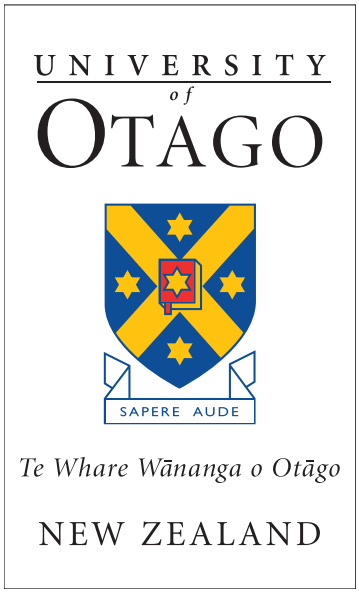

PhD Thesis

Localisation and Tracking of Stationary Users for Extended Reality

Lewis Baker
Dunedin, New Zealand, July 2020

Thesis Supervisors
Stefanie Zollmann
Steven Mills
Tobias Langlotz
Jonathan Ventura

Abstract

In this thesis, we investigate the topics of localisation and tracking in the context of Extended Reality. In many on-site or outdoor Augmented Reality (AR) applications, users are standing or sitting in one place and performing mostly rotational movements, i.e. stationary. This type of stationary motion also occurs in Virtual Reality (VR) applications such as panorama capture by moving a camera in a circle.

Both applications require us to track the motion of a camera in potentially very large and open environments. State-of-the-art methods such as Structure-from-Motion (SfM), and Simultaneous Localisation and Mapping (SLAM), tend to rely on scene reconstruction from significant translational motion in order to compute camera positions. This can often lead to failure in application scenarios such as tracking for seated sport spectators, or stereo panorama capture where the translational movement is small compared to the scale of the environment.

To begin with, we investigate the topic of localisation as it is key to providing global context for many stationary applications. To achieve this, we capture our own datasets in a variety of large open spaces including two sports stadia. We then develop and investigate these techniques in the context of these sports stadia using a variety of state-of-the-art localisation approaches. We cover geometry-based methods to handle dynamic aspects of a stadium environment, as well as appearance-based methods, and compare them to a state-of-the-art SfM system to identify the most applicable methods for server-based and on-device localisation.

Recent work in SfM has shown that the type of stationary motion that we target can be reliably estimated by applying spherical constraints to the pose estimation. In this thesis, we extend these concepts into a real-time keyframe-based SLAM system for the purposes of AR, and develop a unique data structure for simplifying keyframe selection. We show that our constrained approach can track more robustly in these challenging stationary scenarios compared to state-of-the-art SLAM through both synthetic and real-data tests.

In the application of capturing stereo panoramas for VR, this thesis demonstrates the unsuitability of standard SfM techniques for reconstructing these circular videos. We apply and extend recent research in spherically constrained SfM to creating stereo panoramas and compare this with state-of-the-art general SfM in a technical evaluation. With a user study, we show that the motion requirements of our SfM approach are similar to the natural motion of users, and that a constrained SfM approach is sufficient for providing stereoscopic effects when viewing the panoramas in VR.

Acknowledgements

Firstly, I sincerely thank my supervisors Stefanie Zollmann, Steven Mills, and Tobias Langlotz. I also sincerely thank Jonathan Ventura for his remote supervision for such a massive portion of my PhD. Each of my supervisors played a key part in guiding, motivating, and teaching me over the past three years (or more). The amount of personal time and effort they each dedicated to helping me succeed is bewildering, and I know that I could not have produced my best work without their consistent support and insight.

To my fellow students, you have all helped me immensely over the course of my PhD. Thanks to everyone in the Graphics and Vision group for their close insight into my work, the HCI group for their expertise as I eased myself into the world of XR. To Josh, Hamza, Patrick, Wei Hong, Johnny, Reuben, and others, I appreciate all the tea parties, chat, and general support. You all made this an enjoyable and worthwhile journey.

To my family, thank you for everything. I can always rely on your unconditional support and encouragement in everything I do, and my PhD journey was of course no exception. And to Sophie, you stuck by me the whole way, thank you for your unfaltering encouragement and belief in me.

Additionally, various grants have supported the research in this thesis including the National Science Foundation (Award 1464420), the New Zealand Marsden Council (Grant UOO1724), and the MBIE Endeavour Smart Ideas grant. I would also like to thank Animation Research Ltd., the Highlanders, Otago Rugby Football Union, the Forsyth Barr Stadium, and OptaPerform for their support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the recent advancement of mobile devices and head-mounted display hardware, Augmented and Virtual Reality have become popular and widely researched topics. Often referred to collectively as Extended Reality (also X-Reality and XR, Mann et al., 2018), these technologies can be used to enhance or alter a user's perception of reality, and have endless applications. While Augmented Reality (AR) aims to enhance what the user can see with superimposed visualisations, Virtual Reality (VR) refers to an experience in which the user interacts with a completely virtual environment.

Common to both technologies is the necessity of tracking user motion, which allows us to render augmented or virtual content from the correct perspective. While most previous research has focused on unconstrained tracking with 6 degrees-of-freedom (6-DoF: 3 rotational, and 3 translational), many XR use-cases such as AR browsers (Langlotz et al., 2014) and panorama capture (Xiong and Pulli, 2010) actually target *stationary users* – users that are standing or sitting in one place while performing mostly (but not purely) rotational motion (Grubert et al., 2011).

This type of motion is referred to as spherical motion in the literature (Ventura, 2016), due to the camera moving approximately along the surface of a sphere. This type of motion occurs naturally when a user is sitting and rotating a device around their body and offset by their outstretched arms, or while rotating their head while wearing a head-mounted display. Figure 1.1 demonstrates the difference between rotational, and spherical motion. This kind of small translational movement can cause problems for 6-DoF tracking approaches that rely on mapping the environment. This is due to the

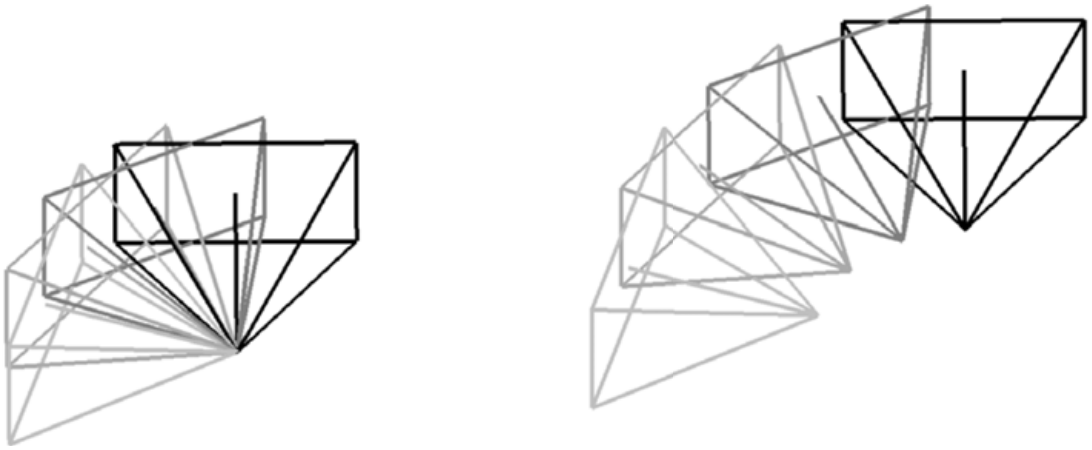

Figure 1.1 Pure rotation trajectory (left) containing no translational movement, compared to a more realistic spherical movement (right).

fact that these methods require significant translational baselines to reliably triangulate corresponding points between two views. For this reason, these types of tracking methods can be unreliable when used to track stationary users in large environments, where the scale of translational movement is small compared to the distances to the objects in view.

While there has been research into rotation-only 3-DoF tracking (e.g. by ignoring small translational movements, Langlotz et al., 2014), these methods have limitations when applied to XR. Firstly, small translational movements can cause large disparities due to parallax when some objects in the environment are up close, causing instability in small indoor scenarios. Secondly, rotation-only tracking does not permit creating visualisations with realistic motion parallax. While rotation-only tracking can be used to stitch panoramas (Xiong and Pulli, 2010), this type of tracking forfeits the opportunity to create more immersive stereoscopic panoramas (Richardt et al., 2013) as it ignores translation altogether.

The spherical motion constraint introduced by Ventura (2016) offers a convenient middle-ground between these two types of tracking systems by approximating stationary motion without neglecting translation altogether. In this thesis, we argue that neither 6-DoF nor rotation-only 3-DoF tracking approaches are suitable for stationary applications. Therefore, we investigate the opportunities of applying the spherical constraint to tracking and localisation for XR, particularly for stationary applications.

1.1 Stationary Users

In many XR use-cases, it can be assumed that the user will be stationary, either seated or standing. The assumption that a camera will be purely rotating has been made many times in the literature, especially in outdoor scenarios (Kurz et al., 2013). This assumption has been made for both VR (Qian et al., 2016), and AR (Langlotz et al., 2014; Yu et al., 2016). In all these works, it is assumed that the user is purely rotating, thus stationary. This assumption can be represented by a camera rotating around its focal point (as depicted in Figure 1.1, left), which is not always realistic.

While this assumption is often sufficient, it is only an approximation of real user motion. In the real world, it would be very unnatural for a user to rotate a device in this way. Usually, the camera would be moving in position, as well as rotating. For example, a seated VR user may move their head left-to-right approximately creating an arc trajectory (as shown in Figure 1.1, right). Usually, the rotation-only assumption is made for large-scale outdoor use-cases where the amount of translational motion is small when compared to the scale of the environment. In these situations, the amount of visual misalignment introduced by the approximation is negligible, and acceptable for casual usage.

Except for some hybrid systems, we note that past research on tracking for XR has focused primarily on one of two assumptions:

1. The user needs to move around with no restriction and must perform sufficient translational movement.

2. The user is likely to stand in one place and mostly rotate their body around a fixed point.

The former assumption can be applied to an unconstrained 6-DoF tracking system, which often results in poor performance or initialisation failure when the user does not move with enough translation. The second assumption can be applied to a 3-DoF tracking system which only tracks rotations. This is simpler and not prone to initialisation problems but can result in poor accuracy due to the oversimplification of the user's motion.

Part of this thesis works toward bridging the gap between these two types of systems, specifically targeting stationary users – users who are standing or sitting in one place, performing mostly rotational movement. We argue that the latest tracking research has

primarily focused on unconstrained tracking (Taketomi et al., 2017), while stationary use cases are still very common.

This thesis aims to investigate new approaches to tracking that reflect the actual motion of stationary users. We aim to alleviate the issues caused by small translational movement in 6-DoF tracking systems, without having to give up motion parallax completely as is the case with rotation-only 3-DoF tracking.

1.2 Stationary Applications

As previously introduced, we aim to investigate the problems that arise when tracking or localising for stationary users. In this thesis, we address these problems from the perspective of two applications: AR visualisations for sport spectators, and stereo panorama generation. Details of the requirements for these applications are introduced in this section.

1.2.1 Localising Sport Spectators

In the previous section, we introduced the concept of stationary users, and the issues that arise when applying typical tracking methods to them. However, in many AR applications, tracking alone does not provide sufficient information to produce visualisations that are correctly aligned to the view of the user. Most tracking methods provide pose within their own co-ordinate system, with ambiguous scale. An example of this kind of tracking system is simultaneous localisation and mapping (SLAM), which often assumes that one frame of its input is positioned at the origin, with no rotation (Mur-Artal et al., 2015).

Localisation is the process of determining the position and orientation of a camera in a known environment. By localising against a common reference model (Reitmayr and Drummond, 2006), multiple users can experience a shared virtual environment. In many AR applications, it can be assumed that a model of the environment is readily available. This could come in the form of 3-D reconstructions produced through photogrammetric techniques like structure-from-motion (SfM) (Snavely et al., 2006; Schönberger and Frahm, 2016), which may also be geo-referenced (Schmalstieg et al., 2011; Zollmann et al., 2014) to provide more context.

An example of such an application would be the Augmented Reality Spectator (AR-Spectator) first introduced in Zollmann et al. (2019). In this example, sport spectators

Figure 1.2 Mock-up example of the Augmented Reality Spectator application. Registration of virtual content, such as the heatmap displayed on the mobile device requires knowing precisely the position and orientation of the device with respect to the field. This information can be obtained through localisation to a prior model. *Original source (modified): CC0 1.0 Universal (CC0 1.0) Public Domain Dedication.*

in a stadium are shown visualisations and statistics relevant to the event overlaid with their surroundings. Certain visualisations such as informative heatmaps may be aligned accurately to the pitch as shown in Figure 1.2. However, aligning these visuals correctly to their surroundings would require localising the user and tracking their motion, which is challenging in large environments.

Another application involving localisation could be the outdoor map example depicted in Figure 1.3, where users are localised and tracking while exploring the city outdoors. Users are able to see visualisations and annotations related to their actual location, registered in 3-D and overlaid with their surroundings. Using an existing model or map of the area, visualisations such as place names can be placed over the corresponding buildings, helping users find their place of interest.

While some localisation systems rely on sensor data such as GPS or inertial sensors, many localisation techniques require an existing model of the environment. In many applications, such as AR browsers, it is feasible that the developers of such applications could prepare such a model in advance and distribute this to the users when required. Furthermore, the SLAM-based tracking systems often produce 3-D reconstructions as an intermediate step in the tracking process. These reconstructions could also be distributed for multi-user localisation.

Figure 1.3 Example of an outdoor AR application. A user may look through an AR browser on their mobile device in order to get directions to a specific location. To correctly display meaningful visualisations such as place names for each building, localisation is necessary.

Assuming an existing model of the environment is available, it is often unlikely that the model corresponds precisely to the actual environment. For example, lighting conditions can change, smaller structures may have moved, and some objects might be missing entirely. These dynamic environments can cause issues for both localisation and tracking. In much of the recent literature, localisation approaches apply the concept of feature matching to assist with pose computation (Sattler et al., 2011). Typically, feature matching aims to find matches of visually similar points between images or (in the case of localisation), 3-D models. In dynamic environments, appearance and features are likely to change, which makes matching difficult.

For this reason, we look into SoftPOSIT (David et al., 2003), an alternative approach to localisation which relies on the geometry of line features rather than appearance, as well as typical feature-based methods that use appearance through feature descriptors. In this thesis, we investigate existing localisation and tracking approaches, and how they can be applied to stationary users in the context of the ARSpectator, as well as developing new tracking techniques that specifically target spherical stationary motion.

Figure 1.4 An example of a stationary user capturing a panorama with their mobile device. Panorama capture is a common feature integrated into most modern smartphones. Left: a user capturing a panorama with a mobile device. Right: The resulting panorama.

1.2.2 Panorama Generation

While tracking is often considered in its application to viewing XR content, it is also important in the *creation* of VR content. Nowadays, as demonstrated in Figure 1.4, smartphones provide the capability to capture 360° panoramic photographs by rotating the device in an outward-facing circle. This type of motion, while being used in a different context, is very similar to the trends we see in AR applications.

Many approaches to producing panoramas work by stitching video frames from the camera, using camera tracking to compute the orientations for each frame. The resulting panoramas can then be viewed in VR by mapping the panoramas to a 3-D surface such as a cylinder. In both the creation and viewing stages of this process, it is common to assume that the user is purely rotating, even when there is a small translational component to their motion as well.

Recently, there has been a surge in popularity of more immersive panoramas. As typical panorama tracking methods apply 3-DoF rotation tracking to estimate the camera motion, these methods result in flat panoramas which, even when viewed in VR, do not display stereoscopic or depth cues to the user. Recent research such as MegaParallax (Bertel et al., 2019) have looked into tracking using traditional 6-DoF methods which are able to exploit this almost rotational movement that users naturally create when moving a device in a circle around their bodies.

The resulting 6-DoF tracking allows for synthetic creation of novel views for each eye, allowing the user to perceive stereoscopic depth. But as the underlying structure-from-motion (SfM) systems that compute the camera positions also rely on significant translational movement, they too suffer from similar issues to 6-DoF SLAM-based

methods. This further motivates the necessity of more work in tracking stationary users where the translation is small, but significant. For this reason, we argue that current SfM techniques are not suitable for stereo panorama generation, and that a spherical constraint could provide a more robust system for stereo panorama generation.

1.3 Contributions

In this section, we list the main contributions of this thesis toward the problems of localisation, tracking, and panorama generation for stationary users. We evaluate a range of localisation methods on large stadium models. We also address the gap between full 6-DoF tracking and rotation-only 3-DoF tracking in the context of tracking for AR, and panorama generation for VR. The main contributions of this thesis are listed below, along with their corresponding chapters:

- Various structure-from-motion (SfM), and stationary motion datasets captured in large outdoor spaces (Chapter 3). As there are few datasets of this type available, the results we present using our own datasets provide insight into how state-of-the art approaches perform under stationary conditions.
- Application of a geometry-based pose estimator SoftPOSIT (David et al., 2003) to a new type of 3-D model produced with SfM (Chapter 4), as it has not yet been applied to complex SfM models to our knowledge. Though this method shows desirable properties for localisation in dynamic environments, we found this method to be better suited to simpler models.
- Comparison of state-of-the-art localisation methods, applied to the ARSpectator problem (Chapter 4). Methods such as active search (Sattler et al., 2012a), ESAC (Brachmann and Rother, 2019), and Bag-of-Words (Gálvez-López and Tardós, 2012) have been shown to perform reliably on a limited range of outdoor datasets. Here, we show their applicability to localisation in large sport stadia.
- Implementation and evaluation of a novel tracking system based on SLAM principles with a spherical motion constraint (Ventura, 2016) (Chapter 5). We show that our novel SLAM system can perform more reliably in large open spaces than state-of-the-art ORB-SLAM. We also contribute:
  - A new method for efficient keyframe selection for spherical SLAM systems, which can replace standard keyframe culling mechanisms (keyframe sphere).

  - Implementation of a method for absolute camera pose estimation using 2 point correspondences and a spherical constraint, based on a solution by Jonathan Ventura.

- Application of the spherical motion constraint to generating stereoscopic panoramas, with evaluation and a user study (Chapter 6). We show that offline tracking based on spherical SfM (Ventura, 2016) combined with flow-based blending (Richardt et al., 2013) produces high quality stereo panoramas, comparable to state-of-the-art MegaParallax (Bertel et al., 2019) with faster computation. We also:

  - Demonstrate the unsuitability of state-of-the-art SfM systems for reconstructing circular videos.
  - Find that users were able to perceive stereo better with our system compared to traditional panoramas in VR, while finding no additional difficulty with the capture process compared to normal panorama capture.

1.4 Thesis Outline

In this chapter, we provided a brief introduction to XR, tracking, and localisation. We note that while much research on tracking has focused on unconstrained motion, the stationary use case is still prevalent in XR scenarios such as the ARSpectator and panorama generation and is deserving of further research. In this thesis we focus on these stationary use cases, and investigate the feasibility of localisation in sport stadia, as well as investigating constrained tracking for AR users, and panorama generation. In this section, we provide an overview of the structure of the thesis, and details of which areas have been published already.

1.4.1 Collaboration and Publications

This thesis is based on work which has already been published in peer-reviewed conference papers. Those papers, and their corresponding chapters in this thesis are listed below, along with a description of my own involvement in those publications.

- Chapter 4 is based on the **paper:** Baker, L., Zollmann, S., Mills, S., & Langlotz, T. (2018, November). SoftPOSIT for Augmented Reality in complex environ-

ments: Limitations and challenges. In *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)* (pp. 1-6). IEEE.

I completed the implementation, data capture, experimentation, and most of the writing in this publication. The stadium dataset in this publication was captured by Steven Mills. My co-authors provided verbal assistance and discussion of implementation, suggestions for experiments, and some writing.

- Chapter 5 is based on the **paper:** Baker, L., Ventura, J., Zollmann, S., Mills, S., & Langlotz, T. (2020, March). SPLAT: Spherical Localization and Tracking in Large Spaces. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (pp. 809-817). IEEE.

  In this publication, I completed most of the design and implementation of the SLAM system, data capture, analysis except for the RealSense comparison, and writing. The final implementation was built on top of an SfM framework and a PnP solution by co-author Jonathan Ventura. My co-authors provided consultation and verbal suggestions throughout, as well as some writing.

- Chapter 6 is based on the **paper**, which received an honourable mention: Baker, L., Mills, S., Zollmann, S., & Ventura, J. (2020, March). CasualStereo: Casual Capture of Stereo Panoramas with Spherical SfM. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (pp. 782-790). IEEE (**Best Conference Paper - Honorable Mentions Award**).

  Early results from this collaboration resulted in the **poster**: Baker, L., Zollmann, S., & Ventura, J. (2019, March). Spherical SfM for Casual Capture of Stereo Panoramas. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (pp. 846-847). IEEE.

  I was involved in the implementation of the stereo panorama system, technical evaluation, conducting the user study, synthetic testing, and dataset capture. The system implementation was directed mostly by co-author Jonathan Ventura, and the writing was a collaborative effort among myself and co-authors.

- **Technical brief**: Zollmann, S., Langlotz, T., Loos, M., Lo, W. H., & Baker, L. (2019). ARSpectator: Exploring augmented reality for sport events. In *SIGGRAPH Asia 2019 Technical Briefs* (pp. 75-78).

The content of this technical brief is not included in this thesis, as I was not the first author. My main contribution was the tracking results. The ARSpectator project as a whole is used as inspiration for my own work in Chapters 4 and 5.

- **Journal paper**: Moore, A., Daniel, B., Leonard, G., Regenbrecht, H., Rodda, J., Baker, L., Ryan, R. & Mills, S. (2020). Comparative usability of an augmented reality sandtable and 3D GIS for education. *International Journal of Geographical Information Science*, 34(2), 229-250.

  The content of this article is not included in this thesis, as I was not the first author. I was involved in building, development and calibration of the AR sandbox, as well as operating the sandbox during the user study.

1.4.2 Summary of Chapters

The structure of this thesis is outlined as follows. Chapter 2 covers the background and related work for the topics of localisation, tracking, and stereo panorama generation in the context of stationary users. As both of our use-cases target stationary users, we discuss limitations of existing datasets, and how we acquire our testing datasets in Chapter 3. Chapter 4 covers the localisation problem in the context of the ARSpectator, and our experiments with geometry-based, as well as state-of-the-art appearance-based localisation methods with our own datasets including two 3-D models of sports stadia.

In Chapter 5, we cover the topic of tracking, and introduce a novel spherically constrained SLAM technique that specifically targets stationary users. In Chapter 6, we explore an application of offline spherical tracking and SfM for the purpose of Virtual Reality with stereo panoramas. Finally, in Chapter 7, we present a summary of the findings and conclusions, and possibilities for future work.

Chapter 2

Related Work

In the previous chapter, we briefly introduced the topics of tracking, localisation, extended reality, and panoramas. In this chapter, we look in detail at the literature in these topics to identify specific areas and problems that deserve further research. This chapter will be divided into sections with essential background topics first, followed by sections on the more specific topics of interest.

First, in Section 2.1, we cover the background on Augmented and Virtual Reality as this provides the motivation and details of the technical requirements for such systems. In Section 2.2, we cover the general background on camera geometry, and the required knowledge for understanding the tracking and localisation approaches for XR. As many tracking and localisation approaches are based on known *a priori* reconstructions, we provide background on reconstruction in Section 2.3. Section 2.4 covers the localisation problem and related work, as localisation is key to providing AR content in a shared or known environment.

Section 2.5 covers the various approaches to tracking single cameras, in order to provide real-time updating of virtual content. As deep learning has become a popular topic of research in all areas of computer vision, in Section 2.6, we discuss the current state-of-the-art methods that apply deep learning to localisation and tracking. Another key application of stationary users is panorama capture, so we cover the related work in

immersive panorama generation in Section 2.7. Finally, in Section 2.8, we provide a summary and discuss the gaps in the research and how we aim to address them in this thesis.

2.1 Extended Reality

Since at least the early 1990's (Azuma, 1993), Augmented Reality (AR) has been widely researched (Schmalstieg and Hollerer, 2016). In recent years, there have been great bounds in its enabling algorithms and hardware, making AR increasingly accessible to mainstream consumers [1]. AR enables people to see and experience an enhanced version of our real environment through the use of virtual computer graphics, spatially aligned to the real world. Naturally, there are endless applications for this technology, such as navigation assistance (Narzt et al., 2006), to games such as Pokémon GO [2] visualised through the camera of a smartphone, to name just a few.

In its early years, AR was much less accessible than it is today. Early prototypes included expensive hardware with heavy head-mounted displays and backpack systems, such as ARQuake (Piekarski and Thomas, 2002). Referred to as Video See-through Displays (Rolland and Fuchs, 2000), these types of devices would often use cameras to capture the outside world, and display the images augmented with virtual content inside a closed off head-mounted display. Nowadays, Optical Head-mounted Displays are becoming more accessible. These devices incorporate semi-transparent screens embedded in glass lenses, allowing the user to naturally see their real environment, with only the augmented virtual content being artificially rendered. Devices such as the Microsoft HoloLens (Microsoft Corporation, 2016), and Magic Leap (Magic Leap, Inc., 2018) offer this type of AR hardware, and similarly the Oculus Quest is available for VR (Oculus VR, 2019). While initially releasing limited development versions, the latest headsets are currently available to developers and enthusiast consumers.

With the increasing ubiquity of mobile devices, mobile AR is one of the most promising approaches to bring AR experiences to the everyday consumer (Henrysson and Ollila, 2004). In mobile AR, users leverage the processing capabilities of their mobile devices (such as a smartphone or tablet) in order to experience AR (Höllerer and Feiner, 2004). Modern smart devices often have high-resolution cameras, fast multi-core processors,

[1] Ryan Donovan: *https://www.information-age.com/augmented-reality-mainstream-123484406/*. August 2019.

[2] Niantic: *https://pokemongolive.com/en/*. Accessed August 2020.

and sufficient memory to perform the computation required to produce AR content. The main disadvantage is that the user must use the display on their device as a window, and all AR content is rendered from the perspective of the mobile device's camera, rather than the perspective of the user (Langlotz et al., 2014).

2.1.1 Mobile Augmented Reality

Mobile AR is desirable due to its convenience, and affordability. However, enabling AR on this kind of hardware brings its own technical challenges. Headsets such as the Magic Leap are designed specifically for Augmented Reality, and as such, feature many different sensors that can be used to track the motion of the devices (Koulieris et al., 2019). In the case of mobile AR, it is often assumed that the only sensor available is a camera, and in some cases an Inertial Measurement Unit (IMU) (Li et al., 2017). This raises the question of how to track the viewing direction of mobile devices with such limited sensory data.

Recently, mobile operating systems provide their own frameworks that support AR application development, namely Google's ARCore for Android (Google LLC, 2018), and Apple's ARKit for iOS devices (Apple, Inc., 2018). These frameworks have resulted in a surge of AR application development by abstracting some of the complex algorithms required to precisely align virtual content in real spaces. Behind the programming interface, these frameworks apply highly researched methods such as simultaneous localisation and mapping (SLAM) to track the movement of the camera, and generate a map of the user's environment (Davison et al., 2007). This information can then be applied to render virtual content in the user's space (Klein and Murray, 2007).

While mobile AR is a relatively new concept (Höllerer and Feiner, 2004) compared to AR as a whole (Azuma, 1993), it has existed long enough for common usage patterns to appear. While Optical See-through AR systems such as Magic Leap allow users freedom to move their whole bodies, there are many application scenarios that do not require much movement from the user. These applications can include seated experiences such as sport spectating (Zollmann et al., 2019) or outdoor tourism scenarios (Narzt et al., 2003), and outdoor AR annotations (Langlotz et al., 2014), and can be implemented effectively with the more ubiquitous mobile AR hardware.

2.1.2 Requirements for Extended Reality

Azuma (1997) says that an AR system has three defining characteristics: it must "combine real and virtual", be "interactive in real time", and be "registered in three dimensions". However, the underlying requirements for both AR and VR have a large amount of overlap. For this reason, these systems are often referred to collectively as Extended Reality (XR) (Mann et al., 2018). The following are some key components that are often involved in the technical implementation of XR systems:

- *Pose estimation* is the process of determining the position and orientation of a camera, usually defined as a rotation (commonly either a 3×3 matrix or a quaternion) and a translation (3×1 vector).
- *Tracking* involves determining the pose of a moving camera in real-time. Usually this involves pose estimation at a base level, but also leveraging similarities between sequential frames of video.
- *Localisation* is interested in determining the pose of a camera in a known environment, such as an existing model.

These terms were briefly introduced in the previous chapter but are essential background to this thesis. To that end, the related work in each of those areas is covered in detail in this chapter. We first cover pose estimation (Section 2.2) and reconstruction (Section 2.3), followed by localisation (Section 2.4) then tracking (Section 2.5). We also discuss the latest deep-learning approaches to these problems (Section 2.6). Then we discuss the related work in panorama generation, as it is closely tied to our focus of stationary users (Section 2.7), and finally summarise the literature findings (Section 2.8).

2.2 Camera Pose

Camera pose can be defined as the position and rotation of a camera within some co-ordinate system. In some cases, camera poses are relative to an unknown origin in an encompassing global co-ordinate system (such as relative to another camera). In other cases, the camera pose can be computed as absolute, and completely defines the cameras position and orientation within the global co-ordinate system.

Given an image or photograph, computing or estimating its camera pose is a fundamental topic in computer vision, and has many applications such as photogrammetry,

tracking, and 3-D reconstruction. Knowing the precise position and orientation of the camera allows us to learn more about the scenes we are looking at, such as the structure and shape of the surfaces, or the distance to an object of interest.

In AR, camera pose can be used to align the virtual camera of a graphical rendering system with a real camera. When this is applied to video in real-time, it can create the illusion that virtual objects are integrated into a real environment. These factors combined form the foundations for AR, and a system that implements these will meet the criteria for AR defined by Azuma (1997).

To estimate camera pose, the projective geometry of the camera must be modelled. A typical camera will refract incoming light rays through a lens which converge on an image plane (or commonly a CMOS sensor). Such as in Hartley and Zisserman (2003), cameras are often approximately modelled as pinhole cameras using the following equation that defines its projective geometry:

$$\mathbf{x} = \mathsf{P}\,\mathbf{X}. \tag{2.1}$$

The matrix P can be used to transform homogeneous 3-D points $\mathbf{X}$ in the world into homogeneous 2-D co-ordinates $\mathbf{x}$ in the camera's reference frame. P is a projection matrix which can be further decomposed, resulting in the camera pose.

The projection matrix P can be represented by

$$\mathsf{P} = \mathsf{K}\,[\,\mathsf{R}\,|\,\mathbf{t}\,], \tag{2.2}$$

where K is a 3×3 matrix representing the internal characteristics of the camera (such as its focal length, and principal point), R is a 3×3 orthonormal matrix representing the rotation, and $\mathbf{t}$ is a vector representing the translation of the camera.

The matrix K, which describes the internal parameters of the camera, can be referred to as the intrinsic camera matrix, or calibration matrix. It is a 3×3 matrix of the form

$$\mathsf{K} = \begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

where f_x and f_y represent the horizontal and vertical components of the focal length, s represents any pixel skew, and (p_x, p_y) represents the pixel co-ordinate of the camera's principal point. In most mobile devices, it is common to have approximate values available through the device's factory calibration, and in other cases these values can

be acquired relatively easily through multiple images of a calibration pattern (Zhang, 2000). Due to the simplicity of calibration, we assume this matrix K to be prior knowledge where applicable in this thesis.

2.2.1 Perspective-n-Point Pose

Perspective-n-Point (PnP) pose estimation is the process of computing the camera pose given n 3-D points in some co-ordinate system and their corresponding 2-D image locations. It differs from relative pose estimation, in that the goal is to compute pose in a global reference frame defined by some known structure in the environment. The term PnP was coined by Fischler and Bolles (1981), who in the same paper introduced Random Sample Consensus (RANSAC) for solving the problem.

The PnP problem has many solutions, with different approaches requiring different numbers for n. Some PnP methods are capable of computing the entire projection matrix P (Triggs, 1999; Hartley and Zisserman, 2003), while others will assume the parameters of K to be known in advance, or assume the 2-D co-ordinates to be normalised (Fischler and Bolles, 1981).

A visual representation of the Perspective-3-Point problem is shown in Figure 2.1. A camera with its centre of projection originated at $\mathbf{O}^c$ observes the projections of three points $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$ to $\mathbf{x}_1^c$, $\mathbf{x}_2^c$, and $\mathbf{x}_3^c$ respectively. The objective is to find the pose R and $\mathbf{t}$ that satisfies these projective constraints. Some approaches may first determine the depths λ_i of these points from the camera's origin, as an intermediate step to finding the transformation $[\,\mathsf{R} \mid \mathbf{t}\,]$.

Hartley and Zisserman (2003) suggest a combination of a linear solution using the direct linear transform (DLT) approach to initialise P combined with the Levenberg-Marquardt Algorithm (LMA, Moré (1978)) to minimise the geometric error. This geometric error is essentially the distance between projections of 3-D points $\mathbf{X}_i$ given the current state of P using the relationship of Equation 2.1, and their actual observed locations $\mathbf{x}_i^c$ in the image plane of camera $\mathbf{C}$.

Other approaches have been successful in solving PnP problems with smaller sets of points, such as Gao et al. (2003), which require only three correspondences. However, in practice, it is possible to attain many more correspondences through feature matching. As feature matching approaches can produce incorrect matches, combining PnP solvers within a RANSAC scheme is a common approach to handling these outliers.

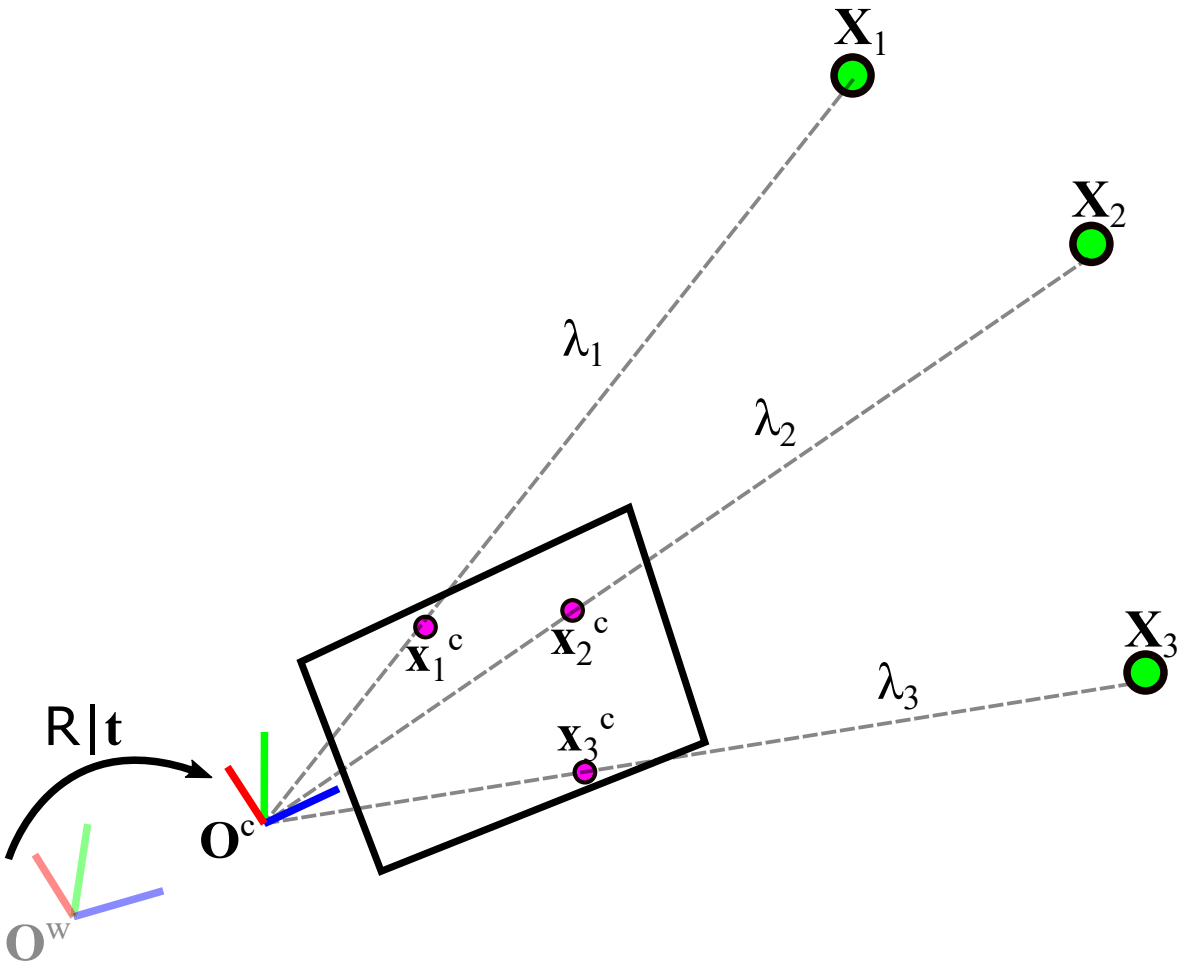

Figure 2.1 Depiction of the Perspective-3-Point (P3P) problem. A camera centred at $\mathbf{O}^c$ observes projections $\mathbf{x}_i^c$ of corresponding 3-D points $\mathbf{X}_i$. The objective of the P3P problem is to find the camera pose $[\,\mathsf{R}\,|\,\mathbf{t}\,]$ that supports those correspondences.

2.3 Reconstruction

As outlined in the previous section, pose estimation approaches often rely on known structure or 3-D points of an environment. Thus, reconstructing this information in the first place is an important aspect of many tracking and localisation systems. 3-D reconstruction is any method of recovering 3-D structure from a scene using observations from sensors such as simple cameras, or advanced depth sensors such as the Kinect structured light camera (Microsoft Corporation, 2010), or LIDAR sensors (Intel Corporation, 2019). In this section, we cover reconstruction from cameras, as they are the most abundant sensor and are essentially guaranteed to be present on all modern mobile devices.

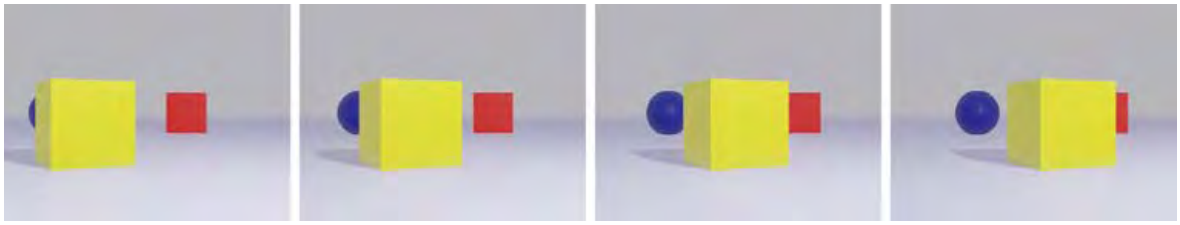

Figure 2.2 Illustration of motion parallax and disparity caused by a translating camera. As the camera moves, the yellow cube begins disoccluding the blue sphere, then occludes the red cube due to differences in depth and disparity.

Reconstructions are often used in localisation (covered below in Section 2.4), as in many cases a prior model of the environment to localise in is assumed to be available (Sattler et al., 2011). In the case of tracking, many methods also map the environment (Davison et al., 2007). In the case of Parallel Tracking and Mapping (PTAM), an initialisation phase reconstructs a number of points from two distinct initial views via triangulation, and this reconstruction forms the basis of the co-ordinate system to be tracked within (Klein and Murray, 2007).

In the case of reconstruction from cameras, two distinct views of the scene with significant translational movement are required. By translating the camera, different objects in a scene exhibit different amounts of parallax or disparity across the two views (i.e. nearby objects will show larger amounts of disparity than far objects, as illustrated in Figure 2.2). This basic concept is fundamental to how humans can perceive depth through binocular vision. In the same way, depth can be computed with reasonable accuracy through matching points between two distinct views and performing triangulation (Hartley and Sturm, 1997).

A camera which is purely rotating about its focal point will not be able to create reliable views for triangulation, as there is no parallax. This is also the case for camera movements with small translational components, and the severity of this effect is proportional to the scale of the environment. In work by Paz et al. (2008), a tracking method is introduced which relies heavily on reconstruction from small-baseline stereo cameras (e.g. SLAM, covered below in Section 2.5), and through simulated experimentation, the authors suggest that stereo triangulation is reliable to depths up to 40 times the baseline of the stereo camera (e.g. approximately 5 m with a 12 cm baseline).

Reconstruction from two images is a fundamental problem in computer vision (Szeliski, 2010; Hartley and Zisserman, 2003), and has various applications from outdoor localisation (Li et al., 2010) to 3-D content creation (Nguyen et al., 2013, 2014). An in-

termediate step for computing a 3-D point reconstruction from two images is to first compute the relative pose between the two images. Hartley and Zisserman (2003) discuss a wide variety of methods for estimating relative pose, such as by decomposition of a fundamental matrix F, computed with the normalised 8-point algorithm (Hartley, 1997).

Relative pose can also be computed through decomposing an essential matrix E computed with the 5-point algorithm such as that described by Nistér (2004). The difference in these approaches is that the 8-point algorithm is a linear solution that does not require knowing the intrinsic parameters K of the camera, while the 5-point algorithm requires known K matrices for each view with a non-linear solution. There have also been advancements in these algorithms that can exploit the known orientation of features such as SIFT (Lowe, 2004), to add constraints to the essential or fundamental matrices, reducing the number of correspondences required (Mills, 2018).

Finding the correspondences required for these algorithms is a challenge in itself, and is commonly handled through matching of feature descriptors such as SIFT (Lowe, 1999, 2004), or ORB (Rublee et al., 2011). While SIFT and some of its variations are widely used in many reconstruction pipelines (Snavely et al., 2006; Schönberger and Frahm, 2016), ORB is often found to show similar matching performance with faster computation time due to more compact binary representations (Khan et al., 2015). Alternatively, after feature descriptors have been computed, corners can be tracked through video frames (Shi et al., 1994) using methods such as optical flow (Bouguet et al., 2001) as depicted in Figure 2.3. Once relative pose is estimated, and a set of corresponding observations between the two distinct frames is established, the points can be triangulated to create an initial reconstruction.

2.3.1 Spherical Structure-from-Motion

In this section, we provide details of spherical motion which was first introduced by Ventura (2016). Here, we provide the required background explaining how the methods in this work enforce a spherical constraint on the computation of the essential matrix E for relative pose estimation. As previously mentioned, relative pose estimation through computation and decomposition of an essential matrix is a key step for many reconstruction systems including both SfM and SLAM systems, and this work in particular is used as a basis for much of the contributions of this thesis.

Spherical motion is an approximation of the type of motion that occurs when a user

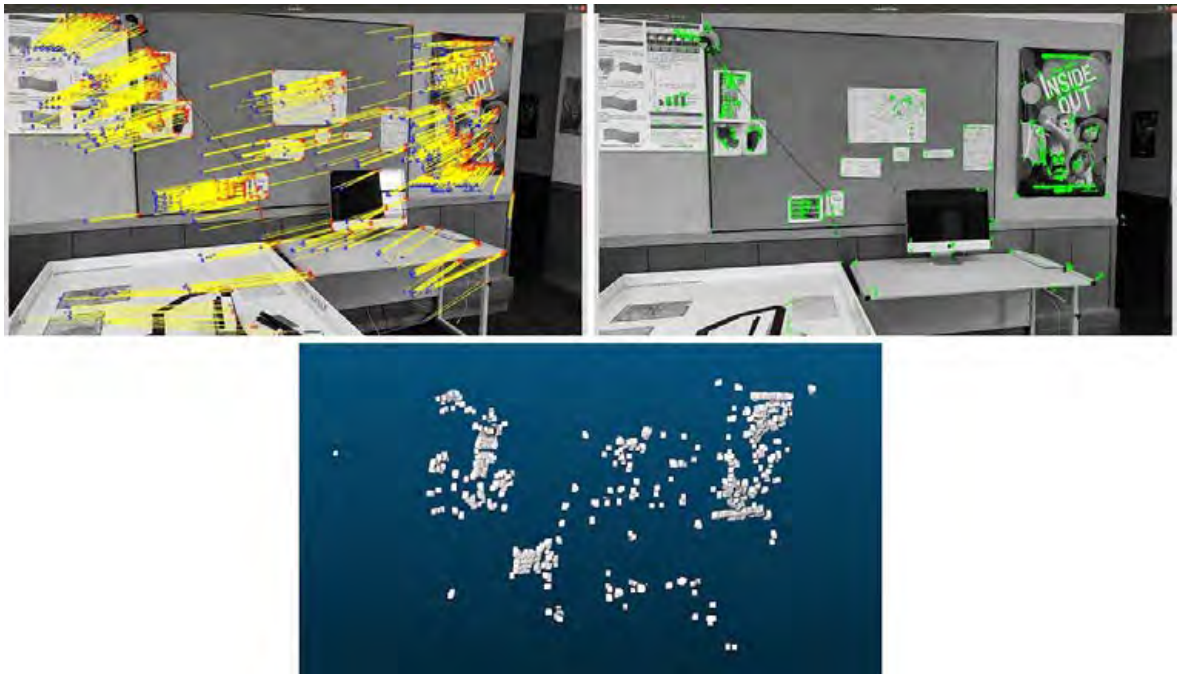

Figure 2.3 Preliminary reconstruction results from early prototyping of our tracking system outlined in Chapter 5. Top-left: points are tracked using standard feature tracking techniques to acquire two distinct views of matching points. Bottom-middle: matching points are reconstructed via triangulation using a relative pose estimate between the views. Top-right: triangulated points are projected back into to the image and align correctly with the dominant corners in the image.

moves or rotates a camera at arm's length from their body. It can include inward-facing cameras such as when capturing self-portraits, or outward facing cameras such as in panorama capture. As discussed later, in Section 2.5, many methods oversimplify this type of motion to being pure-rotations such as by assuming a fixed camera position for each image in a sequence (Kurz et al., 2013). In general cases, the camera position $\mathbf{c}$ can be determined by the rotation and translation where

$$\mathbf{c} = -\mathsf{R}^{\mathsf{T}}\mathbf{t}. \tag{2.4}$$

In spherical motion, the translation component $\mathbf{t}$ is assumed to be equal to a fixed vector $\mathbf{z} = [0\ 0\ 1]^{\mathsf{T}}$, allowing the camera position $\mathbf{c}$ to move freely on the sphere's surface.

In the stationary use-cases we consider in this thesis, we assume the camera is facing outward and so the pose of the camera can be represented as

$$\mathsf{P} = [\,\mathsf{R}\,|\,-\mathbf{z}\,], \tag{2.5}$$

where R represents the unconstrained rotation matrix of the camera (Ventura, 2016). We do not consider the calibration matrix K here as done in Equation 2.2, as the essential matrix is used to relate cameras by point correspondences in the form of pairs of normalised co-ordinates (Hartley and Zisserman, 2003).

Fixing the translation does not result in a fixed camera *position* (as in rotation-only tracking), but it forces the camera to remain at a fixed distance from the origin. This results in the camera positions being constrained to the surface of an imaginary sphere. It also forms a constraint where the viewing direction of the camera is always parallel with the normal of the sphere at every position.

As was detailed by Ventura (2016), fixing the translation resulted in a simplified method for representing the relative pose between two cameras. While absolute translation is fixed with the spherical constraint (i.e. with the origin at the centre of the sphere), the relative translation between the co-ordinate systems centred on the cameras themselves is not fixed. For two outward-facing cameras $\mathsf{P}_1 = [\mathsf{R}_1 \mid -\mathbf{z}]$ and $\mathsf{P}_2 = [\mathsf{R}_2 \mid -\mathbf{z}]$, we want to find the relative pose $[\mathsf{R}_{1\to2} \mid \mathbf{t}_{1\to2}]$ between them which can be defined as

$$\mathsf{R}_{1\to2} = \mathsf{R}_2\mathsf{R}_1^\top, \tag{2.6}$$

for the relative rotation, and

$$\mathbf{t}_{1\to2} = \mathbf{r}_3 - \mathbf{z}, \tag{2.7}$$

for the relative translation, where $\mathbf{r}_3$ is the third column of the rotation matrix R.

In spherical SfM, the primary information we supply is normalised point correspondences. However, to triangulate points, we need to know the camera poses. In general, the essential matrix $\mathsf{E}_{1\to2}$ can be decomposed to give us the poses we need (Ventura, 2016), and relates these correspondences by the equation

$$\mathbf{x}_2^\top \mathsf{E}_{1\to2}\mathbf{x}_1 = 0, \tag{2.8}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ represent the normalised point correspondences. Given the relative pose $[\mathsf{R}_{1\to2}|\mathbf{t}_{1\to2}]$, the essential matrix can also be defined by

$$\mathsf{E}_{1\to2} = [\mathbf{t}_{1\to2}]_\times \mathsf{R}_{1\to2}, \tag{2.9}$$

where $[\mathbf{a}]_\times$ is the skew-symmetric matrix such that $[\mathbf{a}]_\times\mathbf{b} = \mathbf{a} \times \mathbf{b}\ \forall\ \mathbf{b}$.

And for spherical SfM, we can substitute the Equations 2.6 and 2.7 for spherical relative pose into this definition, giving us

$$\mathsf{E}_{1\to 2} = [\mathbf{r}_3 - \mathbf{z}]_\times \mathsf{R}_2 \mathsf{R}_1^\top. \tag{2.10}$$

The work in Ventura (2016) developed solutions for computing the essential matrix in this form for both inward- and outward-facing cameras. This method required only three correspondences (compared to five in Nistér, 2004), and resulted in two possible solutions for the relative rotation, and one for the translation. However, the correct solution could be robustly chosen as only one of the relative rotations would be spherical.

2.4 Localisation

In the previous section, we covered some fundamental background on camera models, pose estimation, and reconstruction. These concepts are fundamental to localisation, as the localisation problem can be viewed as a specific type of pose estimation, where the correspondences between image and model are unknown. In this section, we cover the related work on localisation, including feature-based methods that use structure-from-motion (SfM) models, as well as alternative methods that localise from pure geometry.

The purpose of localisation is to compute the pose of a camera with respect to a model of its environment. It differs from pose estimation essentially in that the correspondences between the model, and the image are unknown. Pose estimation methods, as outlined above, typically require knowing corresponding points between the image and the model in advance, while in the case of localisation, the model and the image are the only input. These can be 2D-to-3D (Sattler et al., 2012b), or a hybrid of 2D-to-3D and 2D-to-2D (Camposeco et al., 2018).

The term *relocalisation* is also important, and it refers to localisation after tracking failure or a previously successful localisation. In this case, it is also common to assume knowledge of where the user was when they were last successfully localised. This type of localisation is relevant to many tracking systems which are outlined in the following section, but for simplicity it is safe to assume that the techniques involved are similar.

In the context of AR, localisation can provide a base positioning from which a real-time tracking system can take over. In some systems, the localisation process is performed

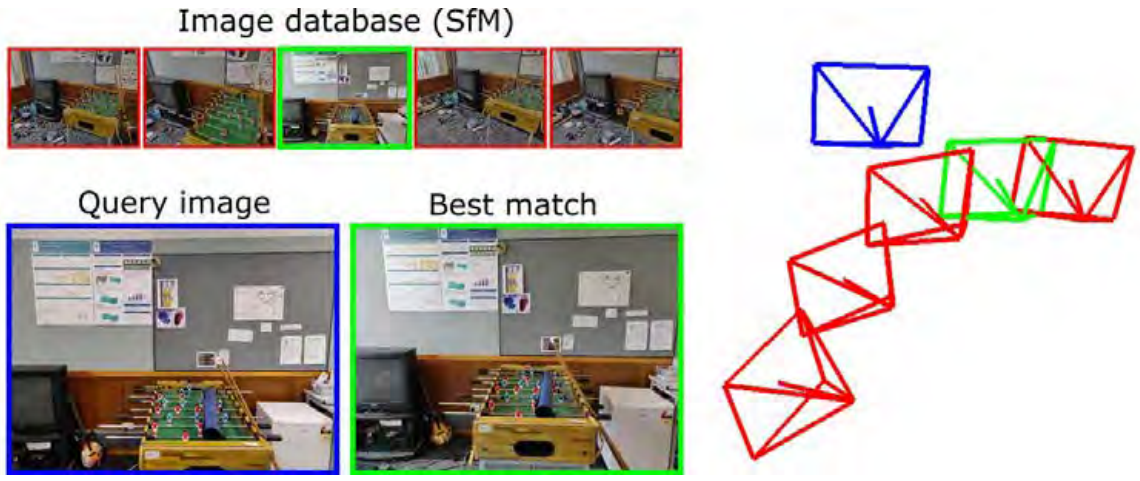

Figure 2.4 Indoor example of Bag-of-Words localisation. BoW is used to find the closest matching image (green) in a database to a given query image (blue). Feature matching can then be applied to find 2D-to-3D correspondences, and compute a pose using PnP methods.

on a separate server (Ventura et al., 2014), though this is not always required.

2.4.1 Localisation with Location Recognition

A common approach to localisation is location recognition, or place recognition. The idea is given a query image, and a collection of localised images in a database, find the closest matching image. In some applications, simply finding a match to a labelled image is enough to localise (Khan and McCane, 2012). However, in most AR applications, more precision through 6-DoF pose estimation is required (Reitmayr and Drummond, 2006; Arth et al., 2009; Liu et al., 2016).

Since the scale-invariant feature transform (SIFT) was introduced in Lowe (1999), there has been much research in applying SIFT to camera localisation. As SIFT descriptors and keypoints can compactly represent an image, many approaches have addressed camera localisation by first using image retrieval techniques such as local feature histograms (Siggelkow, 2002), or inverted indices with a SIFT vocabulary (Wang et al., 2005). Some approaches have combined these image retrieval techniques with Monte Carlo localisation (Wolf et al., 2005), or appearance-based mapping (Cummins and Newman, 2008) to compute 2-D trajectories on a large scale such as a moving car.

Bag-of-words (BoW) methods such as DBoW2 (Gálvez-López and Tardós, 2012) can be used in conjunction with SLAM methods to support re-localisation when the SLAM algorithm exhibits tracking failure. ORB-SLAM (Mur-Artal et al., 2015) employs this

method to regain tracking by applying BoW matching to the keyframes in the SLAM system. The BoW matching is used to efficiently recover correspondences between the keypoints in the query frame and the map points, allowing a full 6-DoF pose estimation using PnP methods (as in Section 2.2.1). The same technique can also be applied to localise from a model produced by SfM, as shown in Figure 2.4.

2.4.2 Model-based Localisation

Recently, SfM has created many opportunities for localised XR applications, especially in large environments. Snavely et al. (2006) have shown that SfM systems can produce accurate reconstructions of landmarks from images automatically pulled from web searches. While these systems require lengthy processing times to build the reconstructions, there are many scenarios where it may be desirable to invest this time into creating an accurate reconstruction, with the intention of re-using the model for fast localisation later on.

Many SfM systems take an incremental approach to reconstruction, adding images one at a time (Schönberger and Frahm, 2016). In this case, each time images are added to the reconstruction, their camera poses are computed, which can itself be considered localisation. However, localisation in its simplest form typically refers to registration of unseen images without the necessity of expanding and re-optimising the reconstruction. Snavely et al. (2006) supports localising user-added images by first requiring the user to indicate an approximate location, resulting in a simpler computation.

Since the release of open-source and free SfM systems such as Bundler (Snavely et al., 2006), VisualSFM (Wu et al., 2011; Wu, 2013), and COLMAP (Schönberger and Frahm, 2016; Schönberger et al., 2016), SfM reconstruction has become a popular research topic. This has resulted in a surge of interest in localisation with such models. Work by Li et al. (2010) focuses on localising separate test sets of images to SfM models of large areas. In contrast to the location recognition approaches outlined in Section 2.4.1, it aims to find matches between points of the 3-D model and features in the images. These correspondences are then applied to pose estimation using a direct linear transform.

As outlined in Section 2.2, knowing correspondences between image and model points is required for absolute pose estimation and as such, having a reliable feature matching strategy is crucial. Li et al. propose to search for matches of 3-D points to 2-D features (P2F). They prioritise 3-D points based on how many views they are visible in, given

the reconstruction. When a match is found for a point $\mathbf{X}$, the priorities of 3-D points in views that also observe $\mathbf{X}$ are increased.

Sattler et al. (2011)'s method investigates the more intuitive feature-to-point direction (F2P) by first assigning 3-D points to words in a vocabulary tree (similar to BoW). Then, given a query image, features in that image undergo the same transformation. This results in a short list of possible 3-D matches for each feature, which are searched using the standard SIFT ratio test (Lowe, 2004).

Later, Sattler et al. (2012a) improved upon this by presenting an active search method that leverages the advantages of matching in both directions. In this approach, once a match has been found with F2P matching, the nearest neighbours of the 3-D point are used in a P2F matching scheme resulting in better performance than pure F2P. In more recent work, this type of prioritised matching was again extended and incrementally improved in Sattler et al. (2016). Li et al. (2012) noted that their initial P2F matching technique was not scalable with very large 3-D point clouds, and thus the authors also developed a bi-directional matching technique.

While the above works have focused on localisation in general, other methods have investigated localising within SfM models specifically for mobile devices. Arth et al. (2009) addressed this by dividing SfM models into potentially visible sets to reduce the matching overhead. However, this approach assumed knowing an approximate location in advance, such as from GPS or a prior localised state. Work by Ventura and Höllerer (2012) demonstrated a system which could map an environment using panoramic photographs captured with omnidirectional cameras. In their system, multiple overlapping views were extracted from the panoramas and processed with an offline SfM system. Their system allowed for localising and tracking mobile devices from this pre-built map and found good localisation performance on a separate server with a simple exhaustive matching method.

There has also been work by Zollmann et al. (2014), which also uses SfM models as a basis for localisation and tracking. Their approach aims to assist micro-aerial vehicle (MAV) pilots using AR visualisations, using a similar global registration method to Ventura and Höllerer (2012). To account for lighting and seasonal differences, they use appearance information from the initial frame to update the SfM reconstruction.

While localisation to a prior SfM model has been widely researched (as above), it has also been applied to CAD models. An early example of such work is by DeMenthon and Davis (1995) who propose POSIT (Pose from Orthography and Scaling with Iterations)

which can estimate the pose of a camera with respect to an object of known geometry using known correspondences. Later, this method was improved to SoftPOSIT in David et al. (2002a) by combining the pose estimation method with SoftAssign (Rangarajan et al., 1997) to automatically determine correspondences in the same iterative loop. This method differs from typical approaches in that it establishes correspondences using purely geometric comparisons (i.e. the projections of point or line geometry), rather than appearance-based descriptors like SIFT.

Campbell et al. (2017) show a method for estimating the globally optimal camera pose and correspondences of a set of 2-D points to a 3-D point cloud without appearance information. The method uses SoftPOSIT as part of its computation to speed up convergence to the global optimum. This method is point-based and has a large computation time for relatively small point clouds. This makes it unsuitable for large line-based SfM models, and interactive AR applications. However, their work supports the need for a pose computation method that does not require appearance-based correspondences, and works toward a similar goal as SoftPOSIT.

Other approaches to model-based localisation are also common, such as the Vuforia library (PTC, Inc., 2015), which includes a model-target localisation function. The current Vuforia implementation requires a 3-D model of the object or environment (a CAD model is recommended) and the user is instructed to roughly align the model with the image before tracking starts. Vuforia also supports tracking from images (i.e. image targets), and mapping of these textures into simple 3-D geometry (such as boxes or cylinders) for initialising a co-ordinate system to track from. Like Vuforia's model target, SoftPOSIT also requires an approximate pose to initialise.

2.5 Tracking

The previous section of related work covered the problem of localisation – how to determine the pose of a camera in a known environment when we have no prior knowledge of correspondences between the image and the model. In this section, we cover tracking. Tracking is the process of determining the pose of a camera as it moves. For many applications, it is also required that this camera pose is computed in real-time. This is particularly true in the case of XR, where the camera pose is used to render visualisations from the user's perspective as they move their device.

In some instances, such as sports broadcasting or virtual studios, it is possible to use dedicated hardware to track the pose of a camera, such as Viz Virtual Studio (Vizrt,

Figure 2.5 Two examples of tracked cameras used in broadcast television. Left: A half-time discussion by rugby commentators in a virtual studio, utilising a 3-D representation of the team formation. Right: a presenter with a 3-D model of a Hoiho for a Bird of the Year presentation, utilising Viz Virtual Studio (Vizrt, 2010) to portray the bird's visual features and habitat *(Sources with permission, Sky Sport NZ, and TVNZ).*

2010), StarTracker (Mo-Sys, 2017), and FRESH (ChyronHego, 2019). These systems are capable of providing sufficient pose data to accurately register visualisations to the view of the broadcast cameras as shown in Figure 2.5. However, this type of hardware requires careful setup in addition to being very expensive, so is not useful for providing XR visualisations tailored to an individual spectator's perspective. Advanced broadcast camera technology is also prevalent outside the studio, in particular with sports broadcasting (Animation Research Ltd., 2020; Thomas et al., 2017).

Beyond the context of sport broadcasting, XR tracking methods typically make use of sensors onboard each user's mobile device, such as their smartphone, tablet, or head-mounted display (Koulieris et al., 2019). These sensors can include one or multiple cameras, inertial sensors, and in some cases depth (RGB-D) sensors. Salas-Moreno et al. (2013) show how depth cameras can track from high-level objects in the scene via object recognition, however it is not always feasible to assume a depth camera would be readily available.

Tracking from a single camera (i.e. monocular) is a fundamental type of tracking, since it is often possible to combine additional inertial measurements using monocular tracking as a base. Since early work by Davison et al. (2007), there have been many examples of monocular tracking systems being used as a starting point, and later extended to support inertial sensor measurements, depth sensors, or stereo cameras (Mur-Artal and Tardós, 2017a,b).

2.5.1 Monocular Tracking

Due to its wide applicability and simple hardware requirements, monocular tracking is a key focus of this thesis. Monocular tracking can refer to any tracking system that uses a single camera as its primary sensor. SLAM has been a popular approach to tracking single cameras, particularly in the field of robotics, since work by Davison and Murray (2002), and Davison (2003) which was later extended to MonoSLAM (Davison et al., 2007). Their method uses an Extended Kalman Filter (EKF) to update camera pose and landmark locations based on measurements from the image. The authors note that the EKF update can be expensive when there are many features in the map, and address this by tracking a small number of dominant features. Other approaches to the SLAM problem have existed for longer, such as work by Dissanayake et al. (2001) which proposed a solution to the SLAM problem for GPS and radar sensors. However, the EKF approach was standard for a while, and there has been research in reducing the filter update complexity (Knight et al., 2001; Montemerlo et al., 2002). But more recently, approaches have taken to separating the mapping and tracking into separate tasks.

A famous example of this kind of monocular tracking being applied to AR is Klein and Murray's Parallel Tracking and Mapping (PTAM) system (Klein and Murray, 2007). In this work, the authors design a tracking system specifically targeting small AR workspaces. This work was one of the first to take a multi-threaded approach to the SLAM problem. Specifically, they separate the tasks of tracking and mapping into two separate threads, where tracking is prioritised for real-time performance, while mapping may update less frequently. However, creating a map usually requires knowing a camera pose (i.e. from tracking) and tracking usually requires a map. To overcome this, the authors implemented a separate initialisation stage. They require the user to start the system, move the camera, then press a key which triggers an initial map reconstruction from which the tracking and AR rendering system can take over.

More recently, ORB-SLAM has been published which offers many improvements over PTAM (Mur-Artal et al., 2015). ORB-SLAM has more capabilities than PTAM and is designed with the similar idea of using multiple threads to increase computational efficiency on multi-core devices. A recent survey of SLAM suggests that SLAM systems should cover three basic components: initialisation of a 3-D map, tracking the motion and further map updates, while an accurate and stable solution would additionally cover global map optimisation and relocalisation (Taketomi et al., 2017).

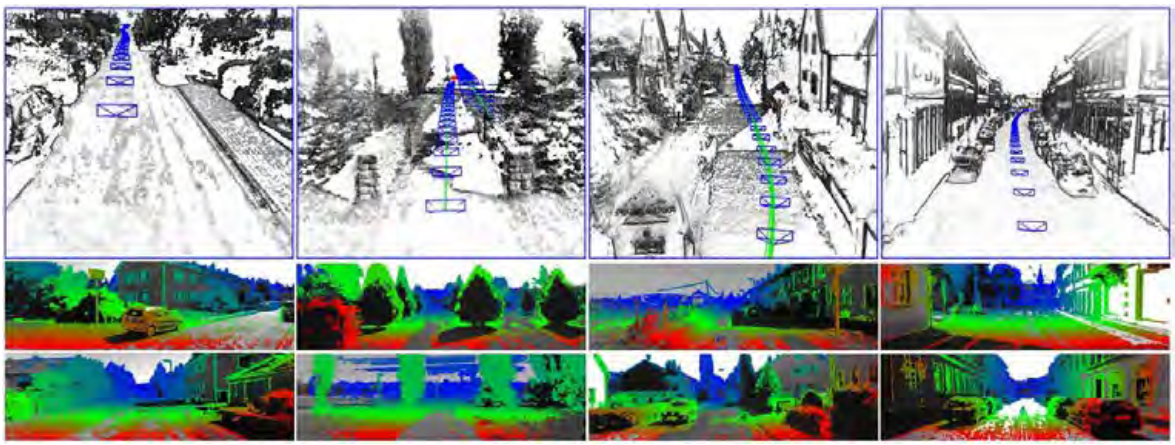

Figure 2.6 Point clouds (top row) and depth maps (bottom row) from a direct SLAM system LSD-SLAM (Engel et al., 2015). Copyright © 2015, IEEE. The resulting semi-dense reconstruction contains many more points due to the direct usage of pixel intensities, rather than features as in indirect methods such as ORB-SLAM (Mur-Artal et al., 2015).

In contrast to PTAM, ORB-SLAM also offered relocalisation, which allows the tracking system to resume tracking after tracking is lost. Methods for relocalisation share similarities with localisation approaches, with the expectation that the camera is viewing part of the scene that has already been mapped. ORB-SLAM also offers improvements in mapping through pose graph optimisation, and loop closure. These allow for reduced drift while tracking, and an even distribution of error when optimising keyframe poses when the camera trajectory loops back to a revisited position (i.e. closing a loop).

Another key aspect of ORB-SLAM is, as the name suggest, its use of ORB (Oriented Robust BRIEF) features for all key components of the system (Rublee et al., 2011). Since the images are being processed by a feature detector and only the resulting features are used (i.e. indirect measurements from the camera sensor), ORB-SLAM is known as an indirect SLAM system. Direct approaches also exist, which take measurements directly from the images (Engel et al., 2014). These systems usually end up with a more dense reconstruction of the scene (see Figure 2.6), which can be more expensive to process and optimise, with the advantage of having less overhead due to the lack of feature extraction. Both indirect and direct methods have been shown to run in real-time, so either approach is feasible for the purposes of XR.

Large-Scale Direct Monocular SLAM (LSD-SLAM, Engel et al., 2014) is one of the most promising direct approaches. This method offers a scalable direct approach by focusing the reconstruction on regions of the images with large intensity gradients. This results in what the authors call a semi-dense reconstruction that ignores untex-

tured surfaces. This is an improvement upon previous methods which created fully dense reconstructions with real-time performance, but with the added requirement of powerful Graphics Processing Units (GPU) acceleration (Newcombe et al., 2011).

However, the use of features has strong advantages over direct approaches. When features are available, it would be possible to use feature-matching approaches such as active search for relocalisation when tracking is lost (Sattler et al., 2012a). Furthermore, it would also be possible to apply Bag-of-Words (BoW, Sivic and Zisserman, 2003) approaches to loop detection and loop closure (Gálvez-López and Tardós, 2012). This BoW approach is used by ORB-SLAM and works by essentially finding the closest keyframe that approximates the relocalisation frame, and rediscovering map point correspondences through feature matching.

While there has not been much work on direct SLAM methods that support relocalisation (some works such as Engel et al. (2017) disable these features when comparing direct to feature-based SLAM), a naive approach would be to relocalise by aligning points (such as through iterative closest point, Besl and McKay, 1992) between the existing reconstruction, and the reconstruction produced by the detached tracking. This, however, could be prohibitively expensive due to the density of the reconstructions, and lack of features for finding such correspondences. More recent work in direct SLAM by Engel et al. (2017) also does not discuss relocalisation with their system.

While direct tracking systems have not focused on relocalisation, there has been research into implementing loop detection in direct SLAM systems, such as Gao et al. (2018). The loop detection problem shares similarities with the relocalisation problem in that both require detecting when the camera has returned to a previously visited place. In this work, the authors work around the limitations of direct approaches by detecting features on keyframes only, allowing for typical BoW style matching among keyframes to detect and close loops.

Some research has also looked into combining the advantages of both feature-based and direct approaches, such as Forster et al. (2014). In this work, they use the direct approach for frame-to-frame motion estimation, while detecting features in keyframes to allow for global optimisation via bundle adjustment. They later improve their method in Forster et al. (2016) by supporting more camera types, stereo rigs, and external inertial sensors. Other approaches have also been taken to combine direct and feature-based SLAM, such as Lee and Civera (2018), who propose maintaining two separate maps, one sparse map from features for global consistency, and a semi-dense map from direct odometry for robust tracking.

2.5.2 Rotation-only Tracking

The works discussed in the previous section all share a common theme, being that they aim to track a camera assuming it has 6 degrees-of-freedom. That is, the camera pose can be described by a rotation R and translation $\mathbf{t}$, as in Equation 2.2. However, some tracking methods reduce computation by estimating pose with fewer parameters, such as via homography estimation (Prince et al., 2002). Using a homography to estimate the camera pose is accurate provided that the scene to be tracked is planar, or the motion being modelled is a rotation.

There are two driving factors behind the use of homographies. In some cases, it is fair to assume that the scene may be planar. For example, tracking a camera for field sports can utilise the homography assumption, provided they are able to segment the planar sports field from the image (Hadian and Kasaei, 2015), or tracking from obviously planar objects such as books (Billinghurst et al., 2001), fiducial markers (Wagner et al., 2008), or board games (Huynh et al., 2009). While not all of these works explicitly apply homography-based tracking, they clearly demonstrate a use-case for tracking from planar objects.

The other application of homographies occurs when the camera motion is purely rotational. This use-case is a little harder to justify with perfect examples. This is because pure rotational movement is very uncommon in the natural world. Especially in the case of XR, where almost all motion that we are interested in tracking is generated by people. However, there are many situations where user motion is *almost* purely rotational. The classic example of this is in panorama generation, in which early work by Brown et al. (2007) successfully presented minimal homography solvers and demonstrated their applicability to panorama construction.

Other approaches have also successfully been applied to panorama generation, many of which assume rotation-only motion, but not necessarily using homographies. DiVerdi et al. (2008) introduced a method for rotation tracking for the purposes of panorama generation. Their method applied optical flow to track corners frame-to-frame, project each 2-D point into a 3-D point at a fixed depth, then align point sets using Horn's method (Horn, 1987) to compute an orientation for each frame. Wagner et al. (2010) introduced a method for rotation tracking which was optimised for mobile device usage. Their approach tracks the rotations using non-linear refinement to minimise reprojection error between 3-D cylindrical projections of points, and their 2-D correspondences.

While these works were successful in applying rotation-only tracking to panorama

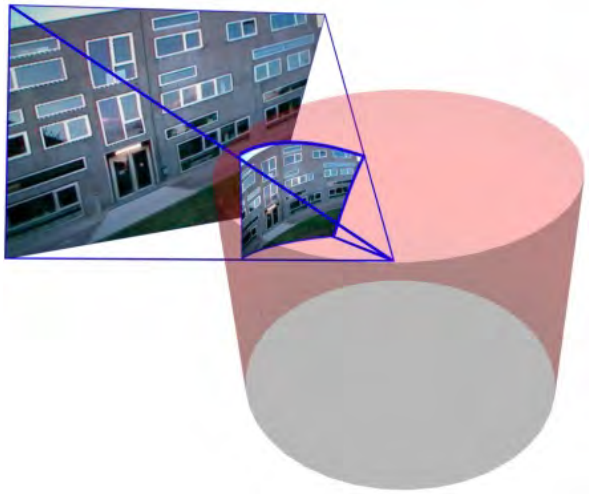

Figure 2.7 Many rotation-only tracking systems utilise panoramas to represent a map of their environment. In this example, the camera's view of the environment is mapped to a cylinder, with the assumption that the camera purely rotates at the centroid of the cylinder (Wagner et al., 2010). Copyright © 2010, IEEE.

generation, work by Langlotz et al. (2010) demonstrated that this type of tracking is also suitable for AR and shared annotations. Later, Langlotz et al. (2011) extended this system with improved panorama re-detection, and improved tracking via integration of measurements from an inertial sensor. These types of mobile annotated AR experiences are referred to as AR browsers (Grubert et al., 2011), and individual usage patterns have been studied. In this user survey, it was reported that most users of AR browsers interacted with the system while standing in one position and rotating.

2.5.3 Sensor Fusion and Hybrid Tracking

Sensor fusion refers to tracking systems that leverage more than one type of sensor, commonly cameras and inertial sensors or gyros. These types of hybrid systems have existed for a long time (Azuma and Bishop, 1994) and have been shown to improve the reliability of a tracking system. The advantage of using inertial sensors is that they provide additional tracking information with no extra computation. While inertial devices used on their own are usually insufficient for XR tracking due to drift (Siciliano and Khatib, 2016), the extra information they provide can be applied to improve the results of visual tracking for both general and rotation-only tracking systems (Langlotz et al., 2011; Mur-Artal and Tardós, 2017b; Qin et al., 2018).

In an effort to create tracking systems which can be used in all tracking scenarios, such

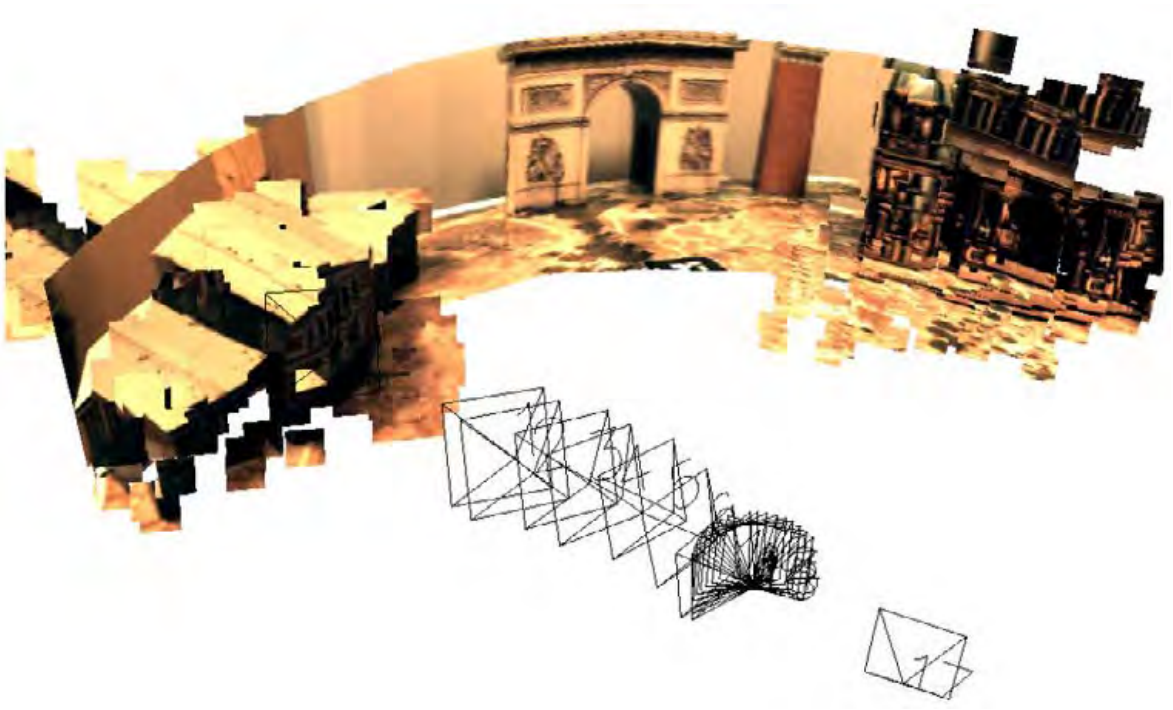

Figure 2.8 Hybrid tracking systems aim to predict the type of motion the user is making and switch their motion model accordingly. In this figure, a panorama of the environment is mapped based on the rotational movement, and additional 3-D points are also generated from the translation (Gauglitz et al., 2012). Copyright © 2012, IEEE.

as small indoor spaces with large translations, to large open outdoor areas, switching hybrid systems have been introduced. Work by Gauglitz et al. (2012) has introduced a system that can switch between rotation-only and general motion tracking based on what the input to the system is. Their method applies the Geometric Robust Information Criterion (GRIC) score introduced by Torr (2002) to determine which tracking model to use. Pirchheim et al. (2013) also took a similar approach to switching between panorama-based tracking, referring to their system as a hybrid tracking system.

2.6 Localisation and Tracking with Deep Learning

Following the emergence of convolutional neural networks (CNNs), there has been a range of research into applying deep learning for computer vision tasks. Deep learning has been applied extensively through various application scenarios ranging from medical (Milletari et al., 2016), to animation (Kim et al., 2018). Alongside this, there has been much recent work that focuses on localising images with deep learning. The

Figure 2.9 Results reported by PoseNet, a deep-learning localisation method (Kendall et al., 2015). Their method shows promising results for localisation in challenging outdoor scenarios and dynamic environments but can suffer from accuracy error as depicted by the misalignment of the model (red) when overlaid with the image. Copyright © 2015, IEEE.

deep learning approaches often aim to either attempt full pose regression with a single CNN (Kendall et al., 2015), or split the process into multiple stages (Brachmann et al., 2017) and applying end-to-end learning on the whole system.

The PoseNet approach introduced by Kendall et al. (2015) uses a single CNN based on the GoogLeNet architecture (Szegedy et al., 2015). The input to the network is a single image, and the network is trained using ground-truth poses from a typical SfM system (Wu, 2013). The resulting poses from the system showed promising results and worked with small numbers of training images. However, as depicted in Figure 2.9 the results often showed issues with accuracy.

Other work by Brachmann et al. (2016) took a different approach that applies concepts of scene co-ordinate regression to learning image localisation. Scene co-ordinate regression is the process of computing the corresponding 3-D co-ordinate for each pixel in an image or RGB-D data (Shotton et al., 2013). The result is like a depth map but in the scene co-ordinate system rather than camera's local co-ordinate system, which

allows for simplified pose estimation.

Later, Brachmann et al. (2017) extended this work by incorporating a differentiable RANSAC, then later showed that scene co-ordinates can be learned by the pipeline as a whole (essentially requiring no prior model) (Brachmann and Rother, 2018), and finally extended to a more scalable solution that incorporates first a gating network that essentially acts as a scene classifier, followed by expert networks that are trained to localise within smaller regions of a larger disconnected model (Brachmann and Rother, 2019). Brachmann and Rother call their approach expert sample consensus (ESAC).

Tateno et al. (2017) propose using a CNN to assist with SLAM. The CNN component of their system is trained to estimate depth values for each pixel of their monocular input images (RGB-D). Their method showed promising results with pure rotational movement, as using the learned depth estimation allows for mapping without translational movement. Though the CNN component requires GPU acceleration to run in real-time, so may not be suitable for current mobile device hardware.

While these deep-learning approaches are promising, localisation approaches such as ESAC report lower accuracy in large environments when compared to state-of-the-art feature-matching approaches (Brachmann and Rother, 2019). To achieve good performance, all these methods rely on GPU acceleration, which is not typically available with consumer-grade mobile phones (Kendall et al., 2015; Brachmann and Rother, 2019). Another limitation of the CNN-based localisation approaches is that they do not consider how to report failed or erroneous localisation results. Typically, feature-based methods will apply a threshold on the number of matches to determine when localisation is unstable (Li et al., 2010; Sattler et al., 2012a).

2.7 Panorama Generation

In the previous sections, we covered localisation and tracking, including tracking methods for mostly stationary users (i.e. rotation-only and hybrid tracking), particularly in the context of AR. In this section, we investigate the application of these tracking methods for another stationary use-case: panorama generation. Panorama generation and panorama capture is becoming common, with most smartphones providing panorama stitching functionality within the stock camera applications. High quality and immersive panoramic experiences have become more desirable for use-cases such as immersive communication (Young et al., 2019), and education (Kavanagh et al., 2016) among many others. With the increased demand for VR content, a lot of re-

cent research has focused on capturing environments in ways which support immersive viewing experiences (Richardt et al., 2020). As VR hardware becomes more accessible, users are able to use these methods to relive experiences, or revisit places by viewing panoramic content in a virtual environment.

The reconstruction of stereo panoramas is a well-researched field. Early works used input from cameras that were moved on perfectly circular trajectories (Shum and Szeliski, 1999; Peleg et al., 2001; Li et al., 2004) in order to capture the necessary views to generate separate left- and right-eye panoramas. The idea is that when using the centre strip of each image a normal panoramic image is generated. When using strips from the left side, a right-eye panorama is generated and vice-versa for the left-eye panorama. These approaches come with some disadvantages when the input camera sequence is not perfectly acquired. When frames of video do not lie perfectly on the circle, they are warped to approximate these positions resulting in visual artefacts such as aliasing or stretching. This often occurs in the case of casual panorama capture, such as a typical user moving a mobile device in a circle at arm's length.

The Megastereo (Richardt et al., 2013) and MegaParallax (Bertel and Richardt, 2018; Bertel et al., 2019) systems compensate for these artefacts by adopting a view synthesis approach. Both methods use SfM in combination with flow-based ray blending in order to compute more accurate synthetic views. In contrast to SfM and view synthesis, Zhang and Liu (2015) use a purely stitching-based method in order to stitch the left and right panoramas from casual input video, but their method requires calibrated stereo cameras which are not common on typical mobile devices.

Stereo panoramas are limited to providing horizontal parallax from a fixed viewing position. Other techniques such as plenoptic modelling (McMillan and Bishop, 1995), view-dependent image-based rendering (Debevec et al., 1998), unstructured lumigraph or light field rendering (Buehler et al., 2001; Davis et al., 2012), and more recent methods (Luo et al., 2018; Hedman et al., 2017; Huang et al., 2017) go beyond stereo panoramas using view synthesis techniques for free viewpoint experiences. In recent years, methods based on deep neural networks (Flynn et al., 2016; Zhou et al., 2016; Kalantari et al., 2016; Zhou et al., 2018; Flynn et al., 2019) have also shown impressive results for view synthesis. However, these approaches require a high density of input views, which generally make them unsuitable for casual capture of a 360° scene.

An alternative approach is to use a stereo or multi-camera rig to facilitate capturing the large set of views needed to stitch a stereo panorama. Facebook and Google provide camera setups that are specifically designed for 360° capture and consist of a ring of

outward facing cameras, such as the Google Jump (Anderson et al., 2016) or Facebook Surround360 (Facebook, Inc., 2016). These camera rigs provide a sparse set of views with known extrinsic and intrinsic calibration combined with view interpolation to produce smooth stereo panoramic output. Similarly, Schroers et al. (2018) presented an approach for computing omnistereo videos from a sparse set of 16 cameras that also allow for virtual head motion. Recently, Broxton et al. (2020) suggest leveraging deep learning based view interpolation with a 46 camera rig for light field rendering. While these rigs are ideal for professional panoramic video capture, they are less than ideal for "casual" capture given the cost, size, and weight of the camera rig. Hedman and Kopf (2018) enabled fast reconstruction of a scene for free-viewpoint rendering from casually captured smartphone video, but require a stereo camera and integrated inertial measurement unit (IMU) for initial depth and motion estimates.

For casual capture of stereo panoramas with a handheld perspective camera, Bertel and Richardt (2018) noted that one of the main challenges is that "structure-from-motion is also known to not be robust for our desired narrow-baseline inside-out capturing scenario". There has been research into applying spherical motion constraints to SfM systems to support this kind of movement. Ventura (2016) provide methods for computing an essential matrix for relative pose estimation under a spherical constraint, and integrate it into an SfM system. Recent work by Sweeney et al. (2019) also investigate methods for computing a fundamental matrix with this constraint for panoramic videos. The authors also found that state-of-the-art SLAM and SfM pipelines were unsuitable for reconstruction in this scenario. These works highlight the need for robust reconstruction for the stationary use-cases which we address in this thesis.

2.8 Summary and Trends

To summarise, in this chapter, we have covered the related work on XR, and what is possible with the latest hardware such as Quest, Magic Leap, and mobile frameworks ARKit and ARCore. We provide the background required to understand the computer vision concepts behind the chapters that follow. We have also summarised the literature in localisation, tracking, and panorama capture. In this section, we will conclude this chapter with final remarks on the trends and gaps in the previous research.

Localisation On the topic of localisation, we have found that approaches differ depending on prior assumptions about the environment. There has been much research on

localisation via location recognition, such as BoW approaches. Methods which utilise SfM models tend to be developing advanced feature matching methods to increase the robustness of Perspective-n-Point pose estimation. Other research has focused on localising with respect to predictable structures in the environment such as building facades. While the feature-based approaches were often shown to work with dynamics such as different lighting and weather, the problem of dynamic environments has been largely absorbed by the robustness of the features themselves and not addressed directly. For this reason, in Chapter 4, we look into state-of-the-art line-based localisation approaches which can support environments such as sport stadia, which are very dynamic in their appearance while maintaining consistent linear geometry. Also, in Chapter 4, we investigate state-of-the-art image-based localisation approaches in the context of localising stationary users at sports events.

Tracking With regard to tracking, the related work has mostly taken one of two approaches: either fully unconstrained tracking (6-DoF), or rotation-only tracking (3-DoF). Many works have shown that either of those tracking approaches can combine inertial sensors to improve robustness, but only a few works have attempted to bridge the gap between the two approaches. For this reason, we focus our attention on tracking for stationary users, and try to explore novel tracking methods for users who are *mostly* rotating. We address this gap with a novel tracking system based on spherical motion in Chapter 5.

Panorama Generation Recent related work in panorama generation are tending toward a more immersive experience such as the stereo panorama, or free viewpoint video. In many cases, it is assumed that a calibrated multi-camera rig is available, but there is also interest in the *casual* capture of panoramas with simple smartphone hardware. The current state-of-the-art approaches tend to use general SfM methods to compute camera poses, which often struggle when users are standing in one place. For that purpose, we aim to investigate the applicability of the spherical SfM method introduced in Ventura (2016) to produce stereo panoramas in Chapter 6.

Chapter 3

Data Capture

In the previous chapter, we covered the related work in localisation, tracking, and panorama generation. We also provided some background to the fundamental computer vision problems involved in addressing these topics. We now focus on the problem at hand: how to provide AR content to users in a large dynamic environment such as a sport stadium. A key component to this goal is sourcing suitable data for our experimentation.

As we saw in the related work, implementing such a system would require localisation with respect to a prior model of the environment. Also, to provide live updates to the AR visualisation, real-time tracking is also required. In this chapter, we investigate what is available in terms of localisation and tracking datasets and determine their suitability for our use case of localising and tracking stationary AR users.

3.1 Existing Datasets

In this section, we outline some of the most widely used datasets and discuss their suitability for our purpose. We group the datasets into two natural categories: photo and video. As the localisation problem is concerned with pose estimation of a single image, most localisation datasets are represented as collections of photographs, while tracking datasets usually consist of video sequences, sometimes with depth data as captured through a depth sensor such as a Kinect (Microsoft Corporation, 2010).

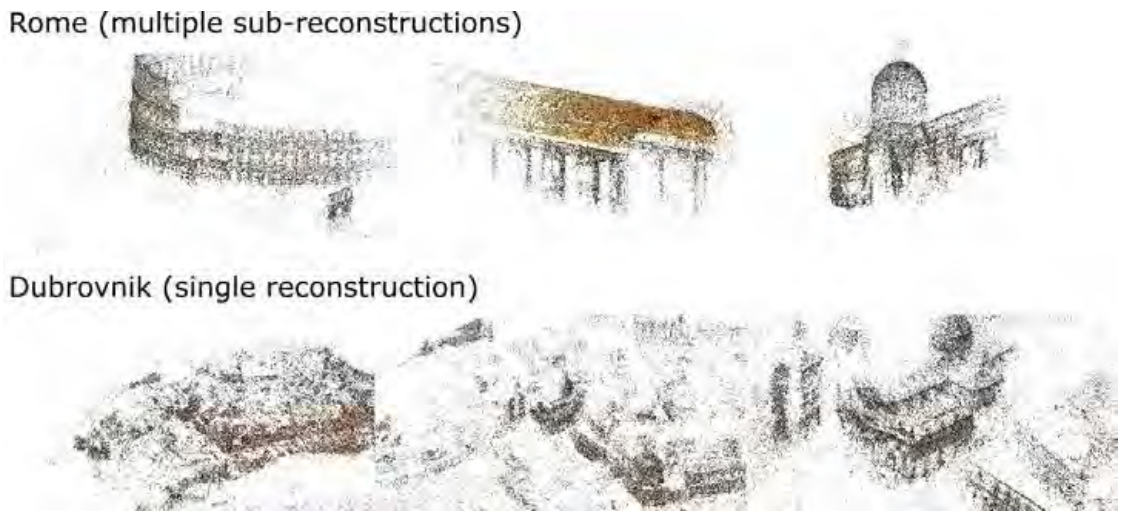

Figure 3.1 Sample point renderings of the public large-scale SfM datasets. Top row: Rome, containing many small reconstructions of various sites of interest, and bottom row: Dubrovnik, containing a single large reconstruction (Li et al., 2010).

3.1.1 Localisation Datasets

Localisation datasets usually consist of images from arbitrary viewpoints in an environment, as well as reconstructions and pose data for each image. In the context of AR, the reconstruction can be used as a guideline for placement of virtual content, and the provided images and pose data can be used as a basis for evaluation of localisation approaches. Here we discuss the suitability of these datasets for the stationary use-cases that we investigate in this thesis.

There has been a lot of research on large-scale localisation, particularly at city-scale. Datasets such as Dubrovnik and Rome, released by Li et al. (2010), contain thousands of images from a range of views in historic city centres. The datasets consist mainly of feature descriptors for each image, and their correspondences to points in 3-D reconstructions as shown in Figure 3.1. They do not openly provide the original images, as they were taken from the image sharing site Flickr [1]. Similar datasets have been published in Sattler et al. (2012b), who provide a full dataset including images, and reconstruction. This dataset was later extended to include both day and night conditions in Sattler et al. (2018) (Aachen Day/Night).

Many public datasets have considered localisation in dynamic environments, specifically regarding natural lighting conditions in outdoor scenarios. However, public localisation datasets for sport events, such as a stadium environment, do not currently exist. As one

[1] https://www.flickr.com/

Figure 3.2 Two similar views of a sport stadium, with dynamic appearance. The presence of spectators, field lights, line markings, and lighting conditions combined result in a drastic difference in appearance when compared to a visit to an empty stadium.

of our motivating applications is to localise and track stationary users, such as sport spectators, we require a dataset that reflects this use-case. The types of dynamics in a sport stadium are not widely represented in outdoor city-scale localisation datasets.

As shown in Figure 3.2, a similar view of a sport stadium can have drastic differences in appearance. The presence of distant spectators creates a unique textural appearance on the (approximately) flat surface of the stands. As this type of dynamic cannot be predicted, localising in this environment is difficult. Challenging dynamics such as inconsistent lighting, advertisements, and field markings are not widely covered in the currently available localisation datasets.

3.1.2 Tracking Datasets

Tracking datasets typically differ from localisation datasets, as tracking is interested in determining camera pose with small frame-to-frame movements. For this reason, most tracking datasets consist of video data, compared to general images found in localisation datasets. Ideally, these datasets would contain ground-truth or reference poses for each frame of video. We also require that the camera motion approximates that of the stationary use-cases which we address in this thesis. Some of the widely used tracking datasets are discussed here.

In the New College dataset (Smith et al., 2009), the authors capture data from a moving robot with various sensors. The authors captured from a stereoscopic camera, a panoramic camera, wheel odometry readings, laser range scanners, and global position-

ing system (GPS) sensors all attached to a single robot. While the movement pattern they captured is outdoor, it contains significant translation movement. An important aspect of this dataset is the necessity of synchronisation of the data from each sensor. Similar to this, the KITTI dataset introduced by Geiger et al. (2013) contains data captured from multiple sensors attached to a moving car.

The outdoor dataset introduced by Kurz et al. (2013) directly targets monocular AR use-cases. This dataset provided ground-truth tracking data for video sequences captured by outdoor stationary users, which is similar to our use-case except that they assume the camera is rotating at a fixed location (i.e. pure rotation). This makes the dataset suitable for evaluating rotation-only trackers for large-scale AR, but lacks the precision we are specifically aiming to address. Additionally, we learned through contacting the authors that the dataset is not currently available as the company that initially authored it, Metaio, was acquired by Apple in 2015 [2].

The 7-Scenes dataset is introduced by Shotton et al. (2013), and Glocker et al. (2013). In these instances, the dataset was used to analyse depth sensor relocalisation techniques, and for the application of dense mapping. As their application scenario was based on attempting to map an environment, the data they captured contains a large amount of translational movement. As the Kinect depth sensor has limited range, they also focused their capture on close-range indoor scenes, making it unsuitable for evaluating stationary tracking methods.

Another commonly used dataset for evaluating SLAM systems is the TUM dataset introduced by Sturm et al. (2012a,b). This dataset contains a wide variety of movement patterns captured from RGB-D cameras, with several sequences that somewhat replicate the type of movement pattern we are targeting. The authors provide some sequences which seem promising. Specifically, sequences labelled *360_hemisphere* where they aim to capture a hemispherical view of a large room. However, the authors move significantly by walking in a circle. Also, the sequences labelled *rpy* (roll-pitch-yaw), contain purely rotational movement. However, in these sequences, the authors make very unnatural movements with the device in an effort to remove translational motion altogether.

While small sections of these datasets may be suitable, they further highlight that most research into tracking with SLAM techniques typically relies either on significant translational motion outdoors (such as from moving cars), demonstrate unnatural pure

[2] Ron Miller, Josh Constine: *https://techcrunch.com/2015/05/28/apple-metaio/*. May 2015.

rotations, or target 6-DoF tracking with depth sensors indoors. As our use case targets large environments with small translations, we found none of these standard datasets to be suitable for our experimentation.

3.2 Stadium Localisation

To prepare for experimentation with state-of-the-art localisation techniques in a stadium environment, we captured data in various formats to process with offline reconstruction approaches. We captured a combination of photo datasets to use with a structure-from-motion (SfM) system such as COLMAP (Schönberger and Frahm, 2016), synchronised stereo video with a custom-built wide-baseline stereo rig for mapping with ORB-SLAM2 (Mur-Artal and Tardós, 2017a), and videos from a Ricoh Theta S dual-fisheye 360° camera which could be used with panoramic localisation and tracking systems such as that suggested by Ventura and Höllerer (2012). We also add a mobile device camera to the capture rig to use the mobile camera footage for testing with similar conditions to real users.

While the usage of some of this data is beyond the scope of this thesis, we simply capture as much data as possible to support future work in this area. This is particularly necessary as there are currently no public datasets for localisation in large sport stadia.

3.2.1 Photographs for SfM

Firstly, we captured photographs with the intention of later processing with offline SfM. These SfM systems are based on triangulation which, as discussed in the previous chapter, requires significant translational movement to achieve reliable results. An example of the capture pattern that we used is shown in Figure 3.3.

In order to capture a complete representation of the stadium, we captured photos in clusters where we aim to capture several overlapping views with a range of orientations at each position. We then move to a different position and repeat the process, while aiming to capture from an even distribution of locations.

Using this technique, we captured photos in two sport stadia. Most of the data was captured in the Forsyth Barr Stadium (FBS) in Dunedin, New Zealand, which has four separate stands around a rectangular pitch (1125 images). We also captured a much smaller dataset in the Melbourne Cricket Ground (MCG) in Australia, which is essentially one large stand surrounding an oval pitch (49 images).

Figure 3.3 Left: example of our complete reconstruction on the FBS dataset. Right: close-up view showing the clustered capture pattern.

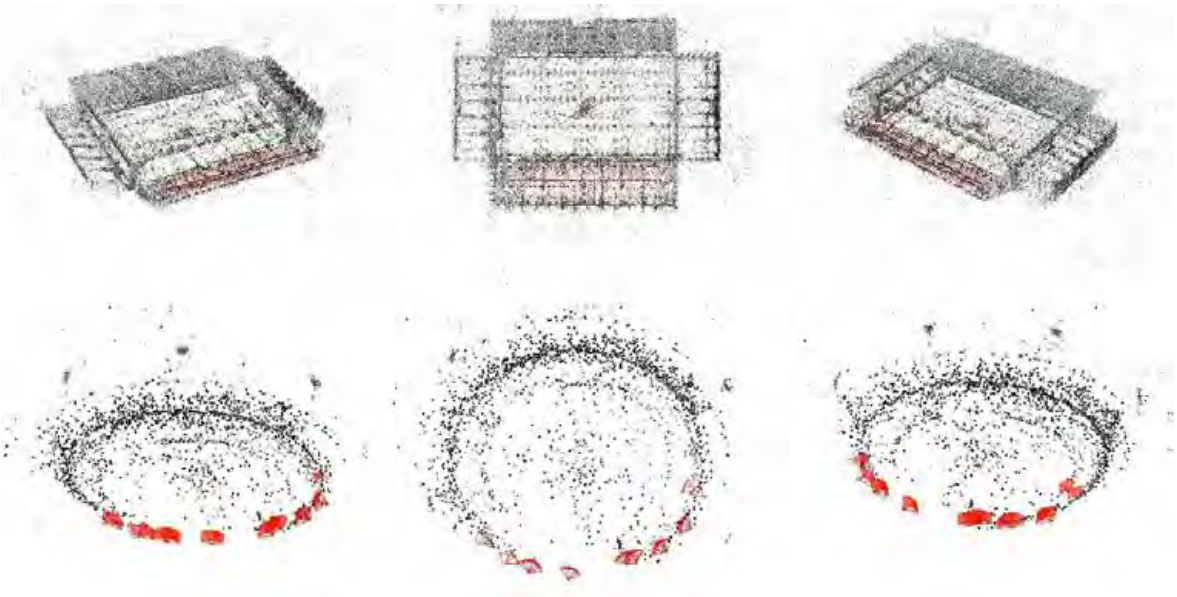

Figure 3.4 Output of our two stadium reconstructions. Top row: FBS dataset with 120454 points. Bottom row: MCG dataset with 2719 points.

With the rectangular stadium captured in FBS dataset, we found good results capturing most of the photographs from the two largest opposing stands, as well as ground-level photos of the two smaller adjacent stands. In our second dataset, we followed a similar capture pattern but from a smaller region.

In both cases, we were able to create a single reconstruction of the stadium using COLMAP. In the case of the MCG data, as expected, the reconstruction was far sparser (2719 points) compared to the FBS dataset (120454 points) due to having far fewer views. However, we use this data to demonstrate the feasibility of reconstruction when data availability is limited. The output of these reconstructions is shown in Figure 3.4.

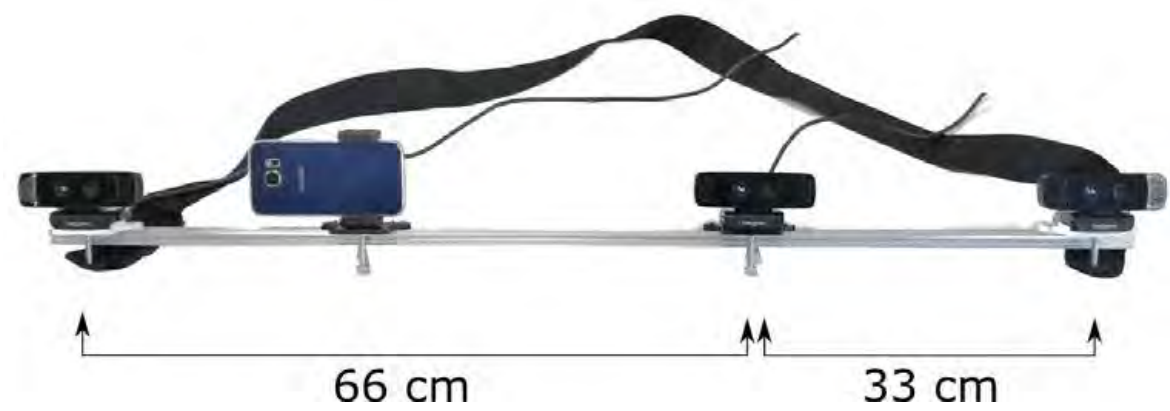

Figure 3.5 Diagram of the stereo rig with four cameras. Three cameras are calibrated before and after the capture session, with three main baseline configurations (33 cm, 66 cm, and 1 m). The mobile device camera is also calibrated to allow for comparisons with the other cameras, which may be used as a baseline or ground-truth.

Figure 3.6 The complete stereo-rig being used to capture in the stadium. Left: close-up of the view from the stereo-rig. Right: how the rig was held, including the backpack setup.

3.2.2 Capture with a Stereo Rig

As there has been a wide range of research on reconstruction through calibrated stereo rigs, we capture this type of data in the FBS dataset. For our capture, we custom-built a portable handheld stereo rig with multiple cameras. A prototype of the rig is shown in Figure 3.5, and an example of the final rig (with the Theta S panoramic camera added), and its usage during our capture at FBS is shown in Figure 3.6.

In total, our rig contained 3 webcam-style cameras (Creative Senz3D), one smartphone camera (Samsung Galaxy S6), and a panoramic camera attached at a higher position in the centre (Ricoh Theta S). The Senz3D cameras are RGB-D cameras, but we only capture the RGB data as the depth range is limited. While the S6 and the Theta have

their own on-board storage, the three Senz3D cameras require a data connection to a laptop PC for recording.

Synchronised Cameras The Senz3D cameras were connected via USB cables to a laptop carried in a backpack (Figure 3.6, right), and the entire rig was carried using a shoulder-strap. The laptop's record functionality was controlled remotely by a tablet to avoid the need for unpacking to start and stop the Senz3D recording procedure, while the S6 and Theta were manually controlled.

The Senz3D cameras were capturing at 30 Hz with 1280×720 resolution and saving uncompressed frames to the laptop's solid state drive (SSD). Writing from three cameras at this rate required a write speed of over 200 MiB per second, so the high maximum write speed offered by the SSD (compared to mechanical hard drives) was essential.

This data was captured using a multi-threaded application, and a thread-safe queue. As frames were acquired from the devices, they were stored in the queue alongside the UTC time in which they were acquired, with a separate thread polling the queue and and writing frames to disk with the timestamp as the filename (following the convention of the EuRoC dataset, Burri et al., 2016).

Approximate Synchronisation The S6 video data provided similar data to the Senz3D cameras, but with a higher resolution of 1920×1080 with the drawback of less precise synchronisation to the other cameras. All cameras recorded a hand-clap at the beginning of each sequence in the case that we needed a rough synchronisation of these other devices later. For our purposes, precise synchronisation of just the Senz3D cameras was sufficient.

The data captured with the Theta camera was also approximately synchronised in this way, but its usage is beyond the scope of this thesis. The main purpose of the Theta capture was to open the possibility of a panorama-based mapping and tracking procedure similar to Ventura and Höllerer (2012). However, we leave this for future work, and instead focus on localising with the SfM photograph data.

3.3 Spherical Spectator Video

The previous section focused on capturing data with the purpose of reconstructing the stadium environment. In this section, we outline our capture process for data that

replicates stationary spectator motion. The purpose of this data is to evaluate how tracking algorithms can perform with limited translational movement in large spaces.

In this case, we do not restrict ourselves to sport stadiums alone, and instead capture data in a range of large environments. For this dataset, we have the following criteria common to all captured sequences:

- Video captured from a mobile device.
- The user holds the device casually at approximately arm's length.
- The user is sitting or standing with only pivotal motion.

These criteria correspond to an outward-facing spherical motion as described by Ventura (2016), and covered in more detail in Chapter 5. As previously discussed, many public datasets do not contain large portions of spherical motion, and any spherical motion within is coincidental. We therefore see value in capturing our own spherical datasets for the purposes of evaluating tracking methods for stationary users.

3.3.1 RealSense Synchronisation

In some of these sequences, we also captured pose data from a RealSense T265 tracking camera. The mobile device and T265 were mounted to a single beam with as small a distance as practical (approximately 5 cm) between the T265 tracking origin, and the mobile device lens. The idea behind this was to capture pose data from state-of-the-art multi-sensor tracking hardware to use as a baseline for comparing monocular tracking methods.

To synchronise the two devices, all pose data recorded by the T265 was recorded with UTC timestamps. To associate accurate UTC time to the mobile device, the current UTC timestamp was displayed on the laptop screen, and all tracking sequences view the screen at the beginning of the sequence, as shown in Figure 3.7. This synchronisation section was trimmed from the final data, with the timestamp propagated forward based on the mobile device's capture frame rate.

3.4 Casual Circular Video

The last type of data we are interested in is circular videos, such as those that would be captured by a user capturing a panorama of their environment. Circular panoramic

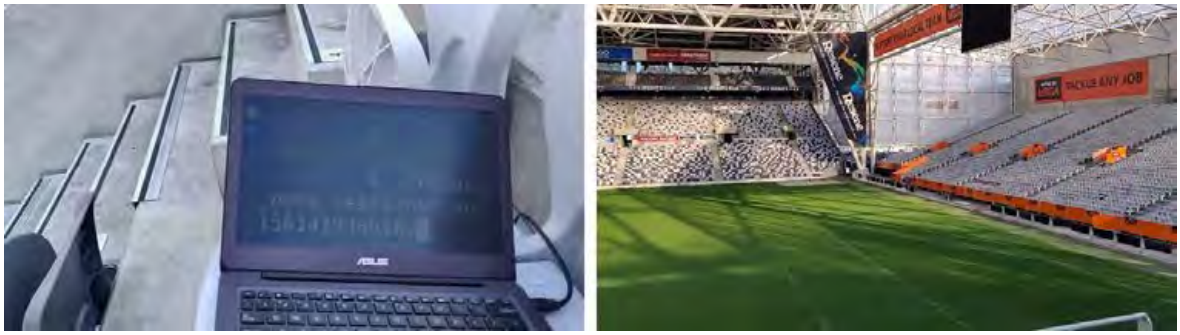

Figure 3.7 Synchronising the RealSense and mobile device capture. Left: an initial frame of video displaying the current UTC timestamp (white) on the PC. Right: the sequence begins looking out into the field for tracking evaluation.

videos are a special case of the general spherical videos described above.

In our case, we have the same requirements as above, except the user must attempt to keep the device level, and rotate in a complete circle (i.e. the video should end with approximately the same view orientation as it started with). We allow for small deviations, such as what would be captured by a casual user. For example, lifting their feet while turning their body, or ending the capture at a slightly lower position than they started.

The latter case is common due to the weight of the device naturally pulling the user's hands lower as they rotate. In our case, we provide no feedback to the user of this kind of deviation. But in typical panorama capture scenarios, users are provided live feedback and hints to guide their trajectory. However, as we are interested in the robustness of tracking stationary users, it makes sense to test the more challenging scenario where the deviations are potentially greater.

There do exist some datasets which meet these criteria already. Works such as Megastereo (Richardt et al., 2013) and MegaParallax (Bertel and Richardt, 2018; Bertel et al., 2019) offer a Mountain dataset, and Ventura (2016) offers a Street dataset. We make use of these datasets, as well as capturing our own, while following the movement style shown in Figure 3.8.

3.5 Dataset Summary

In this chapter, we outlined what is available in terms of localisation and tracking datasets, as well as their limitations. We detailed our capture hardware and custom-

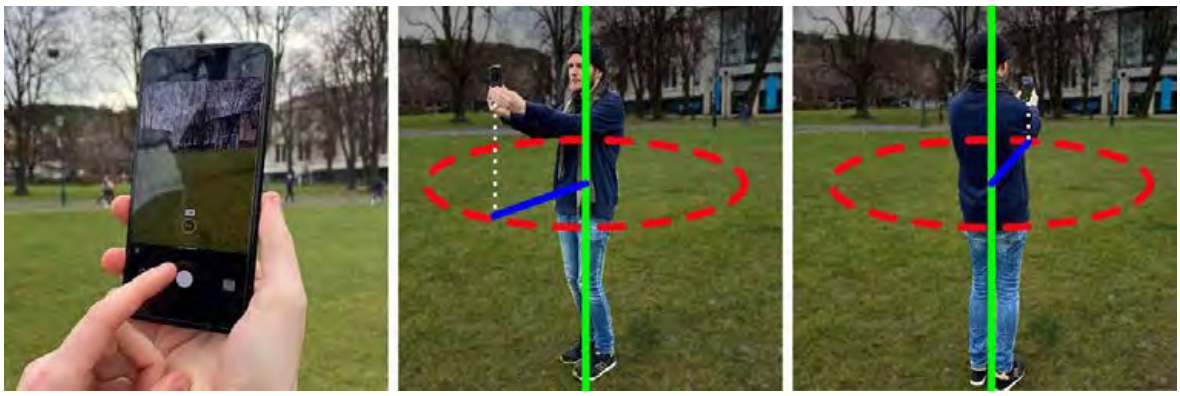

Figure 3.8 An example of the capture process for casual circular videos. Left: the user uses two hands to activate the recording function. Centre and right: the user begins rotating around the Y axis (green). The user approximately tries to keep the device level at a fixed distance in the Z axis (blue) from their body, moving the device in a circle on the X-Z plane (red dashes).

built camera rig, as well as our method for synchronisation of the multiple sensors. A summary of all key datasets used in this thesis is shown in Table 3.1. We include Photo datasets (used in Chapter 4), Spherical Video datasets (used in Chapter 5), and Circular Video datasets (used in Chapter 6). In the remainder of this thesis, we investigate the feasibility of using current state-of-the-art localisation approaches with this data. We also develop new tracking solutions that can process it robustly in the context of AR for sport spectators, and stereo panorama generation for VR viewing.

Table 3.1 Summary and details of all key datasets used in this thesis.

Name	FPS	# Frames	Length (s)	Resolution	Ref. Pose
Photo					
Stadium-FBS	—	1125	—	5184 × 3456	—
Stadium-MCG	—	49	—	4608 × 3456	—
Spherical Video					
Auditorium-close	30	457	15.23	1920 × 1080	—
Auditorium-far	30	471	15.70	1920 × 1080	—
River-close	30	492	16.40	1920 × 1080	—
River-far	30	466	15.53	1920 × 1080	—
Courtyard	30	487	16.23	1920 × 1080	—
Field	30	476	15.87	1920 × 1080	—
Garden	30	461	15.37	1920 × 1080	—
Lab	30	475	15.83	1920 × 1080	—
Stadium-1	60	750	12.50	1920 × 1080	—
Stadium-2	60	703	11.72	1080 × 1920	—
Spectator-1	240	4980	20.75	1920 × 1080	RealSense
Spectator-2	240	7170	29.88	1920 × 1080	RealSense
Spectator-3	240	8490	35.38	1920 × 1080	RealSense
Spectator-4	240	10620	44.25	1920 × 1080	RealSense
Circular Video					
Shrine	30	602	20.07	1080 × 1920	—
NaturePath	30	697	23.23	1080 × 1920	—
Campus	30	667	22.23	1080 × 1920	—
Courtyard360	30	690	23.00	1080 × 1920	—
Lab360	30	695	23.17	1080 × 1920	—
Park360	60	978	16.30	1080 × 1920	—
Stadium360	240	7007	29.20	1080 × 1920	—

Chapter 4

Localisation for Stationary Users

In the previous chapter, we covered the details of our data capture process, as well as a discussion of the standard datasets for localisation in the literature. As we want to focus specifically on stationary use-cases such as the ARSpectator, we captured our own localisation datasets in two sports stadia.

In a stadium context, the presence of spectators can create large regions of point features in the stands which change in appearance between events. Towards this, we first look to localise from the strong linear structures commonly found in a stadium environment. We apply SoftPOSIT for this purpose, a geometry-based method first introduced by David et al. (2003), and test it on various datasets. Our datasets include handmade computer-aided design (CAD) models, as well as complex SfM models which have so far been unexplored for localisation with this method. This part of the chapter has already been published in Baker et al. (2018).

Beyond this, previous research has shown that many localisation approaches successfully rely on descriptors (e.g. SIFT, Lowe, 2004), and robust outlier rejection (e.g. RANSAC, Fischler and Bolles, 1981) to handle dynamics. For this reason, we also evaluate state-of-the-art appearance-based methods in the second part of this chapter. Our evaluation showed that these methods can be used effectively for localisation, with promising initial results toward localisation at live dynamic sports events as well.

Figure 4.1 Two views of a virtual scoreboard, and animated flag floating above a cricket pitch, viewed by a moving camera. Sport broadcasts that include these enhanced visualisations are becoming increasingly common, which motivates working toward an enhanced experience for on-site spectators. *Source, with permission: ICC, Cricket World Cup 2019, https://cricketworldcup.com*

4.1 Introduction

In this chapter, we investigate localisation approaches particularly in the context of the Augmented Reality Spectator (ARSpectator) application that was first introduced in Zollmann et al. (2019). The aim of the ARSpectator application is to provide visualisations and statistics to on-site spectators of sport events to enhance their overall experience. There are countless examples of such visualisations being shown on broadcast TV. In particular, large multi-national competitions such as the ICC Cricket World Cup provide this enriched content to home viewers using AR localisation and tracking methods (see Figure 4.1).

A common approach to localisation for AR is to use a prior model and an image to perform the localisation. These methods often use the appearance of features to find 2D-to-3D correspondences and compute the camera pose (Sattler et al., 2011; Li et al., 2012). In some dynamic environments, the appearance is constantly changing, so finding these feature matches is potentially unreliable. An example is a sport stadium, where a prior model may not be able to capture the appearance of the spectators at any given sport event.

By modelling only the static features in the environment, and detecting those features in 2-D, we can eliminate problems caused by appearance-based descriptors. Line features can help achieve this, since lines will often correspond with structural components of a given scene, whereas points may correspond with textures. Previous work has demonstrated that line features can be effectively used for localisation, such as the work by Zhang and Koch (2013), but there has been little research in line-based localisation with appearance-free lines.

Work by Reitmayr and Drummond (2006) supports the idea of using lines (or edges) for correspondence matching for model-based tracking. In their work, they use a textured 3-D model and an appearance-based edgel (i.e. a pixel corresponding to a textural edge) matching method to align a projection of their 3-D model to their mobile camera feed. Their method requires a pose initialisation from a known location, so cannot be directly applied to localisation from arbitrary positions. This work shows that edges can be applied reliably to localisation and tracking problems, though edge appearance is also an important factor.

Appearance-free methods also exist such as SoftPOSIT (David et al., 2002b). SoftPOSIT is a point-based algorithm that computes both correspondences and pose simultaneously without using appearance-based descriptors. David et al. (2003) show that this method can also be applied to lines. SoftPOSIT has been tested thoroughly in synthetic test cases, and some real applications using CAD models. The results are promising, but it is often not possible to attain CAD models of the environment. Structure-from-motion (SfM) has been shown to produce accurate models of large environments easily (Agarwal et al., 2009), but SoftPOSIT has not been thoroughly tested on these types of models.

There have been a variety of related works in the area of model-based localisation, (also known as image-based localisation). Sattler et al. (2011) show a method for localising images against a large prior point cloud model. Their initial work used 2D-to-3D feature-to-point correspondences, but the authors later expanded their work to employ an efficient search that leverages both 2D-to-3D and 3D-to-2D correspondences using SIFT descriptors readily available in the model (Sattler et al., 2012a). The work of Li et al. (2012) is similar in that it combines "forward" and "inverse matching", also making use of SIFT. In our case, we cannot guarantee that the appearance of the environment will be consistent with the model, so we first investigate the most promising appearance-free approach, SoftPOSIT.

As previously introduced, SoftPOSIT is a method that can be applied to line-models to compute pose without relying on the appearance of the features. SoftPOSIT and some of its variations have been applied in some basic real-life cases, Diaz and Abderrahim (2007) employ SoftPOSIT for localisation of mechanical parts, for example. Since SoftPOSIT can be applied to line models and does not use appearance-based line descriptors, we elect it as our first localisation method for our experimentation.

SoftPOSIT seems to be the most appropriate method that theoretically accommodates our requirement of appearance-free line models. In the following two sections, we

investigate the effectiveness of SoftPOSIT in a range of scene complexities and two model formats. With our initial objective being to localise in a sport stadium, we also demonstrate the limitations of this method when applied to other complex SfM models. As SoftPOSIT has not thoroughly been applied to real data, our contribution is an overview of the kind of performance that can be expected from this method under real-life use cases.

4.1.1 Limitations of Tracking

Alternatively, the simultaneous localisation and mapping (SLAM) approach aims to create a map of the environment and localise the user with respect to the map on the fly. PTAM by Klein and Murray (2007), and ORB-SLAM2 by Mur-Artal and Tardós (2017a), are some key examples of this approach. While showing huge potential for AR, these SLAM methods have limitations which make them unsuitable for stadium localisation. In our use-case, where only monocular vision is available, these SLAM methods require significant translational motion to produce the parallax required to build accurate maps.

A key issue we face is the lack of interoperability between state-of-the-art localisation and tracking systems. While some SLAM-based tracking systems have shown the possibility of relocalising (Williams et al., 2007), and are capable of reusing maps (Mur-Artal and Tardós, 2017b), in practice, these systems cannot reliably relocalise when significant changes affect the appearance of the environment (for example relocalising at a different time of day, or with different weather conditions).

In our testing with ORB-SLAM (Mur-Artal et al., 2015), we found that tracking in a sport stadium can be made relatively stable (as shown in Figure 4.2), but only under very specific and impractical usage conditions. We were able to produce these results provided that the initial map was created with significant translational movement, and that the same camera was used for relocalisation. Additionally, producing precise field-registered visual content required careful manual alignment between the SLAM point cloud, and a prior CAD model of the field dimensions.

While this result looks promising on the surface, it is unfortunately impractical. For the purpose of localising and tracking sport spectators, it is unreasonable to expect them to stand and move to create a reliable reconstruction to track and relocalise from. Furthermore, relocalising against a map created on a specific user's device is not useful for creating registered virtual content which is displayed uniformly to multiple users

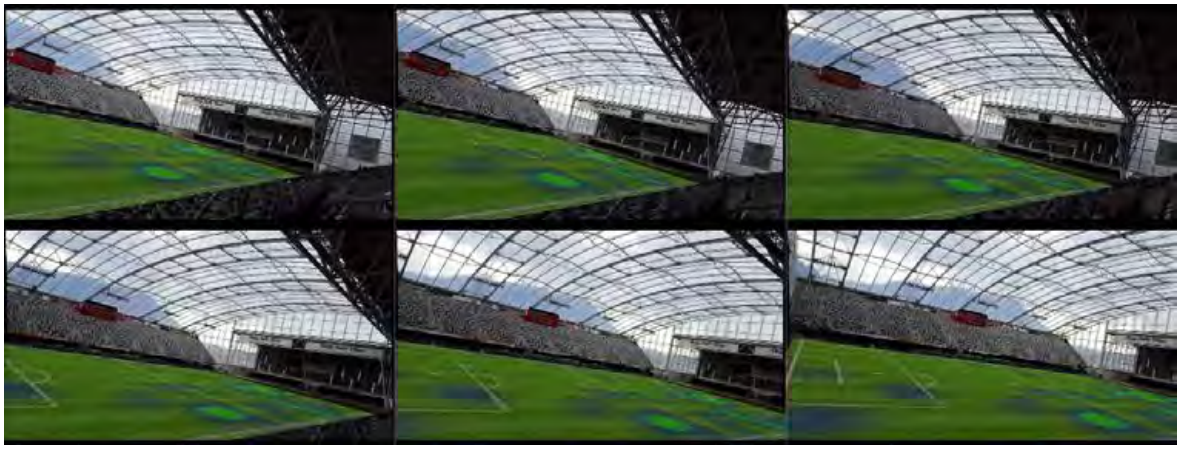

Figure 4.2 An example of ORB-SLAM tracking via relocalisation in a stadium, with an AR overlay. This type of AR visualisation is only possible with ORB-SLAM when initialised with significant translational movement, and careful alignment of the SLAM map to the existing pitch dimensions.

with varying perspectives. Ideally, a pre-built map would be distributed to the users of the system, and AR content would be authored in this co-ordinate frame. For this reason, we first investigate localisation in this chapter, then revisit tracking later in Chapter 5.

4.1.2 Chapter Overview

In this chapter, we aim to solve the problem of localising stationary users within a sport stadium in the context of the ARSpectator. We make the following contributions toward this goal:

- Implement and investigate the usage of SoftPOSIT for localisation with a range of CAD and SfM models.
- Provide implementation details of a simple Bag-of-Words style localiser that operates on SfM models using ORB for image retrieval and SIFT for localisation.
- Compare and evaluate various state-of-the-art localisation approaches with our own SfM models of two sports stadia. This is a novel usage scenario, as this type of localisation in a stadium context has not been investigated before.

First, we investigate the feasibility of using SoftPOSIT, a geometry-based approach to localisation in Section 4.2. We discuss our results on computation time and localisation rate in Section 4.3, with discussion in Section 4.4.

In Section 4.5, we introduce state-of-the-art methods for localising using image descriptors (image-based localisation), as well as our data preparation. In Section 4.6, we introduce the details of the state-of-the-art approaches by Sattler et al. (2012a), Brachmann and Rother (2019), and our own simple implementation of a Bag-of-Words (BoW) style localiser which we compare to COLMAP as a baseline (Schönberger and Frahm, 2016). In Section 4.7, we report our results and observations on the image-based techniques. Finally, in Section 4.8, we conclude and discuss pathways for future work.

4.2 SoftPOSIT for Localisation

In this section, we outline our method for evaluation of SoftPOSIT. Since we already know from previous research that this method works in basic cases, we build a dataset of increasing complexity from a small CAD model of a line-based object to a large 3-D reconstruction of a sport stadium. The process consists of ground-truth pose acquisition, model creation, and localisation with respect to this model with SoftPOSIT. The method is applied to five test cases across two types of models. In the following subsections, we will explain our line-based SoftPOSIT implementation, our two model types, our method for acquiring ground-truth pose, our pre-processing steps, and our datasets.

As explained in Section 4.1, we choose SoftPOSIT as our candidate for localisation since it is appearance independent, and requires less computation compared to other approaches. In this section, we explain our implementation of SoftPOSIT, and some pre-processing steps that we introduce for reducing computational complexity.

SoftPOSIT is an algorithm that can be used for model-based localisation. It takes as input an initial estimate of the pose, a 3-D model (line or point based), and an image of the model (David et al., 2002b, 2003). The model itself must be of sufficient detail that the features represented can also be captured from mobile device cameras. The algorithm has been extensively tested on synthetic examples, and virtual examples (David et al., 2002a).

The method works by using the pose estimate to project model lines into the image. It then creates an initial assignment matrix based on the geometric differences between the projected 3-D lines, and the observed 2-D lines. This step is based on the SoftAssign method of Rangarajan et al. (1997). This matrix is normalised, and then used by the POSIT algorithm to produce an updated pose estimate (DeMenthon and Davis, 1995).

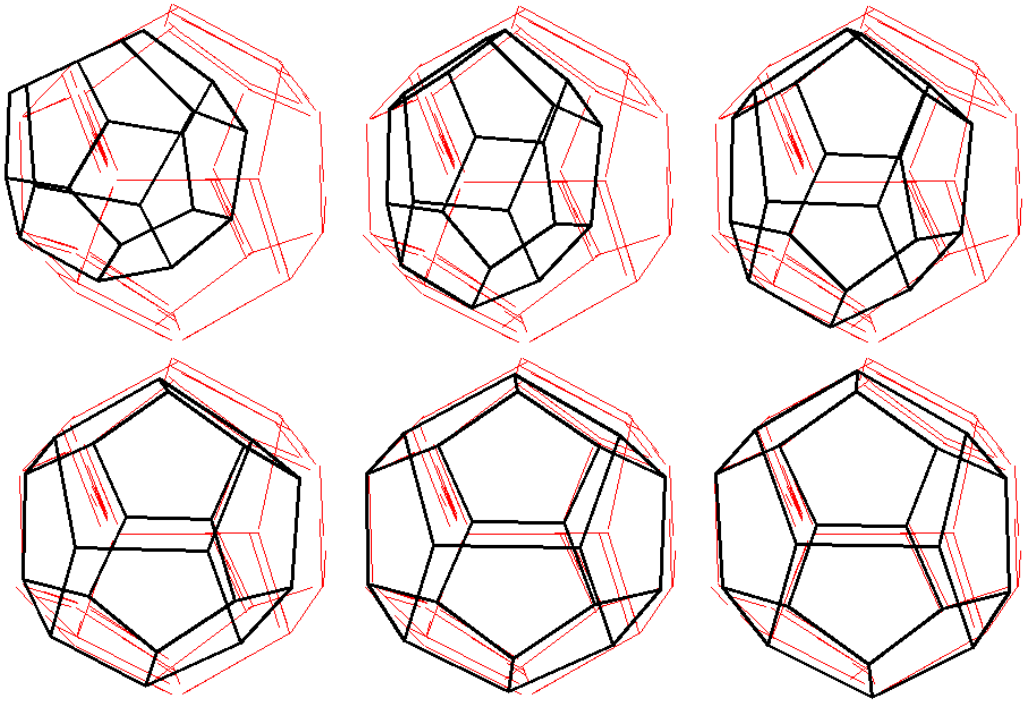

Figure 4.3 Six iterations of SoftPOSIT running on our basic Dodecahedron case (top-left through to bottom-right). The red lines represent the 2-D lines in the photograph detected by the line segment detector (LSD, Von Gioi et al., 2010). Black lines represent the CAD model rendered with the current pose. Initially, the pose estimate is far from correct, but iteratively gets closer until the black and red lines align. The first six iterations are shown here, but more iterations can result in more accurate alignment.

Finally, this process is repeated for several iterations until an accurate pose is achieved. Figure 4.3 shows the first six iterations on our basic Dodecahedron test case.

SoftPOSIT has several parameters that we fix empirically in our experiments. The parameter α acts as a threshold for the assignment matrix. When 3-D lines are projected using the pose estimate, each 3-D line has its difference to each 2-D line computed, and differences above this threshold will be assigned a correspondence that tends to zero. The difference is computed as a weighted combination of the angle between the lines, and the distance between their endpoints. We found that an α value of 0.5 produced good results.

The second parameter β controls the rate of convergence and is increased at each iteration of the algorithm. Small values for β will search a larger space of possible poses but can lead to larger execution times. As suggested in the original work, large β values are better suited for cases where we know that the pose estimate is close to the actual pose (David et al., 2003). We have this parameter set to 20. We observed

that values lower than 20 caused the first few iterations of SoftPOSIT to consistently diverge from the correct pose on our Stadium test case.

4.2.1 Model Types

There are two different types of 3-D models that we consider. The first, which we refer to as CAD models, are created manually in a 3-D modelling software, where all visible lines on the object are represented in the 3-D model. The physical objects that we targeted for these cases have simple geometry and lack complex texture which make them suitable for manual modelling. Our CAD models were measured and modelled to scale using Blender (Blender Foundation, 2016).

The second model type is a line-based SfM model. For these models, we use the point-based VisualSFM system by Wu (Wu, 2013; Wu et al., 2011), and then process the resulting point model with Line3D++ (Hofer et al., 2017) to create the final line model. In this chapter, we assume that creating these models in advance is a required step for localisation in a large space such as a stadium. This is a fair assumption, as these types of software are straightforward to use, and can produce high-quality models with little effort. In our stadium example, we captured 604 photographs from various locations on one stand and fed them through the SfM pipeline above to create the final model. Note that this is based on an early version of the Stadium-FBS dataset described in Chapter 3. The model used here was reconstructed from fewer images and is complete except for one stand (behind the camera in all test images).

4.2.2 Ground-truth Poses

In our SfM test cases, we simply take the output generated by VisualSFM and consider these as our ground-truth poses. The images we use for testing the localisation method are a random selection of 20 SfM input images.

For the CAD models, we placed a chessboard marker of known dimensions at a measured distance from the target object. Using standard techniques and known intrinsic camera parameters, we detect the chessboard corners in the image, and use this to compute the pose (Zhang, 2000). We then offset the origin of our 3-D model by its known translation from the chessboard origin. Finally, the chessboard is masked out of the input when passed to the localiser to prevent the chessboard lines from interfering with the localisation result.

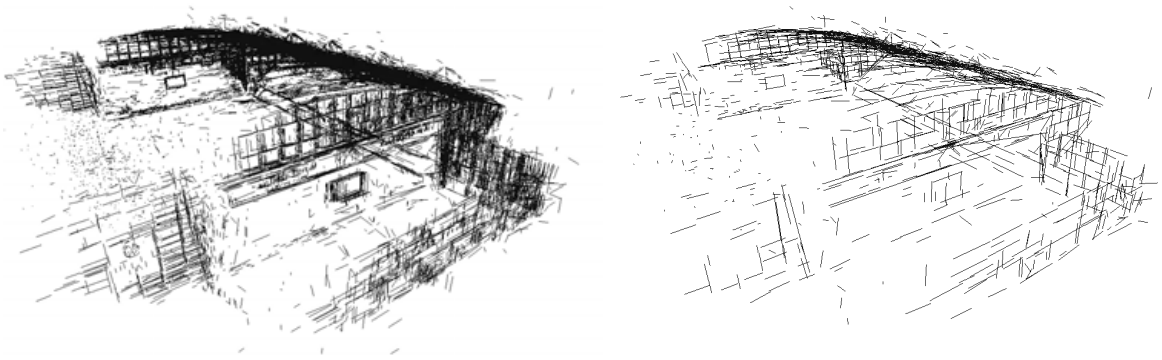

Figure 4.4 Example output from k-means clustering of the 3-D line model, shown from two views. Left: the original output from Line3D++ which contains over 22000 lines. Right: the output of performing k-means with k set to approximately 2,200 (10%).

While both methods can be expected to produce a small amount of error, our main interest is in the rate of localisation, and speed rather than accuracy. In our experiments, we compute a reprojection error by projecting the model lines with both the ground-truth pose, and the estimated pose, and comparing the corresponding projections. We then introduce an error threshold which will encompass the small errors in our measurements, resulting in a metric for the rate of success.

4.2.3 Pre-processing

We found that the execution time of SoftPOSIT was high when run with large SfM models. Our initial stadium model produced by Line3D++ contained over 22000 3-D lines. An example photograph at original resolution contained over 9000 lines. Execution time at this level of complexity can take several minutes per iteration of SoftPOSIT and is therefore not practical, so we considered some approaches to reduce the complexity for our experiments.

We reduced the number of 2-D lines by scaling our input images down to quarter-size. This scaling was applied to all five of the test cases in our dataset. For the SfM models, the number of 3-D lines was further reduced by clustering similar 3-D lines in the model using k-means (Lloyd, 1982), and storing only the representative lines for each cluster. The result of our clustering (with small k for illustration purposes) is shown in Figure 4.4. For each SfM model, we set k to be equal to half the number of 3-D lines in the original model, which is notably denser than the example on the right in Figure 4.4.

Computing k-means clustering involves a metric for determining the distance between the line segment observation and the cluster centres. In our implementation, the distance was taken as the 6-D distance between each line described as the union vector of its endpoints. Two 3-D lines can be identical but have different ordering of their endpoints when they are described as a 6-D vector. For comparing two line segments $(\mathbf{p}_1 \mid \mathbf{p}_2)$ and $(\hat{\mathbf{p}}_1 \mid \hat{\mathbf{p}}_2)$, we use

$$d_1 = || \mathbf{p}_1 - \hat{\mathbf{p}}_1 ||^2 + || \mathbf{p}_2 - \hat{\mathbf{p}}_2 ||^2, \tag{4.1}$$

and

$$d_2 = || \mathbf{p}_1 - \hat{\mathbf{p}}_2 ||^2 + || \mathbf{p}_2 - \hat{\mathbf{p}}_1 ||^2, \tag{4.2}$$

where $\mathbf{p}_n$ is the n^{th} endpoint of a line segment, and take the minimum of the two distances d_1 and d_2 to determine which of the line pair's endpoints are corresponding, and re-order the endpoints in the vector if necessary.

Since we are assuming that we have an initial estimate of our pose (as this is a requirement for SoftPOSIT), we employ one other basic line filtering mechanism. We assume that the pose estimate is reasonably good. Before SoftPOSIT is run, we remove all lines from the 3-D model that are not visible given the initial pose estimate. We achieve this by projecting the endpoints of the 3-D line segments into the image using the pose estimate, and discard lines which do not fully project within the image frame. We discard these incomplete lines, as the line distance methods used by SoftPOSIT (described below) require two visible endpoints. This process removes a large portion of lines which are in the model but not the image. Some observed model lines will also be removed as the initial estimate is not always perfect, but the idea is that SoftPOSIT will be robust to some occlusion as was shown by David et al. (2002b).

4.2.4 Datasets

Our objective is to explore SoftPOSIT's ability to function under test cases of varying difficulties and model types. We constructed a dataset of five test cases, where each test case contains 20 sample photographs. The subject of each test case is an object or environment of which we have a prior model. Figure 4.5 demonstrates a sample image from each test case, and the corresponding 3-D line model that is tested against.

The dataset consists of CAD models of two subjects: a wire-frame dodecahedron, and a snake puzzle. It also contains SfM models of three subjects: the same snake puzzle,

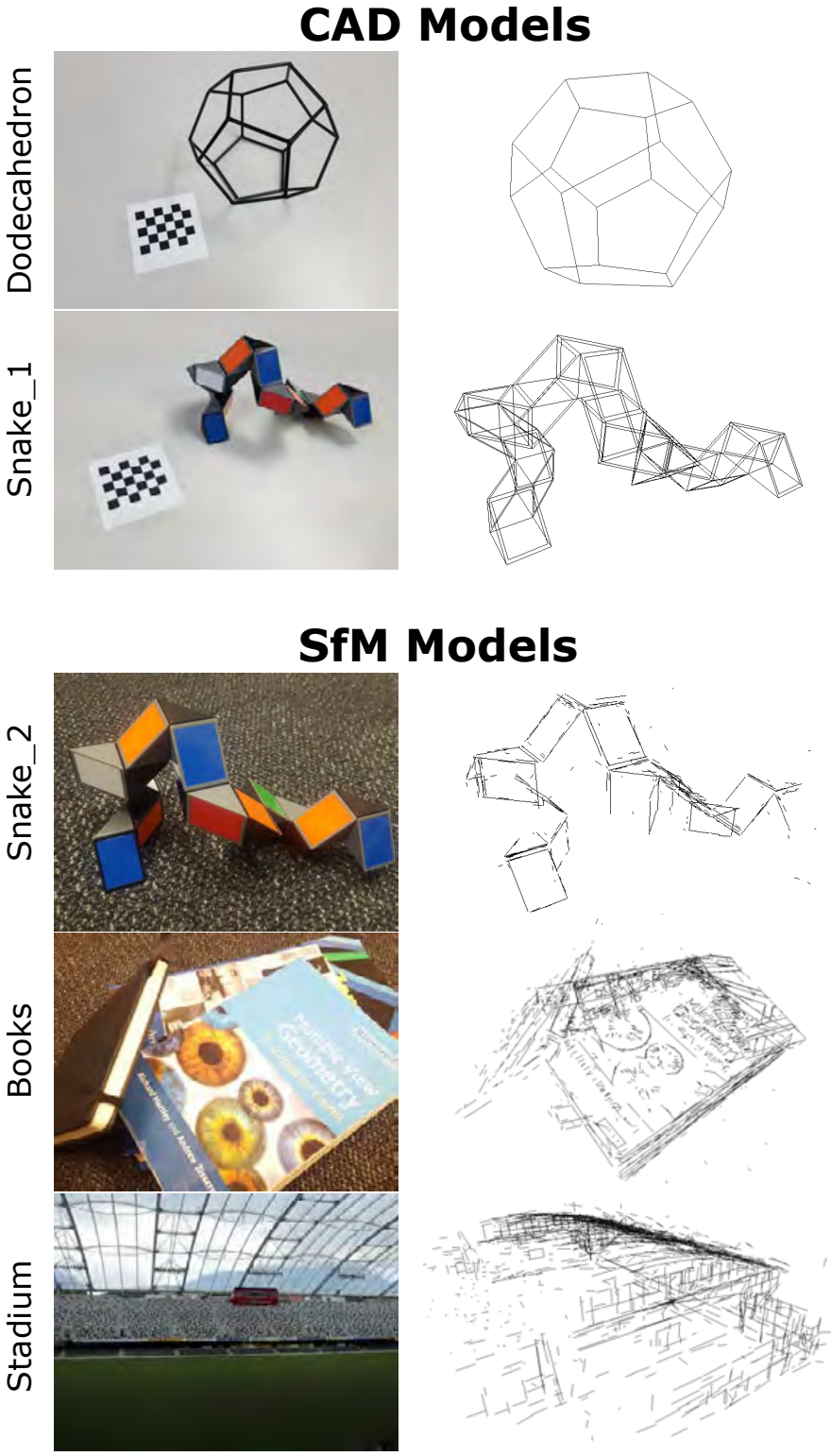

Figure 4.5 The left column shows a sample image from each test case, the right column shows a render of the 3-D model we used for that test case. The CAD models were created manually and contain all of the visible lines in the object. The SfM models are generated by a combination of VisualSFM and Line3D++ and are generally sparser and noisier than the CAD models.

a stack of books, and a sport stadium. The test cases cover some of the key properties that can cause problems for a model-based localiser, such as occlusion, noise, distractor lines, and imperfections in the model.

4.3 Results

In all tests, we set an upper limit of thirty seconds on the execution time unless stated otherwise. We deem any execution time over thirty seconds to be well and truly over the threshold of what is useful for an AR application, even allowing for careful optimisation. In these cases, the tests are terminated early, and the reprojection error is taken based on the last computed pose.

To determine the accuracy of the localisation, we project the visible model lines into the image using both the ground-truth pose, and the final pose output by the localiser. We then compare these projections and compute the reprojection error as the average of the distances between each line endpoint. We present the final error as a percentage of the image width, as well as pixels at 1920×1080 resolution.

We are also interested in the rate of successful localisation. To determine the success rate, we set a threshold on the reprojection error, and compute the percentage of test cases that produce a pose that falls below this threshold. In the following experiments, we set the error threshold to 2% of the image width. This is a generous threshold, corresponding to over 30 pixels at 1920×1080 resolution. This value was chosen to encompass any error contained in our model and pose data.

Our first experiment looks at the effectiveness of our pre-processing methods, the second and third experiments show the overall results of SoftPOSIT on our dataset under both basic and typical use cases, respectively.

4.3.1 SfM Model Pre-processing

In Section 4.2.3, we introduced the idea of pre-processing the SfM models to reduce the complexity of localisation with SoftPOSIT. In this experiment, we show that the execution time of SoftPOSIT can be reduced using this pre-processing method. In the Stadium case, we have a large SfM model that details nearly all lines in the stadium. However, in most cases, a lot of these 3-D lines do not lie in the spectator's field of view.

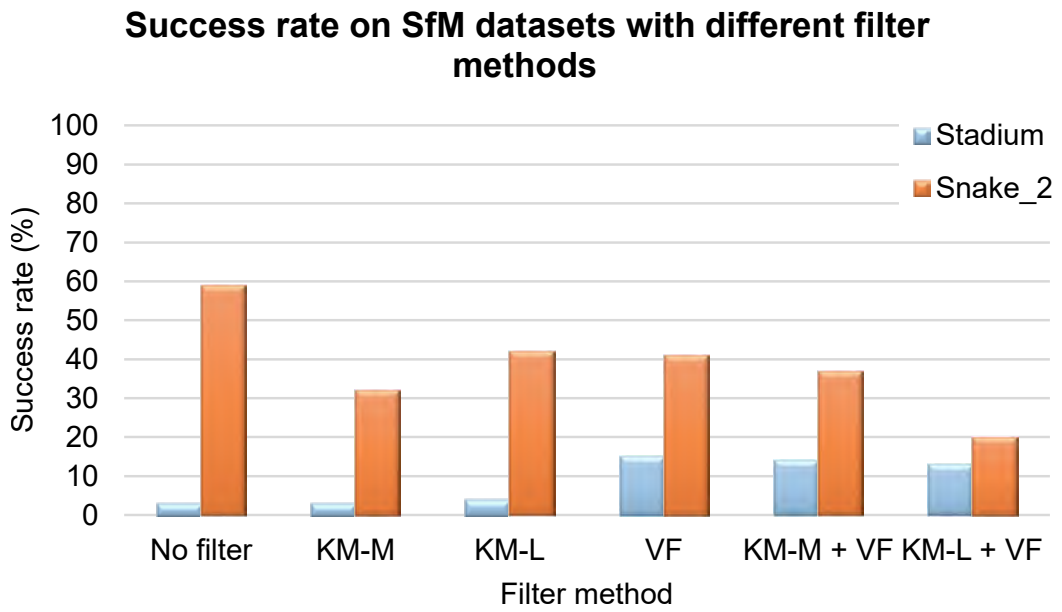

Figure 4.6 Comparison of success rate of different line clustering methods on two SfM models in the dataset. The filter methods are as follows. KM-M: k-means outputting the means of each cluster. KM-L: k-means outputting the longest line in each cluster. VF: visibility filtering.

In this experiment, we tested 2 variations of k-means clustering of the 3-D lines, and a visibility filter based on the initial pose estimate. The first variant of k-means filters the 3-D model and outputs the means (or cluster centres) as the final model (we refer to this as KM-M in Figures 4.6 and 4.7). The second variant of k-means outputs the longest input line assigned to each cluster as the final model (referred to as KM-L). The visibility filter is applied to each individual test case, rather than the entire model. It essentially discards any lines that are not visible, given the initial pose estimate (referred to as VF).

We apply this experiment to two of our SfM datasets, but not the CAD datasets, as the purpose of the filtering is to reduce spurious lines and noise, which are not present in the CAD models. We leave out the Books SfM case, as we show in the following subsections that this case is infeasible. We apply a random perturbation to the ground-truth pose of each sample to simulate real-world inaccuracies in the pose estimate. To generate this error, we iteratively add a small randomly generated rotation increment until the overall reprojection error reaches a randomly selected value between 1% and 10% of the image width.

Figures 4.6 and 4.7 show the results of this first experiment. All test samples were capped at an execution time of 5 minutes instead of 30 seconds to increase the rate of success. Even this large cap is reached, which accounts for the low success rate of

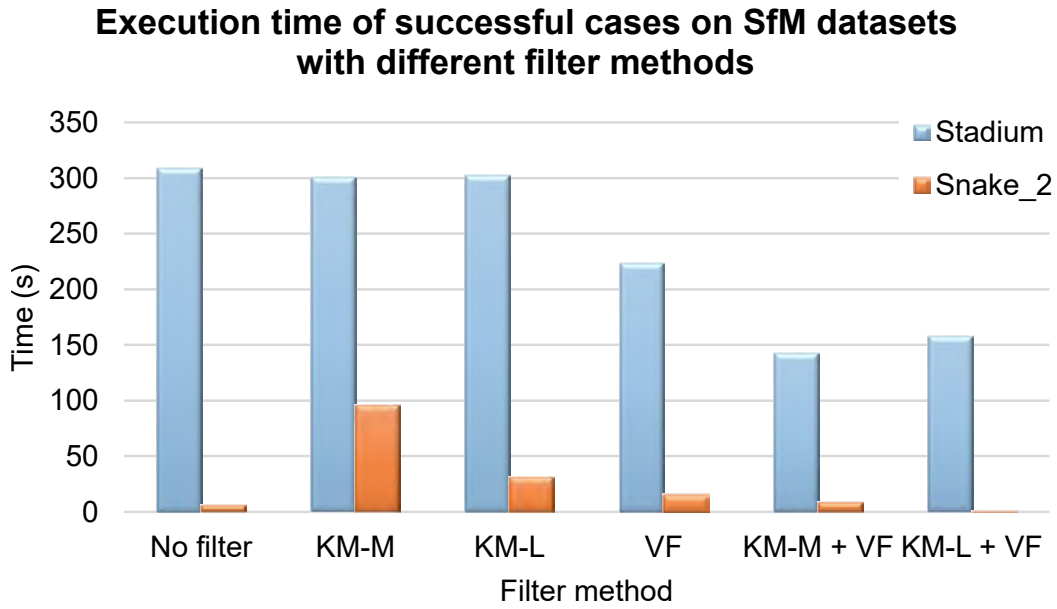

Figure 4.7 Comparison of execution time of different line clustering methods on two SfM models in the dataset. The filter methods are as follows. KM-M: k-means outputting the means of each cluster. KM-L: k-means outputting the longest line in each cluster. VF: visibility filtering. The computation time includes just the SoftPOSIT execution, and the image pre-processing.

the Stadium SfM cases with KM-M, and KM-L filtering. The VF filtering method seemed to have the most visible effect on execution time, bringing the Stadium SfM test samples below the 5 minute execution cap and increasing the success rate.

The Snake SfM results suggest that if execution time is not a priority, the best results will be obtained with no filtering applied. But in practice, it is not feasible to run for extended periods of time. So, applying a filtering method such as VF or a combination of VF and KM-M, can reduce the execution time. More thorough testing of the effect of these filtering methods is left for future work. For the remainder of the experiments, we set the execution cap to 30 second, and use KM-M + VF since it resulted in the best execution times with similar success rate compared to VF for the Stadium SfM test case.

4.3.2 Basic Use-case

Table 4.1 shows the results of our experiments in an ideal use case. In this experiment, we fix SoftPOSIT's parameters, and provided the ground-truth pose as the initial pose estimate. This was done as a base case, to determine how effectively it can localise under the unlikely but possible circumstance that the pose estimate is highly accurate.

Table 4.1 Results from SoftPOSIT in the basic use-case where the provided initial pose estimate is accurate, across a dataset of two CAD and three SfM models. Error is averaged over successful samples and reported as a percentage of image width.

Test Set Size=20 ($n = 20$), for all cases.

Test case	Dodec.	Snake_1	Snake_2	Books	Stadium
Model type	CAD	CAD	SfM	SfM	SfM
Success (%)	90	95	40	0	20
Err. M (%)	0.888	0.980	0.865	—	0.408
Err. SD (%)	0.312	0.323	0.338	—	0.224
Time M (s)	0.050	1.59	0.691	2.47	19.0

The results show that even in the basic case, localisation using SfM models is difficult. The CAD test cases worked relatively well, with an average success rate of over 90% for both. But all of the three SfM cases had low success rates, and the Books case failed to produce an accurate localisation result in all of its samples.

The Books case, while not completely planar, contains a large proportion of lines on the surface of the top-most book. Scenes consisting of co-planar lines are known to be a failure case of the POSIT algorithm, so this result is not surprising. A variation of POSIT (pose from known correspondences) for co-planar scenes is presented by Oberkampf et al. (1996). There has been no work in applying this to SoftPOSIT for points or lines when the correspondences are unknown, but this could be investigated in future.

An interesting result is the Snake test, for which we had both CAD and SfM models. Our SfM pipeline was unable to capture all the lines that were present in the object, so the Snake SfM model was sparse compared to its CAD counterpart. In the cases where the SfM Snake succeeded, it did produce a slightly lower reprojection error than the CAD model. This could be to do with imperfections in the physical joints of the Snake puzzle, which caused it to misalign slightly with the "perfect" CAD model. Since the SfM model was produced using actual observations of the object's edges, the lines that are present have the potential to be highly accurate.

It is also notable that while the Stadium SfM case had a low success rate, it also produced a relatively small reprojection error in success cases. While the success rate leaves much to be desired, it shows promise in its application to AR with an average reprojection error of 0.408% image width (7.8 pixels at 1920×1080 resolution).

Table 4.2 Results from SoftPOSIT in a more typical use-case where the provided pose estimate is inaccurate, across a dataset of two CAD and three SfM models. Error is averaged over successful samples and reported as a percentage of image width.

Test Set Size=20 averaged over 5 runs ($n = 100$), for all cases.

Test Case	Dodec.	Snake_1	Snake_2	Books	Stadium
Model type	CAD	CAD	SfM	SfM	SfM
Success (%)	94	81	36	0	12
Err. M (%)	0.926	1.049	0.744	—	0.496
Err. SD (%)	0.392	0.475	0.456	—	0.367
Time M (s)	0.080	1.65	0.680	2.56	18.7

4.3.3 Typical Use-case

In this experiment, we look at the performance of SoftPOSIT under a more typical use case, where the input pose estimate is erroneous. In an AR application, such an estimate could be obtained from mobile device sensors.

In this experiment, we do not use mobile sensors to acquire a pose estimate. This is because our SfM models are scaled arbitrarily, and we do not assume that our models are aligned to GPS co-ordinates (geo-referenced). Instead, we artificially add error by adding random adjustments to the ground-truth camera pose. Since SoftPOSIT localises by minimising the 2-D error of projected model lines, we pick a pose adjustment that results in a random reprojection error in a predetermined range. In this way, the 2-D error added is consistent regardless of the scale of the model.

The results in Table 4.2 show the success rate, error, and execution time of each of our models when the ground-truth pose is modified randomly. We have 20 testing images for each model, and each image is run with 5 random pose modifications. The average success rates and errors over the 5 trials are presented in this table.

Compared to the basic use-case, the success rates are mostly lower as expected. The exception here is the dodecahedron model. This case was simple, was generally insensitive to inaccuracies in the pose estimate, and often succeeded in cases where the estimate was drastically wrong. The higher percentage here could be attributed to the larger sample size (5 runs per image) due to the randomness in the pose estimate.

4.4 Discussion

In the previous section we outlined the problem of model-based localisation, specifically in cases where our image may not be similar in appearance to our prior 3-D model. We implement a line-variant of SoftPOSIT into an SfM pipeline, and apply our method to some challenging real-life examples using different model types.

Our results showed that with some simpler SfM models, good results can be achieved but not reliably, and that more work is needed to achieve good results in complex scenes such as a sport stadium. We also showed that SoftPOSIT can be effective in some real-life cases, but only when the model is simple. In these cases, it can achieve a good rate of localisation with short execution times.

As it is not always feasible to assume the availability of a CAD model, we look more into localisation methods that can be applied directly to SfM models. Taking the same motivational approach as in the previous chapter, we still argue that SfM models are easily obtainable for large environments such as sports stadia and may therefore be suitable as a basis for localisation. In the remainder of this chapter, we broaden our scope and investigate the possibility of localising in a sport stadium with state-of-the-art appearance-based methods.

4.5 Appearance-based Localisation

Earlier, we investigated the use of SoftPOSIT for this purpose since it does not rely on descriptors which encode the appearance of the environment. However, some recent works have found success with localisation methods that utilise the descriptors, despite being used in dynamic environments (Sattler et al., 2016). The success of these methods suggests that descriptor matching may be robust enough to handle the dynamics of a sport stadium.

Previous research on localisation with SfM models can be coarsely grouped into three categories: feature matching approaches (Li et al., 2010), image retrieval approaches (Khan and McCane, 2012; Wolf et al., 2005), and deep learning approaches (Kendall et al., 2015; Yin and Shi, 2018; Brachmann and Rother, 2019). Here, we take an example of each and compare them on several SfM datasets including two sports stadia.

We focus on the most promising open-source systems, namely ESAC (Expert Sample Consensus) released by Brachmann and Rother (2019), Active Search released by Sattler et al. (2012a), and our own simple implementation of a Bag-of-Words (BoW)

localiser that utilises the fbow library (Muñoz-Salinas and Medina-Carnicer, 2020), which is an optimised implementation of the method by Gálvez-López and Tardós (2012). We compare these methods in their ability to localise with our SfM models built using COLMAP (Schönberger and Frahm, 2016), using COLMAP's vocabulary-tree image registrator as a localisation baseline.

4.5.1 Data Preparation

Our main objective is to analyse various localisation approaches in their ability to localise in a sport stadium environment. Due to limited availability of public stadium datasets, we captured our own in two stadia, as outlined in Chapter 3. Sample images from the dataset are shown in Figure 4.8.

The following describes how we prepare the data for evaluation. The first step in our data processing is to create a sparse reconstruction using the training images of each dataset. For this, we used the open-source COLMAP SfM software from Schönberger and Frahm (2016) with default parameters. In this section, we make use of the MCG and FBS stadium datasets that we discussed in Chapter 3.

Melbourne Cricket Ground (MCG) For this dataset, we captured a small dataset of 49 images from several positions within one side of the stadium during a single visit. The lighting condition was sunny, and the stadium was at near-full capacity. This data was randomly split into 39 training, and 10 test images (approximately 8:2 split).

Forsyth Barr Stadium (FBS) For this dataset, we captured 1,125 images from a range of positions primarily from the two opposing main stands, with some taken from ground level. This dataset was captured over two visits to the empty stadium, one overcast, and one sunny daytime condition. This dataset was randomly split into a training set of 900, and testing set of 225 images (8:2 split).

4.6 Overview of Methods

In this section, we provide an overview of our three chosen localisation methods. Our first method is Active Search, based on work by Sattler et al. (2012a). We choose this method as it is frequently used as a benchmark for image-based localisation approaches and is provided open-source. We also investigate ESAC by Brachmann and

Figure 4.8 Four sample images from each of the stadium SfM datasets. Top row: Forsyth Barr Stadium (FBS) dataset, taken on two visits with two lighting conditions. Bottom row: Melbourne Cricket Ground (MCG) captured in one visit, with near-full stadium capacity.

Rother (2019) as it appears to be among the most promising and scalable of all the deep-learning approaches to localisation, while also being provided open-source by the authors. Our third method is our own implementation of a Bag-of-Words localiser, based on ORB and fbow for image retrieval (Muñoz-Salinas and Medina-Carnicer, 2020; Gálvez-López and Tardós, 2012) while using SIFT for registration (Lowe, 1999). These BoW solutions tend to focus on image-retrieval results, and do not provide results with the type of accuracy that can be achieved when they are used for full 6-DoF localisation, so we implement our own using these libraries.

4.6.1 Active Search

In this chapter, we use the open-source implementation of the Active Search method from Sattler et al. (2012a). In this work, a bi-directional feature matching method is used. First, a descriptor vocabulary is used to quantise the descriptor space and words are assigned to each point in the model, as well as each feature in the query image. For each feature $\mathbf{f}$ in the query image, the 3-D points which share a node in the vocabulary tree are searched for matches using the typical ratio test from Lowe (2004), resulting in an initial match to a point $\mathbf{P}$.

Then, the 3-D points in the neighbourhood of this match are prioritised and matched to the features in the inverse direction using a coarser vocabulary. The point of this bi-directional matching is to exploit the assumption that points in the same 3-D region are likely to share similar visibility.

4.6.2 Expert Sample Consensus

The approach taken in Expert Sample Consensus (ESAC) by Brachmann and Rother (2019) is to train a convolutional neural network to learn scene co-ordinates for a given input image using both scene co-ordinate images, and 6-DoF pose as ground truth. The localisation component uses RANSAC to sample the output scene co-ordinate images, which naturally encode 2D-to-3D correspondences for pose estimation.

The method was first introduced in Brachmann et al. (2017), where they presented a modification to RANSAC which allows the entire pipeline to be differentiable, allowing for gradient-descent end-to-end learning. Their method was later improved in Brachmann and Rother (2018), and their final system ESAC (Brachmann and Rother, 2019) improves scalability by clustering the datasets and first training a scene classifier, followed by an ensemble of expert networks that are able to operate on the smaller scene clusters.

Pre-processing The ESAC localisation approach requires more data to train the CNN, in addition to the pose information from COLMAP. This method utilises ground truth scene co-ordinate images for its own training process. Scene co-ordinate images are like depth maps, except instead of encoding a depth value (i.e. distance from camera to scene) into each pixel, the full 3-D scene co-ordinate is stored resulting in a $3 \times H \times W$ tensor. It is possible to attain a dense representation of this data from a sparse model through dense MVS reconstruction methods (Schönberger et al., 2016); however, Brachmann and Rother (2018) suggest that a sparse representation is sufficient and in fact completely optional, as the entire pipeline is capable of learning the scene structure.

After processing the datasets with COLMAP to acquire a sparse reconstruction, the point cloud is then projected into a small ($H = 60$ and $W = 80$, as in Brachmann and Rother, 2019) representation of the training images using the known pose from the reconstruction. For each pixel, we encode the 3-D co-ordinates of the nearest point that projected to that pixel using a z-buffer. We exclude points behind the camera and leave zeros (white) for empty pixels. Examples of the resulting scene co-ordinate images are shown in Figure 4.9.

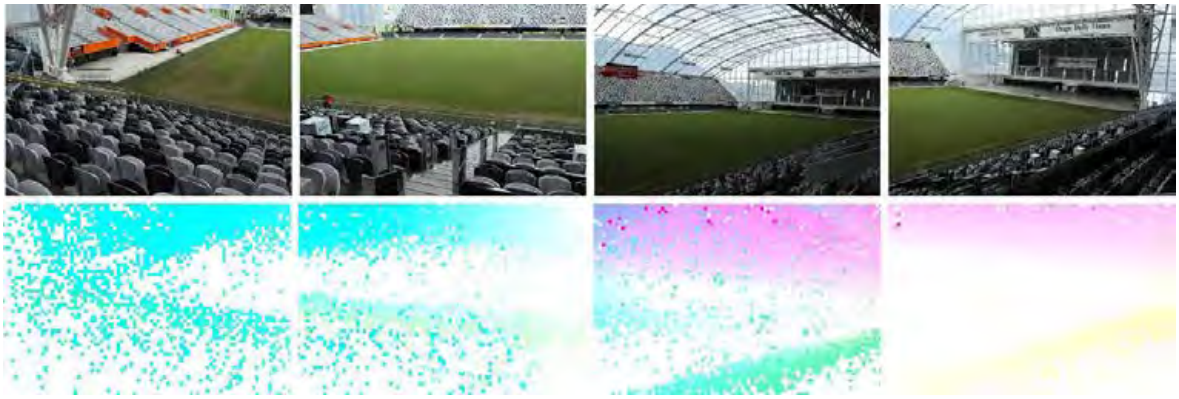

Figure 4.9 Top row: sample images used to train ESAC on the FBS dataset. Bottom row: their corresponding scene co-ordinate images used for training. Each pixel encodes a 3-D point as a floating point RGB value, where the R, G, and B channels represent the X, Y, and Z axes, respectively (scale adjusted for illustration). The values for each pixel are acquired by projection of the points in the sparse model produced by COLMAP.

4.6.3 Naive Bag-of-Words

In this section, we outline our own simple method for localisation based on BoW techniques. Our system is designed to operate on the output of a typical COLMAP reconstruction and represents the expected performance from localisation with a basic implementation. Our system can be used in two stages, pre-processing, and localisation.

We also require a vocabulary file containing representative ORB descriptors. In our implementation, we make use of the ORB-SLAM vocabulary. The BoW approach allows for quick matching between images by using an inverted file that contains both image and keypoint indices for each word. Here, we are interested in the feasibility of fast image retrieval using ORB, while maintaining robust matching via SIFT.

Pre-processing This stage only needs to be completed once per COLMAP model. The objective of this phase is to create an inverted index file, which stores for each word in the vocabulary, a list of image identifiers corresponding to the training images that contain that word. The purpose of this file is to act as a database for the image retrieval, and only needs to be computed once per SfM model.

We first detect 2000 ORB features for each training image. Then map the descriptors to words in the vocabulary using the optimised transformation implementation of fbow (Muñoz-Salinas and Medina-Carnicer, 2020). Finally, the inverted index is updated by appending the image identifier to the corresponding list in the inverted file for all

transformed words in the image.

Localisation To localise a query image, we first detect and map ORB features to the vocabulary as before. Then, for each word in the query image, we parse the list of training images via the inverted file and accumulate votes for each training image that contains that descriptor. The image with the most votes is accepted as the closest match, from which we begin establishing matches to compute a 6-DoF pose.

To achieve this, we then detect SIFT features in the query image, as they tend to show better matching rates than ORB (Karami et al., 2017). Then we apply a FLANN-based matching method (Muja and Lowe, 2009) to find the two nearest neighbours for each potential match, and discard unreliable matches using the ratio test of Lowe (2004). Finally, we apply an iterative solution to the Perspective-n-Point (PnP) problem based on Hartley and Zisserman (2003) within a RANSAC scheme (Fischler and Bolles, 1981) to compute the 6-DoF position and orientation of the query image.

4.6.4 Baseline: COLMAP Registration

As COLMAP is a modern and well-established state-of-the-art system, we use it as a baseline for comparing the accuracy of the different localisation approaches. Most incremental SfM systems include functionality for registering images to the existing model. During the reconstruction, Schönberger et al. (2016) first register images to the existing reconstruction, followed by triangulation of new points, then local optimisation via bundle adjustment. In our experiments, we use the initial image registration stage of this pipeline with vocabulary tree matching (Nistér and Stewénius, 2006), as a baseline for comparing the other localisation approaches.

4.7 Results

We tested all methods in their ability to localise the images of the testing set against the SfM models. In this section, we compare the computation time of both training (one-off computation), and the localisation itself, as well as accuracy metrics when compared to the COLMAP baseline method. We also provide and compare qualitative results from renderings of the sparse point clouds overlaid with the input query images.

Table 4.3 Computation times from various localisation approaches on both stadium datasets.

	Dataset = MCG, **Test Set Size** = 10		**Dataset** = FBS, **Test Set Size** = 225	
Method	Training	Localisation	Training	Localisation
Active Search	2.77 s	158 ms	116 s	221 ms
ESAC (4 experts)	101 h	88 ms	85 h	75 ms
ESAC (1 expert)	34 h	92 ms	22 h	145 ms
BoW	1.27 s	603 ms	30.2 s	1559 ms
COLMAP register	—	420 ms	—	468 ms

Computation The computation times for each of the methods are reported in Table 4.3. Localisation time refers to the mean time to localise over all query images. Training time refers to the time taken to pre-process the model or images for each localisation method. Computation times were measured on a PC with an Intel Core i9-9900KF CPU at 3.6 GHz, 32 GiB memory, and a GeForce RTX 2080 Ti video card.

In the case of ESAC, the training time includes initialising the gating network (classifier), initialising and refining 4 experts, and the end-to-end training stage. For Active Search, this includes the pre-processing stages of parsing the SfM data and computing descriptor assignments with the vocabulary. For our BoW method, this includes all steps detailed under pre-processing in Section 4.6.3, namely the ORB detection, fbow transformation, and creation of the inverted index.

We found that the ESAC approach has the longest training time, which is to be expected from a deep learning approach. However, training time is not particularly critical, and 101 hours is not prohibitively long for a one-off computation. We thus deem all methods to be viable for localisation with regards to training time. ESAC (Brachmann and Rother, 2019) reported the fastest localisation time with an average of 88 and 75 ms on the MCG and FBS datasets, respectively.

In Table 4.3, we see that the BoW approach had the fastest training time, likely due to the optimisations of fbow, combined with more compact ORB descriptors. However, the localisation time was the slowest, due to the overhead introduced by requiring both ORB and SIFT computation for the query images. For this reason, we conclude that the performance potential from using ORB over SIFT is beneficial in the indexing and pre-processing stages, but attempting to leverage the matching robustness of SIFT in conjunction with the simpler computation of ORB can be detrimental to performance.

Figure 4.10 Examples of successful localisation with our BoW approach on the MCG dataset. We report results with a geometric error over 10 pixels as false positives (outlined in red), with true positives outlined in blue. We found one false positive with a geometric error of 18.07 pixels.

This could suggest that using ORB features for both tasks might produce better results, which we suggest for future work.

We find that all methods produce acceptable localisation times, with the slowest being our BoW approach, which is not yet optimised for speed on the localisation side. While our method localised the slowest, realistically the localisation process would only need to happen occasionally in an overall AR system. For example, localising with only the initial frames or keyframes of a SLAM system. These methods therefore do not need to perform in real-time.

Error Metrics The localisation results showed potential for most of the methods, especially with ESAC, and Active Search. Sample results from the MCG dataset using the simple BoW method are shown in Figure 4.10, and ESAC results from the FBS dataset are shown in Figure 4.11.

Out of the 10 MCG test cases, 4 were reported successful by our BoW approach, and are shown in Figure 4.10. We note, however, that some cases are reported successful but exhibit visible error when overlaying the points on the image. In our results, we use '# Reported' to refer to the number of images reported to be successfully registered by the method itself.

Due to the potential for false positives, we take the localisation results from COLMAP

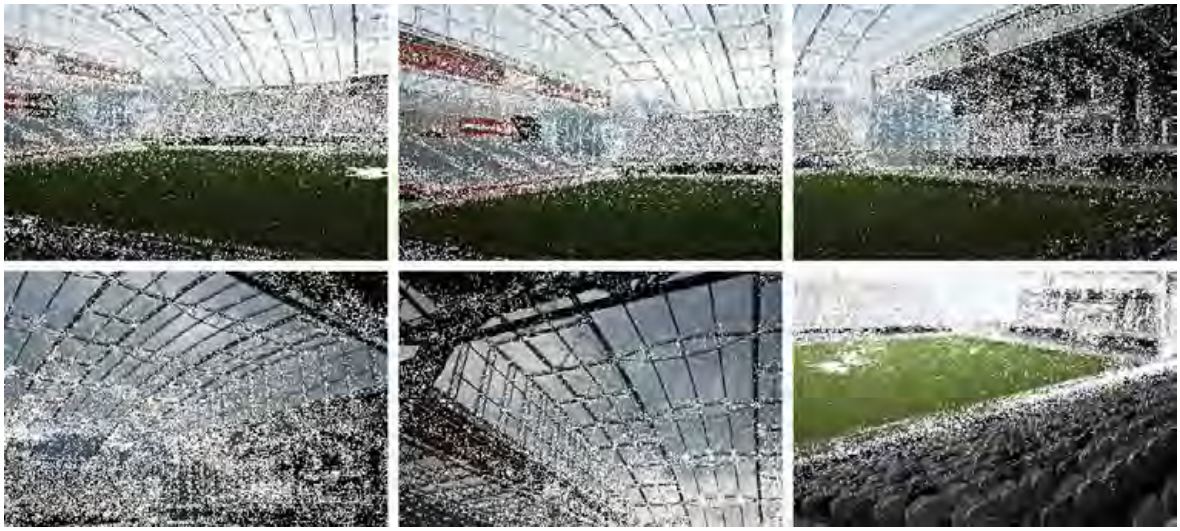

Figure 4.11 Examples of accurate localisation results from ESAC on the FBS dataset. We render the sparse point cloud from COLMAP over the input images using the pose estimate from ESAC and known prior intrinsic calibration.

as a baseline, $\hat{\mathsf{P}}$, and compare with the reported successful pose, P, using a geometric error over all co-visible points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ following the equation:

$$\sum_{i=1}^{n} \frac{|| \mathsf{P}\mathbf{x}_i - \hat{\mathsf{P}}\mathbf{x}_i ||}{n}, \tag{4.3}$$

where

$$\mathsf{P} = \mathsf{K}\,[\,\mathsf{R} \mid \mathbf{t}\,]. \tag{4.4}$$

We then flag any result with a geometric error of less than 10 pixels as a successful registration (# Actual). To get an idea of how accurate the successful registrations are with each method, we finally compute the mean (Err. M) and standard deviation (Err. SD) of the geometric error over all the true positive cases. All methods were tested with images of the same resolution (480 pixels in shortest dimension, as required by ESAC). The 10 pixel threshold corresponds to approximately 1.5% of the image width. To avoid making comparisons between potentially erroneous poses from the COLMAP baseline, each baseline registration was visually checked for quality. Ideally though, it would be preferable to have higher quality baseline or ground-truth poses for the test set, and this is something we suggest is worth investigating for future work.

Table 4.4 Accuracy and success rate of localisation approaches on the MCG dataset.

Dataset = MCG, **Test Set Size** = 10

	ESAC (4 experts)	Active Search (fixed K)	BoW	COLMAP *(baseline)*
# Reported	10	0	4	7
# Actual	6	0	3	7
Err. M (px)	1.67	—	6.99	—
Err. SD (px)	0.86	—	3.04	—

Table 4.5 Accuracy and success rate of localisation approaches on the FBS dataset.

Dataset = FBS, **Test Set Size** = 225

	ESAC (4 experts)	Active Search (fixed K)	BoW	COLMAP *(baseline)*
# Reported	225	215	161	221
# Actual	209	203	119	221
Err. M (px)	1.91	3.27	3.43	—
Err. SD (px)	1.68	2.00	1.81	—

The results for all methods on the MCG and FBS datasets are shown in Tables 4.4, and 4.5, respectively.

Discussion The Active Search method of Sattler et al. (2012a) produced robust registration with the FBS dataset, with higher success rate compared to our BoW results, and with faster localisation time. However, Active Search failed completely on the MCG dataset likely due to the limited number of points in the sparse model.

In our initial testing, many results from Active Search were reported as "successfully registered" while the point renderings were noticeably misaligned. This could be because the provided Active Search implementation computes both intrinsic and extrinsic camera parameters, and often incorrectly estimates large skew values in the intrinsic matrix.

For this reason, we modified the original implementation of Active Search to use the same PnP solver as our BoW implementation (Hartley and Zisserman, 2003; Fischler and Bolles, 1981). Using this method, we supply a fixed camera matrix and estimate

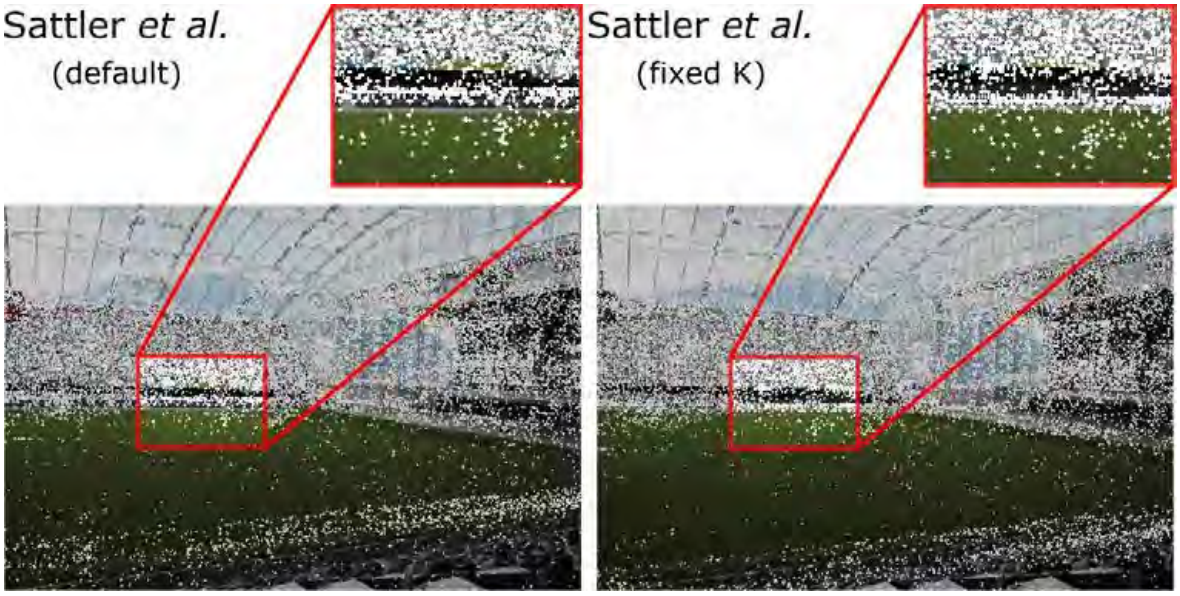

Figure 4.12 A comparison of the accuracy of state-of-the-art Active Search in Sattler et al. (2012a). The default implementation estimates incorrect skew values in the camera matrix. When the camera matrix is fixed, the points register more accurately.

the pose directly, achieving more consistent results. A comparison between the original results to the updated method is shown in Figure 4.12.

To compare the accuracy between the three methods we make three comparisons, so we must account for this. We start with a typical threshold of $p = .05$ and apply Bonferroni correction to get an adjusted threshold of $p = .01\overline{6}$. As each method has the potential to succeed or fail on each sample (producing no accurate output), performing a paired test is impractical. For this reason, we use two-tailed unpaired t-tests here. We compared three groups of accuracy results: ESAC, Active Search, and BoW. ESAC ($M = 1.91$, $SD = 1.68$) showed significantly lower geometric error when compared to Active Search ($M = 3.27$, $SD = 2.00$) with $p < .0001$, and also when compared to BoW ($M = 3.43$, $SD = 1.81$) with $p < .0001$, where both of these p-values fall below our adjusted threshold. However, comparing Active Search to BoW showed no significant difference, $p = .474$.

The smaller error output by ESAC could be due to the fact that our COLMAP poses are actually a baseline and can't be regarded as ground-truth. As Active Search and BoW compute poses independently of the COLMAP training poses, their error could be attributed to the noise in the point cloud, whereas ESAC is trained specifically to replicate the training poses, and is able to refine the point estimates that were provided

in the form of the scene co-ordinate images (Figure 4.9). Some examples of successful localisations from the ESAC approach are shown in Figure 4.11.

4.7.1 Dynamic Localisation

As we have seen in the previous section, the ESAC and Active Search localisation approaches both appear promising for localising sport spectators. However, in our FBS dataset, we only evaluated with images of an empty stadium (for both reconstruction, and localisation). The reprojection errors we previously saw from ESAC were low, which suggest an accurate alignment to the poses from the COLMAP baseline, from which that method was trained.

In this section, we take a qualitative look at how these results would be visualised in a realistic scenario, where the reconstruction was made from images of an empty stadium, and localisation is taken under different dynamic conditions, such as changes in lighting, and presence of other spectators. For these tests, we use different images from what were previously used, captured during a live rugby game.

As depicted in Figure 4.13, we can see that both the Active Search and ESAC approaches have the potential to produce highly accurate localisation. As we do not have ground-truth poses for these additional images, we are unable to perform more evaluation on the accuracy. Though, upon visual inspection of the projected points in Figure 4.13, both methods produce very similar results with good overall alignment of the sparse model to the image. Though these results appear promising, a more thorough evaluation of the effects of the dynamic elements would be beneficial for future work.

4.8 Conclusion

To summarise the findings of this chapter, we have shown how SLAM systems such as ORB-SLAM (Mur-Artal et al., 2015) are not sufficient on their own for localisation in the ARSpectator application, and require significant translational movement to produce a reasonable model. We first looked into applying SoftPOSIT (David et al., 2003) to solve this problem using line-based SfM models. However, we were unable to achieve good localisation performance with complex SfM models such as our sport stadium, with a success rate of 12% with an inaccurate pose estimate, and 20% when given an

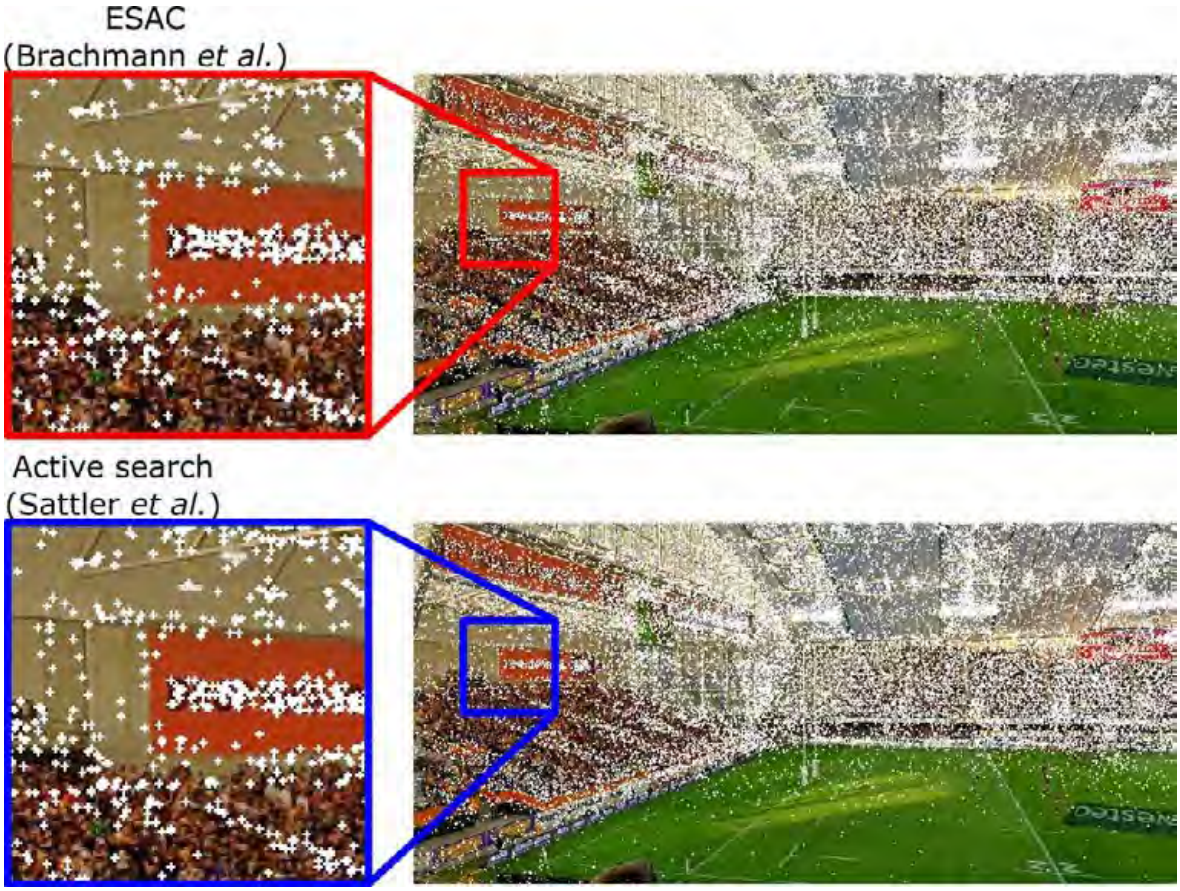

Figure 4.13 Qualitative results of localisation in a dynamic stadium with spectators, and different lighting. As we do not have ground-truth, we cannot report accuracy numbers. However, both methods appeared to produce very similar results when aligning the sparse point cloud with the image using the pose outputs.

accurate pose estimate. The results on smaller models were better, which suggest it may be applicable to other contexts beyond the scope of this thesis.

With the SoftPOSIT results as a starting point, we then looked into more robust localisation approaches such as Active Search (Sattler et al., 2012a), ESAC (Brachmann and Rother, 2019), and our own implementation of a BoW-based localiser. In the second part of this chapter, we showed that these methods may require significant pre-processing or training to be used, but once trained can localise with acceptable processing time.

In terms of localisation accuracy, we found that all methods produced acceptable results for successful localisations, but with many false positives reported by the naive BoW approach. Our final recommendations would be to use the ESAC approach for application scenarios where GPU acceleration is practical, as this method produced

good localisation robustness and accuracy, while also performing reasonably well with the small MCG dataset. A system where localisation is performed on a separate server, with results being transmitted to the mobile devices would be ideal. For cases where localisation must occur on the mobile device, either the Active Search approach or COLMAP approach would be suitable, as their processing times were reasonably low and did not require GPU acceleration.

For future work in this area, we aim to create a complete localisation and tracking system. Such a system would support multiple users localising within a single existing model, which would result in a framework that supports AR content registered to key parts of the stadium in meaningful ways. Another pathway for future work is to compare to methods which localise with respect to the lines of the pitch as in Skinner and Zollmann (2019). This could be combined with field segmentation to assist with the line detection. However, as field lines are not always available, this type of localisation is less widely applicable than the SfM model approaches investigated here.

Chapter 5

Spherical Tracking for Augmented Reality

In Chapter 4, we have explored the possibility of localising in a large dynamic environment such as sports stadia using a variety of methods ranging from geometry-based with SoftPOSIT, to appearance-based with Bags-of-Words, active search, and deep learning. We have found that the appearance-based methods show promising results and discussed the possibility of integrating such a localisation method into a keyframe-based tracking system to create a complete system that can robustly localise and track in large environments.

We also observed that some SLAM-based tracking methods are not able to track robustly in these large environments, due to the small translational movements that stationary users make. Thus, in this chapter, we investigate alternative tracking methods that specifically target this stationary use-case. We explore the application of spherically constrained SfM concepts (Ventura, 2016) into a real-time tracking method. Here we build a tracking system from scratch, and discuss the implementation challenges, and advantages that such a constrained system can offer. The work in this chapter has been published in Baker et al. (2020b).

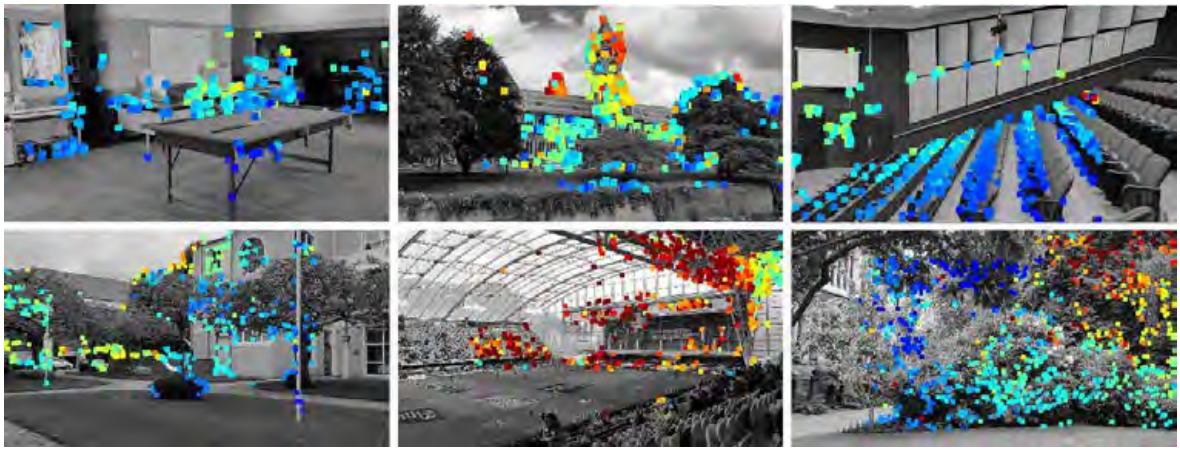

Figure 5.1 Sample depth output from the SPLAT tracking system. The system triangulates 3-D points, and projects them back into the image coloured by their computed depth values. Blue represents close points, and red represents far points.

5.1 Introduction

Driven by the availability of programming frameworks such as Vuforia, ARKit, and ARCore, we have seen a wider adoption of handheld and mobile Augmented Reality (AR) where mobile devices, typically mobile phones, provide an AR interface to the user. Most applications focus on small indoor environments where digital objects are integrated into our physical environment. Examples include digital enhancements of physical books or magazines, augmentation of board games, Augmented Reality games, or previewing digital furniture augmented within our physical environments. While these environments can be handled well by most AR frameworks, large and open environments still pose challenges and are consequently less often targeted.

Key among these challenges is the precise tracking of a mobile device's location and orientation, which is a necessity for AR. When in outdoor environments or large open areas (e.g. stadiums, large spaces), users often have a fixed position, they are standing in place or even seated, observing the scene or event from a single location. Being stationary in one position means that there is insufficient parallax for conventional simultaneous localisation and mapping (SLAM) approaches. One approach is to use panoramic trackers (DiVerdi et al., 2008; Wagner et al., 2010) to address these issues. However, the motion of the user's device is not purely rotational either, which is problematic for panoramic trackers, which assume a perfect rotation centred around the camera.

We experienced these issues first-hand when investigating using an AR interface in a

stadium environment (Zollmann et al., 2019) with the goal of providing live sports spectators rich real-time visualisations and overlays as they are used in sports broadcasting, but this time using an AR interface on a mobile device. However, being in one position is not limited to the scenario where spectators are watching a game or event within a stadium environment. An example where similar usage patterns have been observed are many outdoor AR applications utilising mobile phones, in particular AR browsers on handheld devices, where users have the tendency to stand in one position and then rotate their device to explore the surroundings (Langlotz et al., 2013, 2014; Kurz et al., 2013).

In this chapter, we address the problem of reliably tracking a camera in the situation where users are stationary and performing *mostly* (but not purely) rotational movements. We observe that for a static user, their device is generally held at arm's length, so moves on a roughly spherical surface. However, instead of assuming a pure rotation around the camera centre (rotation-only tracking) or unconstrained movement with a large enough baseline (SLAM), we propose Spherical Localisation and Tracking (SPLAT), a SLAM system based on spherical structure-from-motion (SfM) via 2-D feature tracking (Ventura, 2016), paired with a new spherical 3-D absolute pose solution. Our system is implemented as a keyframe-based tracking and mapping pipeline using ORB features, similar to ORB-SLAM (Mur-Artal and Tardós, 2017a).

We show that this approach is more robust to restricted parallax through experiments on synthetic data (where ground-truth motion is available). We also demonstrate its effectiveness on real video sequences captured in several large spaces, including a stadium environment during a live sports event.

Our tracker is based on the work of Ventura (2016) and extended in the following ways:

- We integrate the SfM methods into a complete real-time simultaneous localisation and tracking system suitable for use in Augmented Reality.
- We detail and implement a method for computing *absolute pose* with a spherical constraint based on a MATLAB solution by Jonathan Ventura.
- We evaluate the performance of our spherical tracking system in various synthetic and real environments.

Our implementation is built from scratch, and loosely based on ORB-SLAM (Mur-Artal and Tardós, 2017a) with the following contributions:

- We apply a spherical constraint to the initialisation, tracking, and bundle adjustment.
- We present a unique *keyframe sphere* which is used to adaptively trigger SLAM initialisation and keyframe selection, while replacing the need for a keyframe culling mechanism.

The remainder of this chapter is organised as follows: Section 5.2 reviews previous approaches to localisation and tracking for AR, and examines the limitations of these approaches in our context. Our new method is explained in detail in Section 5.3, with an explanation of the Spherical P2P solution in Section 5.4. We evaluate our SPLAT system in comparison to ORB-SLAM and homography tracking in Section 5.5 and Section 5.6. Finally, in Section 5.7, we conclude by summarising our findings.

5.2 Background

The problem of determining a camera's pose within an environment (localisation) and updating it over time (tracking) has been studied for a long time. In the context of AR, the dominant approaches are SLAM and panoramic tracking. We review these two approaches, highlighting their limitations for the task of localisation and tracking in large environments with limited (but not negligible) translational movement.

We focus primarily on visual localisation and mapping with RGB cameras. Depth cameras (RGB-D sensors) are sometimes available on mobile devices, but they usually have limited range, making them more suited to indoor scenarios (Young et al., 2020). Inertial sensors are likely to be used in practical applications but are often combined with visual tracking and localisation to overcome sensor drift.

5.2.1 Simultaneous Localisation and Mapping

SLAM is a tracking approach originating from robotics. A robot typically moves through its working environment and while moving, the robot builds up a spatial map of the environment which it uses for tracking and localisation (Dissanayake et al., 2001). While the original approaches often used direct range sensors (Dissanayake et al., 2001) or multiple cameras (Davison and Murray, 2002), later research focusing on AR scenarios showed variations of SLAM approaches that work in real-time using a single camera only, (e.g. MonoSLAM, Davison et al., 2007, or PTAM, Klein

and Murray, 2007). Since then, many approaches for monocular SLAM systems have been introduced that differ mainly in what kind of features are taken or how they are matched. Examples include the use of invariant image features (e.g. ORB-SLAM, Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a), dense image registration (e.g. LSD-SLAM, Engel et al., 2014), objects as landmarks (e.g. SLAM++, Salas-Moreno et al., 2013), and the use of deep learning to estimate pose (Kendall et al., 2015) or compact scene representations (Bloesch et al., 2018).

However, despite the differences in feature handling and matching (or the lack thereof, Engel et al., 2014), most monocular visual SLAM systems rely on the same basic concept of requiring a reasonable parallax at some stages (e.g. initialisation) to compute correct depth information. In our work we use ORB-SLAM, one of the most widely used SLAM systems, to represent a general SLAM system. ORB-SLAM is a keyframe-based SLAM system, meaning that a sparse set of frames are chosen for map update, while the others are used for pose tracking only. It is capable of tracking moving cameras in a variety of environments and mapping the 3-D structure. The method supports loop closure when returning to previously mapped areas, bundle adjustment of the computed map, and relocalisation to recover from tracking failure (Mur-Artal et al., 2015; Mur-Artal and Tardós, 2017a).

While all these SLAM systems have shown potential for tracking in various types of indoor and outdoor environments, they are unable to perform reliably in very large spaces when the user's camera motion is small. This is a significant problem for AR in large open spaces, such as a stadium, as their environment will be much larger than the typical office or home that these systems perform well in. Furthermore, live event spectators, such as sport spectators, will often be constrained to a fixed position or seat, moving only small distances throughout an event, which causes initialisation problems due to the small baselines.

5.2.2 Panoramic Tracking

Tracking of purely rotational movements has also been considered previously within the research community. While one can use integrated hardware sensors such as compass and gyroscopes, these are in practice not reliable enough to deliver precise information for tracking the rotation of a mobile device. Works such as Envisor (DiVerdi et al., 2008) or Panoramic Mapping and Tracking (Wagner et al., 2010; Müller et al., 2016; Young et al., 2019) provide different algorithms for tracking by building a panoramic

map. The former optimises the algorithm for using the GPU to achieve real-time performance, while the latter optimises the handling of the panoramic map and the feature matching to achieve real-time performance on a mobile device. Common to all these approaches is that they require purely rotational movements, and a violation of those (e.g. when not rotating around the camera centre) will introduce errors and ultimately tracking failure.

Panoramic trackers assume that the camera's motion is purely rotational (or that the scene is planar). Under these circumstances, the images of the scene are related by homographies, and so can be mapped to a sphere (or plane) (Szeliski, 2006). This approach has two main drawbacks for our application. The first is that while the camera movement is largely rotational, there is a translational element. This is introduced by the camera position being usually offset, and the fact the the centre of rotation is not the phones' camera centre when held an arm's length from the user. This translation element is restricted (i.e. to arm's reach), but may be large enough that the assumption of (nearly) pure rotational motion is violated. The second is that the visualisation of the event may rely on a 3-D interpretation of the scene, which panoramic trackers do not provide. Rather than forming a 3-D model, panoramic trackers represent the scene as directional rays from the viewer's location.

Hybrid solutions also exist that combine panorama tracking and SLAM. Pirchheim et al. (2013) describe a method to track through periods of rotational motion in a general SLAM system. However, they assume that there is sufficient translational movement at some point in the sequence to create the map. Gauglitz et al. (2012) introduce a method for tracking through both general and rotational motions but do not build and localise from a global map.

5.2.3 Limitations for Users in Large Open Environments

Augmented Reality applications often assume that the user is in a relatively small enclosed space, such as an office or workshop (Klein and Murray, 2007), or moves freely through a larger environment (Reitmayr and Drummond, 2006). In both of these cases, the motion of the user is a significant fraction of the size of the scene, providing a good baseline for triangulation-based reconstruction. Alternatively, the user might view distant objects from a single view point (Langlotz et al., 2013). In this case, the motion of the user is insignificant compared to the size of the scene and panoramic tracking is viable. We are concerned with the intermediate case – where the

user's motion is small, but not negligible, compared to the distance to the scene.

This case arises in our application of Augmented Reality for sports spectators, where a large stadium environment (on the order of 100 m) is combined with limited motion of the user's device (on the order of 1 m). Treating the motion as pure rotation leads to an angular error on the order of 0.5°, which corresponds to tens of pixels of error on a typical HD display. A similar situation arises when a seated user with a head-mounted display (translations on the order of 0.1 m) views a large indoor scene (on the order of 10 m).

The case of a seated or otherwise stationary user does, however, offer some advantages. The key advantage for our purposes is that the user will be likely returning to the same position multiple times and will quickly be able to scan and observe their entire environment.

This means that some of the key advantages of typical SLAM systems, such as loop closure, and regular addition of new keyframes, become more constrained. A SLAM system that is constrained to spectator type motion would need to keep track of fewer keyframes and would rarely need to perform loop closure procedures.

Apart from the focus on our specific use-case of live spectators in a stadium, we believe the applicability of our approach goes further, including many AR applications that use handheld devices in outdoor environments. As Grubert et al. (2011) showed, when using AR browsers or related applications, stationary position and rotation-dominant movement is the dominant movement pattern.

5.3 Method

As we have seen, existing approaches use specific constraints, parallax, or pure rotation that are often violated in the real world. In this section, we present a new approach of spherical localisation and tracking that has, so far, not been widely considered. By doing so, we address the problems that arise when using state-of-the-art SLAM for mostly pure rotational movements. Our system represents a SLAM approach based on spherical SfM (Ventura, 2016).

Contrary to existing solutions, our approach does not assume a large parallax, fixed position, or pure rotational movement. Instead, our approach uses a spherical constraint introduced by Ventura (2016). The spherical constraint relaxes the fixed position of a camera and instead allows a position on the surface of a sphere, and a viewing di-

Figure 5.2 Different spherical movements captured by a mobile phone user in a stadium environment (colour coded for different paths). Left: front view. Right: top view.

rection parallel to the sphere normal (Figure 5.2 shows different near-spherical user movements).

This has several advantages that we want to explore. Firstly, as already shown by Ventura, the spherical constraint allows us to use a minimal representation of the camera pose based on three rotation parameters which only requires three point correspondences to compute the relative pose between two cameras in contrast to using five correspondences in the case of unconstrained movements (Nistér, 2004). More importantly, we argue that this spherical constraint is a better approximation of the user's movement when stationary in a large environment such as in our sports stadium scenario.

In his work, Ventura only investigated the spherical constraint for offline 3-D reconstruction but did not implement a full tracking system or a real-time spherical SLAM approach. In the following, we present our implementation of a SLAM-based system using this spherical constraint, while later evaluating its performance using synthetic and realistic data, and comparing it against state-of-the-art SLAM. As part of our solution, we also propose a method for computing the absolute pose with two 3D-to-2D point correspondences with a spherical constraint.

Our spherical SLAM system takes a unique approach to the selection and storage of keyframes (Section 5.3.1) and, as with many SLAM systems, has three distinct stages: Initialisation (Section 5.3.2), Tracking (Section 5.3.3), and Mapping (Section 5.3.4). Figure 5.3 depicts an overview of the whole system.

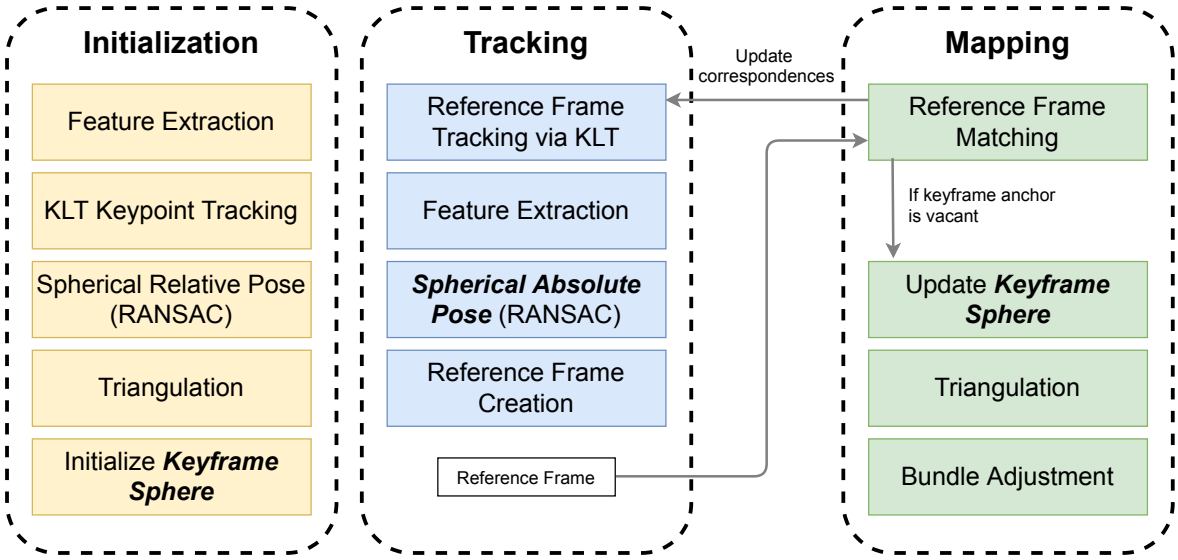

Figure 5.3 High-level overview of our tracking and mapping system. The first component handles initialising a 3-D map and the keyframe sphere, before handing off to the tracking component. The tracking component tracks keypoints from the last reference frame and creates new reference frames when required by the keyframe sphere. The mapping thread updates the 3D-to-2D matches of the latest reference frame asynchronously, triangulates new points, and updates the keyframe sphere.

5.3.1 The Keyframe Sphere

We propose a new method for subdividing the space of possible camera poses which is tailored specifically to a spherically constrained keyframe SLAM system (visualised in Figure 5.4). We compute N points approximately evenly distributed across the surface of a sphere using the method of Saff and Kuijlaars (1997). These points act as anchors for the keyframe selection, with the goal of achieving an even distribution of keyframes. This kind of careful keyframe selection can help a SLAM system in scalability and performance, with N acting as an upper limit on the possible number of keyframes.

Through experimentation, we found good results with $N = 1000$ anchor points. We also set a threshold requiring that a new keyframe's camera centre must be within some small distance of an anchor point. We set this distance threshold to be equal to 25% of the distance between two neighbouring anchors. Setting larger values here can cause issues, such as two similar keyframes being assigned to different neighbouring anchor points.

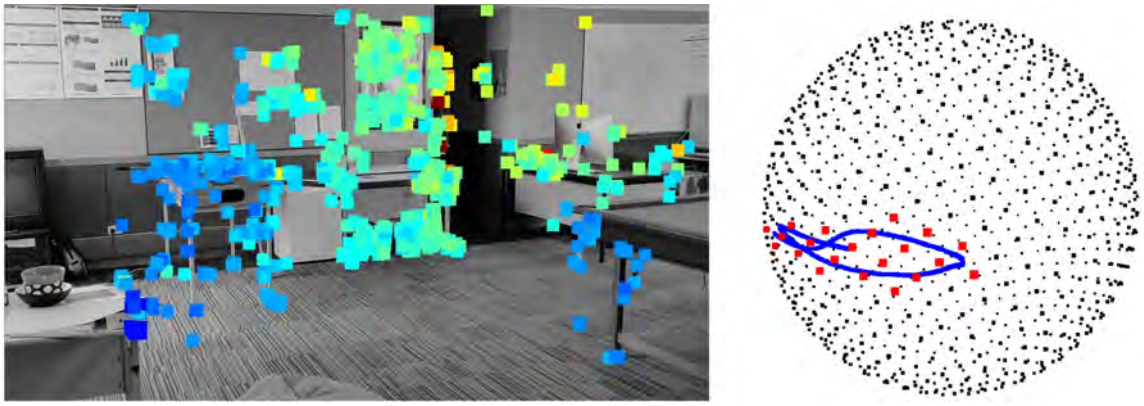

Figure 5.4 Example output of our system. Left: tracked features overlaid with the current input frame. Right: Tracked poses (blue) alongside, occupied (red), and unoccupied (black) keyframe anchor points.

5.3.2 Initialisation

We perform an automatic initialisation step before we can start with the tracking. For our application scenario, this would be quite feasible since we can direct users to scan part of the area with their mobile devices. Our system automatically initialises a map and begins tracking when enough spherical motion has occurred, using the process described below. The overall initialisation step is based on spherical SfM method introduced by Ventura (2016), but with a focus on faster computation by only triangulating from two frames.

Feature Extraction and Tracking Our feature extraction method uses ORB features (Rublee et al., 2011). We extract 2000 keypoints and descriptors in the initial frame. To match keypoints between successive frames, we find matches using a pyramidal KLT feature tracker (Bouguet et al., 2001). At each frame, we use these 2-D matches to make an estimate of the current pose relative to the initial frame with a spherical constraint (Ventura, 2016).

Relative Pose Estimation To determine the relative pose for initialisation, we use the spherically constrained relative pose estimation introduced in Ventura (2016). We assume that the camera moves on a mostly circular path with a constant radius of 1 unit from the origin. Also, we assume that the viewing direction of the camera is in alignment with the normal of the sphere that is given by this radius. These assumptions allow us to simplify the pose estimation. The camera pose can then be described as:

Figure 5.5 3-D point cloud (blue) and spherical camera poses (red) from the indoor Lab sequence.

$[\,\mathsf{R}\,|\,\mathbf{t}\,]$ where $\mathbf{t} = [0\ 0\ -1]^\mathsf{T}$, and the camera centre $\mathbf{c} = -\mathsf{R}^\mathsf{T}\mathbf{t}$. We use the method of Ventura (2016) for both computing and decomposing an essential matrix to compute the spherically constrained relative pose (as introduced in Chapter 2). We use this in combination with Preemptive RANSAC to discard outlier tracks (Nistér, 2005), and to determine an estimate of the poses for the first two camera frames. Preemptive RANSAC is an adaptation of the original RANSAC formulation that aims to reduce the computational overhead of scoring potential hypotheses through preemption of smaller computational blocks.

Triangulating the Initial Map Using the relative pose estimate from the previous step, we determine if the angular motion is sufficient, as determined by our keyframe sphere structure. If the poses of the start and end frames fall within the distance thresholds of two different anchor points, we proceed with initialisation. We then extract features in the final initialisation frame and match them to the initial frame using the feature tracks. Finally, we use the matches and relative pose to perform triangulation to compute the 3-D points (Figure 5.5).

5.3.3 Tracking

After the map has been initialised with two keyframes, and the 3-D points from triangulation, we use this data as input for our tracking method. The tracking step can be described by the following components:

Feature Extraction and Tracking The extraction of ORB features is very similar to the extraction from the initialisation step. Again, we have an upper limit of 2000 keypoints and descriptors. We first use KLT feature tracking (Bouguet et al., 2001) to keep track of matches from the most recent reference frame, or keyframe. We then compute a convex hull (Sklansky, 1982) surrounding the KLT tracked keypoints, and extract a number of new ORB features outside this mask.

We enforce the 2000 keypoint limit which includes both the KLT tracks (and their already known descriptors), as well as the newly detected features. In our current implementation, we extract features in this way from each frame; however, it could be optimised to only compute new descriptors in reference frames, as matching descriptors to the existing 3-D points is only applied to reference frames (as detailed next).

3D-to-2D Feature Matching We next use the extracted ORB features to obtain more 3D-to-2D correspondences. We use the KLT tracks to maintain references to 3-D map points that were found in the last reference frame, which in many cases is enough to track. However, we must also find potential correspondences between the newly detected features, and points which have already been mapped (to avoid triangulating duplicate points).

In a separate thread, we perform simple bruteforce matching between the features in the reference frame (which represents the origin of the current KLT tracks), and the keyframes in the system. Our bruteforce matching essentially computes the normalized hamming distance between each binary descriptor, and filters incorrect matches using a distance threshold. When new matches are found, the correspondences are updated in the reference frame, which can be accessed asynchronously (via their corresponding feature tracks) by the current frame in the main thread.

Absolute Pose Estimation To estimate the pose for each frame, we use a new Perspective-2-Point (P2P) method with a spherical pose constraint within a preemptive

RANSAC scheme to determine the current pose, and an inlier set of matches (Nistér, 2005). Details of this P2P solution are covered in Section 5.4.

Reference Frame We use a single frame (with known pose from tracking) which we refer to as the *reference frame* to store the origin of the currently tracked features. When a frame has been successfully tracked, we decide whether it will become the new reference frame based on its distance to the keyframe anchors. When a new reference frame is selected, the mapping thread matches its keypoints to all neighbouring keyframes and merges the observations. This is equivalent in some respects to loop closure when loops and drift are small. If the closest keyframe anchor is empty, the reference frame will then become a keyframe as described in the following paragraph.

Keyframes The reference frame will become a keyframe if its anchor point in the keyframe sphere is unoccupied. In this case, new points are triangulated using the KLT tracks between the reference frame and the current frame, and bundle adjustment takes place. Since the tracking thread is acquiring many matches through feature tracks to this frame, the new map points are automatically assigned to the current frame as soon as they are ready. This allows new map points to be added asynchronously.

5.3.4 Mapping

Once a frame has been tracked successfully, it is sent to the mapping thread to use the newly visible feature points in the image to update the map. The mapping thread has three main stages:

Guided Matching When a new keyframe is created, the tracking thread follows features from the reference frame using KLT tracking as before. The mapping thread uses these results to guide the matching of keypoints between these two frames.

Updating the Map The previously computed matches are then triangulated in a similar manner to the initialisation phase. If more than 50 map points were triangulated from the matches, then the keyframe is considered successful, and is added to the keyframe sphere structure as described below. We empirically determined 50 to be the lower end of what can result in stable tracking, though larger thresholds may be used.

In some cases, points are triangulated behind the camera corresponding to incorrect initialisation.

Bundle Adjustment In the mapping thread, after triangulation, bundle adjustment takes place to optimise the 3-D point locations, and keyframe camera poses. To enforce a spherical constraint here, we set the parameters for the camera translation to be fixed in the optimisation (Ventura, 2016). Once this process is complete, we remove keyframe references to outlier 3-D points using the same reprojection threshold we set in our tracking thread. We then update the positions of the remaining points, and camera poses of the keyframes. We found that running a small number of iterations (one to two) each time a keyframe is added is a good way to keep map updates frequent. This also allows for faster performance than full bundle adjustment as used in Ventura (2016) at the cost of some accuracy. The use of fewer iterations has also been shown to be beneficial by Engels et al. (2006), where fast computation is needed. In our experiments, we use two iterations of bundle adjustment between map updates.

5.4 Spherical Perspective-2-Point Solution

A key component of our SLAM system is a method for computing *absolute pose* under the spherical constraint, which we integrate into our overall system based on a novel solution by Jonathan Ventura. This is a key component of a tracking system which allows for computing the pose of a camera with respect to a pre-computed 3-D map using correspondences between the observed image points, and the 3-D map point locations. This is widely known as a Perspective-n-Point (PnP) problem. PnP solvers are often used in SLAM systems to determine the pose of the camera with respect to the map. In the unconstrained case, solutions require at least three 3D-to-2D correspondences to determine the pose (Fischler and Bolles, 1981).

Previous work by Ventura (2016) defines spherical motion as having a fixed translation, causing the camera to remain on the surface of an imaginary sphere. This section covers our solution to this problem using two correspondences and the known translation vector $\mathbf{t}$.

The 3-D points $\mathbf{X}^w = \{\mathbf{X}_1^w, \ldots, \mathbf{X}_n^w\}$, and their corresponding observations $\mathbf{x}^c = \{\mathbf{x}_1^c, \ldots, \mathbf{x}_n^c\}$ are related by a known translation $\mathbf{t}$, unknown rotation R, and unknown

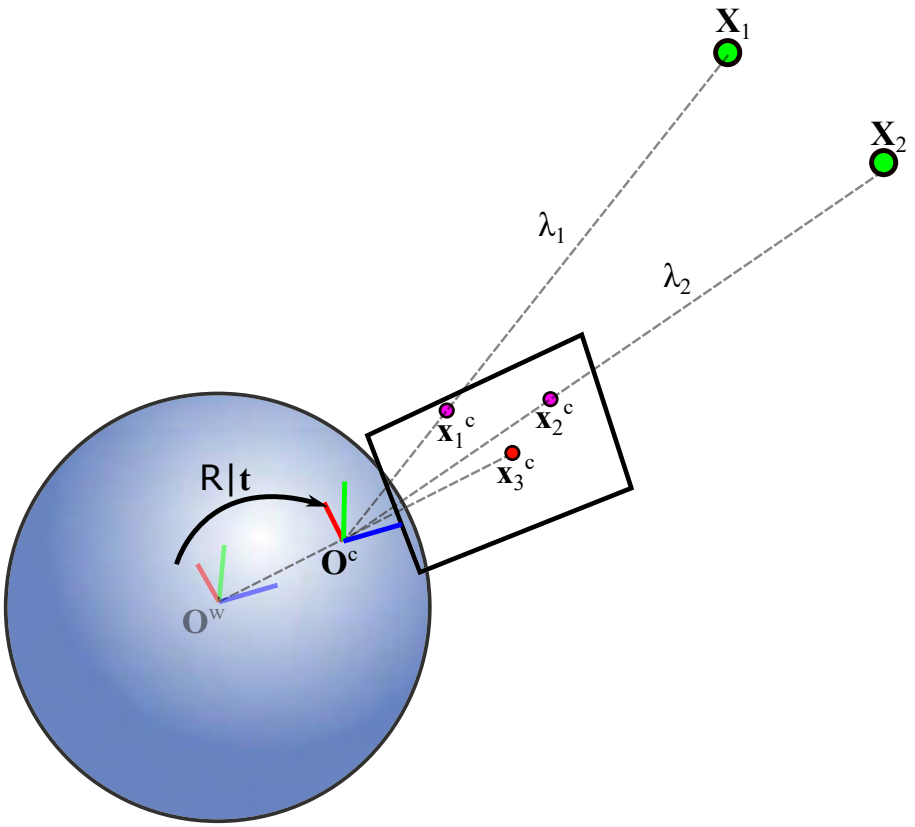

Figure 5.6 Visual representation of the Spherical Perspective-2-Point problem. Note, the third correspondence comes from the projection of the world origin $\mathbf{O}^w$ to the principle point of the camera $\mathbf{x}_3^c$, known through the intrinsic calibration.

scale factor λ_i:

$$\mathsf{R}\mathbf{X}_i + \mathbf{t} = \lambda_i \mathbf{x}_i \tag{5.1}$$

Since we assume that the camera motion is spherical, we can fix $\mathbf{t} = [0\ 0\ -1]^\mathsf{T}$, so our objective is to compute R. Our approach is to first compute the scale factors λ_i which can be used to unproject a homogeneous 2-D observation $\mathbf{x}_i^c$ into a 3-D point $\mathbf{X}_i^c$ in the camera co-ordinate system. Finally, we can find R by aligning the two 3-D point sets using Horn's method for absolute orientation (Horn, 1987).

We also know that the distances between each pair of 3-D points must be the same in both the world co-ordinate system and the unprojected (scaled) camera co-ordinate system. So, we can produce the following equations:

$$||\mathbf{X}_1^w - \mathbf{X}_3^w||^2 - ||\lambda_1 \mathbf{x}_1^c - \lambda_3 \mathbf{x}_3^c||^2 = 0 \tag{5.2}$$

$$||\mathbf{X}_2^w - \mathbf{X}_3^w||^2 - ||\lambda_2 \mathbf{x}_2^c - \lambda_3 \mathbf{x}_3^c||^2 = 0 \tag{5.3}$$

We can also assume that our camera is facing outwards, with the camera axis parallel to the normal of the sphere. This assumption can give us one correspondence for free – that the world origin $\mathbf{O}^w = [0,0,0]^\mathsf{T}$, located behind the camera, should always project to the centre of the image. So, let $\mathbf{X}_3^w = \mathbf{O}^w = [0,0,0]^\mathsf{T}$, $\mathbf{x}_3^c = [0,0,1]^\mathsf{T}$, and $\lambda_3 = -1$.

Equations 5.2 and 5.3 can be solved for λ_1 and λ_2, respectively, using the quadratic formula, resulting in four solutions. Finally, R is computed by applying Horn's method to the two 3-D point sets $\mathbf{X}^w$ and $\mathbf{X}^c$ (Horn, 1987).

To implement this into our overall SLAM system, we perform this computation on minimal samples (two correspondences, plus the spherical correspondence $\mathbf{x}_3^c \rightarrow \mathbf{O}^w$ in Figure 5.6) within a Preemptive RANSAC scheme (Nistér, 2005). As we require our system to be tracking with a minimum of 15 correspondences, we can use an additional (fourth) correspondence ($\mathbf{x}_4^c \rightarrow \mathbf{X}_4^w$) from our matching to determine which of the solutions is correct. For each of the four solutions, we use the full pose to find the projection of $\mathbf{X}_4^w$. We then accept the solution which has the smallest distance between the projected and the observed points.

While Equations 5.2 and 5.3 are correct when the spherical motion constraint is followed, error may be introduced when the user deviates from this assumption. This type of error is to be expected, as it is not practical for a user to follow the spherical constraint perfectly. However, in our testing on real-life data, our implementation was able to reach a good solution for R in most cases, with small deviations from the spherical path as shown later in Section 5.6.

5.5 Synthetic Experiments

Our initial evaluation focuses on synthetic experiments with controlled parameters in order to determine the theoretical limits of SPLAT, and the other tracking algorithms. Brückner et al. (2008) suggest a framework for comparing relative pose estimation algorithms with controlled 3-D point data generated in a cuboid. Our experiments are similar in that we generate a 3-D space of a known size and structure. However, we generate points in a 3-D sphere rather than a cuboid, to have a finer level of control over the space size, and the distance between the generated points and the simulated cameras. In this section, we compare our spherical tracking system to state-of-the-art

Figure 5.7 Example of our synthetic video setup. The world texture is projected onto a sphere with a variable radius (5 in this case), and a camera (white) moves in a circle (orange) within this sphere.

ORB-SLAM (Mur-Artal et al., 2015), which does general 6-DoF tracking, and also a simple implementation of a rotation-only 3-DoF tracker based on homographies.

5.5.1 Data Preparation

We take a high-resolution texture and map it to a sphere within a 3-D rendering software. To avoid unrealistic seams, the texture we use is a high resolution equirectangular photograph of the interior of a church [1] (Figure 5.7).

Generating Videos To simulate camera motion, we focus on the simple case of a camera moving in a circular arc with a fixed speed and radius. We run our experiment over 1000 frames, (0.36° of rotation per frame), and an arc radius of 1 unit.

To determine how the tracking algorithms perform in spaces of various sizes, we fix the circular arc radius of the camera, and its movement and internal calibration parameters.

[1] Original image captured by Jürgen Matern 2018, shared under the Creative Commons Attribution-Share Alike 4.0 International license.

We then vary the radius of the textured world sphere at regular intervals ranging from 2 to 50 units. At each interval, we encode a lossless 30 FPS video, resulting in 49 videos at a resolution of 512×512 pixels, which we use as input for the tracking systems.

Error Evaluation Since we are interested in determining the reliability of these systems, we look to evaluate a *tracking rate*, as a measure of tracking robustness. In our experiment, we define tracking rate as the ratio of the longest sequence of successfully tracked frames with respect to the total number of frames input to the system. We used this error metric (error metric A) to analyse the suitability of the tracking algorithms.

To determine the accuracy of the pose estimation, we evaluate the synthetic results compared to the synthesised ground-truth poses with the odometry evaluation tools in Grupp (2017), which offer the same absolute trajectory error (ATE), and relative pose error (RPE) metrics as Sturm et al. (2012b). The absolute trajectory error (ATE) is the average difference between two estimated locations at each time after least-squares alignment with a rigid transform, while the relative pose error (RPE) measures the difference in estimated motion of the aligned trajectories over a short period.

Normalising Parameters All tracking systems we tested had certain parameters which can influence when the system would report a tracking failure. In all systems, we set the number of detected ORB features to 2000, and the minimum number of 3D-to-2D match inliers to 15. In both SPLAT and the Homography tracker, we set the inlier reprojection threshold to 5.0 pixels at a resolution of 512×512 pixels.

5.5.2 Tracking Rate

In our synthetic tests, we found that all three methods handled the challenges of varying space sizes very differently. Our approach was able to reliably track in all tested space sizes. Our results are summarised in Figures 5.8 and 5.9.

ORB-SLAM Our results from this experiment showed that ORB-SLAM was able to track reliably with small space radii, but as the space increased in size, tracking began to fail. Between radius 2 and 5, tracking was very reliable. Between 6 and 14, tracking was sometimes reliable, but would occasionally lose track. And due to the nature of the circular video sequences, there are no opportunities for the algorithm to relocalise.

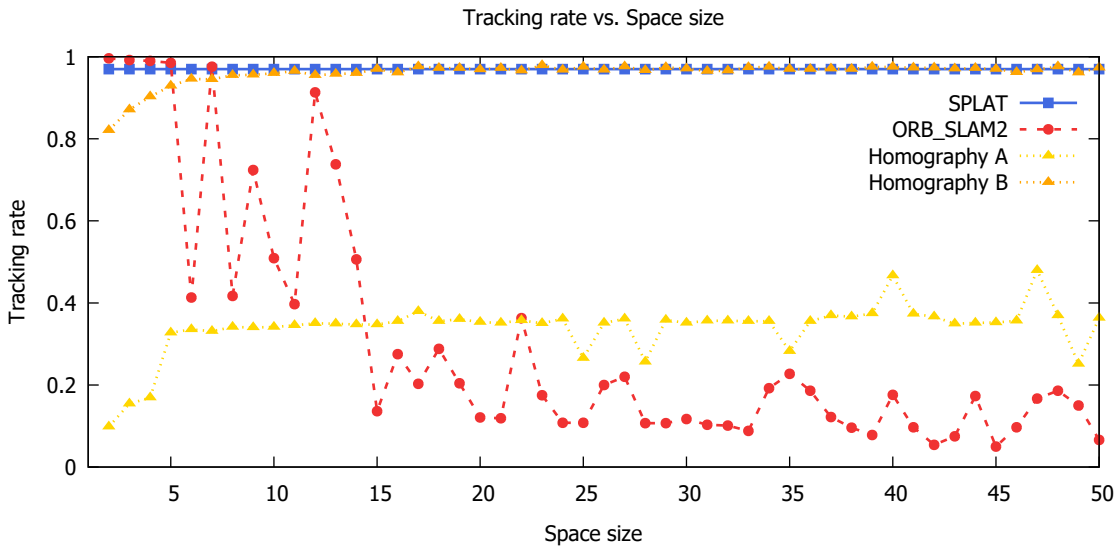

Figure 5.8 Tracking rate of our spherical tracker (blue), ORB-SLAM (red, Mur-Artal et al., 2015), and a homography-based tracker with two metrics (A yellow, and B orange), on our synthetic video sequences with different 3-D sphere radii.

From a space radius of 15 and higher, ORB-SLAM was usually able to initialise, but would lose tracking nearly immediately, resulting in very low tracking rates.

Homography Tracking We also compared our system to a simple homography tracking system based on a simple panorama stitching pipeline [2]. We modified it to perform sequential matching and use the same number of ORB features as the other algorithms. Since this method is essentially performing homography estimation between sequential frames with no propagation of existing matches (through feature tracking), it performed very poorly when using our tracking rate metric A described in Section 5.5.1 (yellow line, Figure 5.8). We include another, more forgiving, tracking rate metric B, which is the percentage of sequential frame-pairs where homography estimation was successful (orange line, Figure 5.8). In both error metrics, the homography estimation was less reliable for smaller space sizes.

SPLAT Our spherical SLAM method was able to track reliably in all test cases. We found that with a small space radius, our algorithm is somewhat sensitive to the frequency of the map updates (e.g. if the keyframe sphere has too few anchor points). We chose 1000 anchor points, as this produced around 70 keyframes for a full circle,

[2] https://opencv.org/release/opencv-3-4-3/

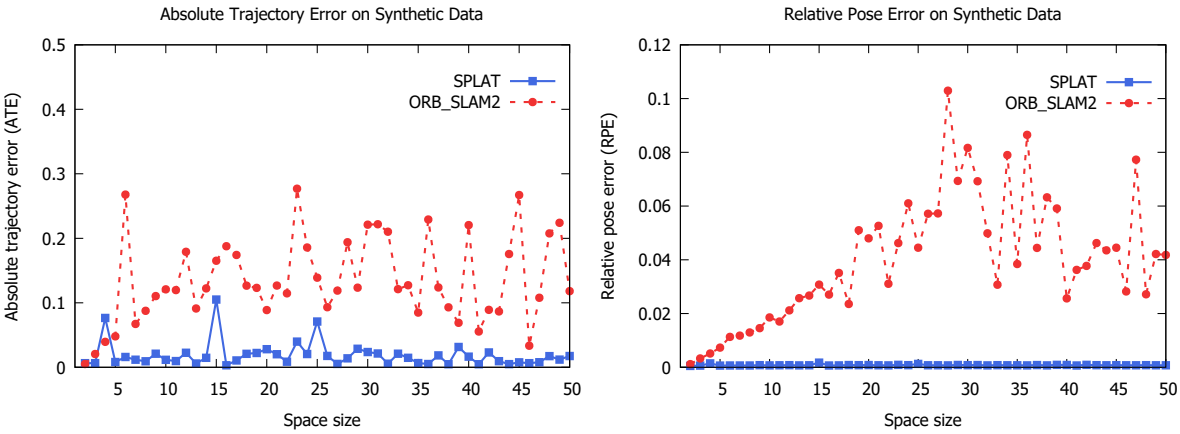

Figure 5.9 Pose error from synthetic testing of our tracking system (blue), compared to ORB-SLAM (red, Mur-Artal et al., 2015) with different 3-D sphere radii.

which is a similar number to ORB-SLAM. We also found good results with as few as 250 and as many as 2000 anchors, with larger values producing more keyframes at the cost of performance.

One of the main challenges of tracking in small spaces is that small camera movements can result in large disparities. As the objects are much closer, they move in and out of view more quickly which can cause similar tracking issues to those of fast movements. We do show, however, that our method is much more reliable than ORB-SLAM in this scenario, and that homography-based tracking can also struggle with smaller space radii. In the next section, we test our method on real data captured from a mobile device, in order to determine how reliably we can track when the movement is not perfectly spherical.

5.5.3 Accuracy

To compare the accuracy of our system to the state of the art, we computed the absolute trajectory error (ATE) and relative pose error (RPE) of both SPLAT and ORB-SLAM with respect to the synthetic ground-truth. In Figure 5.9, we plot the mean ATE and RPE for each synthetic video sequence with the corresponding size of the 3-D sphere. The RPE metrics were taken at 1-frame intervals, which corresponds to 0.36° of circular motion at radius 1 for all tested space sizes.

The ATE results for ORB-SLAM suggest that increasing the space size beyond 10 units introduces large variations in the ATE. Looking at the RPE metric, there is a more steady increase of error up to approximately radius 15, which is similar to the point

where the tracking rate of ORB-SLAM began failing in Figure 5.8. The large variations beyond this point would likely be due to the early tracking failure, resulting in fewer successfully tracked frames to compare with the reference poses. Overall, the results suggested that SPLAT had a lower pose error with both metrics. The error remained low as the space size increased, without suffering from the steady RPE error scaling with respect to space size that was exhibited by ORB-SLAM.

5.6 Real Data Experiments

For our real-life experiments, we captured a set of ten video sequences with different light conditions, and in environments of various scales, including a large sport stadium. The data was captured with a OnePlus 6 smart phone at a frame rate of at least 30 Hz and a resolution of 1920×1080 pixels. Sample frames from each sequence are shown in Figure 5.10.

5.6.1 Experimental Method

During the capture, the mobile phone was rotated around the body of the user at arm's length. This constraint was not enforced strictly, and all sequences were captured casually from either a fixed seated or fixed standing location. To compare our SLAM system to the state of the art, we run ORB-SLAM in a similar way to our system, configured with the same calibration parameters and number of ORB features.

Since we have no ground-truth pose or point data for the real experiments, the experiments in Section 5.6.2 are presented using a tracking rate defined by the states of either tracking or lost tracking. Both systems are configured to avoid *re-initialisation* when tracking is lost and will instead attempt to *relocalise* from the previously initialised map. To measure our pose accuracy, we compare poses output by our system to output from a RealSense T265 Tracking Camera. This device is state-of-the-art hardware, purpose-built for real-time motion tracking. This is discussed in Section 5.6.3.

5.6.2 ORB-SLAM Tracking Rate Comparison

In general, the experiments confirm our initial findings from the synthetic experiments. In particular that ORB-SLAM has problems tracking in large spaces (see Figure 5.11).

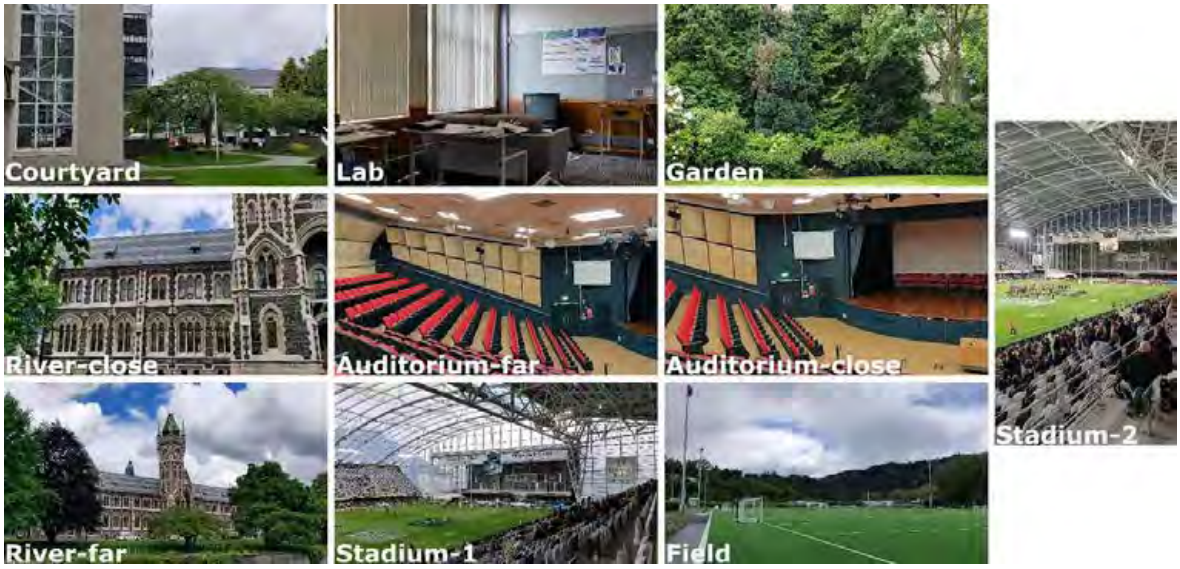

Figure 5.10 Sample frames from each of the ten video sequences used in our real data tracking experiments. The dataset consisted of ten video sequences in various spaces including close-range indoor, and long-range outdoor.

The results from the experiments are depicted in Figure 5.12. We tested our method and ORB-SLAM on several video sequences in the manner described in Section 5.6.1.

ORB-SLAM performed well in large spaces some of the time. In tests Field, and Stadium-1, tracking was robust once initialised. However, in Stadium-2, tracking was lost shortly after initialisation. Both sequences were taken from a sport spectator perspective within a live rugby game. This could suggest that tracking success is dependent on a successful initialisation in large spaces.

SPLAT tracked robustly in all the sequences, with a slightly delayed initialisation in Stadium-1. A delayed initialisation in our system would usually mean that the triangulation resulted in some points behind the camera. These initialisations can be caused by moving objects, such as other spectators, or insufficient parallax, and are discarded.

In general, we found high tracking rates with our algorithm. This is desirable for Augmented Reality applications and would reduce the amount of work needed to be done by a relocalisation module. A common use case of our method would be with full integration into a global localisation module, and perhaps registering to a prior model of the environment. Such registration can be computationally expensive, so maintaining robust tracking is important.

In some of the very large spaces, such as Stadium-1 and Stadium-2, our method takes

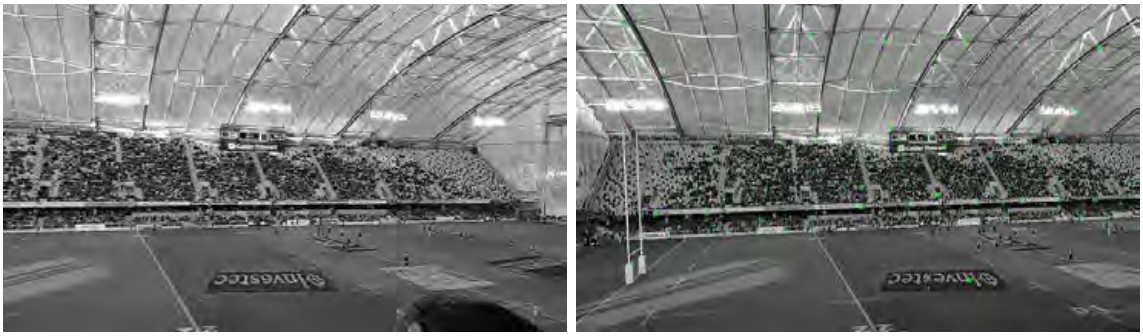

Figure 5.11 ORB-SLAM failing to initialise on one of the stadium sequences. Left) Trying to initialise. Right) Successful initialisation after more frames.

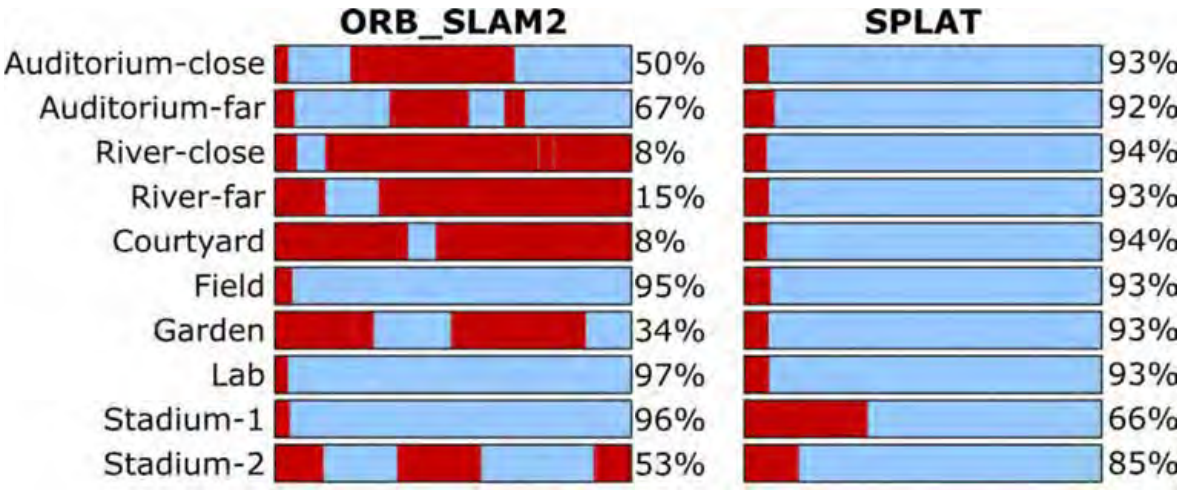

Figure 5.12 Tracking timelines of ORB-SLAM compared to our Spherical SLAM system across ten video sequences at 540p video resolution. Blue represents a successfully tracked frame, red represents an untracked frame. We show the percentage of successfully tracked frames (metric B) alongside each sequence.

a little more time to initialise. But once initialised, tracking is robust. The largest space in all our test videos was River-far, depicting a large building from across a river. This case was able to initialise quickly with SPLAT but did not perform well with ORB-SLAM. Notably, ORB-SLAM initialised more quickly on River-close, which falls in line with the synthetic results, where initialisation was fast on smaller scenes.

In some cases, ORB-SLAM is able to initialise sooner than SPLAT, such as the Stadium-1 and Lab sequences. As mentioned in Section 5.3.2, our system attempts to initialise a map after a predetermined amount of rotation, and in the case of initialisation failure, the entire process is restarted which can result in slow initialisation. This could be optimised by automatically detecting when a good enough initial map

Table 5.1 Absolute trajectory error (ATE) and relative pose error for SPLAT and ORB-SLAM compared to RealSense tracking reference poses. Relative pose errors are computed as translation and rotation differences over 6 frame (0.1s) intervals.

		SPLAT			**ORB-SLAM**		
Data Set	n	ATE (m)	Translation (m)	Angle (deg)	ATE (m)	Translation (m)	Angle (deg)
Spectator 1	1339	0.026 ± 0.017	0.0070 ± 0.0032	0.64 ± 0.35	0.024 ± 0.015	0.0075 ± 0.0034	0.67 ± 0.39
Spectator 2	808	0.016 ± 0.007	0.0076 ± 0.0040	0.68 ± 0.42	0.017 ± 0.009	0.0081 ± 0.0053	0.72 ± 0.48
Spectator 3	519	0.014 ± 0.006	0.0053 ± 0.0041	0.69 ± 0.38	0.013 ± 0.006	0.0058 ± 0.0043	0.71 ± 0.38
Spectator 4	901	0.026 ± 0.016	0.0053 ± 0.0033	1.01 ± 0.57	0.028 ± 0.016	0.0054 ± 0.0034	1.14 ± 0.62

is able to be computed, rather than initialising after a fixed amount of motion. This is similar to how ORB-SLAM operates, and could result in more comparable initialisation speed. As our system is more a demonstration of applicability of the spherical constraint to SLAM, we leave this optimisation for future work.

5.6.3 RealSense Accuracy Comparison

To determine accuracy, and to see how well the spherical motion constraint holds true to real AR user motion, we compared our pose outputs to a RealSense T265 Tracking Camera. The RealSense provides a SLAM solution integrating stereo fisheye cameras with inertial measurement. This provides us with reference trajectories independent of the video stream used to compute SPLAT and ORB-SLAM tracks, but cannot be considered absolute 'ground truth' as the RealSense solution is itself subject to tracking errors. This data was captured from the stadium stands, modelling a spectator viewing a sports event and tracking play on the pitch, and was separate from Section 5.6.2.

The trajectories from SPLAT and ORB-SLAM were aligned to the RealSense tracking path and compared using the TUM evaluation tools (Sturm et al., 2012b). In this work, the authors suggested best practices for comparing estimated trajectories from methods such as SLAM systems with reference trajectories from toher sources. They provide an open source toolset which we use to compute the absolute trajectory error (ATE) and relative pose error (RPE) as previously desribed in Section 5.5.1. For RPE, we evaluate trajectories over a short period (we use 6 frames, or 0.1s). Here we provide RPE values for both translational and angular errors. The RealSense reports translations in units of meters, which we adhere to in our results. These results are shown in Table 5.1.

The ATE values indicate that ORB-SLAM generally gives better alignment to the RealSense data by a small margin, but the relative pose over short periods is more

accurately estimated by SPLAT, with better estimation of both translation and orientation over short time-frames (0.1 seconds). Statistical validation of these differences needs to be dealt with carefully since the relative pose errors are not independent of one another for overlapping time segments. To overcome this, we take sequential non-overlapping 0.1s (6 frame) periods. Furthermore, there are eight comparisons (translation and pose for each sequence), so a Bonferroni correction is applied to our significance threshold. Starting with a threshold of $p = .05$ yields a corrected significance threshold of $p = .00625$. Two-tailed paired t-tests then indicate a significant improvement in RPE for SPLAT over ORB-SLAM for both translation and rotation in the sequence Spectator 1 and for rotation in Spectator 4. Translation in Spectator 3 is close to the corrected threshold ($p = .0077$). All other cases have p-values larger than the uncorrected threshold.

The differences in ATE are more difficult to interpret, since the ATE between SPLAT and ORB-SLAM is generally smaller than the ATE between either tracker and the reference (RealSense) path. This could indicate some overall drift in the RealSense tracking. The short-term relative poses (as used in the RPE transformation) should be more reliable due to the integration of inertial sensors in the RealSense tracking. Despite these limitations, the RealSense provides a reference for comparison that is computed independently from the SPLAT or ORB-SLAM trackers. From this comparison, it appears that SPLAT and ORB-SLAM provide similar accuracy, with SPLAT providing a statistically significant improvement in relative pose estimation over short periods in some cases.

5.6.4 Performance

The spherical tracking method runs currently with an average of 25.1 frames per second with 2000 ORB features per frame, at a resolution of 960×540. With 750 features, we maintain similar tracking rates at a frame rate of 35.6 frames per second. The hardware we use is a modern desktop with an Intel Core i9 processor. The method is so far not optimised for speed.

We profiled the per-frame execution time of our spherical P2P solver and compared this to OpenCV's P3P solver (Bradski, 2000) which is based on the method by Gao et al. (2003). We ran the same video sequence on both algorithms within the same SLAM framework, taking timings before and after the RANSAC computation. We use Preemptive RANSAC (Nistér, 2005) with 500 samples, and a block size of 50.

Table 5.2 Profiling results of OpenCV's implementation for solving Perspective-3-point (OpenCV P3P) problems for general camera motion compared to our constrained Spherical Perspective-2-Point (Spherical P2P) solution. Here we present the mean (M), standard deviation (SD), and standard error of the mean (SEM) for each method.

Method	M (ms)	SD (ms)	SEM (ms)	n
Spherical P2P	22.31	9.68	0.46	443
OpenCV P3P	28.36	10.76	0.51	443

Due to the unconstrained pose from the general P3P solver, the tracking produced unsatisfactory results for an AR application, while our method produced more stable pose output. Both algorithms were able to successfully track to the end of the sequence once initialised, for a total of $n = 443$ frames.

Again, with a significance threshold of $p = .05$, we ran a paired two-tailed t-test on the computation time for both methods. Spherical P2P showed small but significant speedup with $p \leq .0001$. This suggests that using a spherically constrained pose solution in place of a general solution can offer performance benefits, despite our less-optimised overall solution. The mean performance improvement per frame was approximately 6 ms, which could account for a reasonable frame rate boost. The computation results are shown in Table 5.2.

5.6.5 AR Prototype

We integrated the spherical SLAM into an AR prototype. For this purpose, we use the computed poses to place virtual content at the corresponding location within the real environment. The AR prototype uses OpenGL for rendering and the Assimp library [3] for model loading. We implemented different prototypes, one for visualising 3-D content for stadium environments (more specifically Rugby, Figure 5.13, right), one for sightseeing applications visualising labels corresponding to sights in an urban environment (Figure 5.13, left) as well as a guide for lecture theatres. For content placement in these AR applications, we make use of the 3-D point cloud created by the SPLAT system. By selecting a 3-D point in the point cloud with a mouse click, we place content at this location.

Through our AR prototype, we were able to demonstrate some important advantages in visualisation that our SLAM system has over a typical homography-based tracker. In many AR scenarios, it is desirable to place virtual objects at varying depths within

[3] http://www.assimp.org

Figure 5.13 Application scenarios for our AR prototype. Left: sightseeing application with distant billboards. Right: up-close virtual display board for rugby.

the scene (such as small widgets near the user, and large billboards in the distance). While it is possible to render these objects with homography-based tracking, the objects would exhibit no motion parallax with respect to each other as the user moves. This kind of rendering would only be correct if the user rotated about the camera centre, which is unrealistic. With our system, the objects can appear to cross over each other as the user moves, adding an extra depth cue, and resulting in a more convincing visualisation. In our demo, we do not address the fact that our system operates at an arbitrary scale. However, localising tracked frames to an existing map of known scale could provide the information required for scale correction.

5.7 Conclusion

In this chapter, we proposed SPLAT – a SLAM system based on the spherical motion assumption. We developed a method for computing spherically constrained absolute pose relative to a 3-D map and integrated this into a complete tracking solution.

Our method showed more reliable tracking rates compared to state-of-the-art ORB-SLAM, and we compared the accuracy of both methods to reference data from a RealSense tracking camera. We found that while ORB-SLAM had advantages in ab-

solute trajectory errors, our method performed more reliably in synthetic and real experiments. Our experiments showed SPLAT's ability to track reliably in spaces of a range of sizes, and we demonstrated its application through an AR rendering system.

When presented with spherical motion data, a general SLAM system can be expected to operate reliably in small spaces, while a panoramic tracker would perform reliably in large spaces. However, our method can be applied to many environments ranging in scale from an office laboratory to a large sport stadium, provided the camera motion is spherical. This has many direct applications and is useful for situations where the environment scale is unknown.

For future work, we want to focus on improving the performance of the system, as our initial profiling shows performance advantages in using a simpler spherical pose computation. We also think that more investigation into the benefits of the keyframe sphere data structure could be useful, such as using it to reduce global bundle adjustment overhead, in a similar way to ORB-SLAM's essential keyframe graph (Mur-Artal et al., 2015).

There is potential for extending this system to a hybrid method (similar to Gauglitz et al., 2012), combining both spherical and general motion by linking multiple keyframe spheres and tracking general motion between locations. A method like this would be useful in tourist applications, where users are moving from point-to-point, occasionally initiating spherical motion when they stop to look around.

Acquiring ground-truth pose data for large scale video sequences is a priority and would allow for more extensive accuracy evaluations. It is difficult to attain this, as most datasets used for tracking do not follow the spherical motion assumption as discussed in Chapter 3. More accurate reference data could come in the form of video sequences registered to pre-computed SfM models but would be subject to error.

Finally, pairing our tracking system with a global localisation method (such as those discussed in Chapter 4) would improve its usefulness in AR. In a sport stadium, for example, this would allow AR content to be easily placed in meaningful locations and be displayed correctly for multiple spectators.

Chapter 6

Spherical SfM for Stereo Panorama Generation

In Chapter 5, we covered the possibility of applying the spherical motion constraint of Ventura (2016) to real-time tracking of stationary users in large spaces. We found that our approach, SPLAT, was able to track more reliably than state-of-the-art ORB-SLAM through stationary spherical motion in large spaces, while being less constrained than rotation-only 3-DoF trackers.

In this chapter, we explore the application of the spherical SfM concepts from Ventura (2016) to capturing stereo panoramas for VR. Stereo panoramas are a form of panorama that allow for depth perception by displaying two stereoscopic views to the user, creating a more realistic experience when viewed in VR. State-of-the-art methods for creating stereo panoramas are typically not required to run in real-time, as panorama creation can be considered a pre-processing stage, and the viewing of the resulting panoramas can be handled by VR hardware. As previously mentioned, much of the work in this chapter has been published in the poster Baker et al. (2019), and the paper Baker et al. (2020a).

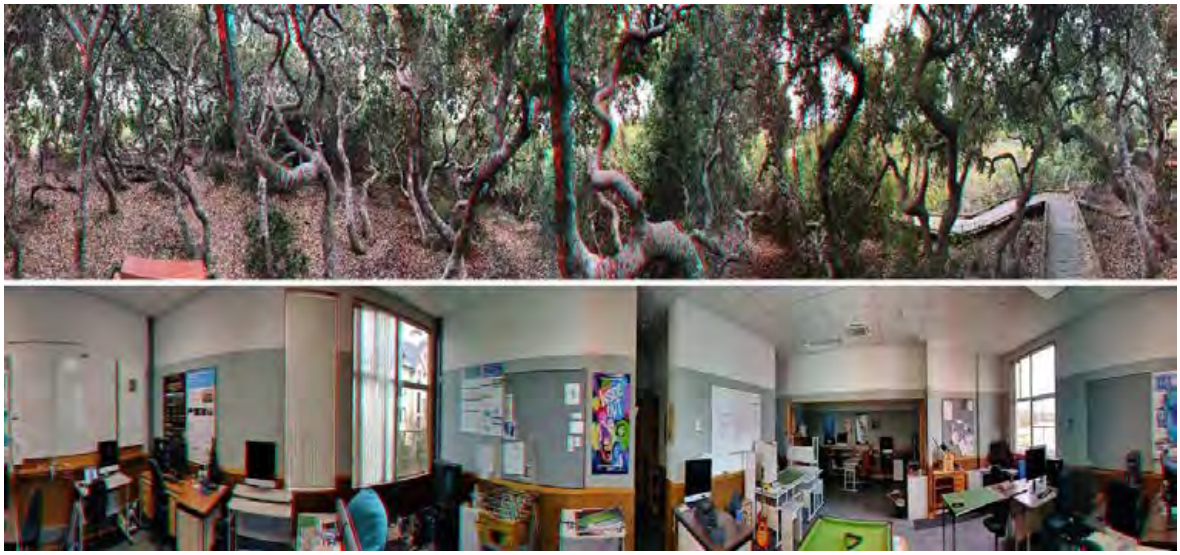

Figure 6.1 Example stereo panoramas produced using our spherical structure-from-motion pipeline, rendered as red-cyan anaglyphs. Top: "ElfinForest", Bottom: "Lab360" sequences (best viewed with red-cyan 3-D glasses).

6.1 Introduction

With the continuous rise of immersive viewing devices, such as head-mounted displays (HMDs), there is a growing need for consumer-grade methods of virtual reality (VR) content capture. The ideal VR content capture modality is inexpensive, lightweight and easy-to-use. For example, most smartphones and cameras today offer a panoramic capture mode, in which the user can stitch together a 360-degree field-of-view image by turning the camera in a circle. However, HMD viewing is greatly enhanced by stereoscopic viewing to give a sense of depth to the user, which a normal panorama cannot provide.

A viable alternative is the stereo panorama (Shum and Szeliski, 1999; Peleg et al., 2001; Li et al., 2004) which does provide stereoscopic, 360-degree viewing—two examples of which are shown in Figure 6.1. To capture a stereo panorama, the camera is spun in a circle while facing outward. Columns from the left and right side of each image are concatenated to form the right- and left-eye panoramas, respectively. The disadvantages of this approach are the need for a perfectly circular trajectory and a high density of views. The angular resolution of the resulting panorama is determined by the number and density of the input images.

Many recent works explore *casual* capture of stereo panoramas (Richardt et al., 2013;

Zhang and Liu, 2015), where the camera can be handheld and spun in a roughly circular trajectory. To recover the pose of each camera, traditional structure-from-motion methods (Schönberger and Frahm, 2016; Schönberger et al., 2016) are used. Once the camera poses have been determined, new views on a perfectly circular path are synthesised using flow-based blending. However, this case of camera motion is especially difficult for traditional structure-from-motion (SfM) methods because of the small baseline between views, and the relative lack of overlap between frames.

In this chapter, we evaluate the use of spherical structure-from-motion (Ventura, 2016) as an effective alternative for reconstructing handheld stereo panorama sequences. Spherical SfM differs from general SfM in that the camera is assumed to move on the surface of an imaginary sphere (i.e. the camera is assumed to maintain a constant distance from the origin and to always be facing directly outward). Handheld stereo panorama capture matches these assumptions well since the camera is held in an outstretched hand and spun roughly in a circle.

The spherical constraint removes disambiguity in two-view relationships and provides a strong, implicit regularisation on the SfM result. In addition, because each camera pose is determined completely by the rotation of the camera, there is no need for scale propagation and incremental reconstruction as in traditional monocular SfM. In this chapter, we show that by using spherical structure-from-motion, we can produce a stereo panorama from an input video sequence in minutes rather than hours as required by previous work (Richardt et al., 2013).

6.1.1 Chapter Overview

In this chapter, we review the theory of spherical structure-from-motion and describe our spherical SfM and stitching pipeline for causal stereo panorama generation. We then evaluate the use of spherical SfM for processing handheld-captured videos and the resulting stereo panoramas produced using the camera pose estimates within a technical evaluation as well as within a user evaluation. Our evaluation on several handheld captured sequences shows that we can reconstruct casual stereo panorama capture trajectories and produce high-quality stereo panoramas, and that even untrained users can produce suitable input videos for our pipeline.

Specifically, our contributions are as follows:

- We describe a spherical structure-from-motion and stitching pipeline for reconstruction of stereo panoramas from casually captured input videos.

- We compare the speed and reliability of our approach to COLMAP (Schönberger and Frahm, 2016), a state-of-the-art general SfM method, on several input videos. We show that spherical SfM reconstructs the videos more reliably and quickly than general SfM.

- We show that spherical SfM reconstruction is accurate enough to produce high-quality stereo panoramas through qualitative evaluation and a quantitative user study.

The remainder of this chapter is outlined as follows. We first provide a brief overview of the required background knowledge in Section 6.2. We then detail our method for applying spherical SfM concepts of Ventura (2016) to panorama generation with flow-based blending (Richardt et al., 2013; Bertel et al., 2019) in Section 6.3. We then report the results of our technical evaluation, including synthetic testing in Section 6.4. Then, we outline our user study, and report the results in Section 6.5. Finally, we conclude this chapter in Section 6.6.

6.2 Background

Our method builds on recent structure-from-motion methods that constrain the solution to motion on a sphere (Ventura, 2016), and applies them to stereo panorama reconstruction. This section provides a brief overview of these areas specifically in the context of panorama capture.

6.2.1 Structure-from-Motion

Structure-from-motion (SfM) is the process of determining the camera poses and 3-D structure of a scene from a video or collection of images. Most systems take an incremental approach, where the reconstruction is grown from an initial pair of images (e.g. Snavely et al., 2006, and Schönberger and Frahm, 2016). An alternative strategy is to attempt to first determine the camera rotations independent of their translations (Wilson and Snavely, 2014) and then estimate the translations afterwards.

These methods form the basis of SfM techniques that can model scenes from large collections of images (Schönberger and Frahm, 2016). Such techniques rely on general camera motion and, as we shall see, can fail in the limited motion made in panorama capture scenarios. Most small-motion SfM methods (Yu and Gallup, 2014; Im et al.,

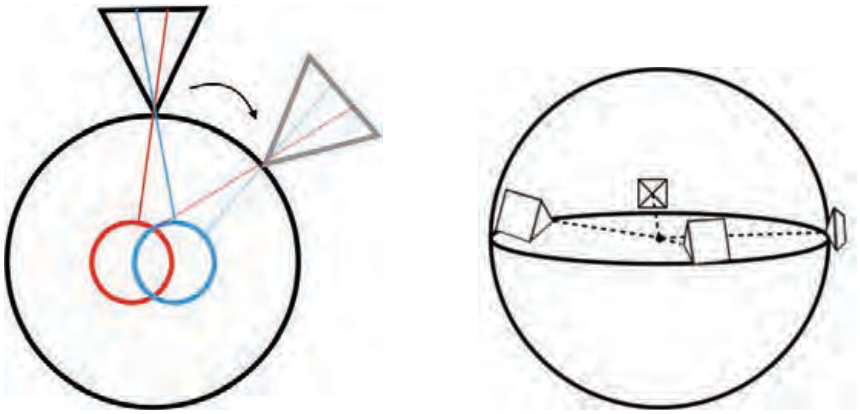

Figure 6.2 Left: Illustration of idealised stereo panorama capture. Images (black triangles) are taken in a dense set of locations along a circular path. Left- and right-eye panoramas (red and blue circles) are stitched together from different columns in the images. Right: Under the spherical motion constraint, the camera can move freely, but only across the surface of the unit sphere. We show that a user causally capturing a stereo panorama with a camera held in an outstretched hand will produce a trajectory that agrees with this constraint well enough to produce a high-quality stereo panorama using our method.

2015; Ha et al., 2016) use the inverse depth parameterisation and depth map regularisation to achieve accurate triangulation, but are not designed to reconstruct a 360-degree stereo panorama. SfM systems can make use of spherical panoramic cameras (e.g. Torii et al., 2005, and Pagani and Stricker, 2011) can be used to construct a stereo panorama (Im et al., 2016) but this is not always practical, as many people do not own panoramic cameras. In this chapter, we use a normal perspective camera as would be found on a typical smartphone or other consumer device for this purpose.

6.2.2 Spherical Structure-from-Motion

In this chapter, we explore the use of a spherical motion constraint (Ventura, 2016) to regularise the reconstruction and overcome the small-baseline problem inherent in stereo panorama capture. Introducing a spherical constraint reduces the number of point correspondences needed between image pairs, and improves robustness to the limited motion, as we show in Section 6.4.

Spherical motion can be seen as an approximation of the type of motion that occurs when a user moves or rotates a camera at arm's length from their body (Ventura, 2016). Here, we focus on its application to outward-facing cameras in the context of casual panorama capture with mobile devices.

In spherical motion, the translation component of the camera pose $\mathbf{t}$ is constrained to a fixed vector $\mathbf{z} = [0\ 0\ 1]^\mathsf{T}$, which allows the camera position to move freely on the surface of the sphere. When the camera is facing outward, the pose of the camera can be represented as

$$\mathsf{P} = [\,\mathsf{R}\,|\,-\mathbf{z}\,]. \tag{6.1}$$

The work of Ventura (2016) introduces this constraint, and provides methods for computing relative pose under these conditions. In this work, Ventura also proposes a full SfM system that utilises feature tracking to estimate relative poses, combined with loop closure and bundle adjustment. We argue that these techniques are ideal for usage in panorama capturing scenarios (as also confirmed in recent work by Sweeney et al. (2019) for computing spherical fundamental matrices for panoramic video), so we apply these techniques extended to suit the specific type of circular motion that is prevalent in panorama capture. We describe our approach to this in the following section.

6.3 Method

In this section, we outline our method for reconstruction of stereo panoramas from casually captured circular video. Similar to regular panorama capture, capturing a stereo panorama requires the user to turn on the spot with their device held out, creating a roughly circular motion path as illustrated in Figure 6.3, (a). The preferred capture orientation is to hold the camera in "portrait" orientation to maximise the vertical field-of-view. Using portrait orientation causes a small visual overlap between views; in other words, feature points are not in the field of view for very long. Additionally, because the circle of motion has a relatively small radius (the length of a human arm) there is not a large baseline between frames for point triangulation. A second issue is that the camera moves in a complete circle before ever reaching a loop closure, so accumulation of drift is unavoidable with general visual tracking.

This capture setup is especially difficult for incremental structure-from-motion systems (Snavely et al., 2006; Schönberger and Frahm, 2016) because there is usually no good initial pair of keyframes to start the SfM process. General essential matrix estimation with a small baseline is unstable and does not give good results, and scale propagation is especially difficult. In addition, reconstruction of 3-D points from an initial pair of

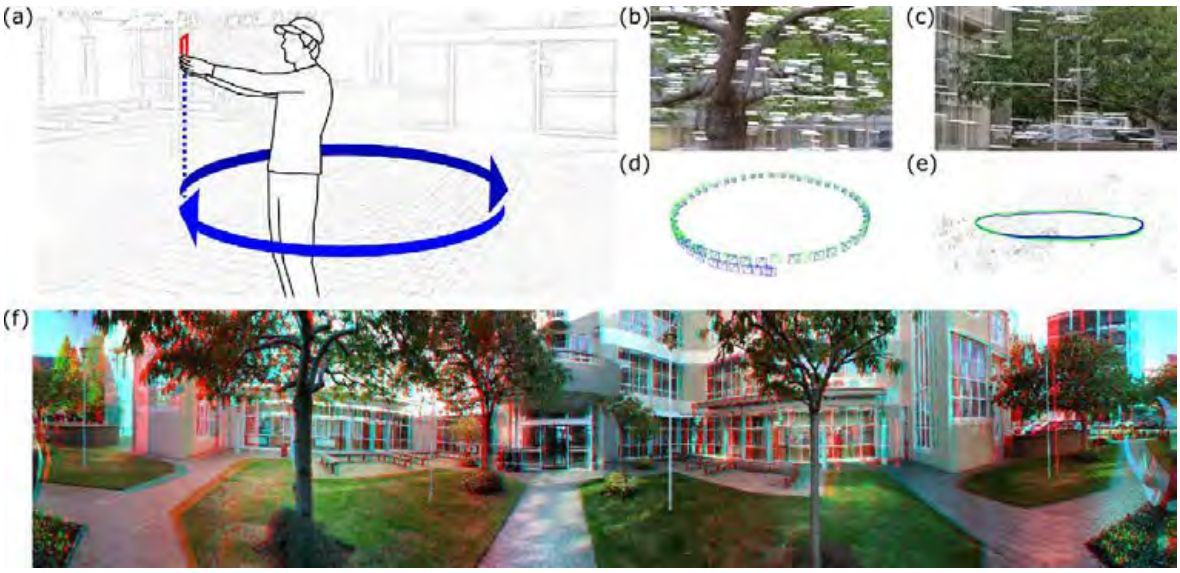

Figure 6.3 Overview of our approach. (a) The user spins around their axis to capture the stereo panorama. (b) to (e) Tracking, loop detection, rotation averaging, and bundle adjustment are performed. (f) Multiple panoramic views are synthesised, and stereo anaglyphs can be rendered.

keyframes gives poor depth estimates due to the small baseline, which in turn makes it difficult to register new views to the reconstruction.

Spherical SfM is a natural fit for casual stereo panorama capture, because the camera moves on an approximately circular path. In spherical SfM, it is assumed that camera centre maintains a constant radius from the origin, so that the camera moves along the surface of an imaginary sphere. Furthermore, the camera viewing axis is assumed to always be parallel to the normal of the sphere, i.e. the viewing axis is coincident with ray from from centre of the sphere to camera centre. For handheld capture, the centre of rotation would be approximately at the shoulder of the arm holding the camera, and the viewing axis pointing out along the user's outstretched arm.

6.3.1 Spherical SfM Reconstruction Pipeline

In this section, we describe our spherical SfM reconstruction pipeline for casual stereo panorama capture. The major modifications to the spherical SfM pipeline introduced in previous work (Ventura, 2016) are the use of SIFT features (Lowe, 2004), automatic sequence sub-sampling, constrained loop closure search, L2 rotation averaging (Chatterjee and Madhav Govindu, 2013), and full 3-D bundle adjustment to improve speed, accuracy and robustness. The input to the pipeline is the video captured by the user

and the intrinsic parameters of the camera. The output is the estimated pose of an evenly distributed sub-sequence of frames from the video. Each step of the pipeline is described in the following.

Frame-to-frame Tracking We process each frame in the video sequentially. In the first frame, we detect SIFT feature points and extract their descriptors (Lowe, 2004). Then we track these features into the next frame using pyramidal Lucas-Kanade (LK) tracking (Lucas et al., 1981; Tomasi and Kanade, 1991). We found that using LK tracking, instead of detecting and matching SIFT features by their descriptors, provides higher accuracy, sub-pixel feature tracks which are critical for 3-D reconstruction with small-baseline image pairs. When we track a feature with LK, we also copy its new descriptor to the tracked feature point in the next frame instead of re-computing the descriptor. After LK tracking, we also detect new SIFT features that are reasonably far from existing features in the image.

We then use these matches to estimate the spherical essential matrix (Ventura, 2016) and separate inliers and outliers in a Preemptive RANSAC loop (Nistér, 2005). Outlier matches are removed from further consideration. The result of this process is an estimated rotation $\mathsf{R}_{i\to j}$ and a set of inlier feature matches between each pair of consecutive frames i, j in the sequence. If the amount of rotation between the two frames is less than one degree, we drop frame j from the sequence and restart the matching process between frames i and frame $j+1$. This loop continues until we have found the next frame in the sequence with a large enough rotation to the current frame, at which point the image pair is added to the reconstruction.

This process of adaptively selecting a subset of reasonably spaced frames from the video helps to avoid sub-sequences with very small or no motion in the video, as these cause problems in the later triangulation and bundle adjustment steps. The adaptive sub-sampling also establishes a rough cap on the maximum time required for the reconstruction process. Note that, without the spherical constraint, it is difficult to automatically prune the video in this way, because in general the essential matrix only defines the translational part of the relative pose up to scale. For example, Bertel and Richardt (2018) manually subsample their input sequences before processing them in COLMAP, as COLMAP is unable to perform this sub-sampling automatically.

To compute the initial camera poses for the reconstruction, we integrate the estimated relative rotations between image pairs, i.e., $\mathsf{R}_j = \mathsf{R}_{i\to j}\mathsf{R}_i$ for each consecutive image pair i, j in the reconstruction. The first camera is fixed to have the identity rotation.

Loop Closure We search for the loop closures using the first thirty and last thirty frames of the sequence, after sub-sampling the sequence to one-degree increments as described above. For each frame in the first thirty frames, we attempt to calculate the relative pose to each of the last thirty frames. Specifically, we match nearest neighbour SIFT features, making use of the ratio test to reject ambiguous matches. For each candidate loop closure pair, we run the Preemptive RANSAC matching procedure to estimate the spherical relative poses between the two images. Each loop closure with greater than 100 inliers is accepted. The loop closures are integrated into the initial pose estimate using rotation averaging as described next.

Rotation Averaging Pose drift accumulates during the frame-to-frame tracking and relative pose integration process. To reduce drift, we perform L2 rotation averaging (Chatterjee and Madhav Govindu, 2013) over the graph of relative rotation estimates produced by the frame-to-frame tracking and loop closure processes. The result of this process is a corrected initial pose estimate for every frame in the sequence.

Structure Initialisation The feature tracks are assembled from the inlier feature matches found during the frame-to-frame tracking and loop closure steps. We then find an initial 3-D point estimate for each feature track by applying direct linear transform (DLT) triangulation (Hartley and Zisserman, 2003) with all observations in each track. We found that having all observations available for triangulation is critical for these sequences because of the small baseline between frame pairs. In contrast to an incremental bundle adjustment, where it is critical to find good image pairs for the initial point triangulation, we can use all available views in the sequence to initialise the bundle adjustment procedure.

Bundle Adjustment Finally, we optimise the re-projection error in a bundle adjustment procedure, updating the 3-D point locations and the camera rotations in an iterative manner. We parameterise points as 3-D vectors instead of applying an inverse depth parameterisation as has been previously used for spherical SfM (Ventura, 2016). Specifically, we find the rotations $\mathsf{R}_1, \ldots, \mathsf{R}_m$ and 3-D point positions $\mathbf{p}_1, \ldots, \mathbf{p}_n$ that minimise the total re-projection error:

$$\sum_{i,j} \mathsf{M}_{i,j} \sigma(||\mathbf{o}_{i,j} - \pi(\mathsf{K}(\mathsf{R}_i \mathbf{p}_j - \mathbf{z}))||^2) \tag{6.2}$$

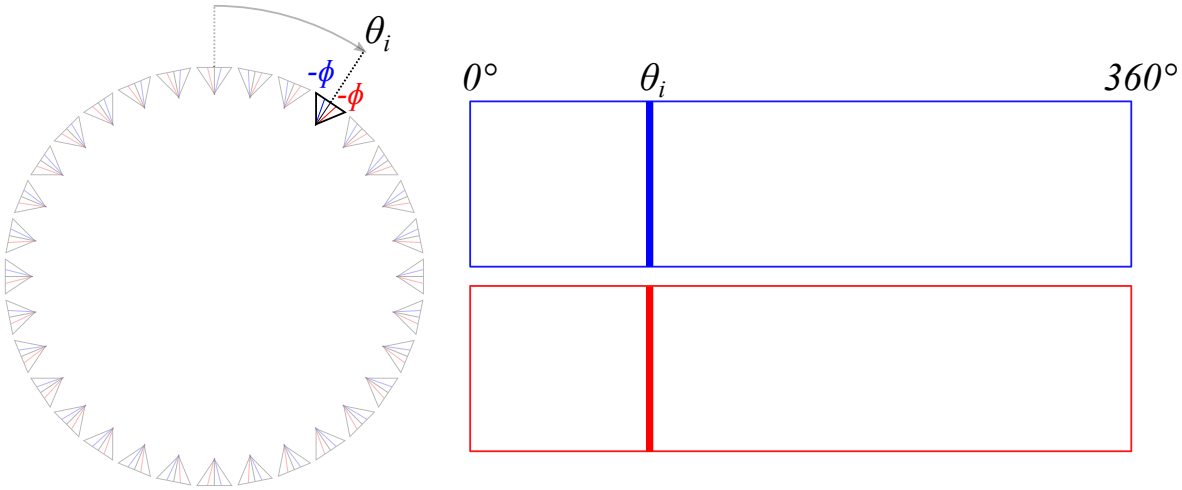

Figure 6.4 Simple interpretation of the approach taken by Shum and Szeliski (1999). For a camera rotating in a perfect circle, stereo panoramas can be made by simply sampling columns at a predetermined angle, ϕ, either side of the centre for each frame of video. Each column of the output panorama corresponds to a frame of video on the ideal circular trajectory, θ.

where M is a Boolean matrix indicating the visiblity of point j in camera i, $\mathbf{o}_{i,j}$ is the observation of point j in camera i, K is the intrinsics matrix, R_i is the estimated rotation for camera i, and $\mathbf{p}_j$ is the estimated position of point j, and $\pi(\cdot)$ is the projection operator. We apply a robust cost function $\sigma(x) = \log(1 + x)$ to reduce the effect of outliers, and use the Ceres Solver (Agarwal et al., 2012) to perform the optimisation.

6.3.2 Stereo Panorama Stitching

Once the SfM pipeline has completed, we have the camera pose estimates necessary to stitch together stereo panoramas. For the ideal case of a perfect circular trajectory with equal angular spacing between frames, Shum and Szeliski (1999) describe the geometry of stereo (or multiperspective) panoramas. Under ideal capture conditions, each input view lies on a circular camera trajectory with an angle of θ around the circle. The columns in an input image are indexed by ϕ, the horizontal angle of the column offset from the centre of the image. Panoramas at different horizontal offsets can be produced by sampling column ϕ from each input view and stitching them into a single panoramic image (Figure 6.4).

With casual capture, the actual cameras deviate slightly from the circular trajectory in terms of pose, and view density. Megastereo (Richardt et al., 2013) and MegaParallax (Bertel et al., 2019) use the input views to synthesise a new set of views perfectly spaced

on the ideal circular trajectory. We adopt their approach to produce the necessary image columns for the output stereo panorama. We review the method briefly here.

Plane Estimation We robustly fit a plane to the camera centres in a RANSAC procedure. Then all cameras are rotated so that the plane normal coincides with the global up vector $\mathbf{y} = [0\ 1\ 0]^\top$. This correction step brings the camera poses closer to the ideal circular trajectory.

Organisation of Images We now order the images by their signed angle around a unit circle on the X-Z plane. To compute the signed angle, we first project the camera centres to the plane and compute each camera's angle θ_i around the circle. Let $\mathbf{x} = [1\ 0\ 0]^\top$ and $\mathbf{c}_i = [c_x\ c_y\ c_z]^\top$ be the camera centre for camera i which is projected to the plane. We compute the signed angle θ_i similar to Bertel et al. (2019) as

$$\theta_i = \text{atan2}(-c_z, c_x). \tag{6.3}$$

Flow-based Blending The final step in our pipeline is to generate the output panoramas using the flow-based blending technique from Bertel et al. (2019), which we outline here. As previously mentioned, the aim of flow-based blending is to synthesise a set of views that are spaced evenly along the perimeter of the circle. This allows us to use the standard panorama sampling of Shum and Szeliski (1999) to construct multiple panoramas for a range of ϕ values. The challenge here is how to adequately compensate for the disparities between the view we are trying to synthesise, and the closest matching views that are available.

Since we have already rotated each of our captured views to lie on the X-Z plane, we can synthesise all pixels in a column of the output panorama using a single pair of images. For each column of our output panorama, we first select the corresponding synthetic view I_D, as well as the two closest matching captured views I_L and I_R which lie in front and on either side of the viewing direction of I_D.

We then generate a proxy 3-D point $\mathbf{X}$ through each pixel in I_D at a fixed depth (we use 10 in our experiments). Then, $\mathbf{X}$ is projected into I_L and I_R giving us two sample pixels $\mathbf{x}_L$ and $\mathbf{x}_R$ which can be used as a basis for synthesising the views (as shown in Figure 6.5, left). From here, a blending weight α is computed based on the angular differences between the synthetic viewing direction, and the viewing directions of the captured views.

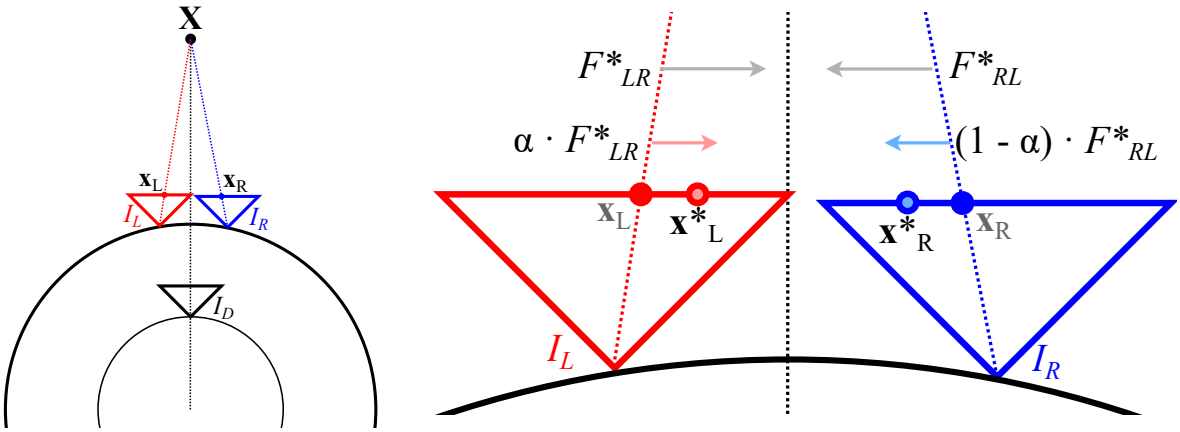

Figure 6.5 A visual representation of the flow-based blending technique introduced by Richardt et al. (2013) which we use in our pipeline. Left: The proxy point $\mathbf{X}$ is projected into the closest matching images I_L, and I_R to initial points $\mathbf{x}_L$ and $\mathbf{x}_R$ respectively. Right: The initial positions are adjusted using the blending weight α and the bidirectional optical flow displacements F^*_{LR} and F^*_{RL}. The final pixel value in the synthesised view I_D is blended with the weight α using the corrected pixel locations $\mathbf{x}^*_L$ and $\mathbf{x}^*_R$ (Bertel et al., 2019).

Applying the blending weight here is sufficient to produce simple stereo panoramas, but is prone to blurring as it does not properly compensate for disparity (Bertel et al., 2019). Richardt et al. (2013) address this problem by using the bidirectional optical flow (Brox et al., 2004) between I_L and I_R to compute the displacements F^*_{LR} and F^*_{RL} at each of the initial sample positions $\mathbf{x}_L$ and $\mathbf{x}_R$. Then, the sample positions are adjusted by the blending weight α using

$$\mathbf{x}^*_L = \mathbf{x}_L + \alpha \cdot F^*_{LR}(\mathbf{x}_L), \tag{6.4}$$

and

$$\mathbf{x}^*_R = \mathbf{x}_R + (1 - \alpha) \cdot F^*_{RL}(\mathbf{x}_R) \tag{6.5}$$

to produce corrected sample locations $\mathbf{x}^*_L$ and $\mathbf{x}^*_R$. A visual representation of this process is shown in Figure 6.5, right. From here, the columns of I_D are then filled in by again applying α to sample the pixel values in the original images at the corrected locations by

$$I_D(\mathbf{x}_D) = (1 - \alpha) \cdot I_L(\mathbf{x}^*_L) + \alpha \cdot I_R(\mathbf{x}^*_R). \tag{6.6}$$

Using these techniques introduced by Richardt et al. (2013) and Bertel et al. (2019), we are able to produce multiple panoramas with a range of ϕ values corresponding to varying levels of parallax.

Table 6.1 Spherical SfM processing times from several sequences, and devices. All sequences were captured at 30 FPS and successfully reconstructed with Spherical SfM.

Dataset	Camera	Resolution	Frames	Time (s)
Street (Ventura, 2016)	Sony a5100	1920 × 1080	435	256
Mountain (Richardt et al., 2013)	Canon S95	720 × 1280	230	177
Shrine	iPhone X	1080 × 1920	602	557
NaturePath	iPhone X	1080 × 1920	697	610
Campus	iPhone X	1080 × 1920	667	520
Courtyard360	iPhone X	1080 × 1920	690	505

6.4 Technical Evaluation

In our evaluation, we compared our spherical SfM pipeline to a typical incremental SfM pipeline for the purposes of reconstructing the circular videos that typically result from stereo panorama capture. We choose COLMAP for our comparison, as it is currently regarded as one of the best performing and most robust SfM systems (Schönberger and Frahm, 2016). We ran our video sequences through COLMAP using known camera intrinsics, sequential matching, and loop-closure. In many cases, COLMAP would fail to complete the reconstruction. Some of these cases produced partial maps before terminating early, and other cases failed to initialise the reconstruction. Table 6.1 summarises the sequences we tested.

Figure 6.6 shows an example of one of the partial reconstructions. It is reasonably clear to see that partial reconstruction from circular motion is possible in some cases, but we found that it is not robust under the COLMAP system, and that these partial reconstructions are not always achieved. In general, we found that COLMAP does not reliably reconstruct handheld circular motion sequences as needed for stereo panorama creation. Probably for this reason, Bertel et al. (2019) recommend first subsampling the input video, reconstructing this sparse set of views with COLMAP, and then registering the remaining views followed by a final bundle adjustment. However, as discussed in Section 6.3.1, selecting this subset of views requires manual intervention.

While most datasets we tested would fail to initialise or produce partial reconstructions with COLMAP, it was able to produce a full reconstruction of our Street sequence without needing to subsample the frames. A rendering of the point cloud and camera trajectory is shown in Figure 6.7. Though all frames were successfully registered,

Figure 6.6 A partial reconstruction output from COLMAP, alongside a corresponding frame from the circular video sequence (NaturePath sequence).

there are issues with the resulting reconstruction, which make it unsuitable for stereo panorama generation. Primarily, there are two large discontinuities in the computed camera trajectory. Specifically, the reconstruction was unable to accurately close the loop between the first and last frames, as well as a large pose difference between two neighbouring frames, 75 and 76 (see Figure 6.7, left). Additionally, the reconstruction took a total of 176 minutes to complete, compared to 256 seconds with spherical SfM. In Table 6.2, we present a survey of methods that are designed specifically for reconstructing stereo panoramas from circular motion video. Our method is flexible in that it only requires a single handheld camera instead of a stereo camera (Hedman and Kopf, 2018) or multi-camera rig (Anderson et al., 2016; Schroers et al., 2018). In comparison to Megastereo (Richardt et al., 2013), our reconstruction time is much faster (3-10 minutes compared to 2-3 hours).

6.4.1 Synthetic Testing

In addition to the evaluation of COLMAP, we also investigated how robust our stereo panorama pipeline is to deviations from the ideal circular trajectory. To do this, we conducted a synthetic test where we move a synthetic camera within a set of randomly generated 3-D points. We generate multiple input video sequences, in an initially circular trajectory, through increasingly elliptical trajectories. While an elliptical trajectory

Figure 6.7 Reconstruction results from COLMAP with the circular Street sequence (Ventura, 2016). We found two large discontinuities in the trajectory between frames 75 and 76, as well as frames 0 and 435, despite their similar perspectives. Left: the four frames of interest. Centre: rendering of the points and cameras (red) reconstructed by COLMAP (Schönberger and Frahm, 2016). Right: close-up of the trajectory, with labelled camera positions.

Table 6.2 Survey of stereo panorama stitching methods.

Method	Images	Time (min)	Hardware
MegaParallax (Bertel et al., 2019)	400	240	Single camera
Megastereo (Richardt et al., 2013)	100 - 300	120 - 180	Single camera
Jump (Anderson et al., 2016)	16	3 - 4	16-camera rig
Schroers et al. (Schroers et al., 2018)	16	20	16-camera rig
Instant3D (Hedman and Kopf, 2018)	20 - 200	0.5	Stereo camera
Ours	200 - 700	3 - 10	Single camera

is unnatural, we use this simply to model an arbitrary non-circular trajectory.

As casual users are likely to make other types of deviation from the ideal circular trajectory (e.g. with a viewing direction that does not align with the normal of the sphere), small deviations in the viewing angle of the camera can be corrected by our system via the plane estimation and image rotation step described in Section 6.3.2. However, the resulting camera poses from tracking these images would potentially result in cameras being positioned incorrectly. For this reason, we investigate what type of issues or artefacts can be seen in the panoramas when the actual camera movement is not circular.

To test this, a set of 3-D points are projected into the synthetic camera using a synthesised elliptical ground-truth pose, and their projections are used to initialise the LK tracking of our pipeline. We use

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{6.7}$$

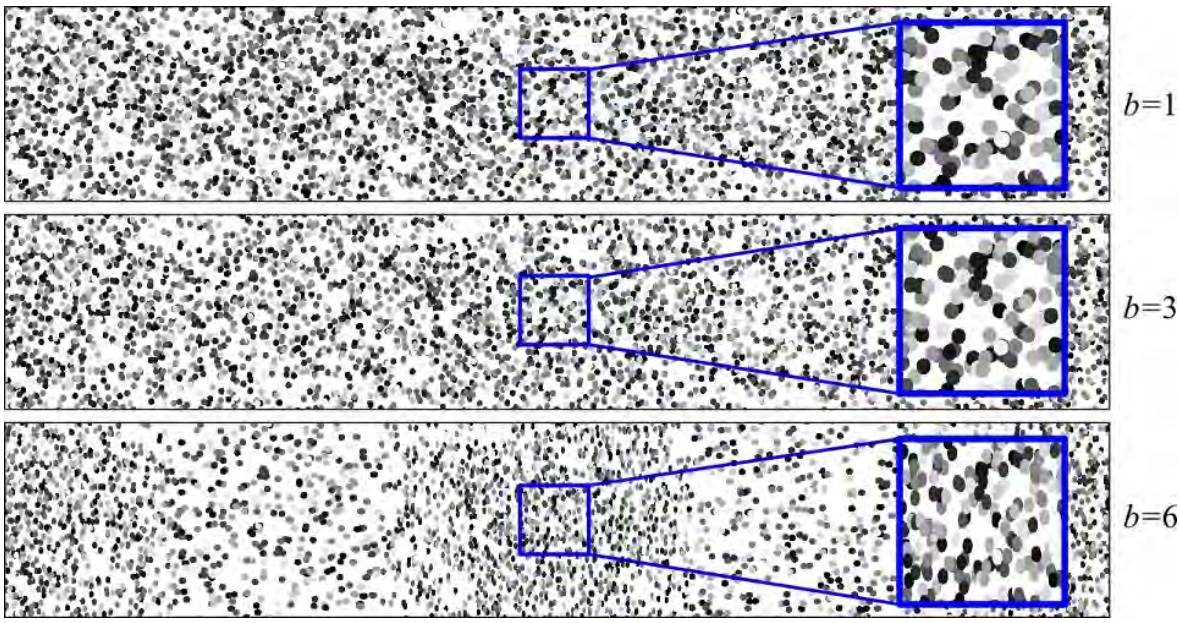

Figure 6.8 Synthetic panoramas produced by spherical tracking of elliptical movement through random points. With $b = 1$, the camera moves in a perfect circle, resulting in an even distribution of points. Our system could generate panoramas with large elliptical deviations; however, the originally circular points become distorted with the large values depicted with $b = 3$ and $b = 6$.

to represent the elliptical trajectory, and vary the parameter b ($1 \leq b \leq 6$) to scale the ellipse in one dimension, resulting in a range of elliptical trajectories of size $2 \times 2b$ units. The parameter a is fixed at 1. We also use the generated poses to render images of the projected points, which are loaded by the panorama generator. This allows us to visualise the effects of the elliptical deviation on the panoramas themselves.

We found that our pipeline was able to track through large elliptical deviations, as shown in Figure 6.8 for b values of 1, 3, and 6. The spherical constraint essentially forces the computed poses to be circular, resulting in artefacts such as distortion of the originally circular points in the output panoramas. We found in our later testing on real data from experts and non-experts (Section 6.5), that obvious instances of these types of artefacts were uncommon, and only occur in the synthetic tests when extreme elliptical deviations are made.

6.4.2 Stereo Panorama Results

Figure 6.9 shows example stereo panoramas produced using our method, rendered as red-cyan anaglyphs. We collected the videos with an iPhone, holding the phone

in an outstretched hand while spinning roughly in a circle. Each video is about 20 to 30 seconds in length. Figure 6.10 shows a visual comparison of our method with Megastereo. Both methods produce effective stereo panoramas, although Megastereo uses negative disparities for distant parts of the scene, so we have larger disparities for objects close to the viewer.

Although we do not have ground truth camera poses to compare against, these panoramas demonstrate that our pipeline is able to reconstruct the camera trajectories accurately enough in order to produce clean, well-stitched images. In addition, inspection of the apparent parallax shows that nearby objects indeed have greater disparity than distant objects throughout the panoramas.

In some sequences, we noticed visual artefacts such as seams near the loop closure point. These are caused by the camera deviating too far from the circular trajectory at the end of the sequence. For example, if the camera moves too far inward or outward, or vertically or horizontally, then the spherical motion assumption breaks and the estimated relative pose between the first and last cameras in the sequence is inaccurate. The flow-based blending helps to hide this inaccuracy and smoothly blend through it, but this is something that could be investigated in future work.

6.5 User Evaluation

In addition to the technical evaluation, we performed a user evaluation. There were two main goals for the user evaluation. The first goal was to create a dataset of videos captured by casual users to further evaluate our approach with non-expert users. We were also interested if there were any difference in results if we give the users no specific instructions (other than to capture a panorama on a mobile phone) or if we give them detailed instructions to follow in the capturing process. Our intention was to test whether untrained users capturing a panorama would naturally perform suitable spherical motion for reconstruction by our pipeline without needing an expert to instruct them about the requirements of our system.

The second goal was to test whether users would perceive a difference when viewing the resulting stereo panoramas in a VR headset compared to standard mono panoramas. Here we wanted to test if the panoramas created by our pipeline would appear convincing and natural to the participants, and how the parallax effect in the stereo panoramas would be perceived by participants. While there have been many studies investigating user perception of omnidirectional content, many of these focus on im-

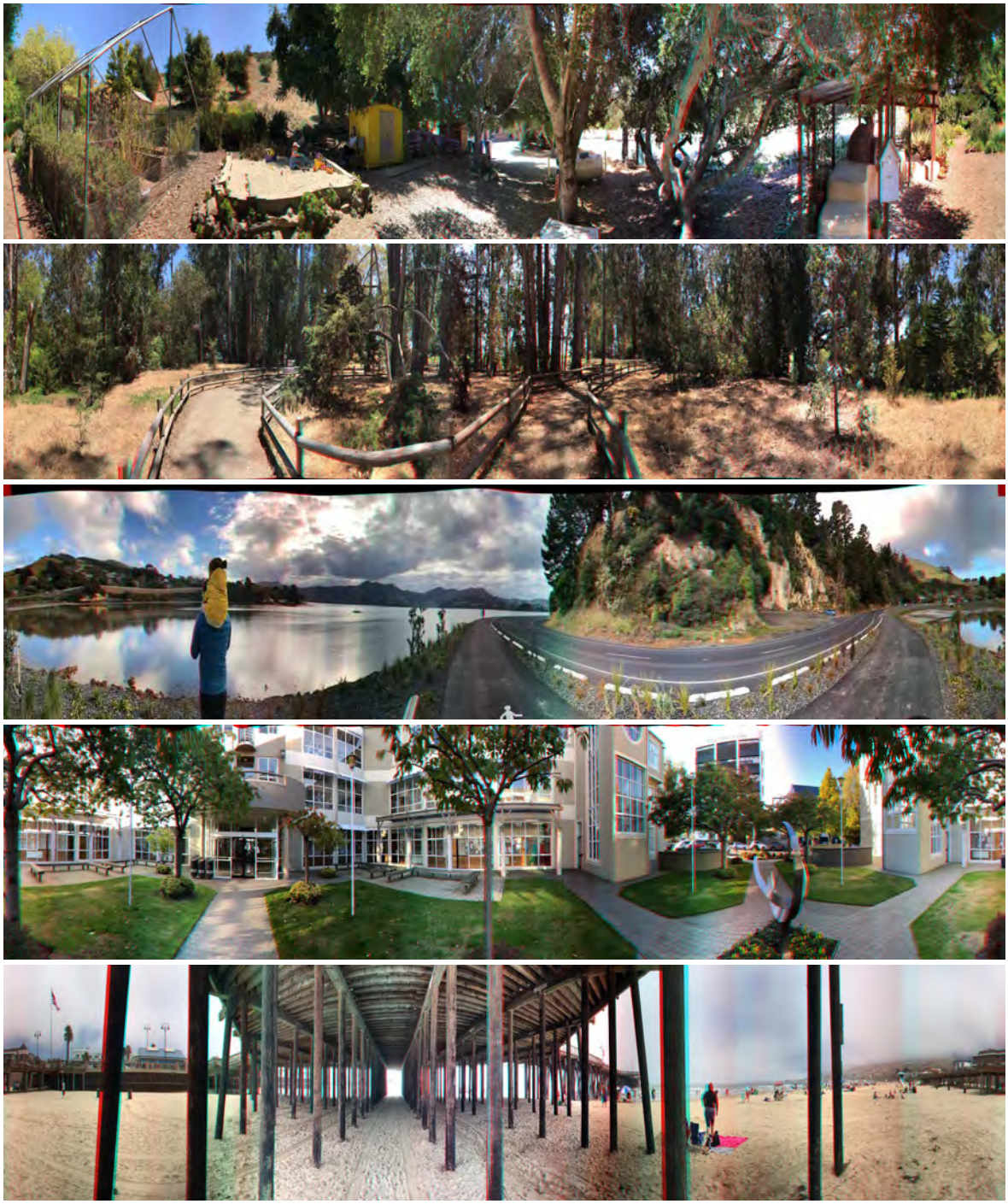

Figure 6.9 Sample stereo panoramas produced using our pipeline, rendered as red-cyan anaglyphs. Each input video was captured with a handheld iPhone, without the use of a tripod or any other device. From top to bottom: Garden, Grove, Bay, Courtyard360, Beach.

Figure 6.10 Anaglyph results compared to Megastereo (Richardt et al., 2013) with closeups for better visibility. We achieve visually similar results compared to Megastereo. Please note that the disparities are computed differently (Megastereo's disparities are adjusted in a way that close objects are roughly at the depth of the image plane and have a near-zero disparity. Far objects are set to be behind the image plane and have a positive disparity). To be viewed with red-cyan glasses.

age quality (Chen et al., 2019b; Upenik et al., 2016). Here, we focus on the user's immersion, and stereoscopic perception of depth which is a unique advantage of stereo panoramas compared to standard omnidirectional video. The study received ethical approval by the University of Otago Ethics committee (D19/161).

6.5.1 Study Design

Participants were asked to perform two tasks: (1) capturing a stereo panorama and (2) exploring a stereo panorama in a virtual reality headset. Our questionnaire is listed in Appendix B.

Task 1: Capture a 360-degree Video The first task was to capture a 360-degree video in portrait mode. There were two conditions for this task. In condition A: "Standard Panorama", the participants were simply asked to use a mobile phone to capture a video in portrait mode rotating around their own axis as if they would capture a panoramic image. In condition B: "Stereo Panorama", the participants were asked to use the mobile phone in portrait mode with one hand with the following instructions:

- Hold the phone away from your body with arm outstretched.
- Press the capture button with your other hand.
- Slowly turn around, trying to stay in the same position.
- Close the circle at 360 degrees.

After each condition, we asked the participants to answer questions with regards to the capturing process. The order of the conditions for the capture task were not randomised as we assume that most users have experience of capturing panoramas on a mobile phone (as confirmed in our demographics questionnaire). Furthermore, if we had provided participants with detailed capture instructions first (condition B), this may have affected how they would capture the standard panorama (condition A). Starting with the "Standard Panorama" condition also ensured that our detailed instructions would not influence their individual expectation of how a panorama should be captured.

Task 2: Viewing Panoramas in VR In the second task of the user evaluation, we let the participants experience a 360-degree rendering of a stereo panorama using a VR headset. The panorama shown in VR was captured by an expert user in the same location that the capture task was held, and the user's own data was processed later for separate evaluation. We used two conditions for the VR experience evaluation: condition A "Stereo": showing two stereoscopic renderings from our casual stereo panorama computation, and condition B "Mono": simply showing two views of the same panoramic image for both eyes. After each condit, we asked the participants to fill in a questionnaire about the VR experience.

In addition, the participants were asked to fill a paper-based demographics questionnaire. Overall, the study took around 25 minutes to complete. Participants were able to decide not to take part in the project without any disadvantage to themselves.

The demographic data collected includes age, gender, ethnicity, and vision impairments, as well as familiarity with similar systems and technologies. The remaining experimental data includes answers regarding the capturing process including Likert-scale number responses to questions regarding the usability of the capturing procedure. No personally identifiable data was collected beyond those included in the demographic questionnaire, and every effort is made to ensure that no data can be linked to any individual participant.

For the experiment, we used an iPhone X for the stereo panorama capture. For the VR experience, we used WebVR displayed on a mobile phone that was attached to a *Zeiss VR One* headset. An annotated screenshot from the VR viewing experience is shown in Figure 6.11.

Hypotheses We aimed to evaluate whether users would find the casual stereo panorama capture process similar to a standard panorama capture, as well as whether the visual quality of our results is sufficient to perceive a stereoscopic effect when viewing them on a VR headset. Thus, we had the following hypotheses:

- H1: Users will find the capturing process for capturing stereo panoramas similar in usability to standard panorama captures.
- H2: The quality of our casually captured stereo panoramas is good enough to convey a stereoscopic effect: Users will be more likely to perceive a stereoscopic effect when watching casual stereo panoramas in a VR headset compared to a standard mono panorama.

Figure 6.11 An annotated screenshot from the VR experience in the user study. The display on the mobile device was split to show different panoramas to each eye. Note the subtle disparity between left and right eyes shown by the nearby tree (condition A, "Stereo"). Typical panoramas would show no disparity here (condition B, "Mono"), resulting in a lesser perception of depth.

Participants We invited 12 participants that were all undergraduate and graduate students (3 female, and 9 male), with ages ranging from 21 to 31. All participants but one had either experience with the capture of panoramas (5) or knew about panorama capture (6). Approximately half of the participants came from a computer science background, while others came from various other disciplines. All participants were to read an information sheet prior, and given the opportunity to ask questions about the terminology used in the questionnaire.

6.5.2 CasualStereo Results

To compare both capturing methods, we used an excerpt from the SUS usability scale to capture usability aspects (Brooke, 1996). The results show that the participants judged both capturing options very similarly as shown in Figure 6.12. For both methods, participants showed a slight tendency towards agreeing with "use frequently" and "learn quickly", disagreeing with "unnecessarily complex" and being "cumbersome". We used the Wilcoxon signed-rank test in order to measure statistical differences with a typical significance threshold of $p = .05$, and did not find any significant differences. As a result we can confirm H1. In addition, we were interested if our method is robust enough to work with *casual* captures of stereo panoramas without specific instructions. We used the captured data from both conditions to test our method and achieve a similar success rate of over 90% for both conditions.

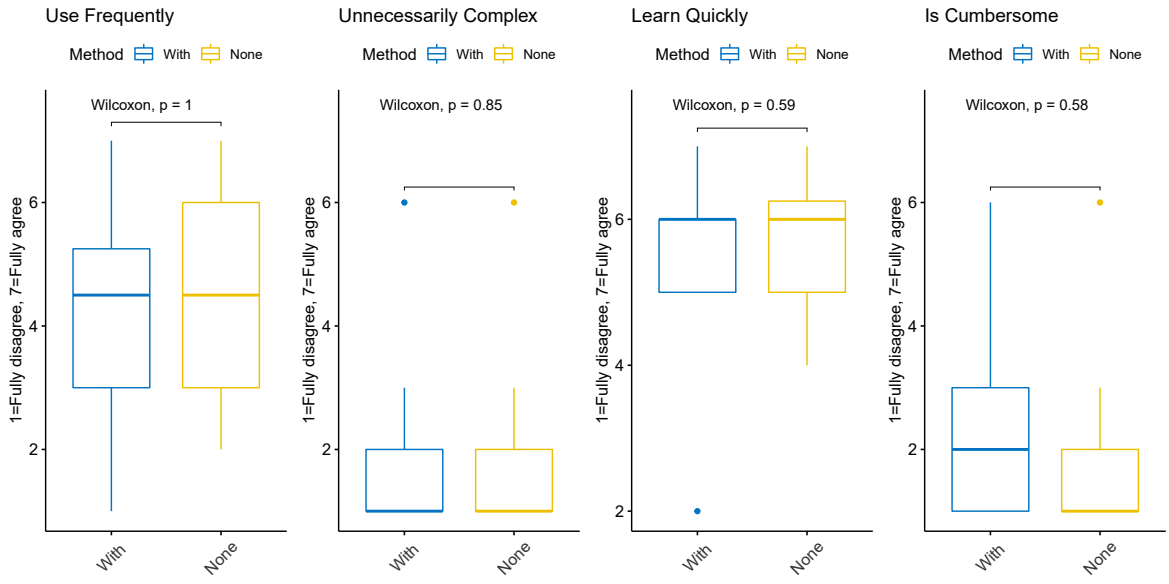

Figure 6.12 Results from the stereo panorama capture process questionnaire, comparing the experience with and without detailed capture instructions.

A Wilcoxon signed-rank test showed that there is a significant effect on the stereoscopic effect ($p = .031$) and descriptive statistics show that the participants are more likely to perceive a stereoscopic effect when being presented with the stereo panorama ($M = 5.58$, $SD = 0.996$) compared to the mono panorama ($M = 4.75$, $SD = 1.22$). This confirms hypothesis H2 (Figure 6.13, Stereoscopic Vision).

6.5.3 Discussion

The results of the user evaluation confirmed our hypotheses. For the capturing with and without instructions, we were able to confirm H1 and can assume that there is no difference in the usability of panorama capture for the users of our approach. It is also worth noting that the participants judged both capturing tasks rather positively, leaning towards "use frequently" and "learn quickly" and rather disagreeing on "unnecessarily complex" and "cumbersome".

The second aspect we were interested in with regards to the user capturing was if this data can be used to successfully create stereo panoramas with our method. 10 out of 12 people were able to capture videos for both conditions (with and without specific instructions) that could be used to successfully construct a stereo panorama. One sequence captured with condition A and one sequence captured with condition B could not be successfully used for creating a stereo panorama case. One failure case

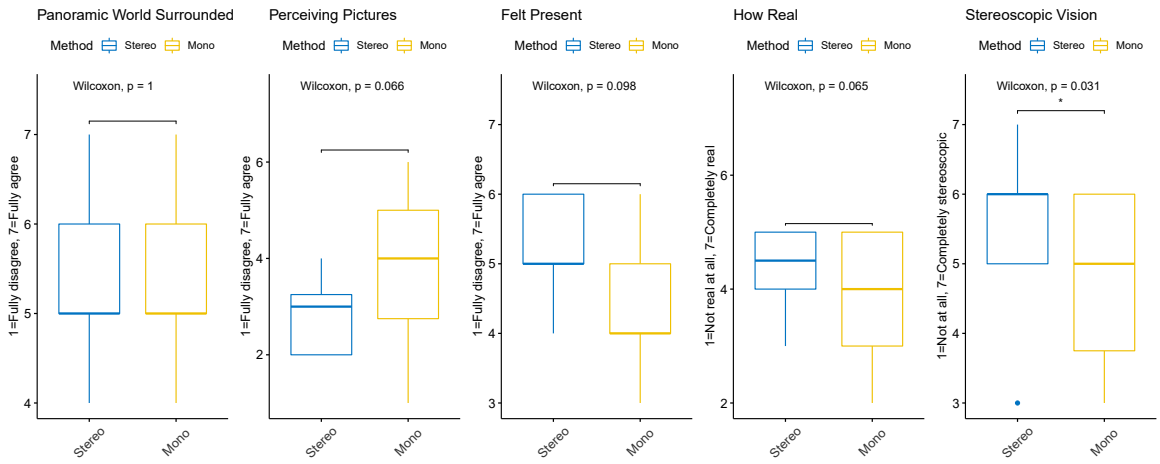

Figure 6.13 Results from the questionnaire comparing the users' experience of stereo and monocular panorama in a VR headset.

was caused by poor focus, and the other by failing to complete a full circle during the capture process. In summary, we experienced similar outputs for both capture conditions.

Some example panoramas as generated by the participants' captures are shown in Figure 6.14. As most panoramas were successfully reconstructed, we visually inspected each panorama for quality, and in most cases could not see any clear difference in quality between the two capture conditions. This would suggest that the detailed instructions we provided in condition B were roughly in line with how users would capture naturally with no instruction (condition A).

We were also able to confirm H2 by showing that there is a significant difference in terms of the perceived stereoscopic effect. Participants were more likely to experience a stereoscopic effect within the stereo panorama condition compared to mono. However, it is worth noting that even with the mono condition, the participants experienced a small stereoscopic effect (with a mean score of 4.75). Also, there was a higher variance in answers for the mono condition, suggesting participants were more indecisive.

6.6 Conclusion

While immersive viewing devices are becoming readily available, content creation for them remains a challenge. We have shown that spherical structure-from-motion can be used to enable robust, effective stereo panorama creation from casually captured video sequences. Our method does not require any specialised hardware – a single-camera

Figure 6.14 Results from processing the circular videos captured by participants of our user evaluation. Condition A represents no capture instruction, Condition B represents with instruction. Most user-captured panoramas were successfully reconstructed, regardless of the capturing condition.

mobile phone is all that is required. Compared to state-of-the-art methods for creating stereo panoramas from monocular video sequences, our method is substantially faster and the spherical constraint in the SfM component gives reliable reconstructions despite limited stereo baselines.

Future technical development could investigate relaxations of the spherical constraint in order to reduce the appearance of the loop-closure seams. While this constraint is key to creating reliable stereo panoramas with small baselines, it is not exactly met in real capture scenarios. In future work, the bundle adjustment phase (Section 6.3.1) could be extended to allow small deviations from this model.

Another direction for future work could be to incorporate a real-time tracking system into the capture process. Tracking here could provide live feedback to the users, guiding the trajectory to ensure a complete panorama. Such a system could be implemented on

a mobile device, or with panorama processing handled on a server to allow for robust capture and generation of stereo panoramas for casual usage.

Chapter 7

Conclusion

In this thesis, we addressed the problems of localisation and tracking of stationary users for Extended Reality applications. There are many XR applications that involve users performing mostly (but not purely) rotational movement in large open spaces. This is particularly prevalent in mobile AR applications, and panorama capture with mobile devices. We explored two examples of this type of application: the augmented reality spectator (ARSpectator), and stereo panorama capture. Through a review of the literature in Chapter 2, we found that traditional localisation and tracking approaches have not focused on these stationary application scenarios.

We saw that while some SLAM-based tracking methods can detect and exclude dynamic objects while tracking, many localisation approaches do not directly focus on dynamic environments. State-of-the-art localisation methods instead rely on outlier rejection, and robust feature matching to handle differences in appearance between the current view of an environment, and that encoded in a prior model.

Additionally, we saw that state-of-the-art tracking methods tended to group into two categories: unconstrained 6-DoF tracking, or rotation-only 3-DoF tracking. While some works aimed to bridge this gap by creating hybrid trackers, no approaches have addressed the problem by utilising a motion model that accurately represents the motion of a stationary mobile user. We saw that existing research in spherical SfM could be extended to support these XR applications for stationary users.

7.1 List of Contributions

The main contributions of this thesis are listed below:

- Captured unique video datasets that represent the motion of stationary users (for tracking and panorama generation), as well photo datasets of environments where these stationary use-cases can be applied (for localisation). We also computed reference poses for each of our camera images via reconstruction with SfM techniques.
- Developed an SfM adaptation of SoftPOSIT, and a method for evaluating localisation performance with respect to line-based CAD and SfM models, which had not previously been explored.
- Show that many state-of-the-art appearance-based localisation methods (Active Search, ESAC, and our own naive BoW approach) can be used effectively to localise in large stadium environments, while showing potential for effectively handling dynamics that are common in these scenarios.
- Developed a novel approach to SLAM that uses a spherical motion constraint, and show that this approach can track more robustly than state-of-the-art ORB-SLAM through evaluation on challenging synthetic and real datasets.
- Proposed a new method for managing SLAM keyframes and initialisation that can only be applied in spherically constrained tracking scenarios (keyframe sphere).
- Developed a pipeline for creating stereo panoramas using concepts from spherical SfM and flow-based blending, and show through a technical evaluation that it is faster and more robust to challenging circular motions than state-of-the-art SfM.
- Showed through a user study that the physical capture requirements of our system are no more difficult to use than natural panorama capturing, and that users can effectively perceive stereo effects when viewing our panoramas in virtual reality.

7.2 Summary of Findings

In this section, we summarise more specifically our results from data capture, and findings from the chapters in the topics of localisation and tracking in the context of the ARSpectator, and stereo panorama capture applications.

7.2.1 Data Capture

We saw that the standard datasets that are typically used for evaluating tracking and localisation methods are not suitable for our use-case of localising and tracking stationary AR users, nor panorama capture. To address this, we captured our own datasets including image datasets of two sports stadia for localisation, spherical spectator videos, and circular panoramic videos.

In addition to these datasets, we also performed synchronisation of the spherical videos to RealSense trajectories, and calibration of video datasets, all of which could support future work in this area. The large collections of SfM images were also processed by the state-of-the-art SfM pipeline COLMAP (Schönberger and Frahm, 2016) to create reconstructions of both stadia as well as reference pose data for each of the images.

7.2.2 Localisation

In the literature, we noted that most localisation approaches utilise robust descriptor matching to handle dynamics. In Chapter 4, we explored the feasibility of methods that do not rely on appearance, as no other recent research had focused on this. We noted that SfM models have become a very popular basis for localisation, and investigated the applicability of a featureless localisation method, SoftPOSIT (David et al., 2003), to localisation from these complex noisy models. While we did not find positive results with complex models, we saw promising results for simpler models which suggest its applicability to simple and controlled environments for which a noise-free prior model is available.

Our findings with SoftPOSIT reinforced the direction of state-of-the-art approaches, in particular that the benefits of appearance and descriptor-based localisation approaches outweigh their shortcomings in handling of dynamic elements. For this reason, we returned to these approaches and evaluated the state-of-the-art methods in the context of two sport stadia. Here, we found that the Active Search approach (Sattler et al., 2012a) performed well, as did ESAC (Brachmann and Rother, 2019). We conclude that ESAC may be better suited to server-based localisation with GPUs, whereas the Active Search approach has potential to be applied to localisation on mobile device hardware.

7.2.3 Tracking

Our review of the existing research on tracking highlighted a gap between fully unconstrained, and rotation-only tracking. To address this gap, we developed a novel tracking technique by extending spherical motion concepts introduced by Ventura (2016). We identified problems with state-of-the-art unconstrained SLAM systems with regard to stationary users, and discussed the shortcomings of rotation-only trackers. In Chapter 5, we showed that our spherically constrained SLAM system, SPLAT, was able to track more robustly than state-of-the-art ORB-SLAM (Mur-Artal et al., 2015) with improved tracking rates in a range of outdoor spherical video sequences.

We also note that previous research in stereo panorama generation found that general SfM pipelines struggle with the small-baseline circular videos that result from stationary users moving a camera in a circle (Bertel et al., 2019). In Chapter 6, we extended the spherical SfM concepts from Ventura (2016) specifically to handle circular videos reliably, and demonstrated its applicability to stereo panorama generation with the flow-based blending of Richardt et al. (2013) and Bertel et al. (2019). We showed that our approach was robust, and able to reconstruct videos captured by multiple real users via a user study of the capture process.

7.2.4 Extended Reality Applications

Finally, we tie together these results on localisation and tracking with their implications to XR applications. With regards to the ARSpectator application, we found that typical feature-based approaches showed the most promising results regarding localisation. In terms of tracking, we found that our SPLAT system can provide reliable tracking for the small translational movement of stationary users, while still performing well in smaller indoor environments. By testing this method on real spectator data in a stadium context, we showed its applicability to tracking in the ARSpectator context.

We also found positive results for applying the spherical constraint of Ventura (2016) to the stationary application of stereo panorama generation with flow-based blending (Richardt et al., 2013; Bertel et al., 2019). Through technical evaluation, we showed that the state-of-the-art general SfM system, COLMAP (Schönberger and Frahm, 2016), was not suitable for reconstructing from circular video sequences due to the small baselines. In our user study, we found that when users followed our specific capture instructions, the resulting capture experience was not significantly different from

how they would capture with no instructions. The user study also showed that users experienced a stronger stereoscopic effect when viewing our panoramas compared to a typical panorama in virtual reality.

7.3 Limitations and Future Work

Limitations While we have made some interesting contributions through the work in this thesis, there are of course limitations in some areas. On the topic of localisation, an area which we did not thoroughly explore is using a form of outlier rejection with our SoftPOSIT approach. We focused on reducing the complexity of the models to improve computation time, but another approach could be to select random 2-D and 3-D correspondences and use SoftPOSIT within a RANSAC scheme. This, however, is different from the way that RANSAC is typically applied (for example compared to Perspective-n-Point methods) as it is usually assumed that we have a set of mostly good matches to begin with, but this would be an interesting direction to explore.

A key limitation of our tracking approach is that while it can be used to accurately model spectator motion, people are much less predictable in real-life scenarios. Our method is not able to detect when the spherical motion assumption is violated, and therefore may continue tracking with erroneous results, or lose tracking in these situations. This could be addressed in future work by adding a model selection mechanism such as Gauglitz et al. (2012), which is able to switch between spherical and general tracking in real-time, depending on how the user moves.

With our stereo panorama pipeline, we were able to create high-quality panoramas, but in some scenarios, there would be visible seams at the loop closure point. While this is a small section of the panorama, this seam can be quite noticeable when viewed in VR headsets. These kinds of errors likely stem from users performing imperfect circular motion, so it would be interesting to see if spherical SfM could be used to first initialise a structure, and removing or relaxing the spherical constraint later in a final pass of pose optimisation with bundle adjustment.

Future Work As the work in this thesis covered multiple topics, there are numerous possibilities for future research. As we saw in Chapter 4, deep learning produced very promising results for localisation. It would be interesting to advance these techniques for the purposes of real-time tracking by accounting for temporal consistency with

recurrent architectures. While this may require GPU acceleration on mobile devices, this might be possible with state-of-the-art hardware.

The localisation results also suggested that ESAC might be very robust to dynamic elements in the environment. For this reason, it would be interesting to investigate using these kinds of approaches to instance segmentation of dynamic objects (Chen et al., 2019a). This could be used either to assist with localisation and tracking in dynamic spaces by removing problematic objects, or to extend our panorama generation pipeline by reducing unwanted artefacts caused by moving objects in the scene.

In our implementation of the SPLAT tracking system in Chapter 5, the main use of the keyframe sphere was directed toward simplifying the keyframe selection process. The known structure of the keyframe anchors could further be applied to reducing co-visibility complexity, as achieved with the essential graph in Mur-Artal et al. (2015). While the system we currently have is sufficient to demonstrate tracking robustness, this kind of adjustment and others may be essential for optimising the process for lower-spec mobile hardware.

One of our main motivating applications in this thesis was the ARSpectator application, so creating a complete localisation and tracking system that utilises a prior model of a stadium would be useful. Applying a promising localisation approach to the keyframes in our SPLAT system could be a way to achieve this, and could be extended further by creating a multi-user collaborative SLAM system (Zou and Tan, 2012). Such a collaborative system would support a live-updating map to track from by triangulating between the larger baselines between different spectators, and would further benefit the robustness of tracking in such challenging spaces. Furthermore, having a complete localisation and tracking system could support further research and user studies in this application.

In our stereo panorama method of Chapter 6, the pipeline is split into a capture and a processing phase. In our user study, verbal instructions were given to the users prior to the capture process. Realistically, panorama capture applications require a live visualisation that instructs the user to adjust angles, and report when they have completed the circle in real-time. Re-running a user-study with live feedback (such as from real-time tracking with SPLAT) could improve reconstruction results, as some of the failure cases were caused by users failing to capture a complete circle.

Appendix A

Acronyms

AR	Augmented Reality.
BoW	Bag of Words.
CAD	Computer Aided Design.
DoF	Degrees of Freedom.
ESAC	Expert Sample Consensus.
FPS	Frames Per Second.
GPS	Global Positioning System.
GPU	Graphics Processing Unit.
IMU	Inertial Measurement Unit.
PnP	Perspective-n-Point.
POSIT	Pose from Orthography and Scaling with Iterations.
RANSAC	Random Sample Consensus.
SfM	Structure from Motion.
SLAM	Simultaneous Localisation and Mapping.
SPLAT	Spherical Localisation and Tracking.
VR	Virtual Reality.
XR	Extended Reality.

Appendix B

User Study Documents

Reporting Sheet for use ONLY for proposals considered at departmental level

---

[Reference Number: *as allocated upon approval by the Human Ethics Committee*]
[*Date*]

Spherical Structure from Motion for Casual Capture of Stereo Panoramas
INFORMATION SHEET FOR PARTICIPANTS

Thank you for showing an interest in this project. Please read this information sheet carefully before deciding whether or not to participate. If you decide to participate we thank you. If you decide not to take part there will be no disadvantage to you and we thank you for considering our request.

What is the Aim of the Project?

Hand-held capture of stereo panoramas involves spinning the camera in a roughly circular path to acquire a large set of different views of the scene. For this purpose, image processing is used to compute where the camera is located and 3D structure of the scene. However, most existing methods for doing fail when working with such circular movements. In order to address this problem, we investigate a new image processing method that uses the assumption of a motion on a sphere. We will investigate if data captured by non-expert users fulfils our assumptions and can be used to compute stereo panoramas from a handheld camera. For this purpose, we will ask participants to capture stereo panoramas on a mobile phone.

What Types of Participants are being sought?

The participants for this study are recruited from undergraduate and graduate students at the University of Otago.

What will Participants be asked to do?

Should you agree to take part in this project, you will be asked to perform the following two tasks using a mobile phone. Both tasks involve capturing a panoramic video sequence with a mobile phone.

1. Task 1 ask you to capture a 360 degree video in portrait mode. There will be two conditions that we will explain more in detail during the study.
2. Task 2 ask you to experience a 360 degree VR rendering using a VR headset. We will show you two different version of this 360 degree rendering.

In addition, you will be asked to fill a paper-based demographics questionnaire and answer questions with regards to the capturing process and the 360 VR experience.

Overall, this study should take no longer than 15 minutes to complete.

Please be aware that you may decide not to take part in the project without any disadvantage to yourself.

What Data or Information will be collected and what use will be made of it?

There will be demographics data, data about usability, as well as the captured panoramic video sequence collected. The demographic data collected includes age, gender, ethnicity, and vision impairments, as well as familiarity with similar systems and technologies. The remaining experimental data includes answers regarding the capturing process and the VR experience including Likert-scale number responses to questions regarding the usability of the capturing procedure and the 360 VR experience.
No personally identifiable data will be collected beyond those included in the demographic questionnaire, and every effort will be made to ensure that no data can be linked to any individual participant.

The data collected will be securely stored in such a way that only those mentioned below will be able to gain access to it. Data obtained as a result of the research will be retained for **at least 5 years** in secure storage. Any personal information held on the participants may be destroyed at the completion of the research even though the data derived from the research will, in most cases, be kept for much longer or possibly indefinitely.

No material that could personally identify you will be used in any reports on this study. Results of this research may be published. The data from this project will be publicly archived so that it may be used by other researchers.

The results of the project may be published and will be available in the University of Otago Library (Dunedin, New Zealand) but every attempt will be made to preserve your anonymity.

Can Participants change their mind and withdraw from the project?

You may withdraw from participation in the project at any time and without any disadvantage to yourself.

What if Participants have any Questions?
If you have any questions about our project, either now or in the future, please feel free to contact either:-

Lewis Baker
Department of Computer Science
+64 3 479 8587
lewis.baker@otago.ac.nz

and

Stefanie Zollmann
Department of Computer Science
+64 3 479 8587
stefanie.zollmann@otago.ac.nz

This study has been approved by the Department stated above. However, if you have any concerns about the ethical conduct of the research you may contact the University of Otago Human Ethics Committee through the Human Ethics Committee Administrator (ph +643 479 8256 or email gary.witte@otago.ac.nz). Any issues you raise will be treated in confidence and investigated and you will be informed of the outcome.

Reporting Sheet for use ONLY for proposals considered at departmental level

---

Spherical Structure from Motion for Casual Capture of Stereo Panoramas
CONSENT FORM FOR
PARTICIPANTS

I have read the Information Sheet concerning this project and understand what it is about. All my questions have been answered to my satisfaction. I understand that I am free to request further information at any stage.
I know that:-

1. My participation in the project is entirely voluntary;

2. I am free to withdraw from the project at any time without any disadvantage;

3. Personal identifying information will be destroyed at the conclusion of the project but any raw data on which the results of the project depend will be retained in secure storage for at least five years;

4. The results of the project may be published and will be available in the University of Otago Library (Dunedin, New Zealand) but every attempt will be made to preserve my anonymity.

I agree to take part in this project.

....
(Signature of participant) (Date)

.....
(Printed Name)

[8. I, as the participant: a) agree to being named in the research, ☐ OR;

b) would rather remain anonymous.] ☐

Date: *Participant #:*

Casual Capturing of Stereo Panoramas

DEMOGRAPHICS QUESTIONNAIRE

Please provide the following information:

- Age: ……
- Gender:

  ☐ Male ☐ Female ☐ Diverse
- Ethnicity:

  ☐ NZ Maori ☐ NZ European ☐ US ☐ Pacific ☐ Chinese ☐ Indian ☐ Other ……
- Do you have *normal* or *corrected to normal* (e.g. prescription glasses or contacts) vision?

  ☐ Yes ☐ No
- Do you have *mobility restrictions*?

  ☐ Yes, the following …… …….. ☐ No
- Do you have any experience with capturing (stereo) panoramas?

  ☐ Yes.

  ☐ No, but I know what it is all about. ☐ No

Date: *Participant #:*

Casual Capturing of Stereo Panoramas

CAPTURING EXPERIENCE QUESTIONNAIRE

Condition A

C1. I think that I would like to use the stereo panorama capturing frequently.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C2. I found the stereo panorama capturing unnecessarily complex.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C3. I would imagine that most people would learn to use the stereo panorama capture quickly.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C4. I found the stereo panorama capture very cumbersome/awkward to use.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

Date: *Participant #:*

Casual Capturing of Stereo Panoramas

CAPTURING EXPERIENCE QUESTIONNAIRE

Condition B

C1. I think that I would like to use the stereo panorama capturing frequently.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C2. I found the stereo panorama capturing unnecessarily complex.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C3. I would imagine that most people would learn to use the stereo panorama capture quickly.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

C4. I found the stereo panorama capture very cumbersome/awkward to use.

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Very Much

Date: *Participant #:*

Casual Capturing of Stereo Panoramas

VR EXPERIENCE QUESTIONNAIRE

Condition A

V1. Somehow I felt that the panoramic world surrounded me.

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V2. I felt like I was just perceiving pictures

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V3. I felt present in the panoramic space.

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V4. How real did the mixed reality world seem to you?

	1	2	3	4	5	6	7	
Not real at all	◯	◯	◯	◯	◯	◯	◯	Completely real

V5. I felt stereoscopic vision .

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Completely stereoscopic

Date: *Participant #:*

Casual Capturing of Stereo Panoramas

VR EXPERIENCE QUESTIONNAIRE

Condition B

V1. Somehow I felt that the panoramic world surrounded me.

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V2. I felt like I was just perceiving pictures

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V3. I felt present in the panoramic space.

	1	2	3	4	5	6	7	
Fully disagree	◯	◯	◯	◯	◯	◯	◯	Fully agree

V4. How real did the mixed reality world seem to you?

	1	2	3	4	5	6	7	
Not real at all	◯	◯	◯	◯	◯	◯	◯	Completely real

V5. I felt stereoscopic vision .

	1	2	3	4	5	6	7	
Not at all	◯	◯	◯	◯	◯	◯	◯	Completely stereoscopic

Bibliography

Agarwal, S., Mierle, K., et al. (2012). Ceres solver. http://ceres-solver.org/. [Software; optimisation framework].

Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building Rome in a day. In *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79. IEEE.

Anderson, R., Gallup, D., Barron, J. T., Kontkanen, J., Snavely, N., Hernández, C., Agarwal, S., and Seitz, S. M. (2016). Jump: virtual reality video. *ACM Transactions on Graphics (TOG)*, 35(6):1–13.

Animation Research Ltd. (2020). Virtual Eye. https://virtualeye.tv/. [Hardware; Software; sport visualisation system].

Apple, Inc. (2018). ARKit. https://developer.apple.com/documentation/arkit. [Software; augmented reality tracking API].

Arth, C., Wagner, D., Klopschitz, M., Irschara, A., and Schmalstieg, D. (2009). Wide area localization on mobile phones. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 73–82. IEEE.

Azuma, R. (1993). Tracking requirements for augmented reality. *Communications of the ACM*, 36(7):50–51.

Azuma, R. (1997). A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4):355–385.

Azuma, R. and Bishop, G. (1994). Improving static and dynamic registration in an optical see-through HMD. In *Proceedings of the 21st annual Conference on Computer Graphics and Interactive Techniques*, pages 197–204.

Baker, L., Mills, S., Zollmann, S., and Ventura, J. (2020a). CasualStereo: Casual capture of stereo panoramas with spherical structure-from-motion. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 782–790. IEEE.

Baker, L., Ventura, J., Zollmann, S., Mills, S., and Langlotz, T. (2020b). SPLAT: Spherical localization and tracking in large spaces. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 809–817. IEEE.

Baker, L., Zollmann, S., Mills, S., and Langlotz, T. (2018). SoftPOSIT for augmented reality in complex environments: Limitations and challenges. In *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE.

Baker, L., Zollmann, S., and Ventura, J. (2019). Spherical structure-from-motion for casual capture of stereo panoramas. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 846–847. IEEE.

Bertel, T., Campbell, N. D., and Richardt, C. (2019). MegaParallax: Casual 360 panoramas with motion parallax. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):1828–1835.

Bertel, T. and Richardt, C. (2018). MegaParallax: 360° panoramas with motion parallax. In *ACM SIGGRAPH 2018 Posters*, SIGGRAPH '18. Association for Computing Machinery, New York, NY, USA.

Besl, P. J. and McKay, N. D. (1992). Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics.

Billinghurst, M., Kato, H., and Poupyrev, I. (2001). The magicbook-moving seamlessly between reality and virtuality. *IEEE Computer Graphics and Applications*, 21(3):6–8.

Blender Foundation (2016). Blender. https://www.blender.org/. [Version 2.77a. Online; accessed 05-April-2016].

Bloesch, M., Czarnowski, J., Clark, R., Leutenegger, S., and Davison, A. J. (2018). CodeSLAM—learning a compact, optimisable representation for dense visual SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2560–2568.

Bonetti, F., Warnaby, G., and Quinn, L. (2018). Augmented reality and virtual reality in physical and online retailing: A review, synthesis and research agenda. In *Augmented Reality and Virtual Reality*, pages 119–132. Springer.

Bouguet, J.-Y. et al. (2001). Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4.

Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., and Rother, C. (2017). DSAC-differentiable RANSAC for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692.

Brachmann, E., Michel, F., Krull, A., Ying Yang, M., Gumhold, S., et al. (2016). Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3364–3372.

Brachmann, E. and Rother, C. (2018). Learning less is more-6D camera localization via 3D surface regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4654–4662.

Brachmann, E. and Rother, C. (2019). Expert sample consensus applied to camera re-localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7525–7534.

Bradski, G. (2000). The OpenCV library. *Dr Dobb's Journal of Software Tools*, 25:120–125.

Brooke, J. (1996). SUS: a quick and dirty usability scale. *Usability Evaluation in Industry*, page 189.

Brown, M., Hartley, R. I., and Nistér, D. (2007). Minimal solutions for panoramic stitching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision*, pages 25–36. Springer.

Broxton, M., Flynn, J., Overbeck, R., Erickson, D., Hedman, P., DuVall, M., Dourgarian, J., Busch, J., Whalen, M., and Debevec, P. (2020). Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 39(4):86:1–86:15.

Brückner, M., Bajramovic, F., and Denzler, J. (2008). Experimental evaluation of relative pose estimation algorithms. In *2008 International Conference on Computer Vision Theory and Applications*, pages 431–438. Springer.

Buehler, C., Bosse, M., McMillan, L., Gortler, S., and Cohen, M. (2001). Unstructured lumigraph rendering. In *Proceedings of the 28th annual Conference on Computer Graphics and Interactive Techniques*, pages 425–432.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W., and Siegwart, R. (2016). The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163.

Campbell, D., Petersson, L., Kneip, L., and Li, H. (2017). Globally-optimal inlier set maximisation for simultaneous camera pose and feature correspondence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–10.

Camposeco, F., Cohen, A., Pollefeys, M., and Sattler, T. (2018). Hybrid camera pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 136–144.

Chatterjee, A. and Madhav Govindu, V. (2013). Efficient and robust large-scale rotation averaging. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 521–528.

Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., et al. (2019a). Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4974–4983.

Chen, M., Jin, Y., Goodall, T., Yu, X., and Bovik, A. C. (2019b). Study of 3D virtual reality picture quality. *IEEE Journal of Selected Topics in Signal Processing*, 14(1):89–102.

ChyronHego (2019). FRESH. https://chyronhego.com/products/in-studio-and-virtual-graphics/fresh-vsar/. [Hardware; tracked studio camera system].

Cummins, M. and Newman, P. (2008). FAB-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665.

David, P., DeMenthon, D., Duraiswami, R., and Samet, H. (2002a). Evaluation of the SoftPOSIT model-to-image registration algorithm. *Technical Reports from UMIACS, UMIACS-TR-2002-22*.

David, P., DeMenthon, D., Duraiswami, R., and Samet, H. (2002b). SoftPOSIT: Simultaneous pose and correspondence determination. In *European Conference on Computer Vision*, pages 698–714. Springer.

David, P., DeMenthon, D., Duraiswami, R., and Samet, H. (2003). Simultaneous pose and correspondence determination using line features. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages 1–8. IEEE.

Davis, A., Levoy, M., and Durand, F. (2012). Unstructured light fields. In *Computer Graphics Forum*, volume 31, pages 305–314. Wiley Online Library.

Davison, A. J. (2003). Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the Ninth IEEE International Conference on Computer Vision-Volume 2*, page 1403.

Davison, A. J. and Murray, D. W. (2002). Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865–880.

Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Aachine Intelligence*, 29(6):1052–1067.

Debevec, P., Yu, Y., and Borshukov, G. (1998). Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Workshop on Rendering Techniques*, pages 105–116. Springer.

DeMenthon, D. F. and Davis, L. S. (1995). Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1-2):123–141.

Diaz, J. and Abderrahim, M. (2007). Modified SoftPOSIT algorithm for 3D visual tracking. *2007 IEEE International Symposium on Intelligent Signal Processing*, pages 1–6.

Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F., and Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241.

DiVerdi, S., Wither, J., and Hollerer, T. (2008). Envisor: Online environment map construction for mixed reality. In *2008 IEEE Virtual Reality Conference*, pages 19–26. IEEE.

Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625.

Engel, J., Schöps, T., and Cremers, D. (2014). LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision*, pages 834–849. Springer.

Engel, J., Stückler, J., and Cremers, D. (2015). Large-scale direct SLAM with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942. IEEE.

Engels, C., Stewénius, H., and Nistér, D. (2006). Bundle adjustment rules. *Photogrammetric Computer Vision*, 2(32).

Facebook, Inc. (2016). Surround360. https://github.com/facebook/Surround360. [Software; 360 video capture framework].

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Flynn, J., Broxton, M., Debevec, P., DuVall, M., Fyffe, G., Overbeck, R., Snavely, N., and Tucker, R. (2019). Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2367–2376.

Flynn, J., Neulander, I., Philbin, J., and Snavely, N. (2016). Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5515–5524.

Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE.

Forster, C., Zhang, Z., Gassner, M., Werlberger, M., and Scaramuzza, D. (2016). SVO: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265.

Gálvez-López, D. and Tardós, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197.

Gao, X., Wang, R., Demmel, N., and Cremers, D. (2018). LDSO: Direct sparse odometry with loop closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2198–2204. IEEE.

Gao, X.-S., Hou, X.-R., Tang, J., and Cheng, H.-F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943.

Gauglitz, S., Sweeney, C., Ventura, J., Turk, M., and Höllerer, T. (2012). Live tracking and mapping from both general and rotation-only camera motion. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 13–22. IEEE.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

Glocker, B., Izadi, S., Shotton, J., and Criminisi, A. (2013). Real-time rgb-d camera relocalization. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 173–179. IEEE.

Google LLC (2018). ARCore. https://developers.google.com/ar. [Software; augmented reality tracking API].

Grubert, J., Langlotz, T., and Grasset, R. (2011). Augmented reality browser survey. *Institute for Computer Graphics and Vision, University of Technology Graz, technical report*, 1101:37.

Grupp, M. (2017). evo: Python package for the evaluation of odometry and SLAM. https://github.com/MichaelGrupp/evo. [Software; odometry benchmarking tools].

Guéziec, A. (2002). Tracking pitches for broadcast television. *Computer*, 35(3):38–43.

Ha, H., Im, S., Park, J., Jeon, H.-G., and So Kweon, I. (2016). High-quality depth from uncalibrated small motion clip. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5413–5421.

Hadian, M. and Kasaei, S. (2015). Fast homography refinement in soccer videos. In *2015 9th Iranian Conference on Machine Vision and Image Processing (MVIP)*, pages 185–188. IEEE.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge University Press.

Hartley, R. I. (1997). In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593.

Hartley, R. I. and Sturm, P. (1997). Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157.

Hedman, P., Alsisan, S., Szeliski, R., and Kopf, J. (2017). Casual 3D photography. *ACM Transactions on Graphics (TOG)*, 36(6):1–15.

Hedman, P. and Kopf, J. (2018). Instant 3D photography. *ACM Transactions on Graphics (TOG)*, 37(4):1–12.

Henrysson, A. and Ollila, M. (2004). UMAR: Ubiquitous mobile augmented reality. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia*, pages 41–45.

Hirschmuller, H. (2005). Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814. IEEE.

Hirschmuller, H. (2007). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341.

Hofer, M., Maurer, M., and Bischof, H. (2017). Efficient 3D scene abstraction using line segments. *Computer Vision and Image Understanding*, 157:167–178.

Höllerer, T. and Feiner, S. (2004). Mobile augmented reality. *Telegeoinformatics: Location-based Computing and Services*, 21.

Horn, B. K. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642.

HTC Corporation (2015). HTC Vive. https://www.vive.com/. [Hardware; virtual reality headset].

Huang, J., Chen, Z., Ceylan, D., and Jin, H. (2017). 6-DOF VR videos with a single 360-camera. In *2017 IEEE Virtual Reality (VR)*, pages 37–44. IEEE.

Huynh, D.-N. T., Raveendran, K., Xu, Y., Spreen, K., and MacIntyre, B. (2009). Art of defense: a collaborative handheld augmented reality board game. In *Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games*, pages 135–142.

Im, S., Ha, H., Choe, G., Jeon, H.-G., Joo, K., and So Kweon, I. (2015). High quality structure from small motion for rolling shutter cameras. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 837–845.

Im, S., Ha, H., Rameau, F., Jeon, H.-G., Choe, G., and Kweon, I. S. (2016). All-around depth from small motion with a spherical panoramic camera. In *European Conference on Computer Vision*, pages 156–172. Springer.

Intel Corporation (2019). Intel RealSense LiDAR camera L515. https://www.intelrealsense.com/lidar-camera-l515/. [Hardware; LiDAR camera].

Kalantari, N. K., Wang, T.-C., and Ramamoorthi, R. (2016). Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10.

Karami, E., Prasad, S., and Shehata, M. (2017). Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images. *arXiv preprint arXiv:1710.02726*.

Kavanagh, S., Luxton-Reilly, A., Wünsche, B., and Plimmer, B. (2016). Creating 360 educational video: a case study. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction*, pages 34–39.

Kazhdan, M. and Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13.

Kendall, A., Grimes, M., and Cipolla, R. (2015). PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2938–2946.

Khan, N., McCane, B., and Mills, S. (2015). Better than SIFT? *Machine Vision and Applications*, 26(6):819–836.

Khan, N. Y. and McCane, B. (2012). Smartphone application for indoor scene localization. In *Proceedings of the 14th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 201–202.

Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Niessner, M., Pérez, P., Richardt, C., Zollhöfer, M., and Theobalt, C. (2018). Deep video portraits. *ACM Transactions on Graphics (TOG)*, 37(4):1–14.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234. IEEE.

Knight, J., Davison, A., and Reid, I. (2001). Towards constant time SLAM using postponement. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, volume 1, pages 405–413. IEEE.

Koulieris, G. A., Akşit, K., Stengel, M., Mantiuk, R. K., Mania, K., and Richardt, C. (2019). Near-eye display and tracking technologies for virtual and augmented reality. In *Computer Graphics Forum*, volume 38, pages 493–519. Wiley Online Library.

Kurz, D., Meier, P. G., Plopski, A., and Klinker, G. (2013). An outdoor ground truth evaluation dataset for sensor-aided visual handheld camera localization. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 263–264. IEEE.

Langlotz, T., Degendorfer, C., Mulloni, A., Schall, G., Reitmayr, G., and Schmalstieg, D. (2011). Robust detection and tracking of annotations for outdoor augmented reality browsing. *Computers & graphics*, 35(4):831–840.

Langlotz, T., Grubert, J., and Grasset, R. (2013). Augmented reality browsers: essential products or only gadgets? *Communications of the ACM*, 56(11):34–36.

Langlotz, T., Nguyen, T., Schmalstieg, D., and Grasset, R. (2014). Next-generation augmented reality browsers: rich, seamless, and adaptive. *Proceedings of the IEEE*, 102(2):155–169.

Langlotz, T., Wagner, D., Mulloni, A., and Schmalstieg, D. (2010). Online creation of panoramic augmented reality annotations on mobile phones. *IEEE Pervasive Computing*, 11(2):56–63.

Lee, S. H. and Civera, J. (2018). Loosely-coupled semi-direct monocular SLAM. *IEEE Robotics and Automation Letters*, 4(2):399–406.

Li, P., Qin, T., Hu, B., Zhu, F., and Shen, S. (2017). Monocular visual-inertial state estimation for mobile augmented reality. In *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 11–21. IEEE.

Li, Y., Shum, H.-Y., Tang, C.-K., and Szeliski, R. (2004). Stereo reconstruction from multiperspective panoramas. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):45–62.

Li, Y., Snavely, N., Huttenlocher, D., and Fua, P. (2012). Worldwide pose estimation using 3D point clouds. In *European Conference on Computer Vision*, pages 15–29. Springer.

Li, Y., Snavely, N., and Huttenlocher, D. P. (2010). Location recognition using prioritized feature matching. In *European Conference on Computer Vision*, pages 791–804. Springer.

Liu, H., Zhang, G., and Bao, H. (2016). Robust keyframe-based monocular SLAM for augmented reality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 1–10. IEEE.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision.

Luo, B., Xu, F., Richardt, C., and Yong, J.-H. (2018). Parallax360: stereoscopic 360 scene representation for head-motion parallax. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1545–1553.

Magic Leap, Inc. (2018). Magic Leap One. https://www.magicleap.com/en-us/. [Hardware; augmented reality glasses].

Mann, S., Furness, T., Yuan, Y., Iorio, J., and Wang, Z. (2018). All reality: Virtual, augmented, mixed (X), mediated (X, Y), and multimediated reality. *arXiv preprint arXiv:1804.08386*.

McMillan, L. and Bishop, G. (1995). Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22nd annual Conference on Computer Graphics and Interactive Techniques*, pages 39–46.

Microsoft Corporation (2010). Kinect. https://developer.microsoft.com/en-us/windows/kinect. [Hardware; depth camera].

Microsoft Corporation (2016). Microsoft HoloLens. https://www.microsoft.com/en-us/hololens/. [Hardware; augmented reality glasses].

Milletari, F., Navab, N., and Ahmadi, S.-A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571. IEEE.

Mills, S. (2018). Four-and seven-point relative camera pose from oriented features. In *2018 International Conference on 3D Vision (3DV)*, pages 218–227. IEEE.

Mo-Sys (2017). StarTracker. https://www.mo-sys.com/products/. [Hardware; tracked studio camera system].

Mohring, M., Lessig, C., and Bimber, O. (2004). Video see-through AR on consumer cell-phones. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 252–253. IEEE.

Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al. (2002). FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 593–598. AAAI.

Montiel, J. M. and Davison, A. J. (2006). A visual compass based on SLAM. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1917–1922. IEEE.

Moore, A., Daniel, B., Leonard, G., Regenbrecht, H., Rodda, J., Baker, L., Ryan, R., and Mills, S. (2020). Comparative usability of an augmented reality sandtable and 3D GIS for education. *International Journal of Geographical Information Science*, 34(2):229–250.

Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *2009 International Conference on Computer Vision Theory and Applications*, pages 331–340. Springer.

Müller, J., Langlotz, T., and Regenbrecht, H. (2016). PanoVC: Pervasive telepresence using mobile phones. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE.

Muñoz-Salinas, R. and Medina-Carnicer, R. (2020). UcoSLAM: Simultaneous localization and mapping by fusion of keypoints and squared planar markers. *Pattern Recognition*, 101:107193.

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163.

Mur-Artal, R. and Tardós, J. D. (2017a). ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262.

Mur-Artal, R. and Tardós, J. D. (2017b). Visual-inertial monocular SLAM with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803.

Narzt, W., Pomberger, G., Ferscha, A., Kolb, D., Müller, R., Wieghardt, J., Hörtner, H., and Lindinger, C. (2006). Augmented reality navigation systems. *Universal Access in the Information Society*, 4(3):177–187.

Narzt, W., Pomberger, G., Ferscha, A., Kolb, D., Reiner, M., Wieghardt, J., Horst, H., Lindinger, C., et al. (2003). Pervasive information acquisition for mobile AR-navigation systems. In *Proceedings Fifth IEEE Workshop on Mobile Computing Systems and Applications*, page 13. IEEE.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011). DTAM: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327. IEEE.

Nguyen, H. M., Wünsche, B. C., Delmas, P., Lutteroth, C., and van der Mark, W. (2013). High resolution 3d content creation using unconstrained and uncalibrated cameras. In *2013 6th International Conference on Human System Interactions (HSI)*, pages 637–644. IEEE.

Nguyen, H. M., Wünsche, B. C., Delmas, P., Lutteroth, C., and van der Mark, W. (2014). A robust system for high-quality reconstruction of 3d objects from photographs. In *Issues and Challenges in Artificial Intelligence*, pages 3–15. Springer.

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770.

Nistér, D. (2005). Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329.

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168. IEEE.

Oberkampf, D., DeMenthon, D. F., and Davis, L. S. (1996). Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511.

Oculus VR (2016). Oculus Rift CV 1. https://www.oculus.com/. [Hardware; virtual reality headset].

Oculus VR (2019). Oculus Quest. https://www.oculus.com/quest/. [Hardware; virtual reality headset].

Pagani, A. and Stricker, D. (2011). Structure from motion using full spherical panoramic cameras. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 375–382. IEEE.

Paz, L. M., Piniés, P., Tardós, J. D., and Neira, J. (2008). Large-scale 6-DOF SLAM with stereo-in-hand. *IEEE Transactions on Robotics*, 24(5):946–957.

Peleg, S., Ben-Ezra, M., and Pritch, Y. (2001). Omnistereo: Panoramic stereo imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):279–290.

Piekarski, W. and Thomas, B. (2002). ARQuake: the outdoor augmented reality gaming system. *Communications of the ACM*, 45(1):36–38.

Pirchheim, C., Schmalstieg, D., and Reitmayr, G. (2013). Handling pure camera rotation in keyframe-based SLAM. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 229–238. IEEE.

Prince, S. J., Xu, K., and Cheok, A. D. (2002). Augmented reality camera tracking with homographies. *IEEE Computer Graphics and Applications*, 22(6):39–45.

PTC, Inc. (2015). Vuforia. https://developer.vuforia.com/. [Online; accessed 31-July-2020].

Qian, F., Ji, L., Han, B., and Gopalakrishnan, V. (2016). Optimizing 360 video delivery over cellular networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 1–6.

Qin, T., Li, P., and Shen, S. (2018). VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020.

Quilón, D., Mohedano, R., Cuevas, C., and García, N. (2015). Unsupervised high-quality soccer field segmentation. In *2015 International Symposium on Consumer Electronics (ISCE)*, pages 1–2. IEEE.

Rangarajan, A., I, H. C., and Bookstein, F. L. (1997). The softassign procrustes matching algorithm. *Lecture Notes in Computer Science*, 1230:29–42.

Reitmayr, G. and Drummond, T. W. (2006). Going out: robust model-based tracking for outdoor augmented reality. In *2006 IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 109–118. IEEE.

Richardt, C., Pritch, Y., Zimmer, H., and Sorkine-Hornung, A. (2013). Megastereo: Constructing high-resolution stereo panoramas. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1256–1263.

Richardt, C., Tompkin, J., and Wetzstein, G. (2020). Capture, reconstruction, and representation of the visual real world for virtual reality. In *Real VR–Immersive Digital Reality*, pages 3–32. Springer.

Rolland, J. P. and Fuchs, H. (2000). Optical versus video see-through head-mounted displays in medical visualization. *Presence: Teleoperators & Virtual Environments*, 9(3):287–309.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571. IEEE.

Saff, E. B. and Kuijlaars, A. B. (1997). Distributing many points on a sphere. *The Mathematical Intelligencer*, 19(1):5–11.

Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., and Davison, A. J. (2013). SLAM++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1352–1359.

Sattler, T., Leibe, B., and Kobbelt, L. (2011). Fast image-based localization using direct 2D-to-3D matching. In *2011 International Conference on Computer Vision*, pages 667–674. IEEE.

Sattler, T., Leibe, B., and Kobbelt, L. (2012a). Improving image-based localization by active correspondence search. In *European Conference on Computer Vision*, pages 752–765. Springer.

Sattler, T., Leibe, B., and Kobbelt, L. (2016). Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1744–1756.

Sattler, T., Maddern, W., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., et al. (2018). Benchmarking 6DOF outdoor visual localization in changing conditions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8601–8610.

Sattler, T., Weyand, T., Leibe, B., and Kobbelt, L. (2012b). Image retrieval for image-based localization revisited. In *Proceedings of the British Machine Vision Conference*, pages 76.1–76.12. BMVA Press.

Schmalstieg, D. and Hollerer, T. (2016). *Augmented reality: principles and practice*. Addison-Wesley Professional.

Schmalstieg, D., Langlotz, T., and Billinghurst, M. (2011). Augmented reality 2.0. In *Virtual Realities*, pages 13–37. Springer.

Schönberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113.

Schönberger, J. L., Zheng, E., Frahm, J.-M., and Pollefeys, M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pages 501–518. Springer.

Schroers, C., Bazin, J.-C., and Sorkine-Hornung, A. (2018). An omnistereoscopic video pipeline for capture and display of real-world VR. *ACM Transactions on Graphics (TOG)*, 37(3):1–13.

Se, S., Lowe, D., and Little, J. (2001). Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 2, pages 2051–2058. IEEE.

Shi, J. et al. (1994). Good features to track. In *1994 Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pages 593–600. IEEE.

Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., and Fitzgibbon, A. (2013). Scene coordinate regression forests for camera relocalization in RGB-D images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937.

Shum, H.-Y. and Szeliski, R. (1999). Stereo reconstruction from multiperspective panoramas. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 14–21. IEEE.

Siciliano, B. and Khatib, O. (2016). *Springer Handbook of Robotics*. Springer.

Siggelkow, S. (2002). *Feature histograms for content based image retrieval*. PhD thesis, University of Freiburg, Freiburg im Breisgau, Germany.

Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, page 1470. IEEE.

Skinner, P. and Zollmann, S. (2019). Localisation for augmented reality at sport events. In *2019 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE.

Sklansky, J. (1982). Finding the convex hull of a simple polygon. *Pattern Recognition Letters*, 1(2):79–83.

Smith, M., Baldwin, I., Churchill, W., Paul, R., and Newman, P. (2009). The new college vision and laser data set. *The International Journal of Robotics Research*, 28(5):595–599.

Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: exploring photo collections in 3D. In *ACM Siggraph 2006 Papers*, pages 835–846. Association for Computing Machinery.

Sony (2016). PlayStation VR. https://www.playstation.com/en-nz/explore/playstation-vr/. [Hardware; virtual reality headset].

Sturm, J., Burgard, W., and Cremers, D. (2012a). Evaluating egomotion and structure-from-motion approaches using the tum rgb-d benchmark. In *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*.

Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012b). A benchmark for the evaluation of RGB-D SLAM systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE.

Sweeney, C., Holynski, A., Curless, B., and Seitz, S. M. (2019). Structure from motion for panorama-style videos. *arXiv preprint arXiv:1906.03539*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Szeliski, R. (2006). Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

Taketomi, T., Uchiyama, H., and Ikeda, S. (2017). Visual SLAM algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1):16.

Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6243–6252.

Thomas, G., Gade, R., Moeslund, T. B., Carr, P., and Hilton, A. (2017). Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3–18.

Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features. Technical report, School of Computer Science, Carnegie Mellon Univ. Pittsburgh.

Torii, A., Imiya, A., and Ohnishi, N. (2005). Two-and three-view geometry for spherical cameras. In *Proceedings of the Sixth Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*, pages 81–88. Citeseer.

Torr, P. H. S. (2002). Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61.

Triggs, B. (1999). Camera pose and calibration from 4 or 5 known 3D points. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 278–284. IEEE.

Upenik, E., Řeřábek, M., and Ebrahimi, T. (2016). Testbed for subjective evaluation of omnidirectional visual content. In *2016 Picture Coding Symposium (PCS)*, pages 1–5. IEEE.

Ventura, J. (2016). Structure from motion on a sphere. In *European Conference on Computer Vision*, pages 53–68. Springer.

Ventura, J., Arth, C., Reitmayr, G., and Schmalstieg, D. (2014). Global localization from monocular SLAM on a mobile phone. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):531–539.

Ventura, J. and Höllerer, T. (2012). Wide-area scene mapping for mobile visual tracking. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 3–12. IEEE.

Verhagen, B., Timofte, R., and Van Gool, L. (2014). Scale-invariant line descriptors for wide baseline matching. In *2014 IEEE Winter Conference on Applications of Computer Vision, WACV 2014*, pages 493–500.

Vizrt (2010). Viz Virtual Studio. https://www.vizrt.com/products/viz-virtual-studio/. [Hardware; tracked studio camera system].

Von Gioi, R. G., Jakubowicz, J., Morel, J.-M., and Randall, G. (2010). LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):722–732.

Wagner, D., Langlotz, T., and Schmalstieg, D. (2008). Robust and unobtrusive marker tracking on mobile phones. In *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 121–124. IEEE.

Wagner, D., Mulloni, A., Langlotz, T., and Schmalstieg, D. (2010). Real-time panoramic mapping and tracking on mobile phones. In *2010 IEEE virtual reality conference (VR)*, pages 211–218. IEEE.

Wang, J., Cipolla, R., and Zha, H. (2005). Vision-based global localization using a visual vocabulary. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 4230–4235. IEEE.

Williams, B., Klein, G., and Reid, I. (2007). Real-time SLAM relocalisation. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE.

Wilson, K. and Snavely, N. (2014). Robust global translations with 1DSfM. In *European Conference on Computer Vision*, pages 61–75. Springer.

Wither, J., Tsai, Y.-T., and Azuma, R. (2011). Indirect augmented reality. *Computers & Graphics*, 35(4):810–822.

Wolf, J., Burgard, W., and Burkhardt, H. (2005). Robust vision-based localization by combining an image-retrieval system with monte carlo localization. *IEEE Transactions on Robotics*, 21(2):208–216.

Wu, C. (2013). Towards linear-time incremental structure from motion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 127–134. IEEE.

Wu, C., Agarwal, S., Curless, B., and Seitz, S. M. (2011). Multicore bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3057–3064. IEEE.

Xiong, Y. and Pulli, K. (2010). Fast panorama stitching for high-quality panoramic images on mobile phones. *IEEE Transactions on Consumer Electronics*, 56(2):298–306.

Yin, Z. and Shi, J. (2018). GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992.

Young, J., Langlotz, T., Cook, M., Mills, S., and Regenbrecht, H. (2019). Immersive telepresence and remote collaboration using mobile and wearable devices. *IEEE transactions on visualization and computer graphics*, 25(5):1908–1918.

Young, J., Langlotz, T., Mills, S., and Regenbrecht, H. (2020). Mobileportation: Nomadic telepresence for mobile devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(2):1–16.

Yu, F. and Gallup, D. (2014). 3D reconstruction from accidental motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3986–3993.

Yu, L., Ong, S.-K., and Nee, A. Y. (2016). A tracking solution for mobile augmented reality based on sensor-aided marker-less tracking and panoramic mapping. *Multimedia Tools and Applications*, 75(6):3199–3220.

Zhang, F. and Liu, F. (2015). Casual stereoscopic panorama stitching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2010.

Zhang, L. and Koch, R. (2013). An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. *Journal of Visual Communication and Image Representation*, 24(7):794–805.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334.

Zhou, T., Tucker, R., Flynn, J., Fyffe, G., and Snavely, N. (2018). Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*.

Zhou, T., Tulsiani, S., Sun, W., Malik, J., and Efros, A. A. (2016). View synthesis by appearance flow. In *European Conference on Computer Vision*, pages 286–301. Springer.

Zollmann, S., Hoppe, C., Langlotz, T., and Reitmayr, G. (2014). FlyAR: Augmented reality supported micro aerial vehicle navigation. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):560–568.

Zollmann, S., Langlotz, T., Loos, M., Lo, W. H., and Baker, L. (2019). ARSpectator: Exploring augmented reality for sport events. In *SIGGRAPH Asia 2019 Technical Briefs*, SA '19, page 75–78, New York, NY, USA. Association for Computing Machinery.

Zou, D. and Tan, P. (2012). CoSLAM: Collaborative visual SLAM in dynamic environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):354–366.