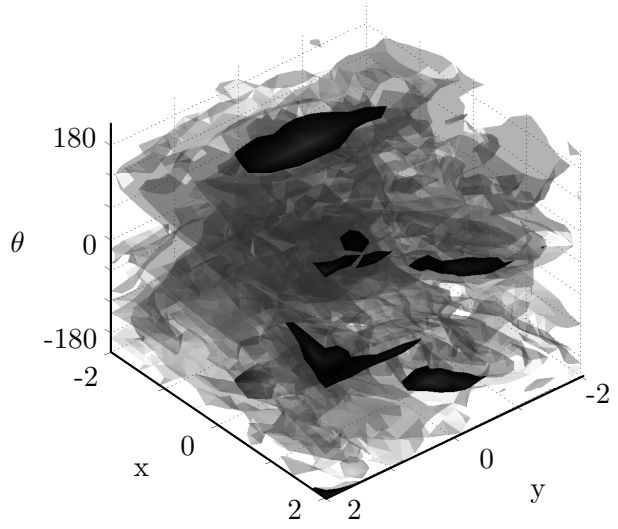


Figure 5.11: Rendering of the shared work surface energy term applied to the scene introduced in Section 5.3.5.

5.3.11 The Uninterrupted Work Surface Term

For some collaboration scenarios, the users may need a shared worksurface of a certain size. The binary energy term $E_{\text{mins}}(\omega)$ signals that a given alignment ω provides at least one surface larger than the minimum area $A_{\text{minSurface}}$. $E_{\text{mins}}(\omega)$ is set to "1" only if a sufficiently large, uninterrupted common surface is available and remains at "0" otherwise.

Since all computations take place on the maps, the minimum area $A_{\text{minSurface}}$ must be expressed in terms of pixel scaling:

$$A^{\square}_{\text{minSurface}} = \frac{c_x c_y}{|\mathbf{M}|_x \, |\mathbf{M}|_y} A_{\text{minSurface}} \tag{5.62}$$

There is already a map of candidate surfaces $\mathbf{M}_{\text{SOBS}}(\omega)$ available from the previous computation steps. This map is simply examined for connected regions of a size greater than $A^{\square}_{\text{minSurface}}$. After transforming the map to a binary format by a simple pixel-wise thresholding operation, the Teh-Chin dominant point detection algorithm [TC89] finds the unconnected regions i and their boundary polygons $\mathcal{V}^i_{\text{border}}$. Subsequently these polygons lead to the area A^i_{region} of each region using the *surveyors area formula* given by Braden [Bra86]. Based on the initial border extraction, the uninterrupted work surface energy $E_{\text{mins}}(\omega)$ examines the size of the distinct regions:

$$\texttt{BorderExtraction}\,(\mathbf{M}_{\mathrm{SOBS}}(\omega)) \Rightarrow \{\mathcal{V}^1_{\mathrm{border}} \dots \mathcal{V}^N_{\mathrm{border}}\} \tag{5.63}$$

$$\forall \mathcal{V}^i_{\mathrm{border}} \in \{\mathcal{V}^1_{\mathrm{border}} \dots \mathcal{V}^N_{\mathrm{border}}\}:$$

$$A^i_{\mathrm{region}} = \texttt{PolygonArea}(\mathcal{V}^i_{\mathrm{border}}) \tag{5.64}$$

$$E_{\mathrm{mins}}(\omega) = \begin{cases} 1 & \text{if } A^i_{\mathrm{region}} \geq A^{\square}_{\mathrm{minSurface}} \Rightarrow \texttt{finish} \\ 0 & \text{else} \end{cases} \tag{5.65}$$

The algorithm tests all found surface contours against the minimum desired size and concludes as soon as a feasible solution is found, hence the `finish` clause in equation 5.65.

The distribution which results from the application of this term to the example scenario consists of just a few local maxima. As seen in Figure 5.12, these maxima are spatially compact, with practically no gradual boundaries. It should be noted that for many room pairings, no suitable work surface might be found at all, depending on the layout of the physical rooms. If there are no suitable physical surfaces to begin with, consequently, no alignment will suffice.

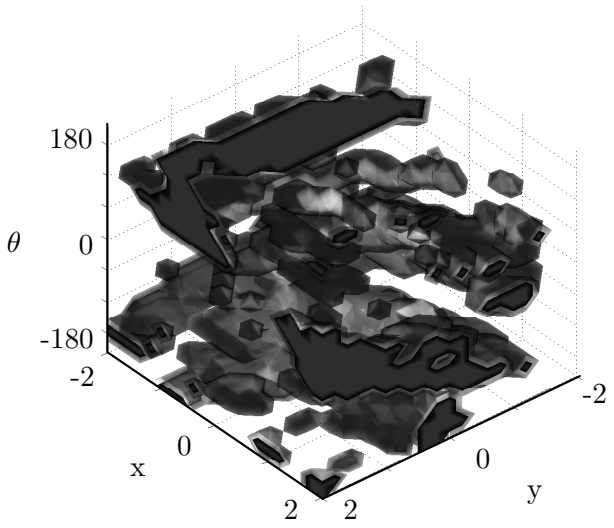


Figure 5.12: Rendering of the uninterrupted shared work surface energy term applied to the scene introduced in Section 5.3.5.

5.3.12 The Geometry Skew Term

The partial terms discussed in the previous sections are utilitarian: They ensure that participants have enough space to engage with each other, find common workspaces and do not injure their personal space. Once these basic requirements are satisfied, the geometry skew term $E_{\mathrm{skew}}(\omega)$ seeks to find aesthetically pleasing alignments by rewarding walls running in parallel.

In a first step, a Canny corner detector [Can86] extracts the wall borders from the floorspace maps of both rooms. Then a probabilistic Hough transform [MGK00] applied

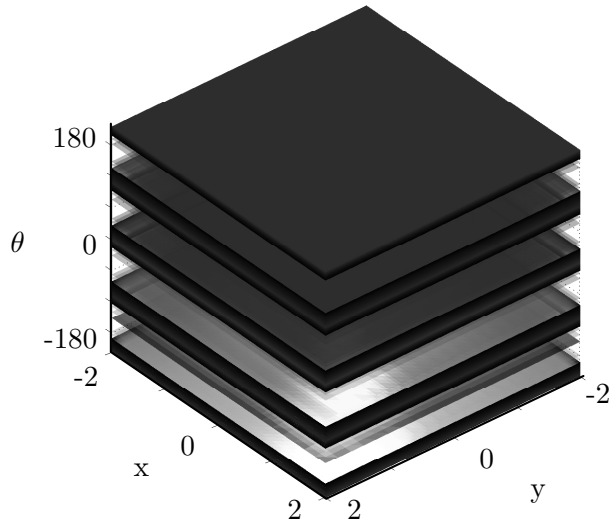


Figure 5.13: Stratified energy distribution of the skew energy term applied to the scene introduced in Section 5.3.5.

to the edges identifies major linear wall segments in room A. The transform returns a list of line segments, which are examined for their main angular direction within the floor plane. A histogram $\mathcal{H}^{A}_{\theta,\text{wall}}$ collects all these angles θ_i.

Subsequently, a function $\mathtt{f}^{i}_{\text{peak}}(\theta)$ is assigned to each histogram peak θ_i, with wrap-around terms added to account for the $180° \rightarrow 0°$ discontinuity.

The algorithm then tests for parallel alignment by generating a second angular direction histogram $\mathcal{H}^{B}_{\theta,\text{wall}}(\omega)$ from the wall map of room B. The angles θ^{B}_{i} corresponding to peaks in the histogram $\mathcal{H}^{B}_{\theta,\text{wall}}(\omega)$ are tested against all the functions $\mathtt{f}^{i}_{\text{peak}}(\theta)$ found in the wall histogram $\mathcal{H}^{A}_{\theta,\text{wall}}$ of room A. $E_{\text{skew}}(\omega)$ is computed as the normalised sum of these peak terms using a standard skew deviation of $c_{\text{skew}} = 36°$:

$$\mathtt{f}^{i}_{\text{peak}}(\theta) = \exp\left(-0.5 \cdot \left(\frac{\theta_i - \theta}{c_{\text{skew}}}\right)^2\right) \tag{5.66}$$

$$E_{\text{skew}}(\omega) = \frac{1}{N^{B}_{\text{peaks}}} \sum_{\forall \theta_B \in \mathcal{H}^{B}_{\theta,\text{wall}}} \max_i \left(\mathtt{f}^{i}_{\text{peak}}(\theta)\right) \tag{5.67}$$

The direction in which the walls are running is influenced only by the rotation of the two floor maps, but not by the x/y offsets contained in the alignment ω. It is therefore easy to pre-compute a look-up table (LUT) for this energy term for use in the later optimisation stage.

In the example scenario this rotation dependence is clearly visible in the stratified structure of the energy distribution in the alignment space. As the rotation term is mapped along the vertical axis, bands of maxima appear when one or more wall sections are brought into parallel alignment with wall segments of the other room. This effect

is shown in Figure 5.13, where the 90° spacing of the maxima reveals the perpendicular alignment of the walls. Note that these maxima are independent from the translation of the alignment.

5.4 Adapting the Weighting Factors

The energy function $E_{\text{total}}(\omega)$ uses a number of weighting factors which govern the relative influence of the single energy terms. These weighting factors are applied in the overall summation of energy terms in Equation 5.28. The setting of these factors depends on the size of the rooms connected by the AR videoconferencing system. Especially for smaller rooms, there are few alignments possible which do not place one user inside a wall or outside of observability. In such cases, it becomes necessary to focus the optimisation on a smaller set of criteria. The weighting factors are then set to small values or even zero for partial energy terms which are most likely not satisfiable. In return, the essential criteria receive higher relative weights. Table 5.1 gives some typical settings for conversation scenarios for rooms of various sizes. If the participating rooms differ in size, the smaller room dictates the constraints on weighting factors.

Table 5.1: Typical weighting factors for conversation scenario

Room	α_{free}	α_{prox}	α_{wall}	α_{head}	α_{surf}	α_{mins}	α_{skew}
$< 5\ m^2$	1.0	0.2	0.5	0	0.5	0	0
$< 10\ m^2$	1.0	0.6	1.0	0.6	0.5	0	0
$> 10\ m^2$	1.0	0.6	1.0	0.6	0.75	0.5	0.5

5.5 Solving the Optimisation Problem

Having defined the energy function, finding a suitable global optimisation approach poses the next challenge. The solution space is a combination of the partial energy terms described in the previous sections. Properties encountered in one term generally propagate to the joint energy distribution. Especially considering the binary nature of the uninterrupted work surface term α_{mins} described in Section 5.3.11, it becomes obvious that the energy distribution is not differentiable. Conventional gradient-based approaches are thereby bound to fail. Additionally, the work surface term $E_{\text{surf}}(\omega)$ has shown a strong tendency to yield a high number of local extrema further complicating the optimisation problem. In the following, we shall consider the suitability of some standard global solvers used for such non-linear, non-convex, non-differentiable solution spaces. Please note that while the energy distribution is non-continuous in the sense of showing discontinuities, the values of the final alignment are *per se* continuous, i.e. not discrete.

5.5.1 Solving the 3 DoF Problem

Introduced in Section 5.3.3, this case assumes fixed user positions and encodes the alignment in ω_3. As only 3 DoF are considered, a wide range of different approaches can be

applied to the problem. Since the constraints are already encoded in the energy function $E_{\text{total}}(\omega)$, no additional equality or inequality constraints need to be formulated. However, all solvers should apply an upper and lower bound on the alignment terms contained in ω_3. The translation terms can be bounded to 50 % of the size of the smaller room, both in x and y direction. Any larger displacement leads to insufficient unobstructed floorspace, provided that the maps are approximately centred on the middle of the rooms. Rotation can be constrained to the range of 0° to 360°[6].

5.5.2 Solving the 9 DoF Problem

This case allows free placement of users within the aligned geometry of the two rooms, as discussed in Section 5.3.3. In order to reduce the computational complexity, the 9 DoF problem can be split into two more tractable sub-problems:

1. Find the optimal alignment for the origins of both rooms. Optimise only for static properties of the rooms.
2. Within the previously found alignment, find the optimal user placement.

The first task leads to a 3 DoF problem similar to the one discussed in the previous section. In this stage, only the geometric properties of the involved rooms are considered. Meanwhile the energy terms relevant to user placement are disregarded:

$$0 \leq \alpha_{\text{head}}^{\text{room}} \leq 1 \tag{5.68}$$
$$0 \leq \alpha_{\text{prox}}^{\text{room}} \leq 1 \tag{5.69}$$
$$0 \leq \alpha_{\text{wall}}^{\text{room}} \leq 1 \tag{5.70}$$
$$\alpha_{\text{free}}^{\text{user}} = 0 \tag{5.71}$$
$$\alpha_{\text{surf}}^{\text{user}} = 0 \tag{5.72}$$
$$\alpha_{\text{mins}}^{\text{user}} = 0 \tag{5.73}$$
$$\alpha_{\text{skew}}^{\text{user}} = 0 \tag{5.74}$$
$$\omega_9^{\text{room}} = (x, y, \theta) \in \mathbb{R}^3 \tag{5.75}$$
$$\hat{\omega}_9^{\text{room}} = \text{argmax}_{\omega}^{3\text{ DoF}}\ E_{\text{total}}(\omega) \tag{5.76}$$

Once the relative alignment of the floor plane origins is fixed, a second optimisation step places the users. In order to reduce complexity, we assume that a user's initial view direction should always be fixed on the conversation partner. This simple assumption constrains the two degrees of freedom representing the user rotations, θ_{A} and θ_{B}. For the optimisation, all terms related to room geometry are disregarded. The optimisation problem is therefore reduced to 4 DoF.

[6]Note that some implementations of the optimisation functions might not be able to deal with the 0° ⇔ 360° wrapping. A simple workaround would be to extend the permissible range in one or two multiples and then add an additional operation after the optimisation to constrain the result to the 0° to 360° range.

$$\alpha_{\text{head}}^{\text{room}} = 0 \tag{5.77}$$
$$\alpha_{\text{prox}}^{\text{room}} = 0 \tag{5.78}$$
$$\alpha_{\text{wall}}^{\text{room}} = 0 \tag{5.79}$$
$$0 \leq \alpha_{\text{free}}^{\text{user}} \leq 1 \tag{5.80}$$
$$0 \leq \alpha_{\text{surf}}^{\text{user}} \leq 1 \tag{5.81}$$
$$0 \leq \alpha_{\text{mins}}^{\text{user}} \leq 1 \tag{5.82}$$
$$0 \leq \alpha_{\text{skew}}^{\text{user}} \leq 1 \tag{5.83}$$
$$\omega_9^{\text{user}} = (x_{\text{A}}, y_{\text{A}}, x_{\text{B}}, y_{\text{B}}) \in \mathbb{R}^4 \tag{5.84}$$
$$\hat{\omega}_9^{\text{user}} = \text{argmax}_{\omega}^{4\ \text{DoF}}\ E_{\text{total}}(\omega) \tag{5.85}$$

The parameters found in the two optimisation stages then yield the solution to the original 9 DoF problem. The user rotations θ_{A} and θ_{B} are calculated directly from the recommended positions using the four-quadrant inverse tangent function atan2:

$$\theta_{\text{A}} = \text{atan2}\,(x_{\text{B}} - x_{\text{A}}\ ,\ y_{\text{B}} - y_{\text{A}}) \tag{5.86}$$
$$\theta_{\text{B}} = \text{atan2}\,(x_{\text{A}} - x_{\text{B}}\ ,\ y_{\text{A}} - y_{\text{B}}) \tag{5.87}$$

Thus, the recommended initial user positions will always have the users facing each other. As for the 3 DoF problem, there are no additional equality or inequality constraints to consider. Upper and lower limits for the geometric alignment apply in the same way. User positions are constrained to the range of available free floorspace. These bounds on translation are not to be confused with the constraints and score penalties applying to placement in walls or obstacles. The idea is merely to limit the range limits to the general dimensions of the actual room. It would not make sense to permit user translation by hundreds of meters if the actual room has only 30 square meters.

Most solvers require an initial pose guess for the user positions. This guess is provided using a fixed grid over the aligned common free floorspace found in the first stage of the optimisation. All grid nodes not situated within free floorspace are dismissed right away. The initialisation then divides the floor into two regions. The dividing border runs through the floor centroid in the mapping plane and lies perpendicular to the primary floor orientation. The image moments of the binary free floor map provide the centroid and floor orientation [Hu62]. User A's initial pose is assumed to be the valid grid node closest to the centre of the first region. The same method places user B in the second region. Thus, both users' initial positions are some distance apart in unobstructed floorspace as shown in Figure 5.14. The solver algorithm will then try to improve on this initial pose.

5.5.3 Brute Force Solver

The topography of the solution space does not affect the performance of a brute force solver[7], provided that it performs at a sufficiently precise resolution. If the evaluation steps

[7]Note that a brute force solver is simply an *exhaustive search* over a predefined range of the solution space using regular step widths.

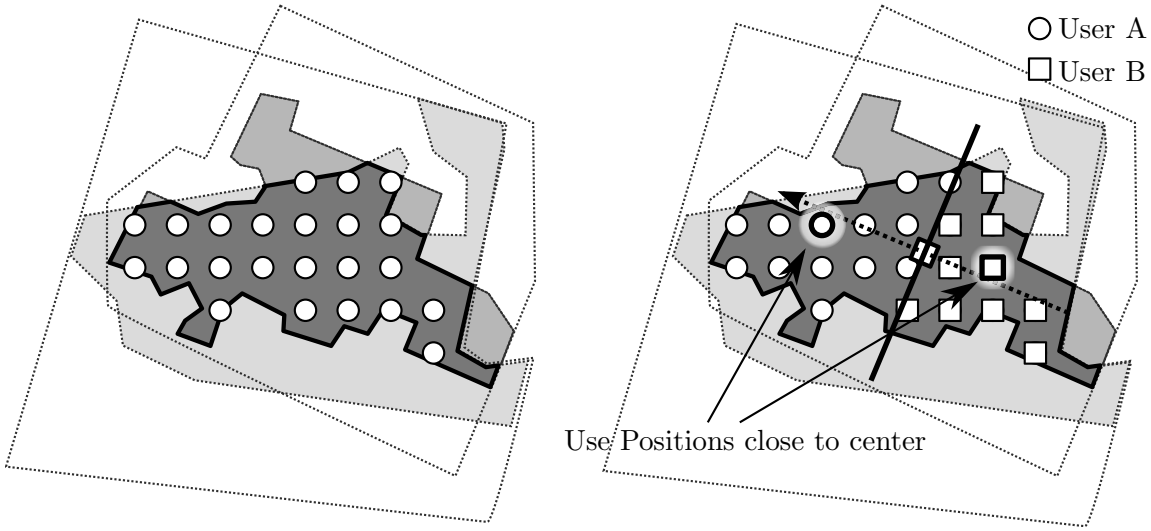


Figure 5.14: Determination of initial user positions. Shown in light gray are the original free floor areas, the darker gray area bounded in black lines shows the consensus free floor.

are chosen too large, the search will miss maxima especially for complex floorplans. Since the 3 DoF solution space is three-dimensional, the complexity becomes $\mathcal{O}(n^3)$ (assuming proportional resolution along each DoF). Therefore a balance between the grid size on which alignments are tested and the overall computation time must be found.

For the 9 DoF scenario the complexity would rise to $\mathcal{O}(n^7)$ (assuming heading constraints and proportional resolution along each DoF). Dividing the optimisation into two subsequent steps, i.e. first aligning the room geometry and then placing the users, alleviates the problem only slightly.

Due to its high computational cost, the brute force approach is useful only as a reference against which more efficient algorithms are tested. For the 3 DoF scenario, this chapter uses exhaustive search to visualize the solution space. The data gathered also serves as the reference for the evaluation of the 3 DoF problem.

This approach is implemented easily using nested loops in combination with parallel processing. The reference solution for the 3 DoF problem translates the rooms from -2 m up to $+2$ m in 0.2 m steps and varies rotation from $-180°$ to $+180°$ in $12°$ steps[8].

5.5.4 Simulated Annealing

Simulated annealing is an established and widely used stochastic solver. There are implementations available for most common programming languages. This dissertation uses the implementation shipped with the Matlab Global Optimisation Toolbox (Matlab 2013a, 8.1.0.604). For adjusting the temperature parameter T, both the Boltzmann cooling schedule ($T = T_0/\log(k_B)$ with annealing parameter k_B) and an abbreviated cooling schedule (i.e. $T = T_0/k_B$ with annealing parameter k_B) are evaluated.

[8]Table A.1 in Appendix A gives the parameters used for generating the reference solution of the 3 DoF problem.

The SA imposes bounds from -2 m up to $+2$ m on the translation and constrains rotation to the range from $-180°$ to $+180°$[9].

5.5.5 Pattern Search

The experiments in the following sections apply the GPS, GSS and MADS methods to the room alignment problem. For sake of comparability, the implementations included in the Matlab Global Optimisation Toolbox (again using Matlab 2013a, 8.1.0.604) are used.

Table A.3 lists the parameters shared by all three approaches, such as bounds on displacement. All three algorithms derive their search patterns from a $2N$ positive basis. Although there is a more efficient $N + 1$ positive basis, the larger basis was favoured due to the potentially high number of local maxima [10]. For the same reason, complete polling of the current iterate was enforced instead of a greedy search strategy. At the cost of efficiency and speed, this reduces the risk of premature convergence in a local maximum.

As for the SA approach, all pattern search algorithms impose bounds from -2 m up to $+2$ m on the translation and limit rotation to the range from $-180°$ to $+180°$[11]. The parametrization of GPS and GSS algorithms follows the literature values given by Torczon *et al.* [Tor97][12]. Similarly, the MADS solver follows Audet's and Dennis's parametrization [AD06][13].

5.5.6 Relaxation of Constraints

There are many possible scenarios where the approaches outlined above will not find an alignment without violating one or more of the constraints on initial user placement. This observation holds especially for the 3 DoF alignment, where the user positions are fixed. Examples include users facing a wall at the time of initialisation, rooms which are simply too small or cramped and finally too demanding parameters set by the users.

A relaxation of the constraints leads to alternative alignments despite unfavourable starting conditions. Once the optimisation signals a failure to converge at a viable solution, the optimisation step is restarted with relaxed constraints. Since unfavourable user positions are the cause for most failures to find an alignment, a typical relaxation sequence might progress as follows:

1. Allow placement of users in areas occupied by low furniture, e.g. a sofa or a chair.
2. Allow placement of users 10% closer than the previous minimum distance.
3. Allow placement of users in areas occupied by higher furniture.
4. Allow placement of users in areas up to one meter behind walls.

[9] Table A.2 in Appendix A shows the full parameters used for the evaluation.

[10] As a reminder: In this context N denotes the number of independent variables, i.e. the degrees of freedom. The smaller $N + 1$ basis has a higher potential for getting caught in local maxima, which is an unfavourable property in scenarios with plenty of local extrema.

[11] Table A.3 in Appendix A shows the full parameters used for the evaluation.

[12] These are also found in Table A.4 in Appendix A.

[13] The values are summarized in Table A.5 in Appendix A.

5. Drop constraints entirely.

For each new step in this sequence, the optimisation is run again. If a solution is found, it is presented to the participants for approval. Otherwise, the algorithm relaxes the next constraint and repeats the alignment procedure. Please note that constraint relaxation might lead to problems in rendering the remote avatars, as they might be initialised behind walls, etc.

In cases where even constraint relaxation fails, the 9 DoF initialisation offers a solution. In this case, the system rejects the current user positions as infeasible and instead asks the users to move to more favourable positions.

5.6 Evaluation of Solver Performance

The lack of a fixed ground truth poses a problem when comparing the performance of different solvers. There is no universal scale on which the quality of room alignments for AR videoconferencing could be graded. There are also no previous publications tackling this problem which might provide guidance in this matter[14]. By necessity, any evaluation can only happen within the framework developed in the previous sections, i.e. using the energy function itself as a reference.

For the 3 DoF optimisation, it is possible to compare the solver performance directly to a pre-generated reference map of energy values found with an exhaustive brute-force search. The alignments found by the various solvers can be compared by two criteria: Firstly, the *Overall alignment score* reflects the return values of the energy function $E_{\text{total}}(\omega)$. Secondly, the *normalised distance* uses a Euclidean distance measure between the alignment found by the solver and the result found by the exhaustive search. The normalisation accounts for the different scaling and units of the decision variables, casting the normalisation over the permitted value range.

Due to the polynomial complexity and the wide envelope of permissible user positions, such a comparison to a reference solution is not possible for the 9 DoF problem. However, the results of the first stage of alignment can be compared to the result of an exhaustive search, since the geometric alignment is a 3 DoF sub-problem.

For both cases, the number of function calls is a good metric for evaluating the efficiency of the solver on this particular problem. While actual computation time depends on the implementation and hardware used, the overall number of function calls should remain the same provided similar solvers are used.

A further important metric is the success rate of the user placements for each algorithm. In the 3 DoF problem, the user positions are fixed to their room reference frame. Especially for smaller and cluttered rooms, the solver might fail to find a legal alignment.

[14] At the time of writing, the IEEE and ACM directories contain no previous publication on this matter. Inquiries directed at other research groups active in remote AR collaboration failed to turn up any relevant previous work.

5.6.1 Establishing a Baseline

For the 3 DoF problem, the brute-force solver discussed in Section 5.5.3 creates a reference map of $E_{\text{total}}(\omega)$ over the three-dimensional solution space. This map can be used to verify the accuracy of the evaluated solvers on a set of pre-defined scenarios. While the quality of the test depends on the resolution of the brute force solver, it is helpful for assessing the overall suitability of different solvers to the problem of automatic room alignment. For visualisation and comparison, the results of the solvers are normalised against the optimum energy value found by the reference mapping. Since the solvers perform at adaptable resolutions of the solution space, they can find alignments which actually return higher $E_{\text{total}}(\omega)$ values than the reference solution. The adaptive resolution leads to instances where the score exceeds the normalised $[0, 1]$ range defined from the reference mapping.

Unfortunately, the 9 DoF problem is not tractable using a brute force approach[15]. In consequence, there is no reference mapping of $E_{\text{total}}(\omega)$ against which other solvers could be compared. However, the geometric mapping alone constitutes only a 3 DoF problem which can be tackled using a brute force approach. Consequently, the evaluation is split into two tests: The accuracy of the room layout matching is tested against a reference mapping created from a brute-force solver. This comparison corresponds to a 3 DoF problem.

For the evaluation, the four solvers were used to align two rooms for an AR videoconference scenario. In the following, these rooms are denoted room A and room B. Each of the two rooms was available in two sizes (large and small) and with cluttered and uncluttered furnishings. The rooms were generated synthetically in a CAD program. For all rooms, the coordinate system origin was placed in the middle of the room. Figure 5.15 shows the layout of each room used for these experiments. Users for room A were placed on the left side of the room, facing towards the middle, while users in room B were placed on the opposite side, also facing the middle. Four cameras observe each room. The camera parameters are equivalent to first generation Kinect units. The cameras are placed in the corners and face the centre of the room, providing full coverage of the floor plane.

Each instance of room A (small-uncluttered, small-cluttered, large-uncluttered, large-cluttered) is therefore aligned to each instance of room B (again four configurations), resulting in 16 possible combinations.

Table 5.2: Room dimensions used in the evaluation procedure.

Room A		Room B	
Small Size	$4 \times 3m$	Small Size	$3 \times 3m$
Large Size	$6 \times 4m$	Large Size	$5 \times 3m$

In order to examine solver sensitivity to initial alignment, the starting values of the floor plane alignment were varied systematically. The various values for initial alignment

[15] Aiming for the same resolution as in the 3 DoF scenario and using heading constraints on user positioning, the computation of the reference map would require $20^7 = 1.28 \times 10^9$ function evaluations. In addition, the $E_{\text{prox}}(\omega)$ term allows for a range of equally admissible solutions in this case, complicating comparisons.

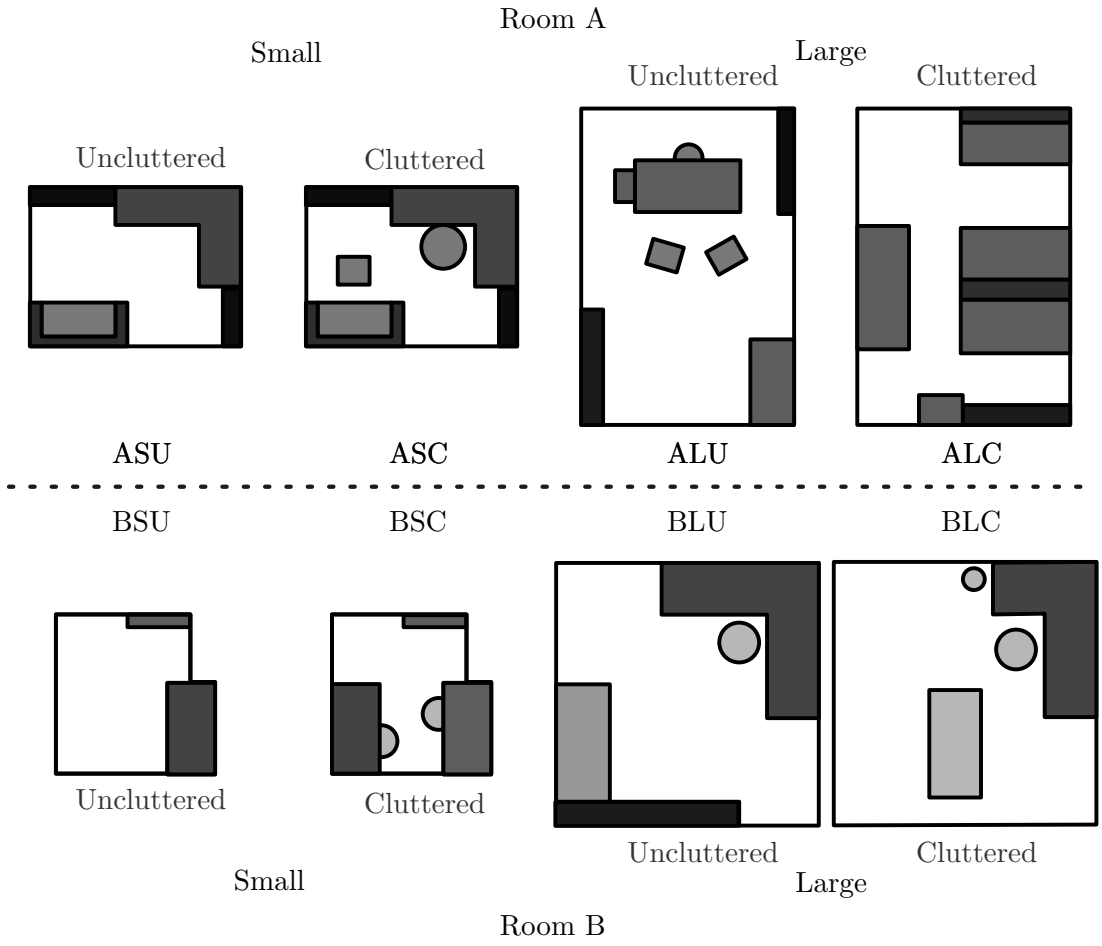


Figure 5.15: Synthetic rooms used for solver evaluation. The three letter abbreviations are composed of room (A or B), size (S - small, L - large) and clutter (C - cluttered, U - uncluttered)

.

can be found in Table 5.3. In total, for each of the 16 possible room combinations, 27 different initial conditions are tested.

Table 5.3: Initial values used for floor plane alignment.

Variable	Initial Values	Unit
x_0	$\{-1.5, 0, 1.5\}$	m
y_0	$\{-1.5, 0, 1.5\}$	m
θ_0	$\{-90, 0, 90\}$	degrees

The energy function weighting factors for the evaluation are based on values found in Table 5.1 for medium-sized rooms ($5\ m^2 \leq A_{\text{room}} < 10\ m^2$). The highest weights of 1.0 are

assigned to free floorspace $E_{\text{free}}(\omega)$ and wall collision avoidance $E_{\text{wall}}(\omega)$. The energy terms for proximity $E_{\text{prox}}(\omega)$, initial heading $E_{\text{head}}(\omega)$ and work surface $E_{\text{surf}}(\omega)$ receive lower weights of 0.6 to 0.5. The geometry skew term $E_{\text{skew}}(\omega)$ and the uninterrupted work surface term $E_{\text{mins}}(\omega)$ receive weights of zero and are therefore not part of this evaluation. The closest permitted user proximity is set to 0.4 m, which presumes a personal conversation[16].

5.6.2 Statistical Tests used for Analysis

Before applying further statistical tests on the result, a *Lilliefors test* is used to test all values for normality of the distributions [Lil67]. Since all gathered data shows strong skew, analysis of variance (ANOVA) testing is not permissible [Mil97]. Instead, the *Kruskal-Wallis analysis* tested the collected distributions for statistical significance [KW52]. Independence between single conditions was then established using *Bonferroni-corrected pairwise Wilcoxon* tests [Mil81; MW47]. The complete results are gathered in Appendix A. The following sections provide a concise summary of the findings.

5.6.3 Comparing Results for 3 DoF

Table 5.4: Percentages of legal user placements and user misplacements, 3DoF. Note that both users can be misplaced at the same time, leading to row sums in excess of 100 %.

Type	Legal	User A error	User B error	Too Close
SA	50 %	50 %	44.90 %	0 %
GPS	42.59 %	54.40 %	51.16 %	0.46 %
GSS	42.13 %	55.09 %	51.62 %	0.46 %
MADS	45.37 %	52.31 %	48.15 %	0.69 %
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$			

When comparing the performance of the four solvers, a strong difference between the SA solver and the three pattern search solvers becomes apparent. As shown in Figure 5.16, the SA approach leads to results with similarly high energy values as found by the brute force ground truth mapping. Although the median percentages are comparable[17], the three pattern search solvers generally show a higher tendency towards negative outliers.

The higher precision of the SA approach is also reflected in the normalised distance to the best solution found by the brute-force solver. While most alignments returned by SA are quite close to the reference solution, the results found by the pattern search algorithms are spread further. This spread is reflected both by higher median distances[18] and greater ranges between the lower and upper quartiles for the pattern search approaches. The

[16]Table A.6 in Appendix A also shows the parameters used for computing $E_{\text{total}}(\omega)$ throughout the remainder of the evaluation.

[17]Median score percentage of best reference value: 98.3 % for SA, 95.2 % for GPS, 94.7 % for GSS and 96.1 % for MADS. Full analysis in Table A.7 in Appendix A.

[18]Median unit distance to best reference value: 0.11 for SA, 0.20 for GPS, 0.21 for GSS and 0.19 for MADS. Full analysis in Table A.8 in Appendix A.

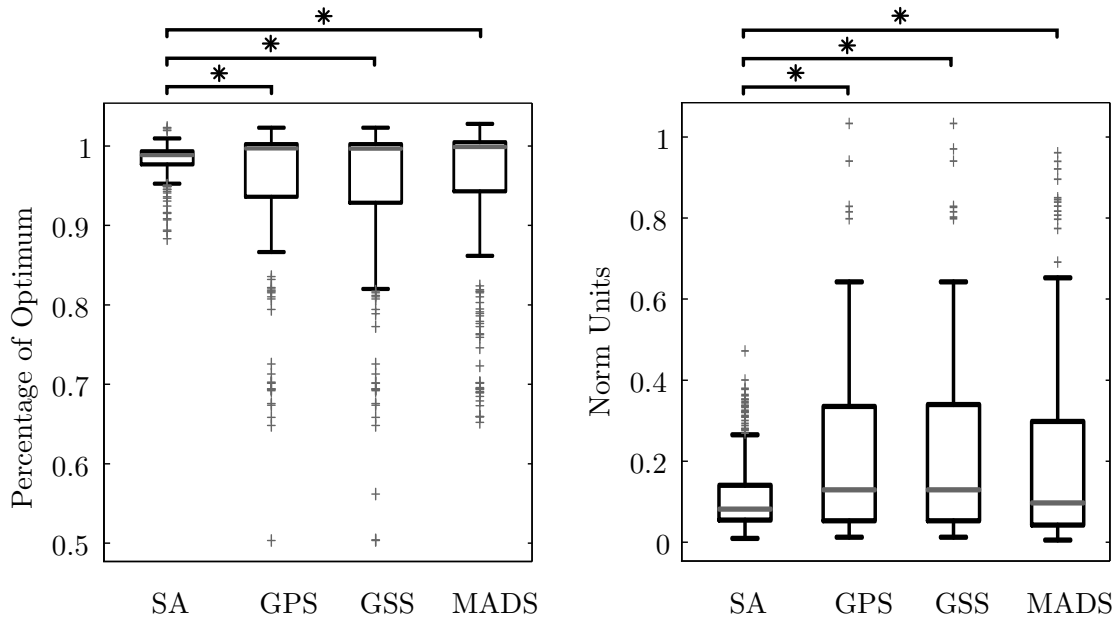


Figure 5.16: Results of the 3 DoF alignment for different optimisation algorithms. (Left) Percentage of best alignment found with reference mapping. (Right) Distance from best alignment in normalized units.

number of user misplacements summarised in Table 5.4 shows a similar pattern. Pattern search leads to more misplacements of users in furniture than simulated annealing. Such constraint violations lead to users placed in walls or too close together. Both cases should be avoided. They arise especially for alignments between the cluttered rooms, as free floorspace is severely limited and users are fixed to their initial positions. This effect is clearly visible when inspecting the count of successful alignments for the cluttered versions of room A (denoted ALC and ASC in Table 5.5). For these two rooms, none of the solvers could find a solution which did not misplace at least one user — even though the initial user placement was in unobstructed and observable space. These failures of the algorithm also underscore the difficulty when setting up a consensus reality meeting and motivate the step to the more flexible 9 DoF alignment.

Within the class of pattern search algorithms, GPS, GSS and MADS performance is virtually indistinguishable for the 3 DoF case. There are no significant differences for the energy returned or the distance to the reference solution. However, the MADS method causes fewer user placement violations than GPS and GSS. The price for this improvement is a higher number of function calls. As Figure 5.17 shows, the MADS optimisation is significantly more costly in terms of function calls than GPS and GSS[19]. Meanwhile, there is no apparent difference in function calls between GPS and GSS. Even though GSS should in theory perform better in the presence of boundaries, the convergence pattern apparently did approach these limits.

[19]Median function calls for complete alignment: 2666 for SA, 384 for GPS, 376 for GSS and 786 for MADS. Full analysis in Table A.9 in Appendix A.

Table 5.5: Percentages of legal user placements by room pairings, 3DoF.

Samples	$3 \times 3 \times 3$ starting positions per pair			
All Methods				
	BSC	BSU	BLC	BLU
ASC	0%	0%	0%	0%
ALC	0%	0%	0%	0%
Simulated annealing (SA)				
	BSC	BSU	BLC	BLU
ASU	100%	100%	100%	100%
ALU	100%	100%	100%	100%
Generalized pattern search (GPS)				
	BSC	BSU	BLC	BLU
ASU	74.0741%	74.0741%	100%	59.2593%
ALU	85.1852%	96.2963%	100%	96.2963%
Generating set search (GSS)				
	BSC	BSU	BLC	BLU
ASU	74.0741%	70.3704%	100%	59.2593%
ALU	77.7778%	96.2963%	100%	96.2964%
Mesh adaptive search (MADS)				
	BSC	BSU	BLC	BLU
ASU	85.1852%	74.0741%	100%	77.7778%
ALU	92.5926%	96.2963%	100%	100%

Comparing the number of function calls, the main drawback of SA becomes apparent. In order to achieve its advantage in quality and user placement, the method requires an order of magnitude more function calls than the pattern search algorithms. This high cost severely hinders its implementation in scenarios where speed or limited computational budget are factors.

In a separate test, a faster annealing schedule was used on the same data. However, as can be seen in Figure 5.18, this leads to greatly reduced precision. Interestingly, the faster cooling schedule does not even reduce function calls significantly. The faster cooling optimisation frequently gets caught in local extrema, wasting iterations on the exploration of the local energy topography. Triggered by a random step to a lower energy state[20], the optimisation escapes from this local extremum after a while and proceeds towards the global optimum. In result, the overall number of energy function calls is actually slightly higher than for the slower cooling schedule. So despite a "faster" cooling schedule, the process actually takes longer to complete.

[20]These random steps to less desirable states are characteristic for the Metropolis algorithm, as described in Section 3.3.

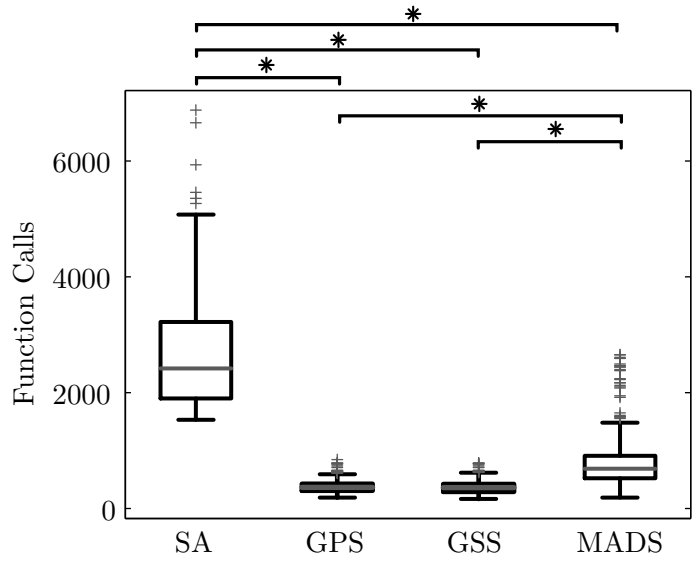


Figure 5.17: Total function calls during the 3 DoF alignment for different optimisation algorithms.

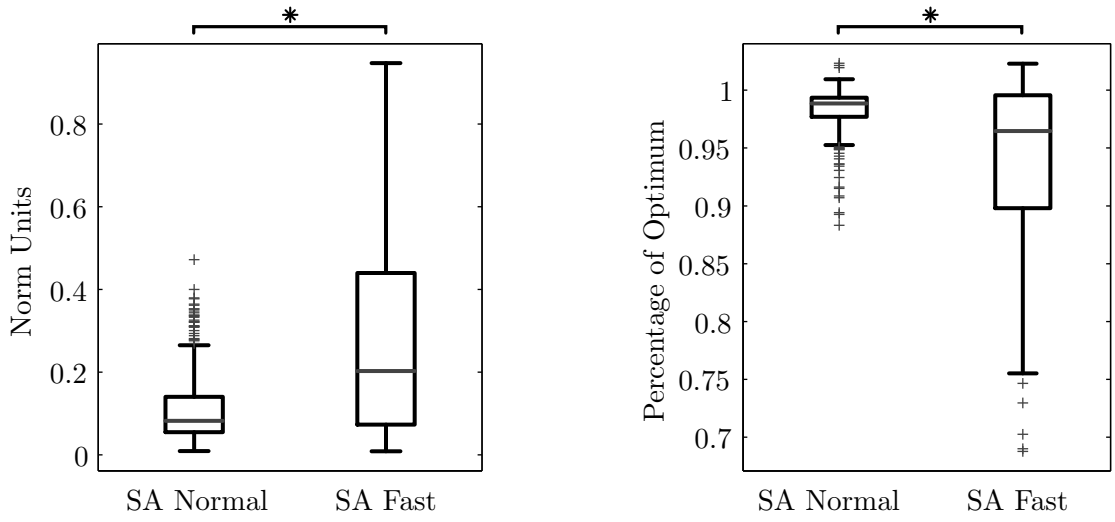


Figure 5.18: Comparison of SA performance between the Boltzmann cooling schedule $T = T_0/\log(k_B)$ and fast cooling schedule $T = T_0/k_B$ in the 3DoF scenario.

5.6.4 Conclusion for 3 DoF

In return for its high computational cost, the SA approach outperforms the pattern search approaches in all other metrics. In those rooms where a conflict-free solution is possible, it finds a legal alignment for every initial pose tested. The final energy scores are consistently higher than those found by pattern search and the resulting poses are closest to the

reference solution. Regrettably, a faster cooling schedule does not accelerate convergence and impacts the precision negatively.

In cases where speed is an issue, the MADS method offers similar performance at 70% reduced function calls. While there are even greater reductions for using GPS and GSS, these come at the price of increased user placement failures.

5.6.5 Comparing Results for 9 DoF

Table 5.6: Percentages of legal user placements and user misplacements, 9DoF. %.

Type	Legal	User A error	User B error	Too Close
SA	100.0 %	0.0 %	0.0 %	0.0 %
GPS	100.0 %	0.0 %	0.0 %	0.0 %
GSS	100.0 %	0.0 %	0.0 %	0.0 %
MADS	100.0 %	0.0 %	0.0 %	0.0 %
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$			

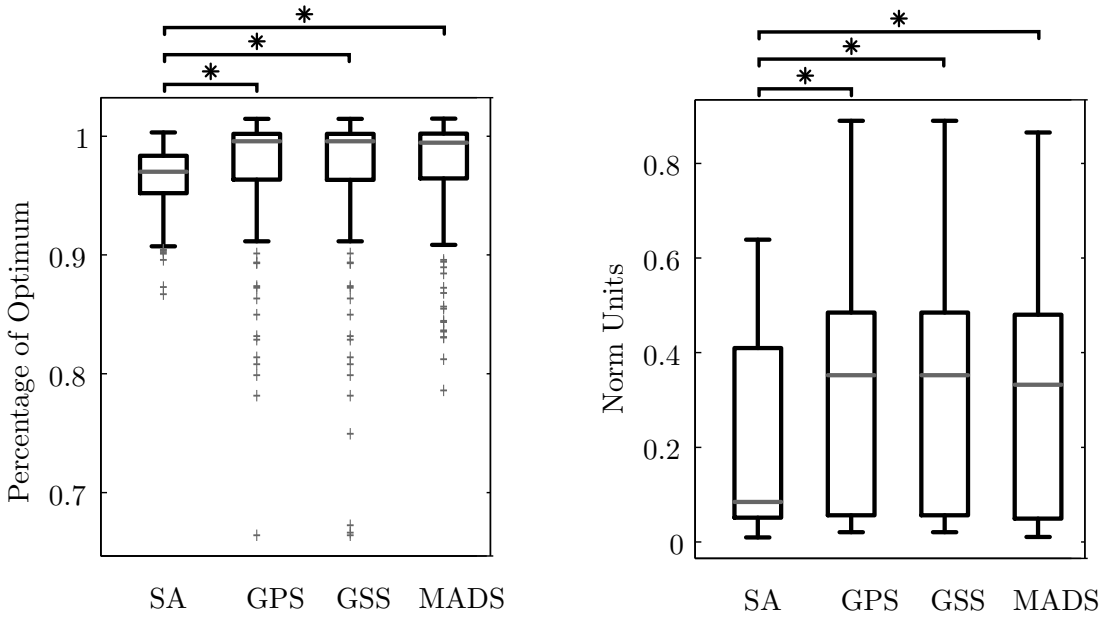


Figure 5.19: Results of the 9 DoF alignment for different optimisation algorithms. (Left) Percentage of best alignment found with reference mapping. (Right) Distance from best alignment in normalized units.

Since the computation of a complete reference map for the 9 DoF problem is not feasible, the evaluation of the algorithms is conducted in two stages. First only the geometric alignment of the room origins is compared against a 3 DoF reference map[21]. As the results

[21]The computation of this map is analogous to the reference map for the 3 DoF problem in Section 5.6.3, but with all user-specific terms set to zero.

of this stage in Figure 5.19 show, the SA approach does not reach the same consistently high $E_{\text{total}}(\omega)$ scores as in the 3 DoF scenario[22]. In this regard, the pattern search algorithms outperform the SA approach. While the overall alignment scores are not promising, the results are at least spatially closer to the reference solutions than the results returned by the pattern search methods[23]. Nevertheless, the high computational cost associated with the SA optimisation and the lower energy scores lead to a clear recommendation towards pattern search methods in this stage.

During the user placement of the second optimisation stage, all tested algorithms placed the users in legal positions. The alignments contained neither obstacle intersections nor proximity violations as shown in Table 5.6. This consistent performance is a great improvement in comparison with the mixed results from the 3 DoF alignment. While the approach with fixed user positions fails consistently for some rooms, the free positioning of users reliably finds conflict-free placements for all room-to-room combinations. On the other hand, this improvement requires additional interaction with the users, guiding them to specific positions in the room before the meeting can take place. The overall number of function calls rises as well, ranging from mere 100 additional calls for MADS optimisation up to staggering 2400 more evaluations for SA methods (median values).

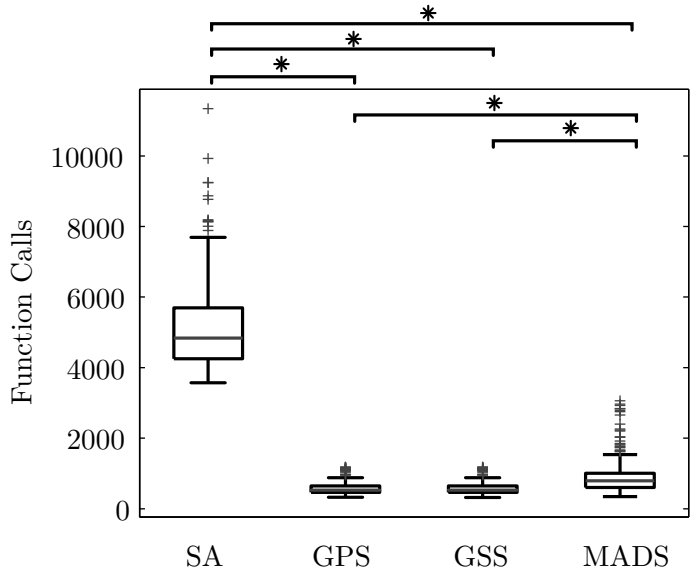


Figure 5.20: Total function calls during the 9 DoF alignment for different optimisation algorithms.

The general trend in the number of function calls is the same as for the 3 DoF problem. Shown in Figure 5.20 are the overall evaluation counts for the entire alignment

[22]Median score percentage of best reference value: 96.6 % for SA, 97.9 % for GPS, 97.7 % for GSS and 97.9 % for MADS. Full analysis in Table A.10 in Appendix A.

[23]Median unit distance to best reference value: 0.20 for SA, 0.37 for GPS, 0.38 for GSS and 0.35 for MADS. Full analysis in Table A.11 in Appendix A.

procedure[24]. For more detail on the two stages, Figure 5.21 shows the count of function calls for the geometry alignment and the user placement separately. As before, the SA approach requires an order of magnitude more evaluations of the energy function. While the overall trends are the same, the smaller difference in the calls for the geometry alignment between MADS and GPS/GSS is interesting to note. For the geometry alignment in the 9 DoF problem, MADS requires about 250 more calls in median than GPS and GSS. For full the 3 DoF problem, the difference amounts to about 400 more calls in median.

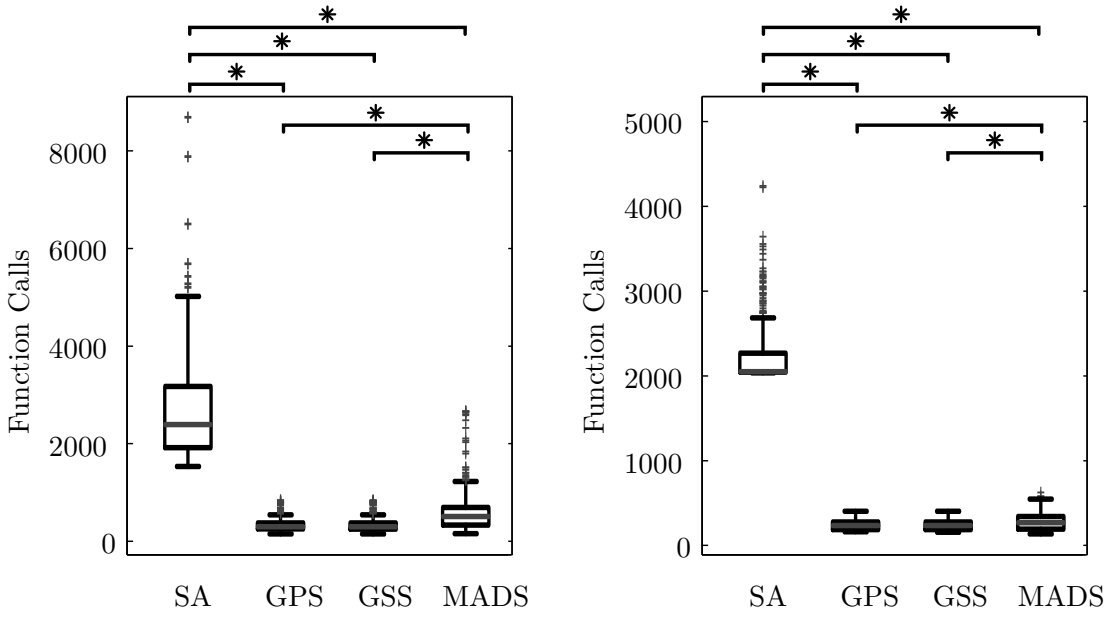


Figure 5.21: Function calls by stage during the 9 DoF alignment for different optimisation algorithms. (Left) Function calls used for geometric alignment. (Right) Function calls required for subsequent user placement.

At the geometric alignment stage, the only difference to the full 3 DoF alignment is the exclusion of the three energy terms connected to user positions. When compared to Figure 5.17, the values visualised in Figure 5.21 therefore serve to show the impact of the user placement terms $E_{\text{head}}(\omega)$, $E_{\text{prox}}(\omega)$ and $E_{\text{wall}}(\omega)$ on computational cost of the 3 DoF alignment. The inclusion of the user terms $E_{\text{prox}}(\omega)$, $E_{\text{head}}(\omega)$ and $E_{\text{wall}}(\omega)$ in the 3 DoF problem increases the number of function calls for MADS by about 250. For the other algorithms, the number of function calls rises by less than fifty calls and the computational cost remains nearly the same. This sensitivity of the MADS algorithm to the user terms is founded in steep energy gradients introduced by the wall avoidance term $E_{\text{wall}}(\omega)$ and the user proximity term $E_{\text{prox}}(\omega)$. The GPS and GSS methods are prompted simply to change the direction of exploration by such steep decreases. The SA method operates similarly.

[24]Median function calls for full 9 DoF optimisation: 5095 for SA, 583 for GPS, 582 for GSS and 884 for MADS. Full analysis in Table A.12 in Appendix A.

The MADS algorithm however first spends additional iterations exploring the finer grid of the POLLING mode before reducing grid size and returning to the SEARCH mode[25].

It is interesting to compare the user placement results to the performance of a brute-force user placement. The grid used to calculate the initial user placements can also be used to conduct a constrained brute-force search for possible user placements (compare Figure 5.14). In this case, all possible starting positions of user A are evaluated against all starting positions of user B. Since the number of unobstructed grid nodes depends on the floor plane alignment, the number of evaluations changes depending on room geometry and alignment. This method evaluates only user positions within the unobstructed floor plane, leading to practically no misplacements as long as the rooms are large enough. The computational cost however is quite high. Performing on the same set of synthetic rooms as the previous approaches, the number of function calls averages to approximately 370, with the lower quartile at 278 and the upper quartile at 462[26]. This average is significantly higher than the number of calls required by the pattern search methods (238 evaluations in average, with the upper quartile at 276). At the same time, the brute-force approach works only on a fixed grid, preventing the system from finding better solutions outside of its scope. These results again underscore the importance of using suitable and flexible optimisation algorithms.

5.6.6 Conclusion for 9 DoF

As was to be expected, the 9 DoF problem results in higher computational cost than the 3 DoF problem. However, the additional computations guarantee legal and conflict-free initial positions of users.

Comparing the different solver approaches, the SA method performs poorly for the 9 DoF problem. Even though the geometric room alignment is markedly close to the solutions provided by exhaustive search, the overall $E_{\text{total}}(\omega)$ scores are lower than results returned by the pattern search approaches. At the same time, the number of function calls and in consequence the computational cost exceeds the costs of all other methods by an order of magnitude.

The performance of GPS and GSS optimisation is virtually indistinguishable for the computation of a consensus reality. The implicit declaration of limits and boundaries in the total energy function $E_{\text{total}}(\omega)$ are the cause of this ambiguity. While there are explicit boundaries on user positions, the convergence process does not approach these and consequently the GSS-specific computation of the exploration pattern is not applied. Both approaches show high final alignment scores while simultaneously requiring the least function calls of all algorithms tested.

The MADS algorithm shows slightly better performance than the GPS/GSS methods. Of all tested algorithms, it returned the highest mean energy for the 3 DoF problem. One drawback is the higher number of function calls required when compared to GPS/GSS optimisation. In summary, the GPS optimisation offers the best balance between alignment

[25] For more detail on the MADS exploration strategy, turn back to Section 3.3.

[26] A 10×10 grid was used. Due to the mapping on the free floor plane not all nodes were queried. Alignment topology also influenced the number of grid points associated with each user.

quality and computational cost. As it is straightforward to implement, it is adaptable to a wide range of implementations and delivers robust and repeatable results.

5.7 Visualising the Consensus Space

Once the alignment is finished, the render engine uses the various maps to generate a visualisation of the consensus space. Since both occupancy maps and the original scans are available in the same reference frame, regions from the maps can be associated directly with vertices from the 3D scans of the rooms. This convenient connection produces an intuitive visualisation of select features, e.g. common work surfaces could be overlaid with a transparent polygon for easy recognition. However, this visualisation should be employed sparingly in order to avoid visual clutter.

Before the maps are displayed as 3D objects, an algorithm transforms them from discrete pixel maps into polygons. This transformation reverses the mapping functions in Equations 5.4 and 5.5 between world and map coordinates. As the z-coordinate corresponds with height, there are two choices for its value. The first approach sets the z-value to the height of the highest physical object at the spot the point is projected to. This choice visualizes work surfaces and other regions which may be elevated above the floor plane. On the other hand, if the map contains obstacles, the rendering should create the illusion of a wall. To achieve this effect, the visualisation anchors the object to the floor plane and stretches the boundary up to the ceiling. For both modes of display, a function maps each pixel back into the 3D consensus reality:

$$\forall \mathbf{p}(x,y)_j > 0 \in \mathbf{M}_{\text{map}}^{\text{room}} :$$

$$v_{j,\text{x}} = \frac{x - 0.5 \cdot s_x}{c_x} \tag{5.88}$$

$$v_{j,\text{y}} = \frac{y - 0.5 \cdot s_y}{c_y} \tag{5.89}$$

$$v_{j,\text{z}} = \begin{cases} \mathbf{M}_{\text{map}}^{\text{room}} & \text{for surfaces} \\ 0 & \text{for obstacle boundaries} \end{cases} \tag{5.90}$$

Instead of blindly transforming all points back into the respective world coordinate systems of the rooms A and B, it suffices to consider the borders of the mapped regions. These boundaries then define meshes suitable for rendering. To this end, the map $\mathbf{M}_{\text{map}}^{\text{room}}$ is first transformed to a binary representation using a simple thresholding operation. This binarisation operation marks the map pixels of all regions which are to be rendered. All other parts of the map are set to zero. The Teh-Chi algorithm then finds the boundary line segments within this binary map [TC89]. Only the N edge points defining these segments are then transformed back into world coordinates. The resulting vertices serve as the basis for constructing the meshes required by the rendering engines. The construction of these meshes varies for the purpose of the map: For boundaries and obstacles, the user should see walls spanning the entire height of the room. On the other hand, work surfaces are best represented using a flat mesh spanning the surface in the horizontal plane.

Creating a wall is achieved by adding a new point over every existing point at a fixed vertical offset z_{offset}, typically 2 meters. The new set $\mathcal{S}_{wall}$ of points provides the vertices for a secondary polygon creation step and contains $2N$ points:

$$\mathcal{S}_{wall} = \left\{ \begin{pmatrix} v_{1,\text{x}} \\ v_{1,\text{y}} \\ v_{1,\text{z}} \end{pmatrix}, \begin{pmatrix} v_{1,\text{x}} \\ v_{1,\text{y}} \\ v_{1,\text{z}} + z_{\text{offset}} \end{pmatrix}, \ldots, \begin{pmatrix} v_{N,\text{x}} \\ v_{N,\text{y}} \\ v_{N,\text{z}} \end{pmatrix}, \begin{pmatrix} v_{N,\text{x}} \\ v_{N,\text{y}} \\ v_{N,\text{z}} + z_{\text{offset}} \end{pmatrix} \right\} \tag{5.91}$$

For rendering engines which rely on right-handed surface normals, this set of points is traversed and triangularised sequentially in order to create the new mesh $\mathcal{M}_{\text{wall}}$. The new mesh is composed of sets of polygons $\mathcal{V}_i$ and their corresponding normal vectors $\mathbf{n}_i$. The following procedure creates floor-to-ceiling meshes from $\mathcal{S}_{wall}$:

$$\forall \mathbf{v}_i \in \mathcal{S}_{wall}$$

$$\hat{p} = \begin{cases} i+1 & \text{if } \texttt{modulo}(i,2) = 0 \\ i+2 & \text{if } \texttt{modulo}(i,2) \neq 0 \end{cases} \tag{5.92}$$

$$p = \texttt{modulo}(\hat{p}, 2N) \tag{5.93}$$

$$\hat{n} = \begin{cases} i+2 & \text{if } \texttt{modulo}(i,2) = 0 \\ i+1 & \text{if } \texttt{modulo}(i,2) \neq 0 \end{cases} \tag{5.94}$$

$$n = \texttt{modulo}(\hat{n}, 2N) \tag{5.95}$$

$$\mathcal{V}_i = \{\mathbf{v}_p, \mathbf{v}_i, \mathbf{v}_n\} \tag{5.96}$$

$$\mathbf{n}_i = \frac{(\mathbf{v}_p - \mathbf{v}_i) \times (\mathbf{v}_n - \mathbf{v}_i)}{\|(\mathbf{v}_p - \mathbf{v}_i) \times (\mathbf{v}_n - \mathbf{v}_i)\|} \tag{5.97}$$

these are collected in a full mesh:

$$\mathcal{N}_{\text{wall}} = \{\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_{2N}\} \tag{5.98}$$

$$\mathcal{P}_{\text{wall}} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_{2N}\} \tag{5.99}$$

$$\mathcal{M}_{\text{wall}} = \{\mathcal{P}_{\text{wall}}, \mathcal{N}_{\text{wall}}\} \tag{5.100}$$

The graphics engine then renders the resulting mesh $\mathcal{M}_{\text{wall}}$ as a wall spanning from the floor to the ceiling.

The process for visualising planar meshes is quite similar. After the binarisation and Teh-Chi algorithm, the N points describing the boundary of each region are used as the starting points for a Delaunay Triangulation [Del34]. This algorithm yields a convex set of triangles $\mathcal{P}^{\text{raw}}_{\text{surface}}$ covering the space between all contour points. Since the map contours are not necessarily convex, a secondary step discards all triangles covering empty regions of the map. As illustrated in Figure 5.22, this refinement step tests the centre of mass of each triangle against the original map. If the polygon spans a concavity which is not part of the surface, the map should show only low values at the centre of mass. Therefore, the concavity test discards this triangle. For the remaining triangles $\mathcal{P}_{\text{surface}}$, the normal vectors $\mathcal{N}_{\text{surface}}$ are set as the normalised cross-product of the two primary vertex directions

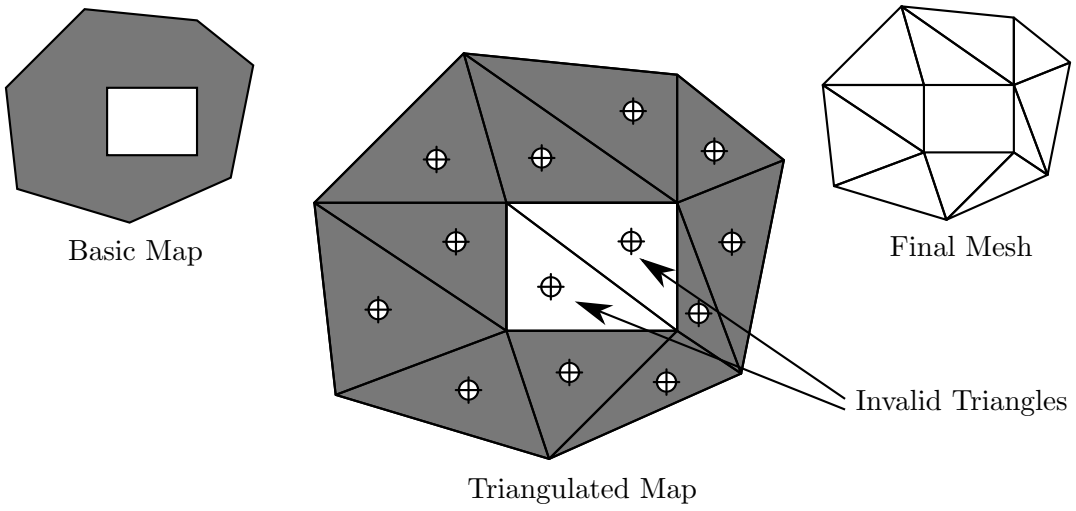


Figure 5.22: Conversion of a map into a mesh using Delaunay triangulation and center of mass concavity detection.

as in Equation 5.97. The resulting mesh $\mathcal{M}_{\text{surface}}$ can then be rendered by the graphics engine.

The following observations were made using a mock-up of a AR videoconference in a CAVE. While not tested on a full-scale system, they guide future development of a feature-complete implementation.

Since current HMDs still support only a rather small field of view, care should be taken that visualisations are obvious to the user. For instance, a visualisation of obstacles in the floor plane requires the user to frequently look down at the floor. On the other hand, visualising obstacles as persistent floor-to-ceiling walls tends to clutter the field of view. A good compromise is the selective rendering of obstacles as volumetric objects which become visible only if the user moves close enough. Under normal conditions, this mode would render the conversation partner and additional AR content. Only as the user moves close to the boundaries of the shared free floorspace, a transparent wall becomes visible. As the user moves closer to the wall, the transparency is reduced linearly. Thus, obstacles start as ghostlike objects at a distance and become more solid as the user moves closer.

Shared work surfaces are another feature which does not need to be displayed at all times. Consequently, the available surfaces are highlighted only when a user wants to place a virtual object. This context sensitivity draws both on the human pose tracking and the internal state of the render engine: If a user is close to a shared table surface and manipulating an object, the table should be highlighted so the user can place the object on its surface.

As the field of view in many HMDs might limited, additional markers at the edge of the augmented field of view can be used to guide the user's gaze into the right direction[27].

[27]The diploma thesis written by Schäfer ties in at this point [Sch13]. In the thesis, markers at the edge of vision guide the user towards specific virtual objects.

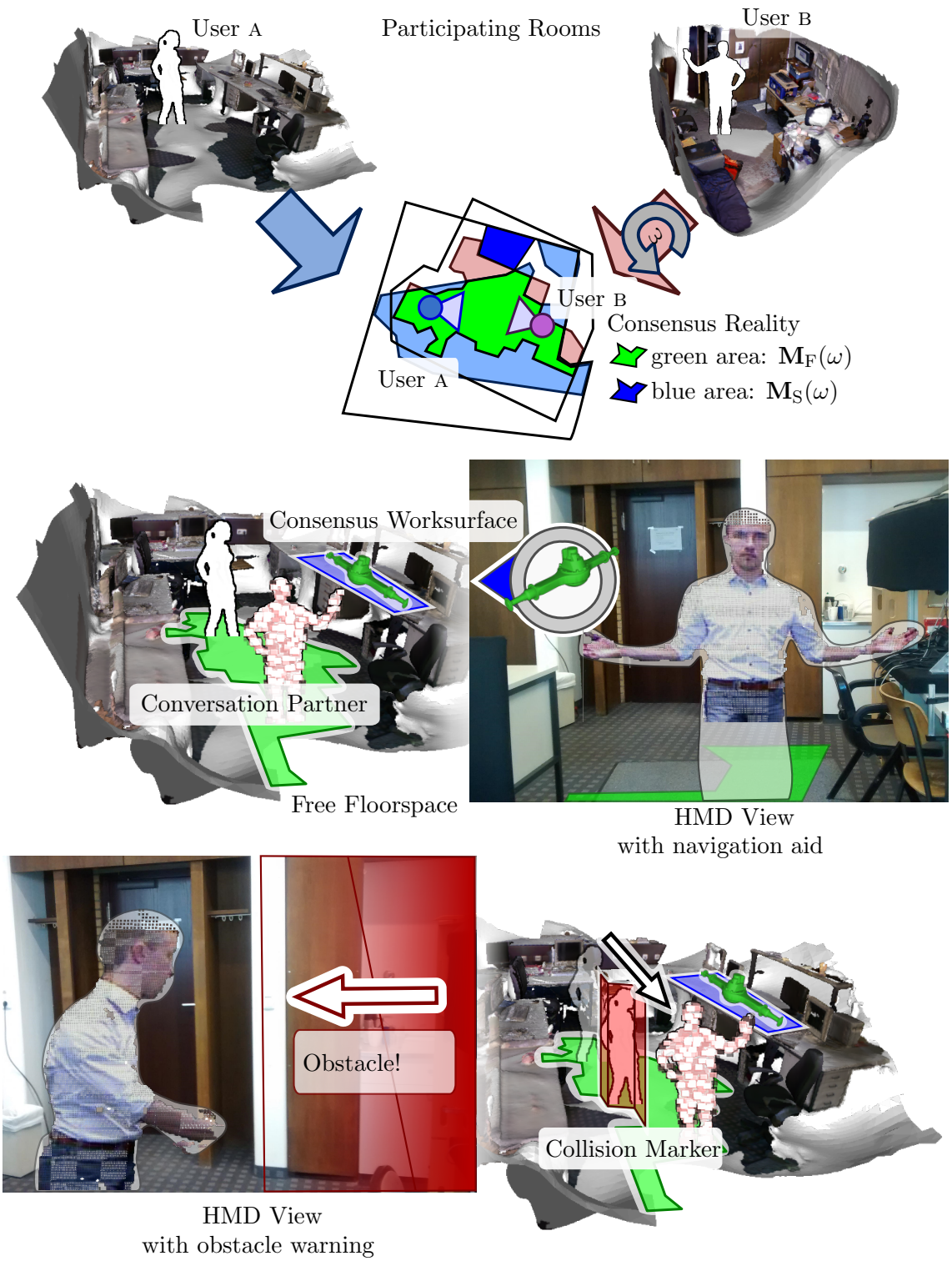
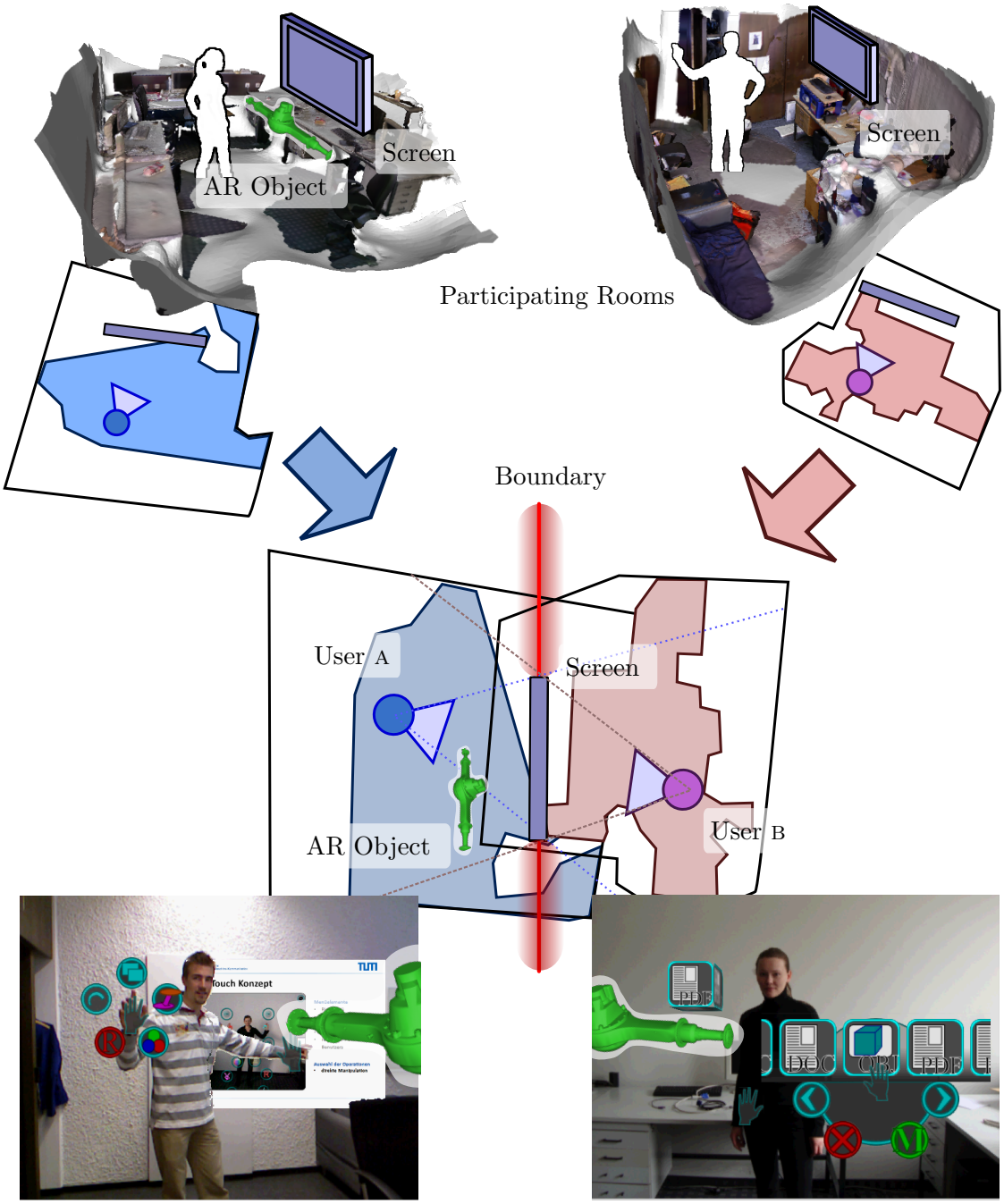


Figure 5.23: Consensus Reality as perceived by user A. The lower schematic shows how collision warnings appear dynamically as the user approaches the boundaries. Size and colour of the navigation aids were adjusted for legibility after printing.



Screen View for User A
with additional AR content

Screen View for User B
with additional AR content

Figure 5.24: Conventional screen based AR videoconferencing. While connecting the two rooms with a window analogy, the screens also act as implicit barriers between the participants' surroundings. Size and colour of the navigation aids were adjusted for legibility after printing. This system was presented at the IEEE ISMAR 2012 [LER12].

In summary, the visualisation should show only what is truly relevant in the current situation in a manner adapted to the limited field of view available on the hardware. Otherwise the visual clutter might quickly overwhelm the user. For instance, if all workspace boundaries were rendered at the same time, they would fill the view in all directions. Meanwhile, visualisation constrained only to the floor plane is bound to lead to confusion with current HMDs due to their limited field of view.

5.8 Advantages of AR Videoconferencing

Figure 5.23 illustrates the integration of these visualisation aids into a typical consensus reality videoconference. The scene shows the conference from the point of view of user A. Through her HMD, she sees user B standing in her room. In order to compensate for a limited field of view, navigation markers point towards active virtual objects in her vicinity. As she moves close to the boundaries of the consensus free floor space, the border appears to her as a virtual obstacle. Correspondingly, user B sees user A standing in his room with the outlines of the consensus reality rendered to his own HMD. In order to facilitate comparison, Figure 5.24 shows the spatial alignment used for a screen-based videoconference. In this case, the two displays serve as a window analogy. However, they also act as implicit dividers between the participants. The users cannot reach through the window or step into the other room. This limitation excludes communication cues that are conveyed through proximity and relative orientation. On the other hand, the consensus reality conference integrates both rooms into a common workspace. Both participants can move and interact freely within this space, using their entire presence and posture as a channel for non-verbal communication.

This integration also leads to marked differences in the handling of AR objects during the conversation. In the screen-bound approach, an object is located either in room A or room B. For illustration, there is a green CAD model shown in both figures. In the conventional videoconference, the model is placed in room A and floats in front of the screen. User B can also see and manipulate it, but by the implication of its placement in her room, it "belongs" to user A. Since AR content is only displayed on the screen surface, the model always takes up some part of the shared view. In contrast, during the consensus reality conversation the object is placed on a physical table. As it is a consensus surface, both participants see it resting on a surface in their respective rooms. There is no implicit sense of ownership conferred by the placement and both users can walk up to it freely. As they share the same space, they can regard the object from the same direction and use natural gestures to point out details to the conversation partner. When the conversation turns to other matters, the model remains on the table without obstructing the users' view.

In both implementations, participants can manipulate objects through gesture driven menus. In the case of the screen-based videoconference, such menus occupy part of the screen connecting the rooms. Thus interaction always obstructs the view of the conversation partner. With the consensus reality, these menus are registered to the 3D AR objects and thus only visible while looking at the object. This enables a less cluttered view of each other. Since the controls are registered to objects instead of screen positions, they are

accessible to both users. So one user could step up to an AR content object and call up the rotation controls by touching its base. The other conversation partner then leans forward and drags the same rotation controls to show the model from a different angle. Both users would interact with the same set of virtual controls, instead of using individual on-screen menus. Thus the co-presence in the consensus reality not only affects the rendering, but also leads to more natural interactions during content manipulation.

5.9 Summary of Chapter

This chapter introduced a new way of integrating videoconference participants into each other's physical environments. Previous approaches put a number of constraints on the interaction of users by defining boundaries separating the users' surroundings. In their simplest form, these boundaries take the shape of the window analogy familiar from screen-bound videoconferencing. In more elaborate implementations, the boundaries become apparent by prioritising one room over the other, as it is practised in remote support scenarios.

Ignoring such boundaries, the discrepancies between the layouts of the rooms lead to breaks in immersion. In order to prevent a user from stepping through the conversation partner's desk, it is necessary to find an alignment which minimises the chance of such conflicts to arise. The approach presented in this chapter uses static maps of the participating rooms to find configurations which afford a maximum of mobility and unobstructed space to both users.

Energy functions encode the criteria for an optimal alignment and guide the optimisation procedure. The chapter explains the rationale behind the design of this function and shows how single functional terms are connected to goals derived from the usage scenario, human perception and social conventions. The energy function thus comes to serve as a quantitative measure for the quality of alignment.

In order to integrate this measure into the context of the AR videoconference outlined in Chapter 2, an automated process is used to find and apply optimal alignments. In this chapter, a number of widely used heuristic search methods were applied to this problem. After comparing the performance over 16 scenarios, each with 27 starting positions, the MADS algorithm has shown the most promising performance on the 3 DoF problem. For the more flexible 9 DoF problem evaluated on the same test-set, the GPS method achieved the best results.

These two optimisation approaches enable the quick and automatic generation of consensus realities for AR videoconferencing respecting goals and constraints set by the user. The consensus reality method itself is the first published approach to unify two remote and heterogeneous spaces into a common collaboration space in the context of AR research.

Chapter 6

Conclusion

At the conclusion of this dissertation, it is worth reviewing the basic elements presented and placing them into the context of a consensus reality videoconference. The initial two chapters defined the basic concept and its background with regard to previous computer supported cooperative work, augmented reality and telepresence research. While these are very active fields of research, there are to date no previous descriptions of comparable systems. The mutual and largely unconstrained inclusion of users into each other's physical surroundings poses a novel concept. The second chapter outlined the general architecture for such a system and described the specific challenges arising from the mutual inclusion of user avatars. Since tackling all these challenges would far exceed the scope of a single dissertation, the following chapters focus on the two optimisation problems lying at the core of this system.

The first challenge described in greater detail is the pose tracking of both users, a wide and active field of research in itself. The spatially unconstrained videoconferencing scenario necessitates an adaptation of previous approaches. The goal is to perform real-time pose tracking on data coming from multiple cameras. Other than most other tracking approaches, the central observation likelihood approximation presented here does not rely on silhouettes extracted from 2D or 2.5D images. Instead, it operates directly on the merged point cloud gathered by the camera system. This enables the seamless integration of data from an arbitrary number of cameras into a single pass of the observation likelihood approximation function. Together with a segmentation scheme and an adapted scattering mechanism, this approximation drives an annealing particle filter performing at up to 40 fps on standard desktop hardware. The user poses provided by the camera system enable the gesture-driven interaction with AR content during the conversation and inform the initial generation of the consensus reality.

Based on the initial pose of the users, the conversation partners' rooms are merged into a shared consensus reality. This poses the second major challenge considered in this dissertation. Previous approaches to remote AR collaboration relied on pre-defined shared workspaces or specially prepared rooms and tables. Unless the conversation partners are working at large companies with specially prepared meeting rooms, one cannot expect such a level of preparation just for a casual conversation. Therefore, the approach shown here works ad-hoc with almost any room. An initial alignment algorithm produces room

configurations which encourage natural and intuitive collaboration. The solution is informed both by pragmatic constraints, such as the existence of free space, as well as by constraints derived from psychological insights. The meeting then takes place in a consensus reality, which unites the physical spaces of both users into a shared reference system. The conversation partners themselves are augmented into each other's room, so that the meeting takes place in a familiar setting for both participants.

Besides the application scenario, both challenges share mathematical concepts used for solving the underlying optimisation problems. Since there is no analytical description available for either of them, they are only solvable using methods which do not employ gradients for optimisation. Many stochastic and pattern search approaches are capable of finding optimal or near-optimal solutions even for problems which have a so-called oracle description: The solver does not know the underlying topography of the problem, but derives its solution simply from polling a "black box" function or sensor. While there are no comprehensive analytical solutions available either for human motion tracking nor for the consensus reality, it is possible to formulate such "black box" functions, or heuristics. For the consensus reality, the time-stationary optimisation problem was approached with pattern search methods and the simulated annealing algorithm. In the human pose tracking, the annealing particle filter extends the concept of the simulated annealing to series of observations. In both cases, the tested optimisation led to good solutions and robust results.

While this dissertation contains a detailed examination of the most important concepts, it cannot cover all aspects of the challenges posed by consensus AR videoconferencing. Absent from this dissertation is a full hardware setup and usage evaluation. While this dissertation was written, some dramatic shifts in the hardware landscape occurred: At the beginning of the dissertation, the introduction of the *Kinect* sensor in November 2010 sparked a rapid adaptation of depth sensors in research. Another remarkable development was the release of the *Oculus Rift* head mounted display, which was adapted to AR applications by William Steptoe *et al.* [SJS14][1]. Besides these two tools, a wide range of powerful and affordable hardware has been made available to researchers in the span of merely five years. The entire system as described in Chapter 2 is implementable with current hardware. Since the hardware is already progressing at great leaps and with the backing of multinational corporations, this dissertation focusses on the underlying methods for computing a consensus reality. For initial design evaluation, a virtual model of the conferencing system was implemented in the institute's CAVE virtual reality environment.

In summary, this dissertation presents five major contributions to the fields of remote collaboration and human pose tracking:

- A definition of the concept of an consensus reality videoconferencing system together with an overview over a generic architecture for its implementation in Section 2.3. This is the first comprehensive description of mutual spatial integration in remote collaboration.

[1] `http://willsteptoe.com/post/66968953089/ar-rift-part-1`. Accessed 02.02.2015

- Adaptations of the APF for human pose tracking based on 3D point clouds in Section 4.3. Section 4.4 presents an evaluation of parameter influence and overall performance. The approach tracks complex motion sequences at more than 40 fps.

- An approximation of observation likelihood for human poses based on 3D point clouds in Section 4.2. The evaluation in Section 4.4 shows improved resolution in comparison to a state-of-the-art reference approach.

- A design of an energy function quantifying the quality of alignment for heterogeneous rooms in Section 5.3. This approach to spatial integration is entirely novel, with no previous descriptions in literature.

- The evaluation of global optimisation methods applied to the alignment energy function in Section 5.6. The optimisation stage automatically aligns two arbitrary rooms with minimal user interaction. This alignment enables the rapid integration of the two rooms into a consensus reality.

These methods allow users to meet in a consensus reality spanning both their rooms. Other than screen-based videoconferencing, this mode of communication does not impose barriers between the participants of a remote meeting. Instead, they share the same physical space and enrich their conversation with a wide range of natural communication cues.

For future work, a full implementation should be built for user studies. Of special interest would be the differences in social interaction between screen based videoconferencing, immersive collaboration tools and the consensus AR videoconference. We should expect that the consensus reality approach fosters a more intuitive and natural interaction, since a wider range of human social cues is transmitted than with the former methods. As the avatars share the same space and are not separated by the window analogy of traditional videoconferences, body posture can be employed more effectively. A person might turn away slightly, back off from a conversation partner or come closer. While these modalities are also available for immersive telepresence, the inclusion of the real physical surroundings should give the conversation a more natural feeling. The users are not simply transferred to some abstract virtual space, but instead remain in their familiar surroundings.

Another interesting topic for future research would be the modification of the consensus reality approach to non-linear alignments of the remote spaces. The approach presented in this dissertation assumes that the rooms remain unaltered. The optimisation adjusts only the position of the origins and possibly the starting positions of the users. However, using approaches borrowed from *non-linear mapping*, the room mapping itself can be warped in order to maximise the common work space. This warping would at first lead to startling motion patterns, as users traverse a non-linear space in front of each other's eyes. A walking user might therefore appear to speed up or slow down even while moving at constant speed, since the underlying space mapping is warped. Solutions to this problem can be found in previous research on *redirected walking*, where short shifts in attention focus are exploited for modifying virtual content unnoticed. Another challenge would arise from pointing and gaze discontinuities. Since a non-linear mapping stretches and compresses the underlying space, pointing gestures and gaze cues appear distorted to the

remote conversation partner. In order to compensate for such effects, the rendering must adjust the user's avatars. There is already active research on this problem in the context of screen based videoconferences, where the offset between camera and screen leads to a gaze discontinuity when looking at the conversation partner. Remapping approaches developed for the flat screen scenario could thus be extended to 3D avatars populating the same space.

Modifications of the consensus reality mapping can also provide support when setting up the camera system. The current alignment procedure assumes camera positions and orientation to be fixed. With some alterations, the system could suggest alternative positions for single camera modules in order to provide better observation coverage of the room. Such suggestions would require a modification of the solver approach. For the current nine DoF alignment problem, two stages are used: The first aligns the rooms, the second finds initial user positions. In order to adjust the camera position, the first stage would need to be preceded by an additional step in which the camera placement for a single room is optimised in order to arrive at a maximum observable floorspace. This additional stage leads to a high-dimensional optimisation problem, since each camera module has six DoF, with usually more than five cameras per room. The problem could be constrained by limiting camera placement to pre-defined regions. Nevertheless, this is still a complex task requiring careful further study. The problem is also relevant to other fields utilising multiple cameras, such as motion tracking systems, surveillance etc.

While consensus reality videoconferencing will not replace other forms of remote collaboration, it provides an additional communication channel tailored for conversation scenarios. As foreshadowed by the *Oculus Rift* or *Samsung Gear* HMDs, we can expect compact VR gaming systems to enter the market within the next three to seven years. The integration of motion capturing systems is the logical extension of such platforms, as shown by the popularity of the *Kinect 2*. Recent patents by the Microsoft Cooperation [DS14; Lat+13], by Samsung [CS14] and a number of other companies give a first impression of the AR/VR platforms envisioned. The major hardware elements required for consensus reality conferencing are all present in the concepts presented by these companies: Small, interconnected camera modules, lightweight HMDs and integration of remote users. Once such systems have become commonplace, the realisation of a consensus reality videoconference outside of a laboratory setting will become almost trivial from a hardware perspective.

This dissertation is intended to inform future research towards this goal, defining the fundamental concepts and showing practical solutions to the central challenges. Even though not all obstacles are cleared yet, the research described in these pages shows the feasibility of the approach and provides the tools for a full implementation.

As a long-term vision, we should expect a future where our conversations are no longer limited to phones and videoscreens. Instead, our conversation partners will appear right in our offices, sit at our table, share our surroundings - even when they are on another continent.

Appendix A

Statistical Analysis of Consensus Reality Alignment

Table A.1: Parameters used for the brute force solver (3 DoF problem only)

Parameter	Value	Unit
Resolution x-direction	0.2	m
Resolution y-direction	0.2	m
Resolution rotation	12	degree
Upper bound x-direction	2	m
Lower bound x-direction	-2	m
Upper bound y-direction	2	m
Lower bound y-direction	-2	m
Upper bound rotation	180	degree
Lower bound rotation	-180	degree

Table A.2: Parameters used for the simulated annealing solver

Parameter	Value	Unit
Upper bound x-direction	2	m
Lower bound x-direction	-2	m
Upper bound y-direction	2	m
Lower bound y-direction	-2	m
Upper bound rotation	180	degree
Lower bound rotation	-180	degree
Initial Temperature	100	none
Convergence Threshold	$\Delta E_{\text{total}}(\omega) \leq 1 \times 10^{-6}$	none

Table A.3: Shared Parameters used by all pattern search solvers

Parameter	Value	Unit
Upper bound x-direction	2	m
Lower bound x-direction	-2	m
Upper bound y-direction	2	m
Lower bound y-direction	-2	m
Upper bound rotation	180	degree
Lower bound rotation	-180	degree
Convergence Threshold	$\Delta E_{\text{total}}(\omega) \leq 1 \times 10^{-6}$	none

Table A.4: Parameters used by GPS and GSS pattern search solvers

Parameter	Value	Unit
Initial Mesh Size	1.0	none
Mesh Expansion Factor	2.0	none
Mesh Contraction Factor	0.5	none

Table A.5: Parameters used by MADS pattern search solver

Parameter	Value	Unit
Poll Parameter Δ^P	$\sqrt{\Delta^M}$	none
Initial Mesh Size	1.0	none
Mesh Expansion Factor	4.0	none
Mesh Contraction Factor	0.25	none

Table A.6: Parameters used for computing $E_{\text{total}}(\omega)$.

Variable	Initial Values	Unit
α_{free}	1.0	none
α_{prox}	0.6	none
α_{head}	0.6	none
α_{wall}	1.0	none
α_{surf}	0.5	none
α_{mins}	0.0	none
α_{skew}	0.0	none
$d_{\text{max, limit}}$	5.0	m
λ_{dmax}	4.0	m
$d_{\text{min, limit}}$	0.4	m
λ_{dmin}	0.8	m
λ_{head}	90.0	degree
Heading mode	`mutual` (see Equation 5.52)	none

Table A.7: Kruskal Wallis Test on Percentage of Brute Force Optimum, 3DoF	
χ^2	39.48
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[0.9769 , 0.98316 , 0.99342]
GPS	[0.93594 , 0.9524 , 1.0024]
GSS	[0.92859 , 0.94772 , 1.0024]
MADS	[0.94289 , 0.96098 , 1.0048]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 0.00010051,\ W_A = 172575 \Rightarrow$ significant
SA vs GSS	$p = 0.0011096,\ W_A = 174879 \Rightarrow$ significant
SA vs MADS	$p = 3.7829e-11,\ W_A = 162587 \Rightarrow$ significant
GPS vs GSS	$p = 0.68816,\ W_A = 188312 \Rightarrow$ insignificant
GPS vs MADS	$p = 0.047591,\ W_A = 179574 \Rightarrow$ insignificant
GSS vs MADS	$p = 0.016899,\ W_A = 178078 \Rightarrow$ insignificant

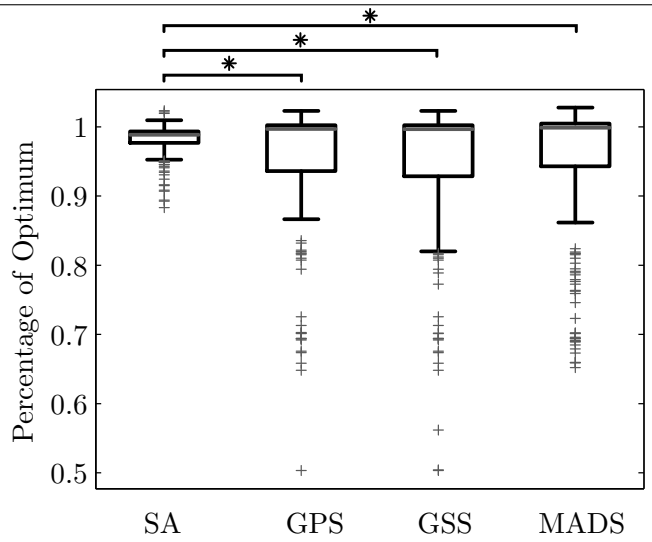


Figure A.1: Boxplots illustrating results from Table A.7.

Table A.8: Kruskal Wallis Test on Normalised Distance to Brute Force Optimum, 3DoF	
χ^2	35.96
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[0.054836 , 0.11014 , 0.14046]
GPS	[0.052904 , 0.20343 , 0.3355]
GSS	[0.052904 , 0.20976 , 0.33975]
MADS	[0.042611 , 0.19055 , 0.29837]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 8.836e-08,\ W_A = 167221 \Rightarrow$ significant
SA vs GSS	$p = 6.5679e-08,\ W_A = 167025 \Rightarrow$ significant
SA vs MADS	$p = 0.0023647,\ W_A = 175689 \Rightarrow$ significant
GPS vs GSS	$p = 0.84484,\ W_A = 186122 \Rightarrow$ insignificant
GPS vs MADS	$p = 0.13227,\ W_A = 192361 \Rightarrow$ insignificant
GSS vs MADS	$p = 0.094132,\ W_A = 192980 \Rightarrow$ insignificant

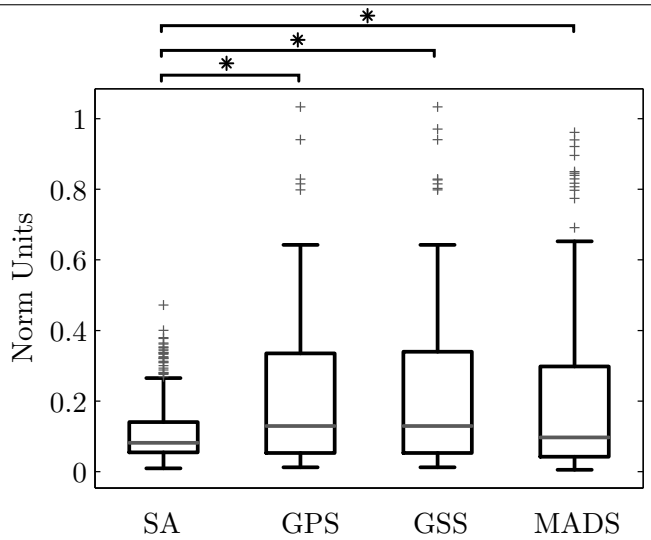


Figure A.2: Boxplots illustrating results from Table A.8.

Table A.9: Kruskal Wallis Test on Function Calls needed for Convergence, 3DoF	
χ^2	1237.52
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[1900 , 2666 , 3223]
GPS	[306.5 , 384 , 426.5]
GSS	[288.5 , 376 , 421.5]
MADS	[523.5 , 786 , 911.5]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 8.8625e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs GSS	$p = 8.8623e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs MADS	$p = 4.4336e-130,\ W_A = 275844 \Rightarrow$ significant
GPS vs GSS	$p = 0.25498,\ W_A = 191015.5 \Rightarrow$ insignificant
GPS vs MADS	$p = 6.7905e-83,\ W_A = 116096.5 \Rightarrow$ significant
GSS vs MADS	$p = 1.4176e-85,\ W_A = 114935.5 \Rightarrow$ significant

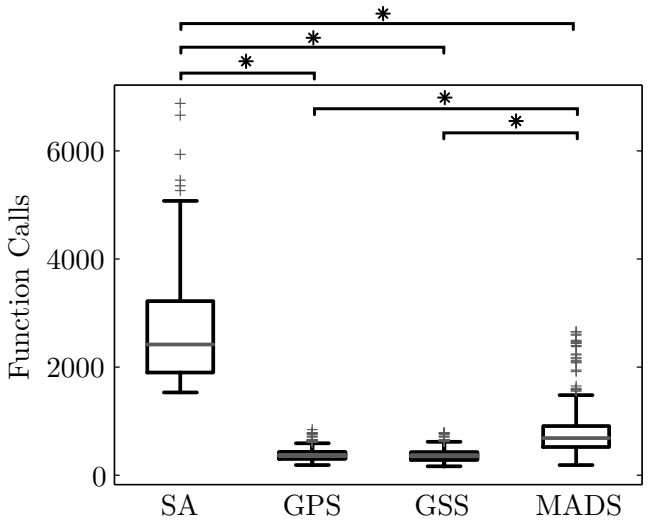


Figure A.3: Boxplots illustrating results from Table A.9.

Table A.10: Kruskal Wallis Test on Percentage of Brute Force Optimum, 9DoF	
χ^2	191.7904
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[0.95201 , 0.96557 , 0.98356]
GPS	[0.96339 , 0.97895 , 1.0019]
GSS	[0.96328 , 0.97704 , 1.0019]
MADS	[0.96446 , 0.97887 , 1.0022]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 8.0714e-31,\ W_A = 144506 \Rightarrow$ significant
SA vs GSS	$p = 1.1963e-29,\ W_A = 145365 \Rightarrow$ significant
SA vs MADS	$p = 2.5613e-28,\ W_A = 146362 \Rightarrow$ significant
GPS vs GSS	$p = 0.88314,\ W_A = 187379.5 \Rightarrow$ insignificant
GPS vs MADS	$p = 0.59184,\ W_A = 188807 \Rightarrow$ insignificant
GSS vs MADS	$p = 0.70156,\ W_A = 188246 \Rightarrow$ insignificant

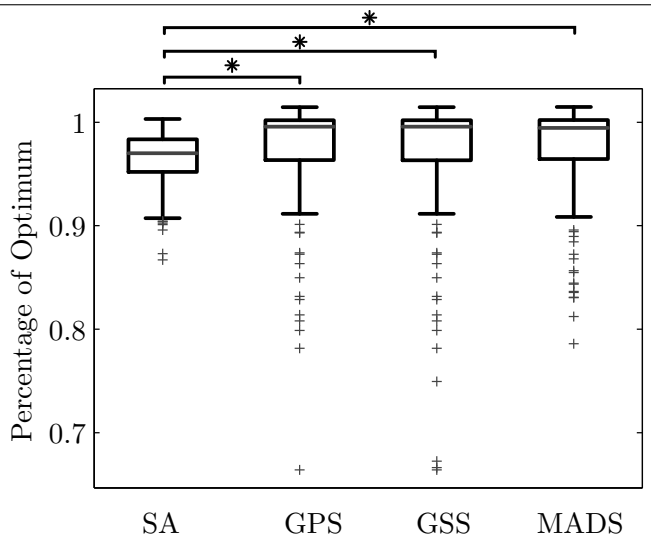


Figure A.4: Boxplots illustrating results from Table A.10.

Table A.11: Kruskal Wallis Test on Normalised Distance to Brute Force Optimum, 3DoF	
χ^2	39.9115
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[0.051314 , 0.20316 , 0.40955]
GPS	[0.056094 , 0.37464 , 0.48472]
GSS	[0.056094 , 0.38701 , 0.48472]
MADS	[0.049114 , 0.35075 , 0.48029]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 5.3579e-08,\ W_A = 166891 \Rightarrow$ significant
SA vs GSS	$p = 2.3059e-08,\ W_A = 166347 \Rightarrow$ significant
SA vs MADS	$p = 6.3969e-05,\ W_A = 172177 \Rightarrow$ significant
GPS vs GSS	$p = 0.89823,\ W_A = 186370.5 \Rightarrow$ insignificant
GPS vs MADS	$p = 0.2424,\ W_A = 191128 \Rightarrow$ insignificant
GSS vs MADS	$p = 0.19922,\ W_A = 191549 \Rightarrow$ insignificant

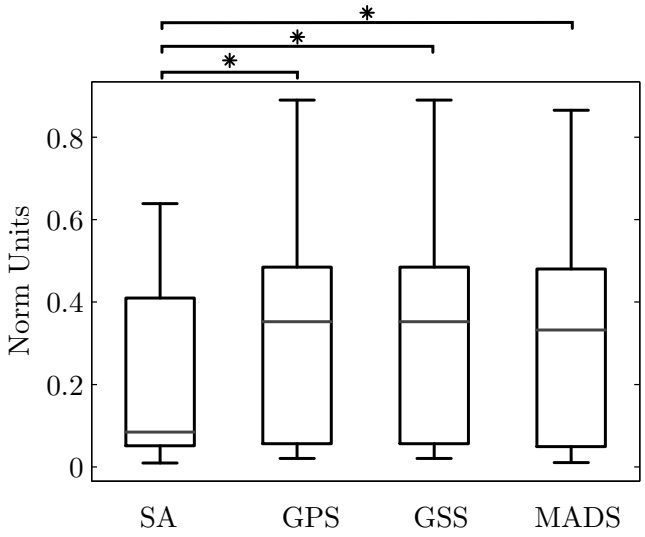


Figure A.5: Boxplots illustrating results from Table A.11.

Table A.12: Kruskal Wallis Test on Total Function Calls needed for Convergence, 9DoF	
χ^2	1126.3863
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[4235 , 5095 , 5735]
GPS	[461 , 583 , 647.5]
GSS	[460.5 , 582 , 647.5]
MADS	[605 , 884 , 1008]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 8.8726e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs GSS	$p = 8.8728e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs MADS	$p = 8.8782e-143,\ W_A = 280152 \Rightarrow$ significant
GPS vs GSS	$p = 0.9044,\ W_A = 187281 \Rightarrow$ insignificant
GPS vs MADS	$p = 9.9534e-47,\ W_A = 134190 \Rightarrow$ significant
GSS vs MADS	$p = 6.6228e-47,\ W_A = 134086.5 \Rightarrow$ significant

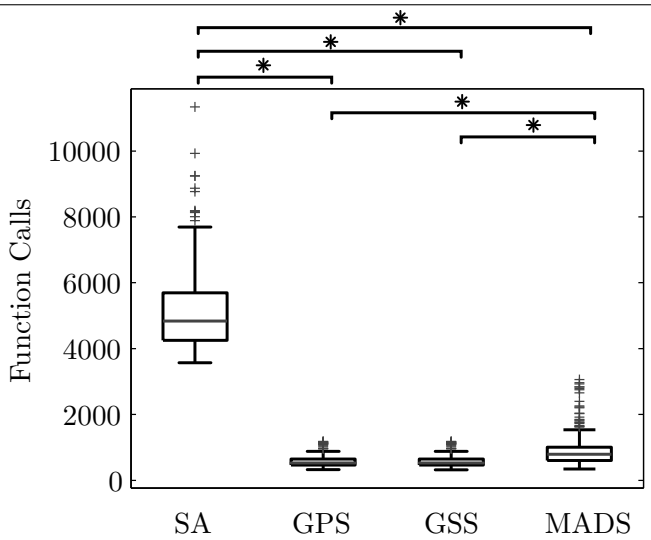


Figure A.6: Boxplots illustrating results from Table A.12.

Table A.13: Kruskal Wallis Test on Function Calls needed for user placement, 9DoF	
χ^2	1126.3863
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[2044 , 2115 , 2224]
GPS	[186 , 238 , 276]
GSS	[186 , 238 , 276]
MADS	[191 , 283 , 340]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 6.774e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs GSS	$p = 6.7724e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs MADS	$p = 7.2019e-143,\ W_A = 280152 \Rightarrow$ significant
GPS vs GSS	$p = 0.95952,\ W_A = 187026.5 \Rightarrow$ insignificant
GPS vs MADS	$p = 3.3522e-10,\ W_A = 163804.5 \Rightarrow$ significant
GSS vs MADS	$p = 2.7e-10,\ W_A = 163681.5 \Rightarrow$ significant

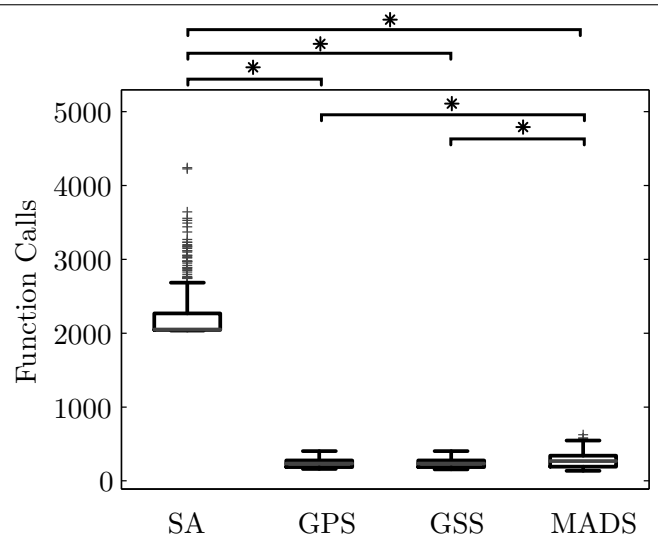


Figure A.7: Boxplots illustrating results from Table A.13.

Table A.14: Kruskal Wallis Test on Function Calls needed for geometry alignment, 9DoF	
χ^2	1126.3863
Samples	$n_{\text{SA}} = n_{\text{GPS}} = n_{\text{GSS}} = n_{\text{MADS}} = 432$
p_0	< 0.01
25 % Quantile, Median, 75 % Quantile	
SA	[1920 , 2652 , 3174]
GPS	[250.5 , 344 , 373.5]
GSS	[247 , 343 , 373.5]
MADS	[331.5 , 601 , 690.5]
Bonferroni-corrected pairwise Wilcoxon tests (testing for 95% significance)	
$\alpha_{\text{Bonferroni}}$	0.0125
SA vs GPS	$p = 8.8662e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs GSS	$p = 8.8664e-143,\ W_A = 280152 \Rightarrow$ significant
SA vs MADS	$p = 1.9051e-134,\ W_A = 277348.5 \Rightarrow$ significant
GPS vs GSS	$p = 0.87349,\ W_A = 187424.5 \Rightarrow$ insignificant
GPS vs MADS	$p = 1.3066e-39,\ W_A = 138535.5 \Rightarrow$ significant
GSS vs MADS	$p = 6.0556e-40,\ W_A = 138323 \Rightarrow$ significant

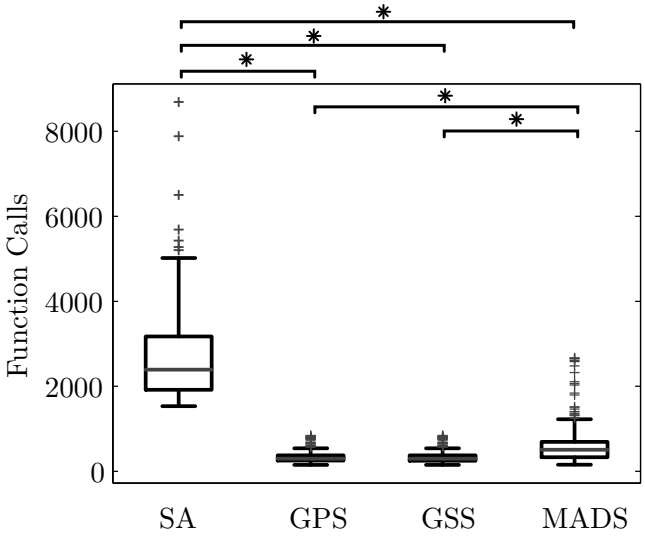


Figure A.8: Boxplots illustrating results from Table A.14.

Glossary

ANOVA	analysis of variance.
APF	annealing particle filter.
AR	augmented reality.
CAD	computer aided design.
CAVE	cave automatic virtual environment.
CR	consensus reality.
CSCW	computer supported cooperative work.
CUDA	compute unified device architecture.
DoF	degrees of freedom.
FOV	field of view.
fps	frames per second.
GPS	generalized pattern search.
GPU	graphic processing unit.
GSS	generating set search.
HMD	head mounted display.
ICP	iterative closest point.
LTMADS	lower triangle mesh adaptive search.
LUT	look-up table.
MADS	mesh adaptive search.
MCMC	Markov chain Monte Carlo.
NP-complete	non-deterministic polynomial-time complete.
NP-hard	non-deterministic polynomial-time hard.

ORTHOMADS	orthogonal mesh adaptive search.
PBAS	pixel-based adaptive segmenter.
PCL	point cloud library.
PDF	probability density function.
PSNR	pixel-wise signal-to-noise ratio.
RGB	red-green-blue.
RGB-D	red-green-blue-depth.
RTP	real-time transport protocol.
RWS	roulette-wheel selection.
SA	simulated annealing.
SDK	software development kit.
SIP	session initiation protocol.
SIRPF	sequential importance resampling particle filter.
SLAM	simultaneous localization and mapping.
SMC	sequential Monte Carlo.
SSD	sum of squared distances.
SUS	stochastic universal sampling.
VR	virtual reality.
VRPN	virtual reality peripheral network.
WAN	wide area network.
WLAN	wireless local area network.

List of Symbols

α	Scaling term used for adjusting Halton direction.
α_E	Flag for setting room alignment energy to zero.
α_{free}	Weighting term for free space energy contribution.
α_{head}	Weighting term for free space energy contribution.
α_{mins}	Weighting term for free space energy contribution.
α_{prox}	Weighting term for free space energy contribution.
α_{skew}	Weighting term for free space energy contribution.
α_{surf}	Weighting term for free space energy contribution.
α_{wall}	Weighting term for wall collision energy contribution.
Δ	Step width.
Δe	Relative error.
Δ^M	Step width used by the MADS algorithm in the SEARCH stage.
Δ^P	Step width used by the MADS algorithm in the POLL stage.
ϵ	Threshold value for set of constraints.
λ_{dmax}	Decay factor in user proximity energy term, maximum distance.
λ_{dmin}	Decay factor in user proximity energy term, minimum distance.
λ_{head}	Decay factor used in user heading energy term.

Ω	Polyhedron containing feasible solutions.
ω_3	Relative pose between two spaces, (2D: x,y,rotation), 3DoF.
ω_9	Relative pose between two spaces and users, (2D: x,y,rotation), 9DoF.
ρ	Factor used in lowering temperature between equilibria in SA.
σ	Threshold for energy function to be considered in equilibrium.
Σ_{ann}	Covariance of scattering noise in APF between annealing steps.
Σ_{IF}	Covariance of scattering noise in PF between frames.
θ_i	Angular direction of a wall line segment.
θ_i^B	Angular direction of a wall line segment.
A	Generic expression for area.
$A_{\text{free}}^{\text{A}}$	Free, observable area around user A.
$A_{\text{free}}^{\text{B}}$	Free, observable area around user B.
$A_{\text{minSurface}}$	Desired minimum size of uninterrupted work surface.
$A_{\text{minSurface}}^{\square}$	Desired minimum size of uninterrupted work surface, pixel-scaled.
A_{region}^{i}	Polygon area of a candidate work surface i.
$\mathbf{B}$	Basic Matrix.
$\mathbf{b}$	MADS first random direction.
$\mathbf{C}$	Generating matrix.
c_{ceil}	Ceiling clearance used in mapping of meshes.
c_{desViews}	Desired number of cameras observing a floor element.
c_{diff}	Consensus surface difference.
$\mathcal{C}^E$	Set of equality constraints affecting current iterate.
c_{floor}	Floor clearance used in mapping of meshes.
c_{freeNorm}	Factor used in normalizing the floor energy term in room alignment optimization.
$\mathcal{C}^I$	Set of inequality constraints affecting current iterate.
c_{max}	Consensus surface maximum height.
c_{maxDepth}	Maximum distance for which camera returns depth values.
c_{minDepth}	Minimum distance for which camera returns depth values.

c_{minViews}	Minimum required number of cameras observing a floor element.
c_{skew}	Skew direction standard deviation.
c_{UserSize}	Size of one user on the map.
c_x	Pixel per meter along x-axis.
c_y	Pixel per meter along y-axis.
$\mathcal{D}$	Polygon of possible exploratory moves around a possible solution.
$\mathbf{d}$	Exploratory move around a possible solution.
$\mathbf{D}_a$	Heading vector of user A.
d_{B}^{A}	Distance between users A and B.
$\mathbf{D}_b$	Heading vector of user B.
$\mathbf{D}_i$	Heading vector of an arbitrary user i.
d_{max}	Fading maximum distance for initializing users.
$d_{\text{max, limit}}$	Fading maximum distance for initializing users.
d_{min}	Fading minimum distance for initializing users.
$d_{\text{min, limit}}$	Limit minimum distance for initializing users.
d_{surv}	Function computing particle survival rate between annealing steps.
E	Generic Energy term used in room alignment optimization.
$\mathbf{e}$	Generic expression for an eigenvector or a unit vector.
ΔE	Change in energy.
$E_{\text{free}}(\omega)$	Partial floor energy term used in room alignment optimization.
$E_{\text{head}}(\omega)$	Partial combined user heading energy term used in room alignment optimization.
$E_{\text{head}}^{A}(\omega)$	Partial user A heading energy term used in room alignment optimization.
$E_{\text{head}}^{B}(\omega)$	Partial user B heading energy term used in room alignment optimization.
$E_{\text{mins}}(\omega)$	Partial minimum surface energy term used in room alignment optimization.
$E_{\text{prox}}(\omega)$	Partial proximity energy term used in room alignment optimization.
$E_{\text{prox}}^{\text{max}}(\omega)$	Partial proximity energy term used in room alignment optimization, maximum component.

$E^{\min}_{\text{prox}}(\omega)$	Partial proximity energy term used in room alignment optimization, minimum component.
$E_{\text{skew}}(\omega)$	Partial skew energy term used in room alignment optimization.
$E_{\text{surf}}(\omega)$	Partial surface energy term used in room alignment optimization.
$E_{\text{total}}(\omega)$	Energy term for a specific room configuration used in room alignment optimization.
$E_{\text{wall}}(\omega)$	Partial wall collision energy term used in room alignment optimization.
$\mathcal{F}$	Generic expression for a reference frame in space.
$\mathcal{F}_{\text{A}}$	Reference frame of room A.
$\mathcal{F}_{\text{B}}$	Reference frame of room B.
f_{coll}	Collision flag, binary.
f_{occl}	Occlusion flag, binary.
$\mathtt{f}^{i}_{\text{peak}}(\theta)$	Gaussian curve centered on peak of angular wall alignment histogram.
$\mathcal{G}$	Set of observed points.
$\mathbf{g}$	Single, observed point.
$\mathcal{G}(\mathbf{Z})$	Set of observed points for specific observation.
$\mathcal{G}(\mathbf{Z}_t)$	Set of observed points for specific observation and time.
$\mathbf{H}$	Householder transform of adjusted Halton direction.
$\mathcal{H}_{\theta,\text{wall}}$	Direction histogram for wall alignment.
$\mathbf{I}$	Generic expression for an identity matrix.
$\mathcal{K}$	Generic expression for a cone set.
k	k-th iteration of optimization.
k_1	Weighting function steepness constant 1.
k_2	Weighting function steepness constant 2.
k_{ann}	Temperature of a given annealing step.
k_B	Annealing parameter.
k_e	Scaling of an ellipsoid body element.
k_{IF}	Interframe scattering coefficient in PF.
$\mathcal{K}_N$	Normal cone set.
$\mathcal{K}_T$	Tangent cone set.

$\mathbf{L}$	MADS lower triangle base for direction generation.
$\mathcal{M}$	MADS Mesh.
m	Index of current annealing step.
$\mathbf{M}_{\mathrm{A}}$	Floorplan map of room A.
$\mathbf{M}^{\mathrm{A}}_{\mathrm{floor}}$	Binary map of floorspace in room A.
$\mathbf{M}_{\mathrm{AO}}$	Map of obstacles in room A, 2D.
$\mathbf{M}^{\mathrm{A}}_{\mathrm{Surface}}$	Binary map of possible work surfaces in room A.
$\mathbf{M}_{\mathrm{B}}$	Floorplan map of room B.
$\mathbf{M}^{\mathrm{B}}_{\mathrm{floor}}$	Binary map of floorspace in room B.
$\mathbf{M}^{\omega}_{\mathrm{B}}$	Floorplan map of room B after alignment.
$\mathbf{M}^{\mathrm{B}}_{\mathrm{Surface}}$	Binary map of possible work surfaces in room B.
$\mathbf{m}_{\bigcirc}$	Circular morphological kernel.
$\mathbf{M}_{\mathrm{CO}}$	Map of consensus obstacles, 2D.
$\mathbf{M}_{\mathrm{F}}$	Map of consensus free space, 2D.
$\mathbf{M}^{\mathrm{i,k}}_{\mathrm{furn}}$	Map of furniture polygon k within an arbitrary room i.
$\mathbf{M}_{\mathrm{FOBS}}$	Map of mutually observable free floor elements, 2D.
$\mathbf{M}_{\mathrm{i}}$	Floorplan map of an arbitrary room i.
$\mathbf{M}_{\mathrm{OBS}}$	Map of mutually observable floor elements, 2D.
$\mathbf{M}^{\mathrm{A}}_{\mathrm{OBS}}$	Map of observability scores in room A, 2D.
$\mathbf{M}^{\mathrm{B}}_{\mathrm{OBS}}$	Map of observability scores in room B, 2D.
$\mathbf{M}^{\mathrm{i}}_{\mathrm{OBS}}$	Map of observability scores in arbitrary room i, 2D.
$\mathbf{M}_{\mathrm{S}}$	Map of consensus surfaces, 2D.
$\mathbf{M}_{\mathrm{SOBS}}$	Map of consensus surfaces, 2D.
$\mathbf{M}_{\mathrm{vic}}$	Map around one user.
$\mathbf{M}^{\mathrm{i}}_{\mathrm{view}}$	Map of camera view cones in arbitrary room i, 2D.
$\mathbf{M}^{\mathrm{i,k}}_{\mathrm{view}}$	Map of view cone for camera k in arbitrary room i, 2D.
$\mathbf{M}^{\mathrm{i}}_{\mathrm{walls}}$	Map of furniture within an arbitrary room i.
$\mathbf{M}^{\mathrm{i}}_{\neg\mathrm{walls}}$	Map of furniture within an arbitrary room i.
$\lvert\mathbf{M}\rvert_x$	Size of map in pixel along x-axis.
$\lvert\mathbf{M}\rvert_y$	Size of map in pixel along y-axis.
$\mathcal{N}$	Integer set of dimension indices of decision variable.
n	Dimension of decision variable.

N_{APF}	Number of annealing steps.
N_C	Count of collisions in the current body pose.
N_{DoF}	Degrees of freedom of a particle.
N_e	Count of ellipsoids in the human body model.
$N_{\mathbf{g}}$	Number of observed points.
$\mathcal{N}(\mathbf{0}, \Sigma_{\text{IF}})$	Scattering of particle between observation frames.
N_p	Number of particles in a specific set.
$N_{\mathbf{r}}$	Number of sample points in the sample point set.
N_σ	Threshold count of iterations without change for breaking loop.
$N_{\text{Threshold}}$	Threshold for collisions in the current body pose.
$\mathcal{O}$	Generic expression for computational complexity.
$\mathcal{P}$	MADS Frame.
$\mathbf{p}$	Single particle.
$\mathbf{p}(x, y)$	Specific pixel within an image or map.
$\mathbf{q}$	Adjusted vector of the Halton sequence.
$\mathbf{q}_e$	Rotation of an ellipsoid body element.
$\mathcal{R}$	Set of sample points from the surface of an ellipsoid.
$\mathbf{r}$	Single sample point from the surface of an ellipsoid.
$\mathcal{S}$	Set of particles.
s_x	Size of map in meters along x-axis.
s_y	Size of map in meters along y-axis.
T	Annealing temperature.
t	Variable used to express a certain time.
T_{B}^{A}	Relative transformation between the two rooms A and B.
$T_{\text{Camera}}^{\text{Origin}}$	Transformation between coordinate system of a camera and a reference frame.
$\mathbf{t}_e$	Rotation of an ellipsoid body element.
$T_{\text{Floor}}^{\text{Origin}}$	Transformation moving the origin into the floor plane.

$\mathbf{u}$	Vector of the Halton sequence.
$\mathbf{u}_{line}$	Single sample point from the surface of an ellipsoid, line to camera.
$\mathbf{u}_{norm}$	Single sample point from the surface of an ellipsoid, direction to camera.
$\mathcal{V}$	Generic expression for a polygon, 2D or 3D.
$\mathbf{v}$	Normalized adjusted vector of Halton sequence.
$\mathcal{V}^{i}_{\text{border}}$	Boundary polygon of a candidate work surface i.
w	Function used for approximating the observation likelihood function.
$w^{\text{basic}}_{\text{r2o}}$	Partial function term for closest reference to observation distances.
w_{o2r}	Function term for closest observation to reference distances.
$w^{\text{basic}}_{\text{o2r}}$	Partial function term for closest observation to reference distances.
w_{r2o}	Function term for closest reference to observation distances.
$\mathcal{X}$	Collection of previous solutions.
$\mathbf{x}$	Generic expression for a Point.
$\mathbf{X}_a$	Position of user A.
$\mathbf{X}_b$	Position of user B.
$\tilde{\mathbf{x}}^{\text{i}}_{\text{shoulder}}$	Shoulder axis of user i.
$\tilde{\mathbf{x}}^{\text{i}}_{\text{spine}}$	Spine axis of user i.
$\mathbf{Z}$	Collection of observations.

References

[AAD08] P. Azad, T. Asfour, and R. Dillmann. "Robust real-time stereo-based markerless human motion capture". *Proceedings of the 8th IEEE-RAS International Conference on Humanoid Robots*. 2008, pp. 700–707.

[AAT13] M. Adcock, S. Anderson, and B. Thomas. "RemoteFusion: Real Time Depth Camera Fusion for Remote Collaboration on Physical Tasks". *Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*. 2013, pp. 235–242.

[Abr+09] M. Abramson, C. Audet, J. Dennis, and S. Digabel. "OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions". *SIAM Journal on Optimization* 20.2 (2009), pp. 948–966.

[AD02] C. Audet and J. Dennis. "Analysis of Generalized Pattern Searches". *SIAM Journal on Optimization* 13.3 (2002), pp. 889–903.

[AD06] C. Audet and J. Dennis. "Mesh Adaptive Direct Search Algorithms for Constrained Optimization". *SIAM Journal on Optimization* 17.1 (2006), pp. 188–217.

[Apo66] Apollonius Borelli, Giovanni Alfonso, Abraham, Abu al-Fath ibn Muhammad ibn al-Kasim ibn Fadh, al Isfahani., *Apollonii Pergæi Conicorum lib. V, VI, VII*. Florentiae: Ex typographia I. Cocchini, 1566.

[ASU] ASUSTeK Computer Inc. *Wavi Xtion*. Accessed: 2015-03-23. URL: `http://event.asus.com/wavi/product/WAVI_Xtion.aspx`.

[AVD92] P. Ariès, P. Veyne, and G. Duby. *A History of Private Life*. Harvard University Press, 1992.

[Azu97] R. T. Azuma. "A survey of augmented reality". *Presence* 6.4 (1997), pp. 355–385.

[Bak87] J. E. Baker. "Reducing bias and inefficiency in the selection algorithm". *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*. 1987, pp. 14–21.

[BCB08] O. Bernier, P. Cheungmonchan, and A. Bouguet. “Fast nonparametric belief propagation for real-time stereo articulated body tracking”. *Computer Vision and Image Understanding* 113.1 (2008), pp. 29–47.

[Ber+11] K. Berger, K. Ruhl, C. Brümmer, Y. Schröder, A. Scholz, and M. Magnor. “Markerless Motion Capture using multiple Color-Depth Sensors”. *Proceedings of the Conference on Vision, Modeling and Visualization (VMV)*. 2011, pp. 317–324.

[BFS04] I. Barakonyi, T. Fahmy, and D. Schmalstieg. “Remote collaboration using Augmented Reality Videoconferencing”. *Proceedings of Graphics Interface 2004*. 2004, pp. 89–96.

[Bil+02] M. Billinghurst, A. Cheok, S. Prince, and H. Kato. “Real world teleconferencing”. *IEEE Computer Graphics and Applications* 22.6 (2002), pp. 11–13.

[BM95] S. P. Brooks and B. J. T. Morgan. “Optimization Using Simulated Annealing”. *Journal of the Royal Statistical Society. Series D (The Statistician)* 44.2 (1995), pp. 241–257.

[Boo+99] A. Booker, J. Dennis J.E., P. Frank, D. Serafini, V. Torczon, and M. Trosset. “A rigorous framework for optimization of expensive functions by surrogates”. *Structural optimization* 17.1 (1999), pp. 1–13.

[Box57] G. E. P. Box. “Evolutionary Operation: A Method for Increasing Industrial Productivity”. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 6.2 (1957), pp. 81–101.

[BR03] C. Blum and A. Roli. “Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison”. *ACM Computing Surveys* 35.3 (2003), pp. 268–308.

[Bra86] B. Braden. “The Surveyor’s Area Formula”. *The College Mathematics Journal* 17.4 (1986), pp. 326–337.

[BSL13] T. J. Burleigh, J. R. Schoenherr, and G. L. Lacroix. “Does the uncanny valley exist? An empirical test of the relationship between eeriness and the human likeness of digitally created faces”. *Computers in Human Behavior* 29.3 (2013), pp. 759–771.

[BSS13] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

[BV09] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2009.

[Can86] J. Canny. “A Computational Approach to Edge Detection”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (1986), pp. 679–698.

[Cay11] L. Cayton. “A nearest neighbor data structure for graphics hardware”. *Proceedings of the First International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures (ADMS)*. 2011, pp. 243–251.

[CGT91] A. Conn, N. Gould, and P. Toint. “A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds”. *SIAM Journal on Numerical Analysis* 28.2 (1991), pp. 545–572.

[CGT97] A. Conn, N. Gould, and P. Toint. “A globally convergent Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds”. *Mathematics of Computation of the American Mathematical Society* 66.217 (1997), pp. 261–288.

[Cho+00] T. Choi, O. Eslinger, C. Kelley, J. David, and M. Etheridge. “Optimization of Automotive Valve Train Components with Implicit Filtering”. *Optimization and Engineering* 1.1 (2000), pp. 9–27.

[Col+77] D. E. Coleman, P. W. Holland, N. Kaden, and V. Klema. *A System of Subroutines For Iteratively Reweighted Least Squares Computations*. Working Paper 189. National Bureau of Economic Research, 1977.

[Con+96] A. Conn, N. Gould, A. Sartenaer, and P. Toint. “Convergence Properties of an Augmented Lagrangian Algorithm for Optimization with a Combination of General Equality and Linear Constraints”. *SIAM Journal on Optimization* 6.3 (1996), pp. 674–703.

[CP00] I. Coope and C. Price. “Frame Based Methods for Unconstrained Optimization”. *Journal of Optimization Theory and Applications* 107.2 (2000), pp. 261–274.

[Cru+92] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. “The CAVE: Audio Visual Experience Automatic Virtual Environment”. *Communications of the ACM* 35.6 (1992), pp. 64–72.

[CS14] K. Cho and D. Shin. “Method for displaying augmented reality image and electronic device thereof”. EP Patent App. EP20,130,160,920. 2014.

[DB92] P. Dourish and V. Bellotti. “Awareness and Coordination in Shared Workspaces”. *Proceedings of the 1992 ACM Conference on Computer-supported Cooperative Work*. 1992, pp. 107–114.

[DBR00] J. Deutscher, A. Blake, and I. Reid. “Articulated body motion capture by annealed particle filtering”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. 2000, 126–133 vol.2.

[Del34] B. N. Delaunay. “Sur la Sphère Vide”. *Bulletin of Academy of Sciences of the USSR* (1934), pp. 793–800.

[DLC08] J. Darby, B. Li, and N. Costen. “Human Activity Tracking from Moving Camera Stereo Data”. *Proceedings of the British Machine Vision Conference*. 2008.

[DR05] J. Deutscher and I. Reid. “Articulated Body Motion Capture by Stochastic Search”. *International Journal of Computer Vision* 61.2 (2005), pp. 185–205.

[DS14] D. Doolittle and S. Sarmast. “Multi-camera depth imaging”. US Patent App. 13/632,776. 2014.

[Fai74] R. Fair. “On the Robust Estimation of Econometric Models”. *NBER Chapters* (1974), pp. 117–128.

[Feb+13] A. Febretti et al. “CAVE2: a hybrid reality environment for immersive simulation and information analysis”. *Proceedings of IST /& SPIE Electronic Imaging, The Engineering Reality of Virtual Reality* 8649 (2013), pp. 13–18.

[FLD07] M. Fontmarty, F. Lerasle, and P. Danes. “Data fusion within a modified annealed particle filter dedicated to human motion capture”. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2007, pp. 3391–3396.

[FLD09] M. Fontmarty, F. Lerasle, and P. Danes. “Likelihood tuning for particle filter in visual tracking”. *Proceedings of 16th IEEE International Conference on Image Processing (ICIP)*. 2009, pp. 4101–4104.

[FP96] K. Fukuda and A. Prodon. “Double description method revisited”. *Combinatorics and Computer Science*. Vol. 1120. Lecture Notes in Computer Science. 1996, pp. 91–111.

[Gan+10] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. “Real time motion capture using a single time-of-flight camera”. *Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 755–762.

[GG84] S. Geman and D. Geman. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6 (1984), pp. 721–741.

[GH97] M. Garland and P. S. Heckbert. “Surface Simplification Using Quadric Error Metrics”. *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. 1997, pp. 209–216.

[Gir+11] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. “Efficient Regression of General-Activity Human Poses from Depth Images”. *Proceedings of the 13th IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2011, pp. 415–422.

[GK03] F. Glover and G. A. Kochenberger. *Handbook of Metaheuristics*. Springer, 2003.

[GKF04] D. Gergle, R. E. Kraut, and S. R. Fussell. “Action As Language in a Shared Visual Space”. *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*. 2004, pp. 487–496.

[GKR94] V. Granville, M. Krivanek, and J.-P. Rasson. “Simulated annealing: a proof of convergence”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.6 (1994), pp. 652–656.

[Glo77] F. Glover. “Heuristics for integer programming using surrogate constraints”. *Decision Sciences* 8.1 (1977), pp. 156–166.

[Gro+03] M. Gross et al. “Blue-c: A Spatially Immersive Display and 3D Video Portal for Telepresence”. *ACM SIGGRAPH 2003 Papers*. 2003, pp. 819–827.

[Gru94] J. Grudin. “Computer-supported cooperative work: history and focus”. *Computer* 27.5 (1994), pp. 19–26.

[GSS93] N. Gordon, D. Salmond, and A. Smith. “Novel approach to nonlinear/non-Gaussian Bayesian state estimation”. *Radar and Signal Processing* 140.2 (1993), pp. 107–113.

[Gur+12] P. Gurevich, J. Lanir, B. Cohen, and R. Stone. “TeleAdvisor: A Versatile Augmented Reality Tool for Remote Assistance”. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2012, pp. 619–622.

[Hal60] J. Halton. “On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals”. *Numerische Mathematik* 2.1 (1960), pp. 84–90.

[Hal63] E. T. Hall. “A System for the Notation of Proxemic Behavior”. *American Anthropologist* 65.5 (1963), pp. 1003–1026.

[Hal69] E. T. Hall. *The hidden dimension*. Vol. 1990. New York: Anchor Books, 1969.

[Hes69] M. Hestenes. “Multiplier and gradient methods”. *Journal of Optimization Theory and Applications* 4.5 (1969), pp. 303–320.

[HG12] M. Hofmann and D. Gavrila. “Multi-view 3D Human Pose Estimation in Complex Environment”. *International Journal of Computer Vision* 96.1 (2012), pp. 103–124.

[HJ61] R. Hooke and T. A. Jeeves. “Direct Search Solution of Numerical and Statistical Problems”. *Journal of the ACM* 8.2 (1961), pp. 212–229.

[HK14] C. Heindl and C. Kopf. *ReconstructMe*. Accessed: 2014-10-28. 2014. URL: `http://reconstructme.net/`.

[Hou58] A. S. Householder. “Unitary Triangularization of a Nonsymmetric Matrix”. *Journal of the ACM* 5.4 (1958), pp. 339–342.

[HSZ87] R. Haralick, S. R. Sternberg, and X. Zhuang. “Image Analysis Using Mathematical Morphology”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9.4 (1987), pp. 532–550.

[HTR12] M. Hofmann, P. Tiefenbacher, and G. Rigoll. “Background segmentation with feedback: The Pixel-Based Adaptive Segmenter”. *Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2012, pp. 38–43.

[Hu62] M.-K. Hu. “Visual pattern recognition by moment invariants”. *Information Theory, IRE Transactions on* 8.2 (1962), pp. 179–187.

[Hub64] P. Huber. “Robust estimation of a location parameter”. *The Annals of Mathematical Statistics* (1964), pp. 73–101.

[IB98] M. Isard and A. Blake. “CONDENSATION - Conditional Density Propagation for Visual Tracking”. *International Journal of Computer Vision* 29.1 (1998), pp. 5–28.

[Iza+11] S. Izadi et al. “KinectFusion: real-time dynamic 3D surface reconstruction and interaction”. *ACM SIGGRAPH 2011 Talks*. 2011, 23:1–23:1.

[Joh88] R. Johansen. *GroupWare: Computer Support for Business Teams*. New York, NY, USA: The Free Press, 1988.

[Kam+12] J. Kammerl, N. Blodow, R. Rusu, S. Gedikli, M. Beetz, and E. Steinbach. “Real-time compression of point cloud streams”. *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)*. 2012, pp. 778–785.

[Kat+00] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. “Virtual object manipulation on a table-top AR environment”. *Proceedings of the 1st IEEE and ACM International Symposium on Augmented Reality (ISAR)*. 2000, pp. 111–119.

[KB13] G. Kurillo and R. Bajcsy. “3D teleimmersion for collaboration and interaction of geographically distributed users”. *Virtual Reality* 17.1 (2013), pp. 29–43.

[KB99] H. Kato and M. Billinghurst. “Marker tracking and HMD calibration for a video-based augmented reality conferencing system”. *Proceedings of the 2nd IEEE International Workshop on Augmented Reality (IWAR)*. 1999, pp. 85–94.

[KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by simulated annealing”. *science* 220.4598 (1983), pp. 671–680.

[KH13] M. Kazhdan and H. Hoppe. “Screened Poisson Surface Reconstruction”. *ACM Transactions on Graphics* 32.3 (2013), pp. 1–13.

[Kim+12] K. Kim, J. Bolton, A. Girouard, J. Cooperstock, and R. Vertegaal. “TeleHuman: Effects of 3D perspective on gaze and pose estimation with a life-size cylindrical telepresence pod”. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*. 2012, pp. 2531–2540.

[Kir84] S. Kirkpatrick. “Optimization by simulated annealing: Quantitative studies”. *Journal of Statistical Physics* 34.5-6 (1984), pp. 975–986.

[KLT03] T. Kolda, R. Lewis, and V. Torczon. “Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods”. *SIAM Review* 45.3 (2003), pp. 385–482.

[KLT06] T. G. Kolda, R. M. Lewis, and V. Torczon. *A generating set direct search augmented Lagrangian algorithm for optimization with a combination of general and linear constraints*. Tech. rep. Sandia National Laboratories, 2006.

[KLT07] T. Kolda, R. Lewis, and V. Torczon. “Stationarity Results for Generating Set Search for Linearly Constrained Optimization”. *SIAM Journal on Optimization* 17.4 (2007), pp. 943–968.

[Kuh82] H. W. Kuhn. “Nonlinear Programming: A Historical View”. *SIGMAP Bull.* 31 (1982), pp. 6–18.

[Kur+08] G. Kurillo, R. Vasudevan, E. Lobaton, and R. Bajcsy. “A Framework for Collaborative Real-Time 3D Teleimmersion in a Geographically Distributed Environment”. *Proceedings of the Tenth IEEE International Symposiumon Multimedia (ISM)*. 2008, pp. 111–118.

[KW52] W. H. Kruskal and W. A. Wallis. “Use of Ranks in One-Criterion Variance Analysis”. *Journal of the American Statistical Association* 47.260 (1952), pp. 583–621.

[Lat+13] S. Latta, K. Geisner, B. Mount, J. Steed, T. Ambrus, A. Zepeda, and A. Krauss. “Multiplayer gaming with head-mounted display”. US Patent App. 13/361,798. 2013.

[Lil67] H. W. Lilliefors. “On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown”. *Journal of the American Statistical Association* 62.318 (1967), pp. 399–402.

[Lor15] H. Lorentz. “The width of spectral lines”. *Koninklijke Nederlandse Akademie van Weteschappen Proceedings Series B Physical Sciences* 18 (1915), pp. 134–150.

[LRH04] J. Lichtenauer, M. Reinders, and E. Hendriks. “Influence of the observation likelihood function on particle filtering performance in tracking applications”. *Proceedings of 6th IEEE International Conference on Automatic Face and Gesture Recognition*. 2004, pp. 767–772.

[LST07] R. Lewis, A. Shepherd, and V. Torczon. “Implementing Generating Set Search Methods for Linearly Constrained Minimization”. *SIAM Journal on Scientific Computing* 29.6 (2007), pp. 2507–2530.

[LTT00] R. M. Lewis, V. Torczon, and M. W. Trosset. “Direct search methods: then and now”. *Journal of Computational and Applied Mathematics* 124.1 - 2 (2000), pp. 191–207.

[Mai+12] A. Maimone, J. Bidwell, K. Peng, and H. Fuchs. “Enhanced personal autostereoscopic telepresence system using commodity depth cameras”. *Computers & Graphics* 36.7 (2012), pp. 791–807.

[Mai+13] A. Maimone, X. Yang, N. Dierk, A. State, M. Dou, and H. Fuchs. “General-purpose telepresence with head-worn optical see-through displays and projector-based lighting”. *IEEE Virtual Reality (VR)*. 2013, pp. 23–26.

[Mar+07] F. Markley, Y. Cheng, J. Crassidis, and Y. Oshman. “Averaging quaternions”. *Journal of Guidance Control and Dynamics* 30.4 (2007), p. 1193.

[Mar+14] M. Martinez-Zarzuela, M. Pedraza-Hueso, F. J. D. Pernas, D. G. Ortega, and M. Anton-Rodriguez. “Indoor 3D Video Monitoring Using Multiple Kinect Depth-Cameras”. *CoRR* abs/1403.2895 (2014).

[MBP95] L. Muhlbach, M. Bocker, and A. Prussog. “Telepresence in videocommunications: A study on stereoscopy and individual eye contact”. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 37.2 (1995), pp. 290–305.

[Met+53] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of State Calculations by Fast Computing Machines”. *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[MF11] A. Maimone and H. Fuchs. “Encumbrance-free telepresence system with real-time 3D capture and display using commodity depth cameras”. *Proceedings of the 10th International Symposium on Mixed and Augmented Reality (ISMAR)*. 2011, pp. 137–146.

[MGK00] J. Matas, C. Galambos, and J. Kittler. “Robust detection of lines using the progressive probabilistic hough transform”. *Computer Vision and Image Understanding* 78.1 (2000), pp. 119–137.

[Mic] Microsoft Corp. Redmond WA. *Kinect for Windows*. Accessed: 2015-03-21. URL: `http://www.microsoft.com/en-us/kinectforwindows/`.

[Mil81] R. G. Miller. *Simultaneous statistical inference; 2nd ed.* Springer Series in Statistics. New York: Springer, 1981.

[Mil97] R. Miller. *Beyond ANOVA: Basics of Applied Statistics*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 1997.

[MMK12] M. Mori, K. MacDorman, and N. Kageki. “The Uncanny Valley (reprint)”. *IEEE Robotics Automation Magazine* 19.2 (2012), pp. 98–100.

[MW47] H. B. Mann and D. R. Whitney. “On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other”. *Ann. Math. Statist.* 18.1 (1947), pp. 50–60.

[NM65] J. A. Nelder and R. Mead. “A simplex method for function minimization”. *Computer Journal* 7 (1965), pp. 308–313.

[NWB00] B. A. Nardi, S. Whittaker, and E. Bradner. “Interaction and Outeraction: Instant Messaging in Action”. *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*. 2000, pp. 79–88.

[Oye+13] O. Oyekoya et al. “Supporting Interoperability and Presence Awareness in Collaborative Mixed Reality Environments”. *Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology*. 2013, pp. 165–174.

[Pap+14] C. Papadopoulos, K. Petkov, A. Kaufman, and K. Mueller. “The Reality Deck - Immersive Gigapixel Display”. *Computer Graphics and Applications, IEEE* PP.99 (2014), pp. 1–1.

[Pet+09] B. Petit, J.-D. Lesage, E. Boyer, J.-S. Franco, and B. Raffin. “Remote and collaborative 3D interactions”. *Proceedings of the 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*. 2009, pp. 1–4.

[Pet+10] B. Petit et al. “A 3D Data Intensive Tele-immersive Grid”. *Proceedings of the International Conference on Multimedia*. 2010, pp. 1315–1318.

[Pla+10] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. “Real-time identification and localization of body parts from depth images”. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2010, pp. 3108–3113.

[PLB12] M. Papageorgiou, M. Leibold, and M. Buss. *Optimierung. Statische, dynamische, stochastische Verfahren für die Anwendung*. 3rd ed. Springer Vieweg, 2012.

[Pop07] R. Poppe. "Vision-based human motion analysis: An overview". *Computer Vision and Image Understanding* 108.1-2 (2007), pp. 4–18.

[Pow69] M. Powell. "A method for nonlinear constraints in minimization problems". *Optimization*. New York, NY, 1969, pp. 283–298.

[Pri+02] S. Prince, A. Cheok, F. Farbiz, T. Williamson, N. Johnson, M. Billinghurst, and H. Kato. "3D live: Real time captured content for mixed reality". *Proceedings of the 1st International Symposium on Mixed and Augmented Reality (ISMAR)*. 2002, pp. 316–317.

[RBB07] A. Ranjan, J. P. Birnholtz, and R. Balakrishnan. "Dynamic Shared Visual Spaces: Experimenting with Automatic Camera Control in a Remote Repair Task". *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2007, pp. 1177–1186.

[RC11] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)". *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2011.

[RL01] S. Rusinkiewicz and M. Levoy. "Efficient variants of the ICP algorithm". *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling*. 2001, pp. 145–152.

[Sal+13] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects". *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013, pp. 1352–1359.

[SBB10] L. Sigal, A. O. Balan, and M. J. Black. "Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion". *International Journal of Computer Vision* 87.1-2 (2010), pp. 4–27.

[SE03] P. J. Schneider and D. H. Eberly. *Geometric tools for computer graphics*. Morgan Kaufmann series in computer graphics and geometric modeling. Amsterdam: Boston, 2003, 1 online resource (xlv, 1009 p.)

[Ser86] J. Serra. "Introduction to mathematical morphology". *Computer Vision, Graphics, and Image Processing* 35.3 (1986), pp. 283–305.

[Sho+11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. "Real-Time Human Pose Recognition in Parts from a Single Depth Image". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2011, pp. 116–124.

[Sig+04] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. "Tracking Loose-Limbed People". *Proceedings of theIEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* 1 (2004), pp. 421–428.

[SK13] C. Schönauer and H. Kaufmann. "Wide Area Motion Tracking Using Consumer Hardware". *The International Journal of Virtual Reality* 12.1 (2013), pp. 1–9.

[SMP05] T. Svoboda, D. Martinec, and T. Pajdla. "A Convenient Multi-Camera Self-Calibration for Virtual Environments". *PRESENCE: Teleoperators and Virtual Environments* 14.4 (2005), pp. 407–422.

[Sod+13] R. S. Sodhi, B. R. Jones, D. Forsyth, B. P. Bailey, and G. Maciocci. "BeThere: 3D mobile collaboration with spatial input". *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2013, pp. 179–188.

[Sör13] K. Sörensen. "Metaheuristics - the metaphor exposed". *International Transactions in Operational Research* (2013), n/a–n/a.

[SR58] R. Sommer and H. Ross. "Social interaction on a geriatrics ward". *International Journal of Social Psychiatry* 4.2 (1958), pp. 128–133.

[Ste+13] F. Steinbruecker, C. Kerl, J. Sturm, and D. Cremers. "Large-Scale Multi-Resolution Surface Reconstruction from RGB-D Sequences". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013, pp. 3264–3271.

[Tay+01] R. M. Taylor II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. "VRPN: A Device-independent, Network-transparent VR Peripheral System". *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. 2001, pp. 55–61.

[Tay+12] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. "The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation". *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012, pp. 103–110.

[TC89] C.-H. Teh and R. Chin. "On the detection of dominant points on digital curves". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.8 (1989), pp. 859–872.

[Tea+00] S. Teasley, L. Covi, M. S. Krishnan, and J. S. Olson. "How Does Radical Collocation Help a Team Succeed?" *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*. 2000, pp. 339–346.

[Tie+14] P. Tiefenbacher, T. Gehrlich, G. Rigoll, and T. Nagamatsu. "Supporting Remote Guidance Through 3D Annotations". *Proceedings of the 2nd ACM Symposium on Spatial User Interaction*. 2014, pp. 141–141.

[Tor97] V. Torczon. "On the Convergence of Pattern Search Algorithms". *SIAM Journal on Optimization* 7.1 (1997), pp. 1–25.

[TPR14] P. Tiefenbacher, A. Pflaum, and G. Rigoll. “Touch gestures for improved 3D object manipulation in mobile augmented reality”. *Proceedings of the 13th International Symposium on Mixed and Augmented Reality (ISMAR)*. 2014, pp. 315–316.

[Tur98] G. Turk. *The interface routines for reading and writing PLY polygon files.* Accessed: 2014-10-28. 1998. URL: `http://cs.nyu.edu/~yap/classes/visual/data/ply/ply/ply.c`.

[WG01] J. Wilhelms and A. V. Gelder. *Efficient spherical joint limits with reach cones.* Tech. rep. 2001.

[WS07] D. Wagner and D. Schmalstieg. “Artoolkitplus for pose tracking on mobile devices”. *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*. 2007, pp. 139–146.

[WSF11] S. Windhager, K. Schaefer, and B. Fink. “Geometric morphometrics of male facial shape in relation to physical strength and perceived attractiveness, dominance, and masculinity”. *American Journal of Human Biology* 23.6 (2011), pp. 805–814.

[ZF09] Y. Zhu and K. Fujimura. “Bayesian 3D Human Body Pose Tracking from Depth Image Sequences”. *Proceedings of 9th Asian Conference on Computer Vision*. 2009, pp. 267–278.

Publications by Author

[Ars+08] D. Arsić, E. Hristov, N. Lehment, B. Hornler, B. Schuller, and G. Rigoll. "Applying multi layer homography for multi camera person tracking". *Proceedings of the Second ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC)*. 2008, pp. 1–9.

[Eyb+12] F. Eyben, F. Weninger, N. Lehment, G. Rigoll, and B. Schuller. "Violent Scenes Detection with Large, Brute-forced Acoustic and Visual Feature Sets". *Proceedings of the MediaEval Workshop* (2012).

[Eyb+13] F. Eyben, F. Weninger, N. Lehment, B. Schuller, and G. Rigoll. "Affective Video Retrieval: Violence Detection in Hollywood Movies by Large-Scale Segmental Feature Extraction". *PLoS ONE* 8.12 (2013), e78506.

[Hof+11] M. Hofmann, M. Kaiser, N. Lehment, and G. Rigoll. "Event detection in a smart home environment using Viterbi filtering and graph cuts in a 3D voxel occupancy grid". *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*. 2011.

[Kai+11] M. Kaiser, G. Heym, N. Lehment, D. Arsić, and G. Rigoll. "Dense point-to-point correspondences between 3D faces using parametric remeshing for constructing 3D Morphable Models". *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV)*. 2011, pp. 39–44.

[KLR11] M. Kaiser, N. Lehment, and G. Rigoll. "Dense point-to-point correspondences between 3D faces with large variations for constructing 3D Morphable Models". *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP)*. 2011, pp. 901–904.

[LAR10] N. Lehment, D. Arsić, and G. Rigoll. "Cue-Independent Extending Inverse Kinematics For Robust Pose Estimation in 3D Point Clouds". *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 2010.

[Leh+09] N. Lehment, D. Arsić, A. Lyutskanov, B. Schuller, and G. Rigoll. "Supporting Multi Camera Tracking by Monocular Deformable Graph Tracking". *Proceedings of the 11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*. IEEE Computer Society. 2009, pp. 87–94.

[Leh+10] N. Lehment, D. Arsić, M. Kaiser, and G. Rigoll. "Automated pose estimation in 3D point clouds applying annealing particle filters and inverse kinematics on a GPU". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2010, pp. 87–92.

[LER12] N. Lehment, K. Erhardt, and G. Rigoll. "Interface design for an inexpensive hands-free collaborative videoconferencing system". *Proceedings of the 11th International Symposium on Mixed and Augmented Reality (ISMAR)*. 2012, pp. 295–296.

[LKR11] N. Lehment, M. Kaiser, and G. Rigoll. "Using segmented 3D point clouds for accurate likelihood approximation in human pose tracking". *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. 2011, pp. 406–413.

[LKR13] N. Lehment, M. Kaiser, and G. Rigoll. "Using Segmented 3D Point Clouds for Accurate Likelihood Approximation in Human Pose Tracking". *International Journal of Computer Vision* 101.3 (2013), pp. 482–497.

[LMR14] N. Lehment, D. Merget, and G. Rigoll. "Creating automatically aligned consensus realities for AR videoconferencing". *Proceedings of the 13th International Symposium on Mixed and Augmented Reality (ISMAR)*. 2014, pp. 201–206.

[LTR14] N. Lehment, P. Tiefenbacher, and G. Rigoll. "Don't Walk into Walls: Creating and Visualizing Consensus Realities for Next Generation Videoconferencing". *Virtual, Augmented and Mixed Reality. Designing and Developing Virtual and Augmented Environments*. Vol. 8525. Lecture Notes in Computer Science. 2014, pp. 170–180.

[TLR14] P. Tiefenbacher, N. Lehment, and G. Rigoll. "Augmented Reality Evaluation: A Concept Utilizing Virtual Reality". *Virtual, Augmented and Mixed Reality. Designing and Developing Virtual and Augmented Environments*. Vol. 8525. Lecture Notes in Computer Science. 2014, pp. 226–236.

Supervised Students' Theses

[Age15] P. Agethen. "Markerloses Motion Capture unter Nutzung mehrerer kostengünstiger Tiefenbildkameras". Master's Thesis. Technische Universität München, 2015.

[Cos10] J. Costales. "Gesture recognition based on movement with a 3D camera". Diploma Thesis. Technische Universität München, 2010.

[Din12] Y. Dincer. "Einfluss von Feedback in einem Remote Touch Interface". Bachelor's Thesis. Technische Universität München, 2012.

[Din14] Y. Dincer. "Extraktion von Bewegungsinformation aus Punktwolken eines 3D-Lidar". Master's Thesis. Technische Universität München, 2014.

[Dre12a] D. Drexl. "Erzeugung eines Ganzkörper-Meshes aus Kinect Daten". Studienarbeit. Technische Universität München, 2012.

[Dre12b] D. Drexl. "RGB-D SLAM for 3D Modeling". Diploma Thesis. Technische Universität München, 2012.

[Dre14] M. Dreiser. "Influence of HMD integration on user performance in remote drone operation". Bachelor's Thesis. Technische Universität München, 2014.

[Eic13] J. Eichler. "Impact of User Representation in Virtual Video Conferences". Bachelor's Thesis. Technische Universität München, 2013.

[Erh12] K. Erhardt. "Berührungsfreie Gestenbasierte Interaktion in Augmented Reality Umgebungen". Bachelor's Thesis. Technische Universität München, 2012.

[Gui14] C. Guillaume. "A new generic system of diminuished reality". Master's Thesis. Technische Universität München, 2014.

[Ker11] L. Kern. "Implementierung des Wii Balance Board in die Virtual Reality Umgebungen". Bachelors's Thesis. Technische Universität München, 2011.

[Ker12] L. Kern. "Hardware Implementation of a Feature Point Detector". Diploma Thesis. Technische Universität München, 2012.

[Kna10] T. Knauer. "Farbbasiertes Hand Tracking zur Navigation in virtuellen Welten". Bachelors's Thesis. Technische Universität München, 2010.

[Kna14] T. Knauer. "Konzept zur Unterstützung von Workshops mit digitalen Medien und Technologien". Master's Thesis. Technische Universität München, 2014.

[Mei11] B. Meiler. "Hand Tracking & Grasp Detection Utilizing Depth Data and Color Images". Diploma Thesis. Technische Universität München, 2011.

[Ost11] D. Ostler. "3D Gestenerkennung in einer Multi-Kamera Umgebung". Bachelors's Thesis. Technische Universität München, 2011.

[Sal14] D. Salesski. "Implementation and Evaluation of an Optimal Compression Scheme for Point Cloud Online Transmission". Master's Thesis. Technische Universität München, 2014.

[Sch12] C. Schäfer. "Automatische Generierung von statischen Raummodellen". Studienarbeit. Technische Universität München, 2012.

[Sch13] C. Schäfer. "Darstellung und Manipulation von verteilten 3D-Arbeitsräumen für die Videokonferenz". Diploma Thesis. Technische Universität München, 2013.

[Sta14] R. Stahl. "Streaming of point clouds and image data in the CAVE". Bachelor's Thesis. Technische Universität München, 2014.

[Usa12] O. Usargil. "Integration von PTAM in ein stationäres Referenzsystem mittels ARTK". Bachelor's Thesis. Technische Universität München, 2012.

[Woh12a] B. Wohlfahrt. "Konzeptionierung eines Prüfplatzes zur Absicherung kamerabasierter Fahrerassistenzsysteme mit Regeleingriffen". Bachelor's Thesis. Technische Universität München, 2012.

[Woh12b] T. Wohlfahrt. "Visual Hull basierende Erstellung 3-dimensionaler und farbiger Objekte". Bachelor's Thesis. Technische Universität München, 2012.