Local Features, Structure-from-motion and View Synthesis in Spherical Video

Hao Guan

Doctor of Philosophy

University of York
Computer Science

March 2017

Abstract

This thesis addresses the problem of synthesising new views from spherical video or image sequences. We propose an interest point detector and feature descriptor that allows us to robustly match local features between pairs of spherical images and use this as part of a structure-from-motion pipeline that allows us to estimate camera pose from a spherical video sequence. With pose estimates to hand, we propose methods for view stabilisation and novel viewpoint synthesis.

In Chapter 3 we describe our contribution in the area of feature detection and description in spherical images. First, we present a novel representation for spherical images which uses a discrete geodesic grid composed of hexagonal pixels. Second, we extend the BRISK binary descriptor to the sphere, proposing methods for multiscale corner detection, sub-pixel position and sub-octave scale refinement and descriptor construction in the tangent space to the sphere.

In Chapter 4 we describe our contributions in the area of spherical structure-from-motion. We revisit problems from multiview geometry in the context of spherical images. We propose methods suited to spherical camera geometry for the spherical-n-point problem and calibrated spherical reconstruction. We introduce a new probabilistic interpretation of spherical structure-from-motion which uses the von Mises-Fisher distribution in spherical feature point positions. This model provides an alternate objective function that we use in bundle adjustment.

In Chapter 5 we describe our contributions in the area of view synthesis from spherical images. We exploit the camera pose estimates made by our pipeline and use these in two view synthesis applications. The first is view stabilisation where we remove the effect of viewing direction changes, often present in first person video. Second, we propose a method for synthesising novel viewpoints.

Contents

List of Figures

List of Tables

List of Symbols

S^2	The 2-sphere
V_s	Number of vertices of triangular mesh on the sphere
F_s	Number of faces of triangular mesh on the sphere
$\mathcal{M}_s$	A triangular mesh
$\mathcal{K}_s$	The adjacency information of a triangular mesh
$\mathbf{v}_{s,i}$	3D coordinates of ith vertex on the sphere
$\mathbf{V}_s$	Coordinates of all vertices in the mesh
$\mathbf{T}_s$	Colors of all hexagonal pixels on the sphere
$\mathbf{t}_{s,i}$	Color of ith pixel on the sphere
$\mathcal{N}_n(\mathbf{v}_{s,i})$	The n-ring neighbourhood of ith pixel on the sphere
c_i	The ith octave
d_i	The ith intra-octave
$\mathrm{Log}_p(q)$	Riemannian log map of point q at base point p on the sphere
T_pS^2	Tangent plane to the sphere at point p
$d_g(p,q)$	Euclidean distance between point p and q
$\mathrm{Exp}_p(q)$	Exponential map of point q at base point p on the sphere
$\mathcal{F}$	FAST score
$SO(3)$	Group of 3D rotations
$\mathbf{w}$	3D world point
$\mathbf{x}$	Point on the sphere represented as 3D unit vector
$\boldsymbol{\Omega}, \boldsymbol{\tau}$	Pose (rotation, translation) of a camera
spherical	Spherical camera projection
$\mathbb{R}^n$	The n dimensional real coordinate space
vMF	The von Mises-Fisher distribution
Pr	Probabilistic model
$\mathbf{E}$	Essential matrix

Acknowledgements

First of all, I am so happy to express my sincere gratitude to my supervisor, Dr. William A.P. Smith for the continuous support, his immense patience, encouragement, friendly advice during my PhD. I am so glad to work with him and I could not have imagined to have a better supervisor for my PhD research.

Besides my supervisor, I want to express many thanks to my internal assessor Professor Edwin Hancock for his critical assessment of my work. I also need to say thanks to the technical staff Alex Cooper who gave me the access to the laboratory and research facilities and helped me lots of things. I thank those people at the CVPR group for their kindly help.

Last but not the least, my sincere and deepest gratitude goes to my family for their continuous and constant support, care and tireless education. My PhD work will be impossible without their love and support.

Declaration

I declare that the research described in this thesis is original work, which I undertook at the University of York during 2012 - 2016. Except where stated, all of the work contained within this thesis represents the original contribution of the author.

This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as references. Some parts of this thesis have been published in conference proceedings and journals; where items were published jointly with collaborators, the author of this thesis is responsible for the material presented here.

Journal Papers

- H. Guan and W.A.P. Smith, Structure-from-motion in Spherical Video using the von Mises-Fisher Distribution. *IEEE Transactions on Image Processing 26.2 (2017): 711-723.*

Conference Papers

- H. Guan and W.A.P. Smith, BRISKS: Binary Features for Spherical Images on a Geodesic Grid. In Proc. CVPR, to appear, 2017.

- H. Guan, W.A.P. Smith and P. Ren, View Stabilisation in Spherical Video. In *Proc. The 12th Workshop on Non-classical Cameras, Camera Networks and Omnidirectional Vision (OMNIVIS)*, 2014.

- H. Guan, W.A.P. Smith and P. Ren, Corner detection in spherical images via accelerated segment test on a geodesic grid. In *Proc. International Symposium*

on Visual Computing, Springer Lecture Notes in Computer Science, vol. 8033, pages 407-415, 2013.

Chapter 1

Introduction

Have you ever observed a spectacular scene on a journey but felt that the narrow field of view of your camera does not capture how it felt to be immersed in the scene? Or perhaps you wish to experience an extreme sport like base jumping or freeride skiing but the head mounted video captured by someone contains camera shake and doesn't look in the direction you would like to see? The answer to these problems may lie in the capture, processing and display of spherical images and video (also known as 360, panoramic or omnidirectional images/video). A spherical image is one with a $180° \times 360°$ field of view (i.e. 4π steradians). In Figure 1.1 we show two visualisations of such a spherical image: on the left the image is shown texture-mapped on the surface of a sphere, on the right a user is interactively viewing a spherical image via a virtual reality head mounted display. Often, a spherical image is stored in a format known as a *panoramic image*. This is an equirectangular projection of the sphere to the plane, leading to an image with $2 : 1$ aspect ratio where pixel columns have equal azimuth angle and rows have equal elevation angle. Conversion to and from panoramic image format is illustrated in Figure 1.2.

Recently, the ability to capture spherical images and video and to view them interactively has been placed in the hands of the consumer. This has been largely driven by the rise of consumer virtual reality [14], kickstarted by the release of the Oculus Rift head mounted display[1]. Virtual reality has a wide range of applications,

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

Figure 1.1: Left: the spherical image sample shown in Matlab. Right: the spherical image view by virtual reality kit.

most obviously gaming and social networking. For specific applications it brings particular advantages. For example, in military simulation[2] it allows soldiers to be tested and trained in a controlled, safe environment. In digital cultural heritage, it allows large numbers of visitors to interact with an artefact or exhibition without overcrowding or damaging the exhibit. Convincing virtual reality requires realistic content. While one means of achieving this is standard computer graphics, this requires vast repositories or hand constructed 3D models and texture maps and human-designed lighting and material properties. Even then, the result is unlikely to be photorealistic.

Spherical video provides a compelling alternative. By definition, it is photorealistic and due to its full coverage of the viewing sphere, it allows interactive selection of viewing direction at playback time. From a practical perspective, spherical images and video can be captured either with an array of perspective cameras that are stitched into a seamless panorama or by using curved mirrors (catadioptric camera) or wide angle lenses (dioptric camera). Of course, compared to computer graphics, the drawback is that content is fixed at capture time and new viewpoints cannot be chosen by the viewer. One possible solution is to use computer vision to synthesise new views. However, many of the fundamental tools of computer vision are designed for planar perspective images and the techniques do not necessarily extend to im-

[2]http://www.vrs.org.uk/virtual-reality-military/

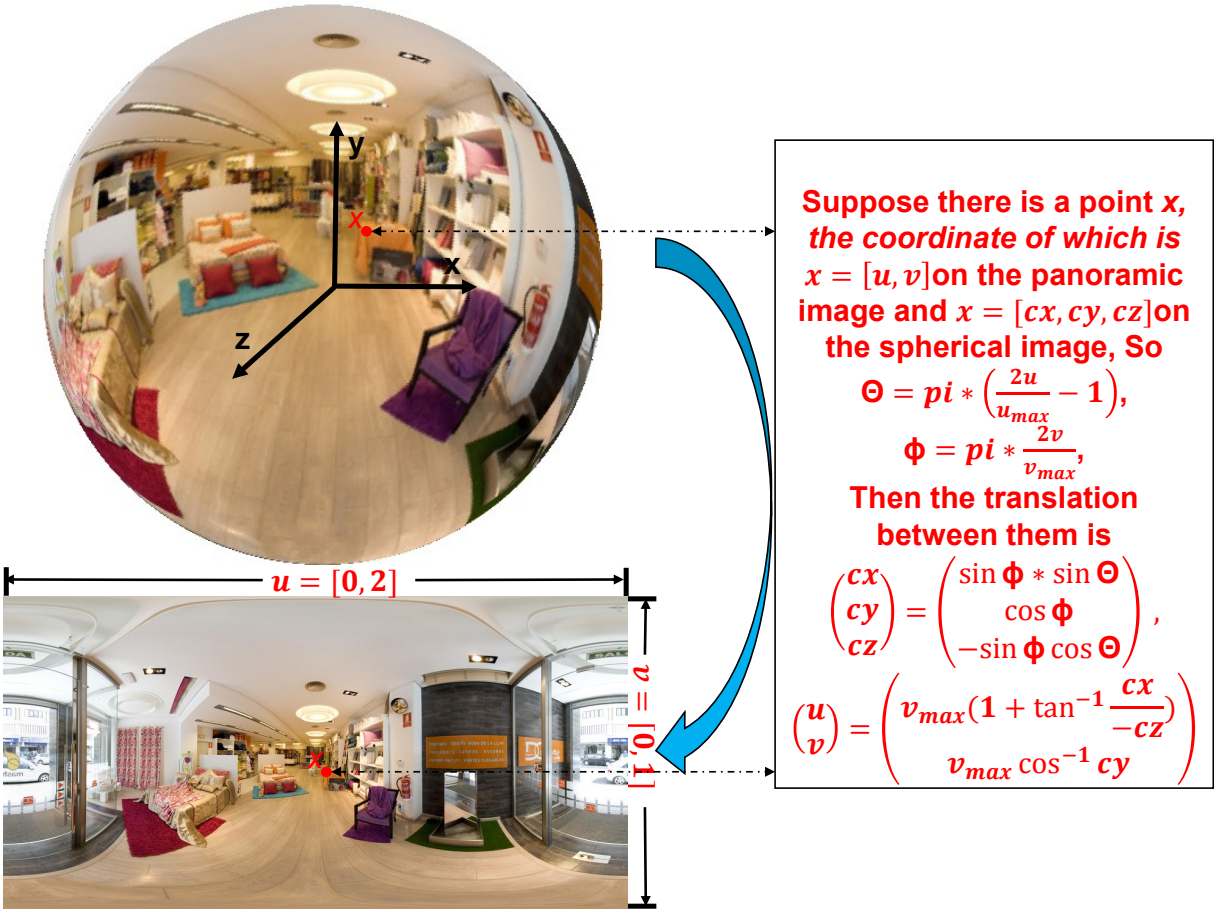


Figure 1.2: The translation between the spherical image and the panoramic image.

ages on the sphere. In this thesis, we develop new techniques in three areas: local features, structure-from-motion and view synthesis, that are specifically designed to operate on spherical images. We focus on these areas as they provide a route to synthesising new views from a captured video.

1.1 Motivation

First person video provides a viewpoint which is highly immersive and engaging for viewers. Consumer cameras designed specifically for capturing high definition first person video are now commonplace. They represent a significant proportion of the digital camera market and are regularly used in production-quality TV broadcasts. However, first person video suffers from a numbers of limitations which reduce its attractiveness and utility in a practical setting:

1. **Rapid changes in viewing direction and camera shake.** The person who wears the camera in some sports such as horse racing, bungee Jumping changes the view direction rapidly and shakes the camera.

2. **Motion blur.** The camera may move too quickly relative to the exposure time of the camera.

3. **Undesirable viewing direction.** Often, the camera is looking in a direction which is not interesting to the viewer. For example, in the horse race, we do not want to look down at the reins, but instead at the horse next to us.

4. **Lack of interactivity.** We cannot change the viewing direction while watching the video.

These limitations mean that viewing such videos is a highly disorientating and often frustrating experience. One approach to tackling these issues is to use spherical video. This gives the advantage of capturing as much information as possible about the surrounding environment and to reduce the effect of the direction in which the camera is facing. From the perspective of stabilisation, another attractive property is that the greatly increased field of view substantially increases the chance of observing features seen previously.

Such spherical video provides an immersive experience for the viewer in the sense that viewing direction can be chosen and varied during playback. This is done either by controlling the viewpoint manually using a mouse (or accelerometers in a tablet) or, more naturally, by changing head pose in a virtual reality head mounted display. However, a major problem with these experiences is that viewing direction also changes as the camera moves and rotates through a scene. This is disorienting for the viewer and can even lead to motion sickness as they lose control over the direction in which they are looking.

1.2 Aims

Our goal is to use first person, omnidirectional video sequences to build a 3D model of the environment through which the camera travels and use these models to ren-

der new sequences from different viewpoints or with stabilised viewing direction. In another words, we will target applications in which first person spherical video is the source of structure-from-motion data. This could be an extreme sports person moving through an environment or even a soldier moving through a battlefield environment. First person video (FPV) allows people to stand in someone else's shoes and experience what they saw. It is often captured using a helmet mounted camera and has been widely used such as extreme sports, search and rescue, law enforcement, military. As well as being engaging and exciting for the viewer, such video can also be useful. For example, the police uses it as evidence in crowd scenarios or it could be used for analysis of training events.

The novelty of my research is that instead of the conventional perspective camera model, we focus on omnidirectional images captured with a spherical camera. From a structure-from-motion perspective, the advantage is that spherical images capture the scene in every direction and have better coverage of the scene. This means that features will last longer, giving the potential for higher accuracy. We also develop techniques that allow new view synthesis from first person, omnidirectional video. We are interested in first person video because it is an engaging and potentially useful viewpoint from which to capture an event or activity. We are interested in omnidirectional video as it allows all viewing directions to be captured simultaneously, eliminating the problem of undesirable viewing direction.

Our aim is to allow first person video to be processed in such a way as to provide a better experience when played back. For example, we may wish to stablise the viewing direction and position in a bumpy video. Or we may wish to move the viewpoint to a completely different position, providing a different perspective on the scene. In this thesis, we follow a particular route to achieving this, though our contributions could be used in alternative pipelines. We design a new local feature allowing robust, sparse point matches between pairs of spherical images. Second, we develop a structure-from-motion framework for estimating sparse 3D scene structure and camera motion from spherical video. Finally, we directly synthesise new views, either by stabilising viewing direction or changing viewpoint using ray casting and photoconsistency. An alternative (not explored here) would be to use our estimated

camera poses to compute dense 3D structure (i.e. spherical multiview stereo) and rendering new views from the estimated models using computer graphics. This future work could follow on from ours.

1.3 Contributions

We now provide a chapter-by-chapter summary of the novel contributions made in this thesis.

1.3.1 Binary Features for Spherical Images on a Geodesic Grid

In this chapter we propose a novel local feature detector that is designed specifically to operate with spherical images. Specifically, we adapt the BRISK framework to work on a geodesic grid representation of a spherical image. In doing so, we make a number of novel contributions. First, we introduce a multiscale geodesic grid representation for spherical images which overcomes problems of distortion and uneven sampling inherent in panoramic images. Second, we propose a variant of the Accelerated Segment Test (AST) corner detector which works with the hexagonal pixels of our representation. Third, we present a discrete scale space by recursively subdividing the geodesic grid and using different sized AST patterns to work a intraoctaves. Finally, we propose the spherical BRISK feature which works on the sphere which includes the novel continuous scale and position refinement in the tangent plane and the novel pattern resampling for the binary feature construction. We have evaluated the performance in terms of the repeatability of the interest point detector and the precision and recall of the feature descriptor. We experiment with both synthetic and real images and include comparison to two previous methods.

1.3.2 Structure-from-motion in Spherical Video using the von Mises-Fisher Distribution

In this chapter, we introduce novel theory that provides the components of a structure-from-motion pipeline that is specifically adapted to spherical image geometry. There are a number of novel ingredients to our work. First, we present a probabilistic model of spherical image formation and use the von Mises-Fisher distribution to model spherical noise. Second, this leads us to a novel objective function for the spherical structure-from-motion problem. Third, in order to initialise minimisation of the non-convex objective function, we propose new methods for pose estimation and 3D world point reconstruction that are specifically adapted to the spherical case. Finally, we present a complete spherical structure-from-motion pipeline that combines these new methods with spherical feature match filtering for robustness. We present quantitative experimental results on both synthetic and real world data. Our results show that our spherical pose estimation methods consistently outperform the classical DLT approach. Moreover, the complete pipeline that includes optimisation of the error term based on the von Mises-Fisher distribution outperforms squared angular error or squared Euclidean distance as used in previous work.

1.3.3 View Stabilisation And Synthesis

In the final contribution chapter, we take some preliminary steps towards using the structure-from-motion output to synthesise new views from the original sequence. Our first contribution is a view stabilisation process. Using the camera pose estimates provided by spherical structure-from-motion, we rotate the frames back onto a canonical view and are able to remove the effect of viewing direction changes. We present qualitative results on real world spherical video sequences. Our approach is sufficiently robust to work well on real world, highly unstable image sequences. Our second contribution is a method for synthesising new viewpoints using only the camera pose estimates and the input images (i.e. we do not use any 3D information). Our method involves casting rays for each pixel in the virtual viewpoint and optimising photoconsistency across epipolar curves over all input images. We

present results for both view interpolation and extrapolation on real world data.

1.4 Thesis Structure

In Chapter 2 we review the existing literature in the areas of interest point detection, feature description, structure-from-motion and SLAM and view stabilisation and synthesis. In Chapter 3 we introduce our multiscale geodesic grid representation for spherical images, corner detector for hexagonal images and our binary feature descriptor. In Chapter 4 we introduce a complete spherical structure-from-motion pipeline that uses a probabilistic model based on the von Mises-Fisher distribution. We introduce novel formulations for the spherical-n-point problem and calibrated spherical reconstruction. In Chapter 5 we focus on applications of the previous two chapters to the problem of novel view synthesis. We present methods for view stabilisation and novel viewpoint synthesis and present preliminary results in this direction. Finally, in Chapter 6 we draw conclusions from the work, discuss weaknesses of the work and suggest directions for future work.

Chapter 2

Literature Survey and Review

In this chapter, we review the literature in the three key areas relevant to our line of research. We begin by reviewing work on interest point detection and feature description, focussing particularly on attempts to extend these methods to spherical images. Next, we review work concerned with estimating camera pose and sparse 3D scene information from a set of spherical images, i.e. spherical structure-from-motion or simultaneous localisation and mapping. Finally, we review previous methods for stabilisation of video and novel view synthesis before drawing some conclusions on the state of the field.

2.1 Interest Point Detection And Feature Description

Interest point detection and feature description are fundamental problems in computer vision with a huge range of potential applications from 3D modelling [86], to image stitching [12] to object and scene recognition [97]. For this reason, there is a vast body of literature in the area, dating back to the early days of the field. We review only the most relevant here but direct the interested reader towards surveys of the topic. For a comprehensive survey of interest point detection see Tuytelaars and Mikolajczyk [96]. Mikolajczyk and Schmid [73] undertook an early survey and comparison of feature descriptors, however this was before the rise of lightweight, binary features. A more recent evaluation of these approaches was undertaken by

Miksik and Mikolajczyk [75]. We note that, in this survey, the best performing of the lightweight binary features was BRISK.

An alternative to 'hand crafted' features that has recently gained huge popularity, is to learn suitable image features as part of a complete processing pipeline that is trained end-to-end. A particularly popular example of such an approach is a Convolutional Neural Network (CNN) which provides state-of-the-art performance in many classification problems such as object recognition [46]. Nevertheless, hand crafted features remain the predominant approach in geometric vision problems where their repeatability and localisation properties are well understood and evaluated.

2.1.1 Interest point detection

Key to the design of a local feature is a means to identify interest points in a manner that is repeatable over changes in appearance caused by extrinsic factors (e.g. changes in viewpoint, lighting or camera). Interest points are typically defined as corners, blobs or regions and the various methods seek to efficiently detect the presence of these generic features, often with an associated scale. We discuss previous methods for interest point detection for both planar and spherical images and focus on corner detection since this is what we use.

Planar interest points Corner detection has a long history in image processing. Perhaps the best known algorithm is the Harris corner detector [38] which is based on the idea that corners should have low self-similarity. It uses an approximation of the second differential of the local sum of squared differences. At corner locations this quantity varies in all directions which is measured by testing for large values of both eigenvalues of the structure tensor. An extension that avoids having to select a threshold for the eigenvalues has been proposed [79].

There has been increased attention on methods that do not require calculation of second derivatives (which are sensitive to noise) and which are computationally efficient enough to be used in real time feature detection and tracking. The SUSAN (Smallest Univalue Segment Assimilating Nucleus Test) test [91] is based on com-

paring a candidate pixel to its circular pixel neighbourhood. It works on the low level image processing. First of all, it applys a circular mask for each given point in an image. Then it compares the different intensity between the neighbouring pixels and the center of the mask(nucleus) and finally repeat the procedure for each pixel of that image. The area with similar value compared with the nucleus is called SUSAN area and its varies depending on the location of the nucleus. It means the SUSAN area will be maximum if it is in a flat region and fall to minimum at an edge and be an even further when around a corner.

More recently, this approach was simplified to consider only pixels lying on the edge of the circular region. In the FAST (Features from Accelerated Segment Test) approach, a sequence of 12 pixels out of 16 must be substantially brighter or darker than the central pixel to be considered a corner. The challenge is to construct a decision tree that allows pixels to be classified using as few tests as possible on average. This was originally done by training a tree using data [85]. This was further improved in the AGAST (Adaptive and Generic Corner Detection Based on the Accelerated Segment Test) [67] framework by building a decision tree which does not have to be trained for a specific scene, but rather dynamically adapts to the image. This has led to extremely low computational requirements, whilst still offering repeatability that outperforms more complex corner detectors.

None of the above interest point detectors are scale invariant. One solution to this is to simply apply the same detector at multiple scales over an image pyramid. A more principled alternative is to develop a detector that can also compute a characteristic scale. Based on the criterion that the scale space need to be linear, the Gaussian filter has been proved to be one of the canonical way to smooth and build it. Lindeberg [53] proposed such a method for Harris corners that uses the determinant of the Hessian or the Laplacian. The SIFT [62] framework approximates the Laplacian of Gaussian by detecting maxima and minima in the difference of Gaussians function applied over scale space. There is evidence that a similar mechanism may be used in biological vision. In an effort to reduce computational cost, the Speeded Up Robust Features (SURF) [9] framework approximates the Hessian very efficiently using integral images.

Spherical interest points Following the success of SIFT-like interest points, a number of previous works [4,19,37] build a scale-space representation with Gaussian kernels on the sphere, and use the spherical Fourier transform to perform convolution on equirectangular images. Keypoints are identified as extrema in the resulting difference of Gaussian images. One possible way to discretize and store the spherical image is to use a mostly hexagonal discretisation. Although not in the context of spherical images. There is a substantial literature in planar hexagonal image processing [72] due to its biological plausibility and the attractive properties of working on a hexagonal pixel lattice. The problem of corner detection in such images has been considered previously, for example by Liu et al. [59]. Instead of storing, displaying and processing the images using square pixels, the authors use hexagonal pixels which bring benefits including more accurate discretisation of curved objects.

2.1.2 Feature Description

Having identified a suitable set of interest points, usually with an associated scale, the second goal is to build a descriptor of the local appearance around the feature along with a distance measure that determines feature similarity. As with interest point detection, invariance to extrinsic factors is desirable. Ideally, the same point viewed under any conditions from any viewpoint would yield the same descriptor. Competing with this aim of robustness is a desire to have features that are distinctive, i.e. descriptors of different scene points yield dissimilar descriptors.

Planar feature descriptors The benchmark feature descriptor is SIFT [62] which is based on histograms of gradient orientations, normalised by the dominant gradient direction. This method gets features on the scale space. The scale space is calculated by convolving the different level Gaussian filters with the original image. This algorithm is to perform a detailed model which is fitted to each feature candidate location to determine their location and scale. According to the gradient direction, each feature is assigned one or more orientations. The SIFT description is highly distinctive and invariant to such as illumination, view point direction, rotation and scale. The drawback of SIFT is the computational cost of computing the descriptor,

combined with its high dimensionality (128 floating point values) making storage and comparison also expensive. PCA-SIFT [44] addresses this directly by using Principal Components Analysis (PCA) to compress the features to 36 dimensions. However, this increases the cost of building descriptor and reduces distinctiveness. In a similar vein, the GLOH (Gradient Location-Orientation Histogram) descriptor [73] computes a richer descriptor than SIFT but then reduces dimensionality to the same as SIFT using PCA. This leads to an improvement in robustness and distinctiveness over SIFT.

An important attempt to improve the efficiency of SIFT features was proposed by Bay et al. [9]. Their Speeded-Up Robust Features (SURF) reduce the complexity of feature detection by using an approximation of a Hessian blob detector based on integral images. Feature descriptors use sums of Haar wavelet responses, again exploiting integral images, and descriptor dimensionality if reduced to 64 dimensions. Taken together, these efficiencies lead to a speed improvement of almost an order of magnitude. Nevertheless, SURF features are still orders of magnitude away from the efficiency required for real time applications such as Simultaneous Location and Mapping (SLAM).

This provided motivation for a new class of descriptor developed mainly over the past 5 years. These methods are characterised by the use of binary descriptors that encode the result of simple image intensity comparisons, enabling feature dissimilarity to be measured with Hamming distance (simply a bitwise XOR followed by a bit count). This makes descriptor extraction, storage and comparison extremely fast. Early work in this direction is due to Agrawal et al. [2] who replace SIFT descriptors with centre-symmetric local binary patterns. An important landmark is the BRIEF (Binary Robust Independent Elementary Features) descriptor [15]. This forms the basis of the most recent binary features. A BRIEF descriptor is built by performing a series of intensity comparisons between random, pre-determined pixel locations around an interest point. In its original form however, it offers no invariance to scale or rotation. Subsequently, ORB (Oriented FAST and Rotated BRIEF) [87] combined BRIEF descriptors with FAST interest points and introduced rotation invariance using corner intensity centroids. Going further, BRISK [49] makes an

approximation of the local gradient direction for rotation normalisation and introduces sub-pixel and sub-octave position and scale refinement. There is empirical evidence that BRISK is the best performing of the binary descriptors [40,75]. It is a novel method for features corner detection, feature description and point matching. First of all, the keypoints are detected on scale-space. It means that they have the scale invariance property. Apart from the similar scale space which is called octave in the paper like SIFT [62] do, they also build intra-octave between them in order to make sure the more smoothness of the scale space. They apply the FAST [85] mask on each pixel at each scale layer to compute the FAST score by comparing the grey value among the neighbouring and the neighbouring scale centre pixels. The point will be regarded as a corner if it is the maximum FAST score. They also use some measurements to refine the corner position and score across the layers. Given a set of keypoints, the lightweight binary string descriptor is generated by the simple brightness comparison test. After getting the descriptor, the matching process is simply based on the hamming distance computation which measures the dissimilarity of the two descriptors. The evaluation of this lightweight binary method shows their robustness and efficiency.

Spherical feature descriptors Recently the omnidirectional vision has become an hot topic. One of its main advantages is that a spherical camera has the wide field of vision which can cover all the scene around it. A recent development has been the creation of interest points and local feature descriptors that account for the geometry of spherical images. Simply using 2D local features on 2D parameterisations of spherical images is unreliable due to the directionally-dependent distortion present. Various authors have previously overcome this problem via pre-processing or simple hacks to improve performance of planar descriptors on spherical images, e.g. [81] generate multiple virtual perspective views and then extract planar features. However, recent work has taken a more principled approach. Feature descriptors for omnidirectional and spherical images can be classified into naive methods that use planar descriptors on a planar embedding of the image and those that account for the spherical geometry and build the feature on the sphere.

A number of previous works have taken the former approach and compute pla-

nar feature descriptors on a planar embedding of a spherical image (most commonly equirectangular or directly on an image acquired by a catadioptric camera). Goedemé et al. [28] and Scaramuzza and Siegwart [89] both compute SIFT features directly on an unwrapped spherical image. Benseddik et al. [11] apply the planar BRISK descriptor to catadoptric images and find improved performance over SIFT. The drawback of these approaches is that the distortion varies with pose changes and so the local appearance around an interest point may vary dramatically, even with only a change in viewing direction (i.e. a rotation).

An improvement over working directly with a planar embedding is to locally adapt the embedding to reduce the distortion. Fiala and Roth [22] transform the sphere to a cube map and then compute SIFT features on each cube face. Mauthner et al. [70] match regions in omnidirectional images by generating virtual perspective views. Pagani et al. [81] regularly distribute points over the sphere and generate a perspective image centred on each point with a fixed field of view. They further introduce affine distortions (in the same manner as Affine-SIFT [76]) providing invariance to both spherical distortion and affine transformations of the scene. Although robust, such an approach dramatically increases computational cost since a large number of images must be generated by resampling and processed independently. Hansen et al. [37] and Arican and Frossard [4] both build descriptors based on circular support regions on the sphere centred on an interest point. Hansen et al. [37] resample to the tangent plane and then build planar SIFT descriptors. Arican and Frossard [4] build log-polar descriptors on the sphere in a way that accounts for the different sampling densities. They build a descriptor that adapts to the specific geometry and non-uniform sampling density of spherical images. They evaluate the performance by comparing the results with the others methods: the one introduced by Cruz-Mota et al. [19] who build the similar SIFT descriptor on the sphere and another one presented by Hansen et al. [37]. The results shows the promising performance of the new descriptors.

The most attractive approaches avoid issues of distortion and boundaries and work directly on the sphere. We are aware of only two previous pieces of work that do so [19, 104]. Cruz-Mota et al. [19] provide a full spherical extension of SIFT.

They use heat diffusion instead of Gaussian filtering on the sphere to compute a spherical scale space. They extend the Scale Invariant Feature Transform [62] to operate in the spherical domain. This approach correctly handles the differential geometry of the spherical image surface providing highly robust features. However, for planar images, the SIFT approach is considered computationally too expensive to be viable for real-time or large scale feature matching problems, such as occur in structure-from-motion. In this case, the situation is even worse since computing the scale space via heat diffusion is expensive. There is therefore a need for lightweight and robust features in the omnidirectional imaging domain. The only work in this direction is the recent SPHORB approach of Zhao et al. [104]. They build on the popularity of binary descriptors and developed a spherical extension of the ORB descriptor on a geodesic spherical grid. We discuss in detail the differences between SPHORB and our proposed spherical feature in the next chapter.

2.2 Spherical Structure-from-motion and Simultaneous Localisation And Mapping

Once features between pairs of images have been matched, structure-from-motion (SFM) or Simultaneous Localisation and Mapping (SLAM) can be used to reconstruct the salient and sparse 3D points along with the position of the camera in each image. The main idea of the structure-from-motion and SLAM is trying to take images captured from different (unknown) viewpoints and estimate the camera positions and sparse 3D scene information. This is done by identifying scene points that are visible in more than one image. The process begins by detecting features, then finding correspondences by matching features. As camera positions and 3D positions are estimated, matches can be refined by removing outliers. Refining both 3D point and camera pose estimates together is a large, sparse nonlinear optimisation problem, known as bundle adjustment. Once camera positions are accurately known, it is possible to compute a dense 3D surface using the technique of multiview stereo. When the images form a sequence with small changes in position (i.e. the camera moves through the scene along a trajectory), the problem is usually referred

to as SLAM and the images are said to have narrow baseline. SLAM is usually done online as a robot moves through a scene. When the images are taken from arbitrary, unconnected positions, the problem is usually referred to as structure-from-motion and the images are said to have wide baseline. It is usually done offline with the whole image set available. There have been lots of high quality algorithms developed to solve this problem [21, 24, 25, 103]. The omnidirectional camera has been used in computer vision because it has the obvious advantage of wide FOV. In this section we describe related work in the area of spherical structure-from-motion and SLAM.

2.2.1 Spherical Structure-from-motion

There have been a number of attempts to extend SFM to operate on spherical images. Typically, this is in the context of robotics, autonomous vehicles or mapping, where the wide field of view enables maps to be constructed more quickly and location estimated with less ambiguity. Some of the earliest work was by Chang et al. [17] who developed an omnidirectional SFM algorithm and provide a comparison to conventional SFM with perspective images. They described the epipolar constraint for catadioptric cameras and estimated the essential matrix using the standard approach of a linear initialisation followed by nonlinear optimisation. Their objective is identical to the perspective case and they only consider the relative pose problem (not 3D reconstruction or optimisation of multiple poses simultaneously). Torii et al. [94] made a similar theoretical contribution for the case of a central projection, spherical camera. They presented a rigorous analysis of the extension of two and three view geometry to the spherical case.

Theoretical aspects of structure-from-motion in the context of spherical images have been considered by a number of researchers. Pagani et al. [81] presented straightforward modifications to perspective SFM to operate with the same spherical camera model that we use. The approach that they propose for pose estimation provides the baseline against which we compute. Pagani and Stricker [80] proposed several variations on the objective function used during spherical SFM. They conclude that two of these error models are the best objectives to handle the epipolar error and pose error. Specifically, the vertical distance to the epipolar plane (ε_p in

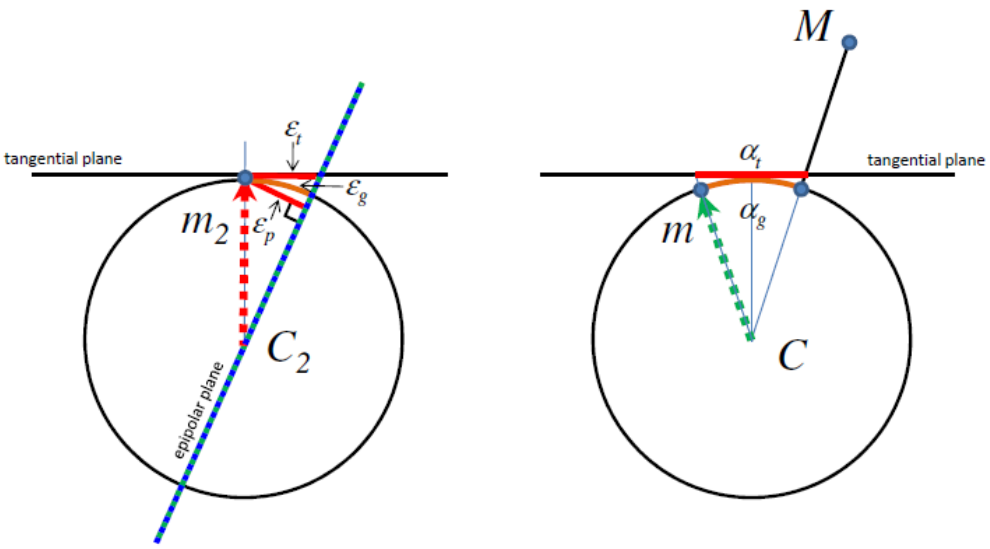


Figure 2.1: Tangential approximations of the errors. [80]

Figure 2.1 left) and the tangential error distance (α_t in Figure 2.1 right). In order to evaluate the performance, they also illustrate a lightweight complete structure-from-motion pipeline. Specifically, firstly they use the ASIFT [76] to detect the salient feature points and match the correspondences of all pairs between neighbouring cameras. For solving the epipolar geometry process, they choose two cameras as initial cameras and compute the relative poses of these cameras. After this initialisation, they repeatedly select the camera with the most matching points. Then, they get the 2D-3D correspondences using the already triangulated points. Thirdly, the pose of camera is computed and then triangulating all the matching points. Finally, sparse bundle adjustment (SBA) [61] is applied to optimise over all camera poses and world points. Their goal was to find error terms that could be efficiently evaluated yet agreed closely with minimising squared angular error. In contrast, the probabilistic model that we propose leads to an objective function based on maximisation of dot products.

Similarly, Krolla et al. [47] also consider different reprojection errors for spherical structure-from-motion, namely: angular (geodesic), Euclidean and tangential distance. The goal is to study uncertainty propagation in the context of optimisation (bundle adjustment) or spherical structure-from-motion problems. An interesting result is that geodesic distance results in corrupted uncertainty estimates. While

useful from a practical perspective, we argue that these distance measures are not justified by an explicit noise or probabilistic model. Our contribution in this regard is to use the von Mises-Fisher distribution to model uncertainty in spherical image formation and use this to derive a novel objective function.

2.2.2 Spherical Probabilistic Models in Vision

Our probabilistic model for spherical SFM uses the von Mises-Fisher (vMF) distribution [23]. It has been widely used in computer vision and machine learning. For example, Banerjee et al. [6] use the Expectation Maximisation algorithm for clustering spherical data according to the vMF. In particular, the vMF has been used in applications related to multiview vision and, more specifically, omnidirectional vision. Pons-Moll et al. [82] present a system for multiview human motion capture in which sensor noise is modelled with the vMF. Ćesić et al. [16] use the vMF for object tracking in a multi-camera system. Markovic et al. [68] extend the approach to spherical images. Bazin et al. [10] use diffusion of particles on the sphere for tracking in catadioptric images using a particle tracking framework. We are not aware of the vMF having been used previously for spherical SFM.

2.2.3 Simultaneous Localization and Mapping

Simultaneous Localisation and Mapping (SLAM), sometimes called visual odometry, are crucial research areas in robotics and autonomous systems. They are used for map building and navigation and have been shown to benefit from using omnidirectional cameras. Although they do not explicitly use a spherical camera model, Tardif et al. [93] present an omnidirectional SLAM using a fixed rig of perspective cameras. By combining efficient and robust landmark tracking and robust pose estimation, they are able to obtain very accurate trajectories for planar motion of a vehicle without a global bundle adjustment step.

Schmeing et al. [90] develop an omnidirectional bundle adjustment procedure which they applied to long omnidirectional image sequences. They form a virtual spherical camera by stitching multiple perspective views. They propose a bundle ad-

justment scheme using the same spherical camera model that we use. They directly adopt the energy term typically used in perspective bundle adjustment, namely sum of squared Euclidean reprojection distance, which does not account for the spherical image geometry.

Gutierrez and Rituerto [36] adapt a state-of-the-art, realtime SLAM system based on the Extended Kalman Filter (EKF) to operate on omnidirectional video. Their method is based on linearisation of a catadioptric camera model and they perform tracking with omnidirectional patch descriptors that are rotation and scale invariant. This paper first implements a spherical camera model. Instead of using the SIFT descriptor compared to the original approach by Andreasson et al. [3], image patches are used in which the rotation and scale invariance are also considered. As a result, the introduced method can be potentially much faster so as to be used in a real time system. Although there already exists some methods using image patches which have rotation invariance, this paper develops an algorithm for their scale invariance. Finally the new patch is compared to conventional patch-based methods to show their superiority. It is also used with a real time SLAM algorithm achieving high accuracy of the estimated trajectory. Similarly, Murillo et al. [77] apply the same approach to data that is similar to that in our stabilisation application, namely first person video captured by a wearable omnidirectional camera for visual odometry.

Torii et al. [95] build 3D city models from Google Street View imagery. Their focus is on building a robust, scalable system applicable to large numbers of high resolution images. Relative pose is computed using standard modifications of perspective methods but they introduce a robust technique for estimating the scale of translations. Bundle adjustment minimises an angular error, as in [50]. Rituerto et al. [83] focus on the SLAM issue using the monocular omnidirectional camera. EKF is again used here and the Spherical Camera Model proposed by Barreto and Araujo [7] is also integrated. They evaluate the results by comparing omnidirectional SLAM with the same application but using SIFT based features on conventional planar images. From the results, they conclude that visual SLAM results using the monocular omnidirectional camera improve trajectory estimation accuracy.

There have been a number of attempts to build hybrid SFM pipelines that incorporate or unify multiple camera geometries. Bastanlar et al. [8] use a central catadioptric model that can also model perspective cameras. This allows them to perform SFM on pairs of catadioptric and perspective images and they propose a preprocessing step to enable standard feature descriptors to be matched between the different image modalities. To enable a linear expression of the epipolar geometry, they use lifted coordinates for omnidirectional image points. In this paper, they also propose a general structure-from-motion pipeline. Moreover, the authors demonstrate that a means to combine a spherical camera and two or more normal cameras with no overlaps. As the results show, the hybrid system can get higher motion estimation accuracy rather than using only a perspective camera system. Gava and Stricker [26] proposed a unified SFM framework that allows uniform treatment of central projection cameras using a sphere as the underlying model. This allows perspective, spherical and hybrid image datasets to be analysed. Lhuillier [50] proposed minimisation of angular error during bundle adjustment as a means to unify multiple camera geometries.

2.3 View Stabilization And Synthesis

One application of feature matching followed by geometric 3D reconstruction is to use the estimated information to synthesise new images. Two particular applications of this idea are view stabilisation and novel viewpoint synthesis. The goal of view stabilisation is to take a video sequence and processing it so that the apparent camera motion varies more smoothly over time. It means that using the camera pose estimates provided by spherical structure-from-motion, we rotate the frames back onto a canonical view and are able to remove the effect of viewing direction changes. The goal of novel viewpoint synthesis is to predicte scene appearance from positions not included in the input images). Here we review work in these two areas.

2.3.1 View Stabilization

View stabilisation for video is a well studied problem for the case where images are captured by a perspective camera. Approaches can be roughly divided into those that estimate and apply a single homography to each frame [27] and those that construct an explicit 3D model and use this in the stabilisation process [54]. The former is easier to estimate and less computationally expensive but can only correctly stabilise scenes that are largely planar. The latter can potentially stabilise any motion through complex scenes but requires estimation of a 3D model (usually via structure-from-motion), a process that is computationally expensive and potentially fragile.

With an estimate of the 3D scene to hand, Liu et al. [54] compute a new stabilised path through the scene and use a grid-mesh to warp the input images in a content-preserving manner. But the result is not physically accurate. A popular recent alternative is to use 2D feature trajectories to guide the warp without explicitly estimating 3D scene information. Liu et al. [55] pose the problem of smoothing feature trajectories as one of low rank matrix factorisation. State-of-the-art approaches use feature trajectories to compute bundles of homographies and apply mesh-based warping [56]. This idea was extended to include user-guided constraints on the stabilisation result [5]. Many other view stabilisation methods have been proposed. Buehler et al. [13] use non-metric image-based rendering techniques to stabilize camera motion. Matsushita et al. [69] use motion inpainting to fill in missing image parts in both dynamic and static scenes. Liu et al. [58] introduce a novel motion model which uses smoothing pixel profiles instead of conventional feature tracking to stabilise the view. Goldstein et al. [30] use projective scene reconstruction to stabilise video from a hand-held camera. Cho et al. [18] address the problem of motion blur. They propose a patch-based approach that finds sharp content and uses this to restore blurred content in nearby frames. They assume that motion between frames can described by a homography. Li et al. [52] present a method using global joint motion estimation and a multiview deblurring technique to generate sharp and high-quality panoramas from motion blurred video. Lee et al. [48] stabilise video using robust feature trajectories. Wang et al. [99] use a new feature

and an improved motion smoothing method for video stabilisation. Going beyond simply filtering camera trajectories, Grundmann et al. [33] compute new camera trajectories composed of constant linear and parabolic segments to better simulate the sort of paths followed in professional cinematography. They do so using linear programming and without explicitly reconstructing the 3D scene. Liu et al. [57] make use of an additional depth camera. Although the depth maps are noisy and incomplete, they are adequate for motion estimation and frame warping.

A particular setting of the stabilisation problem is the creation of so-called "hyper-lapse" videos, i.e. time-lapse videos which follow a similar but stabilised moving camera trajectory. State-of-the-art in this area is due to Kopf et al. [45] who apply sparse structure-from-motion overlapping subsequences in order to compute pose for each video frame. A novel method is proposed for computing a smoothed trajectory in which viewing direction is close to some of the original frames. The output is generated by rendering using a geometric proxy, stitching and blending appropriately selected source frames.

2.3.2 View Synthesis

Closely related to video stabilisation is the problem of view interpolation and synthesis. Zitnick et al. [105] present a system for interactive, free-viewpoint video based on multiview reconstruction of dynamic scene from a static, calibrated multi-camera system. They reduce artefacts around depth discontinuities by matting. Rogmans et al. [84] introduce a framework that unifies stereo correspondence and view synthesis. Gurdan et al. [35] present a framework which is suited to wide baseline video to interpolate multi view image sequences in space and time. Goesele et al. [29] propose a method for smooth interpolation between a pair of views based on a sparse point cloud of the scene. Their method does not aim to produce accurate intermediate views, but rather to give an immersive sense of motion between two points in the scene. The advantage of this method is that it can represent uncertain portions of the scene efficiently. The negative is that the virtual camera center must lie on the line between the original two camera centers. And it is clearly unrealistic. Woodford et al. [101] use graph-cut stereo method to optimise their new view synthesis

method which simultaneously solves the colour and depth with occlusion modeling of the new view pixels. They use data costs and surface smoothness together to form the equation for addressing the problem. After that, a global graph-cut optimised is used to compute strong convincing results. While this method gives a explicit results, the new features and dynamic objects from video are not considered.

In all of these methods, since they use perspective views, the range of possible viewing directions is limited and the quality of the output depends on whether desirable viewing directions were sampled in the input. The only previous work that we are aware of that is applicable to spherical video sequences was due to Kamali et al. [43]. They attempt to stabilise both rotation and translation of the camera. Of course, to do this exactly would require a dense 3D reconstruction of the scene and new views to be rendered from the 3D model. They use structure-from-motion to compute sparse 3D information for pose estimation. The stabilisation process relies on a mesh-based warp by reprojection of the 3D points. Their SFM pipeline is based on simple adaptations of a perspective image pipeline and does not consider the differences inherent in spherical image geometry. Also while this approach may provide a reasonable sense of stabilisation, it cannot consider occlusions or other nonlinear perspective effects due to the translation of the camera position.

2.4 Conclusions

Feature extraction and matching and the subsequent use of these matches in geometric computer vision (i.e. structure-from-motion) is a very well studied problem with well established techniques that work well for planar images. While spherical images provide a number of compelling applications and advantages, in general, previous work on feature matching and 3D reconstruction has heuristically adapted techniques designed for planar perspective images to the sphere. For example, much previous work projects the spherical image to the plane (often using equirectangular projection) and performs feature extraction and description here. In SFM, steps such as relative pose estimation, the n-point problem, triangulation and bundle adjustment apply methods designed for perspective images.

The drawback of this approach is that systematic noise is unintentionally introduced. For example, flattened spherical images contain distortion, particularly significant near the poles, and introduce artificial boundaries. Distance measures in the plane are also not meaningfully related to distance in the spherical image. For SFM, assumptions that are reasonable for perspective images, such as additive 2D Gaussian noise in the image plane, are not well justified when working with spherical images.

For this reason, our goal in this thesis is to revisit the fundamental problems of feature extraction, matching and geometric reconstruction in a way that respects the spherical geometry of the input images. Having done so, we then address the particular application of view stabilisation and synthesis. Here, there is much less previous work, although a number of approaches show promise for the planar image case.

Chapter 3

Binary Features for Spherical Images on a Geodesic Grid

Spherical images arise in a number of contexts. In computer vision, they are captured by omnidirectional cameras such as catadioptric or fisheye imaging systems or by stitching multiple perspective images. Such panoramic images have many applications where their full coverage of the viewing sphere provides a richer source of image features and increases the likelihood of matching features between views. It is more likely that the same scene point will have been observed because of the increased field of view of a spherical image. Spherical images have taken on a particular importance recently due to the rapid rise in popularity of "360 video". There are now commercially available systems for capturing such video[1] and support for their interactive playback is available in the most popular online video repositories[2]. Outside computer vision, any discretely sampled spherical function may be processed as a spherical image. The most obvious examples occur in Geographic Information Systems (GIS) which model the spatial variation of properties of the surface of the Earth [88].

Image processing for spherical images is less well developed than for their planar counterpart. There are two complexities that arise in processing spherical images. The first is that the nature of the spherical surface must be taken into account when

[1] `https://vr.gopro.com/`

[2] `https://support.google.com/youtube/answer/6178631?hl=en-GB`

performing geometric operations. For example, distance between image points depends on geodesic (great circle) distance as opposed to Euclidean distance for planar images. The second is that for discrete images, the spherical surface must be segmented into discrete pixels which can be efficiently indexed. Moreover, omnidirectional vision is considerably less well developed than computer vision for planar images. So, for example, very few low level feature descriptors have been developed for omnidirectional images, the geometry of spherical cameras in a multiview setting has been little researched and view synthesis techniques on the sphere have received almost no research attention. A common means to address both of these issues is to parameterise a spherical image to the 2D plane via a chosen projection (e.g. equirectangular) and then treat the image as if it were planar. The problem with this approach is that any projection introduces distortion and boundaries. A more attractive alternative is to discretise spherical images on the sphere and perform image processing on the surface of the sphere.

One of the most fundamental operations in image processing is to identify points of interest and to build descriptors of local appearance around an interest point. This enables point-to-point matches to be established between images or a compact description of a scene or object to be constructed. The seminal work in this area is the Scale Invariant Feature Transform (SIFT) of Lowe [62]. While SIFT features remain a benchmark for their distinctiveness and robustness to appearance variation, the computational cost of extracting and matching them has led to them being largely superseded by a class of lighter weight features based on binary descriptors. In this chapter, we take one such feature, BRISK (Binary Robust Invariant Scalable Keypoints) [49], as inspiration and extend them to operate on spherical images. The proposed BRISKS (BRISK on the Sphere) framework includes a novel discretisation of spherical images and scale space, interest point detection, binary feature description and matching. We perform experimental evaluation on both synthetic and real images and test robustness to rotation, camera motion, illumination changes and noise.

The most relevant previous work to ours is the spherical extension of the ORB descriptor (SPHORB) by Zhao et al. [104]. Like us, they work on a hexagonal

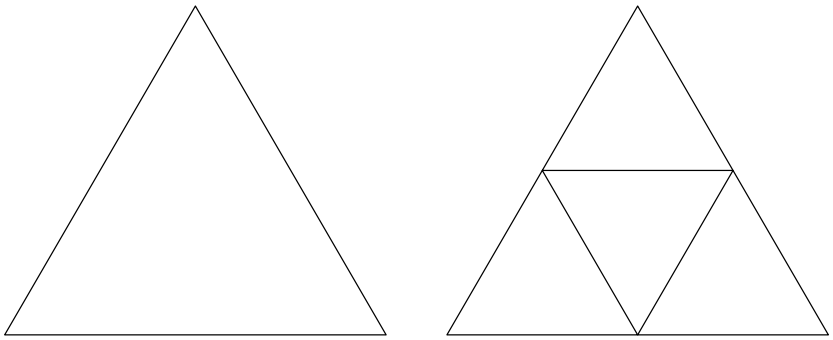


Figure 3.1: Aperture 4 triangle subdivision rule. By adding additional vertices to the middle of each edge, an equilateral triangle is subdivided into four equally sized triangles.

geodesic grid, use our spherical extension of AST [34] and extend a planar binary descriptor to the sphere (ORB [87] in their case, BRISK [49] in ours). However, there are some important differences. First, they transform the spherical geodesic grid to the plane by partitioning the sphere into sets of triangles. This flattening induces distortion and requires additional processing to handle issues at the boundaries of the triangles. All of our processing is intrinsic to the sphere using geodesic distances and concepts from differential geometry (the log map) in order to build a chart around an interest point. Part of their motivation for flattening is efficient indexing and storage. We represent the grid as a mesh and store it using a halfedge data structure which allows efficient access to local neighbourhoods. Second, they use a fixed-size AST pattern for interest point detection. We use different sized patterns that differ by a scale factor of 1.5 in order to detect interest points at intra-octaves. Third, their feature descriptor is built directly on the (flattened) hexagonal grid. This means that irregularities caused by pentagonal pixels cannot be handled and features detected in these regions are discarded. We resample local regions onto a standard pattern in the tangent space meaning that we can handle any pixel structure and sub-pixel refined feature locations. Finally, the BRISK framework that we extend includes sub-pixel position refinement and sub-octave scale refinement, whereas SPHORB does not.

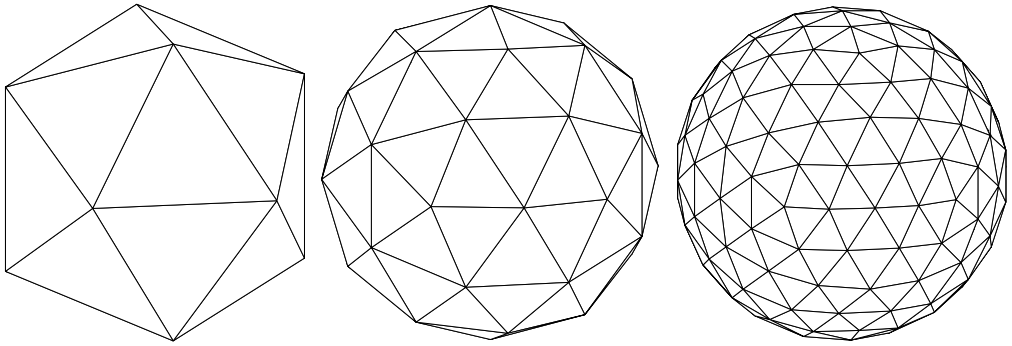


Figure 3.2: Subdivision process for Quaternary Triangular Mesh. Left: icosahedron base surface. Middle: once subdivided. Right: twice subdivided.

3.1 Multiscale Geodesic Grid Representation

In this section we describe how images on the S^2 sphere are discretised, stored, processed in a multiscale manner and represented for local feature detection and description. The reason we need this multiscale geodesic grid representation is that it can overcomes problems of distortion and uneven sampling inherent in panoramic images.

We segment the sphere into discrete pixels via a process of recursive subdivision. Starting with an icosahedron as a base shape, we use an aperture 4 triangle hierarchy. This means each triangle is subdivided into four by adding a vertex to the middle of each edge as shown in Figure 3.1. The newly formed vertices are reprojected to the surface of the sphere. This subdivision provides a triangular segmentation whose surface approximates a sphere with increasing accuracy at each level of subdivision. This is known as a Quaternary Triangle Mesh (QTM) and we show three levels of subdivision in Figure 3.2.

Our image representation is based on hexagonal pixels which are obtained by taking the dual polyhedron of the triangular mesh, i.e. each vertex in the triangular mesh becomes the centre of a hexagonal face. This is shown in Figure 3.3. It is impossible to completely tile a sphere with hexagons. With an icosahedron as the base shape, the triangular subdivision mesh contains 12 vertices with 5 neighbours (regardless of subdivision resolution). Hence, the hexagonal grid at all resolutions contains 12 pixels that are pentagons. These are handled appropriately throughout our pipeline.

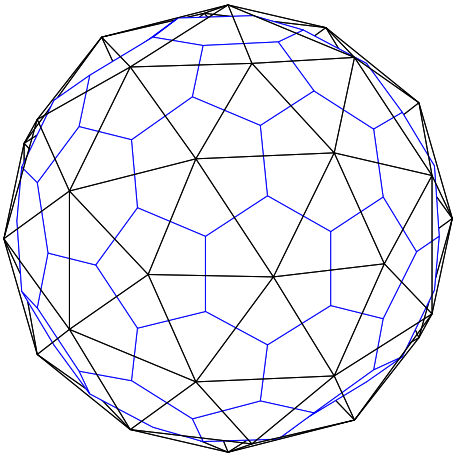


Figure 3.3: Dual polyhedron of the QTM (black) is an aperture 4 hexagon grid (blue).

Table 3.1: Grid resolution versus number of subdivisions

No. subdivisions (s)	No. pixels ($= V_s$)	F_s
0	12	20
1	42	80
2	162	320
3	642	1,280
4	2,562	5,120
5	10,242	20,480
6	40,962	81,920
7	163,842	327,680
8	655,362	1,310,720
9	2,621,442	5,242,880

In a planar, rectangular image, the vertical and horizontal resolution (and hence the number of pixels) can be chosen arbitrarily (although often height and width are set equal and made a power of two to ease multiscale processing). On the other hand a spherical image stored as a geodesic grid has a relatively small set of possible resolutions determined by the subdivision scheme and the level of subdivision. In Table 3.1, we show the number of vertices V_s and faces F_s of the triangular mesh after s subdivisions. The number of pixels in our geodesic grid is equal to V_s (of which 12 are pentagons, the remainder hexagons).

3.1.1 Storage And Indexing

The hexagonal segmentation at subdivision level s and the corresponding spherical image is stored via the corresponding QTM using a triangle mesh $\mathcal{M}_s = (\mathcal{K}_s, \mathbf{V}_s, \mathbf{T}_s)$. The adjacency information is stored in the simplicial complex $\mathcal{K}_s$, whose elements can be vertices $\{i\}$, edges $\{i, j\}$, or faces $\{i, j, k\}$, with indices $i, j, k \in [1 \,..\, V_s]$, where V_s is the number of vertices. Each vertex in the mesh corresponds to a hexagonal pixel. The position of each vertex is stored in the matrix $\mathbf{V}_s \in \mathbb{R}^{3\times V_s}$ which contains the 3D coordinates $\mathbf{v}_{s,i} \in \mathbb{R}^3$ of the respective vertices. Since $\mathbf{v}_{s,i}$ are points on the S^2 sphere, $\|\mathbf{v}_{s,i}\| = 1$. The colour of each hexagonal pixel is stored as a matrix of per-vertex colours, $\mathbf{T}_s \in \mathbb{R}^{3\times V_s}$, associated with the triangle mesh. The colour of the ith pixel is written as $\mathbf{t}_{s,i} \in \mathbb{R}^3$ or $t_{s,i} \in \mathbb{R}$ for a grayscale image. Note that when a new image is loaded, only $\mathbf{T}_s$ changes. The mesh structure, adjacency and neighbourhoods are all fixed for a given subdivision level and so can be precomputed and stored.

Throughout the interest point detection and feature description pipeline, we require efficient indexing of pixel neighbours and local neighbourhoods. To ensure that this is possible, we store the QTM in a half-edge data structure [20]. Ordering of vertices, edges and faces does not affect any of the subsequent processing. This structure allows vertex adjacency queries to be calculated in O(1) time. Hence, in asymptotic terms, accessing local neighbourhoods is the same cost for the geodesic grid as for a 2D planar image.

Immediate Spatial neighbours of a vertex (and hence a hexagonal pixel) i are given by the adjacent vertices in the mesh $\mathcal{N}_1(\mathbf{v}_{s,i}) = \{j | \{i, j\} \in \mathcal{K}_s\}$. We write the n-ring neighbourhood of a vertex as $\mathcal{N}_n(\mathbf{v}_{s,i})$. Neighbours and n-ring neighbourhoods can either be computed on-the-fly using the half-edge structure or precomputed and stored for fast access.

We also define neighbours across scale. This is used for non-maxima suppression and feature scale refinement. A vertex $\mathbf{v}_{s,i}$ at subdivision level s always has a well-defined neighbour at the next finer subdivision level $s + 1$, $\mathcal{N}_1^{s+1}(\mathbf{v}_{s,i}) \in \mathcal{K}_{s+1}$, $|\mathcal{N}_1^{s+1}(\mathbf{v}_{s,i})| = 1$. This is because the vertices of the mesh at subdivision level s are a subset of those at subdivision level $s + 1$. On the other hand, at the next

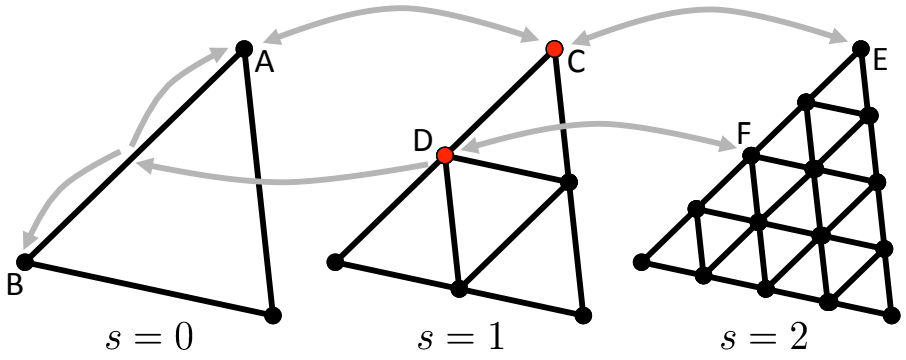


Figure 3.4: Illustration of adjacency across scale. Vertex C has adjacent neighbours at both finer (E) and coarser (A) subdivision. Vertex D has an adjacent neighbour at finer subdivision (F) but we consider it to have two neighbours at coarser subdivision (A and B).

coarser level of subdivision, a vertex can have either one or two closest neighbours $\mathcal{N}_1^{s-1}(\mathbf{v}_{s,i}) \subset \mathcal{K}_{s-1}$, $|\mathcal{N}_1^{s-1}(\mathbf{v}_{s,i})| \in \{1, 2\}$. This is illustrated in Figure 3.4.

3.1.2 Pyramid Generation

For scale invariance, we detect features across scale-space and construct descriptors at an appropriate scale. We therefore require an efficient representation of scale-space. We follow the BRISK framework and discretise to a pyramid comprising n octaves and n intra-octaves. Although the scale discretisation is fairly coarse, we estimate a precise feature scale in continuous scale-space by interpolation.

The subdivision scheme for generating the hexagonal grid lends itself naturally to construction of a spherical image pyramid. The aperture 4 subdivision corresponds to a halving of scale and hence a one octave shift in scale-space. We construct a scale-space pyramid comprising n octaves c_i for $i \in \{0, 1, \ldots, n-1\}$. The octave count is related to the number of subdivisions of the geodesic grid by $i = s_{\max} - s$, where $s_{\max}$ is the number of subdivisions of the maximally subdivided mesh. In our experiments we use $n = 4$ octaves and $s_{\max} = 8$.

The highest resolution image (corresponding to c_0) is created by sampling a panoramic image onto the maximally subdivided mesh using bilinear interpolation. Successive octaves are formed by area-weighted averaging and subsampling. As shown in Figure 3.5, a down-sampled hexagonal pixel is computed as a weighted average of 7 hexagons, with unit weight for the hexagon contained entirely within

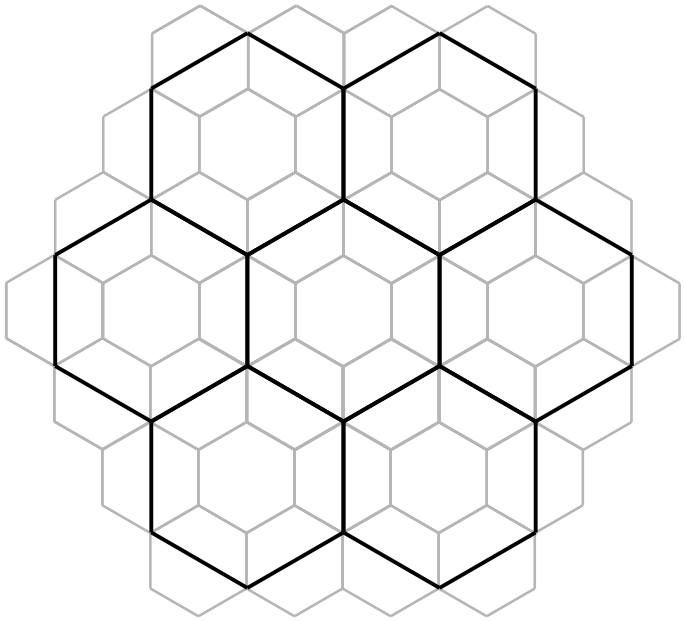


Figure 3.5: Visualisation of hexagonal pixels after triangular subdivision. Hexagons drawn in black are at subdivision level s, those in grey are at subdivision level $s+1$.

the down-sampled hexagon and a weight of 0.5 for the 6 that are half contained. For pentagonal pixels, the same approach is used but there will be only 5 neighbours that are again half contained within the pixel.

To increase resolution in scale, it is desirable to also include intra-octaves. We define n "virtual" intra-octaves d_i that are located in between octaves c_i and c_{i+1}. They are virtual in the sense that we do not compute and store images at the intermediate sizes. Instead, we use an interest point detector that is scaled 1.5 times larger and, when sampling image data for descriptor construction, apply an appropriate scaling to pixel coordinates in the tangent plane. There are two reasons for using virtual intra-octaves. First, there is an efficiency saving in not having to compute and store the intra-octave images. Second, the triangular subdivision scheme cannot produce intra-octaves of appropriate resolution (a scale 1.5 times different to the aperture 4 subdivision can be obtained by an aperture 9 subdivision but the dual polyhedron is not then tile-able with hexagons). The scale t at an octave or intra-octave is given by $t(c_i) = 2^i$ and $t(d_i) = 2^i \cdot 1.5$.

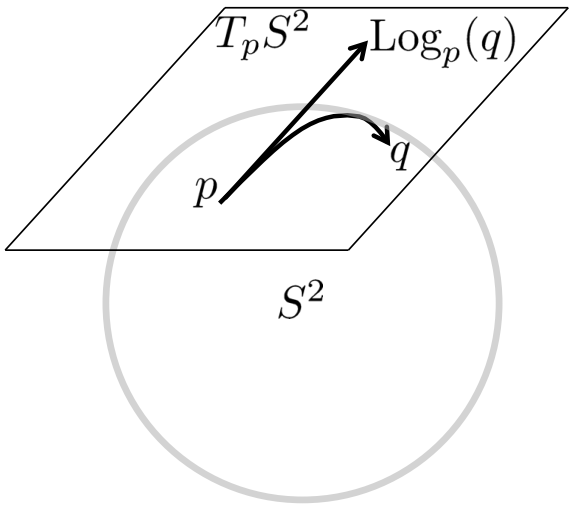


Figure 3.6: The log map on the S^2 sphere.

3.1.3 Tangent Space Representation

For sub-pixel position refinement and sampling data for descriptor construction, it is useful to transform pixels in the spherical image to a local 2D representation. The S^2 sphere is a Riemannian manifold and hence it is possible to build a chart to parameterise the manifold locally. Since our features describe local appearance only, the locality assumption is reasonable.

We build local charts in the tangent plane to the sphere. We do so using the Riemannian log map, $\mathrm{Log}_p : S^2 \to T_pS^2$, which transforms from the sphere to the tangent plane at a point $p \in S^2$. The log map linearises the sphere around the base point in such a way that Euclidean distances from the base point in the tangent plane are equal to geodesic distances, i.e. $d_g(p, q) = \|\mathrm{Log}_p(q)\|$. See Figure 3.6 for a visualisation of the log map.

For the sphere, the log map of a point $q \in S^2$ at base point $p \in S^2$ is given by:

$$\mathrm{Log}_p(q) = \frac{\arccos(\langle p, q\rangle)(q - \langle p, q\rangle p)}{\|q - \langle p, q\rangle p\|}, \tag{3.1}$$

Since we deal with unit vectors corresponding to an embedding of the sphere in Euclidean space, the inner product is simply the scalar product of the embedded vectors:

$$\langle p, q\rangle = \mathbf{p} \cdot \mathbf{q}, \tag{3.2}$$

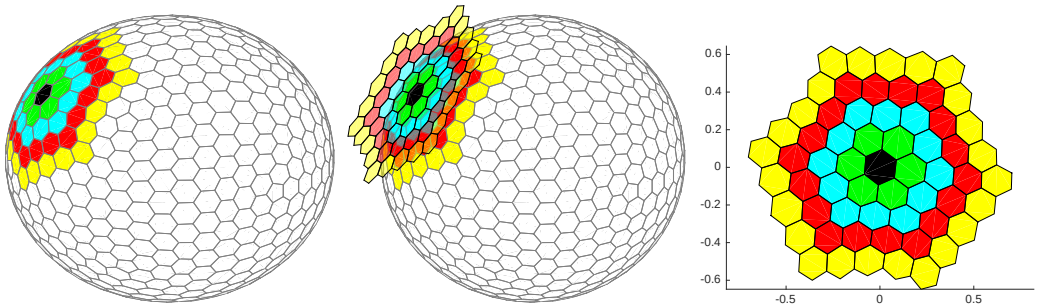


Figure 3.7: The log map applied to a hexagonal geodesic grid at subdivision level $s = 3$. (a) An interest point (black) and its 4-ring neighbourhood on the sphere. (b) The local neighbourhood transformed to the tangent plane via the log map. (c) Local 2D coordinate system.

where $\mathbf{p} = \phi(p) \in \mathbb{R}^3$, $\mathbf{q} = \phi(q) \in \mathbb{R}^3$ and ϕ is an arbitrary embedding. The resulting tangent plane vectors are also subject to this arbitrary embedding. So, to yield 2D coordinates in a standardised coordinate system, we apply a rotation to align the plane with the x-y plane and discard the z coordinate as follows:

$$\mathbf{x}_p(q) = \mathbf{T}\phi(\mathrm{Log}_p(q)) \in \mathbb{R}^2, \tag{3.3}$$

where the transformation is given by

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{R}(\boldsymbol{\alpha}, \theta), \tag{3.4}$$

where $\mathbf{R}(\boldsymbol{\alpha}, \theta) \in SO(3)$ is a rotation matrix that depends upon $\mathbf{p}$ with rotation axis

$$\boldsymbol{\alpha} = \mathbf{p} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_y \\ -p_x \\ 0 \end{bmatrix}, \tag{3.5}$$

and rotation angle $\theta = \arccos(p_z)$. The inverse of the log map is the exponential map, $\mathrm{Exp}_p : T_pS^2 \to S^2$. This transforms a tangent vector $v \in T_pS^2$ at base point p to the sphere:

$$\mathrm{Exp}_p(v) = p\cos\theta + \frac{v\sin\theta}{\theta}, \qquad \theta = \|v\|. \tag{3.6}$$

In Figure 3.7 we visualise the application of the log map to our hexagonal geodesic

grid. In (a) we show an interest point (coloured black) and its 4-ring neighbourhood. In (b) we show the neighbourhood transformed to the tangent plane at the interest point via the log map (note that the tangent plane is embedded in 3D space in a way that depends on the base point). Finally, in (c) we show the 2D coordinate system after applying the transformation in Equation 3.3 to remove the effect of the embedding.

3.2 Interest Point Detection

We use the FAST corner detector as the basis of our interest point detection. We begin by describing how the accelerated segment test is extended to operate on a spherical geodesic grid. We then describe how we apply non-maxima suppression both spatially and across scale in our pyramid representation. Finally, we show how we perform position refinement in space and scale, enabling a continuous estimate of feature position and scale.

3.2.1 Accelerated Segment Test On A Geodesic Grid

In the original FAST approach, a circle of radius 3.4 was discretised onto a square pixel lattice using Bresenham's algorithm. Subsequently, alternate patterns have been considered, approximating circles of radius 1, 2 and 3, which contain 8, 12 and 16 pixels respectively. To test for the existence of a corner, a consecutive sequence of length k pixels must be brighter or darker than the central pixel by a threshold t_{corner}. In practice, we manually select this threshold such that the desired numbers of features are obtained. Originally, k was chosen as 12, corresponding to a 45° corner. This value of k was also computationally efficient since candidate corners could be discounted after testing as few as 3 pixels. Subsequently, it has been found that optimal performance occurs when k is chosen to be the smallest value that will not detect edges. i.e. if the sequence is of length m then $k = \lceil (m+1)/2 \rceil$.

We extend this approach to operate on discrete geodesic grids composed of triangular or hexagonal pixel lattices. Note that we presented an early version of this idea in [34] and the same idea was subsequently used by Zhao et al. [104]. In general,

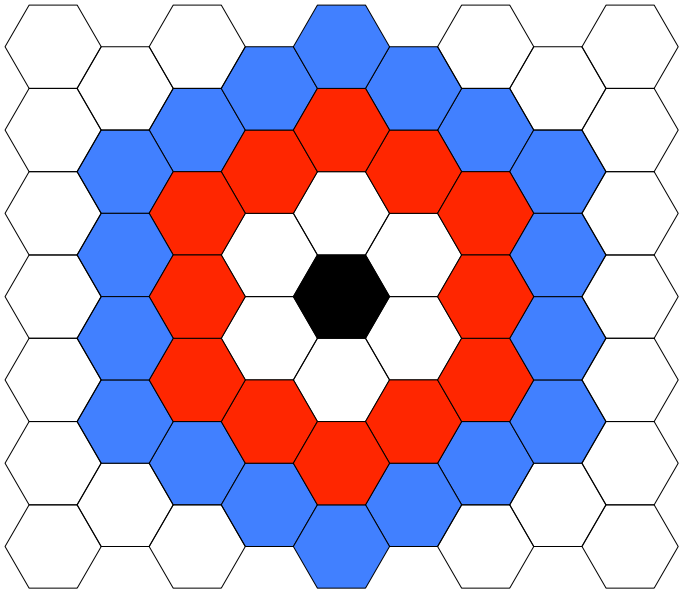


Figure 3.8: AST pattern for hexagonal lattice. Radius 2 shown in red, radius 3 in blue for corner test at black pixel.

curves and circles drawn on a hexagonal lattice appear smoother because of the improved angular resolution afforded by having six equidistant neighbours compared to only four in a square lattice [72]. The FAST patterns on a square lattice are only invariant to rotations of 90°, 180° and 270°, whereas the hexagonal patterns are invariant to rotations of 60°, 120°, 180°, 240° and 300°.

According to the papers [34] [104], it has been proved that the radius 3 corner detector is the best one which returns the highest repeatability. We consider circles of radius 2 and 3 on a hexagonal lattice, as shown in Figure 3.8, containing 12 and 18 pixels respectively. Note that the larger pattern differs in scale (radius) by a factor of 1.5 from the smaller one. We use this fact to detect interest points at intra-octaves. To detect interest points at octave c_i, we use the radius 2 pattern on the geodesic grid at subdivision level $s = s_{\max} - i$. To detect interest points at intra-octave d_i we use the radius 3 pattern, on the geodesic grid at the same subdivision level.

There are a number of special cases to deal with due to the impossibility of completely tiling a sphere with hexagons. Irrespective of level of subdivision, there are 12 pixels with pentagonal shape and hence 5 neighbours. In the radius 2 case, corners centred on pentagonal pixels have circles containing 10 pixels and we test for sequences of length 6. Corners centred on pixels adjacent to a pentagonal pixel

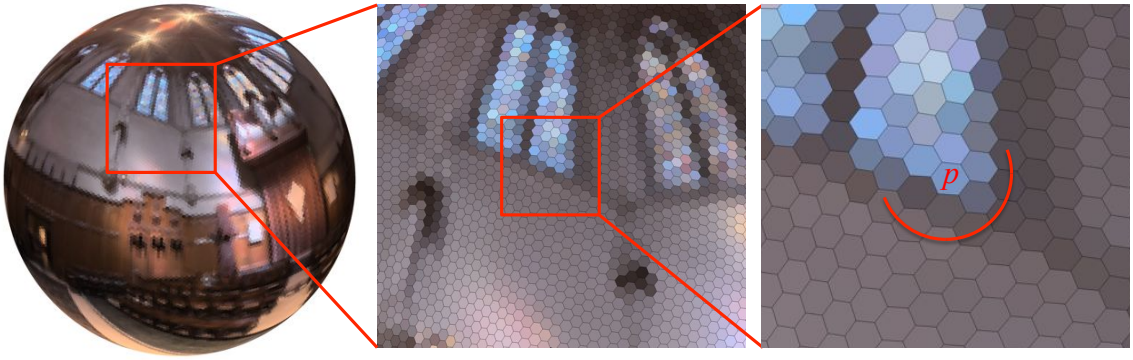


Figure 3.9: An example of a pixel passing the hexagonal 7-12 AST.

have circles of 11 pixels and again we test for sequences of length 6. Finally, the radius 3 case has special cases of circles containing 15, 16 or 17 pixels which we test for sequences of length 8, 9 and 9 respectively. Note that the radius 2 circle shares the same number of pixels as the radius 2 circle on a square lattice. This means that the AGAST [67] decision tree for 12 pixels can be used on a hexagonal lattice without modification.

In Figure 3.9 we show an example of a pixel passing the hexagonal 7-12 test. The pixel labelled p is regarded as a corner because the 7 pixel sequence (marked in red) lying on the 12 pixel, radius 2 circle are all darker than p by more than the threshold t_{corner}.

FAST score is to evaluate the strength of the corner. We firstly select the initial FAST score carefully in terms of the tradeoff between the corners robustness and quantity. Then we compute the corner saliency for corners by finding the largest value for the threshold $t_{\text{score}}(p)$ at which the point is still considered a corner. This is done efficiently using binary search on the interval $[t_{\text{corner}}, 1]$.

3.2.2 Non-maxima Suppression

We use the feature saliency score to apply non-maxima suppression. For every detected corner, we compute the corner saliency for adjacent pixels (even if they were not detected as corners) at the same scale. We also compute corner saliency for neighbours across scale (if there are two neighbours at the next coarser scale, we take the one with higher saliency as the neighbour). We require a feature to

have maximum saliency amongst all of its neighbours and remove any that do not. The computed saliency scores are saved as they are used subsequently for position refinement.

3.2.3 Position And Scale Refinement

We refine the spatial position of each feature to sub-pixel accuracy and estimate a continuous scale. If a feature has been detected at vertex $\mathbf{v}_{s,i}$, we fit a 2D quadratic function in a least squares sense to the set of tangent plane coordinates and FAST scores, $\mathcal{F}(\mathbf{v}_{s,i})$, of the 1-ring neighbourhood of $\mathbf{v}_{s,i}$:

$$\mathcal{F}(\mathbf{v}_{s,i}) = \big([0\ 0]^T, t_{\text{score}}(\mathbf{v}_{s,i})\big) \cup \Big\{\big(\mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,j}), t_{\text{score}}(\mathbf{v}_{s,j})\big) \,|\, j \in \mathcal{N}_1(\mathbf{v}_{s,i})\Big\}. \tag{3.7}$$

We compute the position and FAST score of the maxima of the quadratic function in closed form and transform the resulting point back to the sphere using the inverse of the transformation applied in Equation 3.3 followed by the exponential map. We perform the same sub-pixel position refinement on neighbourhoods at adjacent scales, i.e. $\mathcal{F}(\mathbf{v}_{s+1,i})$ and $\mathcal{F}(\mathbf{v}_{s-1,i})$. For these neighbourhoods, it is possible that the central pixel is not the maximum or that the fitted quadratic is not even concave (and hence has no maximum). In these cases, we simply take the position and score of the pixel in the neighbourhood with maximum score.

We now have sub-pixel refined positions and scores for the detected feature and for scales either side of the detected feature. Finally, we estimate a continuous scale by fitting a parabola to the scores as a function of scale, compute the scale of the maximum of the parabola and interpolate the refined position between the position at the original scale and that in the scale direction of the parabola maximum. The refined scale and position are assigned to the feature.

3.3 Descriptor Extraction

The final step in the pipeline is to generate feature descriptors for the detected features. Following the BRISK framework, this is done in four steps. First, we

make an estimate of the local gradient direction in order to assign a direction to the feature. Second, the local neighbourhood is rotated to normalise this direction (introducing rotation invariance). Third, the local neighbourhood is sampled onto a fixed sampling pattern. Finally, the binary descriptor is generated based on a fixed set of intensity comparisons on this descriptor.

3.3.1 Orientation Normalisation

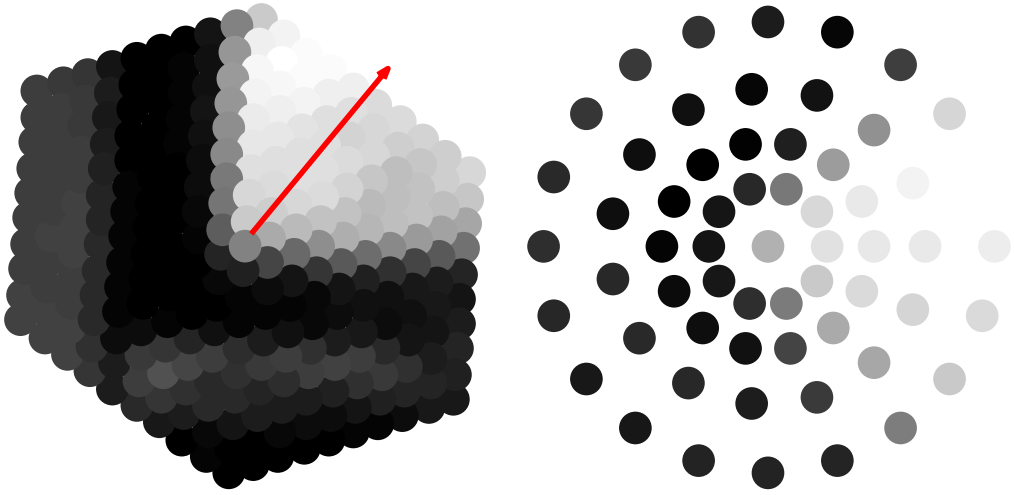
We begin by computing an estimate of the local gradient direction for each feature. To do so, we use the 9-ring neighbourhood around the feature point at the octave in which the feature was detected. Suppose that a feature was detected at vertex $\mathbf{v}_{s,i}$. The gradient according to a pair of pixels $\mathbf{v}_{s,j}$ and $\mathbf{v}_{s,k}$, $j,k \in \mathcal{N}_{1..9}(\mathbf{v}_{s,i})$, in the neighbourhood of the feature is computed in the tangent plane by:

$$\mathbf{g}(\mathbf{v}_{s,j},\mathbf{v}_{s,k}) = \mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,k}) - \mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,j}) \cdot \frac{t_{s,k} - t_{s,j}}{\|\mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,k}) - \mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,j})\|^2}. \tag{3.8}$$

Where $t_{s,k}$ and $t_{s,j}$ are the intensity values at these pixels $\mathbf{v}_{s,j}$ and $\mathbf{v}_{s,k}$ respectively. The estimate of the overall characteristic direction:

$$\mathbf{g} = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \frac{1}{|\mathcal{L}|} \sum_{(j,k)\in\mathcal{L}} \mathbf{g}(\mathbf{v}_{s,j},\mathbf{v}_{s,k}) \tag{3.9}$$

is computed by averaging the local gradient direction estimates between all "long distance" pairs. That is the set of pairs, $\mathcal{L} = \{(j,k) \mid j,k \in \mathcal{N}_{1..9}(\mathbf{v}_{s,i}) \wedge \|\mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,k}) - \mathbf{x}_{\mathbf{v}_{s,i}}(\mathbf{v}_{s,j})\| > \delta_{\min}\}$, whose Euclidean distance is greater than a threshold, $\delta_{\min}$. We choose the threshold to be the radius of the neighbourhood, defined as the point on the 9-ring that is minimum distance from the base point. The motivation for using long distance pairs in the original BRISK framework is that local gradients annihilate each other and therefore do not influence the global gradient estimate. In Figure 3.10(a) we show an example of the intensities in the neighbourhood around a feature and the estimated characteristic direction. Orientation is normalised by applying a rotation of angle $\arctan2(g_y, g_x)$ to the tangent plane coordinates of the pixels in the local neighbourhood.



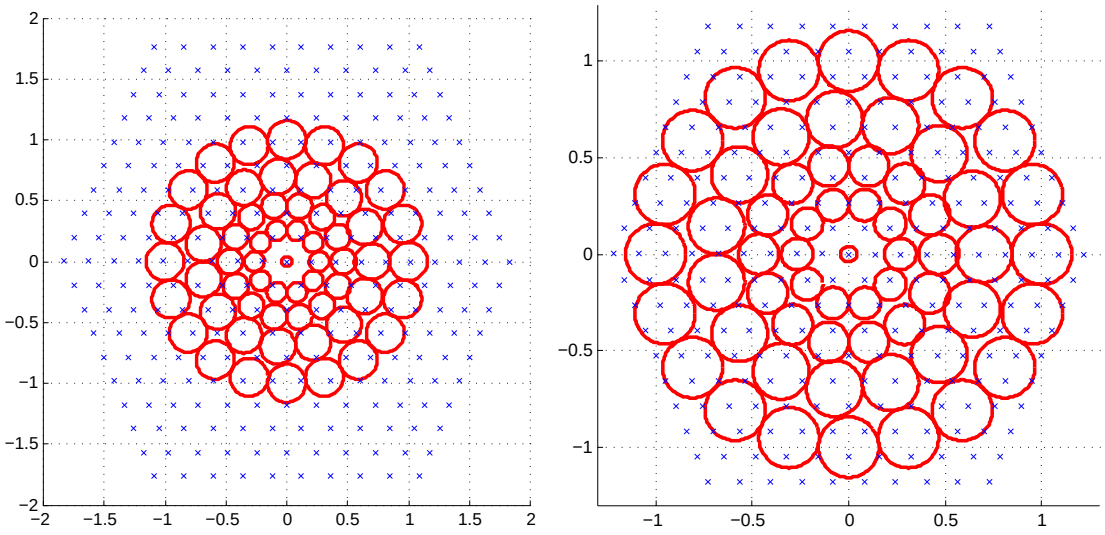
(a) Characteristic orientation. (b) Sampled intensities.

Figure 3.10: Orientation normalisation and sampling. On the left we show a neighbourhood around a detected feature. The estimated characteristic direction for the feature is indicated by the red arrow. On the right we show the intensities after sampling the rotated neighbourhood onto the standard pattern.

3.3.2 Sampling

In order to construct the feature descriptor, we sample the rotation-normalised intensities onto a standard pattern. This serves a number of purposes. First, it allows us to deal with any irregularities in the pixel structure caused by pentagonal pixels. Second, the sampling uses Gaussian smoothing which reduces aliasing effects. Third, it provides a standardised set of image locations from which a fixed set of intensity comparisons can be used to create the feature descriptor.

We use the same pattern as in the BRISK framework. However, note that we are operating in the tangent space to the sphere rather than on a 2D image. The sampling pattern is shown in Figure 3.11 and comprises 60 points sampled inside the circle of the radius 1 neighbourhood, $\mathbf{s}_1, \ldots, \mathbf{s}_{60} \in \mathbb{R}^2$. Sample points are plotted as red circles and the radius of the circle corresponds to the standard deviation, $\sigma_1, \ldots, \sigma_{60}$, of the Gaussian used for smoothing at that sampling point. We denote the smoothed, sampled intensity at sample point $\mathbf{s}_i$ as $I(\mathbf{s}_i, \sigma_i)$. After rotation normalisation and sampling, the image region shown in Figure 3.10(a) results in the sampled intensities shown in Figure 3.10(b). It is from these intensities that we



(a) Sampling octave feature.

(b) Sampling intra-octave feature.

Figure 3.11: The sampling pattern consists of points (shown as red circles) distributed over a radius 1 circle. The 9-ring neighbourhood around a feature is scaled onto the pattern as shown (shown as blue crosses). Features detected at octaves and intra-octaves have a scale applied that differs by a factor of 1.5.

compute the intensity comparisons to build the feature descriptor.

Pixel neighbourhoods are scaled according to the scale of the detected feature prior to sampling. Since we do not explicitly compute intra-octave images, intra-octave features are scaled by a factor of 1.5 on the octave image at which they were detected. This is illustrated in Figure 3.11.

3.3.3 Descriptor Generation

The bit string descriptor is built using intensity comparisons on a set of short-distance pairs $\mathcal{S}\{(i,j) \mid i,j \in \{1,\ldots,60\} \wedge \|\mathbf{s}_i - \mathbf{s}_j\| < \delta_{\max}\}$. The motivation for only using comparisons between pairs of locations that are spatially close is that feature similarity then only requires brightness variations to be locally consistent. This reduces sensitivity to spatially varying illumination. The short distance threshold, $\delta_{\max}$, is chosen to yield a bit string of the desired length. We follow BRISK [49] and BRIEF64 [15] and use 512 bit descriptors. For our pattern, this corresponds to a value of $\delta_{\max} = 0.6378$. Pairs are evaluated in an arbitrary but fixed order,

$S_1 \in \mathcal{S}, \ldots, S_{512} \in \mathcal{S}$, yielding the bit string descriptor b, where

$$b_i = \begin{cases} 1 & \text{if } I(\mathbf{s}_j, \sigma_j) < I(\mathbf{s}_k, \sigma_k) \\ 0 & \text{otherwise} \end{cases} \quad \text{where } S_i = (j, k). \tag{3.10}$$

The dissimilarity between two feature descriptors is given by their Hamming distance. Since this amounts to nothing other than a bitwise XOR followed by a bit count, this can implemented very efficiently.

3.4 Experimental Results

We evaluate our proposed BRISKS features on both synthetic rendered images and real images. The use of synthetic images allows us to arbitrarily control illumination and also means that ground truth correspondences can be computed for images taken from different viewpoints. We include comparison to two previous methods. The first applies classical planar SIFT to unwrapped equirectangular panoramic images, as done by [28, 89]. The second is the spherical extension of SIFT introduced by Cruz-Mota et al. [19]. We use standard feature evaluation metrics [73, 74] adapted to the sphere. Feature detection performance is measured using repeatability [74]. Feature description and matching performance is measured using 1−precision and recall curves, giving a metric that is invariant to the matching threshold. The synthetic images are rendered using the spherical camera sensor in the Mitsuba renderer [42]. We use the Babylonian City scene from the Medieval City collection (courtesy of Johnathan Good). We render both panoramic images and depth maps. The real images we use are a 10 image subset of the SUN360 dataset [102], as shown in Figure 3.12, spanning a range of different scene types. We resize input images for SIFT and SSIFT so that the number of pixels is approximately equal to the number of pixels in our finest subdivision mesh.

3.4.1 Detector Repeatability

The repeatability score measures the ability of the interest point detector to detect features at image points in different images corresponding to the same scene location.

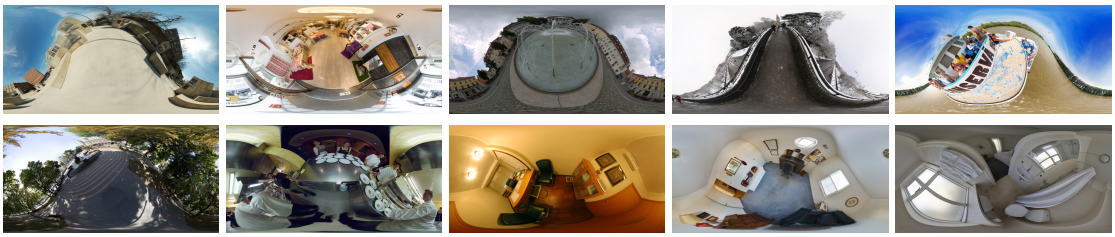


Figure 3.12: Subset of the SUN360 dataset [102] used in our experiments.

	Rotation and Translation	Illumination
Ours	**0.93** (419)	**0.64** (414)
SSIFT [19]	0.65 (412)	0.49 (414)
SIFT [28, 89]	0.57 (392)	0.41 (396)

Table 3.2: Repeatability on synthetic images. Second column shows average results for camera rotation and translation. Third column shows results for changing illumination. Average number of detected features shown in brackets.

For each detected interest point in the first image, we project the point into the scene, project it back into the second image and check whether an interest point has been detected at the corresponding location. Our criteria for a successful repeated detection is simply to measure the spherical distance between the reprojected and closest detected interest points and test whether it smaller than a threshold of 2°. The repeatability score is the ratio between the number of repeat detections and the smaller of the number of interest points in the pair of images.

In the synthetic images, we use the ground truth depth map to project image points into the scene allowing us to compute correspondence even with viewpoint changes. For the real images we only apply rotations and add noise and hence simply need to rotate image points to test for repeated detections. For all three methods, we adjust the detection threshold to yield similar numbers of interest points in each image.

In Table 3.2 we show repeatability results for the synthetic images. The second column shows results for pose changes (i.e. the rotation and translation of the camera differs between views). We rendered 6 images in which the camera follows a linear trajectory with a distance of 50 units along the x axis between each image. We also apply a rotation about the z direction of 60° between each image. See Figure 3.13 for example images. We measure repeatability between pairs of consecutive images

	Rotation	Noise		
		10dB	15dB	20dB
Ours	**0.94** (384)	**0.90** (379)	**0.93** (383)	**0.93** (385)
SSIFT [19]	0.86 (391)	0.76 (389)	0.79 (379)	0.83 (374)
SIFT [28, 89]	0.70 (391)	0.65 (371)	0.67 (378)	0.66 (384)

Table 3.3: Repeatability on SUN360 images. The second column shows repeatability under rotation. The third to fifth columns show results for varying levels of additive noise (noise level shown as signal to noise ratio). Average number of detected features shown in brackets.

and show results averaged over all pairs. In the third column we show results for changing illumination. For three viewpoints, we render images with illumination simulating 12 noon and 5:00 pm in the afternoon with a 60° rotation about the z axis between images. See Figure 3.14 for example images. We measure repeatability between each pair and average results over all pairs.

In Table 3.3 we show repeatability results for the real images. The second column shows results for rotation. For each scene we generate six images differing by rotations of 60° about the z axis. Columns three to five show results with additive noise. We add Gaussian noise to each image with the variance selected to obtain a desired signal to noise ratio. See Figure 3.15 for example images.

3.4.2 Descriptor Precision And Recall

We evaluate our descriptor on the same sets of images as for the detector evaluation. We do so using the standard 1−precision and recall metrics. For each feature in the first image we find the nearest neighbour feature descriptor in the second image (using Hamming distance for our descriptors and Euclidean distance for SIFT and SSIFT). We vary the matching threshold and plot how precision and recall vary. Recall is the ratio between the number of correct matches and the number of feature pairs that have correspondences. 1−precision is the ratio between the number of incorrect matches and the total number of matches. Correct matches are defined in the same way as in the repeatability score.

Figure 3.16 shows the averaged curves for the synthetic images. Rotation and translation results are on the left, illumination results on the right. Note that these

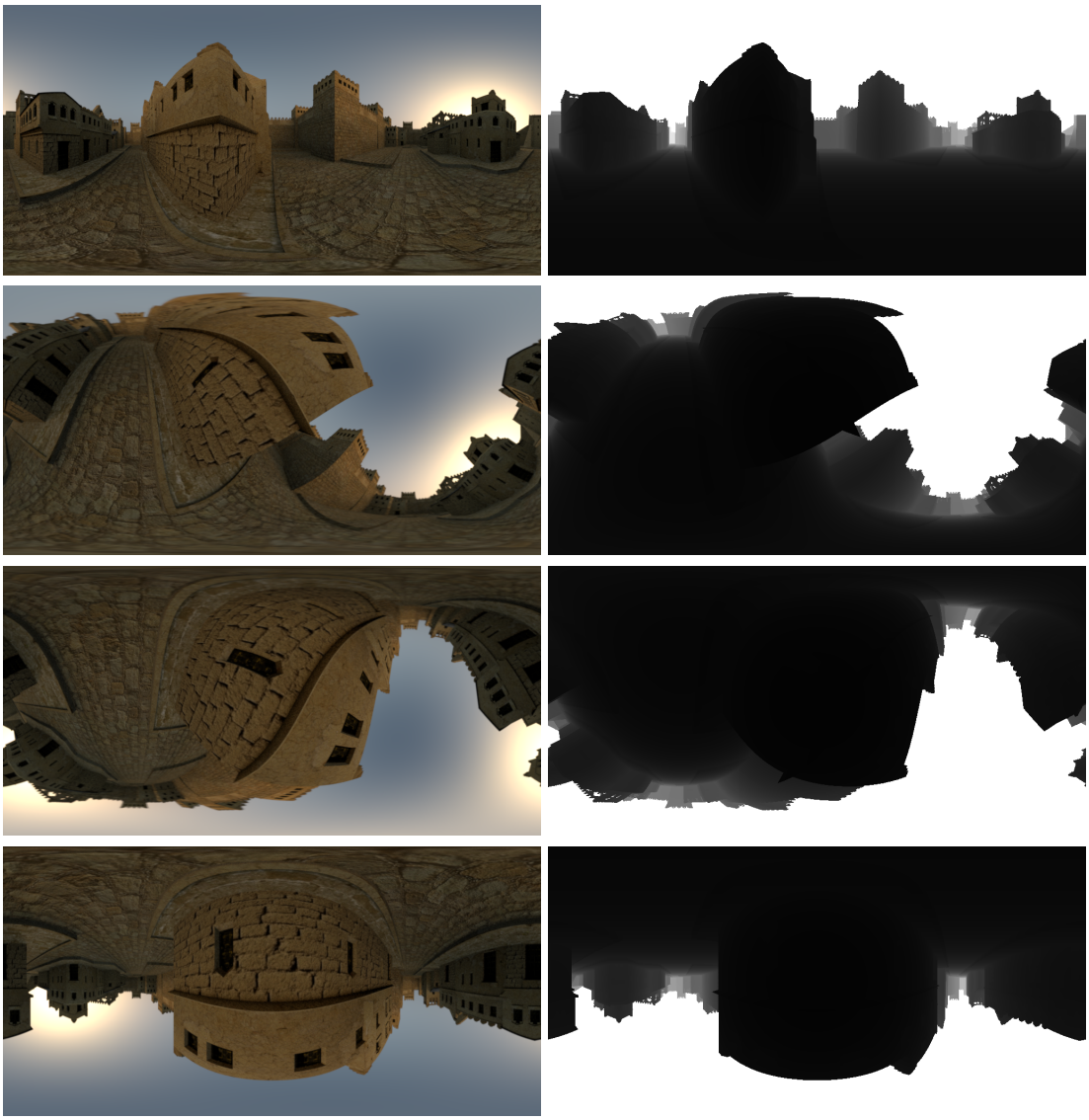


Figure 3.13: Rendered synthetic images for changing camera rotation and translation (left) and their corresponding ground truth depth maps (right)

are very challenging images since there are repeated texture patterns (e.g. bricks and paving) and the changes in pose and illumination cause dramatic changes in appearance. Figure 3.17 shows the averaged curves for the real images. Results for rotation only are shown in the top left, results for varying additive noise are shown top right and in the second row. These images are less challenging than the synthetic ones since camera position and illumination is fixed.

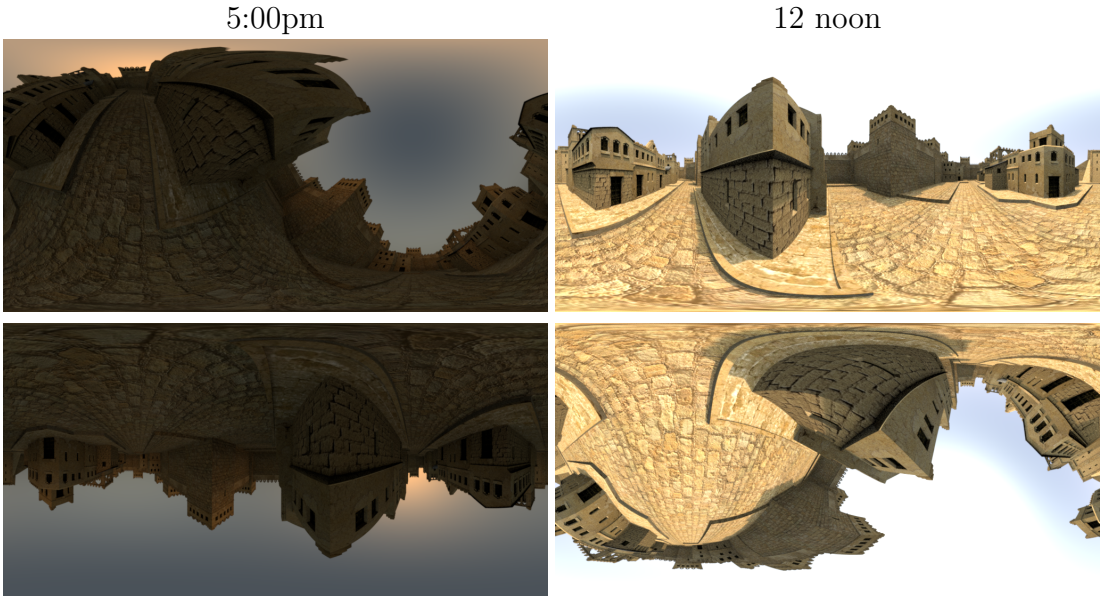


Figure 3.14: Rendered synthetic images for changing illumination and rotation. Each row shows a pair rendered at the same position.

3.5 Conclusions

In this chapter we have proposed a binary feature for spherical images. This requires rethinking a number of fundamental aspects of a local feature such as how an image is discretised and stored, how to build a discrete scale space and how to perform feature detection and description without having to project the spherical image to a planar embedding. Regarding the computational complexity, all local operations are $O(1)$ - i.e. projecting local neighbourhood to tangent plane, computing orientation, resampling - so long as neighbourhood size is fixed. FAST feature detection is $O(n)$ where n is number of pixels. All feature descriptor extraction steps are $O(f)$ where f is number of features.

Despite having significantly lower computational complexity than SIFT-like methods and a descriptor that is 16 times smaller (for 128D SIFT descriptor with double precision floats, or 8 times smaller for single precision), across all experimental conditions on both synthetic and real images, our method significantly outperforms SSIFT which, in turn, outperforms the naive application of SIFT to equirectangular images. We believe these differences in performance are primarily caused by the following reasons. First, because of our geodesic grid representation, we do not

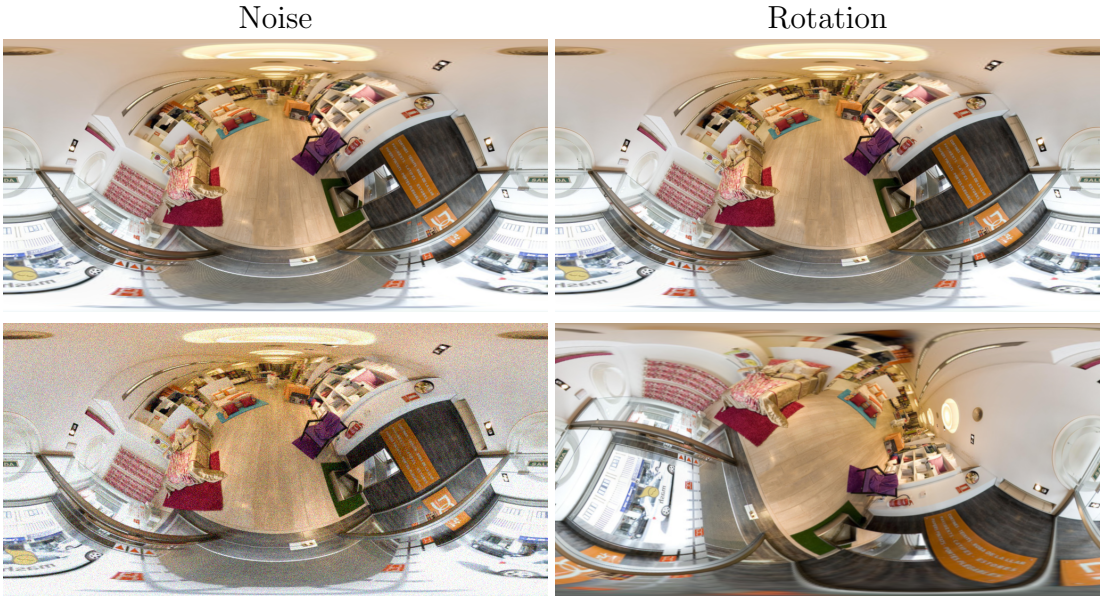


Figure 3.15: For the real images (first row) we add noise (second row left) and apply rotations (second row right).

suffer from irregular sampling of the images and inherent changes in appearance between different views. This is particularly important in comparison to planar SIFT where the equirectangular mapping introduces very large, view dependent distortions. Second, binary features have already been shown to outperform SIFT in the planar case [49] and so it is not surprising that the same performance gain is seen in the spherical case. Finally, SIFT on the sphere builds a spherical scale space using the Spherical Fourier transform which introduces an aliasing effect. Our scale space pyramid is built in a manner that is analogous to a classical image pyramid and does not introduce aliasing effects.

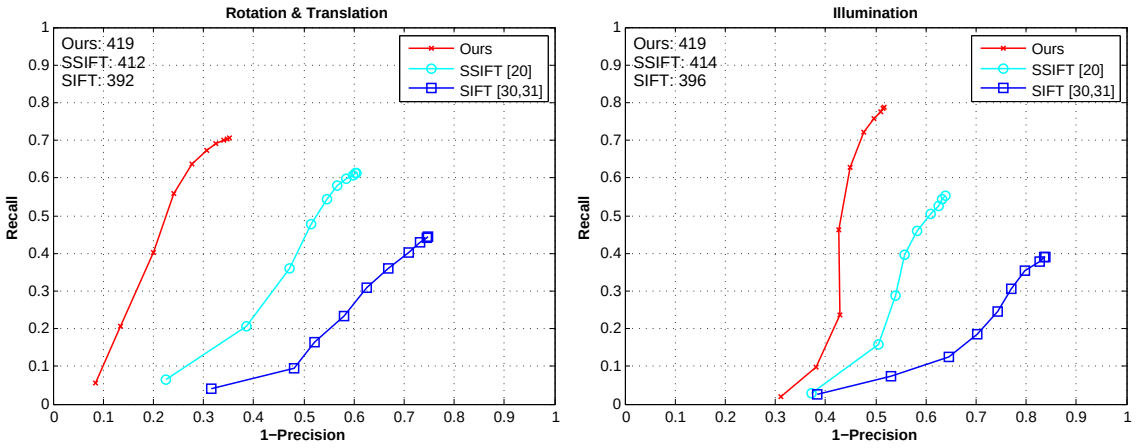


Figure 3.16: 1−precision and recall curves for the synthetic images. The numbers of correspondences are shown on the left corner in each figure.

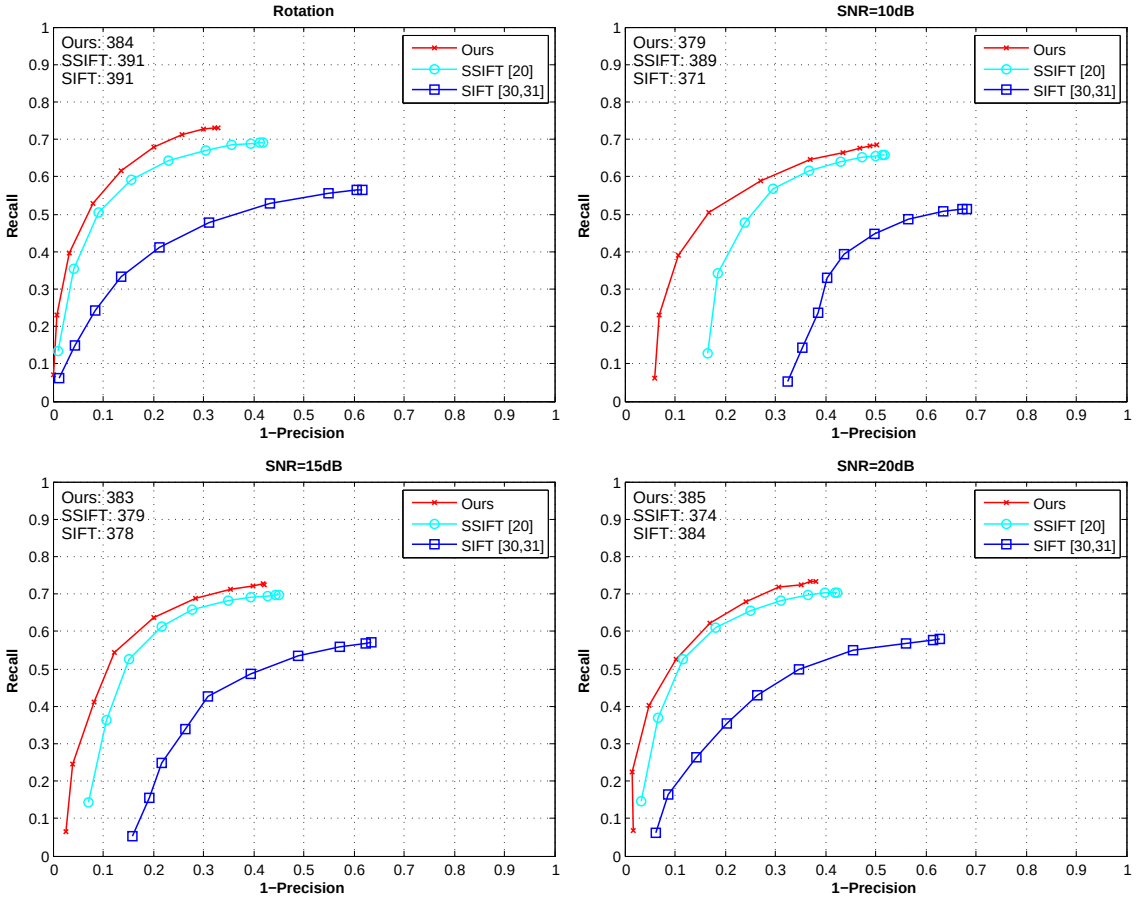


Figure 3.17: The 1-precision curves of real data for our method, SSIFT and SIFT. The numbers of correspondences are shown on the left corner in each figure.

Chapter 4

Structure-from-motion in Spherical Video using the von Mises-Fisher Distribution

Interactive viewing of spherical video provides an experience that is more immersive [98] than narrow field of view video since a virtual viewing direction can be chosen and varied during playback. The goal of 'surround video' [78] has recently reached the mainstream through the availability of 360 cameras, support for 360 playback in web video services and the release of consumer virtual reality head mounted displays.

The benefits of spherical video and the sense of immersion are particularly felt in the case of first person (or egocentric) sequences. First person video captured with traditional narrow-field-of-view cameras suffered from rapid changes in viewing direction leading to a highly disorientating experience for the viewer. Spherical video offers a potential solution to this problem since all possible viewing directions are captured. However, as the pose of the spherical camera changes, so too does the virtual viewing direction. This is also disorienting for the viewer and can even lead to motion sickness as they lose control over the direction in which they are looking.

This motivates the need for algorithms that can robustly stabilise spherical video captured in the real world with large and rapid changes in camera pose. From a computer vision perspective, spherical video is in some ways attractive. The greatly increased field of view substantially increases the chance of observing features

seen previously and the spherical camera projection model is particularly simple (it does not require intrinsic calibration). However, the adaptation of computer vision techniques such as feature extraction and structure-from-motion (SFM) to spherical images has received limited attention and the majority of previous work has heuristically adapted techniques developed for perspective images.

4.1 Preliminaries

The projection of a 3D world point, $\mathbf{w} = [u\ v\ w]^T$, to a point in a spherical image, $\mathbf{x} = [x\ y\ z]^T$ (with $\|\mathbf{x}\| = 1$), for a spherical camera whose coordinate system differs from world coordinates by a rotation $\boldsymbol{\Omega} \in SO(3)$ and translation $\boldsymbol{\tau} \in \mathbb{R}^3$ is given by the spherical camera model:

$$\mathbf{x} = \mathbf{spherical}[\mathbf{w}, \boldsymbol{\Omega}, \boldsymbol{\tau}] = \frac{\boldsymbol{\Omega}\mathbf{w} + \boldsymbol{\tau}}{\|\boldsymbol{\Omega}\mathbf{w} + \boldsymbol{\tau}\|}. \tag{4.1}$$

Note that this model is quite general (it is applicable to catadioptric and dioptric cameras or spherical panoramas obtained via stitching [71, 92]) and abstracts away from how the input images were acquired.

Unlike a pinhole (perspective) camera, a spherical camera has no intrinsic parameters. Yet there is a close relationship between a pinhole and spherical camera. In a pinhole camera, projection involves rescaling rays such that they lie on the image plane. In a spherical camera, rays are normalised to lie on the unit sphere. Hence, pinhole image points are 2D coordinates, usually represented using homogeneous coordinates. Spherical image points are unit vectors in $\mathbb{R}^3$ (i.e. points on the S^2 sphere).

4.1.1 Noise Model

The estimated position of a feature in a spherical image is noisy for reasons including sensor noise, sampling issues (exacerbated by stitching or warping to obtain the full spherical image) and inaccuracies of the feature detector. The vMF distribution is a parametric distribution for directional data and has properties analogous to those

of the multi-variate Gaussian distribution for data in $\mathbb{R}^n$. Hence, where the assumption of additive, normally distributed noise with spherical covariance is assumed for perspective projection of 3D points to a 2D image plane, so the vMF distribution is appropriate for 3D points projected to the unit 2-sphere.

For the 2-sphere, the PDF of the vMF for the random unit vector $\mathbf{x}$, $\|\mathbf{x}\| = 1$ is given by:

$$\text{vMF}_{\mathbf{x}}[\boldsymbol{\mu}, \kappa] = C(\kappa)\exp(\kappa \boldsymbol{\mu}^T \mathbf{x}), \tag{4.2}$$

where the normalisation constant is given by:

$$C(\kappa) = \frac{\kappa}{4\pi \sinh \kappa}, \tag{4.3}$$

$\boldsymbol{\mu} \in \mathbb{R}^3$ is the mean direction ($\|\boldsymbol{\mu}\| = 1$) and $\kappa \in \mathbb{R}$ is the concentration parameter.

Under the assumption of vMF-distributed noise, we can write a probabilistic spherical camera model as:

$$\Pr(\mathbf{x}|\mathbf{w}, \boldsymbol{\Omega}, \boldsymbol{\tau}) = \text{vMF}_{\mathbf{x}}[\mathbf{spherical}[\mathbf{w}, \boldsymbol{\Omega}, \boldsymbol{\tau}], \kappa]. \tag{4.4}$$

Hence, the expected position of a spherical image point is the projection of the corresponding 3D point via the spherical camera model (4.1) but the observation is subject to vMF noise about the projection of the 3D point.

4.2 Spherical Structure-from-motion

The goal of spherical structure from motion is to choose the most likely 3D positions of the I observed points and the pose of the J cameras that observed those points.

The maximum likelihood solution with vMF noise is given by:

$$\hat{\theta} = \arg\max_{\theta} \left[\sum_{i=1}^{I} \sum_{j=1}^{J} \log[\Pr(\mathbf{x}_{ij} | \mathbf{w}_i, \boldsymbol{\Omega}_j, \boldsymbol{\tau}_j)] \right] \tag{4.5}$$

$$= \arg\max_{\theta} \sum_{i=1}^{I} \sum_{j=1}^{J} \log[\text{vMF}_{\mathbf{x}_{ij}}[\textbf{spherical}[\mathbf{w}_i, \boldsymbol{\Omega}_j, \boldsymbol{\tau}_j], \kappa]] \tag{4.6}$$

$$= \arg\max_{\theta} \sum_{i=1}^{I} \sum_{j=1}^{J} \left(\frac{\boldsymbol{\Omega}_j \mathbf{w}_i + \boldsymbol{\tau}_j}{\| \boldsymbol{\Omega}_j \mathbf{w}_i + \boldsymbol{\tau}_j \|} \right)^T \mathbf{x}_{ij}, \tag{4.7}$$

where θ contains the unknown world points $\{\mathbf{w}_i\}_{i=1}^{I}$ and the extrinsic parameters for each camera $\{\boldsymbol{\Omega}_j, \boldsymbol{\tau}_j\}_{j=1}^{J}$. In practice, not all cameras observe all features so a feature matrix is used to keep track of this and the summation is only over 3D points with corresponding observations.

Hence, the solution maximises the dot product between the unit vector in the direction of the estimated 3D point position and the observed unit vector on the sphere. Contrast this with the error term used in previous attempts at spherical SFM (e.g. [43, 50, 80]) which minimises total squared angular error:

$$\theta_{\text{ang}} = \arg\min_{\theta} \sum_{i=1}^{I} \sum_{j=1}^{J} \arccos^2 \left[\left(\frac{\boldsymbol{\Omega}_j \mathbf{w}_i + \boldsymbol{\tau}_j}{\| \boldsymbol{\Omega}_j \mathbf{w}_i + \boldsymbol{\tau}_j \|} \right)^T \mathbf{x}_{ij} \right]. \tag{4.8}$$

While this objective is intuitive, it is not justified by an explicit spherical noise model. Moreover, it requires an additional inverse trigonometric function and exponentiation for each term compared to the vMF-derived objective function.

4.2.1 Pipeline

We begin by providing a high level overview of our spherical structure-from-motion pipeline. In the following subsections we then present in detail the theory associated with each step of the pipeline and our novel formulations of the spherical-n-point problem and calibrated spherical reconstruction. Finally, we provide pseudocode to describe the algorithm in more detail.

The general pipeline is as follows:

1. Select a pair of images, robustly estimate the spherical essential matrix and

decompose it to find the rotation and translation from image 1 to 2. Estimate position of observed world points using calibrated spherical reconstruction.

2. Align new image to 3D world points visible in that image by solving the spherical-n-point problem.

3. Reconstruct 3D world points for which the new view provides a second observation by solving the calibrated spherical reconstruction problem.

4. Perform bundle adjustment over all estimated values.

5. Repeat steps 2-4 until all images included.

To resolve the unknown scale ambiguity, the length of the first translation is fixed to 1 and this is enforced during nonlinear optimisation. During nonlinear optimisations involving pose estimates, we represent rotation matrices in axis-angle space. We use MATLAB function vrrotmat2vec to implement. It is rotated around axis by the right-hand rule.

4.2.2 Two View Spherical Geometry

Two view epipolar geometry for spherical images closely follows that of the planar perspective formulation. Torii et al. [94] showed that if $\mathbf{x}$ and $\mathbf{x}'$ are two unit vectors representing corresponding points in two spherical images, then the essential matrix $\mathbf{E}$ satisfies: $\mathbf{x}^T\mathbf{E}\mathbf{x}' = 0$. Note that since spherical cameras do not have intrinsic parameters and image points are not represented in angular coordinates, there is no distinction between the essential and fundamental matrices. Also, in contrast to the perspective case, rather than being 2D image points represented as 3D homogeneous coordinates, $\mathbf{x}$ and $\mathbf{x}'$ are unit vectors in $\mathbb{R}^3$.

The essential matrix can be decomposed [39] as: $\mathbf{E} = [\boldsymbol{\tau}]_\times \boldsymbol{\Omega}$, where $\boldsymbol{\Omega}$ and $\boldsymbol{\tau}$ are the rotation and translation which relate the camera coordinate systems between

the two views and $[.]_\times$ is the cross product matrix:

$$[\mathbf{x}]_\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}. \tag{4.9}$$

To perform this decomposition [39], we take the singular value decomposition (SVD) $\mathbf{E} = \mathbf{U}\mathbf{L}\mathbf{V}^T$. The translation is given by $\boldsymbol{\tau} = \pm\mathbf{u}_3$ and the rotation matrix by $\boldsymbol{\Omega} = \mathbf{U}\mathbf{W}^{-1}\mathbf{V}^T$ or $\boldsymbol{\Omega} = \mathbf{U}\mathbf{W}\mathbf{V}^T$ where

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.10}$$

Hence, there are four possible combinations of translation vector and rotation matrix. In the perspective case, this fourfold ambiguity is resolved by choosing the solution which places points in front of the camera. For the spherical case, we propose a slightly different procedure. It is useful to rewrite the spherical camera equation for the first two views in terms of scale parameters λ and λ':

$$\lambda\mathbf{x} = \mathbf{w}, \tag{4.11}$$

$$\lambda'\mathbf{x}' = \boldsymbol{\Omega}\mathbf{w} + \boldsymbol{\tau}, \tag{4.12}$$

where $\lambda, \lambda' > 0$. Spherical point $\mathbf{x}$ is observed by the first camera whose coordinate system coincides with world coordinates. Substituting and rearranging provides three linear equations in terms of the two scale parameters:

$$\lambda'\mathbf{x}' - \lambda\boldsymbol{\Omega}\mathbf{x} - \boldsymbol{\tau} = \mathbf{0}. \tag{4.13}$$

We use this equation to select from the four possible combinations of rotation matrix and translation vector extracted from $\mathbf{E}$. Since we expect both scale parameters to be positive, for each point pair we substitute each of the four $(\boldsymbol{\Omega}, \boldsymbol{\tau})$ and solve for λ and λ'. If both are positive, we cast a vote for the solution $(\boldsymbol{\Omega}, \boldsymbol{\tau})$. The solution with

the most votes is chosen. Note that the substitution to obtain (4.13) eliminates $\mathbf{w}$ and allows us to test the positivity of λ and λ' without first solving for $\mathbf{w}$.

In practice, we use Random Sample Consensus (RANSAC) to estimate the essential matrix from point correspondences between the first pair of images. Any robust model fitting procedure could be used here [51, 60, 63–66]. We use RANSAC for its simplicity and robustness. We then decompose as above to obtain the rotation and translation between these two images.

4.2.3 The Spherical-n-point Problem

For images 3 and onwards, we estimate their pose relative to the current estimate of the 3D scene. Estimating camera pose (i.e. extrinsic parameters) relative to a known 3D scene is a well-studied problem for perspective images and is known as the perspective-n-point (PnP) problem. We propose here a variant of this problem adapted specifically to spherical images and refer to it as the spherical-n-point (SnP) problem.

Given I 3D world points and their corresponding spherical projections $\mathbf{w}_i$, the maximum likelihood solution for the extrinsic parameters of the new camera is given by:

$$\hat{\mathbf{\Omega}}, \hat{\boldsymbol{\tau}} = \underset{\mathbf{\Omega}\in \mathrm{SO}(3), \boldsymbol{\tau}\in\mathbb{R}^3}{\arg\max} \sum_{i=1}^{I} \left(\frac{\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}}{\|\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|} \right)^T \mathbf{x}_i. \tag{4.14}$$

where $\mathbf{x}_i$ is the actual observation spherical point. This is an optimisation problem that is constrained (since $\mathbf{\Omega}$ must be a rotation matrix) and non-convex. Hence, the globally optimal solution cannot be found in closed form and optimisation may only provide a local minima. The goal therefore is to develop a robust and efficient means to compute a good estimate of the extrinsic parameters which can be used to initialise a non-linear optimisation of (4.14).

The classical approach for computing initial estimates for the rotation and translation (in both perspective and spherical SFM) is the DLT method (originally so-called for the perspective case by Abdel-Aziz and Karara [1]). This is a linearisation of the non-convex objective based on a collinearity condition between a 3D point and its projection. From the same starting point, we derive a modification of the

standard DLT that is specifically adapted to spherical image geometry.

Following the DLT method, we can express the collinearity condition for each visible scene point as follows:

$$\lambda \mathbf{x}_i = \boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}. \tag{4.15}$$

This is a linear similarity relation whose solution is also a solution to the following linear equation:

$$\mathbf{x}_i \times (\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}) = \mathbf{0}. \tag{4.16}$$

Intuitively, (4.16) says that the vector to the 3D point in the camera coordinate system should be parallel to the unit vector to the corresponding spherical point (i.e. their cross product is the zero vector). Each 3D point contributes three equations to a system of linear equations (although note that since the cross product matrix has rank 2, only two of the equations are linearly independent). The complete system of equations can be expressed as a homogeneous system:

$$\mathbf{A}\mathbf{b} = \mathbf{0}, \tag{4.17}$$

where

$$\mathbf{A} = \begin{bmatrix} [\mathbf{x}_1]_\times \begin{bmatrix} u_1 & v_1 & w_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & u_1 & v_1 & w_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & u_1 & v_1 & w_1 & 1 \end{bmatrix} \\ \vdots \\ [\mathbf{x}_I]_\times \begin{bmatrix} u_I & v_I & w_I & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & u_I & v_I & w_I & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & u_I & v_I & w_I & 1 \end{bmatrix} \end{bmatrix} \tag{4.18}$$

and the vector

$$\mathbf{b} = [\omega_{11}\ \omega_{12}\ \omega_{13}\ \tau_x\ \omega_{21}\ \omega_{22}\ \omega_{23}\ \tau_y\ \omega_{31}\ \omega_{32}\ \omega_{33}\ \tau_z]^T \tag{4.19}$$

contains a vectorised version of the rotation matrix and translation vector (ω_{ij} refers to the (i,j)th element of the rotation matrix $\boldsymbol{\Omega}$ and $\boldsymbol{\tau} = [\tau_x\ \tau_y\ \tau_z]^T$). Since this is a homogeneous system, we observe that: 1. there is always a trivial solution $\mathbf{b} = \mathbf{0}$, 2. if $\mathbf{b} \neq \mathbf{0}$ is a solution then $k\mathbf{b}$ is also a solution.

Due to noise, we do not expect an exact solution to this problem. Hence, we may instead minimise a least squares criterion: $\|\mathbf{A}\mathbf{b}\|^2$. In order to resolve the

arbitrary scaling and to avoid the trivial solution, the standard approach is to solve a minimum direction problem of the form: minimise $\|\mathbf{Ab}\|^2$ subject to $\|\mathbf{b}\| = 1$. This minimisation problem is straightforward to solve using an SVD of $\mathbf{A}$. Note that this imposes no constraint that the elements of $\boldsymbol{\Omega}$ should form a valid rotation matrix. Hence, the elements of $\mathbf{b}$ corresponding to the rotation matrix are transformed to the closest valid orthogonal matrix by solving an orthogonal Procrustes problem. The scale of the translation vector is found by computing the mean scale between the estimated rotation matrix and the raw estimate taken from $\mathbf{b}$.

Finally, there is a sign ambiguity that must be resolved: negating the rotation and translation still satisfies the collinearity condition in (4.15). This ambiguity arises because the condition is minimised both by a vector pointing in the same direction as a spherical point, but also its negative. However, only one of the two possible solutions will yield a valid rotation matrix. We test whether the determinant of the rotation matrix is positive and, if not, negate both the translation vector and rotation matrix. This is the baseline method (used previously [81]) against which we compare in our experimental evaluation.

The drawback to this approach is that there is no guarantee of positive depth for all points or even a large majority. In practice, for spherical images (where features may be observed in all directions) we observe that the classical DLT often aligns a 3D point and the corresponding spherical point in opposite directions. For this reason, we propose instead to integrate the depth test into the optimisation. To do so, we impose positive depth:

$$\mathbf{x}_i \cdot (\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}) \geq 0. \tag{4.20}$$

as either a soft or hard constraint on the solution of (4.17). Importantly, both of our formulations remain convex optimisation problems and we show that they outperform the classical DLT approach.

4.2.3.1 Hard Constraint

We can express the dot product constraints in matrix form as $\mathbf{Cb} \geq \mathbf{0}$ where

$$\mathbf{C} = \begin{bmatrix} u_1x_1 & v_1x_1 & w_1x_1 & x_1 & u_1y_1 & v_1y_1 & w_1y_1 & y_1 & u_1z_1 & v_1z_1 & w_1z_1 & z_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_Ix_I & v_Ix_I & w_Ix_I & x_I & u_Iy_I & v_Iy_I & w_Iy_I & y_I & u_Iz_I & v_Iz_I & w_Iz_I & z_I \end{bmatrix}.$$

Hence, the constrained minimisation problem is

$$\min_{\mathbf{b}} \|\mathbf{Ab}\|^2 \text{ s.t. } -\mathbf{Cb} \leq \mathbf{0}.$$

This is a standard inequality constrained homogeneous least squares problem. Unfortunately, it can still be solved by the trivial solution $\mathbf{b} = \mathbf{0}$. Imposing the quadratic equality constraint $\|\mathbf{b}\| = 1$ as for the DLT method leads to a quadratically-constrained quadratic programming (QCQP) problem. This is non-convex like the original optimisation problem (4.14) for which we sought a convex initialisation. Instead, we impose a simple linear equality constraint on one element of $\mathbf{b}$. Note that all elements of $\boldsymbol{\tau}$ (stored in b_4, b_8 and b_{12}) could be zero and elements of $\boldsymbol{\Omega}$ (stored in $b_{1\ldots3}$, $b_{5\ldots7}$ and $b_{9\ldots11}$) could be zero or negative. So forcing one element to unity may lead to a very poor solution.

For this reason, we solve the problem six times with different linear equality constraints $b_1 = \pm 1$, $b_2 = \pm 1$ and $b_3 = \pm 1$ (since a row of a rotation matrix must contain at least one non-zero entry). Which ever solution gives the lowest residual error (once the appropriate elements of $\mathbf{b}$ have been transformed to the closest rotation matrix) is taken as the solution. Explicitly, we solve the following constrained linear least squares problem:

$$\min_{\mathbf{b}} \|\mathbf{Ab}\|^2 \text{ s.t. } -\mathbf{Cb} \leq \mathbf{0} \wedge (b_1 = \pm 1 \vee b_2 = \pm 1 \vee b_3 = \pm 1), \tag{4.21}$$

and recover $\boldsymbol{\Omega}$ and $\boldsymbol{\tau}$ from $\mathbf{b}$ as for the standard DLT method.

4.2.3.2 Soft Constraint

Imposing positive depth as a hard constraint may force the estimated camera pose to be highly inconsistent with other measurements. For example, this is particularly

the case when the data contains correspondence errors between 3D world points and spherical image points. For this reason, we propose a variant in which negative depths are penalised as a soft constraint. To do so, we penalise the square of any negative dot products:

$$\|\mathbf{Ab}\|^2 + \gamma\| \max\{\mathbf{0}, -\mathbf{Cb}\} \|^2, \text{ s.t. } b_1 = \pm 1 \vee b_2 = \pm 1 \vee b_3 = \pm 1, \tag{4.22}$$

where the max operation is applied component-wise and γ weights the penalty term (we use $\gamma = 1$ in our experiments). The penalty term amounts to the sum of the square of the positive values of $-\mathbf{Cb}$. Importantly, this is still convex and we solve it using CVX, a package for specifying and solving convex programs [31, 32].

4.2.3.3 Weights

Finally, we consider a weighted variant of our method that allows the convex optimisation to be related back to our original objective function.

Although it is not immediately apparent, (4.16) is implicitly applying weights to each of the sets of linear equations. Expanding the cross product makes this clearer:

$$\min_{\mathbf{\Omega},\boldsymbol{\tau}} \sum_i \|\mathbf{x}_i \times (\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau})\|^2 = \min_{\mathbf{\Omega},\boldsymbol{\tau}} \sum_i \| \sin(\phi_i)\|\mathbf{x}_i\|\|\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|\mathbf{m}_i \|^2, \tag{4.23}$$

where ϕ_i is the angle between $\mathbf{x}_i$ and $\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}$ and $\mathbf{m}_i$ is a unit vector orthogonal to $\mathbf{x}_i$ and $\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}$. Since $\|\mathbf{x}_i\| = 1$ and the direction of $\mathbf{m}_i$ does not affect the magnitude of the expression, this simplifies to:

$$\min_{\mathbf{\Omega},\boldsymbol{\tau}} \sum_i \|\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|^2 \sin^2(\phi_i). \tag{4.24}$$

It now becomes clear that by minimising the cross product, we actually minimise an angular error (the square of the sine of the angle) weighted by $\|\mathbf{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|^2$, i.e. the square of the Euclidean distance from the camera to the point. What this means is that points that are further from the camera are weighted more heavily in the optimisation. This is unlikely to be desirable since the accuracy of feature detection and matching is likely to degrade for points that are further away.

Let us now introduce weights into the minimisation as follows:

$$\min_{\boldsymbol{\Omega},\boldsymbol{\tau}} \sum_i k_i \|\mathbf{x}_i \times (\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau})\|^2, \tag{4.25}$$

and define the weights as: $k_i = \|\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|^{-2}$. Following the same derivation as above and writing in terms of cosine by the Pythagorean identity, this weighted minimisation is equivalent to:

$$\max_{\boldsymbol{\Omega},\boldsymbol{\tau}} \sum_i \cos^2(\phi_i) = \max_{\boldsymbol{\Omega},\boldsymbol{\tau}} \sum_i \left[\left(\frac{\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}}{\|\boldsymbol{\Omega}\mathbf{w}_i + \boldsymbol{\tau}\|} \right)^T \mathbf{x}_i \right]^2 \tag{4.26}$$

We now see a close relationship to the probabilistic formulation in (4.14). Namely, the weighted cross product objective differs from the probabilistic objective only in the fact that it squares the dot product terms. In practice however, we cannot compute the desired weights in (4.2.3.3) since this requires the rotation and translation of the camera to already be known. Hence, we propose an iterative reweighting approach. First, we use the unweighted version to compute an initial rotation and translation estimate. We use this to compute weights for each point and then re-estimate rotation and translation using the weighted version. This process can be iterated to convergence and used with both the soft and hard constraints.

4.2.4 Calibrated Spherical Reconstruction

With estimates of the poses of J cameras to hand, the 3D position of a point $\mathbf{w}$ observed by those cameras can be computed by maximising likelihood with respect to $\mathbf{w}$:

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} \sum_{j=1}^{J} \left(\frac{\boldsymbol{\Omega}_j\mathbf{w} + \boldsymbol{\tau}_j}{\|\boldsymbol{\Omega}_j\mathbf{w} + \boldsymbol{\tau}_j\|} \right)^T \mathbf{x}_j. \tag{4.27}$$

Again, following the DLT approach we write the collinearity criterion as a cross product:

$$[\mathbf{x}_j]_\times (\boldsymbol{\Omega}_j\mathbf{w} + \boldsymbol{\tau}_j) = \mathbf{0}. \tag{4.28}$$

and rewrite this as a system of J linear equations in terms of the unknown 3D point position $\mathbf{w}$: $\mathbf{Bw} = \mathbf{d}$, where

$$\mathbf{B} = \begin{bmatrix} \omega_{31}^1 y_1 - \omega_{21}^1 z_1 & \omega_{32}^1 y_1 - \omega_{22}^1 z_1 & \omega_{33}^1 y_1 - \omega_{23}^1 z_1 \\ \omega_{11}^1 z_1 - \omega_{31}^1 x_1 & \omega_{12}^1 z_1 - \omega_{32}^1 x_1 & \omega_{13}^1 z_1 - \omega_{33}^1 x_1 \\ \omega_{21}^1 x_1 - \omega_{11}^1 y_1 & \omega_{22}^1 x_1 - \omega_{12}^1 y_1 & \omega_{23}^1 x_1 - \omega_{13}^1 y_1 \\ \vdots & \vdots & \vdots \\ \omega_{31}^J y_J - \omega_{21}^J z_J & \omega_{32}^J y_J - \omega_{22}^J z_J & \omega_{33}^J y_J - \omega_{23}^J z_J \\ \omega_{11}^J z_J - \omega_{31}^J x_J & \omega_{12}^J z_J - \omega_{32}^J x_J & \omega_{13}^J z_J - \omega_{33}^J x_J \\ \omega_{21}^J x_J - \omega_{11}^J y_J & \omega_{22}^J x_J - \omega_{12}^J y_J & \omega_{23}^J x_J - \omega_{13}^J y_J \end{bmatrix}, \tag{4.29}$$

and

$$\mathbf{d} = \begin{bmatrix} \tau_z^1 y_1 - \tau_y^1 z_1 \\ \tau_x^1 z_1 - \tau_z^1 x_1 \\ \tau_y^1 x_1 - \tau_x^1 y_1 \\ \vdots \\ \tau_z^J y_J - \tau_y^J z_J \\ \tau_x^J z_J - \tau_z^J x_J \\ \tau_y^J x_J - \tau_x^J y_J \end{bmatrix}. \tag{4.30}$$

The superscripts on ω and τ indicate with which camera the extrinsic parameters are associated.

We again solve in a least squares sense by minimising $\|\mathbf{Bw} - \mathbf{d}\|^2$. Unlike the SnP problem, this linear system is not homogeneous and hence constraints to avoid a trivial solution are not required. However, the same problem arises that the collinearity condition is satisfied by placing a 3D point in the opposite direction of a spherical point and so we use the same hard or soft constraints as above.

To do so, we rewrite the dot product constraint in (4.20) in terms of $\mathbf{w}$ and stack the J equations in a system of linear equations, yielding the following inequality constraint:

$$-\mathbf{Fw} \leq \mathbf{g}, \tag{4.31}$$

where

$$\mathbf{F} = \begin{bmatrix} \omega_{11}^1 x_1 + \omega_{21}^1 y_1 + \omega_{31}^1 z_1 & \omega_{12}^1 x_1 + \omega_{22}^1 y_1 + \omega_{32}^1 z_1 & \omega_{13}^1 x_1 + \omega_{23}^1 y_1 + \omega_{33}^1 z_1 \\ \vdots & \vdots & \vdots \\ \omega_{11}^J x_1 + \omega_{21}^J y_1 + \omega_{31}^J z_1 & \omega_{12}^J x_1 + \omega_{22}^J y_1 + \omega_{32}^J z_1 & \omega_{13}^J x_1 + \omega_{23}^J y_1 + \omega_{33}^J z_1 \end{bmatrix}, \tag{4.32}$$

and

$$\mathbf{g} = \begin{bmatrix} \tau_x^1 x_1 + \tau_y^1 y_1 + \tau_z^1 z_1 \\ \vdots \\ \tau_x^J x_J + \tau_y^J y_J + \tau_z^J z_J \end{bmatrix}. \tag{4.33}$$

This can be enforced as either a hard or soft constraint in exactly the same way as for the SnP methods described above.

Algorithm 1 Spherical Structure-from-motion

INPUTS: Set of spherical images

OUTPUTS: Camera poses $\{\mathbf{\Omega}_j, \boldsymbol{\tau}_j\}_{j=1}^{J}$, 3D world points $\{\mathbf{w}_i\}_{i=1}^{I}$

Extract features for images 1 and 2
Find matches $M_{1,2} = \{(a,b) | \mathbf{x}_{a1} \text{ matches } \mathbf{x}_{b2}\}$.[1]
// (Optional) Remove possible non-scene points:
for (a,b) **in** $M_{1,2}$ **do**
  if $\mathbf{x}_{a1} \cdot \mathbf{x}_{b2} > t_2$ **then**
    $M_{1,2} := M_{1,2} \setminus \{(a,b)\}$
  end if
end for
Compute pose transformation from view 1 to view 2 (refer Section 4.2.2).
Estimate 3D world points $\{\mathbf{w}_i\}_{i=1}^{|M_{1,2}|}$ using calibrated spherical reconstruction (refer Section 4.2.4).
// Remove noisy points:
for $j := 1$ **to** 2 **do**
  for $i := 1$ **to** $|M_{1,2}|$ **do**
    if $(\textbf{spherical}[\mathbf{w}_i, \mathbf{\Omega}_j, \boldsymbol{\tau}_j] \cdot \mathbf{x}_{ij}) < t_3$ **then**
      Remove $\mathbf{x}_{i1}$, $\mathbf{x}_{i2}$ and $\mathbf{w}_i$ from the reconstruction.
    end if
  end for
end for
Bundle adjustment of 3D world points and camera pose via nonlinear optimisation of (4.7).
for $j := 3$ **to** J **do**
  Extract features for image j
  Find matches $M_{j,j-1} = \{(a,b) | \mathbf{x}_{a,j} \text{ matches } \mathbf{x}_{b,j-1}\}$.[1]
  // (Optional) Remove possible non-scene points:
  for (a,b) **in** $M_{j,j-1}$ **do**
    if $\mathbf{x}_{a,j} \cdot \mathbf{x}_{b,j-1} > t_2$ **then**
      $M_{j,j-1} := M_{j,j-1} \setminus \{(a,b)\}$
    end if
  end for
  // Set of previously observed features:
  $M_{\text{prev}} := \{a | (a,b) \in M_{j,j-1} \wedge (b,c) \in M_{j-1,j-2}\}$
  // Set of newly observed features:
  $M_{\text{new}} := \{a | (a,b) \in M_{j,j-1} \wedge a \notin M_{\text{prev}}\}$
  Compute initial estimate of $\mathbf{\Omega}_j, \boldsymbol{\tau}_j$ by solving SnP on the set $\{(\mathbf{x}_{aj}, \mathbf{w}_a) | a \in M_{\text{prev}}\}$ (refer Section 4.2.3).
  Refine estimate of $\mathbf{\Omega}_j, \boldsymbol{\tau}_j$ by nonlinear optimisation of (4.14).
  Estimate new 3D world points $\{\mathbf{w}_i\}_{i \in M_{\text{new}}}$ using calibrated spherical reconstruction (refer Section 4.2.4).
  // Remove noisy points:
  for i **in** M_{new} **do**
    if $(\textbf{spherical}[\mathbf{w}_i, \mathbf{\Omega}_j, \boldsymbol{\tau}_j] \cdot \mathbf{x}_{ij}) > t_3$ **then**
      Remove $\mathbf{x}_{ij}$ and $\mathbf{w}_i$ from the reconstruction.
    end if
  end for
  Bundle adjustment of 3D world points and camera poses 2 to j via nonlinear optimisation of (4.7).
end for

[1]We follow [62] and only retain matches where the ratio between first and second nearest neighbour distances is less than a threshold, i.e. we require that $\|\mathbf{d}_{a1} - \mathbf{d}_{b2}\| / \|\mathbf{d}_{a1} - \mathbf{d}_{c2}\| < t_1$ where $\mathbf{d}_{a1}$ is the feature descriptor for spherical point $\mathbf{x}_{a1}$ and $\mathbf{d}_{b2}$ and $\mathbf{d}_{c2}$ are the first and second nearest neighbours of $\mathbf{d}_{a1}$ respectively.

4.2.5 Implementation

We show our complete structure-from-motion pipeline in Algorithm 1. For efficiency, we only compute feature matches between adjacent images in a sequence. For denser scene reconstruction, it would be necessary to also test for feature matches between a new image and earlier images in the sequence. Also, we impose no smoothness constraint on the camera poses in the sequence. Although this would likely improve results, it would also obscure the accuracy of pose estimates obtained solely from our proposed SnP and structure-from-motion pipeline. Finally, initialisation from the first two images may not always be a good choice when there is little motion between the first two frames.

Our algorithm relies on the selection of three parameters. t_1 is the threshold on feature distance ratios and determines the quality of match required for a feature to be included in the reconstruction (we use $t_1 = 0.75$ in our experiments). t_3 is the threshold on re-projection error and is used to filter outliers (we use $t_3 = \cos 10^\circ = 0.985$).

The parameter t_2 relates to the removal of "non-scene" points. In the case of egocentric image sequences (i.e. 'first' or 'third' person camera viewpoints), some features will correspond to the camera support and person or vehicle carrying the camera. Also, for a camera rig that contains a nadir hole, there may be a consistent missing region in each image. These features will not move relative to the camera in the same way as the fixed scene and provide spurious correspondences. While it is possible to segment features into different motion clusters, we suggest a much simpler heuristic. The position of such points in the spherical images will be approximately fixed. Hence, in lines 3-7 and 21-25 we filter points by removing any whose position between images is closer than a threshold (we use $t_2 = 0.995$).

Nonlinear refinement and global bundle adjustment requires optimisation of (4.7). Note that this is not a nonlinear least squares problem like in the perspective case. Hence, we use a trust-region algorithm as implemented in the `fminunc` Matlab function.

4.3 Experimental Results

We now present experimental results for pose estimation, structure-from-motion. We begin by solving the spherical-n-point problem on synthetic data, allowing us to evaluate the effect of noise on the camera pose estimates. Then, we evaluate the complete structure-from-motion pipeline on two real world datasets for which ground truth camera trajectories are available.

4.3.1 Experimental Setup

We quantitatively evaluate the accuracy of estimated camera pose on synthetic data and for image sets with known ground truth pose. To do so, we use the following performance metrics.

To measure the accuracy of camera rotation, we compute the geodesic distance in the space of 3D rotations [41] between the ground truth and estimated rotation matrices. The 3D rotation group form a compact Lie group $SO(3)$ which has a natural Riemannian metric. From this, the notion of geodesic distance between two rotation matrices $\mathbf{\Omega_1}$ and $\mathbf{\Omega_2}$ follows:

$$d_g(\mathbf{\Omega_1}, \mathbf{\Omega_2}) = \left\| \log\left(\mathbf{\Omega_1}\mathbf{\Omega_2}^T\right) \right\|, \tag{4.34}$$

where $\log(\mathbf{\Omega})$ is the logarithmic map of $\mathbf{\Omega}$ from $SO(3)$ to $so(3)$ (i.e. the transformation to axis-angle representation). Hence, the error measure amounts to the rotation angle of the rotation matrix $\mathbf{\Omega_1}\mathbf{\Omega_2}^T$, which is in radians and is zero if $\mathbf{\Omega_1} = \mathbf{\Omega_2}$ and in general is bounded: $d_g(\mathbf{\Omega_1}, \mathbf{\Omega_2}) \in [0, \pi]$. We define the rotation error as $\varepsilon_{\text{rotation}} = d_g(\mathbf{\Omega}_{\text{groundtruth}}, \mathbf{\Omega}_{\text{estimated}})$.

To measure the positional accuracy, we compute the implied camera centre in world coordinates from the estimated translation: $\mathbf{c} = -\mathbf{\Omega}^T\boldsymbol{\tau}$ and compute the Euclidian distance $\varepsilon_{\text{position}} = \|\mathbf{c}_{\text{groundtruth}} - \mathbf{c}_{\text{estimated}}\|$.

4.3.2 Spherical-n-point Problem

We evaluate the alternative methods we propose for solving the SnP problem using synthetic data. We randomly generate a camera rotation (by sampling uniformly from axis-angle space) and centre (by sampling from $\mathcal{N}(0, 10^2)$ for c_x, c_y and c_z). We then randomly generate n 3D points (by sampling from $\mathcal{N}(0, 100^2)$ for u, v and w). We add Gaussian noise to the 3D point positions (sampled from $\mathcal{N}(0, \sigma^2)$), project the 3D points to the virtual spherical camera using (4.1) and finally add vMF noise to the spherical image points. To do so, for each point we randomly sample from the vMF distribution given in (4.4) using the method described by Wood [100]. Every experimental configuration is repeated 1,000 times and the results averaged.

We evaluate five different methods. We refer to the classical DLT method as `D`. The method we propose in Section 4.2.3.1 using a hard constraint is referred to as `H` or as `HW` when we employ the reweighting scheme in Section 4.2.3.3. Similarly, the soft constraint in Section 4.2.3.2 is referred to as `S` and as `SW` for the reweighted variant.

We consider two common scenarios for the SnP problem. The first is for a minimal ($n = 6$) set of points. This scenario arises when, for example, RANSAC is used to fit to noisy sets of points where each random sampling selects a minimal subset of the data. We show results for this scenario in Table 4.1. We vary the value of the concentration parameter of von Mises-Fisher noise over the set: $\kappa = \{10, 50, 200, 500, 1,000, \infty\}$. This noise corresponds to mean angular errors in the spherical point positions of 22.8°, 10.3°, 5.13°, 3.18°, 2.26° and 0° respectively. We vary the 3D point position noise over the set: $\sigma = \{30, 10, 5, 1, 0.1, 0\}$. We have emboldened the best rotation and position result for each noise setting. First, notice that all of our proposed variants outperform the classical DLT over all noise settings. Second, the weighted variants typically perform worse than the unweighted versions. This is caused by poor estimates of the weights using the noisy pose estimate from the previous iteration. As the algorithm converges, the weights do not necessarily converge to better estimates of the camera-point distances. Finally, it is clear that the unweighted hard constraint yields the most consistent performance, both in terms of the rotation and camera position accuracy. Hence, it is this method that

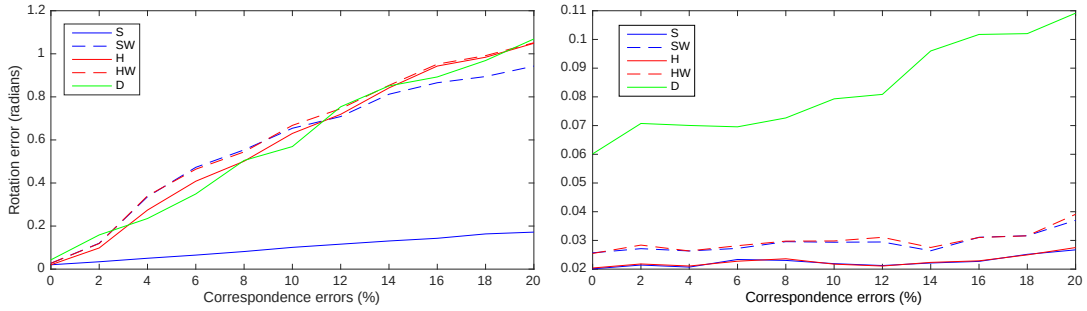
Table 4.1: Quantitative spherical-n-point results for minimal ($n = 6$ points) setting.

κ \ σ		30	10	5	1	0.1	0
10	S	407.89 / 1.08	4207.06 / 0.90	149.41 / 0.91	324.14 / 0.87	778.94 / 0.90	16614 / 0.90
	SW	115.28 / 1.59	766.35 / 1.35	549.76 / 1.34	520.64 / 1.31	384.95 / 1.37	380.47 / 1.68
	H	**112.28** / **0.98**	**111.63** / **0.81**	168.05 / **0.84**	**189.25** / **0.80**	**223.59** / **0.82**	224.02 / **0.82**
	HW	145.02 / 1.05	112.81 / 0.83	**132.80** / 0.86	230.35 / 0.83	232.88 / 0.83	**165.82** / 0.83
	D	424.45 / 2.08	366.93 / 1.95	346.17 / 1.95	436.31 / 1.99	749.66 / 1.96	443.06 / 1.92
50	S	812.36 / 0.75	100.74 / 0.56	158.73 / 0.53	**74.79** / 0.49	261.48 / **0.51**	142.01 / **0.50**
	SW	327.38 / 1.18	212.97 / 0.86	200.96 / 0.78	130.33 / 0.67	232.02 / 0.80	125.29 / 0.87
	H	**98.89** / **0.69**	**77.65** / **0.54**	**68.98** / **0.52**	89.51 / **0.48**	**66.23** / **0.51**	**73.21** / **0.50**
	HW	163.31 / 0.70	79.43 / 0.55	69.94 / 0.53	99.97 / 0.49	75.85 / 0.52	73.31 / 0.52
	D	665.70 / 1.83	275.56 / 1.43	264.86 / 1.33	335.02 / 1.30	272.57 / 1.35	260.41 / 1.33
200	S	112.62 / 0.65	66.79 / 0.40	93.27 / 0.34	53.75 / 0.30	50.19 / 0.30	67.38 / 0.30
	SW	217.38 / 1.01	60.36 / 0.55	84.88 / 0.45	183.96 / 0.37	49.21 / 0.39	71.31 / 0.42
	H	**81.05** / **0.62**	**49.58** / **0.39**	**54.86** / **0.33**	74.04 / **0.29**	**37.94** / **0.30**	**41.11** / **0.29**
	HW	84.94 / 0.64	53.79 / 0.41	57.78 / 0.35	**73.50** / 0.30	39.81 / 0.32	41.21 / 0.31
	D	365.15 / 1.61	300.32 / 1.08	253.10 / 0.94	278.37 / 0.78	311.84 / 0.85	173.13 / 0.82
500	S	211.08 / 0.64	57.10 / 0.36	31.65 / 0.24	28.71 / **0.19**	25.94 / **0.20**	32.53 / **0.21**
	SW	203.39 / 1.05	193.72 / 0.47	30.92 / 0.30	28.07 / 0.21	34.87 / 0.24	33.69 / 0.24
	H	**89.37** / **0.61**	**52.93** / **0.36**	**30.85** / **0.24**	**27.64** / **0.19**	**25.38** / **0.20**	**28.56** / **0.20**
	HW	93.34 / **0.61**	53.36 / **0.36**	31.59 / 0.25	28.35 / 0.21	26.42 / 0.21	29.11 / 0.22
	D	480.48 / 1.65	258.87 / 1.00	118.14 / 0.68	92.21 / 0.57	130.19 / 0.55	287.67 / 0.62
1,000	S	150.86 / 0.63	81.79 / **0.30**	37.28 / 0.25	24.39 / 0.16	20.64 / **0.14**	**25.00** / **0.15**
	SW	276.97 / 0.97	75.53 / 0.40	32.05 / 0.24	21.00 / 0.17	21.56 / 0.15	57.38 / 0.18
	H	**86.47** / **0.61**	**38.27** / **0.30**	**28.11** / **0.23**	**20.95** / **0.15**	**20.28** / **0.14**	31.22 / **0.15**
	HW	86.81 / **0.61**	38.91 / **0.30**	28.22 / 0.24	21.11 / 0.16	21.51 / 0.15	32.47 / 0.16
	D	464.48 / 1.55	171.83 / 0.89	111.65 / 0.62	156.68 / 0.43	93.61 / 0.39	65.92 / 0.41
∞	S	145.76 / 1.22	77.10 / 0.35	29.23 / 0.18	5.21 / **0.04**	0.48 / **0.004**	**0** / **0**
	SW	**16.81** / 1.35	**21.80** / 2.16	**19.71** / 1.96	5.30 / 0.06	0.49 / **0.004**	**0** / **0**
	H	83.52 / **0.61**	38.05 / **0.31**	25.62 / **0.18**	**5.17** / **0.04**	**0.48** / **0.004**	**0** / **0**
	HW	86.07 / 0.62	41.56 / 0.32	26.10 / 0.19	5.24 / **0.04**	0.51 / 0.0042	**0** / **0**
	D	463.44 / 1.62	170.31 / 0.81	94.87 / 0.49	8.72 / 0.08	0.53 / 0.0044	**0** / **0**

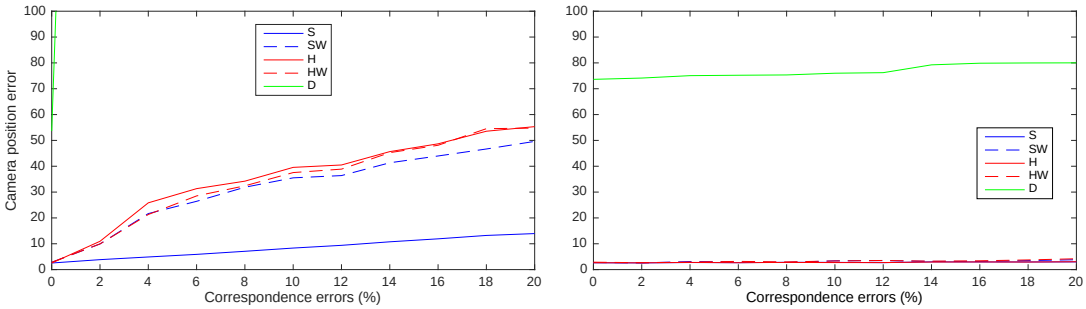
We vary spherical image point noise (κ is the concentration parameter of von Mises-Fisher noise) and Gaussian noise on the 3D point positions (σ is the standard deviation of the noise). The first value is the Euclidean distance between actual and estimated camera centres. The second value is the geodesic distance between actual and estimated camera rotation matrices.

we use in the structure-from-motion results that follow.

The second scenario we evaluate is fitting to a large set of points (in this case $n = 100$) which contains correspondence errors. To simulate correspondence errors, we take a subset (whose size is varied between 0% and 20%) of the points and randomly permute the 3D-spherical correspondences. Results are shown in Figure 4.1. On the left we use all points, testing resilience to outliers. Here, a different picture emerges. In the presence of correspondence errors, it is clear that the unweighted soft constraint yields significantly more robust performance. The classical DLT performs about as well as our other proposed methods in terms of rotation accuracy but is much worse in terms of camera position estimation. On the right, we use RANSAC in conjunction with each method to remove outliers. In this case, all our variants significantly outperform the DLT.



(a) Errors in estimated camera rotation.



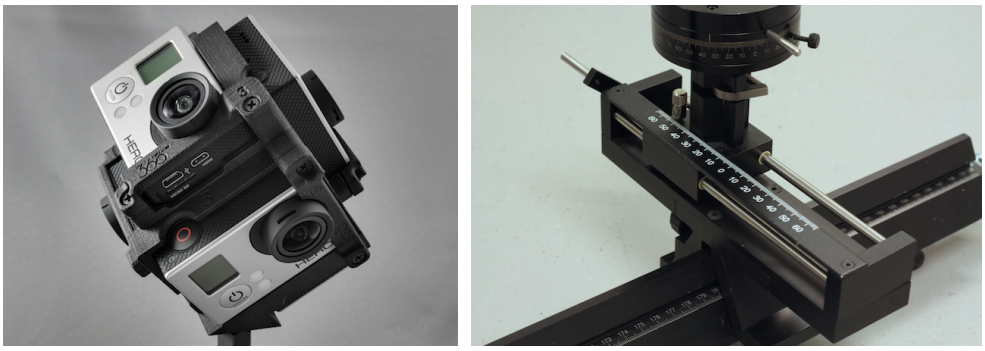
(b) Errors in estimated camera centres.

Figure 4.1: Quantitative spherical-n-point results with correspondence errors. We fix the noise ($\kappa = 200$, $\sigma = 2$) and the number of points ($n = 100$) and vary the number of random correspondence errors. Left: using all points, right: RANSAC.

4.3.3 Quantitative Structure-from-motion Results

Quantitative evaluation of structure-from-motion on real image sequences is difficult since accurate ground truth of 3D world positions is hard to obtain. However, with calibrated camera motion, we can evaluate the accuracy of the camera motion trajectory estimated by our structure-from-motion pipeline. For this experiment, we use a Freedom360 spherical mount containing 6 GoPro Hero3 Black cameras (see Figure 4.2(a)). When the 6 images are stitched together, this provides full 180° by 360° images with no nadir blind spot.

We attach the camera rig to a mount which allows both calibrated rotation and side-to-side translation and then attach this to a linear optical rail (see Figure 4.2(b)). This allows calibrated translation in the u-w plane and rotation about the v axis. We use this set up to acquire two sequences. The first comprises a linear motion trajectory over 1.6m in the z direction in increments of 20cm. The second follows a scaled sine curve of the form $u = a\sin(bw)$ with increments of 20cm in the



(a) Freedom360 camera rig. (b) Linear optical rail and mount.

Figure 4.2: Equipment for ground truth sequence capture.

Figure 4.3: Sample images from the linear trajectory dataset shown as equirectangular images.

w direction. We implement the pipeline by MATLAB. For each sequence the total running time is approximately 2 hours.

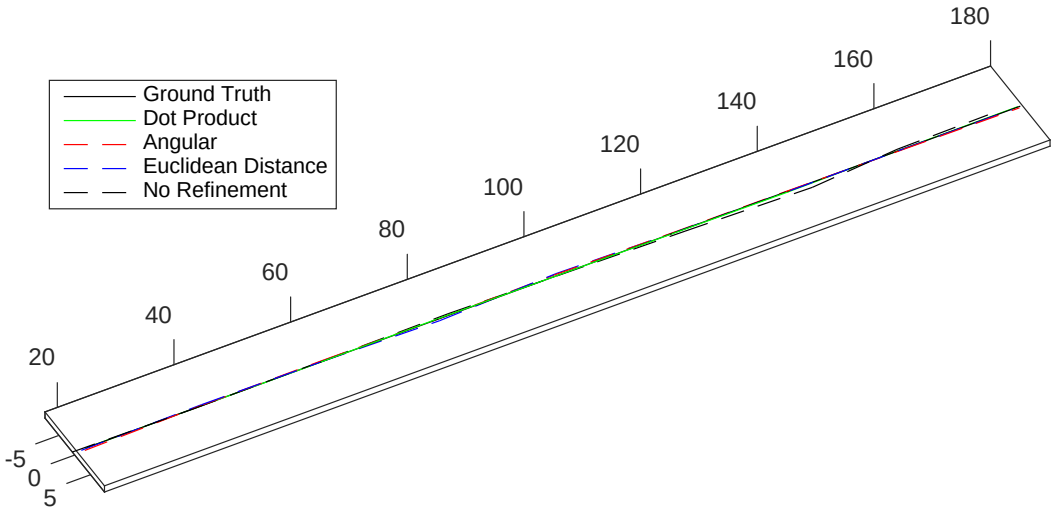
We run our complete structure-from-motion pipeline on the resulting image sequences. We show two sample images from the linear trajectory sequence in Figure 4.3 which are visualised using an equirectangular projection (i.e. latitude and longitude mapped linearly to vertical and horizontal coordinates respectively). We evaluate four variants of the algorithm. The first applies no nonlinear refinement. In other words, we simply solve the convex SnP and calibrated spherical reconstruction problems for each new image and perform no nonlinear optimisation or bundle adjustment. The second and third methods minimise objective functions used in previous work, namely the squared angular error (4.8) and Euclidean distance between spherical image points and projected 3D world points. Finally, our proposed method maximises likelihood under the von Mises-Fisher distribution, amounting to maximisation of a sum of dot products (4.7).

The ground truth and estimated trajectories are plotted in Figure 4.4 and quan-

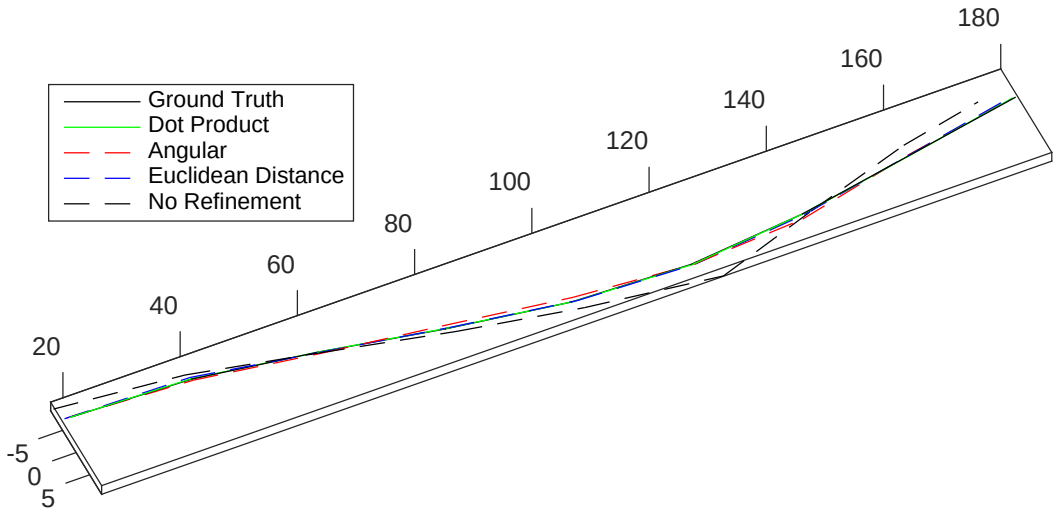
Table 4.2: Quantitative results: trajectory estimation.

Objective function	Linear trajectory		Curved trajectory	
	$\varepsilon_{\text{position}}$	$\varepsilon_{\text{rotation}}$	$\varepsilon_{\text{position}}$	$\varepsilon_{\text{rotation}}$
Squared Euclidean distance	0.67	0.0098	0.69	0.0093
Squared angular error	0.32	0.0066	0.66	0.0085
Dot product	**0.24**	**0.0030**	**0.21**	**0.0041**
No refinement	2.25	0.0119	2.75	0.0096

Errors are mean Euclidean distance between camera centres in centimetres followed by rotation error in radians.



(a) Linear trajectory.



(b) Curved trajectory.

Figure 4.4: Ground truth and estimated camera trajectories for real world image sequence (zoom for detail). Results are shown without optimisation and with optimisation of three different objective functions. Distance units are centimetres.

titative results are shown in Table 4.2. The errors shown are mean Euclidean distance between ground truth and estimated camera centres (in centimetres) after Procrustes alignment of the estimated trajectory to the ground truth. Maximisation of the probabilistic objective function provides the best results on both datasets. They appear qualitatively close in the plot in Figure 4.4 but Table 4.2 contains the quantitative results which show that there is a significant difference. Although there is a significant difference, the total error of the angular criterion is still small and so it is difficult to visualise in Figure 4.4.

4.4 Conclusion

In this chapter we have described a spherical structure-from-motion algorithm that incorporates a well justified spherical noise model (the von Mises-Fisher distribution) which leads to an objective function that is both cheaper to evaluate and performs better than previously used objectives. We have also presented constrained and weighted versions of the spherical-n-point and calibrated spherical reconstruction problems that outperform classical DLT-based approaches under a wide range of noise settings. We implement the pipeline by MATLAB. In the future, we will consider to use another programming language like C++ to implement so that we could calculate the computational time.

Chapter 5

View Stabilization And Synthesis

In this chapter we present two methods for novel view synthesis, making use of the structure-from-motion results obtained by the method in the previous chapter. First, we propose a method for stabilising the viewing direction in spherical video for the purpose of improving the experience of interactively viewing the video. By view stabilization, we mean the temporal smoothing of the camera poses. We are interested in first person video which can make people experience what others captured. View stabilisation is a way to improve the experience because if the video is unstable, it always makes people lose directions, especially when wearing some device like oculus rift. To do so, we require only the camera rotation at each frame and rotate each spherical image back to a canonical coordinate system. The effect is to remove the effect of viewing direction changes. Second, we present a simple method for synthesising a view from a completely new camera position, not included in the input sequence. This potentially allows for a more sophisticated camera stabilisation in which the motion trajectory, as well as the camera orientation, is smoothed. Our method uses only the camera pose estimates (no 3D information is used). For each pixel in the virtual view, we search for the scene depth that maximises photo-consistency over all input images. We show that this simple method is able to predict appearance from both interpolated and extrapolated viewpoints.

5.1 View Direction Stabilisation

We begin by presenting a method for stabilising the viewing direction in a spherical video. While spherical video captures all possible viewing directions, rotation of the camera induces a rotation of the viewing direction for someone interactively viewing the video. This can be extremely disorientating as the user feels like they have lost control over the viewing direction (i.e. the video turns without them turning their head). Correcting this problem requires only an accurate estimate of the camera pose in each frame of a video, exactly as provided by our SFM pipeline. In this section, we describe this process and also how to extract virtual perspective views from the stabilised spherical video, enabling stabilised virtual fly-throughs of the captured video.

5.1.1 Preliminaries

We consider a video sequence of J frames. The rotation matrix associated with the camera in frame j has been estimated by SFM as $\mathbf{\Omega}_j$. We assume that frame 1 is in the canonical view and since $\mathbf{\Omega}_1 = \mathbf{I}_3$, stabilisation is achieved by applying the inverse rotation for each frame to the corresponding image. The pose of the camera includes the rotation and translation relative to first camera coordinate. As long as we rotate back the camera according to the camera rotation, the facing direction of the cameras will appear to be fixed.

In practice, the spherical images are stored in panoramic image format. We write the pixel intensity of the jth image at row r and column c as $E_j(r, c)$. The spherical image point (i.e. unit vector) $\mathbf{x}(r, c)$ associated with pixel (r, c) is given by the following conversion:

$$\mathbf{x}(r,c) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \sin\phi\sin\theta \\ \cos\phi \\ -\sin\phi\cos\theta \end{bmatrix} = \begin{bmatrix} \sin\left(\pi\frac{r}{r_{max}}\right)\sin\left(\pi\left(\frac{2c}{c_{max}}-1\right)\right) \\ \cos\left(\pi\frac{r}{r_{max}}\right) \\ -\sin\left(\pi\frac{r}{r_{max}}\right)\cos\left(\pi\left(\frac{2c}{c_{max}}-1\right)\right) \end{bmatrix}. \quad (5.1)$$

The inverse conversion is given by:

$$r(\mathbf{x}) = r_{max} \arccos y, \tag{5.2}$$

$$c(\mathbf{x}) = r_{max} \left(1 + \arctan\left(-\frac{x}{z}\right)\right). \tag{5.3}$$

where $\|\mathbf{x}\| = 1$, $r \in 1 \ldots r_{max}$, $c \in 1 \ldots c_{max}$, r_{max} and c_{max} are the maximum row number and maximum column number of the panoramic image respectively.

5.1.2 Stabilisation

Stabilisation requires us to apply the inverse of the rotation $\mathbf{\Omega}_j$ to each spherical point $\mathbf{x}_i$ in the jth image and then regularly re-sample the resulting irregularly sampled data. Equivalently, we can apply the forward rotation for each spherical point in the stabilised image and use bilinear interpolation to interpolate the intensity at the resulting non-integer positions in the original panoramic image. The latter option is preferable since bilinear interpolation of regularly sampled data is faster than resampling of irregularly sampled data (which requires construction of a triangulation of the data).

Assume that $\mathbf{x}_i$ is a spherical image point in the canonical pose. The corresponding direction in the jth image is given by $\mathbf{x}'_i = \mathbf{\Omega}_j \mathbf{x}_i$. The colour for the pixel in the stabilised image is obtained by looking up pixel position $(r(\mathbf{x}'_i), c(\mathbf{x}'_i))$ in the image E_j using bilinear interpolation. We repeat this process for all pixels in the image and for all images in the sequence.

5.1.3 Virtual Perspective View Calculation

In order to visualise our view stabilisation results more clearly, we generate a virtual perspective view with fixed viewing direction. This simulates what a perspective viewer would have observed if they fixed their viewing direction relative to the world. The way to transformed the spherical points to the tangent plane is via the log map and the inverse of the transformation is via the exponential map. A virtual perspective view is obtained by the stereographic projection. It is shooting rays from the centre of spherical projection through a virtual image plane and interpolating the

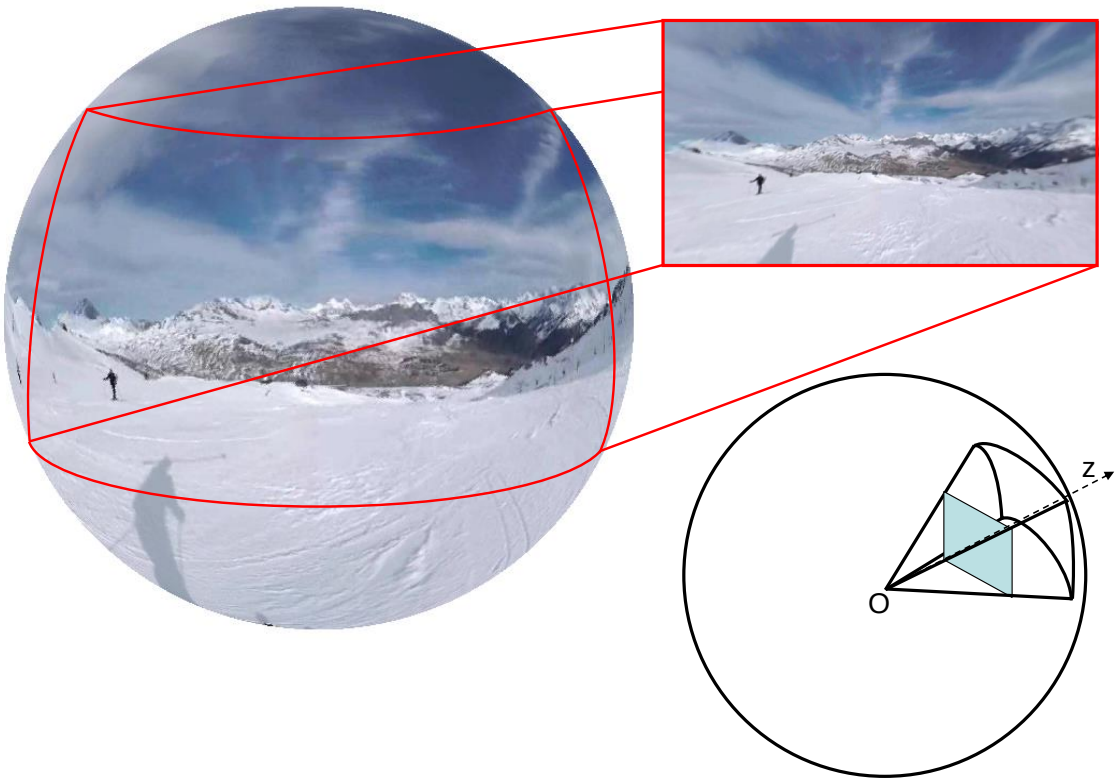


Figure 5.1: Generating a virtual perspective view. Top left: the input spherical image with the extent of the virtual perspective image overlaid. Top right: the resampled virtual perspective image. Bottom right: visualisation of relationship between virtual perspective image plane and spherical sampling region.

spherical image at the resulting point. A perspective view is defined by specifying a horizontal and vertical angular field of view (in our method, we use 116° and 83° respectively) and a sampling rate in units of pixels per degree. We assume that the desired view is along the z-axis. The process is illustrated in Figure 5.1.

We arbitrarily set the virtual image plane at $z = 0.1$ and define its extent in spherical coordinates using:

$$\begin{aligned} x_{\max} &= 0.1 \tan \frac{h}{2}, \\ y_{\max} &= 0.1 \tan \frac{v}{2}, \end{aligned} \tag{5.4}$$

where h and v are the horizontal and vertical angular field of view respectively. We now sample regularly over the range $(-x_{\max}..x_{\max}, -y_{\max}..y_{\max})$. We define the sampling rate as s pixels per degree and hence the spherical image point corresponding

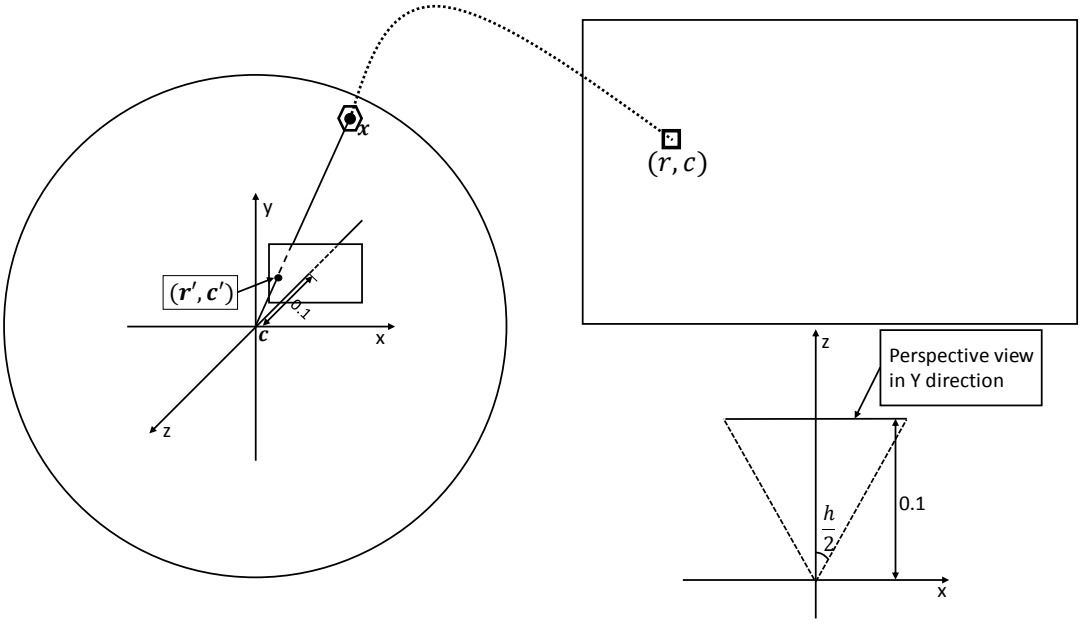


Figure 5.2: Transformations in computing a virtual perspective view. Left: a virtual image plane is defined and a ray for each pixel is cast to the spherical surface. Top right: the spherical point is converted to panoramic image coordinates and the colour interpolated. Bottom right: top down view of the perspective viewing geometry.

to pixel (r, c) in the virtual perspective view is given by:

$$\mathbf{x} = \frac{[-x_{\max} + cs \quad -y_{\max} + rs \quad 0.1]^T}{\|[-x_{\max} + cs \quad -y_{\max} + rs \quad 0.1]\|}. \tag{5.5}$$

Finally, we convert the spherical direction to pixel coordinates in the panoramic image $(r(\mathbf{x}), c(\mathbf{x}))$ and use bilinear interpolation to lookup the colour. We repeat this process for all pixels in the virtual perspective view and for all images in the sequence, leading to a perspective rendering of the whole video sequence. This process is illustrated in Figure 5.2. Note that exactly this process is applied in real time when a spherical video sequence is viewed interactively (for example using a virtual reality head mounted display or through 360 video players).

5.1.4 Structure-from-motion Implementation Optimisations

There are a number of optimisations that can be made to the structure-from-motion pipeline in the context of view stabilisation. Since our goal is only to estimate a rotation matrix per frame, we are not concerned directly with the accuracy of trans-

lations or 3D positions of observed points (though of course, the two are linked). In addition, we need only a very sparse (but high quality) set of point correspondences to robustly estimate the rotation matrix (which has only three degrees of freedom). Finally, distant points are likely to be more reliable since their direction is unchanged (or only subtly) by translations of the camera. With these observations in mind, we suggest a number of optimisations. First, to ensure the robustness of point matches we remove matches where the ratio between the best and second best match is small (i.e. the matched points are not sufficiently distinctive). We set this threshold much higher than would be usual in structure-from-motion, typically leading to around 30 matching points per pair of frames. In addition, we do not try to match points in frame i to those in frames $< i - 1$ (i.e. matching to points that were observed earlier and then missed before being observed again). This dramatically reduces the search space for point matching and yields fewer incorrect matches. In the case of egocentric image sequences (i.e. 'first' or 'third' person camera viewpoints), some features will correspond to the camera support and person or vehicle carrying the camera. These features will not move relative to the camera in the same way as the fixed scene. While it is possible to segment features into different motion clusters, we suggest a much simpler heuristic. Such points are likely to be much closer to the camera than more reliable scene points. Hence, we filter points by removing a proportion of points that are estimated to be closest to the camera.

5.2 View Interpolation And Extrapolation

We now consider the more challenging task of view interpolation and extrapolation. The goal here is to synthesise images with novel *viewpoints* that were not observed in the input sequence. There would be many applications for such a tool. By smoothing the computed camera trajectory and rendering new views, we could reduce unwanted effects such as camera shake and bumps. It may enable new information to be extracted when reviewing events from a new perspective. In this section we propose a simple method to achieve this which does not require a 3D model of the scene.

Our goal is to synthesise an image that would have been observed by a virtual

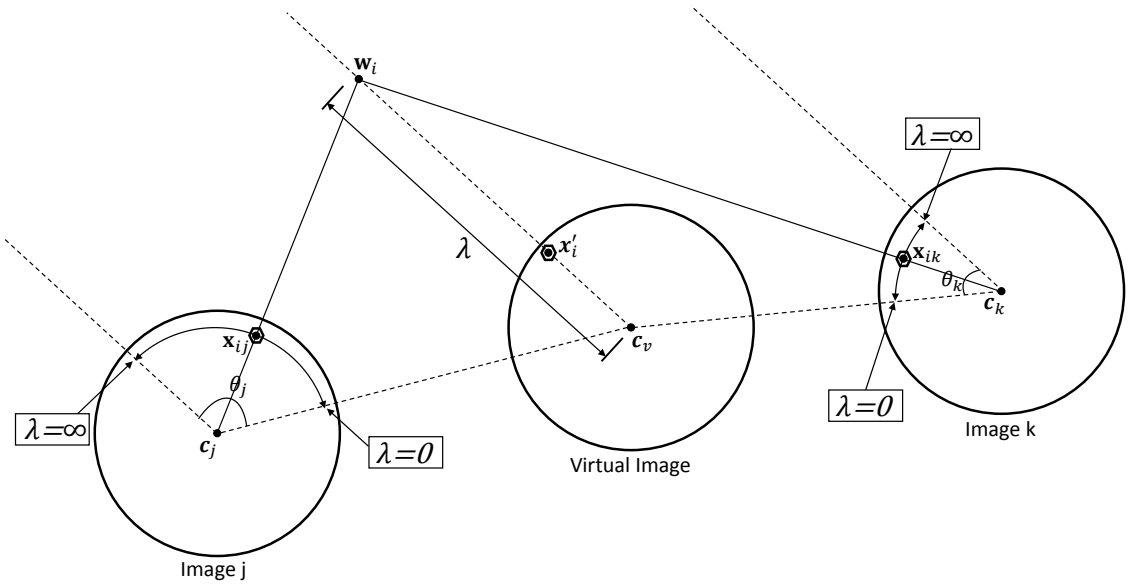


Figure 5.3: The epipolar geometry for a virtual view and multiple input views. $\mathbf{c}_v$ is the camera centre of the virtual view. $\mathbf{c}_j$ and $\mathbf{c}_k$ are camera centres of two real views. λ is the unknown scene depth of the world point corresponding to pixel $\mathbf{x}'_i$ in the virtual view. We show a hypothetical position for world point $\mathbf{w}_i$.

camera with pose $(\boldsymbol{\Omega}_v, \boldsymbol{\tau}_v)$. Suppose the ith pixel in the virtual panoramic image has coordinate (r_i, c_i), where $r \in 1 \ldots r_{\max}, c \in 1 \ldots c_{\max}$. The corresponding ith spherical point $\mathbf{x}'_i$ can be calculated by Equation 5.1. The world point $\mathbf{w}_i$ corresponding to the ith pixel is given by:

$$\mathbf{w}_i(\lambda_i) = \lambda_i \boldsymbol{\Omega}_v^T \mathbf{x}'_i - \boldsymbol{\tau}_v. \tag{5.6}$$

where λ_i is the unknown scene depth. If we knew the scene depth, we could simply compute the world point, project it into any of the input images and look up the colour. However, with no 3D scene information, scene depth is unknown. Hence, we pose the problem as one of searching for the scene depth that maximises photo-consistency (i.e. the similarity of appearance of the world point projected into all input images).

Suppose we have an estimate of the scene depth. Then, in the jth camera, world point $\mathbf{w}_i$ projects to:

$$\mathbf{x}_{ij}(\lambda_i) = \frac{\boldsymbol{\Omega}_j \mathbf{w}_i(\lambda_i) + \boldsymbol{\tau}_j}{\|\boldsymbol{\Omega}_j \mathbf{w}_i(\lambda_i) + \boldsymbol{\tau}_j\|}. \tag{5.7}$$

We now define a neighbourhood $\mathcal{N}(\mathbf{x})$ as the set of pixels in a square region around

pixel $(\text{round}(r(\mathbf{x})), \text{round}(c(\mathbf{x})))$. We use a simple definition of photo-consistency based on block matching. For pixel i in the virtual image, the total photo-consistency cost for a scene depth estimate λ_i is given by summing over all pairs of images and measuring the block matching distance:

$$\varepsilon(\lambda_i) = \sum_{j=1}^{J-1} \sum_{k=j+1}^{J} \sum_{(r,c)\in\mathcal{N}(\mathbf{x}_{ij}(\lambda_i))} \sum_{(s,t)\in\mathcal{N}(\mathbf{x}_{ik}(\lambda_i))} \left(E_j(r,c) - E_k(s,t)\right)^2,$$

where $E_j(r,c)$ and $E_k(s,t)$ are the panoramic images corresponding to the jth and kth cameras respectively. For colour images we also sum over colour channels. Scene depth is positive but unbounded, i.e. $\lambda \in (0,\infty)$. However, this range traces out a finite segment on an epipolar curve in each image (see Figure 5.3). Hence, the search space is finite over the pixels in the input images that are covered by the epipolar curves.

Concretely, the ith virtual spherical image point $\mathbf{x}'_i$ projects to an $\mathbf{x}_{ij}$ in input image j that lies on the epipolar curve with start and end point given by:

$$\lambda = 0 \Rightarrow \mathbf{x}_{ij} = \frac{\mathbf{\Omega}_j(-\mathbf{\Omega}_v^T\boldsymbol{\tau}_v) + \boldsymbol{\tau}_j}{\|\mathbf{\Omega}_j(-\mathbf{\Omega}_v^T\boldsymbol{\tau}_v) + \boldsymbol{\tau}_j\|}, \tag{5.8}$$

$$\lambda = \infty \Rightarrow \mathbf{x}_{ij} = \mathbf{\Omega}_j\mathbf{\Omega}_v^T\mathbf{x}'_i. \tag{5.9}$$

To define the search space, we find the longest epipolar curve over the whole image set and define a sampling rate corresponding to the angular pixel sampling rate in that curve. These sample points are converted back into scene depth values and the block matching cost evaluated exhaustively. This ensures that we sample at least every pixel in every image (but does mean that we over sample all but one image).

In Figure 5.4 we show epipolar curves corresponding to a point in one image. In the top image we select a point on the corner of a desk. In the second and third images we show the epipolar curves corresponding to that point for the range $\lambda \in (0,\infty)$. We see that both curves intersect the correct point as expected. Since the middle curve is longer, we would sample uniformly along this, compute the corresponding scene depths and then evaluate the block matching cost for each of these depths.

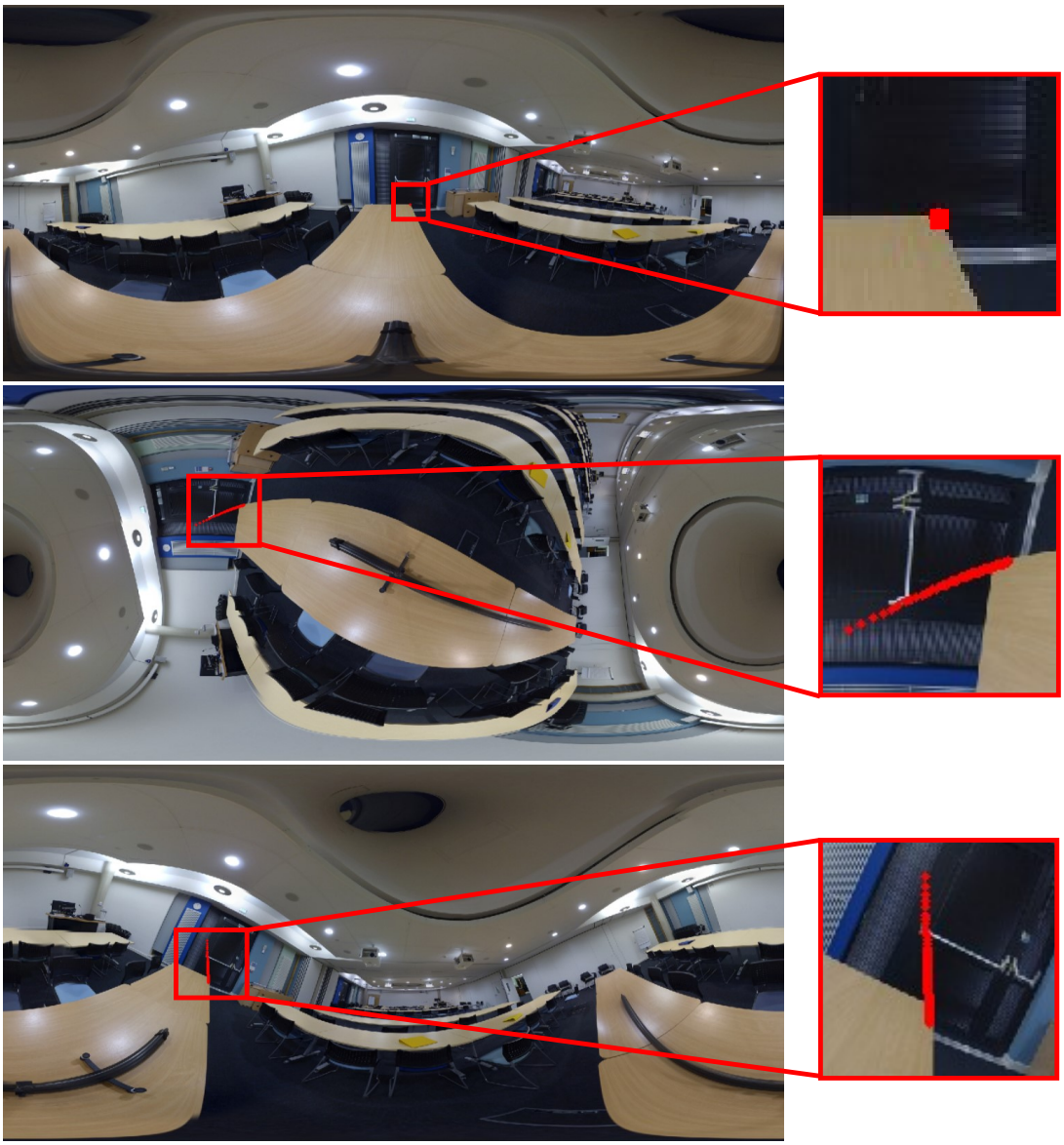


Figure 5.4: Epipolar curve example. In the top image, we select a point. In the second and third images (which have different viewpoints and rotations), we plot the epipolar curves corresponding to this point.

The process described above is effectively a brute force search for the correct correspondence between images. For high resolution images this quickly becomes prohibitively expensive. Hence, we can adopt a coarse-to-fine strategy. We start with a low resolution virtual image and use brute force searching. We then upsample the estimated scene depths and use these values to restrict the search to a small range around the existing scene depth estimate. This process is repeated until the desired resolution is reached. Besides improving speed, this approach also encour-

ages smoothness since each scene depth estimate is initialised with a value that is consistent with its neighbours.

Finally, once we have found the most photo-consistent depth for each pixel in the virtual view we are able to synthesise the novel viewpoint. To do so, for each pixel we simply average the colours observed by each of the input images at the point along the epipolar curve corresponding to the selected scene depth for that pixel. A more sophisticated or robust process could be used here to compute a colour from the selected pixels but we found that a simple averaging was sufficient to prove the principle of the method.

To summarise, our view interpolation/extrapolation algorithm can be summarised into the following steps:

1. Initialise virtual image to lowest resolution
2. For each pixel in virtual image, initialise search range for lambda=0 to lambda=infinity
3. For each pixel in virtual image, search over search range and compute photo-consistency measure over all input images
4. If desired resolution reached, compute output image as mean of colours of corresponding pixels for each virtual pixel and terminate.
5. Increase resolution of virtual image
6. Compute new depth map by upsampling depth map at lower resolution
7. For each pixel in virtual image, set search range as plus or minus some tolerance around depth in upsampled depth image Return to step 3

The computational complexity of this algorithm is linear in the number of resolutions, R, the number of pixels in the highest resolution virtual image, P, and the angular sampling of the longest epipolar curve, A. i.e. overall complexity is O(RPA).

5.3 Experimental Results

We now present experimental results of stabilisation and synthesis on several real world spherical video sequences.

5.3.1 Stabilisation

Our goal is the stabilisation of viewing direction and, hence, the critical performance measure is the accuracy of the estimated rotation matrix at each frame. Numerous measures have been used to measure error or distance between rotation matrices. We have described the most natural measure which is geodesic distance in the space of 3D rotations [41] in Chapter 4. We begin by using this measure to evaluate the accuracy of our SFM pose estimates on synthetic data. Next we present qualitative stabilisation results.

5.3.1.1 Quantitative Rotation Accuracy

To evaluate the quantitative accuracy of the SFM rotation estimates, we use synthetic 3D scene data, camera positions and spherical image projections. We randomly generate 50 3D point positions, distributed normally about the first camera. We then generate a smooth motion trajectory and random rotation matrices for another four cameras. We project the 3D points to each spherical image and add noise to their spherical positions. To do so, for each point we randomly sample from the von Mises-Fisher distribution given in Equation 4.4 using the method described by Wood [100]. We vary the value of κ in order to study the effect of spherical noise on the accuracy of the recovered rotation matrices. In more intuitive terms, $\kappa = 1,000$ gives a mean angular error of 2.3° and $\kappa = 5,000$ gives a mean angular error of 1°.

We solve the resulting structure-from-motion problem for the noisy spherical points. In the nonlinear optimisation steps (including bundle adjustment) we consider both our proposed objective function and the squared angular error. We also include results for a highly efficient version which uses only the linear solutions at each step, i.e. no nonlinear optimisation and no bundle adjustment. Each experiment is repeated 100 hundred times with new random points and averaged results are shown in Figure 5.5. Our objective function outperforms squared angular error in all but one experimental condition, despite being considerably cheaper to compute. The linear version is worse, but still gives an error less than 0.1 radians in all but one case. This provides evidence that our estimated rotation matrices are of sufficient quality to be useful for view stabilisation.

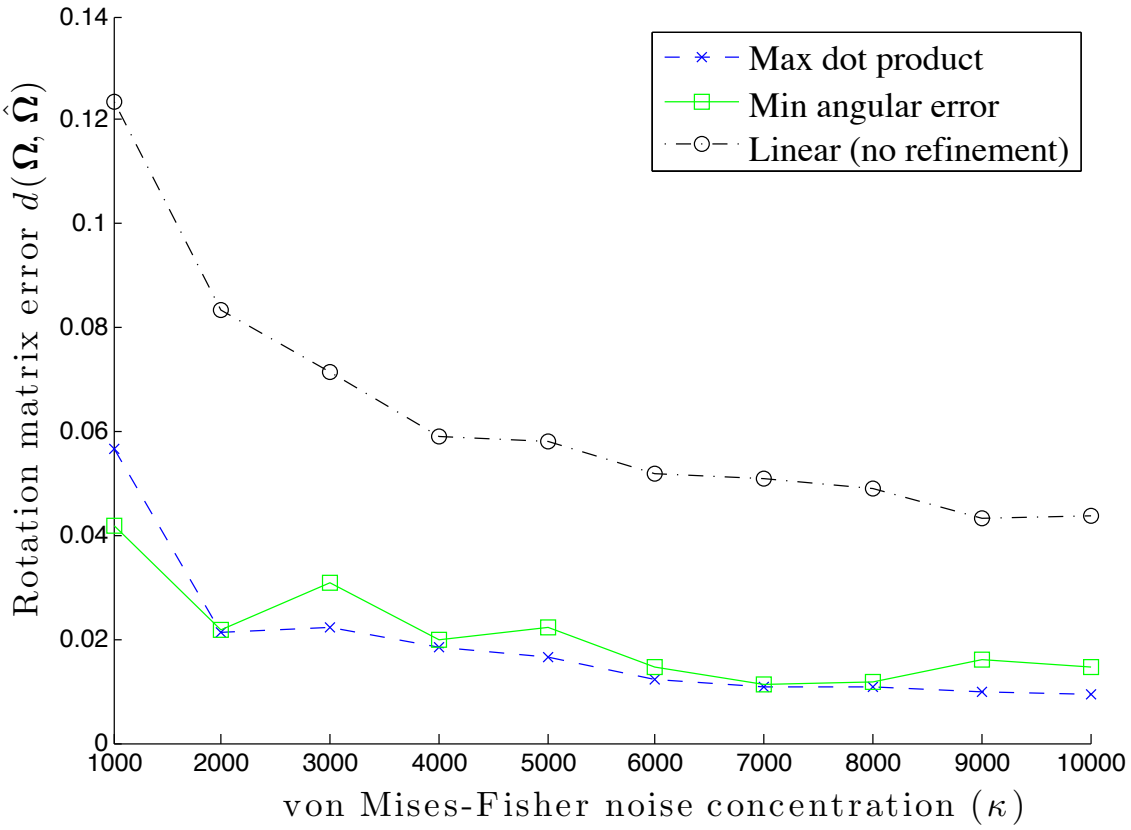


Figure 5.5: Quantitative errors in estimated rotation matrices.

5.3.1.2 Qualitative Results

We now qualitatively evaluate our view stabilisation process on some real, first person spherical videos. In the first video sequence[1] a skier is descending a piste and captures video using a Freedom360 spherical mount and 6 GoPro Hero3 Black video cameras and using the same rig as in Figure 4.2(a). The sequence is "third person" in that the camera rig is mounted on a monopod attached to the backpack of the skier (hence, the rig moves with the skier but does not turn with their heads, as in a first person view). We show qualitative results from a portion of this sequence and also a mountain bike sequence captured using the same setup in Figures 5.8, 5.9, 5.10 and 5.11 [2]. In the selected frames, the skier makes a 180° turn to the left and tilts left whilst doing so and the biker makes a 90° degree turn.

Hence, in Figure 5.8 it is evident in the raw frames from the sequence (shown in

[1]Video courtesy of: Ignacio Ferrando, Abaco Digital (www.abaco-digital.es).
[2]`https://www.youtube.com/channel/UC_C55XLPa1qGe3GFcpyGtBw`

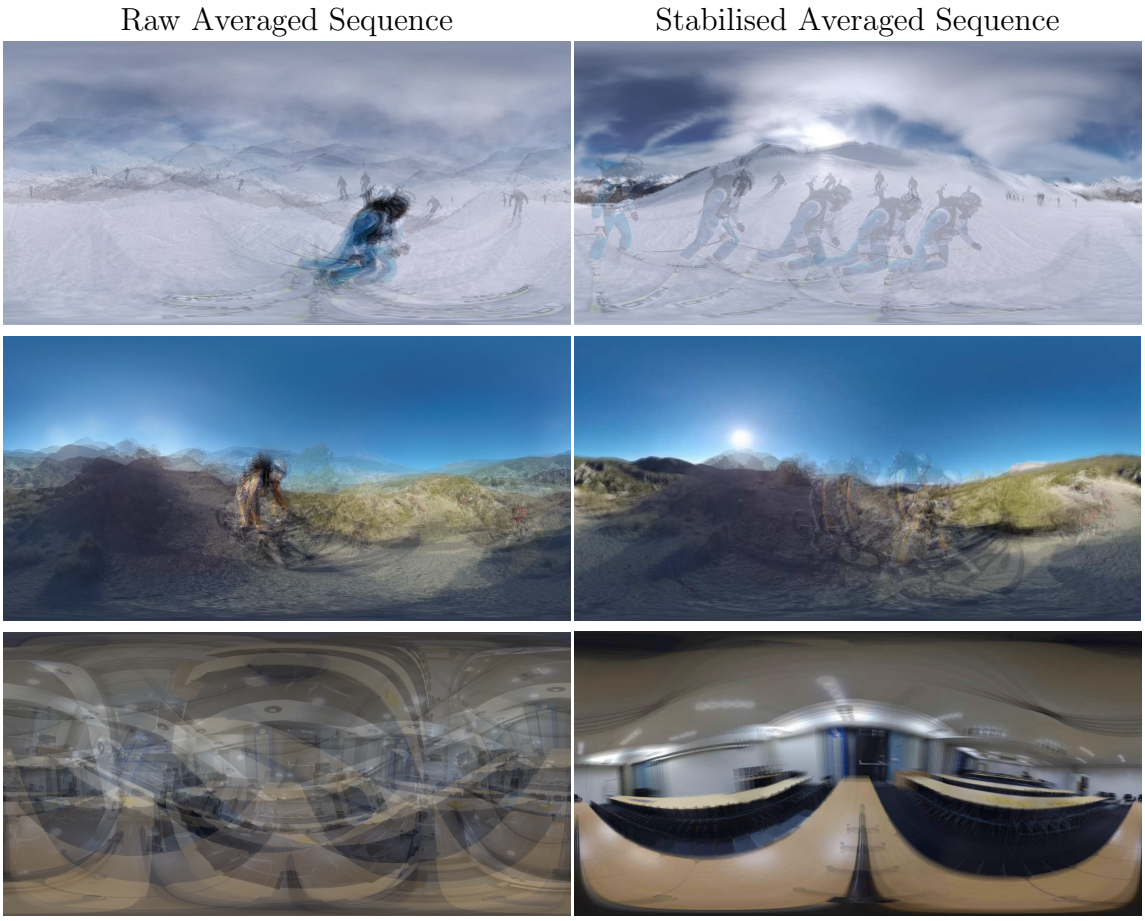


Figure 5.6: Averaged images for raw sequence (left) and stabilised sequence (right). From top to bottom, the datasets are about the skier, biker and office respectively.

the left column) that the environment is moving (note the position of the mountain peak that starts left of centre and the change in the shape of the horizon). On the other hand, the skier (who is approximately fixed relative to the camera) remains in the same position. In the stabilised frames (shown in the right hand column), we have rotated each frame back to the pose of the first frame in the sequence using the rotation matrix estimated by our spherical SFM pipeline. The effect is that distant points (whose direction remains approximately constant in the world coordinate system) are stabilised to an approximately constant position. The effect of viewing the stabilised sequence as a video is of following the same trajectory as in the original but with viewing direction remaining fixed. To further illustrate this, in Figure 5.6 we show images produced by averaging the raw (left) and stabilised (right) frames. In the raw sequence, the blurred mountains make it clear that view direction is

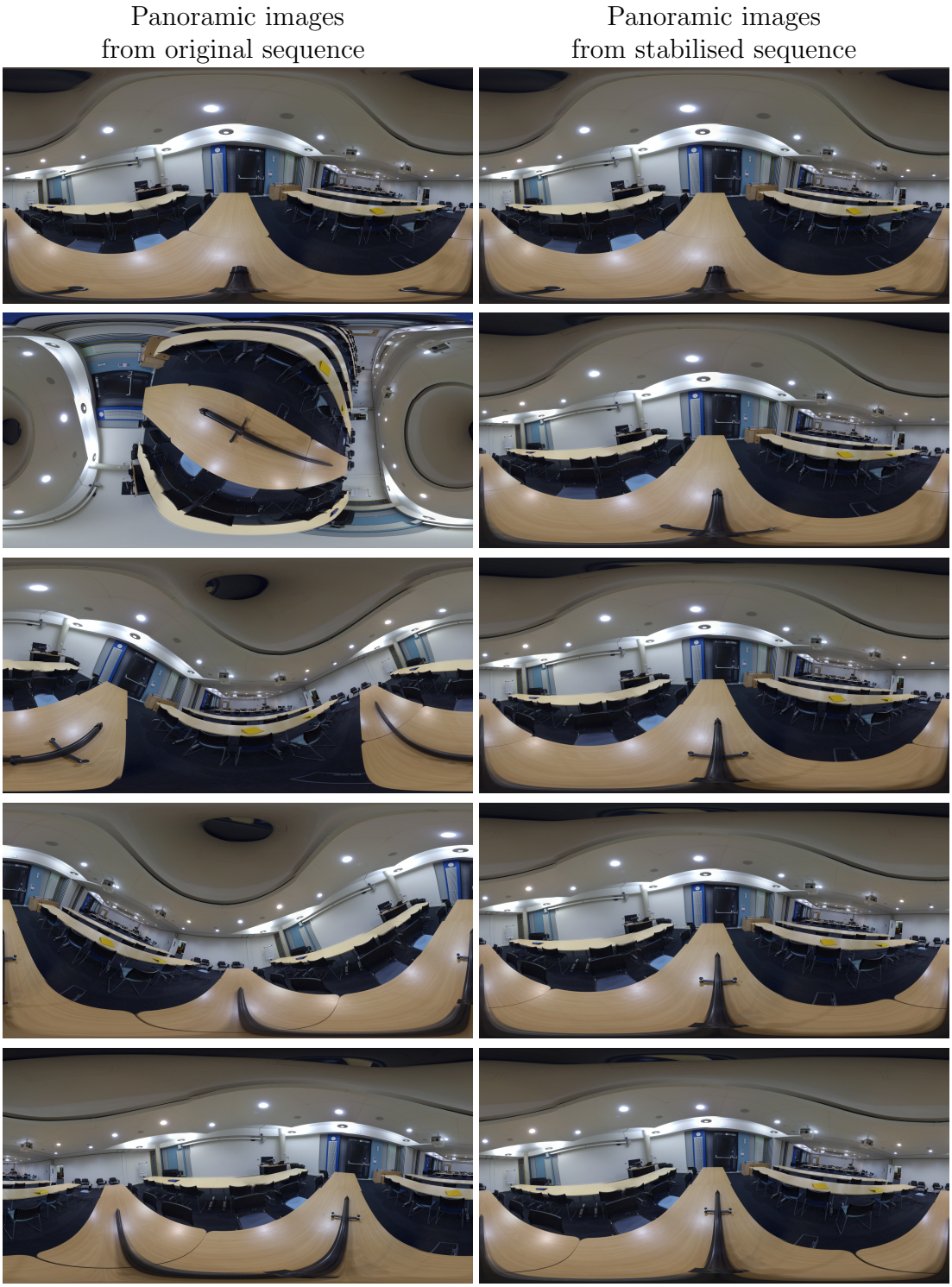


Figure 5.7: Raw panoramic frames from the spherical video sequence (left) and stabilised frames (right).

changing relative to the world. However, in the stabilised version, distant points are clearly visible and hence appear in approximately the same position in each frame. More results are shown in Figure 5.9 and 5.6. We also use the data we capture ourselves which has no static object (e.g. the people who wear the camera) in the scene. As the result shows in Figure 5.6 and 5.7 it stabilizes the view significantly.

To further illustrate this, in Figure 5.10 and Figure 5.11 we use the raw and stabilised panoramic images to render a virtual pinhole (perspective) view facing along the positive z-axis (roughly the direction of travel) and with a horizontal and vertical field of view of 116° and 83° respectively. This corresponds to the sort of view that may be produced when interactively viewing 360 videos. The effect of stabilisation is now very clear. We have also experimented with viewing the raw and stabilised videos on an Oculus Rift head mounted display. Anecdotally, there is a dramatic difference between the two and the sense of immersion is improved by having control over viewing direction.

5.3.2 View Interpolation And Extrapolation Results

Finally, we provide a qualitative evaluation of our view interpolation and extrapolation method. We begin with a simple example where the input images differ only in rotation. Although correspondence could be achieved here in a trivial way using the known rotations, we compute them using block matching to provide an indication of performance when there are no viewpoint effects to consider. We show the results in Figure 5.12 for a sequence of three rotations where we wish to interpolate a fourth intermediate rotation. On the left we show the synthesised image and the corresponding ground truth. Some noise is evident, primarily around sharp features where a small error in correspondence leads to noticeable artefacts.

Next, we interpolate a novel viewpoint from a sequence that we captured (the same as used in the previous chapter). The sequence includes both linear translation and arbitrary rotation and we interpolate a viewpoint that is 20cm from the closest view. Results are shown in Figure 5.14. Again, there is a good agreement between predicted and ground truth views. The key elements of the scene are predicted to be in the correct position, though again there is noise evident, this time primarily

Figure 5.8: Raw panoramic frames from the spherical video sequence (left) and stabilised frames (right).

Figure 5.9: Raw panoramic frames from the spherical video sequence (left) and stabilised frames (right).

Figure 5.10: Virtual perspective views from the raw spherical video sequence (left) and the stabilised sequence (right).

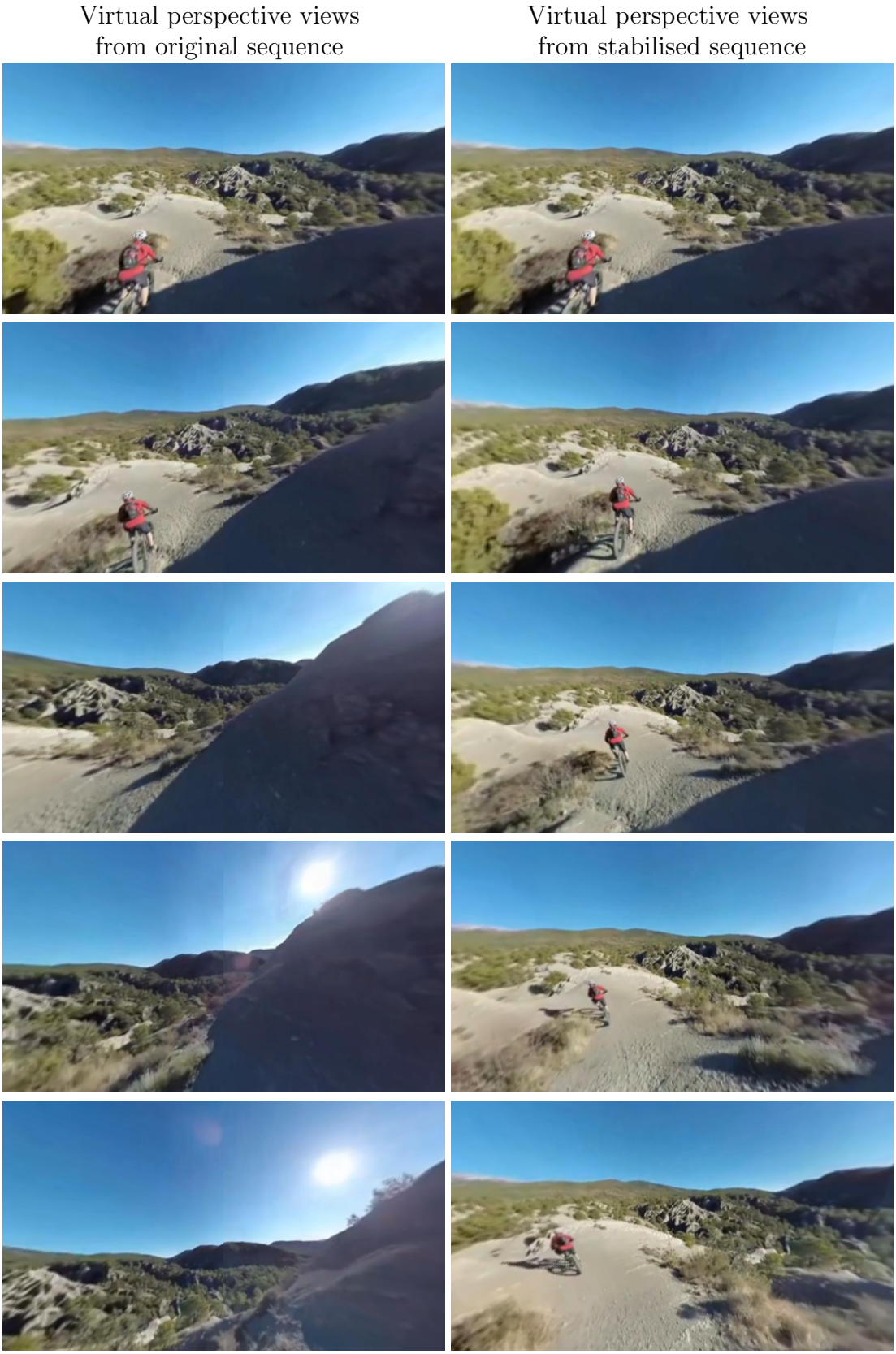


Figure 5.11: Virtual perspective views from the raw spherical video sequence (left) and the stabilised sequence (right).

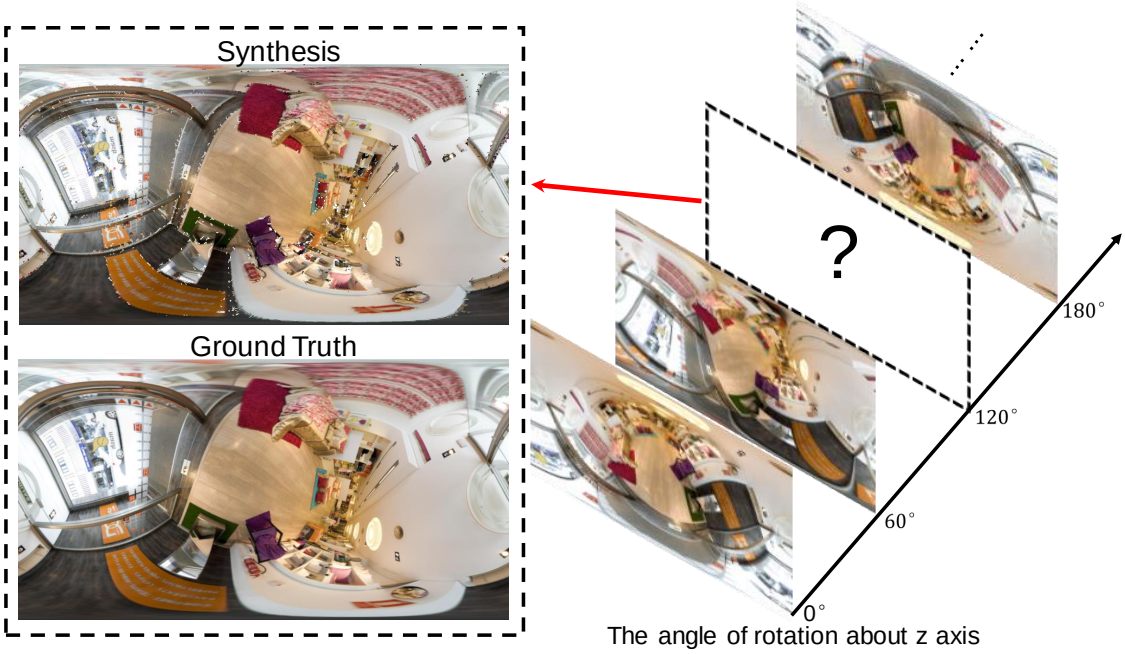


Figure 5.12: The interpolated image result. The synthetic image (left-top) and raw image (left-bottom) comparison; The image sequence is shown in right part.

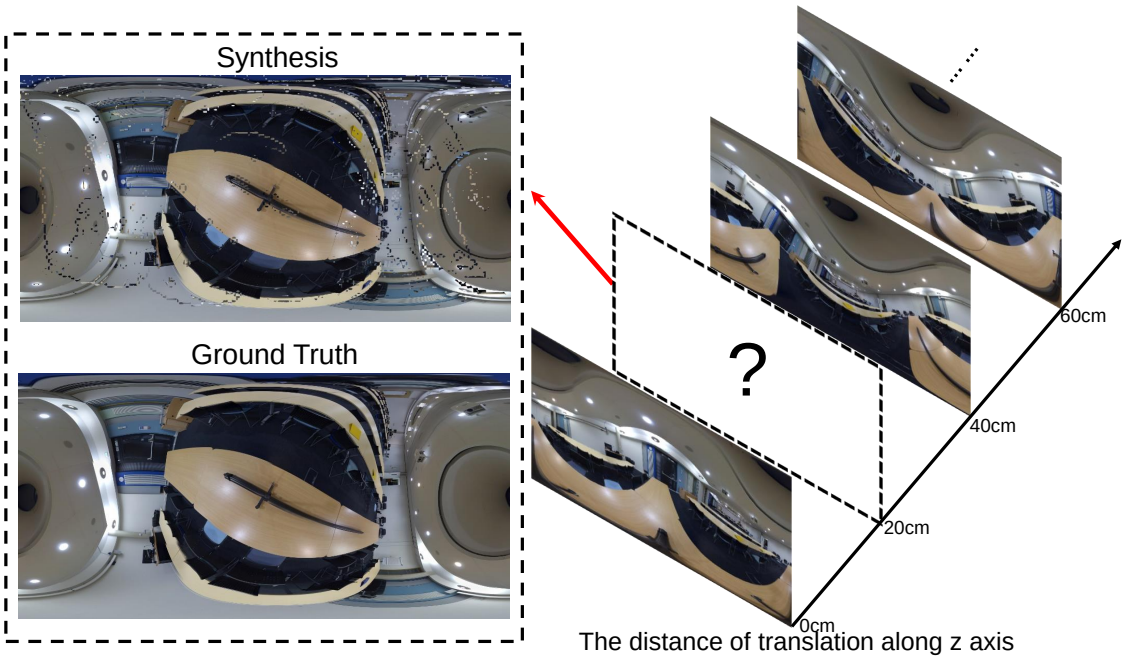


Figure 5.13: The interpolated synthesis image result. The synthetic image (left-top) and raw image (left-bottom) comparison; The image sequence is shown in right part.

in textured regions.

Finally, in Figure 5.14 we extrapolate an image with a viewpoint outside the range of the input images (20cm beyond the end of the captured sequence). Clearly,

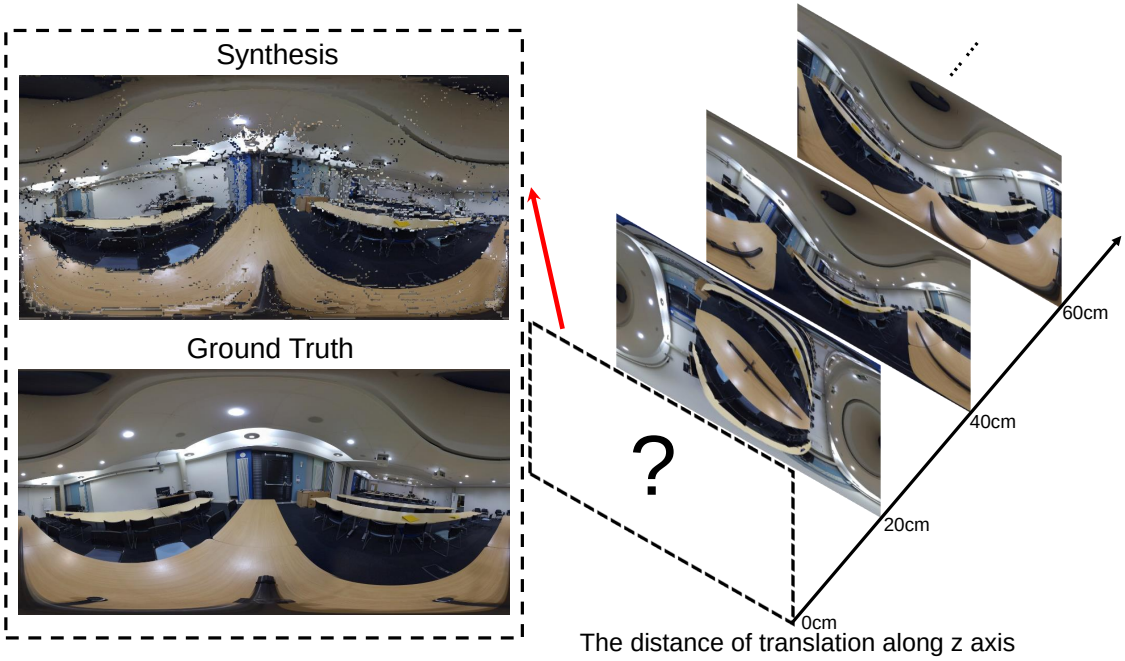


Figure 5.14: The extrapolated synthesis image result. The synthetic image (left-top) and raw image (left-bottom) comparison; The image sequence is shown in right part.

this is more challenging and the result begins to break down. However, at a coarse level, again the overall structure of the scene is correctly predicted. In an extrapolation scenario, additional spatial smoothness terms on the estimated scene depth would clearly help.

5.4 Conclusion

In this chapter we have presented two approaches to spherical view synthesis and some preliminary results in this direction. Our view stabilisation results show that our structure-from-motion pipeline is sufficiently robust that it can be applied to challenging real world sequences and yield much improved stability of view direction. We note that our SFM pipeline currently includes no temporal smoothness term and so it is likely that these results could be improved further with this addition. Our novel viewpoint synthesis approach is very simple yet shows in principle that new viewpoints can be synthesised using only the estimated camera poses. The weakness is that a lack of a principled spatial smoothness term means that the synthesised

views can contain holes, discontinuities and other artefacts. The obvious extension is to combine the two approaches enabling stabilisation of both viewing direction and viewpoint. This is something that has already been addressed in the literature for planar perspective images but has not yet received attention for spherical images. Finally, it is also obvious that discarding the 3D information obtained by SFM (even though it is sparse) is wasteful and it would make sense to incorporate this into the scene depth estimation.

Chapter 6

Conclusion

In this chapter, we summarise the contributions made in this thesis, critique the work undertaken and make some suggestions for future work.

We have presented contributions in the area of computer vision for spherical images. Specifically, we have made contributions to the problems of feature extraction and description, structure-from-motion and novel view synthesis.

We began in Chapter 3 with the low level problem of extracting local features. We develop an interest point detector and binary feature descriptor for spherical images which is inspired by the recent success of the binary features. We adapt the BRISK framework to operate on spherical images. All of our processing is intrinsic to the sphere and avoids the distortion inherent in storing and indexing spherical images in a 2D representation. We discretise images on a spherical geodesic grid formed by recursive subdivision of a triangular mesh. This leads to a multiscale pixel grid comprising mainly hexagonal pixels that lends itself naturally to a spherical image pyramid representation. For interest point detection, we use a variant of the Accelerated Segment Test (AST) corner detector which operates on our geodesic grid. We estimate a continuous scale and location for features and descriptors are built by sampling onto a regular pattern in the tangent space. We evaluated the repeatability, precision and recall of our proposed feature on both synthetic spherical images with known ground truth correspondences and real images and tested robustness to changes in viewpoint, rotation, illumination and noise. We conclude that our feature provides state-of-the-art performance for the problem of matching

spherical images under a range of conditions.

We present a complete pipeline for computing SFM from sequences of spherical images. We revisit problems from multiview geometry in the context of spherical images. In particular, we propose methods suited to spherical camera geometry for the spherical-n-point problem (estimating camera pose for a spherical image) and calibrated spherical reconstruction (estimating the position of a 3D point from multiple spherical images). An interesting result is that geodesic distance results in corrupted uncertainty estimates. While useful from a practical perspective, we argue that these distance measures are not justified by an explicit noise or probabilistic model. Our contribution in this regard is to use the von Mises-Fisher distribution to model uncertainty in spherical image formation and use this derive a novel objective function based on maximisation of dot products. This model provides an alternate objective function that we use in bundle adjustment. We evaluate our methods quantitatively and qualitatively on both synthetic and real world data and show that our methods developed for spherical images outperform straightforward adaptations of methods developed for perspective images. As an application of our method, we use the structure-from-motion output to stabilise the viewing direction in fully spherical video.

Finally, we also took some preliminary steps towards stabilising spherical video and synthesising novel viewpoints. We achieve stabilisation by using the SFM camera pose estimates to rotate each image back to a canonical view. This has the effect of removing viewing direction changes. We then presented a simple method to synthesise new viewpoints given only the input images and camera pose estimates. Our experimental results show that the stabilisation method is applicable to real world sequences and is qualitatively successful at stabilising videos with dramatic changes in viewing direction. Our novel viewpoint synthesis results show only that the method is capable in principle of creating new views but that they are not photorealistic. We conclude that more work is needed in this direction to create views that are indistinguishable from real video.

6.1 Critical Analysis

In Chapter 3 we present our spherical binary feature: BRISK on the Sphere. To our knowledge, there are only two previous methods that build local features for spherical images and do so intrinsically on the sphere (SPHORB and SIFT on the Sphere). We were unable to compare to SPHORB as, at the time of writing, there is no publicly available implementation. From an implementation point of view, a major critique of our work is that we have been unable to test the computational efficiency of our features in a meaningful way since the implementation is only in a prototype form (implemented in MATLAB). For use in the real world, a fast C or C++ implementation would be required and then meaningful timing comparisons could be made to competing methods.

From a theoretical point of view, our solution to the problem of feature detection at intra-octaves is unsatisfactory. Since there is no way to subdivide the sphere to obtain an intra-octave tessellation with the required properties, we resorted to using a feature detector of a different size to find intra-octave features. It is then not clear how best to select the corner detection threshold for the two different feature detectors in order to ensure features of the same saliency are found at both octaves and intra-octaves. More generally, our method requires the selection of a number of parameters (e.g. number of octaves, size of sampling pattern, bit length of descriptor etc) and there is no obvious way to select these values other than intuition or through empirical evaluation.

In Chapter 4 we propose a full pipeline for spherical structure-from-motion. Again, the weaknesses of this work can be divided into practical issues and theoretical weaknesses. In practice, our spherical images were obtained with a rig of perspective cameras and the images stitched together. Since the cameras cannot be positioned to have coincident optical centres, there is some error in this stitching process. Hence, it could be argued that we should instead process the perspective images directly with a constraint that they move rigidly together, rather than viewing it as a spherical image processing problem.

There are a number of theoretical drawbacks to our approach. First, we include no temporal smoothness constraints. This is a positive in the sense that we could

process unordered sets of images. However, since our intended application is video sequences, it seems sensible to incorporate temporal smoothness. Second, we have no principled way to choose the first two images to begin the reconstruction. We simply take the first two in the sequence. It is possible that these could be degenerate (i.e. no camera motion) and the reconstruction therefore of very low quality when preceding frames are added. Third, our bundle adjustment objective leads to a nonlinear optimisation problem. Unlike in standard SFM however, it is not a nonlinear least squares problem and so we cannot take advantage of the highly efficient tools available for sparse nonlinear least squares optimisation. Instead, we must resort to generic nonlinear optimisers with a resultant drop in speed.

In Chapter 5 we present two methods for video stabilisation and view synthesis. Although both methods prove the principle that our pipeline can lead to view synthesis from spherical video, both are methodologically quite simple and suffer from a number of weaknesses. The most obvious criticism of the view stabilisation method is that it stabilises only viewing direction, not viewpoint. To do so, we could either have followed an image-based warping approach or otherwise have developed methods for full dense 3D reconstruction and rendered new views from the model. The novel view synthesis method we did propose took neither of these approaches. Instead, it did exploit 3D pose information but only reconstructed the scene implicitly (finding the most photo-consistent depth implied a depth measurement for each pixel but we did not use this for 3D reconstruction). Although this leads to a simple implementation, this approach has serious drawbacks. First, we do not deal with occlusions. This means the most photo-consistent depth will be different for different cameras depending on whether they are occluded or not. Second, our use of averaging to combine colours from the input images means that occlusions will introduce artefacts into the averaged colour. Third, we did not impose any spatial smoothness prior (apart from implicitly through the multiscale approach). In stereo vision it is common to assume that depth is piecewise smooth. This would surely help our method obtain better synthesised views.

6.2 Future work

There are a number of ways that feature descriptors could be extended. First, there is scope to explore alternative sampling patterns and comparison pairs, exploiting any developments in 2D feature description that may improve performance. Second, we would like to apply our features to applications such as structure-from-motion with panoramic images or realtime visual odometry with omnidirectional cameras. Third, the approach could be extended to dense matching to enable applications such as motion segmentation, optical flow or stereo to be addressed. Finally, an interesting alternate approach would be to extend deep learning methods to our spherical image representation. For example, a CNN could operate on our geodesic grid with convolution operations taking place on the sphere. This would allow us to learn features on the sphere that may be appropriate for higher level visual tasks.

We could extend our SFM pipeline in a number of ways. First, we could investigate other methods like image segmentation to robustly remove object in the scene that move statically with the camera. Second, we believe that an efficient optimisation method could be designed to deal specifically with our vMF objective function, in the same spirit as previous work on sparse bundle adjustment. Third, we would like to explore temporal smoothness constraints on the camera motion trajectory. This incorporate prior information such as that the camera is moving primarily on the ground plane. Fourth, for unordered image sequences we could develop methods to automatically choose a pair of images to initialise the reconstruction. Fifth, we could consider the problem of 'loop closure' by comparing new images to the whole previous sequence to find matches between overlapping trajectories.

There are also many areas of future work for SFM application. First, we would like to explicitly cluster points into those moving with the camera and those fixed in the world. This is important for first or third person video sequences. Second, we would like to conduct perceptual experiments to verify that the stabilised sequences provide a better experience when viewed by humans via a head mounted display. Finally, we would like to explore viewpoint interpolation in more detail and investigate whether the mesh-warping based stabilisation algorithms that have proven effect for perspective images can be extended to the spherical case. This would allow both

viewpoint stabilisation but also other applications such as spherical hyper-lapse or free viewpoint video from motion sequences.

Abbreviations

FPV	First Person Video
GIS	Geographic Information Systems
SIFT	Scale-invariant feature transform
BRISK	Binary robust invariant scalable keypoints
CNN	Convolutional neural network
SUSAN	Smallest Univalue Segment Assimilating Nucleus Test
FAST	Features from Accelerated Segment Test
AGAST	Adaptive and Generic Corner Detection Based on the Accelerated Segment Test
SURF	Speeded-Up robust features
AST	Accelerated Segment Test
ORB	Oriented FAST and rotated BRIEF
PCA	Principal Components Analysis
GLOH	Gradient Location-Orientation Histogram
SLAM	Simultaneous location and mapping
BRIEF	Binary robust independent elementary features

SPHORB	Spherical extension of the ORB descriptor
QTM	Quaternary Triangle Mesh
SFM	Structure-from-motion
DLT	Direct linear transform
vMF	Von Mises-Fisher distribution
SVD	Singular value decomposition
RANSAC	Random Sample Consensus
PnP	perspective-n-point
SnP	spherical-n-point
QCQP	quadratically-constrained quadratic programming

References

[1] Y. I. Abdel-Aziz and H. M. Karara. Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In *Proc. Symposium on Close-Range Photogrammetry*, 1971.

[2] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. CenSurE: Center surround extremas for realtime feature detection and matching. In *Proc. ECCV*, pages 102–115, 2008.

[3] Henrik Andreasson, André Treptow, and Tom Duckett. Self-localization in non-stationary environments using omni-directional vision. *Robotics and Autonomous Systems*, 55(7):541–551, 2007.

[4] Zafer Arican and Pascal Frossard. Sampling-aware polar descriptors on the sphere. In *Proc. ICIP*, pages 3509–3512. IEEE, 2010.

[5] J. Bai, A. Agarwala, M. Agrawala, and R. Ramamoorthi. User-assisted video stabilization. *Computer Graphics Forum*, 33(4), 2014.

[6] Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. In *Journal of Machine Learning Research*, pages 1345–1382, 2005.

[7] Joao P Barreto and Helder Araujo. Issues on the geometry of central catadioptric image formation. In *Proc. CVPR*, volume 2, pages II–422. IEEE, 2001.

[8] Yalin Bastanlar, Alptekin Temizel, Yasemin Yardimci, and Peter Sturm. Multi-view structure-from-motion for hybrid camera scenarios. *Image and Vision Computing*, 30(8):557–572, 2012.

[9] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. ECCV*, pages 404–417, 2006.

[10] Jean Charles Bazin, Kuk-Jin Yoon, IS Kweon, Cedric Demonceaux, and Pascal Vasseur. Particle filter approach adapted to catadioptric images for target tracking application. In *Proc. BMVC*, 2009.

[11] H. E. Benseddik, H. Hadj-Abdelkader, B. Cherki, and O. Djekoune. Binary feature descriptor for omnidirectional images processing. In *Proc. IPAC*, number 81, 2015.

[12] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vis.*, 74(1):59–73, 2007.

[13] Chris Buehler, Michael Bosse, and Leonard McMillan. Non-metric image-based rendering for video stabilization. In *Proc. CVPR*, volume 2, pages II–609. IEEE, 2001.

[14] Grigore C Burdea and Philippe Coiffet. *Virtual reality technology*, volume 1. John Wiley & Sons, 2003.

[15] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proc. ECCV*, pages 778–792. Springer, 2010.

[16] Josip Ćesić, Ivan Marković, and Ivan Petrović. Moving objects tracking on the unit sphere using a multiple-camera system on a mobile robot. In *Proc. Intelligent Autonomous Systems*, pages 899–911, 2016.

[17] Peng Chang and Martial Hebert. Omni-directional structure from motion. In *Proc. Omnidirectional Vision*, pages 127–133. IEEE, 2000.

[18] Sunghyun Cho, Jue Wang, and Seungyong Lee. Video deblurring for hand-held cameras using patch-based synthesis. *ACM Transactions on Graphics (TOG)*, 31(4):64, 2012.

[19] Javier Cruz-Mota, Iva Bogdanova, Benoît Paquier, Michel Bierlaire, and Jean-Philippe Thiran. Scale invariant feature transform on the sphere: Theory and applications. *International journal of computer vision*, 98(2):217–241, 2012.

[20] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer Verlag, Berlin, 1997.

[21] M-A Drouin, Martin Trudeau, and Sebastien Roy. Geo-consistency for wide multi-camera stereo. In *Proc. CVPR*, volume 1, pages 351–358. IEEE, 2005.

[22] Mark Fiala and Gerhard Roth. Automatic alignment and graph map building of panoramas. In *Proc. IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, 2005.

[23] N. I. Fisher, T. Lewis, and B. J. J. Embleton. *Statistical Analysis of Spherical Data*. Cambridge University Press, 1987.

[24] Yasutaka Furukawa. *High-fidelity image-based modeling*. ProQuest, 2008.

[25] Pau Gargallo and Peter Sturm. Bayesian 3d modeling from images using multiple depth maps. In *Proc. CVPR*, volume 2, pages 885–891. IEEE, 2005.

[26] C. C. Gava and D. Stricker. SPHERA - a unifying structure from motion framework for central projection cameras. In *Proc. International Conference on Computer Vision Theory and Applications (VISAPP)*, 2015.

[27] M. L. Gleicher and F. Liu. Re-cinematography: Improving the camera dynamics of casual video. In *Proc. 15th International Conference on Multimedia*, pages 27–36, 2007.

[28] Toon Goedemé, Tinne Tuytelaars, Luc Van Gool, Gerolf Vanacker, and Marnix Nuttin. Omnidirectional sparse visual path following with occlusion-robust

feature tracking. In *Proc. Workshop on omnidirectional vision, camera networks and non-classical cameras (OMNIVIS)*, 2005.

[29] Michael Goesele, Jens Ackermann, Simon Fuhrmann, Carsten Haubold, Ronny Klowsky, Drew Steedly, and Richard Szeliski. Ambient point clouds for view interpolation. *ACM Transactions on Graphics (TOG)*, 29(4):95, 2010.

[30] Amit Goldstein and Raanan Fattal. Video stabilization using epipolar geometry. *ACM Transactions on Graphics (TOG)*, 31(5):126, 2012.

[31] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[32] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[33] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Proc. CVPR*, pages 225–232. IEEE, 2011.

[34] H. Guan and W. A. P. Smith. Corner detection in spherical images via accelerated segment test on a geodesic grid. In *Proc. International Symposium on Visual Computing*, pages 407–415, 2013.

[35] Tobias Gurdan, Martin R Oswald, Daniel Gurdan, and Daniel Cremers. Spatial and temporal interpolation of multi-view image sequences. In *Pattern Recognition*, pages 305–316. Springer, 2014.

[36] Daniel Gutierrez, Alejandro Rituerto, JMM Montiel, and JJ Guerrero. Adapting a real-time monocular visual slam from conventional to omnidirectional cameras. In *Proc. International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 343–350. IEEE, 2011.

[37] Peter Hansen, Peter Corke, Wageeh Boles, and Kostas Daniilidis. Scale invariant feature matching with wide angle images. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1689–1694. IEEE, 2007.

[38] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proc. Alvey vision conference*, volume 15, page 50. Citeseer, 1988.

[39] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[40] Jared Heinly, Enrique Dunn, and Jan-Michael Frahm. Comparative evaluation of binary features. In *Proc. ECCV*, pages 759–773, 2012.

[41] D. Q. Huynh. Metrics for 3D rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009.

[42] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[43] M. Kamali, A. Banno, J. C. Bazin, I. S. Kweon, and K. Ikeuchi. Stabilizing omnidirectional videos using 3D structure and spherical image warping. In *IAPR Conference on Machine Vision Applications*, pages 177–180, 2011.

[44] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proc. CVPR*, 2004.

[45] Johannes Kopf, Michael F Cohen, and Richard Szeliski. First-person hyperlapse videos. *ACM Transactions on Graphics (TOG)*, 33(4):78, 2014.

[46] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[47] B. Krolla, C.C. Gava, A. Pagani, and D. Stricker. Consistent pose uncertainty estimation for spherical cameras. In *Proc. International Conference on Computer Graphics, Visualization and Computer Vision*, 2014.

[48] Ken-Yi Lee, Yung-Yu Chuang, Bing-Yu Chen, and Ming Ouhyoung. Video stabilization using robust feature trajectories. In *Proc. Computer Vision*, pages 1397–1404. IEEE, 2009.

[49] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proc. ICCV*, pages 2548–2555. IEEE, 2011.

[50] Maxime Lhuillier. Effective and generic structure from motion using angular error. In *Proc. ICPR*, volume 1, pages 67–70, 2006.

[51] Xiangru Li and Zhanyi Hu. Rejecting mismatches by correspondence function. *International Journal of Computer Vision*, 89(1):1–17, 2010.

[52] Yunpeng Li, Sing Bing Kang, Neel Joshi, Steven M Seitz, and Daniel P Huttenlocher. Generating sharp panoramas from motion-blurred videos. In *Proc. CVPR*, pages 2424–2431. IEEE, 2010.

[53] Tony Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vis.*, 30(2):79–116, 1998.

[54] F. Liu, M. L. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3D video stabilization. *ACM Trans. Graph.*, 28(3):44:1–44:9, 2009.

[55] F. Liu, M. L. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Trans. Graph.*, 30(1):4:1–4:10, 2011.

[56] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled camera paths for video stabilization. *ACM Trans. Graph. (Proceedings of SIGGRAPH)*, 32(4), 2013.

[57] Shuaicheng Liu, Yinting Wang, Lu Yuan, Jiajun Bu, Ping Tan, and Jian Sun. Video stabilization with a depth camera. In *Proc. CVPR*, pages 89–95. IEEE, 2012.

[58] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *Proc. CVPR*, pages 4209–4216. IEEE, 2014.

[59] Si Jing Liu, Sonya Coleman, Dermot Kerr, Bryan Scotney, and Bryan Gardiner. Corner detection on hexagonal pixel based images. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1025–1028. IEEE, 2011.

[60] Yonghuai Liu, Luigi De Dominicis, Baogang Wei, Liang Chen, and Ralph R Martin. Regularization based iterative point match weighting for accurate rigid transformation estimation. *IEEE transactions on visualization and computer graphics*, 21(9):1058–1071, 2015.

[61] Manolis Lourakis and Antonis Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical report, Technical Report 340, Institute of Computer Science-FORTH, Heraklion, Crete, Greece, 2004.

[62] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004.

[63] Jiayi Ma, Yong Ma, Ji Zhao, and Jinwen Tian. Image feature matching via progressive vector field consensus. *IEEE Signal Processing Letters*, 22(6):767–771, 2015.

[64] Jiayi Ma, Ji Zhao, Jinwen Tian, Xiang Bai, and Zhuowen Tu. Regularized vector field learning with sparse approximation for mismatch removal. *Pattern Recognition*, 46(12):3519–3532, 2013.

[65] Jiayi Ma, Ji Zhao, Jinwen Tian, Alan L Yuille, and Zhuowen Tu. Robust point matching via vector field consensus. *IEEE Transactions on Image Processing*, 23(4):1706–1721, 2014.

[66] Jiayi Ma, Ji Zhao, and Alan L Yuille. Non-rigid point set registration by preserving global and local structures. *IEEE Transactions on image Processing*, 25(1):53–64, 2016.

[67] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Proc. ECCV*, pages 183–196. Springer, 2010.

[68] Ivan Markovic, François Chaumette, and Ivan Petrovic. Moving object detection, tracking and following using an omnidirectional camera on a mobile robot. In *Proc. ICRA*, pages 5630–5635, 2014.

[69] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization with motion inpainting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(7):1150–1163, 2006.

[70] Thomas Mauthner, Friedrich Fraundorfer, and Horst Bischof. Region matching for omnidirectional images using virtual camera planes. In *Proc. Computer Vision Winter Workshop*, 2006.

[71] Branislav Micusik and Tomas Pajdla. Para-catadioptric camera auto-calibration from epipolar geometry. In *Proc. ACCV*, pages 748–753, 2004.

[72] L. Middleton and J. Sivaswamy. *Hexagonal Image Processing*. Springer, 2005.

[73] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, 2005.

[74] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1-2):43–72, 2005.

[75] Ondrej Miksik and Krystian Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *Proc. ICPR*, pages 2681–2684, 2012.

[76] Jean-Michel Morel and Guoshen Yu. ASIFT: A new framework for fully affine invariant image comparison. *SIAM J. Imaging Sci.*, 2(2):438–469, 2009.

[77] Ana Cristina Murillo, Daniel Gutiérrez-Gómez, Alejandro Rituerto, Luis Puig, and Josechu J Guerrero. Wearable omnidirectional vision system for personal localization and guidance. In *Proc. Egocentric (First-Person) Vision*, pages 8–14, 2012.

[78] Frank Nielsen. Surround video: a multihead camera approach. *The Visual Computer*, 21(1-2):92–103, 2005.

[79] Julia Alison Noble. *Descriptions of image surfaces.* PhD thesis, University of Oxford, 1989.

[80] A. Pagani and D. Stricker. Structure from motion using full spherical panoramic cameras. In *Proceeding of the 11th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, 2011.

[81] Alain Pagani, Christiano Gava, Yan Cui, Bernd Krolla, Jean-Marc Hengen, and Didier Stricker. Dense 3d point cloud generation from multiple high-resolution spherical images. In *Proc. International Conference on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, pages 17–24, 2011.

[82] Gerard Pons-Moll, Andreas Baak, Juergen Gall, Laura Leal-Taix, Meinard Mueller, Hans-Peter Seidel, and Bodo Rosenhahn. Outdoor human motion capture using inverse kinematics and von mises-fisher sampling. In *Proc. ICCV*, pages 1243–1250, 2011.

[83] Alejandro Rituerto, Luis Puig, and JJ Guerrero. Visual SLAM with an omnidirectional camera. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 348–351. IEEE, 2010.

[84] Sammy Rogmans, Jiangbo Lu, Philippe Bekaert, and Gauthier Lafruit. Real-time stereo-based view synthesis algorithms: A unified framework and evaluation on commodity GPUs. *Signal Processing: Image Communication*, 24(1):49–64, 2009.

[85] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(1):105–119, 2010.

[86] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *Int. J. Comput. Vis.*, 66(3):231–259, 2006.

[87] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proc. ICCV*, pages 2564–2571, 2011.

[88] Kevin Sahr, Denis White, and A Jon Kimerling. Geodesic discrete global grid systems. *Cartography and Geographic Information Science*, 30(2):121–134, 2003.

[89] Davide Scaramuzza and Roland Siegwart. Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Trans. Robotic. Autom.*, 24(5):1015–1026, 2008.

[90] Benno Schmeing, Thomas Läbe, and Wolfgang Förstner. Trajectory reconstruction using long sequences of digital images from an omnidirectional camera. *DGPF Tagungsband*, 2011.

[91] Stephen M Smith and J Michael Brady. Susana new approach to low level image processing. *International journal of computer vision*, 23(1):45–78, 1997.

[92] Peter Sturm and Srikumar Ramalingam. A generic concept for camera calibration. In *Proc. ECCV*, pages 1–13, 2004.

[93] Jean-Philippe Tardif, Yanis Pavlidis, and Kostas Daniilidis. Monocular visual odometry in urban environments. In *Proc. IROS*, pages 2531–2538, 2008.

[94] A. Torii, I. Atsushi, and O. Naoya. Two-and three-view geometry for spherical cameras. In *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, 2005.

[95] Akihiko Torii, Michal Havlena, and Tomas Pajdla. From Google street view to 3D city models. In *Proc. OMNIVIS*, pages 2188–2195, 2009.

[96] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Found. Trends. Comp. Graphics and Vision.*, 3(3):177–280, 2008.

[97] M Vidal-Naquet and S Ullman. Object recognition with informative features and linear classification. In *Proc. ICCV*, pages 281–288, 2003.

[98] Mirjam Vosmeer and Ben Schouten. Interactive cinema: engagement and interaction. In *Interactive Storytelling*, pages 140–147. Springer, 2014.

[99] Yu-Shuen Wang, Feng Liu, Pu-Sheng Hsu, and Tong-Yee Lee. Spatially and temporally optimized video stabilization. *IEEE transactions on visualization and computer graphics*, page 1, 2013.

[100] A. T. A. Wood. Simulation of the von Mises Fisher distribution. *Communications in statistics-simulation and computation*, 23(1):157–164, 1994.

[101] Oliver J Woodford, Ian D Reid, Philip HS Torr, and Andrew W Fitzgibbon. On new view synthesis using multiview stereo. In *Proc. BMVC*, pages 1–10, 2007.

[102] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *Proc. CVPR*, pages 2695–2702, 2012.

[103] Tianli Yu, Ning Xu, and Narendra Ahuja. Shape and view independent reflectance map from multiple views. *International journal of computer vision*, 73(2):123–138, 2007.

[104] Qiang Zhao, Wei Feng, Liang Wan, and Jiawan Zhang. SPHORB: a fast and robust binary feature on the sphere. *Int. J. Comput. Vis.*, 113(2):143–159, 2015.

[105] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered

representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM, 2004.