Multi-Scale Methods for Omnidirectional Stereo with Application to Real-Time Virtual Walkthroughs

by

Alan Brunton

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Ph.D. degree in
Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

© Alan Brunton, Ottawa, Canada, 2012

Abstract

This thesis addresses a number of problems in computer vision, image processing, and geometry processing, and presents novel solutions to these problems. The overarching theme of the techniques presented here is a multi-scale approach, leveraging mathematical tools to represent images and surfaces at different scales, and methods that can be adapted from one type of domain (eg., the plane) to another (eg., the sphere).

The main problem addressed in this thesis is known as stereo reconstruction: reconstructing the geometry of a scene or object from two or more images of that scene. We develop novel algorithms to do this, which work for both planar and spherical images. By developing a novel way to formulate the notion of disparity for spherical images, we are able effectively adapt our algorithms from planar to spherical images.

Our stereo reconstruction algorithm is based on a novel application of distance transforms to multi-scale matching. We use matching information aggregated over multiple scales, and enforce consistency between these scales using distance transforms. We then show how multiple spherical disparity maps can be efficiently and robustly fused using visibility and other geometric constraints. We then show how the reconstructed point clouds can be used to synthesize a realistic sequence of novel views, images from points of view not captured in the input images, in real-time.

Along the way to this result, we address some related problems. For example, multi-scale features can be detected in spherical images by convolving those images with a filterbank, generating an overcomplete spherical wavelet representation of the image from which the multi-scale features can be extracted. Convolution of spherical images is much more efficient in the spherical harmonic domain than in the spatial domain. Thus, we develop a GPU implementation for fast spherical harmonic transforms and frequency domain convolutions of spherical images. This tool can also be used to detect multi-scale features on geometric surfaces.

When we have a point cloud of a surface of a particular class of object, whether generated by stereo reconstruction or by some other modality, we can use statistics and machine learning to more robustly estimate the surface. If we have at our disposal a database of surfaces of a particular type of object, such as the human face, we can compute statistics over this database to constrain the possible shape a new surface of this type can take. We show how a statistical spherical wavelet shape prior can be used to efficiently and robustly reconstruct a face shape from noisy point cloud data, including stereo data.

Acknowledgements

I foremost thank my supervisors, Jochen Lang and Eric Dubois for their important advice, encouragement and insight. They have been willing to assist in literature searches, to give comments on drafts, even on weekends as deadlines approach, and provide frank and realistic assessments of intermediate results. They have always had good ideas for how to improve both methods and presentation of results. Throughout they have allowed me to work independently, allowing me to learn by doing. They even tolerated a move to Germany for the final year of my studies. I have been very fortunate to have them as supervisors.

I thank Chang Shu for his collaboration, advice and his help in writing and presenting our results in Chapter 4. Chang has been an invaluable source of perspective and insight for many years.

I thank the members of the committee for their time, effort and feedback.

I thank Ahmad Al-Kabbany, Houman Rastgar and Kris Woodbeck for their help with data acquisition.

I thank Stefanie Wuhrer and Augusto Salazar for their assistance in improving Chapter 4. I also thank Augusto for providing the parameterized BU-3DFE face database.

This work was primarily supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). The work for Chapter 4 was also supported by the National Research Council of Canada (NRC). I thank the MMCI Cluster of Excellence at Saarland University for providing me with a place to work and technical support for the final year of my studies. I thank the graphics group at the Max Planck Institut für Informatik (Saarbrücken) for providing a stimulating environment for collaboration.

Contents

1 Introduction **1**
1.1 Motivation of the Problem 2
1.2 Problem Statement 3
1.3 Summary of Existing Methods 4
1.4 Overview of the Proposed Approach 7
1.5 Summary of Results 8

2 Background and Literature Review **10**
2.1 Notation 11
2.2 Spherical Representations 12
2.2.1 Spherical Harmonics 12
2.2.2 Spherical Convolution 15
2.2.3 Spherical Wavelets 17
2.2.4 Spherical Parametrizations 18
2.3 Wavelets and Multi-Scale Schemes in Stereo Matching 20
2.4 Model-based Stereo 23
2.4.1 Stereo Capture of Human Faces 26
2.5 Multi-View Stereo 27
2.6 Spherical Stereo 29
2.7 Image-based Modeling and Novel View Synthesis 30
2.8 Discussion 34

3 Spherical Harmonic Transforms and Convolution on the GPU for Panoramic Imaging **36**
3.1 Introduction 37
3.2 Parallel Spherical Harmonic Transforms Using CUDA 39
3.3 Examples 42

3.3.1 Speed 43
3.3.2 Numerical Accuracy 43
3.4 Discussion 45

4 A Wavelet Prior for Model-Based Stereo 50
4.1 Introduction 51
4.2 Overview 53
4.3 The Model 54
4.3.1 Subdivision Resampling 55
4.3.2 Wavelet Decomposition 55
4.3.3 Statistical Prior 56
4.3.4 Observations 58
4.4 The Energy Function and Minimization 59
4.5 Experiments 63
4.5.1 Implementation 63
4.5.2 Results 64
4.5.3 Additional Experiments 66
4.6 Discussion 69

5 Efficient Multi-Scale Stereo of High-Resolution Planar and Spherical Images 73
5.1 Introduction 73
5.1.1 Related Work 75
5.2 Multi-Scale Stereo Matching 76
5.2.1 Multi-Scale DSI Chaining with Distance Transforms 76
5.2.2 Initial DSI Computation 79
5.2.3 Time and Space Complexity 80
5.2.4 Disparity Post-Processing 81
5.3 Spherical Stereo 82
5.3.1 Spherical Disparity 82
5.3.2 Modifications of Planar Algorithm 86
5.4 Results 86
5.4.1 Planar Stereo 87
5.4.2 Spherical Stereo 90
5.5 Discussion 90

6 Spherical Stereo with Wavelet Filtering for Novel View Synthesis 94
6.1 Introduction 94
6.1.1 Related work 95
6.1.2 Overview 98
6.2 Disparity Estimation 99
6.2.1 Initial Disparity Estimation 99
6.2.2 Disparity Map Fusion 100
6.2.3 Data Capture and Calibration 102
6.3 Disparity Map Filtering 106
6.4 View Synthesis 107
6.5 Experiments 108
6.5.1 Implementation 108
6.5.2 Results 108
6.6 Discussion 116

7 Conclusions 117
7.1 Thesis Summary 117
7.2 Thesis Contributions 119
7.3 Future Work 121

List of Tables

3.1 Mean absolute coefficient error from inverse-forward round-trip transforms for different algorithms and implementations at different problem sizes. 44

5.1 Timings in seconds per image for the different stages of our algorithms for different image sets on our test system. The number of Mpixels of each image is given in parentheses. 87

5.2 Results for our method on the KITTI stereo benchmark. The leftmost column gives disparity error thresholds; pixels with disparity error greater than this value are considered outliers. The next two columns give the number of outliers for each threshold for non-occluded pixels and all pixels, respectively. The last two columns give the average disparity error for non-occluded and all pixels. 88

5.3 Ranking information for the KITTI stereo benchmark as of August 30, 2012. The third and fourth columns have the same meaning as their counterparts in Table 5.2, and the disparity error threshold used is 3 pixels. 89

List of Figures

2.1 A cubic checkerboard pattern resampled on the sphere via the spherical harmonic domain (maximum degree 179). 15
2.2 A Laplacian of Gaussian filter stereographically projected on the sphere via the spherical harmonic domain (maximum degree 179). 15
2.3 The spherical convolution of the image in Figure 2.1 with filter in Figure 2.2 in the spherical harmonic domain. 16
2.4 Diagram of stereographic projection and dilation. 17

3.1 Pseudocode listing of kernel performing parallel projection of summation vectors $\mathbf{s}_m$ onto associated Legendre vectors $\mathbf{P}_{lm}$. 40
3.2 Pseudocode listing of kernel to compute the inner product in cosine domain. . . 46
3.3 A spherically sampled panoramic image and convolution results with five filters from the filterbank. Note that the intensity levels in the convolution results are normalized to the range of values for each result. The higher frequency results have smaller ranges of values. 47
3.4 Running times for different forward SH transform algorithms and implementations for different problem sizes. Each data point was computed as the mean of 27 trials. Abbreviations: GPU-direct-C is the direct method, complex basis, on the GPU; GPU-direct-R is direct, real basis, on the GPU; GPU-semi is the semi-naive method, real basis, on the GPU; CPU-semi is the semi-naive method from SpharmonicKit; CPU-hybrid is the hybrid algorithm from SpharmonicKit. 48

3.5 Running times for different inverse SH transform algorithms and implementations for different problem sizes. Each data point was computed as the mean of 27 trials. Abbreviations: GPU-direct-C is direct method, complex basis, on the GPU; GPU-direct-R is direct method, real basis, GPU; GPU-semi is semi-naive, real basis, GPU; CPU-semi is the semi-naive method from SpharmonicKit; CPU-hybrid is the hybrid algorithm from SpharmonicKit. 49

4.1 Left: The surface $\bar{\mathbf{f}}$ reconstructed from the mean shape vector $\bar{\mathbf{s}}$. Right: false color visualization of the magnitude in metres of the standard deviation of the model parameters associated with each vertex in the full-resolution grid mesh, $|\sigma^k|$. 57
4.2 Left to right: input image, initial point cloud from PMVS [1], reconstruction before refinement (level 4), after refinement. 65
4.3 Reconstruction results for stereo pair used in Figure 4.2 with a point cloud generated using the method described in Chapter 5 (left) and the revised fitting procedure from Section 4.5.3, levels 3, 4 and 5, respectively. 65
4.4 Left to right: input image, initial point cloud, reconstruction, mean face for comparison. 65
4.5 Fitting the model to a laser scan. Left to right: point cloud, reconstruction before refinement, after refinement, original mesh. 67
4.6 Reconstruction results (second row) for the same data as in Figure 4.5 using the training and fitting method from Section 4.5.3. 67
4.7 Fitting the model to a laser scan. Left to right: point cloud, reconstruction before refinement, after refinement, original mesh. 68
4.8 Mean shape and variations of different levels of wavelet coefficients from the BU-3DFE database (neutral expression). Variation ranges are given in meters. . 70
4.9 Reconstruction results (rows 1 through 6) for range data of a subject wearing glasses (final row). 71
4.10 Reconstruction results (second row) for a face cropped from the CAESAR full body scan database. 72

5.1 The distance transform (solid line) of a sampled function (points). The distance from the envelope points is shown dashed. 78
5.2 A stereo pair from the KITTI dataset before (top) and after (bottom) radiometric equalization. Note the change in brightness of the road prior to equalization. 80
5.3 Epipolar geometry of spherical images. 83

5.4 An input image and the disparity map from our method for the first stereo pair from the KITTI dataset [2]. The color coding is shown in the bar on the right. . 90
5.5 Disparity estimates after each stage of our pipeline. From top to bottom: input image, disparity after multi-scale matching, disparity after fusion, disparity after filtering. Disparity range is the same as for Figure 5.4. 91
5.6 An image from "fountain-P11" high-resolution data set [3] and our disparity result. The close-up on the center of the fountain shows the disparity color-mapping rescaled for better visualization of the fine-scale detail. 92
5.7 A spherical image with RD-sampling and the resulting disparity map. 92
5.8 Top: A spherical image with RD-sampling and the resulting disparity map from our method. Bottom: single-scale matching (Section 5.2.2) and multi-scale matching (Section 5.2) (before post-processing). Disparity range is the same as Figure 5.7. 93

6.1 Top: an input RD-map spherical image and corresponding estimated disparity map; the disparity range and color scale is given on the right. Bottom: reference view from another panorama not used for reconstruction (left) and synthesized view from same viewpoint (right). 96
6.2 Stages in disparity estimation: pairwise matching (a), DSI merging (b), disparity map fusion (c). 99
6.3 Illustrations of an occlusion (left) and a free-space violation (right). Note that although the points are labeled with, *e.g.*, $\hat{d}_i$, they actually show the points obtained by scaling the pixel direction by, *e.g.*, $1/\hat{d}_i$. 101
6.4 Two views of the layout of capture positions and novel views for the Tabaret Hall dataset. See text for explanation. 109
6.5 Comparison of disparity maps generated by graph cut approach and our method (right). 110
6.6 Disparity after different stages of our algorithm. Left to right: pairwise matching, merging, fusion, filtering. 111

6.7 Close-up perspective views of disparity maps after different stages of our pipeline as shown in Figure 6.6. The same perspective view of the input image is shown on the right for context. Note how the pairwise disparity estimate is overall accurate, but blurs the disparity from the chairs on the background. This is reduced in both the merge and fusion steps. The filtering steps aligns disparity discontinuities with edge images and smooths over the disparity estimates that were zeroed-out by the fusion stage. 111
6.8 Comparison of synthesized novel views. Clockwise from top-left: reference view; point cloud from 1 disparity map computed with graph cuts; point cloud from 1 disparity map computed with our method; point cloud from 4 disparity maps computed with our method. 112
6.9 Comparison of synthesized novel views. Same ordering as Figure 6.8. This view has significant depth variation and discontinuities. Note how the discontinuities from the graph cut estimation do not correlate with actually depth discontinuities in the scene, where as our method does. 113
6.10 Comparison of synthesized novel views. Same ordering as Figure 6.8. This view has many slanting surfaces at different depths. Large gaps appear in the graph cut estimate, whereas our method incurs only small gaps which are corrected by using more disparity maps. 114
6.11 Comparison of synthesized novel views. Same ordering as Figure 6.8. Even from only one disparity map, our method produces only small gaps along object boundaries. With four disparity maps, the approximation of the reference view is very close. 114
6.12 Comparison of synthesized novel views. Same ordering as Figure 6.8. Aside from avoiding the obvious gaps from the discretely estimated disparity maps, our method better estimates the shape of fine-scale objects. Note especially how the chair legs appear straighter with our method. 115
6.13 Comparison of novel views for a different scene. Same ordering as Figure 6.8. Again, with our method, gaps are localized at depth discontinuities. 115

Chapter 1

Introduction

Novel view synthesis (NVS), or Image-based Rendering (IBR), encompasses a set of computer graphics and computer vision techniques which use as a basic source of information a set of captured images of a particular scene to show the scene from points of view other than those of the input images, possibly arbitrary viewpoints. The advantages of IBR, versus traditional geometry-based graphics techniques, are primarily two-fold. First is that the complexity of the scene does not affect the complexity of the rendering algorithm, which is dependent upon the resolution of the desired image or view to be rendered, and the resolution of the captured images or any primitives derived from them.

Second, an effective IBR technique allows a real environment to be displayed with much less manual effort than the traditional approach of creating a geometry and appearance model of the various objects that make up the scene. For practical scenes, creating these models by hand is very labour intensive. Whereas an IBR technique may place some restrictions on the viewpoints from which the scene can be viewed or may have less than complete coverage of the scene geometry, the process should be mostly automatic, requiring minimal user intervention.

This document proposes a system for real-time NVS (RT-NVS), which we define as generating a sequence of novel views at video frame rates, that uses panoramic images as its basic primitive, and derives a representation to allow the user to navigate freely within the captured environment. By "navigate freely" we mean that the user is not restricted to "jumping" from panorama to panorama, but can view the scene from viewpoints in between adjacent panoramas, or even from arbitrary viewpoints. Our system generates novel views in real-time by solving the underlying problem of multi-view stereo reconstruction from panoramic images. While our motivation and primary application are RT-NVS, our stereo reconstruction pipeline can be applied to the related problems of passive geometry sensing, robot navigation, and

even autonomous driving. The image-based representation and related rendering technique presented in this thesis are suitable for both small enclosed scenes and scenes with large open spaces.

NVS or IBR involve reconstructing the *plenoptic function* over its entire domain from a set of samples (images). The plenoptic function, denoted here by $\mathcal{F}$, represents the intensity of light as a function of seven parameters: position (x, y, z), direction (θ, ϕ), time t and wavelength λ. It is commonly reduced to five parameters by assuming a fixed time and mapping to a colour space instead of specifying individual wavelengths,

$$\mathcal{F} : \mathbb{R}^3 \times S^2 \longrightarrow \mathcal{C}$$

where S^2 denotes the unit sphere in three dimensions (the "two-sphere"), and $\mathcal{C}$ is the colour space, *e.g.*, RGB. For ease of notation, we will consider $\mathcal{F}(x, y, z, \theta, \phi)$ to be a 3D vector function with dimensions for red, green and blue. The plenoptic function was first defined by Adelson and Bergen [4]. Image-based rendering was first defined as follows by McMillan and Bishop [5]:

> Given a set of discrete samples (complete or incomplete) from the plenoptic function, the goal of image-based rendering is to generate a continuous representation of that function.

A panoramic image is a sampling of the plenoptic function in (nearly) all directions at a single point. Thus, by considering collectively panoramic images captured at several locations throughout an environment, we obtain a sampling of the plenoptic function that is (relatively) sparse in position and (relatively) dense in direction. The goal of this thesis then, is to derive a multi-resolution representation from these samples that gives continuous coverage of the plenoptic function. The multi-scale representations and algorithms presented here are suitable for panoramic images, which are parameterized on the 2-sphere, denoted by S^2.

1.1 Motivation of the Problem

The problem of novel view synthesis, or view interpolation or IBR, is primarily motivated by applications where the user wishes to navigate virtually within some real environment; imagine a tele-presence application where a remote user can navigate freely about the meeting space and view the proceedings from any angle, or watching a sporting event or concert from any angle, or a virtual tour of a museum or historical site, or a multi-player online shoot-out in the streets of your hometown. The use of image-based techniques for capturing and modeling

such environments is motivated by the ease of capture as compared to active imaging systems. While it is possible to use laser scanners to capture the geometry of a scene, and then texture map the geometry with images of the scene, this process is much more time consuming, labour intensive and expensive than using images alone. By using panoramic images we reduce the number of images required to fully capture a scene.

1.2 Problem Statement

We define the NVS problem as follows. Given a set of calibrated panoramic images I_1 to I_N of a static indoor or outdoor scene, compute the image V which would be obtained by placing a camera at a given new viewpoint that is not in the original set. The image V may be either a panoramic or perspective image, the task of generating either one differing only by the final projection used and the number of pixels required to achieve the same spatial resolution. The output image V should exhibit parallax as the novel viewpoint moves relative to the input image. We extend this to define the problem of RT-NVS: given a set of calibrated panoramas I_1 to I_N of a static indoor or outdoor scene, compute a sequence of views V_1 to V_M over a period of time not greater than $M\Delta t$ where Δt is an amount of time between frames that results in full-motion video (i.e. $1/30\ s$ or at least $1/15\ s$). We further constrain the sequence of viewpoints to lie along a continuous path or trajectory within the scene.

In practice, due to limited resources, we cannot simultaneously capture panoramic images at numerous locations, so the images are captured sequentially. Thus, we assume that the environment we are capturing remains static throughout the capture process. A more advanced version of the problem, in which we allow a dynamic scene during capture, would require us to deal not just with occlusions, but also with moving objects that break the multi-view geometry required to perform reconstruction, as well as points in one panorama that have no correspondence in another panorama because the object in question has moved in or out of view in the time between the two captures.

The amount of processing required to go from a calibrated set of panoramas to a sequence of novel views is too much to be performed in real-time. However, most of it can be performed in an offline pre-processing, or authoring, stage. In this stage, we generate an intermediate representation of the scene that can be used to generate novel views in real-time. In our proposed approach, we generate a set of depth or disparity maps, one for each panorama, which allow for efficient (real-time) reprojection of the pixels of the input panoramas into a novel perspective view.

We have therefore transformed the problem of NVS, or reconstructing the plenoptic func-

tion, into the problem of multi-view stereo reconstruction. Abstractly, this problem can be stated as taking a set of input images I_1 to I_N and generating a geometric representation G of the environment. Different methods in the literature generate different types of geometric representations [6]. While there are many choices for G, in this thesis we choose to reconstruct point clouds due to their simplicity and flexibility, and the fact that they can always be processed further into continuous representations. To tackle the problem of multi-view reconstruction, we must tackle the following sub-problems: camera calibration or structure-from-motion, stereo matching, and multi-view geometry fusion.

1.3 Summary of Existing Methods

This section presents a summary of the existing methods for solving the problem at hand, describes the state-of-the-art, and discusses why the methods proposed here are expected to improve upon the state-of-the-art. A more detail literature review is given in Chapter 2. The different kinds of NVS algorithms can be roughly categorized as follows: view interpolation, column or strip pasting, layer-based methods, image-based methods, and particle-based methods. We also discuss the overlapping category of panoramic methods. Since we are reconstructing dense point clouds and rendering them, one could argue that the system presented here falls into the category of particle-based methods. However, since our point clouds are directly computed from disparity maps, one could also argue the system presented here qualifies as a layer-based method. In fact, this thesis presents a system that is a hybrid of layer- and particle-based methods.

View interpolation dates back to Chen and Williams in 1993 [7]. View interpolation is a restricted form of view synthesis in which the novel viewpoint lies on the line between two captured viewpoints. View Morphing [8] computes perspective-correct warping between the input images and the novel viewpoint.

Let us use the term column-pasting methods to refer to methods that fuse vertical strips from densely captured images into the novel view, when the motion of the input images is entirely in the plane. Concentric Mosaics [9] also restrict the novel view to lie within a limited region, essentially a disc, and did not necessarily construct a fully perspective view.

In Plenoptic Stitching [10], panoramas are captured on intersecting smooth paths through the environment and loops created by these paths are detected. Novel views from within one of these loops are synthesized by sampling vertical (in the scene) strips from the panoramas on the boundary of the loop with the aid of a simple geometric proxy. Sea of Images [11] captures panoramic images very densely, and stores them in a multi-resolution hierarchy to

reduce the problem of NVS to one of data retrieval and working-set management. Both of these systems used catadioptric mirror-based panoramic cameras, and had drawbacks in terms of capture costs and in terms of the resolution and quality of the synthesized views.

More recently, StreetSlide [12] provides interactive navigation of street-view panoramic imagery. Normally, the user is considered to be in a "bubble" and views perspective projections of a single-viewpoint panorama, but the user can back-up until they pop out of the bubble to view a multi-perspective panorama generated by pasting columns from adjacent panoramas according to the directions of the columns in the novel view. This provides a wide-angle view of the street, but this approach assumes densely captured panoramas, imposes restrictions on the novel views that can be generated and does not generate a strictly perspective novel view.

If one performed multi-view or binocular stereo using the input images, that is finding points in the images that correspond to the same point in the scene, then one could use the resulting geometric information to generate novel views. By finding correspondences between points in the images, we can reconstruct the geometric layout of the scene by intersecting the rays passing through the corresponding pixels. Many NVS or IBR methods explicitly perform stereo to render new views. This is particularly the case for RT-NVS where the novel view must be generated at video rates and there is not time to do so from scratch, but one can leverage the power of graphics hardware to render texture-mapped geometric models in real-time, and this is the case for the system presented here. One class of NVS method that does so is that which we call layer-based techniques. Layer-based techniques have been very successful and widely employed in NVS and free-viewpoit video (FVV) applications. The approach presented here shares many similarities with layer-based methods, however it retains flexibility and simplicity by generating point clouds from multiple points of view instead of multiple layers relative to one viewpoint.

Shade et al. [13] proposed layered depth images (LDI), which allowed semi-occluded regions to become visible as the user changed the viewpoint. Zitnick et al. [14] used a layered representation similar to LDI for video view interpolation. They constructed a mesh on-the-fly from depth maps and render with images as texture using GPU. This approach continues to be a benchmark for new video-rate view synthesis methods.

One recent piece of work that has both planar and panoramic aspects is layered depth panoramas (LDP) [15], which extends LDI to have a panoramic field-of-view. Planar images are captured in an approximately cylindrical arc then used to construct panoramic layers of depth and color information. Like LDI, this allows semi-occluded regions to become visible as the novel viewpoint moves, and incorporates a wide field-of-view. However, this approach computes a single set of depth and color layers from a set of planar images, which naturally

limits the set of correct novel views that can be generated, and while the field of view in the examples shown was larger than a planar image, it did not cover the full sphere, thus making navigation of the scene less immersive.

Another type of NVS method that relies on stereo depth estimation is particle-based methods. Hornung and Kobbelt [16] use particle-based multi-view stereo to create view-dependent proxies for free-viewpoint resampling of a set of input images. Their approach allows interactive 3D user navigation, displaying novel views for significant changes in viewpoint from the original images. However, we anticipate our panoramic approach to give better coverage of the scene per input image acquired.

Ambient point clouds [17] provide smooth transitions between input views by visualizing the uncertainty in regions where depth cannot be reliably computed. Textured meshes and standard point clouds are used to render geometry that can be reconstructed, and the ambient point clouds are used to fill in the holes and uncertain regions. This reduces the visual artifacts of unreliable depth estimates, however, the novel viewpoint is restricted to lie near the line connecting the two camera centers.

Many NVS methods are very similar to stereo methods, with the main difference being that constraints are enforced on the resulting synthesized view rather than on the surface. We call these methods image-based methods. Fitzgibbon et al. [18] enumerate the minima in colour space of a photoconsistency function for each pixel in the novel view, and then impose a prior that patches of the novel view should be similar to patches extracted from the input images. To capture the rich statistics of the input images a dictionary of patches from the images was built, and the high-order graph restricted the optimization framework to localized search and made the approach unsuitable for RT-NVS. Pairwise dictionary priors [19] were developed to allow the use of pairwise Markov random fields (MRF) with an image dictionary prior making the optimization more effective and more efficient.

Another image-space approach to NVS constructs a joint conditional random field (CRF) on the pixels of the novel view, and estimates depth only as a means to compute potentials on the color values of the pixels of the novel view [20]. Intensity values are compared directly to those of the input images, as are the output of convolution with a set of steerable filters to enforce the condition that neighborhoods of pixels in the novel view resemble neighborhoods of pixels in the input images. However, it is not clear that this method can be implemented to run in real-time anytime soon.

In our case, we propose to perform stereo reconstruction from omnidirectional images, generating omnidirectional depth or disparity maps, and from there dense point clouds. Previous work on panoramic stereo dates back to Ishigura and Tsuji [21], followed by the work of Kang

and Szeliski [22] who performed feature-based omni-directional multi-baseline stereo. More recently, Micusik and Kosecka [23] reconstructed piece-wise planar models from panoramic images using super-pixel-based stereo. Shi [24] first proposed the notion of spherical or angular disparity, and performed simple, but real-time, spherical binocular stereo. Kim and Hilton [25, 26] use the same disparity formulation within a more complex framework for static environment modeling. We propose to formulate spherical disparity using two quantities to overcome some of the limitations of the existing formulation. Pagani et al. [27] reconstruct dense point-clouds from high-resolution spherical images using a spherical adaptation of Patch-based Multi-View Stereo (PMVS) [1], although they do not indicate the running time of their approach, and the novel reconstruction method here is faster than the original PMVS.

Naturally, when proposing a technique for NVS from panoramic images, we must consider both previous efforts tackling this specific problem and state-of-the-art NVS methods from planar images which may be adapted to panoramas. Previous efforts at immersive NVS systems from panoramic images include Plenoptic Stitching [10], Sea of Images [11] and Street-Slide [12]. These methods can all be considered pasting or stitching methods, and require a densely captured set of panoramic images.

1.4 Overview of the Proposed Approach

Point cloud and layer-based NVS algorithms have shown to be successful in the literature, especially for interactive and real-time view generation. They also have the benefit of being as accurate and precise as the input images allow. We therefore propose algorithms for generating point-cloud geometric proxies and rendering algorithms for RT-NVS. We propose to take as input a set of spherical panoramas, to maximize immersiveness, and to construct a dense point cloud from them. Each point consists of a 3D position and an RGB color. The first major technical hurdle is estimating a disparity map for each panorama from which to generate a point cloud. The second major technical hurdle is to render a complete view using these point clouds, since parts of the scene may not be visible in any of the input images, and for certain viewpoints the point cloud may not be dense enough to generate a view without holes. This involves some form of in-painting, or filling in the missing portions of the novel view.

Along the way we develop some image processing tools for spherical images. In Chapter 3 we present a GPU implementation of a fast spherical harmonic (SH) transform algorithm. The SH transform is the spherical analog to the Fourier transform for planar signals, and thus results in a frequency domain representation of a spherical image. Like Fourier coefficients of planar images, SH coefficients can be used for fast spherical convolution algorithms, which we also

implement on the GPU. This allows us to efficiently apply a series of filters to a panoramic image. Using this technique, we can efficiently generate a filterbank-based overcomplete wavelet representation of a spherical image. While this representation has uses in spherical image processing, such as multi-scale edge detection, it has also been shown to be useful for geometry processing, specifically analyzing the cortical folding of the brain from MRI data [28].

In Chapter 4, we show that noisy, incomplete point clouds can be efficiently and robustly fit using a statistical shape prior, in the case where we know the type of surface we are trying to reconstruct, *e.g.*, , a human face. This first involves building a statistical prior from a set of parameterized training examples of the shape in question. This gives a set of shape parameters, which can then be tuned to fit a noisy point cloud. Robustness comes from a prior based on the statistics computed during the training phase. We show that a wavelet basis is an effective basis in which to perform the learning, due to its decorrelating and localizing properties.

To compute a depth or disparity map for a given panorama (the reference image), we begin by computing a disparity space image (DSI) [29], which consists of a matching cost or dissimilarity over the disparity range for each pixel in the disparity map. In Section 5.3 we present a general, continuous formulation of the disparity space of spherical images, which is easily adapted, abstracts the underlying sampling of the sphere, allows efficient reprojection of directions in one image to directions in another, allows efficient occlusion testing, and allows efficient disparity map fusion. We use this formulation to efficiently sample spherical DSI.

In Chapters 5 and 6, we begin by robustly estimating spherical disparity maps using multi-scale technique, followed by a fusion stage that combines multiple disparity maps in a way that removes outliers and encourages piecewise-smooth disparity estimates. Following this, we export a colored point-cloud from each resulting disparity map.

1.5 Summary of Results

Chapter 3 presents a GPU method for spherical harmonic transforms and spherical convolution as used for spherical image processing. A GPU implementation in CUDA [30] of the semi-naive method of Healy et al. [31] demonstrates significant speed-up over serial implementations of more complex algorithms, while a GPU implementation of the direct method requires little memory and can be applied to large transform sizes. We demonstrate how our transform can be used for filterbank-based overcomplete spherical wavelet decomposition of panoramic images.

Chapter 4 presents results for wavelet-model-based stereo reconstruction of human faces, although the approach is fully generalizable to any shape that is topologically equivalent to the

sphere. A generalized B-spline spherical wavelet [32] forms the basis of a multi-resolution statistical model of the shape of the human face, which is used as a prior and modified to maximize the posterior distribution within a Bayesion framework. This model is computed from a set of registered laser scans of human faces in a learning stage, and then fit to noisy stereo data by sampling the learned wavelet coefficient distributions. The model is then refined using anisotropic second-order smoothing and photoconsistency energies. The wavelet transform, along with most of the framework, is implemented on the GPU to accelerate model optimization. The statistical prior offers robustness to outliers in the initial stereo data, and gives the reconstructed shape in a corresponded model, in terms of the same set of model parameters, thus enabling additional statistics to be computed more easily. This chapter shows how we can use statistical geometric analysis to regularize stereo matching with robustness to noise, and outliers such as occlusions and specularities.

Chapter 5 presents a multi-scale method for binocular stereo reconstruction from pairs of planar and spherical images that is efficient in terms of both time and space requirements while avoiding greedy decisions at coarse levels that limit possible solutions at finer scales. A novel disparity formulation for calibrated spherical images is also presented, which allows the planar algorithm to be extended in an efficient and geometrically correct way, and allows any underlying sampling of the spherical input images to be used. We demonstrate the accuracy of the proposed approach on a standard planar stereo benchmark and provide visually pleasing results for the spherical case.

Chapter 6 presents a complete and viable system for planar and spherical multi-view stereo reconstruction of dense point clouds, with the application of real-time novel view synthesis. Multi-view visibility-based fusion is adapted to spherical images and disparity, and the approach is extended to refine the disparity maps in the process. This chapter further gives an accurate calibration procedure for spherical images, using spherical reprojection error.

Chapter 2

Background and Literature Review

We begin by reviewing existing work that relates to the methods proposed here. This will touch upon many different areas of research and we will attempt to group them appropriately into the sections below.

We begin our review with spherical parameterizations, techniques for representing functions on the sphere, in Section 2.2. This is clearly important for omnidirectional images, as we want to be able to store and process these images efficiently. This starts with a review of spherical harmonics (Section 2.2.1), a frequency domain for spherical functions. In Chapter 3, we present an efficient GPU implementation of spherical harmonic transforms and frequency domain convolutions of spherical functions (Section 2.2.2). We then move on to spherical wavelet representations in Section 2.2.3, which extend the standard Euclidean domain wavelet representations to either the sphere or in some cases to manifolds sampled with subdivision connectivity. We use a biorthogonal spherical wavelet in Chapter 4 as a multi-scale surface representation and build a multi-scale statistical model of the surface of the human face. We in turn use this for model-based stereo and noisy point cloud parameterization. In Chapter 6 we use a redundant spherical wavelet transform to filter spherical disparity maps. We then look at spherical parameterizations: ways to sample the sphere at a finite number of points. This includes latitude-longitude or equiangular sampling, which have efficient spherical harmonic transforms as discussed in Chapter 3, cube-maps, octohedral mappings and the Rhombic-Dodecahedron mapping among others. In Chapters 5 and 6 we use the Rhombic-Dodecahedron mapping for spherical binocular and multi-view stereo, and novel view synthesis. This mapping has high uniformity and low distortion and is relatively efficient to sample and compute the direction of a given pixel.

We then review in Section 2.3 how wavelets and general multi-scale schemes have been

used for stereo correspondence. Typically this involves applying a wavelet transform to the input images and matching the resulting coefficients instead of the original pixel values, or constructing an image pyramid and matching the coarsest levels first and using that estimate constrain subsequent searches at finer levels, effectively searching a greater disparity range at a low computational cost. In Chapter 4 we use a spherical wavelet to represent the surface to be reconstructed, and build a statistical model to robustly reconstruct a particular class of surface (we demonstrate for the human face). In Chapter 5 we present a novel application of distance transforms to multi-scale image matching, which subsamples the disparity grid, but uses the full resolution input images at all scales. This method avoids making greedy decisions at coarse scales, while efficiently constraining adjacent scales to agree with each other in terms of disparity estimates.

In Section 2.4 we review model-based stereo, where a statistical prior is used to constrain stereo matching, and stereo reconstruction of human faces and passive facial performance capture. In Chapter 4, we present a statistical wavelet model-based method for object type-specific stereo reconstruct and point cloud parameterization. We demonstrate this method for human faces with our statistical model learned from a database of registered laser scans.

Section 2.5 provides a brief overview of recent multi-view stereo methods that are either top-performing methods or are related to the method presented in Chapter 6 for multi-view omnidirectional stereo. Section 2.6 reviews formulations and algorithms for spherical disparity and spherical stereopsis. In Chapter 5 we present a novel spherical disparity formulation, and demonstrate an efficient multi-scale method for high-resolution planar and spherical images.

We conclude our review in Section 2.7 with a discussion of existing methods for NVS or IBR. In Chapter 6 we show how multi-view spherical stereo can be used to perform NVS in real-time.

Before we proceed with the review, however, let us introduce some basic notation to be used in the remainder of this thesis.

2.1 Notation

A panoramic image can be represented as function defined on the sphere, S^2, and hence as function with parameters θ and ϕ, which we refer to as spherical coordinates (θ, ϕ) where $\theta \in [0, \pi]$ is the colatitudinal angle and $\phi \in [0, 2\pi]$ is the longitudinal angle; abusing terminology slightly we will refer to θ as latitude, even though $\theta = 0$ is the north pole, or positive z-axis, and not the equator ($\theta = \pi/2$). We will at times use the notation $\omega = (\theta, \phi)$ as shorthand, and at times we will allow ω to denote a vector $(x\ y\ z)$ or angular (θ, ϕ) representation of direction

interchangeably.

Note that while any panoramic image can be thought of as a function on the sphere, it may be sampled on any surface that is topologically equivalent to a sphere, *e.g.*, a cube, an octohedron, or any convex polyhedron. In the continuous domain all are equivalent. However, in the discrete (i.e. sampled) domain they are quite different. Each sample or pixel defines a ray from the center of the sphere intersecting the sphere at the pixel's location. We explore some of the differences, advantages and disadvantages of these different samplings of the sphere in Section 2.2.

We will at times need both subscripts and superscripts, *e.g.*, $g_{lm}^{(n)}$ to denote the spherical harmonic coefficient indexed by l and m of the level-n wavelet component $g^{(n)}(\theta, \phi)$ of a function $f(\theta, \phi)$. When using superscripts, we will use parentheses as above to distinguish the superscript from an exponent.

In general we will denote vectors with lower-case bold letters and treat them as column vectors or single-column matrices, for example $\mathbf{x} = [x\ y\ z]^T \in \mathbb{R}^3$, with the exception of using ω to denote a directional vector. Other types of matrices we will denote with upper-case letters such as A, and their components by the lower-case letters with subscripts, such as a_{11}, in row-major order. Sets and spaces we will denote with caligraphic letters, such as $\mathcal{S}$. Functions may be denoted with either upper or lower-case letters, and either bold or not depending on whether vector-valued or not.

2.2 Spherical Representations

There are many ways to represent a function on the sphere. These can be roughly categorized into sampled or spatial domain representations, frequency domain representations, or spherical wavelet representations, which are a combination of the first two. Sampled representations must address a trade-off between the uniformity of sampling density over the surface of the sphere and the complexity of the data structure for storing those samples. For example, sampling in spherical coordinates (θ, ϕ), also referred to as latitude and longitude, results in highly non-uniform sampling. However, the sampling is very simple, is mostly spatially contiguous, and allows for fast spherical harmonic transforms (Section 2.2.1).

2.2.1 Spherical Harmonics

The SH transform decomposes a spherical function into its frequency components using an infinite set of basis functions. Spherical harmonics appear in a variety of applications in fields

such as geophysics [33], medical imaging [28, 34], graphics [35, 36, 37, 38] and computer vision [39, 40], to name a few.

We first define the parameterization of our domain, the unit sphere in $\mathbb{R}^3$, denoted by S^2. We parameterize S^2 by the colatitude $\theta \in [0, \pi]$, or the angle from the positive z-axis, and the longitude $\phi \in [0, 2\pi)$, the angle counterclockwise about the positive z-axis from the positive x-axis. This is a convenient and popular choice for spherical coordinates [41].

Given a real-valued spherical function $f(\theta, \phi)$ $f : S^2 \to \mathbb{R}$, we can express a band-limited approximation to this function in terms of the SH basis functions as

$$\hat{f}(\theta, \phi) = \sum_{l=0}^{N-1} \sum_{m=-l}^{l} f_{lm} Y_{lm}(\theta, \phi) \tag{2.1}$$

where f_{lm} are the SH coefficients expressing f in the frequency domain, and Y_{lm} is the SH basis function of degree l and order m. The function $\hat{f}$ is the approximation of f with bandwidth N or degree $N - 1$. The SH coefficients are computed by taking the scaled inner product of f with the corresponding basis function

$$f_{lm} = \frac{1}{4\pi} \langle f, Y_{lm} \rangle$$

which can be expressed as the integral

$$f_{lm} = \frac{1}{4\pi} \int_{\theta=0}^{\pi} \int_{\phi=0}^{2\pi} f(\theta, \phi) Y_{lm}^*(\theta, \phi) \sin\theta d\theta d\phi \tag{2.2}$$

where the factor $\frac{1}{4\pi}$ is the result of the chosen normalization, * denotes complex conjugation, and $\sin\theta$ accounts for the reduced solid angle of $d\phi$ towards the poles.

The basis functions themselves are defined in terms of the normalized associated Legendre functions $\bar{P}_{lm}$ of degree l and order m

$$Y_{lm}(\theta, \phi) = \bar{P}_{lm}(\cos\theta) e^{im\phi} \tag{2.3}$$

where we use the relation $Y_{l,-m} = (-1)^m Y_{lm}^*$ [42] to compute all spherical harmonics in terms of the associated Legendre functions $\bar{P}_{lm}(x)$ with $m \geq 0$, which satisfy the three term recurrence [33] (adapted from Gradshteyn and Ryzhik [43] p.1004)

$$\begin{gathered} \bar{P}_{lm}(x) = \alpha_{lm} x \bar{P}_{l-1,m}(x) - \beta_{lm} \bar{P}_{l-2,m}(x),\ 0 \leq m \leq l-2 \\ \bar{P}_{l,l-1}(x) = \sqrt{2l+1} x \bar{P}_{l-1,l-1}(x),\ l \geq 1 \\ \bar{P}_{ll}(x) = \sqrt{\tfrac{2l+1}{2l}} y \bar{P}_{l-1,l-1}(x),\ l \geq 2 \\ \bar{P}_{0,0}(x) = 1,\ \bar{P}_{1,1}(x) = \sqrt{3} y \end{gathered} \tag{2.4}$$

where $x = \cos(\theta)$, $y = \sin(\theta)$ and the constants

$$\alpha_{lm} = \sqrt{\frac{(2l-1)(2l+1)}{(l-m)(l+m)}}$$

and

$$\beta_{lm} = \sqrt{\frac{(2l+1)(l+m-1)(l-m-1)}{(2l-3)(l-m)(l+m)}}$$

complete the recurrence. Note that there are many different recurrences that can be used to compute the associated Legendre functions. We use a slightly different recurrence in Chapter 3 for the so-called semi-naive method of Healy et al. [31]; the basic form is the same, but the starting condition $\bar{P}_{mm}(x)$ is computed via a closed-form expression, and the recurrence coefficients α and β are slightly different.

We also consider the real SH basis [35, 44],

$$y_{lm}(\theta,\phi) = \begin{cases} \sqrt{2}\Re(Y_{lm}(\theta,\phi)) = \sqrt{2}\bar{P}_{l|m|}(\cos\theta)\cos(m\phi) & \text{if } m > 0 \\ \sqrt{2}\Im(Y_{lm}(\theta,\phi)) = \sqrt{2}\bar{P}_{l|m|}(\cos\theta)\sin(|m|\phi) & \text{if } m < 0 \\ Y_{lm}(\theta,\phi) = \bar{P}_{l|m|}(\cos\theta) & \text{if } m = 0 \end{cases} \tag{2.5}$$

which is applicable because our input images are real-valued. These functions are called *tesseral harmonics* for $|m| < l$ and *sectoral harmonics* for $|m| = l$ [43]. Although we use the complex form in most of our discussion, analogous discussion applies to the real form. We implement both the real and complex forms on the GPU and compare them to existing implementations in Section 3.3.

Let us now consider discrete versions of (2.2). Let our discrete samples be along the equiangular grid $\theta_k = (2k+1)\pi/4N$ and $\phi_j = \pi j/N$. The discrete SH transform is then

$$f_{lm} = \frac{1}{4\pi}\sum_{k=0}^{2N-1}\sum_{j=0}^{2N-1} a_k^N f(\theta_k,\phi_j)e^{-im\phi_j}\bar{P}_{l|m|}(\cos\theta_k) \tag{2.6}$$

where the weights $a_k^N = 2\pi^2\sin\theta_k/4N^2$ are finite approximations to the differentials in (2.2).

Evaluation of (2.6) takes $O(N^2)$ operations, assuming we are using $O(N^2)$ storage for a table or memo of the previous values $\bar{P}_{l-1,m}(\cos\theta_k)$ and $\bar{P}_{l-2,m}(\cos\theta_k)$ for all k and all m such that $0 \le k < 2N$ and $0 \le m < N$. Since there are N^2 coefficients direct computation of the transform takes $O(N^4)$ operations. Due to their occurrence in such a wide variety of fields, efficient evaluation of the spherical harmonic basis functions and efficient transforms have been studied in depth. It is well known that one can use a separation of variables to perform the computation in $O(N^3)$ operations, as described in Section 3.2. In Section 3.1, we review existing asymptotically fast algorithms.

2.2.2 Spherical Convolution

Convolution of two spherical functions can be performed in the frequency domain. Figures 2.1, 2.2 and 2.3 illustrate this. A cubic checkerboard pattern was transformed to the spherical harmonic domain, and the resulting approximation was resampled on a sphere (Figure 2.1). Figure 2.2 shows a 2D Laplacian of Gaussian (LoG) filter stereographically projected onto the sphere. Figure 2.3 shows the result of convolving these two functions in the frequency domain.

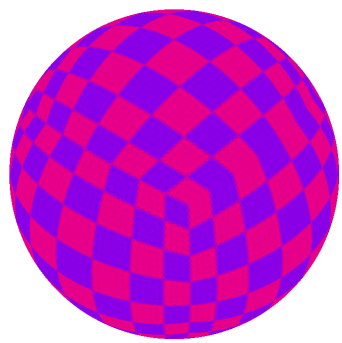

Figure 2.1: A cubic checkerboard pattern resampled on the sphere via the spherical harmonic domain (maximum degree 179).

Figure 2.2: A Laplacian of Gaussian filter stereographically projected on the sphere via the spherical harmonic domain (maximum degree 179).

Stereographic dilation of functions on the sphere expands or contracts them about the north pole, or positive z-axis. The spherical function is projected to the tangent plane at the north pole, and dilated in the plane, and reprojected back onto the sphere. The method of projection is

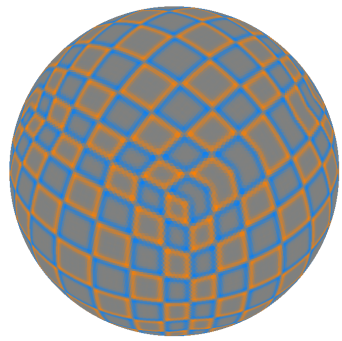

Figure 2.3: The spherical convolution of the image in Figure 2.1 with filter in Figure 2.2 in the spherical harmonic domain.

known as stereographic projection. Stereographic projection and dilation are illustrated in Figure 2.4. In this figure, we have the direction (θ, ϕ) projecting to the point $(r\cos\phi, r\sin\phi, 1)$, and vice versa, where $r = 2\tan(\theta/2)$. When dilating a spherical function, the direction (θ, ϕ) gets mapped to (θ', ϕ) where

$$\theta' = 2\arctan\left(\frac{1}{a}\tan\frac{\theta}{2}\right)$$

and a is the dilation factor. Thus if we denote by D_a the dilation operator by factor a, then the application of this operator to a function $f(\theta, \phi)$ becomes

$$[D_a f](\theta, \phi) = \frac{1 + \tan^2(\theta/2)}{1 + (1/a\tan(\theta/2))^2}\frac{1}{a}f\left(2\arctan\left(\frac{1}{a}\tan\frac{\theta}{2}\right), \phi\right) \tag{2.7}$$

where the leading factor is a normalization constant to preserve the inner product of two functions under stereographic dilation [45].

Spherical convolution shares some properties of its abelian counterpart, but not all. Spatial convolution on the sphere is equivalent to point modulation in the frequency domain, but it is not commutative. In general, convolution of two spherical functions results in a function that is defined on the rotation group $SO(3)$; only if one of the functions is radially symmetric about the north pole does the convolution result in another spherical function.

Because spherical convolution is not commutative, we must designate one function the input, and the other the filter. Let us compute the convolution of an input function $f(\theta, \phi)$ with the filter $h(\theta, \phi)$, which we restrict to be radially symmetric. Let their convolution be denoted $g(\theta, \phi) = f(\theta, \phi) * h(\theta, \phi)$. We first perform the SH transform of both the input and the filter

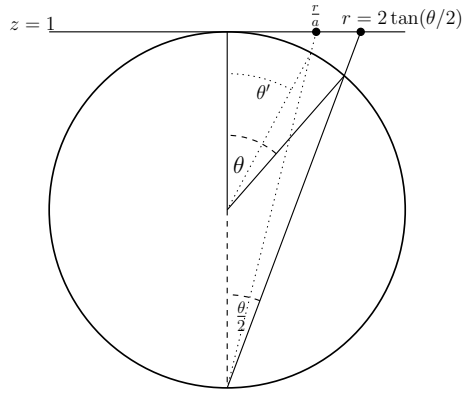

Figure 2.4: Diagram of stereographic projection and dilation.

to get coefficients f_{lm} and h_{lm}. The coefficients g_{lm} are given by

$$g_{lm} = \sqrt{\frac{4\pi}{2l+1}} h^*_{l0} f_{lm}. \tag{2.8}$$

2.2.3 Spherical Wavelets

Wavelets are among the most widely used mathematical tools in computer science and electrical engineering. Wavelets were initially defined for regularly sampled functions, *e.g.*, on a 2D grid. Spherical wavelets were introduced by Schroder and Sweldens [46] and used for surface representation [46, 45, 28] and segmentation [34], and BRDF representation [46]. Spherical wavelets are defined on a manifold with some kind of subdivision hierarchy. For example, Schroder and Sweldens defined a triangular subdivision hierarchy in which vertices are added on the midpoints of existing (spherical) edges.

A wavelet transform consists of decomposing a function by projecting it onto a set of basis functions, called *scaling* and *wavelet* functions. These basis functions are typically defined as shifted and dilated version of a particular function, called the *mother wavelet*. In the case of spherical wavelets, these functions are all defined on the sphere, or a surface that is topologically equivalent to a sphere.

Many different types of bases have been used for spherical wavelet decompositions: Haar bases [46, 47], vertex bases [46], a generalized B-spline basis [32], and overcomplete bases [45, 28].

In Chapter 3 we use a GPU implementation of the spherical harmonic transform to efficiently convolve spherical images with a filterbank of spherical filters, generating an overcomplete spherical wavelet representation [45].

In Chapter 4 we use generalized B-spline spherical wavelets [32] as a multi-resolution shape representation for model-based stereo of human faces. This wavelet transform uses quadrilateral Catmull-Clark subdivision connectivity for upsampling and downsampling, and was previously used for segmenting parts of the brain from 3D data [34].

In Chapter 6 we adapt the edge-avoiding algorithm à-trous to spherical wavelets using the RD-map sampling (see Section 2.2.4) for spherical disparity map filtering. Edge-avoiding wavelets were first introduced by Fattal [48]. In this type of transform the basis functions are data-dependent in that they depend on the similarity between pixels. That is, if we think of each level of a wavelet transform as applying a local filter to the pixels at that level, then the filter weights depend on the difference between pixel values. The algorithm á-trous was first introduced by Dutilleux [49] and Holschneider et al. [50] and provides an efficient algorithm for overcomplete or redundant wavelet transforms. It works by maintaining a fixed number of non-zero filter weights at all levels, but adding an increasing step size between these non-zero filter weights at coarser levels, giving the filter a larger extent at the same computational cost. Dammertz et al. [51] combined edge-avoiding and à-trous basis functions for global illumination filtering, and this is the transform that we extend to the spherical domain by adapting the filter windows to an efficient spherical sampling. We further introduce a static weight term in the transform to account for robust confidence information computer during stereo matching and disparity map fusion (see Chapter 6). These types of edge-avoiding wavelets are useful for denoising and contrast enhancement (see Hanika et al. [52] for an optimized version) because the edge-sensitivity makes the fine-scale wavelet (detail) coefficient behave even more like i.i.d zero-mean Gaussian noise.

2.2.4 Spherical Parametrizations

So far we have discussed the spherical coordinate, or latitude-longitude grid, parametrization of the sphere and mesh-based parametrizations. There are a number of other ways to parameterize functions on the sphere, especially into one or more 2D grids.

Cube Map

The cube map is composed of six square 2D planar images, each image having a vertical and horizontal field of view of $\pi/2$ radians and a focal length of half the side length of the cube. The six images naturally make up the six faces of a cube. This format has the good property that each face is a perspective projection planar image and it is widely supported in graphics

hardware, however it incurs considerable distortion near the edges and corners of the cube, and it has highly non-uniform sampling density.

Octohedral and Periodic Quincuncial Mappings

Spherical parametrizations can be derived from an octohedron by mapping the triangular faces of one pyramid to the northern hemisphere and the triangular faces of another pyramid to the southern hemisphere [53]. This also know as a periodic quincuncial mapping [54]. This has the advantage that there is only one singularity, and discontinuities can be relegated to a single hemisphere if that is desirable.

Rhombic Dodecahedron Map

By combining the vertices of a cube and an octohedron, one can obtain a 14 vertex, 12 face polyhedron called a rhombic dodecahedron, because each face is an identical rhombus. By projecting the edges of the rhombi onto the sphere, one gets spherical rhombi which share boundaries defined by six great circles. This is the basis for the rhombic dodecahedron map (RD map) [55]. Samples are located on skew great circles interpolated between the bounding great circles. This makes for a sampling pattern with very low discrepancy (a measure of uniformity of the sample distribution), and with low area and shape distortion. This means that block-based encoding methods (*e.g.*, wavelet and DCT-based) that were originally designed for planar images and video can be reliably applied to this spherical mapping. This mapping also supports highly efficient data indexing over the sphere. These properties are also desirable for our purposes of sampling panoramic images to compute matching costs, and to store panoramic geometry. In Section 5.3 we describe a continous formulation of the disparity space of two spherical images that abstracts the underlying parameterization, thus it can be applied to omnidirectional images parameterized by spherical coordinates (lattitude-longitude), or a more uniform sampling such as the RD map.

Other Parametrizations

In Chapter 4 we use stereographic projection to parameterize the surface of the human face, which we treat as a vector-valued function on a subset of the sphere. While stereographic projection maps the whole sphere to a plane, some parameterizations used in global illumination algorithms map one or more hemispheres to the plane [56]. The projected disk parameterization projects straight down into a disc with the pole at the center. This has the same non-uniformity problems as spherical coordinates, so a variant projects first to a paraboloid, then

down to the disk to distribute the elevation angle more evenly. The concentric map transforms the projected disk into a unit square to make sampling easier.

2.3 Wavelets and Multi-Scale Schemes in Stereo Matching

There are two main ways to use wavelets for stereo matching. The most common way is to represent the input images with a wavelet decomposition and to match the wavelet coefficients rather than the pixel intensities. The second way is to model the surface to be reconstructed using wavelet coefficients and impose a prior or regularization on those coefficients rather than on the surface itself. We first discuss the former then the latter.

The motivation for performing matching, in particular correlation-based matching, between wavelet coefficients of a transformed image pair, as opposed to directly between pixel intensities, comes from the properties of the wavelet transform. The obvious one is the fact that the wavelet transform provides a formalized and highly-structured framework for performing multi-resolution matching. The performance-based motivation for using a wavelet hierarchy as opposed to, *e.g.*, Gaussian pyramids, is due to the decorrelating and localizing properties of the wavelet transform. The wavelet transform decorrelates data, meaning that wavelet coefficients are less correlated, i.e. the depend less on one another, than do the original pixel intensities or the downsampled pixels in a Gaussian pyramid. Furthermore, wavelet coefficients are highly localized in both frequency and space, having very compact support. From these properties, one can infer that local correlation-based matching methods will be more effective when applied to wavelet coefficients than to ad-hoc multi-resolution schemes.

Mallat [57] used the zero-crossings of the wavelet transform to obtain correspondence between signal and image pairs. Simoncelli et al. [58] examined the performance of shiftable multi-scale transforms for stereo matching of synthetic images among other applications. By "shiftable" it is meant that applying the transform to a shifted input image will result in an output that is shifted by the same amount, versus applying the transform to an unshifted image. This is also referred to as "shift-invariance", and is an important property for a wavelet or other multi-scale transform to have when the goal is multi-scale stereo matching of the coefficients. If we consider the simple case of a fronto-parallel planar surface with perfectly Lambertian reflectance properties that has a constant disparity in a stereo image pair, the two images of the surface are shifted versions of one another. A wavelet or multi-scale transform is applied to the images so that a coarse estimate of the disparity can be found by matching the coarsest scale coefficients first, and this disparity estimate is refined by matching coefficients at subsequent finer scales. If the transform is not shift-invariant, then the coefficients at any

given subband will not be shifted versions of one another because energy will have transferred between adjacent coefficients due to aliasing, and any disparity estimation based on matching these coefficients will produce erroneous results.

Hsieh et al. [59] used a wavelet transform for multi-scale edge-point detection and applied it to image-registration. Edge points are identified as local maxima of an edge response that is computed as the product over a number of scales of the magnitude of the wavelet coefficients. Their appoach uses a line-fitting method to determine the direction of the edge at all edge points, and computes an "angle histogram" expressing the distrubtion of edge directions within each image. The images are initially aligned by finding the peaks in the distributions of edge directions of the two images computing the rotation to align them. Subsequently, possible pairwise feature point matches are pruned prior to performing correlation-based matching on the original images, not the wavelet coefficients.

Magarey and Dick [60] used complex wavelets to perform multi-resolution stereo image matching. Their approach decomposes each image into six sub-images at each scale, and they define a matching criterion at each scale to be the sum over the six sub-images of the squared difference in the wavelet coefficients. At each level, this matching term is mixed with a smoothing term to give an energy which is minimized locally. The results of one level are used to initialize the next level in a typical coarse-to-fine manner.

Shim [61] developed a wavelet-based method for stereo vision. This method performs intra-scale area-based matching between wavelet coefficients of the two stereo images, and performs inter-scale back-tracking to ensure consistency between scales.

Shi et al. [62] applied sum-of-squared-differences (SSD) matching to the results of a shift-invariant wavelet transform. They computed the SSD between coefficients at all scales of the wavelet transform and assign to each pixel the disparity at the scale with the minimum SSD score.

Caspary and Zeevi [63] borrow from approximation theory to develop a wavelet-based multi-resolution stereo algorithm. They use wavelet pyramids of the input image as well as an iterative algorithm based on a wavelet representation of the horizontal and vertical disparity fields. Once again, the algorithm begins at the coarsest level and uses the results to initialize the problem at the next finer level.

Liu et al. [64] based an approach to micro stereo matching on the wavelet transform and the projective invariance of non-uniform rational b-splines (NURBS). A wavelet transform is used for denoising and multi-scale edge extraction, and NURBS are subsequently fitted to the extracted edges. The NURBS curves extracted from the left image are then projected into the right image and the problem of correspondence estimation becomes the problem of computing

the difference between the two sets of curves (those projected from the left image and those extracted from the right image).

Bhatti and Nahavandi [65] perform stereo matching using wavelet and multi-wavelet analysis. A multi-wavelet transform generates multiple scaling and wavelet coefficients per data point, as opposed to one each for scalar wavelets. At the coarsest level, they combine correlation-based matching with symbolic tagging and geometric refinement to obtain reliable correspondences to pass on to the finer-level disparity estimation stages. At finer resolutions they perform an iterative local search combined with left-right-consistency checking. They achieve convincing results with this very involved approach.

Li et al. [66] used a two dimensional monogenic wavelet transform to develop a robust stereo image matching approach. They define a cost function that measures difference in amplitude, orientation and phase of the wavelet coefficients, and that also incorporates a smoothness and discontinuity term. They show encouraging results on two examples.

The above methods apply local correlation-window-based matching to wavelet and multi-wavelet coefficients rather than pixel values, or use the wavelet coefficients to extract edge or other feature information to match. In Chapter 4 we pursue a different approach, representing the surface to be as a vector of spherical wavelet coefficients, and using a statistical model to regularize them. In Chapter 5, we pursue yet another type of multi-scale matching approach, and leave the input images unchanged except for an optional radiometric equalization. We instead reduce the number of pixels at coarser scales for which matching is performed, which allows matching information to be aggregated over larger scales, and enforce consistency between scales with a novel application of distance transforms.

The properties of wavelet transforms also lend themselves to techniques that represent the surface to be reconstructed with a wavelet decomposition and then regularize the wavelet representation instead of the surface directly. The independence properties of the wavelet coefficients justify regularizing them independently of one another, which can greatly simplify the optimization scheme, as explained in Chapter 4.

Miled et al. [67] used a wavelet domain representation of the disparity map to regularize stereo reconstruction with an edge-preserving smoothing prior. In Chapter 4, by contrast, we explore the use of a wavelet representation of a surface to develop a statistical shape prior for stereo reconstruction of class-specific objects. This type of approach, in which the wavelet domain is used to model a particular type of surface, is a special case of so-called model-based stereo.

More generally, multi-scale image matching has been widely used for both stereo and motion estimation (optical flow). Some coarse-to-fine matching methods are essentially two-stage

methods [68, 69]. First, a coarse model is estimated, based on *e.g.*, segmented shape [68] or reliably-matched feature or support points [69]. Second, this coarse model is used to restrict a per-pixel disparity or depth search. Geiger et al. [69] proposed a generative disparity model by triangulating the matched support points and using the resulting planar disparity interpolation to define a probability distribution over the possible disparity values. Disparity values are inferred for the remaining pixels by maximizing this posterior distribution. Other methods use image pyramids [70, 71, 72] and iteratively refine disparity in a coarse-to-fine way, with the search range for the fine-scale disparity values being a narrow bracket around the disparity estimate from the previous coarser scale. Hirschmueller [71] uses mutual information between two images as a pixel-wise matching criterion. One image is first warped using the current disparity estimate, and the joint probability distribution and joint entropy of corresponding pixels is estimated then subtracted from the entropy of the individual images. The process is initialized at the coarsest level with a random disparity map, which is iteratively updated to increase mutual information at each pixel for a fixed number of iterations. Coarse disparity estimates are subsequently inherited by the finer levels and refined to increase mutual information with a restricted disparity range for each pixel. Mutual information as a matching cost is combined with a semi-global optimization, which involves 1D aggregation along 16 directions emanating from each pixel. This is less computationally expensive than global approaches such as belief propagation [73] or graph cut [74], but is much more accurate than local correlation-window methods. Other multi-scale matching approaches rather reduce the number of pixels at which costs are computed, not the resolution of the input images. This not only reduces the cost of the coarse estimates, but allows any type of aggregation or message passing algorithm to combine information over greater distances in the image. An example is hierarchical belief propagation (HBP) [75], where dyadic downsampling of the disparity grid allows message passing iterations to propagate information over larger portions of the image. This better preserves small differences in disparity between nearby pixels and the full resolution of disparity values. The approach presented in Chapter 5 is most similar to this type of multi-scale method. However, while HBP directly inherits message values from coarse scales to initialize fine scale iteration and proceeds without further reference to the coarse-scale information, our method is non-iterative and explicitly enforces consistent disparity estimates at consecutive scales.

2.4 Model-based Stereo

When reconstructing a surface using stereo methods, it can be greatly helpful to have a model for the surface to be reconstructed. That is, if we can say that the surface belongs to a well-

defined subset of possible surfaces, rather than any general surface, we can simplify the problem by parameterizing this subset in a useful way. This model, or parameterization, should reduce the number of variables to optimized, or simplify the procedure by which the model parameters can be estimated, or strongly regularize the possible surfaces that can be reconstructed. Such an approach is called model-based stereo.

Having a model also makes it possible to compute statistics on that model, if a training set of example surfaces conforming to that model are available. Computing statistics on raw depth or disparity maps is not helpful, because in the general case the same pixel in two different disparity maps are not related in any way, even if they are of the same type of object (i.e. the relative pose of the objects with respect to the stereo rig may be different). However, if we take many surfaces of the same type, and for each compute a set of model parameters that have the same physical meaning in each case, we can compute relevant statistical information from this training set or database. With such statistics in hand, we can construct prior distributions for the model parameters, which can be used constrain or regularize the values of those parameters for the surface to be reconstructed, thus regularizing the surface itself. In Chapter 4 we propose a wavelet model and statistical prior for model-based stereo and apply it to human face capture.

In this section we review existing work done on model-based stereo and stereo systems designed for 3D capture of human faces. While there is some overlap in that some model-based stereo methods are applied to human faces, i.e. a model is developed for the shape of the human face and fitted to stereo data, many systems designed for face capture do not leverage shape modeling, but are rather based on creating conditions under which the surface of a face is a good candidate for stereo reconstruction. This typically involves controlling the lighting conditions and ensuring sufficient pixel coverage. We first review model-based stereo followed by stereo-based face capture.

The idea behind model-based stereo is to assume one is trying to reconstruct the shape of a particular type or class of object or surface, to derive a model for the shape or structure of that object/surface, and to fit the parameters of that model to the stereo data. This inherently regularizes the reconstructed surface to vary according to the variability in the model parameters. If the model parameters are learned from a database of sample shapes, then the reconstructed surface will vary according to the variability of the training set. Another type of model-based stereo allows the model to be composed of geometric primitives commonly found in architectural structures [76]. This type of model-based stereo often also requires some user intervention, such as selecting lines that correspond to edges of the primitives. The primitive-composed model is then used to aid subsequent disparity or depth estimation.

The most prominent example of model-based stereo is also designed for human faces.

Amberg et al. [77] leveraged the 3D morphable model (3DMM) of the face from Blanz and Vetter [78] by performing Principal Component Analysis (PCA) on a set of training examples of 3D laser-scanned faces. The face was modeled by the mean of the set of training faces plus a linear combination of eigenvectors representing principal modes of shape variation in the training set. The mean face was computed by taking the mean of each coordinate of every vertex of every face mesh in the training set. The set of shape parameters are the coefficients applied to the shape eigenvectors before they are added to the mean. The transform from a face mesh to shape parameters consists of subtracting the mean and multiplying the result by a matrix whose rows are the eigenvectors. The inverse transform from shape parameters to vertex positions consists of multiplying the vector of shape parameters by the transposed matrix and adding the mean face shape.

With this model in hand, Amberg et al. [77] formulate model-based stereo for faces as large optimization problem where they solve for the shape parameters and calibration matrices of the input images to minimize a complicated energy function consisting of many terms. The *prior* term represents the probability that a human face has a particular shape and is computed using the $L2$-norm of the shape parameters, since these parameters represent the deviation of a particular face from the mean. Their energy function also includes silhouette term, a photoconsistency term and a landmark term. Due to the global support of the PCA transform (every shape parameter depends on every vertex position and every vertex position depends on every shape parameter), the shape parameters must be optimized simultaneously. Due to the large number of shape parameters, this means a local gradient-based optimization is needed, with the mean face serving as the initial estimate.

Sun et al. [79] used strong shape priors for stereo. They used an object-specific Markov random field (MRF) to integrate shape priors seamlessly with a weak prior on surface smoothness for articulated and deformable objects. Romeiro and Zickler [80] coupled a 3DMM with an occlusion map defined on the model shape to reconstruct faces in the presence of occluding objects. Tonko and Nagel [81] used model-based stereo of non-polyhedral objects for automatic disassembly tasks. Zhao et al. [82] performed stereo reconstruction by first fitting an approximate global parametric and then refining the model using local correspondence processes.

In Chapter 4, we apply a spherical wavelet transform to the surface mesh before perform statistical analysis, making our prior model a localized one in both space and frequency, and meaning our shape parameters are independent. This is unlike PCA, which has global support, meaning every shape parameter depends on every point on the surface and vice-versa. Since our shape parameters are independent, they can be optimized one at a time in coarse-to-fine

manner, unlike principal components, which must be optimized simultaneously. The localized model also allows the independent priors to be combined in anyway, allowing greater variation than a global model for the same training data.

2.4.1 Stereo Capture of Human Faces

Recently, several stereo systems have been developed for high-resolution face capture that do not use model-based stereo. Bradley et al. [83] use 14 high-definition synchronized video cameras and high-quality lighting to perform high-resolution passive capture of facial animation. With the number and resolution of the cameras they are able to use skin pores, blemishes and hair follicles as features to match in the multi-view stereo process. They incorporate automatic surface tracking based on optical flow with automatic drift correction and edge-based mouth tracking, which together yield high-quality, time-varying, textured face models.

Beeler et al. [72] also developed a multi-view stereo system designed for faces that yields impressive results. Their system can incorporate an arbitrary number of cameras. They begin with the simple approach of multi-resolution winner-takes-all block-matching using normalized cross-correlation (NCC), applied to neighboring pairs of cameras, but subsequently apply several stages of refinement. During the initial matching a Gaussian pyramid is used and results from lower resolution levels restrict the disparity search at higher levels. They further apply smoothness, uniqueness and ordering constraints to the initial matching process. An extensive refinement process is then applied to both the raw disparity maps and the subsequently generated surface meshes. In both cases, the refinement consists of a linear combination of a photometric consistency term and a surface consistency term that favors smooth solutions. The photometric consistency term essentially selects a maximum of the NCC, while the surface consistency term is an anisotropic second-order smoothing estimate. Finally, the surface mesh undergoes *mesoscopic augmentation*, wherein features too small for stereo matching are extracted using a high-pass filtered values for all surface points. The local high-frequency geometry is adapted to match the mesoscopic field computed on the mesh. A follow-up paper [84] combined this reconstruction with an image-space tracking technique for high-accuracy, high-detail passive facial performance capture. The tracking method is similar to that of Bradley et al. [83], with the addition of *anchor frames* which prevent drift and allow the same mesh structure to be tracked across multiple performance capture sessions of the same actor. Anchor frames are detected as those similar to the reference frame using a simple feature-description technique.

Furukawa and Ponce [85] developed a system for dense 3D motion capture of human faces

in which the faces are painted with special make-up to provide texture for the stereo matching algorithm. Their method deforms a laser-scanned model while regularizing with non-rigid tangential deformations to account for the complex deformations made by skin stretching in widely varying expressions. Several earlier works also made use of face paint or markers [86, 87, 88, 89].

In Chapter 4, we apply a model-based stereo approach to human face capture. Our approach leverages the robustness of a statistical shape prior in the wavelet domain, and uses the properties of the wavelet coefficients to simplify the procedure for optimizing this prior in conjunction with a data term fitting the model to noisy stereo data. We follow this with a stereo-matching and smoothing refinement stage. Both the wavelet shape space and the refinement stage allow us to reconstruct shapes not found in the linear subspace spanned to the examples in the training set, while still achieving the robustness of a statistical shape prior.

2.5 Multi-View Stereo

Multi-view stereo has been a long-standing important research problem in computer vision, with applications not just for view synthesis, but for any application of passive 3D sensing, such as architectural and cultural heritage modeling. Seitz et al. [6] reviewed and classified the existing multi-view stereo methods at the time. They classified MVS algorithms based on scene representation (volumetric, polygonal meshes, depth maps), photo-consistency measure, visibility model, shape prior, reconstruction algorithm, and initialization requirements. In this thesis, we propose the use of a depth/disparity map-based multi-view spherical stereo approach for RT-NVS. Reviewing all of the multi-view stereo literature is beyond our scope, and we instead review the most successful and most closely related methods. For example, while many early approaches were voxel-based, computing a 3D occupacy grid [90, 91, 92] or a 3D signed distance function [93, 94] within a variational framework, the storage and computational costs of discretizing a 3D volume generally do not allow these techniques to achieve the same level of detail as mesh- and depth map-based methods, which have been able to keep pace as the resolution of the images used in multi-view reconstruction has increased.

One of the top performing methods is Patch-based Multi-View Stereo (PMVS) [1]. This method reconstructs an oriented point cloud: each point has both a position and normal. The method proceeds by initial feature matching with epipolar constraints, feature patch expansion, and filtering stages. The filtering stage includes removal of patches lying outside the true surface (as determined by the other patches) and removal of patches lying inside the true surface. Another top performer is a mesh-deformation based method by Vu et al. [95]. This method

begins with a dense point cloud, and extracts a visibility-consistent mesh using a global energy minimization with visibility and surface quality terms. This mesh is then refined using a variational energy minimization, where the energy includes a photoconsistency term and a discrete thin-plate regularization term, which are mixed in a strategic way to apply greater regularization in areas where the photoconsistency is less reliable.

Goesele et al. [96] presented a depth-map based method for multi-view stereo, which evaluated the confidence of the depth map estimate when combining multiple depth maps. While the method did not necessarily generate a complete model of the scene or object, the reconstructed point were accurate. This method is, however, quite slow. A later work handled the extreme lighting, visibility and scale changes to perform multi-view stereo reconstruction from community photo collections [97]. This involved reasoning at the image level to determine which subsets of images were compatible for matching (based on similar lighting conditions, scale, sufficient baseline), and at the pixel level to handle clutter, occlusions and local lighting variations. Global view selection is performed by computing a weighted sum based on which SIFT [98] features are shared amount views.

Merrell et al. [99] presented a depth map fusion method that processes depth maps in real-time, making practical reconstruction from hundreds of thousands of individual depth maps computed from correlation-based matching. Zhang et al. [100] perform multi-view reconstruction from videos of static scenes, using disparity map bundle-optimization and fusion. This approach uses hierarchical belief propagation (HBP) [75] to compute initial disparity maps for each frame, with the aid of segmentation and plane-fitting, then incorporates inter-frame constraints between disparity maps into the likelihood function and re-runs HBP. Finally, a space-time fusion step enforces consistency between adjacent frames' disparity maps. Bradley et al. [101] present an efficient multi-view reconstruction method based on robust binocular stereo and surface meshing. Scaled window matching is uses a first-order differential model to account for distortions of the surface due to changes in viewpoint. This followed by surface reconstruction: point-cloud merging, downsampling, cleaning and meshing.

The multi-view stereo method for planar and spherical images presented in Chapter 6 is a disparity map-based method, and is most similar to Merrell et al., and to a lesser extent Zhang et al. (we do not use a global optimization to estimate individual disparity maps). Our fusion stage is something of a mix of the fusion methods of Merrell et al. and Zhang et al.

2.6 Spherical Stereo

Early work on omni-directional or panoramic stereo goes back to Ishigura and Tsuji [21], who comprised panoramas from rotated camera views, and obtained coarse approximations to the depth of objects by their movement in the views used to comprise the panoramas. They then used omni-directional binocular stereo to refine the object ranges similarly to traditional binocular stereo.

Li et al. [102] showed that the epipolar geometry of multiperspective cylindrical panoramas is approximately horizontal, similar to rectified planar images, which allows the use of standard stereo algorithms for reconstruction. Li et al. sample uniformly in inverse radial distance. In Chapter 6 we do this as well, however, we derive different epipolar relationships for single-perspective spherical panoramas. Kang and Szeliski [22] performed 3D scene reconstruction from omnidirectional multi-baseline stereo. They used feature tracking and the 8-point structure-from-motion algorithm to obtain relative pose of panoramas and 3D points, followed by multi-baseline stereo matching for a dense 3D point cloud to mesh. They noted that the wide field-of-view allows the relative pose of the panoramic viewpoints to be determined with much greater accuracy than standard planar image structure-from-motion due to the fact that features are matched in greater variety of directions relative to each viewpoint. Shum et al. [103] proposed an interactive method of constructing 3D models from panoramic images.

Previous approaches have also been proposed to perform stereo matching on spherical images [24, 104, 25, 26, 105, 27]. Shi [24] performs real-time stereo by window-based matching in angular disparity. Kim and Hilton [25, 26] use the same disparity formulation with a PDE-based method for scene modeling. Pagani et al. [27] reconstruct dense point-clouds from high-resolution spherical images using a spherical adaptation of PMVS [1]. They show impressive reconstructed point clouds of archaeological and cultural heritage sites, although they do not indicate the processing time required for their approach. The multi-view reconstruction method presented in Chapter 6 runs much faster than the original PMVS. Bagnato et al. [105] develop a variational approach for structure-from-motion from spherical images. While our formulation shares some commonalities with previous ones, we make use of two disparity quantities that allow us to both sample efficiently along epipolar arcs, and apply cost filtering and smoothing in a geometrically correct way. Since the same angular disparity (angle of rotation about the normal of the epipolar plane) results in different distances from the center of projection for different directions (pixels) on the sphere, cost filtering or disparity smoothing in angular disparity does not result in regularizing the distance of reconstructed points from the

center of projection. We describe in detail these differences in Section 5.3. Further, our stereo algorithm is different from all of these approaches.

2.7 Image-based Modeling and Novel View Synthesis

Recalling the definition cited in the Chapter 1, image-based rendering attempts to complete a continuous representation of the plenoptic function, given a set of discrete samples of that function. This is often accomplished by restricting the set of possible viewpoints, by obtaining geometric information about the scene, by simplifying or approximating that geometry, or by all of the above. In their survey [106], Shum et al. place different IBR methods along a continuum parameterized by the amount of geometric information they use.

Here, we categorize NVS methods as in Section 1.3: view interpolation, column or strip pasting, layer-based methods, image-based methods, and particle-based methods. Typically view interpolation and column pasting methods use the least amount of geometric information, followed by image-based methods, while layer-based and particle-based methods typically involve reconstructing some form of dense depth-based geometric proxy.

The original view interpolation method is due to Chen and Williams [7]. In view interpolation, the novel viewpoint is restricted to lie on the line connecting the image centers of two images. This restriction reduces the amount of geometric information that must be recovered to synthesize a novel view. In View Morphing, Seitz and Dyer [8] compute perspective-correct warping between the input images and the novel view based on manually selected correspondences between the input images. The light field [107] requires virtually no geometric information, however, they restrict the novel view to lie in a viewing plane, which is parallel to the object plane where the content to be rendered is placed. The lumigraph [108] is very similar to the light field, but includes depth corrections to the mapping of light rays from the object to the viewing plane.

As mentioned in Section 1.3, we use the term column or strip pasting to describe methods that fuse vertical strips from densely captured images into a novel view, which is not necessarily fully perspective. This type of method is commonly used with panoramic input images. Concentric Mosaics [9] use planar input images, but capture them at video rate along a circular path using a mechanical arm moving at a fixed speed. This results in a set of densely captured images, each a fixed distance from its neighbors. Near-perspective views can be generated for viewpoints in the circle or disc of radius equal to half the length of the mechanical arm by pasting columns from the input images based on the line-of-sight of pixels in the novel view.

Column-pasting methods for NVS based on panoramic images include that of Aliaga and

Carlbom [10] and Aliaga et al. [11]. In the former, catadioptric panoramic images are used along with a simple geometric proxy to render novel views. Panoramic images are captured in an irregular grid of intersecting smooth paths through the environment. The intersections of these paths are found to construct image loops. Novel views within an image loop are reconstructed by warping the pixels from the boundary of the loop to the novel view. While the pixels warped from the input images are not vertical columns in the input images, they consist of lines that correspond to vertical lines in the environment (as a result of the catadioptric image formation). In the latter, omnidirectional images are captured every few inches and compressed and stored in a multi-resolution hierarchy that allows to prefetch the necessary images to generate a novel view by resampling them in real-time. A simple geometric proxy is used to facilitate the resampling.

Another approach to interactive navigation of street-view panoramic images is Street-Slide [12]. As the user zooms out, or backs up, they are transitioned from a single-viewpoint panorama, or "bubble", to a multi-perspective panorama that is generated by pasting columns from nearby panoramas according to the novel view direction. This gives a wide-angle view of the street. This approach assumes densely captured panoramas, so as to have enough columns corresponding to the directions of columns in the novel view, does not reconstruct a fully perspective novel view, and restricts the set of viewpoints that can be synthesized. This method is nonetheless very effective for browsing street-level imagery in an interactive manner.

Another column-pasting method uses the crossed-slits projection [109], which is a generalization of the pinhole projection model. This method tends to be restricted to horizontally moving camera configurations.

As we increase the amount of geometric information used in generating novel views, NVS methods look more like stereo methods. For methods we call image-based, the main difference from stereo reconstruction is that constraints are placed, or regularization performed, on the novel view, rather than the surface geometry. Depth values are typically enumerated for each pixel in the novel view, with a photoconsistency function evaluated at each depth. However, as in the prominent example of Fitzgibbon et al. [18], an energy function is minimized over the pixel colors in the novel view rather than the depth values. The minima in depth of the photoconsistency function are used as candidate colors of the respective pixel. Fitzgibbon et al. proposed dictionary-based image priors to regularize the pixel values. A dictionary of 5 pixel patches is constructed from the input images. The prior energy term for a patch in the novel view is then the distance to its nearest neighbor in the dictionary. This results in a novel view that is locally very similar to the input images. Because of the high-order nature of the prior, the optimization had to be performed using iterated conditional modes (ICM), which is prone

to get stuck in local minima. Woodford et al. [19] developed pairwise dictionary priors that allowed dictionary-based priors to be expressed in terms of the two-node cliques, thus allowing the energy function to be minimized using global optimization methods for pairwise MRFs (such as belief propagation, graph cuts, or tree-reweighted message passing). The reduction in clique size is accomplished by restricting the patch dictionary for each clique to image regions suitable for that clique using the epipolar geometry of the input images and the novel viewpoint.

Another image-space method constructs a joint conditional random field (CRF) on the pixels of the novel view [20]. Again depth estimates are obtained only as a means to compute potentials on the color values of the novel view pixels. For each pixel in the novel view, the color is compared directly to the color of the corresponding pixels in the input images. However, the result of convolving the novel view with a set of steerable filters is compared to the result of convolving the input images with the same filterbank, again for each pixel in the novel view and its correspondences in the input images. This second term forces pixel neighborhoods in the novel view to be similar to pixel neighborhoods in the input images, but only one value per filter per pixel must be compared to the input images, rather than the entire patch (neighborhood). An online learning algorithm optimizes the global parameters of the CRF.

For the RT-NVS application of this thesis, it is more common to explicitly perform stereo reconstruction, and then texture the resulting geometry. This is because such representations can be very efficiently rendered using modern graphics hardware. We consider two main categories of these types of approaches: layer-based and particle-based. One prominent early approach that does not fit well into either category is the work of Debevec et al. [76], in which photogrammetry techniques leveraging user input and model-based stereo are used to reconstruct the geometry of architectural scenes from photographs. The reconstructed model is then textured with the photographs using a view-dependent technique that weights the influence of the input images based on the angle between their viewpoint and the virtual viewpoint.

The approach we propose to pursue in this thesis builds upon, and estimates a set of, layered depth panoramas (LDP) [15], which are themselves an extension of layered depth images (LDI) [13]. These layered representations take multiple input images of the scene, and reconstruct multiple color and depth layers. While the layers are typically defined in the reference frame of one of the input images, they can be rendered efficiently from any viewpoint using either the splatting method used by Shade et al. [13], or using a mesh-based method that leverages the capabilities of modern GPUs [14]. Multiple layers means multiple color and depth values per pixel, which allows objects that are occluded in one input image, but not in another,

to become visible as the user moves the viewpoint away from that image's center of projection.

Zheng et al. [15] constructed a single LDP with a virtual viewpoint from multiple (around 20) planar images by formulating the problems of estimating depth and color as discrete labeling problems with single-node and pairwise energies, and solving them using a graph cut method [74]. The approach begins with a cylindrical sweeping stage that computes the median color and a robust variance for a set of depth values for each pixel in a cylindrical panorama (i.e. for each (ϕ, z, d) tuple). Each depth is designated with a particular label, and the optimal labeling minimizes an energy function comprised of data term and a smoothing term. For the data term they used the robust variance computed during the cylindrical sweep. For the smoothing term they used the product of a monotonically increasing function of depth difference between adjacent pixels and a monotonically decreasing function of the difference between median colors at those pixels and depths. This encouraged depth discontinuities to coincide with intensity or color edges in the input images.

Solutions were found first for the depth of the first layer, followed by the color of the first layer, then by any subsequent layers (they found two or three layers were often sufficient). When solving for the color at a given layer, the labels identified the image from which to take the color to assign to the layer. Again the energy function was the sum of a data term and a smoothing term, with the data term enforcing that the image used should be most aligned with the virtual camera. Again the smoothing term was a product of a monotonically increasing function of the difference between adjacent labels and a monotonically decreasing function, this time of the difference between adjacent depths. This lowers the smoothing costs at depth discontinuities. Once the first depth layer is estimated, subsequent layers were prohibited from having depths less than or equal to that of the first layer.

Micusik and Kosecka [23] constructed piecewise-planar 3D models of urban environments from sequences of streetview panoramic images. Noting that using planar images in urban environments can be difficult due to the fact that a single image often only contains a single planar facade with repetitive structure, they employed a piecewise perspective panoramic camera model with a single virtual center of projection (essentially a cube map without the top and bottom), the wide field-of-view of which they leverage for robustness in their pose estimation stage. Following this is a sweeping stage that finds candidate projections (depth and normal) for a set of super-pixels, identified via segmentation, from each panorama. Super-pixels are treated as planar surfaces, which are then projected into nearby panoramas. Subsequently, a MAP labeling problem is solved to assign one of the candidate depths and normals to each super-pixel, and the resulting depth maps are then fused using a viewpoint dependent approach.

Lei et al. [110] perform free viewpoint video rendering with a space-time consistent depth

recovery framework. They extend the region tree binocular stereo method to include a temporal neighbor between regions in successive frames by motion estimation. Their results improve on the temporally independent depth maps of Zitnick et al. [14], however it is not clear that the improved depth maps result in a better novel view.

Particle-based methods for free viewpoint rendering from planar images also use multi-view stereo reconstruction to generate a dense geometric proxy for rendering. One example is the work of Hornung and Kobbelt [16]. A particle cloud is generated for each input image, where each particle has a silhouette-sensitive orientation, and photo-consistency is computed over a particle's volume rather than in image space or on the estimated surface. They further propose to use a logarithmic sampling of the depth of the particles in the scene, rather than linear or inverse sampling of depth. We propose to achieve greater coverage of the scene and better handling of dynamic environments through a panoramic layered representation. We also suggest that the inverse sampling of depth is preferable because it provides better estimates of nearby objects, where parallax is more noticable.

Ambient point clouds [17] are combined with standard point clouds and textured meshes for view interpolation. Ambient point clouds have a size that depends on the uncertainty of their estimated position. They first reconstruct the geometry that can be reconstructed with high certainty in the form of interpolated depth maps that indicate which points are reconstructed and which are set on an impostor plane. For pixels that could not be reconstructed, an ambient point cloud is created by distributing the color of that pixel along the viewing ray in the form of points placed randomly along that ray. This approach avoids visual artifacts in regions where depth information is unreliable by incorporating the uncertainty into the visualization. However, the novel viewpoints are restricted to be near the line connecting the two camera centers to maintain the proper epipolar geometric relationship between the input images and the novel view.

Takahashi et al. [111] sample slits from physically nearby panoramic images for arbitrary view point rendering, but do not reconstruct geometry. This method is similar to that of Aliaga and Carlbom [10] and Aliaga et al. [11]. Zhu et al. [112] generated and analyzed panoramic epipolar plane images (EPI) on a moving platform, taking vibration into account.

2.8 Discussion

As we have seen in the review of related work, many directions have been explored in the task of synthesizing novel views. Many of these directions have found success, but nothing so far has generally solved the problem of RT-NVS. The solution we propose in this thesis is

based on multi-view stereo reconstruction of the scene. By estimating a disparity map for each input image, we can generate corresponding point clouds, and either merge them or leave them separate. These point clouds can be efficiently rendered from any viewpoint, and with a density, coverage and accuracy that are determined by the input images. By using spherical images, we can cover an entire scene with fewer images than would be necessary with planar images. This however, creates additional challenges in terms of performing stereo reconstruction from spherical images.

To solve the sub-problem of spherical multi-view stereo reconstruction we first solve pairwise spherical stereo matching (Chapter 5). We perform a spherical sweeping stage in which we compute dissimilarity measures for all pixels in a spherical depth map, generating a spherical disparity space image (DSI) [29] using a novel multi-scale matching framework described in Section 5.2, which minimizes an energy over DSI at all scales, in a time- and space-efficient way. We develop a formulation of the disparity space of spherical images in Section 5.3 that is continuous, efficient to sample, and abstracts the underlying sampling structure of the spherical images. Unlike previous formulations we are able to obtain a near-uniform sampling of the disparity space. In Chapter 6 we extend this approach to multi-view reconstruction, and demonstrate real-time virtual walkthroughs of real environments by rendering the reconstructed point clouds.

In Chapter 3, we make a contribution to spherical image and geometry processing by presenting a GPU method for performing SH transforms and spherical convolutions on the GPU. This is a useful tool for applying a filterbank to a spherical image or spherically sampled geometric data, decomposing it into an overcomplete spherical wavelet representation. This can be used for multi-scale feature detection.

In Chapter 4, we make a contribution to model-based stereo, by introducing a wavelet shape prior, which can be efficiently optimized by global search. While we formulate this as a model-based stereo approach, and apply it to human faces, it is in fact a more general model-based approach to parameterizing point clouds of a specific object type.

Chapter 3

Spherical Harmonic Transforms and Convolution on the GPU for Panoramic Imaging

In this chapter[1], we present a GPU implementation of the semi-naive method [31] for the spherical harmonic transform. We use this method to transform spherical images into the frequency domain (spherical harmonic coefficients), where convolution is faster than in the spatial domain. In subsequent chapters, we propose to use this approach to decompose spherical panoramas into an overcomplete spherical wavelet representation that is invariant under rotations of the underlying parameterization. This means that the decomposition of a rotated image is equal to the same rotation applied to the decomposition of the unrotated image. This is due to the overcompleteness or redundancy of the representation, which in turn is due to the fact that all resolution levels are sampled at the full resolution, and only the level-of-detail is decreased. The wavelet transform is computed by convolving the image with a set of filters (a filterbank), that act as bandpass filters, giving the components of the image at different resolutions. We then use this image representation in matching spherical panoramas.

This representation is useful for multi-scale edge detection. In this chapter, we demonstrate this for spherical images. However, the same principle applies to detecting edge features in geometric data mapped to the sphere. For example, this same overcomplete spherical wavelet representation has been used to detect cortical folding in MRI data, and study the development of these folds over time [28].

The same approach allows us to generate other feature spaces from spherical images as

[1]Content of this chapter which overlaps with prior publication [113] is copyright AK Peters 2010.

well. For examples, by convolving with a Gaussian filterbank, i.e. several Gaussian filters with different standard deviations, we can compute Difference of Gaussian (DoG) features, which are commonly used in image matching. For example, a spherical version of SIFT [98] as presented by Cruz-Mota et al. [114] might be efficiently implemented using the techniques presented in this chapter.

The work of this chapter has been published in the Journal of Graphics, GPU and Game Tools [113].

3.1 Introduction

Spherical harmonics (SH) have been proven useful in many areas including computer graphics, computer vision and image processing. They allow one to examine the different frequency components of data in the spherical domain in isolation and as a whole. When the data are sampled in Euclidean space, the main tool is standard Fourier analysis. When the data are sampled on the sphere, the equivalent is spherical harmonic analysis. In this paper, we present GPU methods implemented using compute unified device architecture (CUDA) [30] for the forward and inverse spherical harmonic transform, spherical convolution, and stereographic dilation.

We show an example of how to apply the transform to panoramic image data and perform decomposition and reconstruction of the data according to an overcomplete spherical wavelet model. The model is a variant of that initially proposed by Yeo et al. [45]. Analogous to filtering and Fourier coefficients in Euclidean domains, the model enables filtering in the spherical domain by a point-wise product of SH coefficients [41]. Spherical convolution is more easily carried out in the spherical harmonic domain than in the spatial domain.

We provide two methods for SH transforms. One is a GPU implementation of the direct transform. The main advantage of this implementation versus others is its low storage requirements and its simplicity when implemented on the GPU. While slower than asymptotically fast algorithms on the CPU that employ involved data structures with high memory costs [41, 115, 116], the low memory costs and simplicity allow it to accommodate large transform sizes in GPU memory and allow it to handle non-power-of-two sized transforms. The second method is a GPU implementation of the semi-naive algorithm by Healy et al. [31], which utilizes the DCT to reduce the size of the summations that must be performed. This implementation is faster than CPU implementations of the same algorithm as well as more complex algorithms by a factor of between 5 and 6 depending on the size of the transform. Next, we summarize the necessary background for our implementation of the SH transform.

In Section 2.2.1, we gave the formulation of the SH transform, both continuous and discrete versions, and noted that the complexity of the direct implementation of the transform was $O(N^3)$ where N is the bandwidth of the transform. Now let us discuss asymptotically fast algorithms.

Driscoll and Healy [41] proved convolution theorems for the sphere by generalizing well known and useful results from the conventional Fourier analysis, and presented the first $O(N^2 \log^2 N)$ algorithm for the forward transform and an $O(N^3)$ algorithm for the inverse. Among their key results is the formula for frequency domain computation of the convolution of two spherical functions, and the result that the transform of a band-limited function can be computed exactly by means of finite sums of the sampled values of that function, as in (2.6).

Mohlenkamp [117] presented algorithms that run in $O(N^{5/2} \log N)$ and $O(N^2 \log^2 N)$ time. The former, referred to as the one-dimensional algorithm, although asymptotically slower, performed better in practice than the latter, referred to as the two-dimensional algorithm. In his experiments, Mohlenkamp observed that the one-dimensional algorithm was faster by a factor of three than the direct algorithm at a problem size of $N = 512$.

Healy et al. [31] presented algorithms that run in $O(N^2 \log^2 N)$ time exploiting shifted recurrence relations of the Legendre functions by smoothing, subsampling and precomputing lower frequency basis functions, and computing higher degree basis functions in term of lower ones. They generalized the recurrence relation by iterating it forward r times, producing trigonometric polynomials $A_r^{(L)}$ and $B_r^{(L)}$ of degree r and $r-1$ that circumvent the computation of higher degree associated Legendre functions. Smoothing and subsampling are performed through the discrete cosine transform (DCT), using the observation that a trigonometric polynomial of degree r has at most $r+1$ non-zero DCT coefficients. From the orthogonality of the DCT, the inner product of the DCT coefficients is equal to the inner product of the samples, hence fewer operations are needed in the DCT domain. This is the basis of the semi-naive algorithm, which we implement on the GPU. Asymptotically faster variants, such as the classic Driscoll and Healy algorithm [41] and the simple-split and hybrid methods of Healy et al. [31], employ different forms of a divide-and-conquer approach that splits the transform into successively smaller problems where the lower-degree trigonometric polynomials can be used to accelerate the transform. However, Healy et al. found some of these asymptotically faster variants to have stability problems for certain orders, and the semi-naive algorithm to be faster in practice than virtually all of them except the hybrid method.

Suda and Takami [118] introduced split Legendre functions, and used fast polynomial interpolation and the fast multipole method (FMM) to accelerate the Legendre transform, approximating the SH transform in $O(N^2 \log N)$ time. Ramamoorthi and Hanrahan [36] use the

separation of variables for low frequency SH that runs in $O(NN_s)$, assuming $N < \sqrt{N_s}$ where N_s is the number of samples.

While there are many implementations of the SH transforms [119, 115, 116, 120, 121, 122], we compare to SpharmonicKit [120] in Section 3.3, because it is widely used and includes implementations of the algorithms of Driscoll and Healy [41] and Healy et al [31]. Another parallel implementation is ccSHT [122], which uses the message passing interface (MPI) for parallel execution on a cluster or network, but it does not use the GPU.

3.2 Parallel Spherical Harmonic Transforms Using CUDA

We use CUDA [30] for our parallel implementation. Parallel computation in CUDA is comprised of thread blocks. Each block contains the same number of threads, and the blocks are organized into a grid. The blocks must be executable in any order and in parallel to allow for scalability. Within a block, all threads have access to the same *shared memory*.

We begin with the $O(N^3)$ separation of variables approach, which we refer to as the direct method. For each value of m and k such that $-N < m < N$ and $0 \leq k < 2N$, we compute the sum over all j

$$s_{mk} = \sum_{j=0}^{2N-1} f(\theta_k, \phi_j) e^{-im\phi_j} \tag{3.1}$$

which amounts to a shifted one-dimensional Discrete Fourier Transform for each value of k. We can compute these using the Fast Fourier Transform (FFT) implemented in the cuFFT library, which allows one to run a batch of one-dimensional FFTs in parallel.

Let us now write the sums as $2N - 1$ vectors $\mathbf{s}_m$ for each m. We transpose the data from row-major, one row per value of k, to column major, one column per value of m, using a simple kernel, which also applies the necessary scale factors to the FFT result. The computation of coefficient f_{lm} is now reduced to the discrete inner product

$$f_{lm} = \frac{1}{4\pi} \langle \mathbf{s}_m, \mathbf{P}_{lm} \rangle = \frac{1}{4\pi} \sum_{k=0}^{2N-1} [\mathbf{s}_m]_k [\mathbf{P}_{lm}]_k \tag{3.2}$$

where $\mathbf{P}_{lm} = [\bar{P}_{lm}(\cos\theta_0) \ \ldots \ \bar{P}_{lm}(\cos\theta_{2N-1})]^T$ is the vector of $2N$ discrete samples of the normalized associated Legendre function $\bar{P}_{lm}$, and $[\cdot]_k$ denotes component k of a vector.

We can see that this computation takes $O(N^3)$ time as follows. There are $O(N)$ vectors $\mathbf{s}_m$ each of length $O(N)$, which are projected onto $\mathbf{P}_{lm}$ for $l = m, \ldots, N-1$. Hence each of $O(N)$ vectors undergoes an $O(N)$ operation $O(N)$ times. Put another way, the inner product

```
projectSums(l,N,numThreads,sumsPerThread,sums,legendreTable)
m = blockIdx.x - l //-l to l
iVector = N - 1 + m //index of sum-vector 0 to 2N-2
t = threadIdx.y * blockDim.x + threadIdx.x //thread index (2D)
kStart = t * sumsPerThread //start index for this thread
sharedResult[t] = 0
for k = kStart to kStart + sumsPerThread
  s = sums[iVector,k] //result s_mk of Fourier transform
  plmm1 = legendreTable[1,iVector,k] //Legendre for l-1
  plmm2 = legendreTable[2,iVector,k] //Legendre for l-2
  theta = (0.5 + k) * dtheta
  x = cos(theta), y = sin(theta)
  plm = normALF(l,m,x,y,plmm1,plmm2) //apply recurrence
  sharedResult[t] += s * plm //update sum for thread
  legendreTable[0,iVector,k] = plm //store Legendre result
end for
if t == 0 then
  for k = 1 to numThreads - 1 //sum over all threads
    sharedResult[0] += sharedResult[k]
  end for
  write sharedResult[0] to coefficient storage
end if
```

Figure 3.1: Pseudocode listing of kernel performing parallel projection of summation vectors $\mathbf{s}_m$ onto associated Legendre vectors $\mathbf{P}_{lm}$.

in equation (3.2) takes $O(N)$ time and there are N^2 coefficients f_{lm} to be computed in this manner.

We assign one block per value of m, and multiple threads to perform the summation in shared memory, with one thread completing the summation by adding the intermediate results and writing the final result to global memory. The method loops $l = 0, \ldots, N-1$, at each value computing in parallel for each $m : |m| \leq l$ the inner product (3.2) as follows. From the thread index t we obtain the subset of vector components to be multiplied and summed, $[k_t, k_{t+1} - 1]$. The thread then loops over the appropriate values of k and computes $[\mathbf{P}_{lm}]_k$ using recursion (2.4), with $\mathbf{P}_{l-1,m}$ and $\mathbf{P}_{l-2,m}$ stored from the previous two iterations. Precomputing and storing a full table of Legendre functions would be too costly in terms of space to justify the minor improvement in speed. However, below we discuss how the DCT can be used to

make precomputation efficient. Next, the thread computes the component product $[\mathbf{s}_m]_k[\mathbf{P}_{lm}]_k$ and adds this to the thread's own tally f^t_{lm} for thread t. The intermediate sums f^t_{lm} are stored in shared memory so that each intermediate sum is accessible by all threads. Once all threads have completed their part of the summation, thread $t = 0$ computes the full summation from the intermediate sums, $f_{lm} = \frac{1}{4\pi}\sum_t f^t_{lm}$. A pseudocode listing of the kernel performing this operation for a given value of l is shown in Figure 3.1. The function `normALF` on line 12 is a device function that implements the recursion (2.4). The value of `dtheta` on line 10 is $\pi/2N$.

We can compute these inner products more efficiently, as mentioned in Section 2.2.1, by computing these inner products in the cosine domain. If we precompute $\mathbf{P}_{lm}$ and perform a Discrete Cosine Transform (DCT) on it, we can reduce both the number of operations needed and the space required for the precomputed data. The associated Legendre function $\mathbf{P}_{lm}$ is a trigonometric polynomial of degree $l + 1$, which means that only its lowest $l + 1$ DCT coefficients are non-zero. Further, if $l - m$ is even then only the even coefficients are non-zero, and only the odd ones if $l - m$ is odd. If we also take the DCT of the input vectors $\mathbf{s}_k$ then the inner product can be computed in $O((l + 1)/2)$ multiplications and additions. Note that the algorithm remains $O(N^3)$, but the constant is substantially lowered. We can perform the inner product in the cosine domain because the DCT uses an orthonormal basis, i.e. it is an energy preserving transform. In practice Healy et al. [31] found that this method, called the semi-naive method, performed equally well as their hybrid method for $N < 1024$ and nearly equally well for $N = 1024$, as corroborated by our experiments in Section 3.3.

We implement the DCT again using cuFFT and using the relations from Vetterli and Nussbaumer [123], allowing us perform all $2N - 1$ DCTs in parallel. Before executing the cuFFT plan, a kernel re-orders the input samples. After the FFT, a kernel applies a rotation and scaling to the resulting Fourier coefficients to give the DCT coefficients. We can make use of shared memory to sum the DCT coefficient products more efficiently by assigning one thread block to each coefficient, where the number of threads in each block is a power of two. Each thread is assigned a small number of terms to sum, followed by a parallel reduction scheme to sum over the threads. Let the number of threads in each block be 2^n. Then by iterating for $j = 0, \ldots, n - 1$ the sum $f^t_{lm} \leftarrow f^t_{lm} + f^{t+t_j}_{lm}$ where $t_j = 2^j$ and $t \mod 2t_j = 0$, we are left with the total sum in f^0_{lm}. Figure 3.2 gives a pseudocode listing for the kernel that projects the DCT coefficients of the input vectors onto the non-zero DCT coefficients of the Legendre functions.

An inverse transform is performed using the semi-naive method by transposing the sums-of-products. That is, for each order m and DCT coefficient index k, we sum over the relevant degrees $l \geq m, k$ the product of the SH coefficient and the precomputed Legendre coefficients.

This is followed by an inverse DCT for each m, and subsequently an inverse FFT for each k.

While this approach greatly reduces the running time for the SH transform, the memory footprint greatly increases. For $N = 1024$, the precomputed data does not fit in graphics memory on a 1 GB card. While the storage requirement of the direct method is $O(N^2)$, the storage requirement of the semi-naive is $O(N^3)$. The DCT coefficients of the Legendre functions can be stored in single precision without significant impact on the accuracy of the transform, cutting the storage requirement in half, but this is not enough.

3.3 Examples

Figure 3.3 shows our intended application. A panoramic image (top left) is filtered using spherical convolution in the frequency (SH) domain using a five-filter filterbank, creating an overcomplete spherical wavelet representation [45]. Similarly to conventional Fourier analysis, spherical convolution is more efficient (a point-wise product) in the SH domain than the angular domain. In the case of a filterbank, the image needs to be transformed once, and then it can be easily filtered for each resolution level. The resulting wavelet is overcomplete in that all resolution levels are equally sampled.

We adapt the spherical wavelet model introduced by Yeo et al. [45] and used by Yu et al. [28]. It is based on an extension of continuous filterbank theory to the 2-sphere [45], and as such a spherical function is decomposed into its various resolution frequency components by spherical convolution with a set of decomposition or analysis filters producing a set of wavelet functions or coefficients. Reconstruction is performed similarly by inverse spherical convolution with a set of reconstruction or synthesis filters, and summing the results. Due to the wide convolution kernels used in our filterbank, spatial domain convolution would become impractical and therefore we perform convolution in the frequency domain. The filters are stereographic dilations of a mother wavelet. In the case of Figure 3.3, the mother wavelet is the planar Laplacian of Gaussian filter of radius 4, i.e. 4 is the standard deviation of the Gaussian, stereographically projected onto the sphere.

We show experiments to evaluate both the speed and the numerical accuracy of our implementations. In our first experiment we compare the running time of our SH transform implementation versus a reference implementation, SpharmonicKit [120], which implements the algorithms of Driscoll and Healy [41] and Healy et al. [31]. In our second experiment we compare the accuracy of our implementation to that of the reference implementation.

Our test data are cubic panoramas of approximately 6 Mpixels of RGB data. We first resample each cube onto the latitude-longitude grid of dimensions $2N \times 2N$, where N is the

bandwidth of the transform. We performed our tests on a desktop PC running Linux with a Pentium Core2 Quad CPU, 8 GB of memory, and a NVidia GeForce GTX 280 GPU with 1 GB of graphics memory.

3.3.1 Speed

Figures 3.4 and 3.5 show the experimental running times of different methods for the forward and inverse SH transforms, respectively.

From these figures we see that while a GPU implementation of the direct method is not sufficient to compete with more advanced algorithms that use large data structures, a GPU implementation of the semi-naive bests the hybrid algorithm of SpharmonicKit by a factor of 5 or 6 depending on the transform size. Our direct implementation computes the Legendre functions on the fly, without the need to store large look-up tables, which allows it to compute large transforms in limited graphics memory and to compute non-power-of-two sized transforms. Note that while our semi-naive method uses the real-basis and the SpharmonicKit implementation uses the complex basis, they too use a shortcut for real data, only computing the inner products for $m \geq 0$ and using the relation $Y_{l,-m} = (-1)^m Y^*_{lm}$, so the number of Legendre transforms is the same.

The inverse transform times for the GPU semi-naive method are slower than the forward times, more than double for $N = 512$. This is because although the computation is transposed, the precomputed DCT coefficients of the Legendre functions are not transposed in memory. Thus, for the inverse transform memory accesses of the precomputed data are not contiguous across threads in the same block. The result is that these reads from global memory cannot be coalesced and memory access is slower. This could be alleviated by storing a transposed version of the DCT coefficients in main memory, and swapping it with the forward precomputed data in graphics memory before performing an inverse transform.

3.3.2 Numerical Accuracy

We conduct two accuracy experiments. The first experiment tests the accuracy of projecting a function into the SH domain and then reconstructing it. This involves comparing the distribution of error in spherical coordinates. Starting with an input grid, the forward transform is applied and then the inverse, and the errors are analyzed. The second experiment is the inverse of the first. We start with SH coefficients, and transform to the spatial domain and then back, measuring the error in the resulting coefficients. As opposed to the first experiment, these errors result only from the inaccuracies in the computation, since were are starting with

N	GPU-direct-C	GPU-direct-R	GPU-semi	FST-semi-memo
128	6.50×10^{-5}	6.50×10^{-5}	1.04×10^{-6}	2.19×10^{-11}
256	1.15×10^{-5}	1.15×10^{-5}	7.24×10^{-7}	2.76×10^{-11}
512	2.03×10^{-6}	2.03×10^{-6}	5.38×10^{-7}	3.31×10^{-11}

Table 3.1: Mean absolute coefficient error from inverse-forward round-trip transforms for different algorithms and implementations at different problem sizes.

a function that is by definition band-limited. The results of this experiment are compared to SpharmonicKit.

The original input images were cubic panoramas with three 8-bit channels (RGB), i.e. the values are in the range $[0, 255]$. These cubes were resampled in spherical coordinates before being input to the forward SH transform. We measured the mean absolute difference between the red channel of pixels in this input spherical image and those of the reconstructed spherical image obtained by applying an inverse transform to the results of the forward transform. We found that the error was very similar for all methods, ranging from less than 4.6 for $N = 128$ to less than 1.5 for $N = 512$. This is as expected, since adding high-frequency coefficients improves the approximation of color discontinuities. (We report results for the red channel, but the accuracy of other channels are virtually identical, as expected.)

Table 3.1 shows the absolute error of the inverse-forward round-trip transforms for different methods at different bandwidths. These statistics were computed as follows. The given method's forward transform was used to generate 27 sets of coefficients from the same set of spherical images as before. These coefficients were used to initialize an inverse transform. After each inverse transform, the resulting spherical grid was used as input to a forward transform, and the absolute difference of the resulting coefficients from the initial ones was measured. Table 3.1 shows that the GPU-semi method has slightly smaller and the FST-semi-memo program has significantly smaller mean coefficient error than either GPU-direct method. This is likely due to the fact that the semi-naive method uses the DCT to reduce the number of additions and multiplications in the inner-product step. The GPU implementation of the semi-naive method uses single-precision floating point values in several places to save memory consumption, and this likely explains the loss of accuracy with respect to the CPU implementation. Note that this is not reflected in the reconstruction error, indicating that in all cases the floating-point error is insignificant compared to the reconstruction error or the magnitude of the application data.

3.4 Discussion

We have presented a parallel implementation of the direct and semi-naive algorithms for the SH transform on the GPU using CUDA. The main advantage of our direct implementation is its low memory requirement, while our semi-naive implementation is 5 to 6 times faster than the state-of-the-art algorithms on the CPU depending on the size of the transform.

Performance could be further improved by using more efficient FFT algorithms and implementations [124, 125] for the GPU rather than cuFFT. Using texture references may accelerate, via caching, access of global device memory. The performance of the inverse semi-naive algorithm on the GPU could be improved by keeping a transposed version of the precomputed data in system memory, and copying it to the graphics memory before performing an inverse transform.

While our results as presented do not demonstrate real-time performance on current architectures, GPU processing power is accelerating more rapidly than that of CPUs. The implementation presented here can be used for multi-scale feature detection in both spherical image processing and spherical geometry processing.

Further improvements in efficiency could be achieved with some low-level modifications of the way reduction-summations are performed, to reduce the number of thread warps that must be active in a given step. More generally however, the notion of a computational plan, where threads are assigned *a priori* to particular portions of the summation for particular coefficients, could achieve a more uniform distribution of computational workload, and therefore greater efficiency.

```
0  projectSemi(N,sumsPerThread,dctLeg,dctSums,coeffs)
1  b = blockIdx.y * gridDim.x + blockIdx.x //coefficient index
2  l = floor(sqrt(b)) //get degree l from index b
3  m = b - l*l - l //get order m from l and b
4  s = m + N - 1 //index of sum-vector
5  t = threadIdx.x //thread index (1D)
6  kstart = t * sumsPerThread
7  kstop = kstart + sumsPerThread //range of thread summation
8  __shared__ subsum[2^nIters] //shared intermediate storage
9  subsum[t] = 0
10 if (l-m) is even
11   kstart += (kstart mod 2) //force kstart to be even
12 else
13   kstart += (1 - (kstart mod 2)) //force kstart to be odd
14 jstart = kstart / 2 //precomputed data is zero-stripped
15 kstop = min(kstop, l+1) //upper bound for whole summation
16 legOff = getSemiNaiveOffset(l, m) //Legendre offset
17 sumOff = s * 2N //offset for sum-vectors
18 for (k = kstart, j = jstart; k < kstop; k+=2, j++)
19   subsum[t] += dctLeg[legOff + j] * dctSums[sumOff + k]
20 step = 1 //sum over all threads
21 for (iter = 0; iter < nIters; iter++)
22   __syncthreads()
23   if t is a multiple of 2 * step
24     subsum[t] += subsum[t + step]
25   step *= 2
26 if t == 0
27   coeffs[b] = subsum[0] //coefficient result
```

Figure 3.2: Pseudocode listing of kernel to compute the inner product in cosine domain.

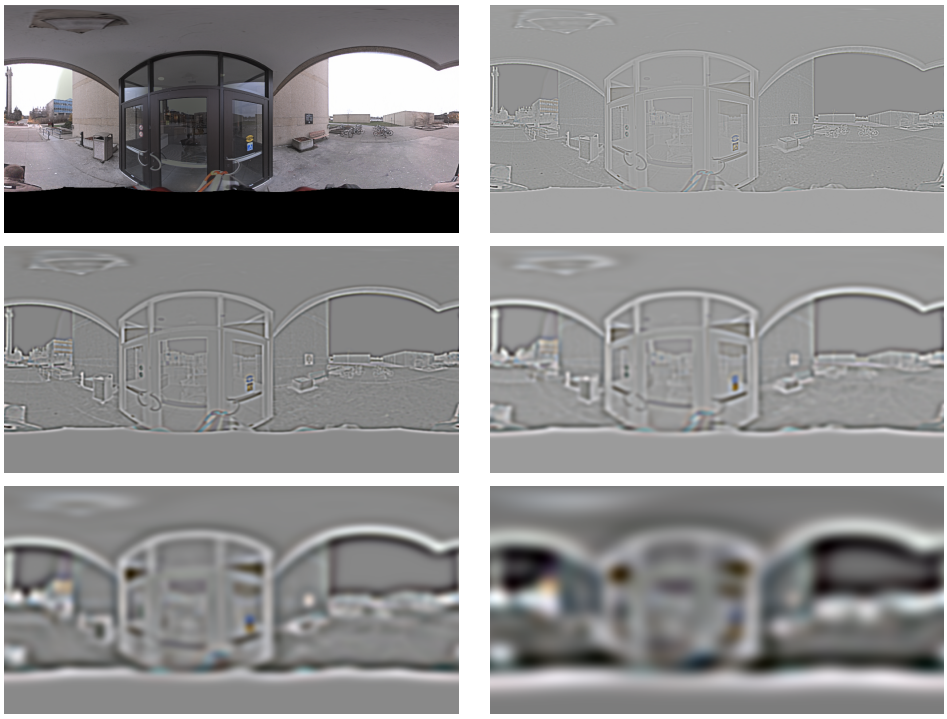

Figure 3.3: A spherically sampled panoramic image and convolution results with five filters from the filterbank. Note that the intensity levels in the convolution results are normalized to the range of values for each result. The higher frequency results have smaller ranges of values.

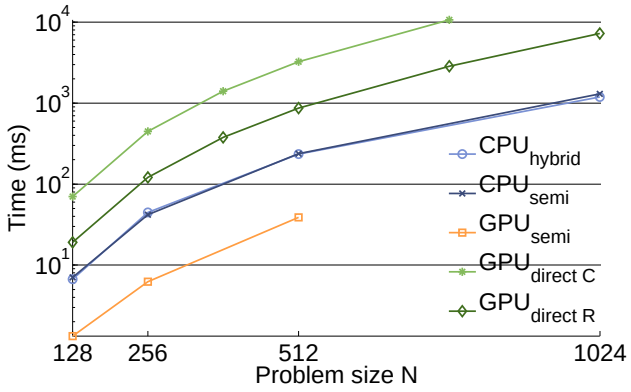

Figure 3.4: Running times for different forward SH transform algorithms and implementations for different problem sizes. Each data point was computed as the mean of 27 trials. Abbreviations: GPU-direct-C is the direct method, complex basis, on the GPU; GPU-direct-R is direct, real basis, on the GPU; GPU-semi is the semi-naive method, real basis, on the GPU; CPU-semi is the semi-naive method from SpharmonicKit; CPU-hybrid is the hybrid algorithm from SpharmonicKit.

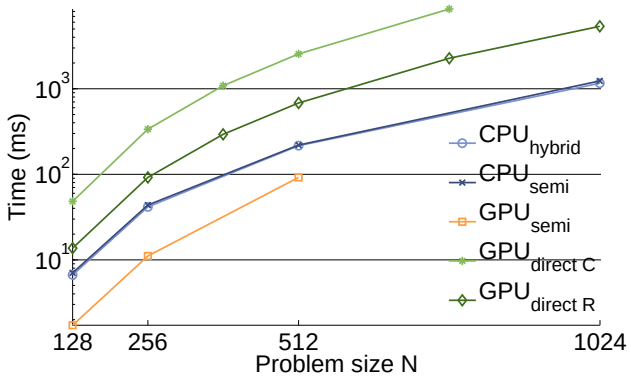

Figure 3.5: Running times for different inverse SH transform algorithms and implementations for different problem sizes. Each data point was computed as the mean of 27 trials. Abbreviations: GPU-direct-C is direct method, complex basis, on the GPU; GPU-direct-R is direct method, real basis, GPU; GPU-semi is semi-naive, real basis, GPU; CPU-semi is the semi-naive method from SpharmonicKit; CPU-hybrid is the hybrid algorithm from SpharmonicKit.

Chapter 4

A Wavelet Prior for Model-Based Stereo

The following chapter[1] describes an approach to stereo reconstruction of class-specific objects that uses a statistical prior shape model of the object in question to constrain the stereo-matching process. This type of approach is known as model-based stereo, and it alleviates some of the most difficult problems in stereo matching such as occlusion and specularity. While this chapter demonstrates the approach for two planar images of a human face, the approach can be generalized to any shape that can be mapped to a sphere, and could be applied to panoramic images as well. This approach is dependent on the existence of a statistical shape model learned from a training set of examples of the type of shape in question. In the absence of such a prior model, or in the case where we wish to reconstruct general surfaces, other more general priors, including weak priors such as surface smoothness, must be used instead.

The shape model used in this chapter is based on a subdivision surface wavelet decomposition, a biorthogonal spherical wavelet. This allows us to sample the space of human faces in a multi-resolution divide-and-conquer technique, and find the face that optimizes the combination of the shape prior and a data cost. This is due to the decorrelating properties of the wavelet basis, as explained below, which allow us to optimize shape parameters independently in a coarse-to-fine manner.

This chapter advances us towards our goal of NVS from panoramic images by illustrating how the wavelet domain can be used to regularize stereo reconstruction, in this case using a statistical prior. In Chapter 6 we use another form of spherical wavelet without statistical information to smooth disparity maps in a way that accounts for color information from the input images, and confidence information about the disparity estimates themselves.

A preliminary version of this chapter has been published in the Canadian Conference on

[1]Content of this chapter which overlaps with prior publication [126] is copyright IEEE 2011.

Computer and Robot Vision (CRV) 2011 [126].

4.1 Introduction

Stereo reconstruction of class-specific objects, *e.g.*, human faces, may benefit from prior knowledge of the shape of the objects to be reconstructed. The prior models, learned from statistical analysis of 3D shapes, constrain the reconstruction. In practice, it is necessary to build the prior models in a compact representation so that they can be used efficiently to infer 3D shapes from images. Principal component analysis (PCA) has been widely used for this purpose. For example, Amberg et al. [77] learn a PCA model from 3D scans and vary the model to best fit the input images. However, PCA is a global model, which means every parameter affects the whole shape. For stereo, it is desirable to have a prior model that only influences the shape variations locally. We propose a statistical wavelet model for representing the shape variations and use it in a Bayesian framework to reconstruct human faces from two or more images.

Statistical shape priors have been used within single-view reconstruction techniques, *e.g.*, Blanz and Vetter [78]. Using multiple views gives us an observation that expresses the likelihood of a shape based on 3D geometric constraints, i.e. perspective correspondence between the views, as well as surface appearance.

Stereo reconstruction can be broken into binocular stereo and multi-view stereo. Although we only use two images, our approach is more along the lines of multi-view approaches and could straightforwardly be extended to more views. Scharstein and Szeliski [127] provided a survey of the binocular case, and Seitz et al. [6] surveyed and classified the multi-view techniques.

Amberg et al. [77] used model-based stereo to reconstruct human faces. They used a 3D morphable model (3DMM) [78] based on PCA of a training set of faces. In such a framework, shapes are modeled as the mean shape plus a linear combination of eigenvectors which represent eigen shape variation modes, computed from the training set. The mean shape was computed as the mean points' coordinates. PCA is a transform with global support; each shape parameter depends on each vertex position, and vice versa. This means that all parameters have to be optimized together, typically in a complex gradient descent fashion, which may get stuck in a local minimum. In contrast, we use a wavelet basis that has localized support in both space and frequency. This makes the transform computationally efficient, both in terms of time and space, and allows us to use a simple global optimization algorithm. The wavelet basis also better captures the variability present in the training data; because the model parameters are independent, we can generate shapes not found in the linear subspace spanned by the training

set.

Recently several methods for high-quality stereo capture of human faces have been proposed [83, 72, 85]. While we do not reconstruct with the same precision as some of those methods, we do not require special face-paint, or near-optimal lighting conditions or high-accuracy calibration. Our approach is computationally efficient and can be orders of magnitude faster than these methods. Our approach can incorporate any range information, i.e. our method is not restricted to stereo data. Additionally, we obtain a face mesh that is in a common coordinate frame and in correspondence with the shape database, making our approach ideal for tracking and recognition.

When optimizing a model to best fit a set of observations, it is desirable that the representation be compact, meaning that most of the energy resides in a few coefficients. Thus only those relatively few parameters need to be manipulated to fit the model. PCA, Fourier and spherical harmonic representations are compact, but the basis functions are not localized in space, hence the coefficients in one scale have to be optimized together, making the optimization problem complex. By using a wavelet basis we are able to solve the optimization in a simple divide-and-conquer approach. Li et al. [34], performed segmentation of 3D Neuroradiological data using a statistical wavelet prior similar to the one we use here. As in their approach, we use a generalized B-spline subdivision spherical wavelet [32].

Sun et al. [79] used strong shape priors for stereo. They used an object-specific Markov random field (MRF) to integrate shape priors seamlessly with a weak prior on surface smoothness for articulated and deformable objects. Romeiro and Zickler [80] coupled a 3DMM with an occlusion map defined on the model shape to reconstruct faces in the presence of occluding objects. While we do not explicitly model occluding objects, our prior is strong enough to reconstruct shapes in the presences of occlusions. Tonko and Nagel [81] used model-based stereo of non-polyhedral objects for automatic disassembly tasks. Zhao et al. [82] performed stereo reconstruction by first fitting an approximate global parametric and then refining the model using local correspondence processes. Koterba et al. [128] studied the relationship between multi-view Active Appearance Model (AAM) fitting and camera calibration.

Wavelets and other multi-resolution techniques have previously been used for stereo for over a decade, but most often [60, 66, 64, 61, 63, 65] a wavelet transform is applied to the images and then the resulting wavelet coefficients are matched in a coarse-to-fine manner, which allows for larger displacements between corresponding pixels. Miled et al. [67] used a wavelet domain representation of the disparity map to regularize stereo reconstruction, but their prior is an edge-preserving smoothing prior, as opposed to a statistical shape prior.

Wavelet shape priors, have been used previously for medical image segmentation [34, 129],

and for single-view reconstruction [130, 131].

In this chapter we make the following contributions: the use of a statistical wavelet shape prior for fast, robust model-based stereo reconstruction of human faces; a sampling-based Bayesian framework that can incorporate arbitrarily many priors and observations; the re-parameterization of human face scans with a subdivision sampling compatible with our wavelet model. Our approach is robust with respect to noisy observations and works under sub-optimal lighting conditions. While we demonstrate our technique for faces, we emphasize that we can generalize it to any shape that can be parameterized using a compatible subdivision surface. The reconstructed shape is captured in a common, registered shape space making it immediately ready for other processing, such as tracking and recognition. Additionally, we leverage a GPU-based implementation that accelerates the bottlenecks in our pipeline.

4.2 Overview

We begin by learning a statistical wavelet model of the human face, and then use it to robustly fit the model to noisy stereo data followed by stereo matching refinement. Given a database of corresponded triangular meshes of laser scans of human faces, for each face we resample it into a subdivision surface, and then decompose the surface using a wavelet basis [32] into independent components that are localized in space and frequency. We then learn statistics on the distributions of the resulting wavelet coefficients over a training set, and then use them as a statistical prior to guide stereo reconstruction within a Bayesian framework.

Formally, we parameterize the shape of the human face by a high-dimensional vector space $\mathcal{S}$, and learn a model for the prior probability $P(\mathbf{s})$ of a shape parameter vector $\mathbf{s} \in \mathcal{S}$. We then model the observational likelihood of a set of input images $\mathcal{I}$ and a point cloud from a general-purpose stereo algorithm $\mathcal{Y} = \{\mathbf{y}_i : \ i = 0, \ldots, N_y - 1\}$, and we solve for the *maximum a posteriori* (MAP) configuration of $\mathbf{s}$ given $\mathcal{I}$ and $\mathcal{Y}$,

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} P(\mathbf{s}|\mathcal{I}, \mathcal{Y}) \tag{4.1}$$

where $\mathcal{I} = \{I_L, I_R\}$ for the purposes of this chapter. By Bayes' Theorem we have the posterior

$$P(\mathbf{s}|\mathcal{I}, \mathcal{Y}) = \frac{P(\mathbf{s})P(\mathcal{Y}|\mathbf{s})P(\mathcal{I}|\mathbf{s}, \mathcal{Y})}{P(\mathcal{Y})P(\mathcal{I}|\mathcal{Y})} \tag{4.2}$$

which, because $\mathcal{I}$ and $\mathcal{Y}$ are constant, simplifies to

$$P(\mathbf{s}|\mathcal{I}, \mathcal{Y}) = c_{obs} P(\mathbf{s})P(\mathcal{Y}|\mathbf{s})P(\mathcal{I}|\mathbf{s}, \mathcal{Y}) \tag{4.3}$$

where c_{obs} is a constant of proportionality. We further factor the prior into a component based on the statistics of the training set and a smoothing component: $P(\mathbf{s}) = P_{st}(\mathbf{s})P_{sm}(\mathbf{s})$. More details of the model are given in Section 4.3. We compute the MAP configuration through energy minimization as described in Section 4.4. Our straightforard optimization technique, which is a combination of Monte Carlo sampling or particle filtering methods and iterative partial maximization, is made feasible by the properties of the wavelet basis. We implement our technique using GPU programming and standard image processing tools in a framework that allows the incorporation of additional observations and priors as described in Section 4.5, where we also present results on stereo data and low-resolution, relatively noisy laser range scans.

4.3 The Model

We model the surface we wish to reconstruct as a wavelet decomposition of a subdivision surface. Although we map the face to a plane, we use a second-generation subdivision wavelet scheme [32] that allows our model to be extended to any surface that can be mapped onto the unit sphere.

In the learning phase of our method, we start with a database or training set of triangular meshes of laser-scanned faces. However, these meshes are not subdivision surfaces, which prevents the use of a wavelet model, so we resample them onto a Catmull-Clark subdivision grid by stereographic projection of a template face. Corresponding vertices of all faces are mapped to the same point on the plane, that of the template, to preserve correspondence. We then decompose the surface into its wavelet coefficients. This, in turn, allows us to compute a statistical model of the wavelet coefficients generated from a database of such scans, and use it as a strong statistical prior to guide surface estimation.

Before we can proceed we must rigidly align the shapes in the training set, i.e. put them all in the same coordinate system, so that the variation in the 3D coordinates of the vertices is due only to the change in the shape, and not to any rotation, translation or scaling. Hence, we first align the triangular meshes with each other using generalized Procrustes alignment (GPA) [132], which iteratively aligns each shape in the set to the average shape of the set. After each iteration the average shape is recomputed using the realigned shapes. We then rigidly align the resulting mean face to the template mesh.

4.3.1 Subdivision Resampling

The triangular meshes are resampled onto a quadrilateral Catmull-Clark subdivision surface configured as a regular 2D grid as follows. We stereographically project the template mesh onto a plane aligned with the front of the face and passing through its centroid, saving the mapping as 2D coordinates in the plane. Let this plane be at $z = 1$. Stereographic projection maps the entire surface of a sphere to a plane, mapping a 3D point to the point in the plane at $z = 1$ that is passed through by the line-segment connecting the 3D point and the point $[0\ 0\ \ -1]^T$. Let $\mathbf{p}_n$ be an arbitrary vertex in the template triangular mesh. This vertex is mapped to a point in the plane $\mathbf{x}_n$ by stereographic projection as follows,

$$\mathbf{x}_n = \begin{bmatrix} x_n/(z_n+1) \\ y_n/(z_n+1) \end{bmatrix}$$

where $\mathbf{p}_n = [x_n\ y_n\ z_n]^T$. Let $\mathbf{p}_{i,n}$ be the same vertex in triangular mesh i from the training set; $\mathbf{p}_{i,n}$ is mapped to the same position in the plane $\mathbf{x}_n$ as is the template vertex $\mathbf{p}_n$, for every mesh i in the training set. In this way, corresponding vertices are always mapped to the same position in the plane, and because all meshes have the same connectivity and all their faces are planar, any point in the plane that is covered by applying this mapping to any one of the meshes corresponds to the same point on any of the other meshes under the same mapping. Thus, we can resample the surfaces while maintaining the correspondence between surfaces.

The planar coordinates are used as texture coordinates to resample the triangular mesh using the rasterization capabilities of the GPU. The locations of grid points are chosen by defining an orthographic projection of the texture coordinates using the rectangle bounding the mesh in the plane. The resolution of the grid is determined by taking an arbitrary base resolution and subdividing it the desired number of times.

4.3.2 Wavelet Decomposition

Let us denote the surface by $\mathbf{f} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, and further by $\mathbf{f}(\mathbf{x})$ at grid point $\mathbf{x} = (x, y)$. The wavelet model is then expressed by

$$\mathbf{f}(\mathbf{x}) = \sum_{n \in V(0)} \mathbf{v}_n^0 \phi_n^0(\mathbf{x}) + \sum_{j=0}^{J} \sum_{m \in W(j)} \mathbf{w}_m^j \psi_m^j(\mathbf{x}) \tag{4.4}$$

where the terms are defined as follows. The set of vertices in the level-j approximation of the surface is denoted by $V(j)$. The approximation is refined through subdivision by adding the vertices in $W(j)$, thus $V(j+1) = V(j) \cup W(j)$. The basis function $\phi_n^0(\mathbf{x})$ denotes the

lowest-resolution scaling function centered on the vertex indexed by n, and $\psi_m^j(\mathbf{x})$ denotes the level-j wavelet basis function centered on the vertex indexed by m. The corresponding coefficients $\mathbf{v}_n^0$ and $\mathbf{w}_m^j$ form our shape representation which we wish to best fit to an observation by minimizing an energy function. Specifically, we concatenate the coefficients into a shape parameter vector $\mathbf{s}$, with the lowest resolution coefficients coming first. Let us denote the (3D) coefficient indexed by k by $\mathbf{s}^k$.

Because these basis functions have only local support, $\mathbf{f}(\mathbf{x})$ only depends on a few coefficients, and the coefficients can be computed in linear time. Both decomposition and reconstruction are comprised of a series of lifting operations, which take $O(1)$ complexity at each node at each resolution level. Since there are $N/4^{J-j}$ nodes in level j, and $N+N/4+N/16+\ldots < 2N$, the total number of times the lifting operations must be applied is $O(N)$ in either the decomposition or reconstruction. The lifting operations effectively predict one coefficient using its neighbors in the grid, subtract the prediction from the true value leaving the residual component that is not correlated to the neighbors according to that prediction model or filter. Because the transform is biorthogonal, we may assume the coefficients are fully decorrelated, i.e. independent, and can be optimized individually.

4.3.3 Statistical Prior

We now define the observation and prior components of our model. We model the prior probability of the wavelet coefficients as independent Gaussian distributions, and compute statistics on a database of face shapes. Let $\mathbf{s}_i$ denote the shape parameter vector of face i in the database $\mathcal{F} = \{\mathbf{f}_i, \mathbf{s}_i, \mathbf{r}_i\}_1^n$, where n is the number of faces in our database and $\mathbf{r}_i$ will be defined shortly. Our prior model is defined by three shape quantities. The first is simply the mean shape parameter vector

$$\bar{\mathbf{s}}^k = \frac{1}{n}\sum_{i=1}^{n} \mathbf{s}_i^k \tag{4.5}$$

for $k = 0, \ldots, P-1$, where P is the number of shape parameters and each 3D coefficient vector $\mathbf{s}^k$ in a shape parameter vector $\mathbf{s}$ can be treated independently because of the decorrelating and localizing properties of the wavelet basis functions. The face surface $\bar{\mathbf{f}}$ that is reconstructed by applying (4.4) to $\bar{\mathbf{s}}$ is shown in Figure 4.1.

While we can perform statistical analysis on each $\mathbf{s}^k$ independently of other values of k, we must consider their three components together. Each $\mathbf{s}^k$ is a 3D vector representing either the scale (absolute value) or the detail (relative value) of the shape at a particular frequency and spatial location. However, the coordinate axes in general do not correspond to the main

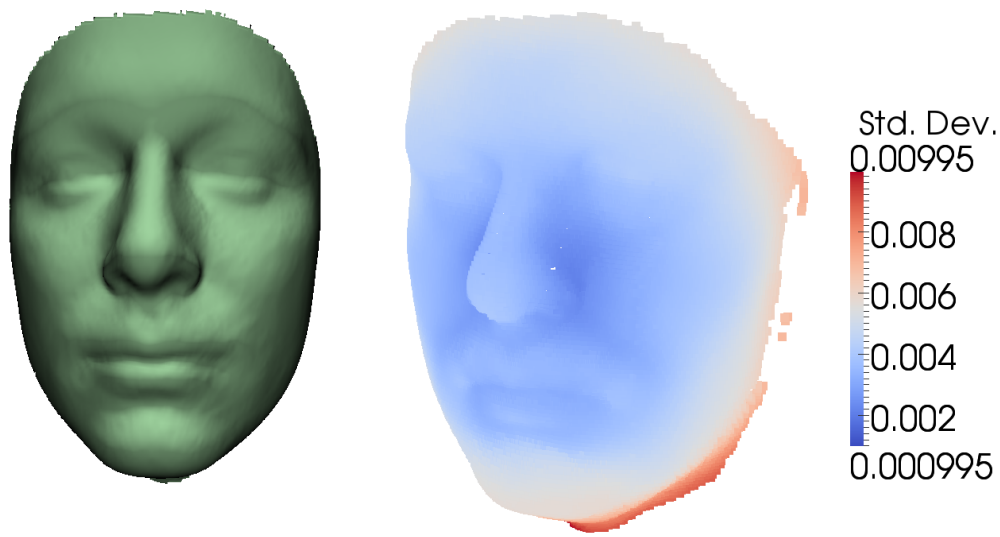

Figure 4.1: Left: The surface $\bar{\mathbf{f}}$ reconstructed from the mean shape vector $\bar{\mathbf{s}}$. Right: false color visualization of the magnitude in metres of the standard deviation of the model parameters associated with each vertex in the full-resolution grid mesh, $|\sigma^k|$.

directions of variation in $\mathcal{F}$ of $\mathbf{s}_i^k,\ i = 1, \ldots, n$. Therefore, we perform PCA on each set of coefficient vectors, to obtain 3D vectors $\mathbf{r}_i^k$ that represent position along the directions of greatest variation, and 3×3 matrices U^k that transform these coordinates to our original world coordinate system, as in

$$\mathbf{s}_i^k = \bar{\mathbf{s}}^k + U^k \mathbf{r}_i^k \tag{4.6}$$

where we write $\mathbf{s}^k = [x_s^k, y_s^k, z_s^k]^T$ and $\mathbf{r}^k = [x_r^k, y_r^k, z_r^k]^T$ to denote the components of these vectors, and $\mathbf{r} = [\mathbf{r}^{0T}, \mathbf{r}^{1T}, \ldots]^T$ to denote the complete vector of statistical shape parameters.

Due to the orthogonality of the wavelet basis functions and the basis of the principal component analysis, we may justify assuming that $\mathbf{r}^k$ is independent from $\mathbf{r}^m$ for $m \neq k$, and that the components x_r^k, y_r^k and z_r^k form zero-mean Gaussian distributions that are independent from each other. From the training set we can learn the standard deviation of each component $\sigma^k = [\sigma_x^k, \sigma_y^k, \sigma_z^k]^T$. The standard deviation across the surface is shown in the right side of Figure 4.1. This allows us to write the prior probability of a surface $\mathbf{f}$ as

$$P(\mathbf{f}) = P(\mathbf{s}) = P(\mathbf{r}) \tag{4.7}$$

where $\mathbf{f}$ relates to $\mathbf{s}$ by (4.4), and $\mathbf{s}$ relates to $\mathbf{r}$ by (4.6) and

$$P(\mathbf{r}) = \prod_k P(\mathbf{r}^k) \tag{4.8}$$

and

$$P(\mathbf{r}^k) = \left(\frac{1}{\sigma_x^k\sqrt{2\pi}}e^{-\frac{(x_r^k)^2}{2(\sigma_x^k)^2}}\right) \cdot \left(\frac{1}{\sigma_y^k\sqrt{2\pi}}e^{-\frac{(y_r^k)^2}{2(\sigma_y^k)^2}}\right) \cdot \left(\frac{1}{\sigma_z^k\sqrt{2\pi}}e^{-\frac{(z_r^k)^2}{2(\sigma_z^k)^2}}\right) \tag{4.9}$$

4.3.4 Observations

To model the likelihood of observing the point cloud $\mathcal{Y}$ and the images $\mathcal{I}$, given a shape parameter vector $\mathbf{s}$, we reconstruct the surface $\mathbf{f}(\mathbf{x})$ from $\mathbf{s}$ using (4.4). We assume the point cloud is a noisy approximation of the surface $\mathbf{f}$, with approximately zero-mean Gaussian noise. Hence, we represent the probability of observing the point set, given the current as an exponential distribution on the distance of the model to the point set. Specifically, we use the sum-of-squared distances of the model vertices to their nearest neighbors in the point cloud. In practice, most stereo algorithms have some systemic error in addition to noise, but we alleviate this by using a truncation threshold on the nearest neighbor distance in addition to using a prior.

We further assume the surface is approximately Lambertian, and project the surface into both images and perform stereo matching during a refinement stage. Although human skin can be both quite specular and translucent, we counteract the effects through the statistical prior and through matching techniques. We use robust matching costs to account for outliers due to specularities and we explicitly take self-occlusion into account. Our framework can be extended to include additional occluders. We further use an anisotropic second-order smoothing energy to regularize the refinement. We define our likelihood as an exponential distribution, where because they are deterministically related,

$$P(\mathcal{I}|\mathbf{r}, \mathcal{Y}) = P(\mathcal{I}|\mathbf{s}, \mathcal{Y}) = P(\mathcal{I}|\mathbf{f}, \mathcal{Y}) \tag{4.10}$$

and

$$P(\mathcal{I}|\mathbf{f}, \mathcal{Y}) \propto \exp(-E_M(\mathbf{f}, \mathcal{Y}, I_L, I_R)) \tag{4.11}$$

where E_M is a matching cost defined in Section 4.4. The matching depends on the point cloud in that the nearest neighbor distance is used to determine how far to sample during stereo refinement, when the mesh $\mathbf{f}$ is optimized with respect to 4.11.

While a smoothness constraint is conceptually a prior, i.e. the prior knowledge that the face is piecewise smooth, because it is applied to the mesh vertices and not to the wavelet coefficients, we treat it as part of the refinement process, optimized in conjunction with the matching cost.

4.4 The Energy Function and Minimization

In this section, we derive an energy or objective function from our probability model, and describe how we minimize that function. Our energy function consists of two parts, again representing observation and prior of our model.

Our first data cost is the sum-of-squared distances of the model vertices $\mathbf{f}$ to their nearest neighbors in the point cloud $\mathcal{Y}$, obtained as an initial estimate from a conventional stereo algorithm. On initialization, we find the nearest neighbor of each vertex in $\mathbf{f}$. The energy is then

$$E_{NN} = \sum_{\mathbf{x}} \min\left(\|\mathbf{f}(\mathbf{x}) - \mathbf{y}_{\mathbf{x}}\|, \tau_{NN}\right) \tag{4.12}$$

where $\mathbf{y}_{\mathbf{x}} \in \mathcal{Y}$ is the nearest neighbor of $\mathbf{f}(\mathbf{x})$, and τ_{NN} is a truncation constant to mitigate the effect of outliers, noise and incomplete data (i.e. holes in the point cloud where the initial stereo estimate could not reconstruct the surface).

Our matching energy E_M is defined over the surface map and the input images,

$$E_M(\mathbf{f}, \mathcal{Y}, \mathcal{I}) = \sum_i \sum_{j \neq i} \sum_{\mathbf{x}} w_M(\mathbf{x}) D(I_i, I_j, \mathbf{f}(\mathbf{x})) \tag{4.13}$$

where $i, j \in [0, |\mathcal{I}| - 1]$ denote the each pair of reference and matching images in the set, D is a point-wise dissimilarity measure, and $w_M(\mathbf{x})$ is a per-vertex weight. The dissimilarity D is defined as

$$D(I_i, I_j, \mathbf{p}) = 1 - NCC(I_i(\mathbf{q}), I_j(H_{ij}\mathbf{q})) \tag{4.14}$$

where $NCC(\cdot, \cdot)$ is the normalized cross-correlation (averaged over the red, green and blue channels) of two image patches containing an equal number of samples, $I_i(\mathbf{q})$ denotes the image patch around point $\mathbf{q}$, which is the projection of $\mathbf{p}$ into image I_i, and H_{ij} is the homography mapping a point in the image plane of I_i to the plane tangent to the surface at point $\mathbf{p}$ with normal $\mathbf{n}$ to the image plane of I_j. This is given by

$$H_{ij} = K_j \left(R_{ij} - \frac{\mathbf{t}_{ij}\mathbf{n}^T}{d_i} \right) K_i^{-1} \tag{4.15}$$

where K_i and K_j are the intrinsic calibration matrices of I_i and I_j, the matrix $[R_{ij}|\mathbf{t}_{ij}]$ transforms coordinates relative to I_i to coordinates relative to I_j, and d_i is the depth of $\mathbf{p}$ with respect to I_i. Thus, H_{ij} projectively maps points in the image patch surrounding $\mathbf{q}$ in image I_i, to the tangent plane to the surface at point $\mathbf{p}$, to the correspondging point in I_j.

The per-vertex weight is designed to reflect the reliability of the dissimilarity or photo-consistency function D. As described further below, the refinement stage iteratively takes

the current mesh $\mathbf{f}$, and for each vertex $\mathbf{f}(\mathbf{x})$ with normal $\mathbf{n}(\mathbf{x})$, samples the smoothing and photoconsistency functions at points above and below the mesh vertex along the normal direction. Let these sample points be denoted by $\mathbf{p}_m$, and thus we have matching cost samples $D(I_i, I_j, \mathbf{p}_m)$, where m is an index. Let $\hat{D}(\mathbf{x}) = \min_m D(I_i, I_j, \mathbf{p}_m)$ denote the minimum dissimilarity or matching cost for a given vertex. Then the matching weight is given by

$$w_M(\mathbf{x}) = \sum_m \Big(D(I_i, I_j, \mathbf{p}_m) - \hat{D}(\mathbf{x}) \Big) \tag{4.16}$$

which is greatest when there is one low miminum of the matching cost, and the rest of the samples are high. Hence, the weight is highest when the matching cost gives the most distinctive information about the surface. We further use depth-buffering to determine if a point is visible in both the reference and matching images. If it is not, $w_M(\mathbf{x})$ is set to zero.

For a smoothing energy term we use the distance of a vertex to an average of the neighboring vertices. We first compute the smoothed vertex position

$$\bar{\mathbf{f}}(\mathbf{x}) = \frac{w_x(\mathbf{f}(\mathbf{x}_l) + \mathbf{f}(\mathbf{x}_r)) + w_y(\mathbf{f}(\mathbf{x}_u) + \mathbf{f}(\mathbf{x}_d))}{2(w_x + w_y)} \tag{4.17}$$

where $\mathbf{x}_l = (x-1, y)$, $\mathbf{x}_r = (x+1, y)$, $\mathbf{x}_u$ and $\mathbf{x}_d$ are defined similarly. The horizontal weight is defined as

$$w_x = \exp\left(-\left(\|\mathbf{f}(\mathbf{x}_l) - \mathbf{f}(\mathbf{x})\| - \|\mathbf{f}(\mathbf{x}_r) - \mathbf{f}(\mathbf{x})\| \right)^2 \right)$$

and w_y is defined similarly. This smoothing is a variation of the one used by Beeler et al. [72] for disparity map refinement. Because we have a quadrilateral mesh, we can apply it to the vertex coordinates. The smoothing energy is then

$$E_{sm}(\mathbf{f}) = \sum_{\mathbf{x}} \left\| \mathbf{f}(\mathbf{x}) - \bar{\mathbf{f}}(\mathbf{x}) \right\| \tag{4.18}$$

reflecting how each vertex in $\mathbf{f}$ deviates from the anisotropic average of its neighbors (4.17).

The prior or regularization term in our energy function is taken directly as the negative logarithm of the prior $P(\mathbf{r})$. That is,

$$E_{st}(\mathbf{r}) = \sum_k \left(\frac{(x_r^k)^2}{2(\sigma_x^k)^2} + \frac{(y_r^k)^2}{2(\sigma_y^k)^2} + \frac{(z_r^k)^2}{2(\sigma_z^k)^2} \right) . \tag{4.19}$$

We combine these to get our energy or objective function,

$$E(\mathbf{s}) = w_{st} E_{st}(\mathbf{r}) + w_{NN} E_{NN}(\mathbf{f}) + E_M(\mathbf{f}) + w_{sm} E_{sm}(\mathbf{f}) \tag{4.20}$$

where $\mathbf{f}$ and $\mathbf{r}$ are related to $\mathbf{s}$ as before, and w_{st}, w_{NN} and w_{sm} are user-controlled parameters.

To minimize (4.20) we break the optimization into two parts. We first use sampling methods based on the learned distributions of our model parameters to minimize E_{st} and E_{NN}. As noted by Li et al. [34], we can serialize the parameter sampling because the orthogonality of the wavelet basis and the principal components allows us to assume independence between parameters. This leads to a complexity of $O(PS)$ for P parameters and S samples instead of $O(P^S)$ without the independence assumption. We examine two sampling strategies: uniform sampling and stochastic sampling. In the first case, we sample each parameter (x_r^k,y_r^k or z_r^k) uniformly within three standard deviations (*e.g.*, $\pm 3\sigma_x^k$). In the second case, values of, for example, x_r^k are chosen at random from the distribution $\mathcal{N}(0, \sigma_x^k)$.

After each sample value x_r^k, the wavelet coefficient $\mathbf{s}_k$ is reconstructed using the PCA eigenvectors U^k and the mean shape coefficient $\bar{\mathbf{s}}^k$, then the face surface $\mathbf{f}$ is reconstructed using (4.4).

One of the main drawbacks of statistical shape priors, especially for face reconstruction, is that they produce overly regularized results that do not deviate sufficiently from the mean shape. This is particularly a concern for faces, where much of the identifying detail is contained in finer scales. Hence, after optimizing the first five levels of coefficients using the sampling method described above, we follow with an iterative mesh refinement stage that minimizes E_M and E_{smooth} together. Following refinement we transform the surface back into the model parameters. We formulate this two-stage optimization as iterative partial maximization, where we optimize the model in terms of one part of the energy function and then in terms of the other. The first part is the combined energy of the statistical prior and the nearest neighbor distance. The second part is the smoothing and matching energies.

As mentioned above, refinement proceeds by sampling along the normal directions for each vertex in the mesh. Thus for mesh vertex $\mathbf{f}(\mathbf{x})$ with normal $\mathbf{n}(\mathbf{x})$, we have sample points $\mathbf{p}_m = \mathbf{f}(\mathbf{x}) + m\delta_{\mathbf{x}}\mathbf{n}(\mathbf{x})$ for $m = -N_r, \ldots, N_r$, where $\delta_{\mathbf{x}}$ is a user-controlled step size parameter. The sample that minimizes the combination of smoothing and matching energies is taken as the new vertex position. That is,

$$\mathbf{f}(\mathbf{x}) \leftarrow \arg\min_m E_{ref}(\mathbf{p}_m)$$

where E_{ref} is the per-vertex combined smoothing and matching energy,

$$E_{ref}(\mathbf{p}_m) = w_{sm} \left\| \mathbf{p}_m - \bar{\mathbf{f}}(\mathbf{x}) \right\| + w_M(\mathbf{x}) D(I_i, I_j, \mathbf{p}_m) \tag{4.21}$$

favoring smooth surfaces where matching information is missing or unreliable. Refinement is performed by iteratively sampling in this way for each reference-matching image pair in succession.

Algorithm 1 gives high-level pseudocode for the reconstruction algorithm.

```
/* Paramter sampling                                                    */
foreach shape parameter k do
    foreach component of r^k, r ∈ {x_r^k, y_r^k, z_r^k} do
        initialize r = 0 and compute initial energy E = w_st E_st(r) + w_NN E_NN(f)
        for sample i = 0 ... S − 1 do
            compute sample value r_i
            generate wavelet coefficient using local PCA matrix s^k = U^k r^k
            perform inverse wavelet transform
            compute energy E_new = w_st E_st(r) + w_NN E_NN(f)
            if E_new < E then
                set r = r_i and E = E_new
            end
        end
    end
end
/* Refinement                                                           */
foreach vertex f(x) where x = [x, y]^T is a grid point do
    initialize E = E_ref(f(x))
    for m = −N_r, ..., N_r do
        set p_m = f(x) + m δ_x n(x)
        compute E_new = E_ref(p_m)
        if E_new < E then
            set f(x) = p_m and E = E_new
        end
    end
end
```

Algorithm 1: Pseudo-code for the reconstruction algorithm.

4.5 Experiments

This section documents our experimental validation of our approach, including the implementation, and the results obtained.

4.5.1 Implementation

Our implementation uses CUDA, OpenGL, OpenCV and CLAPACK. Our images were captured using a Canon Rebel EOS XTi 400D, a 10 Mpixel digital SLR camera, and downsampled by a factor of two. For calibration we used publicly available structure-from-motion software [133]. For initial stereo estimates we used PMVS [1] and OpenCV's graph cut stereo [134].

We perform registration manually, selecting landmarks first on the template face model, then selecting the same points in the same order in the initial stereo/range data. From this an initial estimate of the similarity transform between the model space and the initial data is computed using linear least-squares.

For our wavelet we started with a base mesh of 2×3 and subdivide eight times to get the full-resolution grid mesh of 129×257. We perform the resampling of the training set using OpenGL and GLSL. Each vertex in the template mesh is stereographically projected onto a plane aligned with the front of the face. Then, each corresponding vertex in every other mesh is mapped to the same position in the plane, thus preserving correspondence. The initial stereographic projection is performed on the CPU, but the resampling of the meshes in the training set is performed on the GPU. This makes the learning part of the approach very fast. To learn from a training set of 100 faces takes only a few minutes, plus the time to first perform the GPA to align the faces.

The (inverse) wavelet transform (4.4) must be performed for every sample value of every sampled coefficient. In our current framework, we optimize the first five levels of coefficients ($P = 561 \times 3 = 1683$ model parameters) using independent parameter sampling, with $S = 50$ samples per parameter. This means the surface must be reconstructed from the model parameters 84150 times, hence the speed of the transform is crucial to the speed of the overall approach. In practice, some parameters have very small variation in the database (*e.g.*, $\sigma_x^k < 10^{-12}$) and we do not sample those parameters, so the total number of inverse wavelet transforms that must be performed is 60750 in our current setup. Each transform takes 0.202ms using our CUDA-based GPU implementation, for a total of 12.272s spent reconstructing the surface. This is compared to slightly over 1 ms per transform using a highly optimized CPU implementation. (Note that with the dimensions we are using, the entire wavelet data fits in

the CPU cache, making this virtually optimal CPU performance.) The GPU wavelet transform splits the computation into blocks that overlap by two vertices/coefficients on all sides, reads the coefficients into shared memory, and performs the lifting operations in shared memory. If the number of blocks required is less than the number of multi-processors on the GPU then the transform can be performed in-place, writing to the same global memory it reads from. One evaluation of E_{NN} takes 0.671ms, for a total of 41.550s over the entire algorithm. In total, the parameter sampling takes just over 67s. Global statistical models typically require fewer parameters (*e.g.*, 50 or 100 principle components), and are optimized using gradient-descent-based approaches, meaning they can be fitted in less time, however, this optimum is local and requires a good initial estimate. Our method leverages the properties of the wavelet decomposition to perform global search in each model parameter in an efficient way, allowing us to avoid local minima while maintaining low running times.

The refinement stage takes 0.794s for 200 iterations (per reference-matching image pair), three samples per vertex per iteration, and two images. It also breaks the computation into overlapping blocks. For each vertex in each block, we use a reference thread and a matching thread, which share the computation. One thread computes the normal, while the other computes the anisotropic average, both using the neighboring vertices in shared memory. The reference thread then samples the window in the reference image, while the matching thread samples the window in the matching image, each thread storing the samples in shared memory. (We use 3×3 windows for NCC.) The remainder of the NCC computation is divided between the two threads, and the resulting matching cost for each sample saved for computation of the per-vertex matching weight.

4.5.2 Results

Figure 4.2 shows the results of our reconstruction algorithm applied to a stereo pair, with an initial point cloud from a general stereo algorithm [1]. Despite the noise in the original point cloud, the reconstruction after parameter sampling (second from right) captures the shape of the face with some artifacts due to the independence of the shape parameters, while the post-refinement reconstruction smooths the artifacts while preserving shape detail. Note how the reconstruction captures the fact that the mouth is slightly lower on the left side than on the right side, as in the input image. This figure demonstrates the robustness of our method to noise. Figure 4.4 shows another result, this time with the initial point cloud from graph cuts [134]. It is again quite noisy, and it also exhibits fronto-parallel bias and severe outliers due to the highly non-Lambertian reflectance of the subject's glasses. Note our goal is to reconstruct the

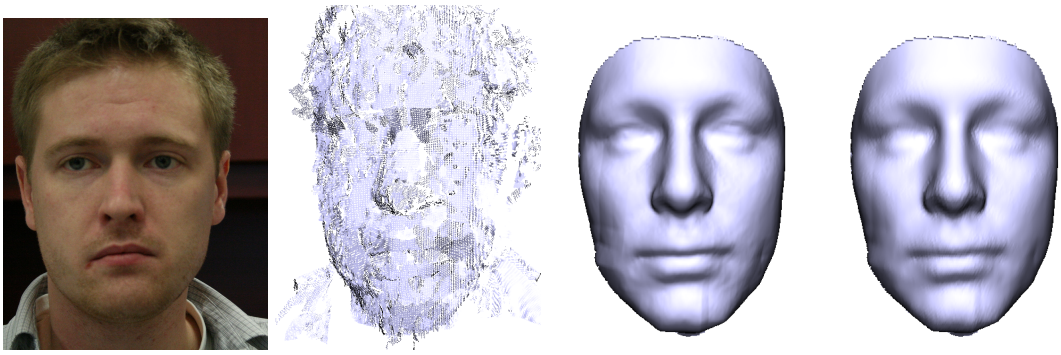

Figure 4.2: Left to right: input image, initial point cloud from PMVS [1], reconstruction before refinement (level 4), after refinement.

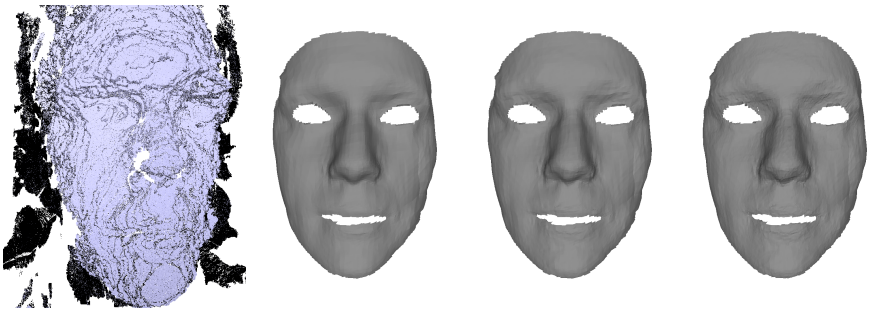

Figure 4.3: Reconstruction results for stereo pair used in Figure 4.2 with a point cloud generated using the method described in Chapter 5 (left) and the revised fitting procedure from Section 4.5.3, levels 3, 4 and 5, respectively.

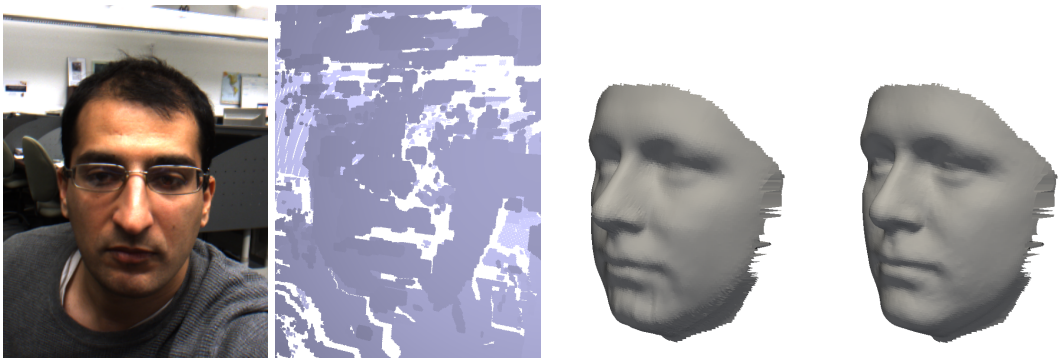

Figure 4.4: Left to right: input image, initial point cloud, reconstruction, mean face for comparison.

face geometry, not that of accessories or glasses or a hand in front of the face that might be present in some scenarios. The mean face is shown next to the reconstruction for comparison. Note how the nose is elongated and the cheek bones are more prominent in the reconstruction as in the input image. This figure demonstrates the robustness of our method to severe outliers and occlusions.

Figure 4.5 and 4.7 show the results of reconstruction by fitting the prior model to laser-scan data: faces or heads cropped from full body scans in the CAESAR database [135]. Since these point clouds are more reliable than stereo data, we increase the weight w_{NN}. Note that they are quite low resolution, however, and incur substantial blurring and some noise due to the subject shifting during the acquisition process. Note how the shape of the nose in Figure 4.5 is captured accurately without the noise that is present in the original mesh (far right). The reconstruction captures the shape of the nose and cheek bones in both cases while smoothing the surface and increasing the resolution. The noise or artifacts in the reconstructions are along the outside of the face where the prior is less reliable. Since there are no images to go with these point clouds, the refinement consists only of smoothing.

4.5.3 Additional Experiments

The results presented this far were obtained using a private database of 100 scans of Caucasian males of military age in neutral expression. This results in a learned model with relatively low variation compared to that found in the population at large. We have subsequently also experimented with training using 100 subjects in neutral expression from the BU-3DFE database [136].[2] This database covers both sexes, multiple ethnicities and different age ranges. It additionally has multiple expression in multiple levels, although we only consider neutral expression here. We limit ourselves to neutral expression to constrain the scope of this chapter. Learning a model incorporating varying expression is a logical and natural avenue for future work. This database was registered using the method by Salazar et al. [137], with the exception that ground truth landmarks were used instead of automatically located ones. Figure 4.8 shows the result of training with neutral expressions of 100 subjects from the BU-3DFE database. The first row shows the mean shape, while the subsequent rows show the magnitude of the standard deviation $\left\|\sigma_x^k\ \sigma_y^k\ \sigma_z^k\right\|$ for wavelet coefficients at levels 0 through 5 as measured in meters. Blue indicates low variation, green medium variation and red indicates high variation. The coloring is normalized for each level; in absolute terms the variation of all coefficients decreases as the level increases, as shown by the color-legends on the right. Note how the

[2]The experiments in this section were carried out in collaboration with Stefanie Wuhrer and Augusto Salazar.

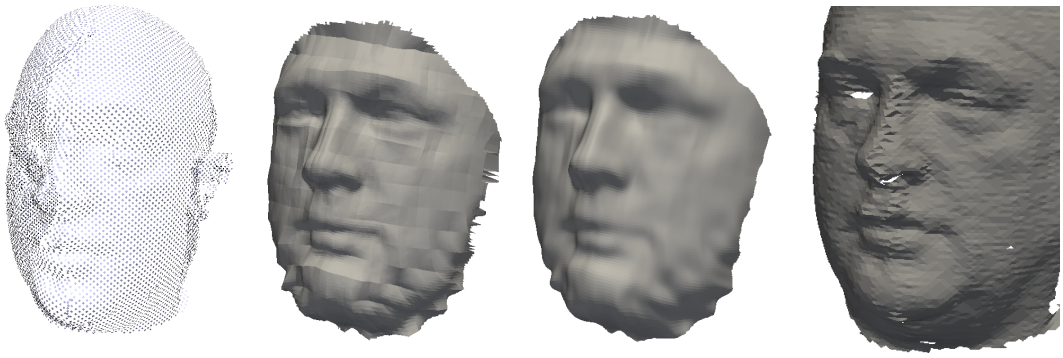

Figure 4.5: Fitting the model to a laser scan. Left to right: point cloud, reconstruction before refinement, after refinement, original mesh.

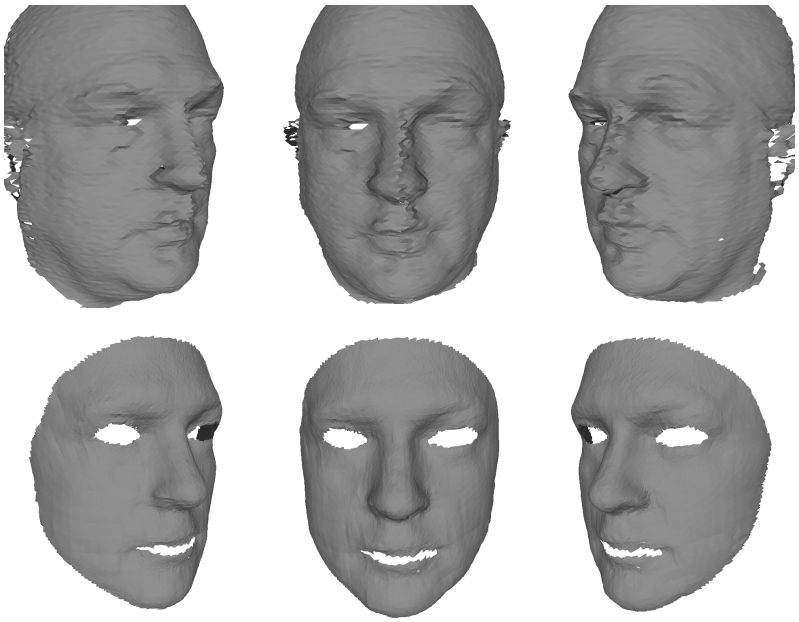

Figure 4.6: Reconstruction results (second row) for the same data as in Figure 4.5 using the training and fitting method from Section 4.5.3.

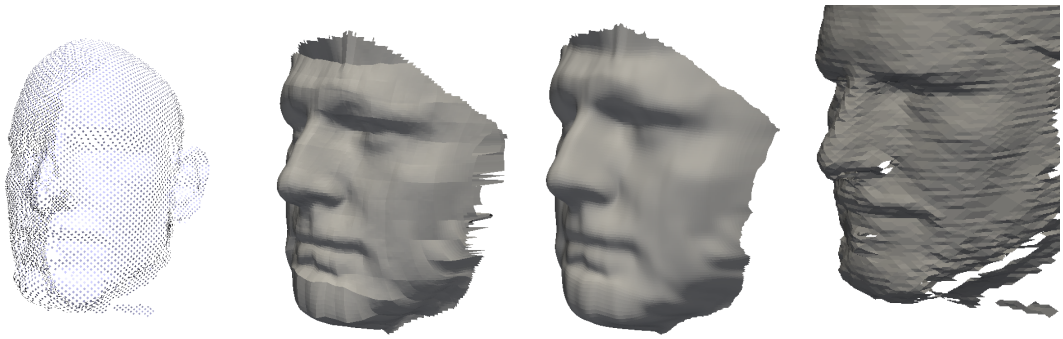

Figure 4.7: Fitting the model to a laser scan. Left to right: point cloud, reconstruction before refinement, after refinement, original mesh.

variation is concetrated more on particular facial features, rather than on the boundary as in Figure 4.1.

We also experimented with some modifications to the implementation as follows. First, one of the major sources of problems in the fitting stage was the large variation along the boundary of the face template encountered during training. This was due to the parameterization mapping from the template triangle mesh to the quad mesh not being surjective. Some quad mesh vertices around the outside of the map are assigned the position $[0\ 0\ 0]^T$. This results in step functions along the boundary, resulting in large wavelet coefficient values, and hence, disproportionately high variances of the those coefficients over the database. This effect is now mitigated by diffusing the geometry into these null-regions from the valid vertices neighboring them, in an iterative process.

Second, the Gaussian distribution learned for each wavelet coefficient is now only used to restrict the search. The Gaussian prior used previously tends to pull the shape towards the mean, which causes it to lose detail and distinctiveness [138]. Some authors [138] enforce that the sampled shape should have a fixed distance from the mean, restricting the shape parameters to a hypersphere in normalized PCA space, but this can result in unrealistic shapes (*i.e.*, non-human faces) along the directions of the principal components themselves. Here, the Gaussian prior is replaced with a so-called "box" prior:

$$P(x_r^k) = \begin{cases} 1 & |x_r^k| < \lambda\sigma_x^k \\ 0 & \text{otherwise} \end{cases} \tag{4.22}$$

and similarly for y_r^k and z_r^k, where λ is a parameter indicating how many standard deviations to search (*i.e.*, this was set $\lambda = 3$ in the preliminary version). In the following we set $\lambda = 1$. This is because the new database used for training has greater variation than the previous one.

We tested the effects of these changes using point cloud data only. We also changed the subdivision hierarchy for these experiments: we started with a base mesh of 5×7 and subdivided six times. We found empirically that each of these changes helped improve the reconstruction. We found the diffusion of the boundary vertices reduced the artifacts along the boundary. The combination of the new prior and more variable training data allowed us simultaneously avoid the bias of the Gaussian prior towards the mean shape (increasing distinctiveness) and avoid sampling very low probability parameter values far from the mean. Having more vertices at the coarsest scale allows better capture of the overall shape. These experiments included no refinement stage. For range data, this means no smoothing, since there are no images to perform image matching anyway. Note that an additional benefit of using the new prior without refinement is that the only parameters required for the method are λ and τ_{NN}. We set λ as described above and set $\tau_{NN} = 1.0$cm. Figure 4.9 shows the reconstruction results after fitting coefficients at levels 0 (first row) through 5 (second to last row) to a scan from the Bosphorus database [139] where the subject is wearing glasses, causing severe outliers in the original point cloud (last row). This figure demonstrates the robustness of our method to both occlusions (*i.e.*, the frames of the glasses) and severe outliers. Figures 4.6 and 4.10 show our model fitted to faces or heads cropped from the CAESAR full body laser scan database [135]. Note that these scans are much lower resolution than either the model or the Bosphorus data, and as a result the nearest neighbor energy is less reliable. Nonetheless we capture both the overall shape and shape details. Figure 4.6 shows how the modified method with new training performs on the same data as in Figure 4.5. While perhaps less detail is reconstructed, the overall shape is more accurate and no artifacts are introduced on the boundary. Figure 4.10 also captures the nose quite well. More detail could be recovered by increasing the value of λ. Figure 4.3 shows a point cloud generated for the stereo pair from Figure 4.2 using the multi-scale stereo method described in Chapter 5 and the model fitted using this new fitting procedure (without refinement). The point cloud from the stereo method is shown on the left, followed by the fitted model to level 3, 4 and 5. Compared with Figure 4.2, we see that significantly more distinctiveness is preserved. For stereo data, which is noisier than laser range scans, reconstructing to only 4 or 5 levels, instead of all 6, may be preferable.

4.6 Discussion

We have presented a method for fast and robust model-based stereo using a statistical wavelet shape prior. We have demonstrated this approach for human faces using both stereo and laser-scan data. The most practically important future work would be to automate the registration of

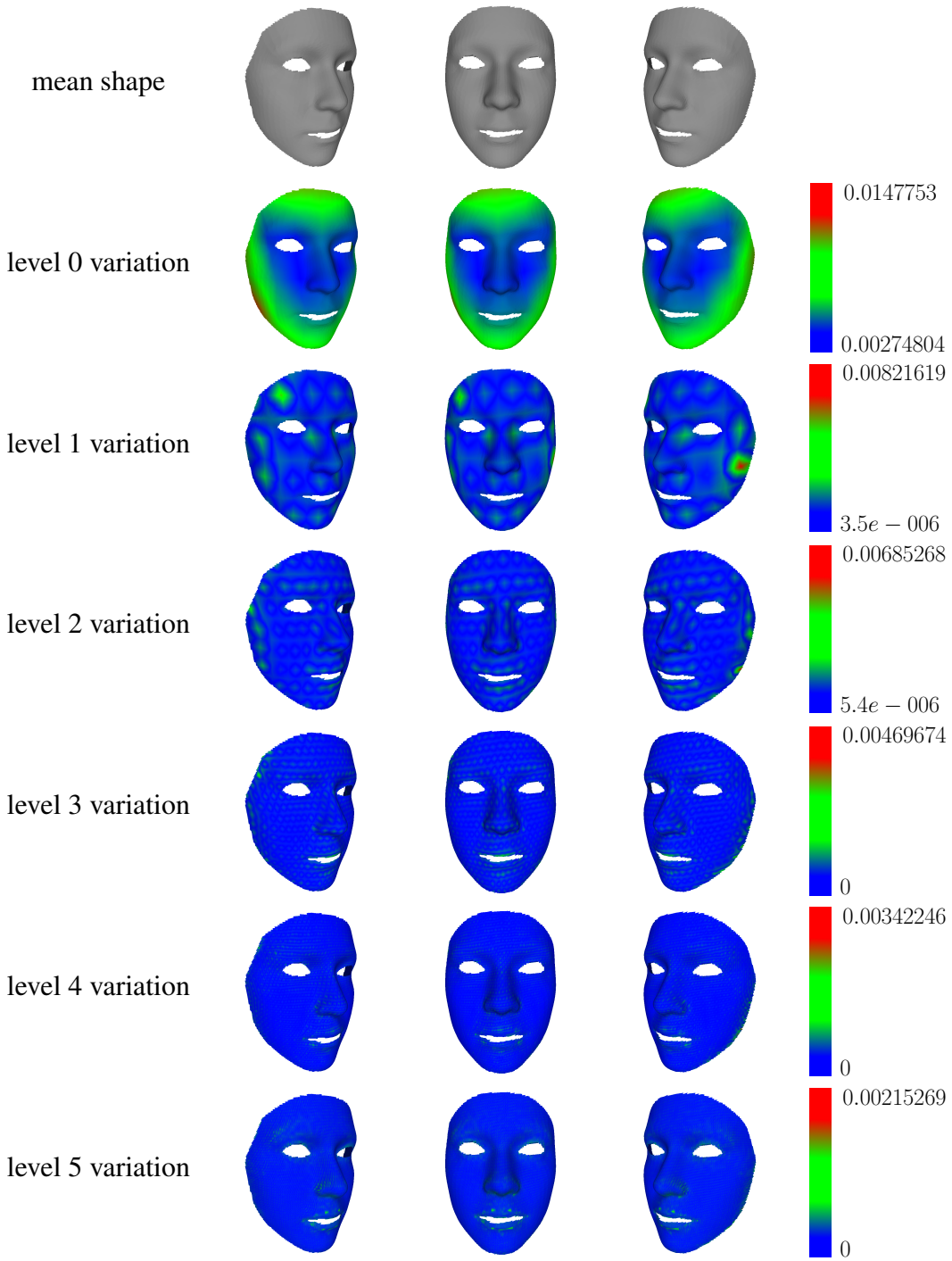

Figure 4.8: Mean shape and variations of different levels of wavelet coefficients from the BU-3DFE database (neutral expression). Variation ranges are given in meters.

Figure 4.9: Reconstruction results (rows 1 through 6) for range data of a subject wearing glasses (final row).

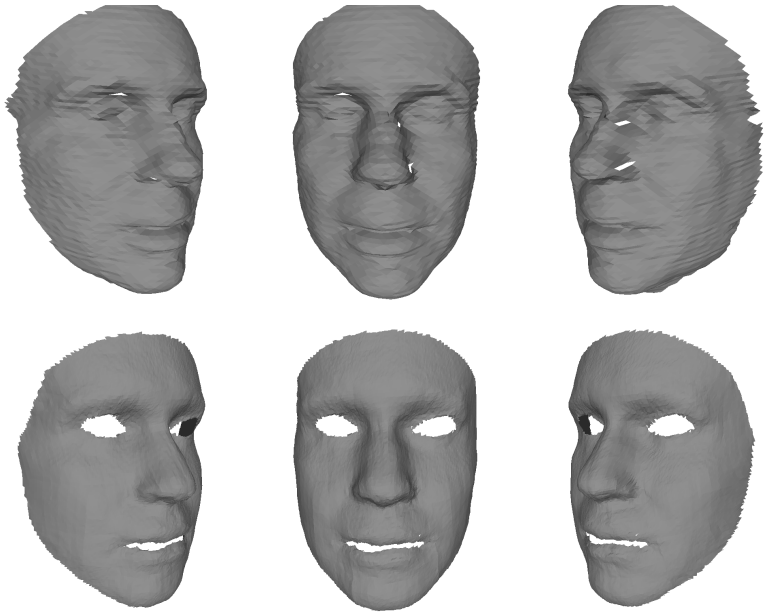

Figure 4.10: Reconstruction results (second row) for a face cropped from the CAESAR full body scan database.

the input point cloud with the template model. This is currently done with user-specified landmarks, but automatic extraction of landmarks is possible, and would lead to a fully automatic approach. Another interesting direction for future work is to use a model that incorporates expression variation and train this model using a database with expression variation, such as the full BU-3DFE database [136], and learn both shape and expression parameters in the wavelet domain. One could use this framework for stereo tracking. If the model sufficiently separated face shape and expression, the entire sequence could be used to regularize the shape parameters while a temporal smoothing term could regularize the change in expression over time. Here, we have only considered a statistical model of shape, but it would also be interesting to incorporate a texture model as in a localized, wavelet domain, version of a full morphable model.

It should be noted that when a training set is not available for a given shape, or when we wish to reconstruct general surfaces, we cannot construct a statistical shape prior and we must resort to different models. In Chapter 6 we explore another method for using the wavelet domain to regularize disparity maps, in this case with adaptive smoothing.

Chapter 5

Efficient Multi-Scale Stereo of High-Resolution Planar and Spherical Images

In the previous chapters we have examined multi-scale representations of spherical images and spherical wavelet representations of surfaces. In this chapter[1] we look at multi-scale *algorithms* for disparity estimation from both planar and spherical stereo pairs. Specifically, we look at efficient ways to incorporate matching information aggregated over different scales without making hard, greedy decisions at coarse scales that prevent us from finding the correct fine-scale disparity later.

A version of this chapter has been accepted to and will appear at the International Conference on 3D Imaging and Modeling, Processing, Visualization and Transmission (3DIMPVT) 2012 [140].

5.1 Introduction

In this chapter we demonstrate the effectiveness and efficiency of using distance transforms for the purposes of multi-scale stereo matching of high-resolution images. Given a set of disparity space images (DSI) at multiple scales, our multi-scale framework combines them efficiently without making greedy decisions at each level by computing a per-pixel distance transform of the DSI at each scale. This allows us to find the disparity at the finest scale for which the combined cost of all scales, and differences between disparities estimated at adjacent scales, is

[1]Content of this chapter which overlaps with prior publication [140] is copyright IEEE 2012.

minimized. This approach encourages consistent matching at multiple scales while allowing fine-scale detail to be retained. We also show how the minima of the distance transform can be used to reduce the storage requirements for multi-scale DSI. We extend our approach to spherical stereo with a novel formulation of disparity for spherical images.

Often multi-scale methods are used to allow for a large search range by reducing the image resolution so that the same search range at coarser scales covers a larger relative portion of the image. In such a scheme, the correspondence result at coarse levels is used as a start point or mid-point to constrain the search at the next finer level. This type of approach is often very effective for smooth surfaces such as human faces [72, 83]. While this makes it more tractable to search large correspondence domains (disparity ranges), it means making hard, greedy decisions at coarser scales, which restrict possible solutions at finer scales. Thus if an incorrect match is chosen at a coarse-resolution, it may be impossible to recover from that mistake at finer resolutions.

In our case, we do not use multiple scales to reduce the search range, but rather to aggregate matching information over larger portions of the image domain by reducing the number of pixels at which the DSI is computed. We then combine inter-scale matching information to get the full resolution disparity estimate. Thus, we do not rule out disparity candidates at finer scales as a result of poor matching scores at coarser scales. Instead, we minimize an energy function at each pixel that balances consistency between adjacent scales and the matching cost computed at each scale. A similar approach has been previously formulated as a path-finding problem within a matching hierarchy [141], and solved using optimal path-finding via Dijkstra's algorithm [142], however our distance transform approach is more efficient in terms of both time and space requirements.

In this chapter, we make the following contributions: a time- and space-efficient multi-scale matching method for stereo correspondence based on a novel use of distance transforms that preserves fine-scale detail by avoiding greedy decisions at coarse scales; and a novel spherical disparity formulation which allows us to extend our method to spherical images.

Here, we limit our discussion to the context of planar and spherical binocular stereo matching. However, we emphasize that our method is designed to be easily extended to multi-view reconstruction for both planar and spherical cases, as done in Chapter 6, and we note that the methods proposed here can also be applied to two-dimensional motion estimation.

5.1.1 Related Work

Multi-scale image matching has been widely used for both stereo and motion estimation (optical flow). Providing a full survey of such methods is beyond the scope of this chapter and instead, we review methods we consider representative of the different variations. Some coarse-to-fine matching methods work essentially in two-stages [68, 69]. First, a coarse model is estimated, based on *e.g.*, segmented shape [68] or reliably-matched feature or support points [69], then this is used to restrict a per-pixel disparity or depth search. Geiger et al. [69] propose a generative disparity model using the planar disparity interpolation of triangulated support points. Other methods use image pyramids [70, 71, 72] and iteratively refine disparity estimates in a multi-scale coarse-to-fine way, with the coarse estimates used as initial values for the fine-scale estimates. Hirschmüller [71] uses mutual information between the images, with one image warped using the current disparity estimate, as a pixel-wise matching criterion within such a framework. A random disparity map is used to initialize the coarsest level of the pyramid, which is then iteratively updated to increase mutual information at each pixel. The disparity map at the coarser levels is inherited by the finer levels and refined to increase mutual information. Other approaches to multi-scale matching only reduce the number of pixels at which costs are computed, not the resolution of the images; an example is hierarchical belief propagation (HBP) [75], where dyadic downsampling of the disparity grid allows message passing iterations to aggregate information over larger portions of the image domain. This better preserves small differences in disparity between nearby pixels. Our approach has the most in common with this type of multi-scale method. However, while HBP directly inherits message values from coarse scales to initialize fine scale iteration, our method is non-iterative and explicitly enforces consistent disparity values between consecutive scales.

Previous approaches have also been proposed to perform stereo matching on spherical images [24, 104, 25, 26, 105, 27]. Shi [24] performs real-time stereo by window-based matching in angular disparity. Kim and Hilton [25, 26] use the same disparity formulation with a PDE-based method for scene modeling. Pagani et al. [27] reconstruct dense point-clouds from high-resolution spherical images using a spherical adaptation of PMVS [1]. Bagnato et al. [105] develop a variational approach for structure-from-motion from spherical images. While our formulation shares some commonalities with previous ones, we make use of two disparity quantities that allow us to both sample efficiently along epipolar arcs, and apply cost filtering and smoothing in a geometrically correct way. We describe in detail these differences in Section 5.3. Further, our stereo algorithm is different from all of these approaches.

5.2 Multi-Scale Stereo Matching

5.2.1 Multi-Scale DSI Chaining with Distance Transforms

In contrast to previous approaches, we do not use multiple scales to reduce the search space for matching, but rather to aggregate information over larger scales to improve the quality of matching. Particularly, if fine scale features are smoothed over at coarser scales, than this lack of texture leads to matching uncertainty for many pixels at coarser scales. Hence, making hard decisions at coarser scales that restrict the search space for finer scales can irrevocably preclude finding the correct match. This is especially problematic for narrow foreground objects.

A distance transform of a function $f : \Omega \to \mathbb{R}_{\geq 0}$ for some domain Ω, is given by [143]

$$\mathcal{D}_f(x) = \min_y \left(f(y) + \text{dist}(x, y) \right) \tag{5.1}$$

where $x, y \in \Omega$ and $\text{dist}(x, y)$ is a distance measure between x and y. In our stereo matching application, $\Omega \subset \mathbb{R}_{\geq 0}$ is the disparity range, x and y represent disparity values, and f is a cost volume or disparity space image (DSI) storing a dissimilarity value or cost for every disparity value at every pixel. It is known [143, 75] that if the distance function is of the form $\text{dist}(x, y) = g(y - x)$, (5.1) can be computed for all x in time linear in the number of samples used to parameterize Ω (i.e. the number of values of x). It has been further shown [144] that $\mathcal{D}_f$ can be losslessly represented with only its minima and the parameters of the distance function g. In this chapter, we leverage these insights to efficiently, in terms of both time and space, minimize a per-pixel energy function that balances consistent disparity estimates at different scales and the DSI values at individual scales.

Our energy function at each pixel is of the form

$$E\left(\{d_{ij}^0, d_{ij}^1, \ldots, d_{ij}^L\}\right) = \sum_{l=0}^{L} D_{ij}^l(d_{ij}^l) + \sum_{l=1}^{L} g(d_{ij}^l - d_{ij}^{l-1}) \tag{5.2}$$

where i and j are reference and matching images (in this chapter $i, j \in \{0, 1\}$), l denotes the scale or level, D_{ij}^l is the DSI at level l, giving a cost for each disparity value, and g is a cost function penalizing differences between disparity estimates at consecutive resolutions or scales. Here, we use $g(x) = \lambda \min(|x|, \tau_d)$, where λ is a scaling and τ_d a truncation parameter. The choice of a truncated L_1-metric is based on its robustness to outliers, which in our case correspond to situations where the coarse-scale DSI has smoothed over some edge that is detected in the finer-scale DSI, and the simplicity with which the distance transform can be computed using this distance. This is a commonly used cost function for, *e.g.*, differences in disparity between neighboring pixels in pairwise Markov models [75].

The DSI at scale l is a function $D^l_{ij} : \Omega^l_I \times \Omega \rightarrow \mathbb{R}_{\geq 0}$ mapping a level-l pixel and a disparity value to a cost value. The domain Ω^l_I denotes the subset of the image grid used at level l, where $\Omega^L_I = \Omega_I \subset \mathbb{R}^2$ is the full-resolution image grid, and Ω is the disparity range as noted above. Thus, $D^l_{ij}(\mathbf{u}, d)$ gives the cost of matching pixel $\mathbf{u}$ in image I_i to pixel $\mathbf{u} - [d\ 0]^T$ in image I_j at scale l. In (5.2) we have dropped the pixel parameter from the notation for conciseness since this energy is evaluated per-pixel. The disparity grids at coarser levels are, naturally, subsampled from the finer scales. That is, $\Omega^l_I \subset \Omega^{l+1}_I$. Thus, we calculate coarser scales not by reducing the size of the input images but instead by sampling the DSI at only a subset of the pixels, and downsample (Gaussian filter then subsample) the guide image used by the underlying stereo-matcher described in the next section. This means that we reduce computational complexity at coarser scales, and that information is aggregated over larger distances with the same aggregation window size. This provides an important regularization, without the need to explicitly smooth disparity values at this stage. Images with large aspect ratios (*e.g.*, those in the KITTI benchmark [2]), where one image dimension is much larger than the other, can have problems with uniform downsampling. The square windows used to filter the DSI cause information to be aggregated over a much larger portion of the image domain in one dimension than in the other. We counteract this using an adaptive downsampling, that adjusts the level of downsampling in each dimension according to the aspect ratio of the image. Let the input image dimensions be $W \times H$ and the dimensions of the level l grid be $W^l \times H^l$. Thus $\Omega^L_I = \{0, \ldots, W-1\} \times \{0, \ldots, H-1\}$. Without loss of generality, suppose $W^{l+1} \geq H^{l+1}$, and let the aspect ratio be $a^{l+1} = W^{l+1}/H^{l+1}$. The dimensions of the level l grid will be $W^l = W^{l+1}/2$ and $H^l = H^{l+1}/\max(2/\left\lfloor a^{l+1} \right\rfloor, 1)$. Thus, if $a^{l+1} < 2$ then we have the standard dyadic subsampling, and otherwise only the larger dimension is subsampled. Let the step sizes of level l be $s^l_x = W/W^l$ and $s^l_y = H/H^l$. The level l disparity grid is $\Omega^l_I = \{s^l_x k \ : \ k = 0, \ldots, W^l - 1\} \times \{s^l_y k \ : \ k = 0, \ldots, H^l - 1\}$. Importantly, by not downsampling the input images we retain the full texture information when computing the DSI at the downsampled disparity grid.

The energy (5.2) can be rewritten as

$$E\left(\left\{d^0_{ij}, d^1_{ij}, \ldots, d^L_{ij}\right\}\right) = D^0_{ij}(d^0_{ij}) + \sum_{l=1}^{L} \left(D^l_{ij}(d^l_{ij}) + g(d^l_{ij} - d^{l-1}_{ij})\right) \tag{5.3}$$

where we take the highest-resolution disparity value d^L_{ij} as the estimate for the given pixel. This is the disparity

$$\hat{d}^L_{ij} = \arg\min_d E^L_{ij}(d) \tag{5.4}$$

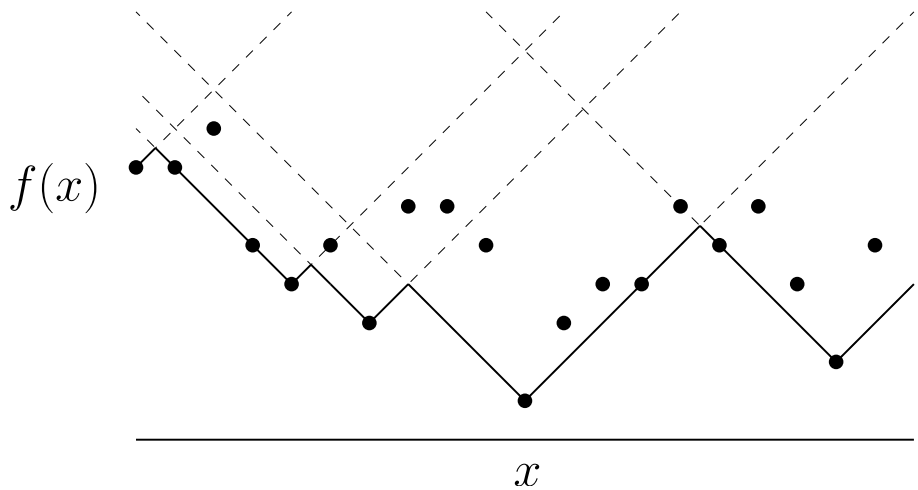

Figure 5.1: The distance transform (solid line) of a sampled function (points). The distance from the envelope points is shown dashed.

where

$$E_{ij}^{l}(d) = \min_{d'} \left(D_{ij}^{l}(d') + E_{ij}^{l-1}(d') + g(d' - d) \right) \tag{5.5}$$

which we observe has the same form as (5.1). We define $E_{ij}^{-1}(d) = 0 \; \forall d$. Thus by applying a distance transform to the DSI at each successive resolution (starting at the coarsest), and using a winner-take-all minimization, we end up with the highest-resolution disparity for which the energy function (5.2) is minimized. Additionally, by incorporating the envelope point transform (EPT) [144], and only storing the envelope points, or local minima of the distance transform, we greatly reduce the storage requirements. This is key for high-resolution images where loading multiple DSI into memory would not be practical even on a modern desktop or workstation. Note that while this is a lossless representation of the *distance transform* of the DSI, some information of the original DSI is lost due to the smoothing enforced by the distance transform. Since we are typically only interested in the minima of the DSI, this is not a problem. We fix the finest (full-resolution) scale at $L = 3$ in all examples in this chapter, although in the future it would be interesting to determine the optimal number scales automatically based on the size of the images and the amount of detail therein. We set τ_d to 5% of the disparity range.

The concepts of a 1D distance transform and the associated envelope points are shown in Figure 5.1. The samples of a function $f(x)$ are shown as dots, whereas the distance transform $\mathcal{D}_f(x)$, with distance function $g(x) = |x|$ is shown as a solid line. The dashed lines show $f(x_e) + g(x - x_e)$ for envelope points x_e. Using a truncated distance function would simply

result in fewer envelope points, and a distance transform with flat regions.

5.2.2 Initial DSI Computation

For single-scale matching we base our approach on the cost-volume filtering of Rhemann et al. [145]. We choose this method to compute a DSI at each scale because of its combined efficiency and accuracy in textured areas, although this method could be substituted with another cost-aggregation method possessing these properties. Our multi-scale framework provides robustness to low-texture and occlusions that typically give local methods trouble. The method of Rhemann et al. initially computes a cost volume by evaluating single-pixel dissimilarities for all pixels and disparities, and then filters each disparity-slice of this cost volume independently using the guided image filter [146], using the reference image as the guidance image.

When computing a pixel-wise dissimilarity measure between reference image I_i and matching image I_j, we first radiometrically equalize the images, by subtracting the mean-filtered images $\bar{I}_i$ and $\bar{I}_j$ from the original images: $\hat{I}_i(\mathbf{u}) = I_i(\mathbf{u}) - \bar{I}_i(\mathbf{u})$ and $\hat{I}_j(\mathbf{u}) = I_j(\mathbf{u}) - \bar{I}_j(\mathbf{u})$. This process is described in more detail below. We then compute the DSI as

$$\begin{aligned} D_{ij}(\mathbf{u}_i, d) = (1-\eta)\min(&\left|\hat{I}_i(\mathbf{u}_i) - \hat{I}_j(\mathbf{u}_j)\right|_1, \tau_1) \\ +\eta\min(&\left|\|\nabla I_i(\mathbf{u}_i)\|_2 - \|\nabla I_j(\mathbf{u}_j)\|_2\right|_1, \tau_2) \end{aligned} \tag{5.6}$$

where $\mathbf{u}_i$ is a pixel location in image I_i, $\mathbf{u}_j$ is the corresponding location in I_j mapped to by disparity d, $\mathbf{u}_j = \mathbf{u}_i - [d\ 0]^T$ in the rectified planar case, $\eta \in [0, 1]$ is a parameter balancing the difference of pixel colors and gradient magnitudes, and τ_1 and τ_2 are truncation thresholds. We use $\eta = 0.9$, $\tau_1 = 0.028$, $\tau_2 = 0.008$, as proposed by Rhemann et al. These values are set assuming that fixed-precision images are interpreted to have color-intensity values in $[0, 1]$.

This dissimilarity measure is then filtered in the reference image domain independently for each value of d using guided image filtering [146]. The guided filter weights are given by

$$W_{\mathbf{uv}}(I) = \frac{1}{|\omega|^2} \sum_{\mathbf{s}:\mathbf{u},\mathbf{v}\in\omega_\mathbf{s}} \left(1 + \frac{(I(\mathbf{u}) - \mu_\mathbf{s})(I(\mathbf{v}) - \mu_\mathbf{s})}{\sigma_\mathbf{s}^2 + \epsilon_{GF}}\right) \tag{5.7}$$

where I denotes the guide image (the reference image in our case), $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{s}$ denote pixel locations, $\omega_\mathbf{s}$ denotes the $(2r+1) \times (2r+1)$ window centered on $\mathbf{s}$, $\mu_\mathbf{s}$ and $\sigma_\mathbf{s}$ denote the mean and variance of the guide image within $\omega_\mathbf{s}$, and $|\omega|$ denotes the number of pixels in a window. We then filter the disparity map, $D_{ij}(\mathbf{u}_i, d) \leftarrow \sum_\mathbf{v} W_{\mathbf{u}_i\mathbf{v}}(I_i) D_{ij}(\mathbf{v}, d)$. We use a window radius $r = 9$ and variance-control parameter $\epsilon_{GF} = 0.01^2$ for the guided filter. The guided filter has similar behavior, in terms of weights resulting from a particular image, as the bilateral

Figure 5.2: A stereo pair from the KITTI dataset before (top) and after (bottom) radiometric equalization. Note the change in brightness of the road prior to equalization.

filter [147], but it can be computed exactly as a series of box filters, and hence in $O(1)$ time per pixel. For a full discussion of the properties of the guided filter we refer the reader to He et al. [146].

The use of the mean-subtracted images helps account for specular surfaces such as road surfaces, cars and tile floors. This subtracts the local average intensity of the pixel values, where the mean is computed over a square window. We use a window "radius" of $2r$, so the window is $(4r+1)\times(4r+1)$. Thus, if a texture pattern is more or less preserved in both images, but in one image the specular reflectance of the surface material cause all intensity values to be increased by an approximately constant amount, subtracting the local mean will give much better matching results than comparing the original pixel intensities. The rank filter [148] or mutual information [71], could also be used to deal with general and drastic radiometric differences such as illumination and exposure changes. However, since we assume either a static scene *or* synchronously captured images, we by definition assume static illumination, and the simple mean-filtering suffices. We re-normalize $\hat{I}_i$ and $\hat{I}_j$ to have values in the range $[0, 1]$. Figure 5.2 demonstrates how this equalizes the images using a stereo pair from the KITTI dataset. Note how the difference in brightness of the road between left and right images is removed.

5.2.3 Time and Space Complexity

Because we divide the number of pixels at which we sample the disparity space at each resolution, our overall complexity remains linear in the number of pixels in the full-resolution image, regardless of the number of scales we use. The computation time of the distance transform applied to each DSI is linear in the number of pixels and the number of disparity values. This

is also the case for the DSI sampling and filtering. That is our computation time is $O(NM)$, where N is the number of pixels and M is the number of disparity values sampled. We set $M = 128$ unless otherwise indicated.

The storage requirements depend on the number of pixels and the number of envelope points. Let $\bar{\mathcal{E}}$ denote the average number of envelope points over the pixels of a DSI, then the total storage complexity of our method is $O(\bar{\mathcal{E}}N)$ where N is the number of pixels. During computation, we only need to store $O(1)$ cost values per pixel in addition to updating the list of envelope points.

5.2.4 Disparity Post-Processing

Although the matching algorithm presented above aggregates the DSI over large neighborhoods at multiple scales, the matching is done per-pixel in a winner-take-all fashion, without enforcing any inter-pixel constraints on the disparity estimates. This section describes post-processing steps applied to the disparity estimates.

Fusion: Disparity is estimated for both views and the disparity maps are fused to remove outliers and enforce the uniqueness constraint. In a multi-view setting with many disparity maps, we would like to use a multi-view fusion method [99, 100], as we do in Chapter 6, but in the two-frame setting we use the following method to efficiently distinguish between occlusions and mismatches. We warp the matching image to the reference image viewpoint. Areas occluded in the matching image will have no value in this warped disparity map, and we set them to 0. In the rest of the disparity map, we compare the reference disparity to the warped disparity. If they agree, we average them and output this disparity. Where they do not agree we mark them as mismatches and again set the disparity to 0. Agreement is determined by testing $|d_i(u) - d_i^j(u)| < \epsilon_i(u)$ where d_i is the disparity map of the reference image I_i, d_i^j is the warped disparity map and $\epsilon_i(u) = \epsilon \max(d_i(u), \Delta d)$ is a relative threshold, where Δd is the disparity step size. We use $\epsilon = 0.05$.

We discard disparity regions of less than 50 pixels as recommended by Hirschmüller [71]. We use the remaining non-zero disparities as sites and compute a Voronoi tessellation of the disparity map. We assign the disparity at each Voronoi site to each pixel within its Voronoi cell. This will assign foreground disparities to some background pixels, so for each pixel designated as occluded we find the nearest smaller disparity along the epipolar line. Mismatched pixels are iteratively set to the average of their 8-neighborhood, which results in an approximately planar interpolation of the Voronoi centers. We denote the fused disparity map $d_i^{(f)}$. This disparity is subsequently filtered to fit the edge information of the reference image.

Filter: To smooth the disparity map output from the fusion stage, we apply a median filter using a histogram approach. Median filters are a common post-processing technique in stereo to smooth the disparity in a way that reduces the influence of outliers. The following approach lets us calculate the median filter in time per pixel independent of the filter size and incorporate edge information. We set the number of histogram bins to $M/2$. For each bin b, we test if each pixel's disparity is less than the upper-bound of that bin, $d^{(f)}(u) < h_b$, giving a binary image. Applying a box filter to this binary image would then give the number of pixels $c_b(u)$ within a given window of each pixel that are less than h_b. Thus we can find the median value within the window by finding the first bin where $c_b(u)$ is at least half the number of pixels in the window. We account for edge information in the input image by applying the guided filter, instead of a box filter, to each bin's binary image, with the same parameters as we use for the DSI. Denote the bin containing the weighted median value $m(u)$ with upper bound $h_{m(u)}$, and let the filtered binary image be $c_{m(u)}$, the "count" for that bin (actually a weighted average). We avoid the additional quantization effect of using $M/2$ bins by computing the weighted sum $d^{(m)}(u) = (w_{m(u)}h_{m(u)} + w_{m(u)-1}h_{m(u)-1})/(w_{m(u)} + w_{m(u)-1})$ where $w_{m(u)} = 0.5 - c_{m(u)-1}$ and $w_{m(u)-1} = c_{m(u)} - 0.5$. This is our final disparity result.

5.3 Spherical Stereo

We now show how to extend our method to omnidirectional images, using a novel spherical disparity model. The low time- and space-complexity of our method makes it ideal for typically high-resolution omnidirectional imagery. With some work, we extend cost volume filter to spherical images, and because our multi-scale scheme operates independently for each pixel, it can be applied to any sampling scheme for any image domain. The fusion and filtering stages of our pipeline are also extended to spherical images.

5.3.1 Spherical Disparity

Consider the disparity space of a pair of calibrated spherical images, the reference image I_i and the matching image I_j. Calibrated spherical images give rise to an epipolar constraint that is analogous to the epipolar constraint between calibrated planar images. Instead of constraining the point in the matching image to a line, it is constrained to lie on the great circle created by intersecting the sphere of the matching panorama with the epipolar plane. In the following we denote pixel positions as in $\hat{\mathbf{u}}$ to indicate that they are directions, *i.e.*, points on the unit sphere. The pixel in I_i with direction $\hat{\mathbf{u}}_i$ and the epipole $\mathbf{e}_{ij}$ define a plane, with normal $\hat{\mathbf{n}} = \mathbf{e}_{ij} \times \hat{\mathbf{u}}_i$,

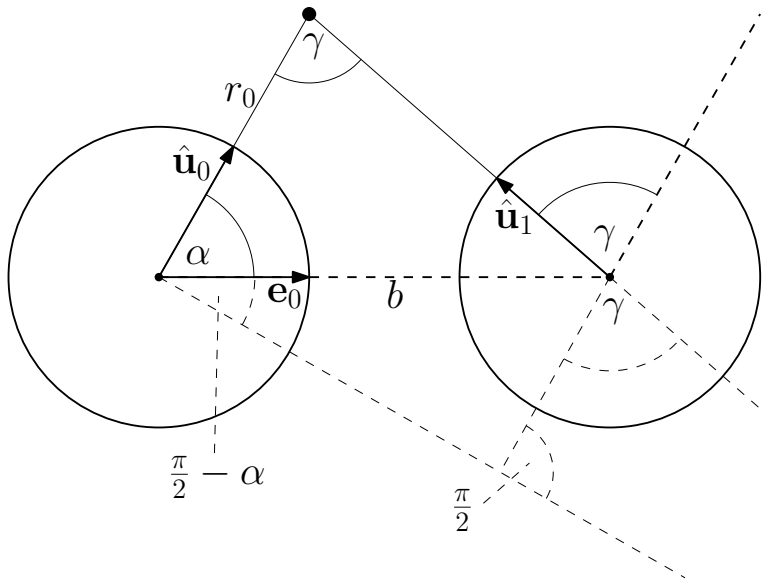

Figure 5.3: Epipolar geometry of spherical images.

in which the corresponding direction $\hat{\mathbf{u}}_j$ in I_j must lie. We simplify our formulation by using our calibration to pre-rotate our images into the same coordinate frame, so that we only need to consider the translations between them.

We model the disparity space of two spherical images using three different, but closely related, notions of disparity. These notions are illustrated in Figure 5.3, which shows the triangulation of a 3D point in the epipolar plane of two spherical images. The first we call *angular disparity*, denoted by γ, by which we mean the rotation in the epipolar plane of a pixel in the reference image to the corresponding point in matching image. That is, the rotation of $\hat{\mathbf{u}}_i$ to $\hat{\mathbf{u}}_j$ about $\hat{\mathbf{n}}$. The other two quantities, *radial disparity* and *normalized radial disparity*, are proportional to one another and denoted d and $\hat{d}$, respectively. If we denote by r_i the distance from the center-of-projection of I_i to a point in the scene along the ray of $\hat{\mathbf{u}}_i$, then $d = b/r_i$ and $\hat{d} = d/b = 1/r_i$, where b is the baseline between the two images.

Angular disparity and radial disparity can be related to each other quite simply. Specifically, by examining Figure 5.3, we can derive

$$\tan\gamma = \frac{d\sin\alpha}{1 - d\cos\alpha} \tag{5.8}$$

where α is the angle between $\hat{\mathbf{u}}_i$ and $\mathbf{e}_{ij}$. Then, the sampling location in I_j can be computed efficiently as

$$\hat{\mathbf{u}}_j = \hat{\mathbf{u}}_i \cos\gamma + \hat{\mathbf{v}} \sin\gamma$$

where $\hat{\mathbf{v}} = \hat{\mathbf{n}} \times \hat{\mathbf{u}}_i$. This allows us to map a pixel direction in the reference image to a pixel direction in the matching image without having to reconstruct the scene point and reproject it. This can be seen from the observation that a line of length r_0 as the adjacent side to an angle γ at a point in the scene forms a right-angle triangle with opposite side of length (dashed lines in Figure 5.3)

$$s = b\cos(\pi/2 - \alpha) + \frac{b\sin(\pi/2 - \alpha)}{\tan(\pi/2 - \gamma)}$$

from which, using the trigonometric identities $\cos(\pi/2 - \theta) = \sin(\theta)$, $\sin(\pi/2 - \theta) = \cos(\theta)$ and $\tan(\pi/2 - \theta) = 1/\tan(\theta)$ we get

$$s = b\sin\alpha + b\cos\alpha\ \tan\gamma$$

and hence

$$\tan\gamma = \frac{s - b\sin\alpha}{b\cos\alpha}$$

and since we know $\tan\gamma = s/r_0$ and $s = r_0\tan\gamma$, we get

$$\tan\gamma = \frac{r_0\tan\gamma - b\sin\alpha}{b\cos\alpha} = \frac{r(\tan\gamma - d\sin\alpha)}{b\cos\alpha} = \frac{\tan\gamma - d\sin\alpha}{d\cos\alpha}$$

$$d\cos\alpha\tan\gamma = \tan\gamma - d\sin\alpha\ \rightarrow\ d\sin\alpha = (1 - d\cos\alpha)\tan\gamma$$

and dividing by $(1 - d\cos\alpha)$ we get (5.8). These equations assume the two panoramas are rotationally aligned (they have the same underlying parameterization), but to relax this assumption to the case where we know the relative rotation simply requires us to apply a 3×3 rotation matrix to $\hat{\mathbf{u}}_i$ and $\hat{\mathbf{v}}$.

This formulation has a number of nice properties. Because it simply maps one point on the unit sphere to another point on the unit sphere, the disparity space can be sampled arbitrarily densely and it can be used for any underlying sampling of the sphere. Because γ remains the same if we swap the reference and matching images, and $\hat{d}$ has the same geometric meaning regardless of b, these two quantities can be used to efficiently compare disparity estimates to cross-check for occlusions, combine multiple DSI for the same reference image, and perform multi-view visibility-based fusion of disparity maps. To simply map $\hat{\mathbf{u}}_i$ to $\hat{\mathbf{u}}_j$ does not require to explicitly compute γ or its sine and cosine, but merely to use $\hat{\mathbf{u}}_i$ and $\hat{\mathbf{v}}$ as an orthonormal basis for the epipolar plane and to normalize the vector $[1 - d\cos\alpha, d\sin\alpha]^T$.

Note that, although similar, this disparity model has some important distinctions from previous spherical disparity models. Previous works [24, 25, 26] have used the singularities of the angular disparity at the epipoles to produce a spherical rectification sampling, by using a latitude-longitude sampling with the poles located at the epipoles and dual-epipoles ($-\mathbf{e}$). This

has the advantage of being able to perform scanline searching as in the case of planar binocular stereo, since a great circle of constant longitude lies in a single epipolar plane. However, it requires the images to be resampled for every pair of images, and it means that the images are most densely sampled (and disparity is most densely estimated) in directions with the least amount of parallax. It also means sampling in angular disparity, which does not have the same geometric meaning (in terms of distance from the center of projection), for different pixels. By sampling our images using the RD-map scheme [55], as described in Section 2.2.4, we achieve nearly uniform sampling of the sphere and the disparity space. For these reasons, our approach of sampling in normalized radial disparity then mapping to angular disparity to sample the matching image is much better suited to multi-view spherical disparity. Our approach also works for any underlying sampling of the spherical images. In practice, we found that the singularities at $\mathbf{e}$ and $-\mathbf{e}$, where there is no parallax between images, did not create artifacts any more prominent than those that typically arise in stereo matching due to occlusions, specularity, or texture-less regions. Note that in Figures 5.8 and 5.7, one cannot infer the location of the epipoles from any increased density of mismatches.

Let us here make an additional note about how we can refine calibration information using this disparity formulation. Suppose we can only get an estimate of the rotation between panoramas and not the baseline. By initially setting $b = 1$ for all image pairs, so $\hat{d} = d$, and sampling $d \in [0, 1/\kappa)$, where κ denotes the ratio of the estimated minimum depth in the scene to one of the baselines, we can then use the above formulation to estimate the ratio between two baselines with respect to the same reference image. Given disparity maps d_{ij} and d_{ik}, where i, j and k are the indices of the reference image I_i and two different matching images I_j and I_k, we can estimate the ratio between the baseline b_{ij} between I_i and I_j, and the baseline b_{ik} between I_i and I_k. Let us denote this ratio by $\hat{b}^i_{jk} = b_{ij}/b_{ik}$. We can estimate this ratio as

$$\hat{b}^i_{jk} = \arg\min_b \int_{\hat{\mathbf{u}} \in S^2} |d_{ij}(\hat{\mathbf{u}}) - b d_{ik}(\hat{\mathbf{u}})| \, d\hat{\mathbf{u}} \tag{5.9}$$

if we can reliably estimate disparity values, or at least identify and discard those estimates that are not reliable. Performing this minimization for all mutually matched triplets of images would give an estimate of the baselines up to a constant scale factor. Note that this is not as straightforward if angular disparity is used. While we note this interesting property of the formulation presented above, to investigate this further is beyond the scope of this thesis, and we leave this for future work.

5.3.2 Modifications of Planar Algorithm

In the spherical case, we perform all DSI sampling, disparity fusion, and disparity filtering in normalized radial disparity $\hat{d}$ rather than radial disparity d. Note that this would also make it easier to extend our method to multi-view settings, since a value of $\hat{d}$ has the same meaning regardless of the baseline.

We implement the rhombic-dodecahedron maps in CUDA from the description of Fu et al. [55]. This primarily involves device functions to sample a given direction, compute the direction of a given pixel, and look up neighboring pixels along the border of the spherical rhombi.

Guided Filter: This requires computing the integral image on spherical images. Computing the integral image requires the samples to constitute a partially order set, which is not the case for general sampling of the sphere. Latitude-longitude sampling provides such a partial ordering, but the singularities at the poles would make it impossible to apply a box or mean filter that aggregated information across the poles. The choice of the RD-map sampling is helpful here: each spherical rhombus defines a partial ordering, and we can compute the integral image on each spherical rhombus independently. When computing the mean filter, we split the window into sub-windows each contained in a single rhombus, compute the sum over the sub-windows using the integral image, then combine them to get the mean filter over the entire window. Thus we retain the efficiency of the original technique while extending it to spherical images.

Fusion: Warping of disparity maps is done by converting (normalized) radial disparity into angular disparity using (5.8). This maps pixels to the corresponding locations in the target viewpoint, and we then compute the equivalent radial disparity value for that viewpoint and pixel, using

$$\hat{d} = \frac{\sin\gamma}{b\sin(\gamma+\alpha)} \tag{5.10}$$

where α is computed in the reference image to which the disparity map is being warped. To compute the Voronoi diagram on the sphere, we use the angle between pixels as the distance metric rather than Euclidean distance.

5.4 Results

In this section we evaluate our approach on a standard benchmark and publicly available planar data sets and our own omnidirectional data sets. We implemented our method using C++ and CUDA, and ran our experiments on a workstation with an Intel Xeon 3.2 GHz, 12 GB of RAM,

	KITTI (0.44)	"fountain-P11" (6)	panoramas (3)
matching	7.078	54.099	41.105
fusion	0.022	0.453	1.817
filtering	0.878	14.54	5.415
total	7.978	69.092	48.34

Table 5.1: Timings in seconds per image for the different stages of our algorithms for different image sets on our test system. The number of Mpixels of each image is given in parentheses.

and an NVidia Quadro 4000 with 2 GB. Table 5.1 gives the computation times of the different stages of our algorithm. It can be seen that our algorithm is linear in the number of pixels and disparities in practice. Figure 5.5 shows the disparity estimates after different stages of our approach. As can be seen, the multi-scale matching produces mostly high-quality disparity estimates, with a few outliers remaining. These are almost all removed in the fusion stage, and the disparity discontinuities are aligned with the image edges in the filtering stage, while the disparity estimates away from edges are smoothed.

5.4.1 Planar Stereo

We evaluate our method using the KITTI benchmark [2], which is targeted to the application of autonomous driving. These images are approximately 0.44 Mpixels, and are in an uncontrolled setting, with large texureless regions, specular surfaces and saturated pixels. They are further challenging because the (benchmark) images are grayscale and thus we are not able to obtain as much information from the guidance image when filtering the DSI. Ground truth disparities are available via a laser scanner that was synchronized with the stereo cameras. The error for every disparity estimate is computed using the nearest available ground truth pixel (the laser scanner does not produce a fully dense disparity map). For about the top third of the disparity map, no ground truth is available (the scanner does not cover this area), and no error is evaluated. Our error scores are shown in Table 5.2. The first column gives the disparity error threshold above which is considered an outlier. The second and third columns give the percentage of disparity estimates that are outliers among non-occluded pixels and among all pixels, respectively, according to the threshold in the first column. The last two columns give the average disparity error among non-occluded pixels and among all pixels. Our method improves substantially over the single-scale cost volume filtering [145] (rank 11) and falls between ELAS [69] and SDM [149] at rank 7. The full ranking as of the submission of this thesis is given in Table 5.3. While some methods perform better than ours, many do not

Error	Out-Noc	Out-All	Avg-Noc	Avg-All
2 pixels	16.43 %	17.94 %	1.9 px	2.2 px
3 pixels	10.68 %	12.11 %	1.9 px	2.2 px
4 pixels	8.10 %	9.41 %	1.9 px	2.2 px
5 pixels	6.54 %	7.74 %	1.9 px	2.2 px

Table 5.2: Results for our method on the KITTI stereo benchmark. The leftmost column gives disparity error thresholds; pixels with disparity error greater than this value are considered outliers. The next two columns give the number of outliers for each threshold for non-occluded pixels and all pixels, respectively. The last two columns give the average disparity error for non-occluded and all pixels.

estimate a full disparity map and others require solving a complex global optimization. The top ranking method, listed as PCBP [150], *e.g.*, is described as a slanted-plane MRF method using particle convex belief propagation method, and requires segmentation into 1000 superpixels and takes 5 minutes to run. Further, our method could incorporate any single-scale stereo matcher that calculates a cost over the disparity range at each pixel, or from which one could interpolate such a cost volume. One could even use belief propagation [75, 144] at each scale, although this would greatly expand the storage and computational costs at each scale. Our method could also be a front-end to one of the semi-global variants listed on the benchmark. In addition to the quantitative evaluation, we show a disparity result for two of the benchmark stereo pairs in Figures 5.4 and 5.5, where the latter shows the results after the different stages of our algorithm: matching, fusion and filtering.

We tested our method on two images from the "fountain-P11" data set from the dense multi-view benchmark [3], with resulting disparity maps shown in Figure 5.6. We only use two frames from this multi-view data set. Instead of rectifying the images, we use disparity as simply the reciprocal of the depth, i.e. $d = 1/z$. The "fountain-P11" images are 3072×2048. For 128 disparity levels this requires 3GB to store the finest resolution DSI and about 4GB for all scales combined. In contrast our finest resolution EPT-compressed DSI take about 200MB at an average of 4 envelope points per pixel. We obtain globally smooth disparity maps that retain fine-scale detail, as shown in the close-up. Note that the background region at the right edge of the input image in Figure 5.6 is not visible in the other image, and hence the disparity values are interpolated from nearby visible pixels. For the "fountain-P11" images we did not perform radiometric equalization as described in Section 5.2.2, since the images contain very few highlights, and we set the guided filter window size to $r = 36$. Point cloud renderings of

#	Method	Out-Noc	Out-All	Avg-Noc	Avg-All	Density	Time	Environment
1	PCBP	4.13 %	5.45 %	0.9 px	1.2 px	100.00 %	5 min	4 cores @2.5Ghz (Matlab+C/C++)
2	iSGM	5.16 %	7.19 %	1.2 px	2.1 px	94.70 %	8 s	2 cores @2.5Ghz (C/C++)
3	ITGV	6.31 %	7.40 %	1.3 px	1.5 px	100.00 %	7 s	1 core @3.0Ghz (Matlab+C/C++)
4	BSSM	7.50 %	8.89 %	1.4 px	1.6 px	94.87 %	20.7 s	1 core @3.5Ghz (C/C++)
5	OCV-SGBM	7.64 %	9.13 %	1.8 px	2.0 px	86.50 %	1.1 s	1 core @2.5Ghz (C/C++)
6	ELAS	8.24 %	9.95 %	1.4 px	1.6 px	94.55 %	0.3 s	1 core @2.5Ghz (C/C++)
7	**MS-DSI (ours)**	10.68 %	12.11 %	1.9 px	2.2 px	100.00 %	8 s	8+ cores @2Ghz (C/C++)
8	SDM	10.98 %	12.19 %	2.0 px	2.3 px	63.58 %	1 min	1 core @2.5Ghz (C/C++)
9	GCSF	12.06 %	13.26 %	1.9 px	2.1 px	60.77 %	2.4 s	1 core @2.5Ghz (C/C++)
10	GCS	13.37 %	14.54 %	2.1 px	2.3 px	51.06 %	2.2 s	1 core @2.5Ghz (C/C++)
11	CostFilter	19.96 %	21.05 %	5.0 px	5.4 px	100.00 %	4 min	1 core @2.5Ghz (Matlab)
12	OCV-BM	25.39 %	26.72 %	7.6 px	7.9 px	55.84 %	0.1 s	1 core @2.5Ghz (C/C++)
13	GC+occ	33.50 %	34.74 %	8.6 px	9.2 px	87.57 %	6 min	1 core @2.5Ghz (C/C++)

Table 5.3: Ranking information for the KITTI stereo benchmark as of August 30, 2012. The third and fourth columns have the same meaning as their counterparts in Table 5.2, and the disparity error threshold used is 3 pixels.

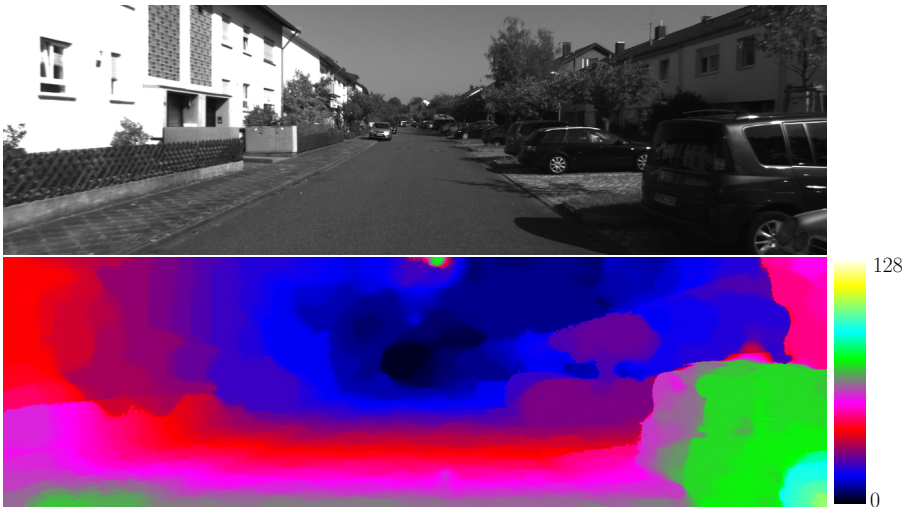

Figure 5.4: An input image and the disparity map from our method for the first stereo pair from the KITTI dataset [2]. The color coding is shown in the bar on the right.

this dataset can be seen in our video[2].

5.4.2 Spherical Stereo

We provide visual evaluation on omnidirectional images sampled with 3 Mpixels using the rhombic dodecahedron-map scheme [55], as shown in Figures 5.7 and 5.8. Note how the images exhibit bright lights, specularities, saturated pixels and lens artifacts. The estimated disparity map is overall smooth, but fine-scale details have been faithfully retained. For example, in Figure 5.8 the chair legs in the lower left corner have been estimated very plausibly even though they are both thin and specular. Note that for these data sets we set $r = 18$.

5.5 Discussion

We have presented a time- and space-efficient multi-scale stereo matching method based on a novel application of distance transforms to the DSI. This multi-scale framework provides regularization across large portions of the image domain, while preserving detail, because it does not explicitly smooth neighboring disparity values. This framework can incorporate any single-scale matching cost computation. We have further presented a novel spherical disparity formulation that allows for both efficient sampling along epipolar arcs, and geometrically cor-

[2]Videos accompanying this thesis can be downloaded from the author's website.

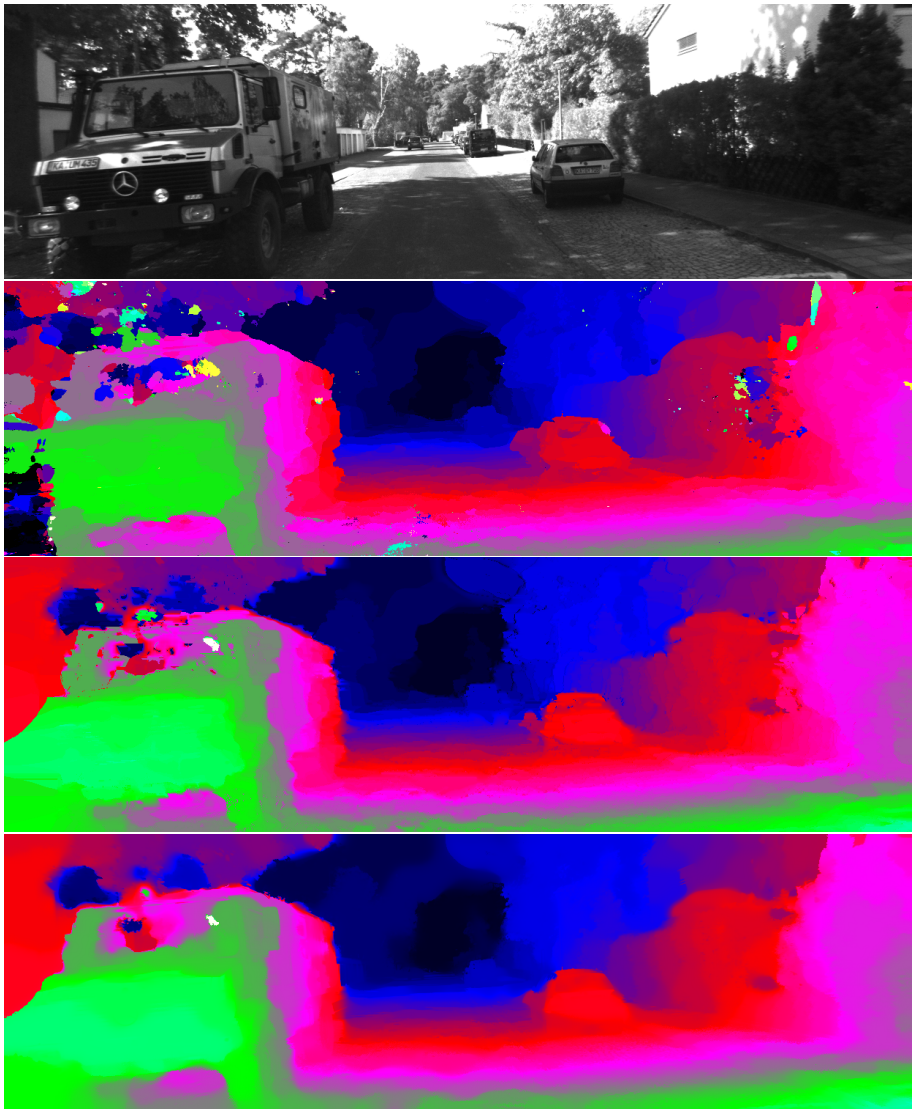

Figure 5.5: Disparity estimates after each stage of our pipeline. From top to bottom: input image, disparity after multi-scale matching, disparity after fusion, disparity after filtering. Disparity range is the same as for Figure 5.4.

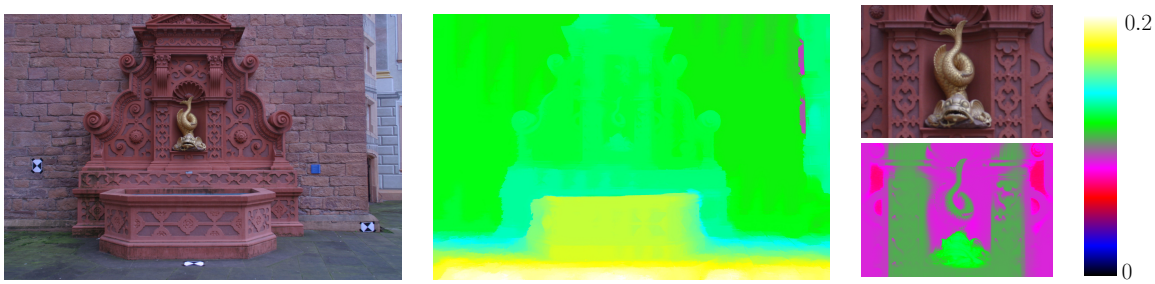

Figure 5.6: An image from “fountain-P11” high-resolution data set [3] and our disparity result. The close-up on the center of the fountain shows the disparity color-mapping rescaled for better visualization of the fine-scale detail.

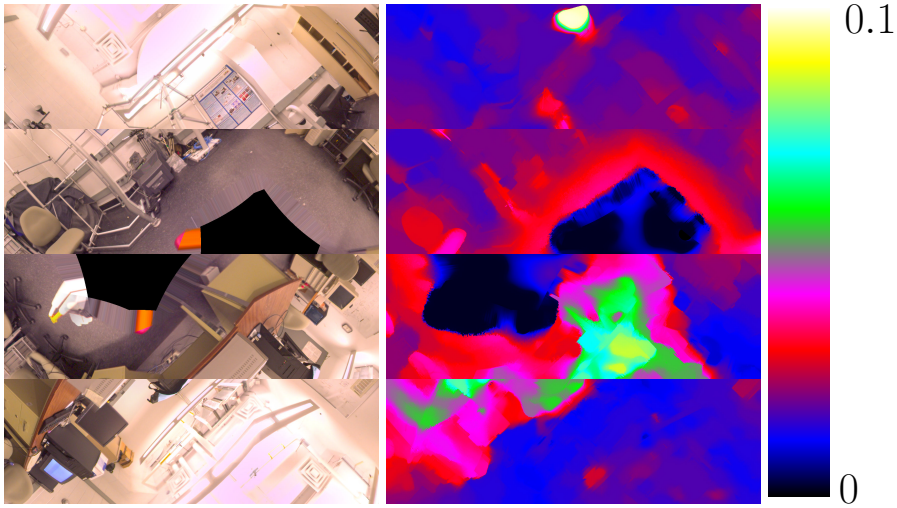

Figure 5.7: A spherical image with RD-sampling and the resulting disparity map.

rect cost aggregation and disparity smoothing. We have demonstrated state-of-the-art results for planar stereo and similar high-quality results for spherical stereo. In Chapter 6 we extend this method to multi-view stereo for both planar and spherical images. Future work includes applying our multi-scale matching framework with different methods to calculate single-scale DSIs and investigating how to avoid testing disparity values that cannot minimize (5.2).

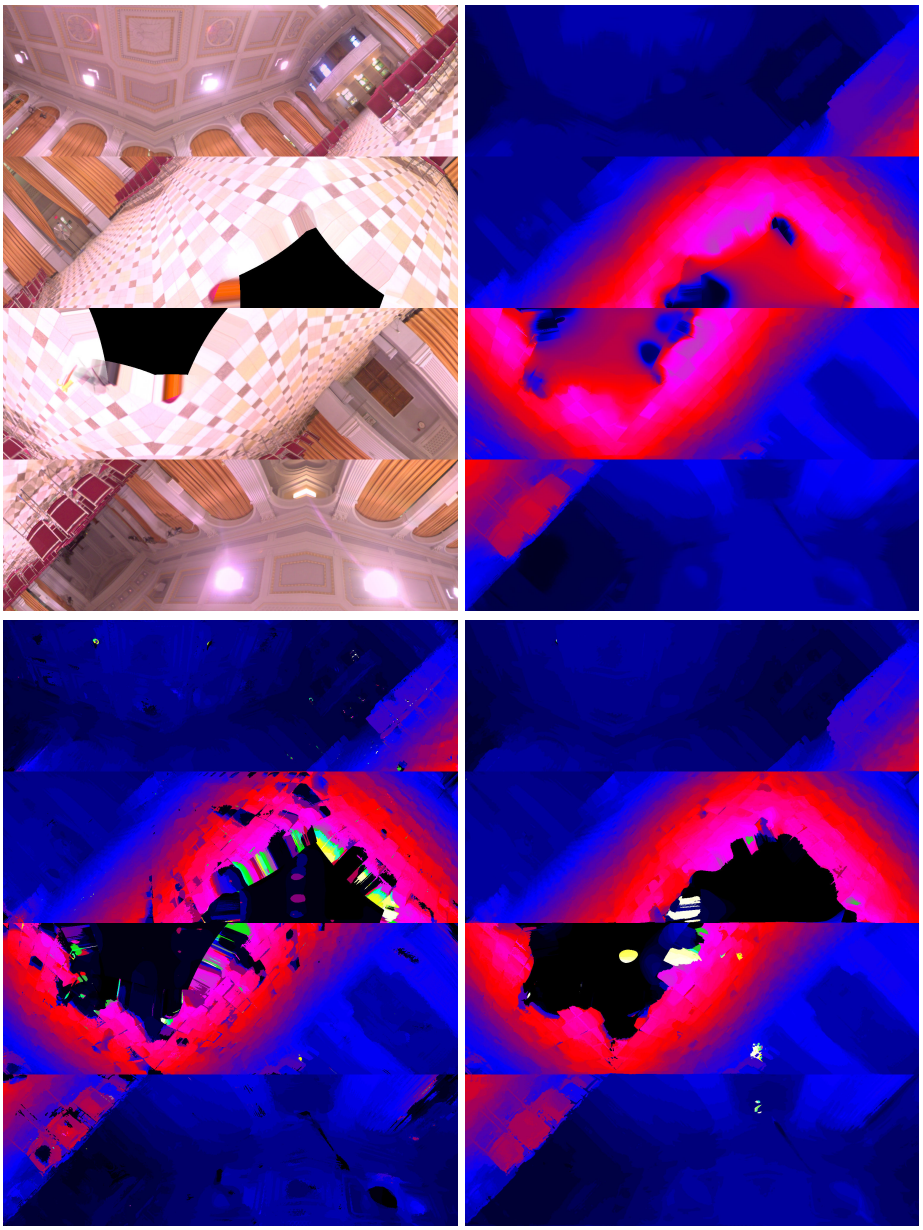

Figure 5.8: Top: A spherical image with RD-sampling and the resulting disparity map from our method. Bottom: single-scale matching (Section 5.2.2) and multi-scale matching (Section 5.2) (before post-processing). Disparity range is the same as Figure 5.7.

Chapter 6

Spherical Stereo with Wavelet Filtering for Novel View Synthesis

To perform real-time novel view synthesis often means performing some amount of stereo reconstruction, because this allows efficient rendering of the reconstructed models. To perform stereo reconstruction from spherical images requires a framework for disparity or depth estimation from spherical images. In this chapter we develop a framework for disparity estimation from spherical images, building upon the multi-scale stereo matching method and spherical disparity formulation presented in Chapter 5. This formulation allows us to efficiently merge multiple pairwise disparity estimates, perform efficient occlusion tests, and efficient disparity map fusion enforcing multi-view visibility constraints. We then perform wavelet-filtering of the disparity maps, taking into account color information from the input images, and robust confidence information estimated during the initial disparity estimate phase. Our method involves an offline pre-processing stage, where we estimate disparity and generate the point clouds, and an online, real-time, rendering or view synthesis stage, where we allow the user to navigate freely in the environment at over 30Hz.

6.1 Introduction

In this chapter we address the problem of real-time virtual navigation in environments captured with multiple panoramic images, generating a smooth sequence of planar images at novel viewpoints. While this problem has not yet been satisfactorily solved in the literature, we make a number of significant contributions and present a complete viable system. Our method for real-time novel view synthesis (NVS) is based on a spherical disparity layer representation of

the scene. The real-time constraint on NVS makes stereo reconstruction a good candidate, because once the geometry is reconstructed, novel views can be generated very efficiently, and a good deal is known about using disparity or depth layers for NVS. Our motivation for using omnidirectional disparity layers lies in that we can immersively cover an environment with relatively few images and layers, as compared to planar images. Additionally, it has been documented that the wide field-of-view allows for more accurate structure-from-motion [22, 23], and we demonstrate accurate calibration of panoramas using a spherical formulation. Figure 6.1 shows a spherical image with rhombic-dodecahedron map (RD-map) sampling [55], the estimated disparity map for the same viewpoint, a reference view from another panorama not used in the reconstruction process, and synthesized view from that viewpoint. We use RD-map sampling because of its high uniformity, low distortion and the fact that individual spherical rhombi can be treated similarly to planar images.

In this chapter we make the following contributions: a system for real-time general viewpoint NVS based on omnidirectional images and disparity layers; a visibility-based fusion of spherical disparity maps for outlier removal, similar to that of Merrell et al. for planar depth maps [99]; an iterative disparity filtering step based on a multi-weight wavelet transform that takes color and confidence into account. An additional contribution is a spherical-geometry-based calibration algorithm for omnidirectional images. We show how our reconstruction can be used for virtual walkthroughs of environments captured with panoramic images. We demonstrate our results with a video showing a virtual walkthrough of a real environment[1].

6.1.1 Related work

The use of panoramic depth layers for NVS is inspired by layered depth panoramas (LDP) [15], similar to layered depth images (LDI) [13], in which a dense primary panoramic color and depth layer, and a few concentric sparse secondary layers are estimated from a set of planar images. In contrast, our method takes multiple panoramic images and estimates multiple dense non-concentric panoramic (*i.e.*, from different viewpoints) colored point clouds. Additionally, while LDP use discrete optimization to regularize the depth estimates, we use a continuous formulation to robustly estimate disparity.

Depth map-based multi-view stereo methods are popular because they are potentially very straightforward, and they can take advantage of techniques from binocular stereo to enforce smoothing constraints. Some examples of depth map-based multi-view stereo methods include the methods of Goesele et al. [96] and of Merrell et al. [99]. Although there are few examples

[1] Videos accompanying this thesis can be downloaded from the author's website.

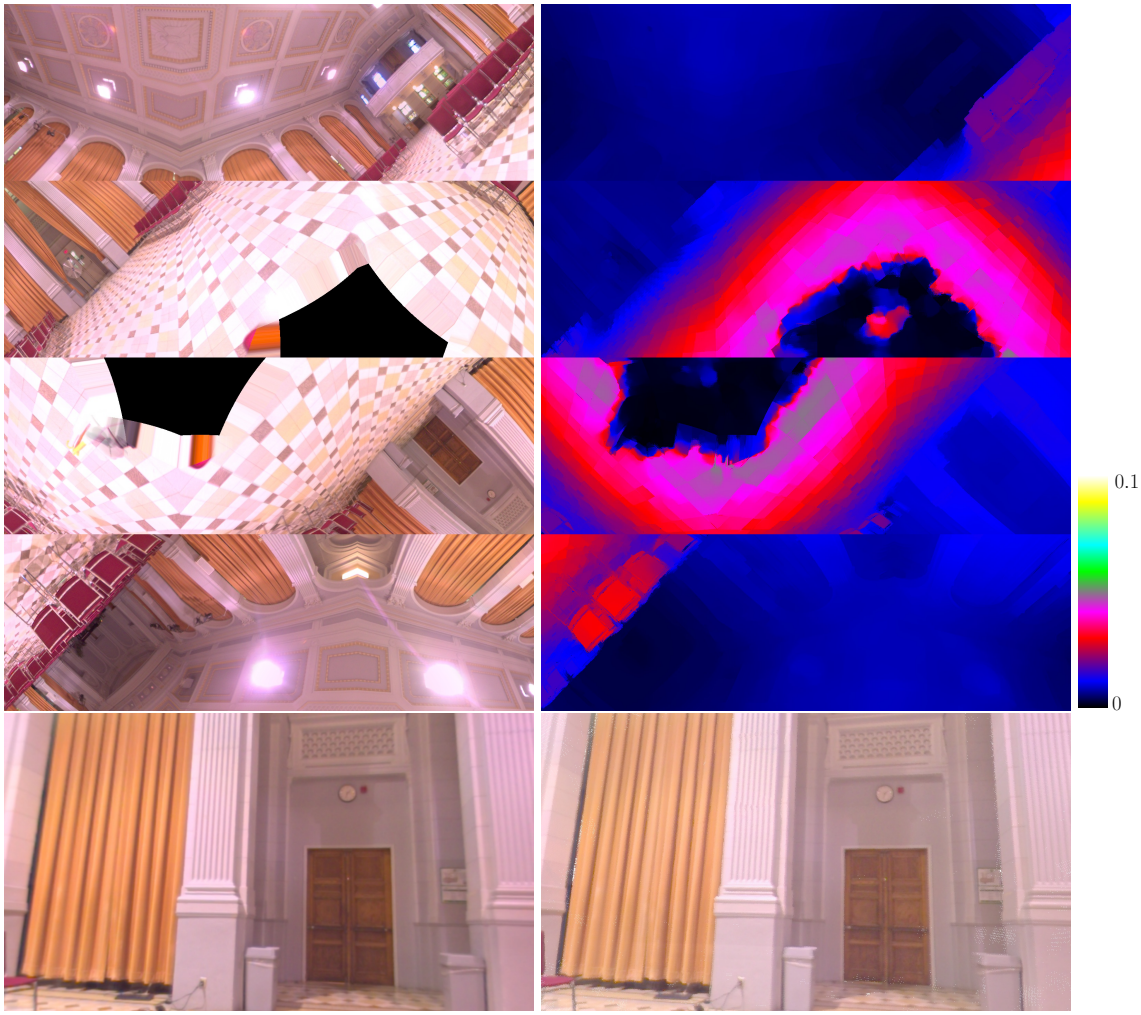

Figure 6.1: Top: an input RD-map spherical image and corresponding estimated disparity map; the disparity range and color scale is given on the right. Bottom: reference view from another panorama not used for reconstruction (left) and synthesized view from same viewpoint (right).

of multi-view omnidirectional stereo, depth-map based methods seem straightforward to adapt due to their flexibility. We compute a spherical disparity map for each panorama, but we do not meld them into a single mesh. We extend the visibility-based fusion of Merrell et al. [99] to spherical disparity maps. However, we leave out the surface meshing step and keep the result as a set of overlapping disparity layers.

The wavelet domain has previously been used to regularize stereo estimation. Miled et al. [67] used a wavelet domain representation of the disparity map to regularize stereo reconstruction with an edge-preserving smoothing prior. Our method is similar in that we use the wavelet domain to enforce edge-preserving smoothing, but we further incorporate side information, including color and confidence, to enhance the filtering process. We use the edge-avoiding à-trous wavelet used by Dammertz et al. [51] for filtering of global illumination images.

NVS methods that use panoramic images include Plenoptic Stitching [10], Sea of Images [11] and Street-Slide [12], which all take vertical strips of the input images and paste them into the novel view. Methods for NVS based on disparity layer reprojection include LDP [15] and LDI [13]. The rendering of these layer-based representation can be done using a splatting technique [13] or using a mesh-based method [14]. Our method is based on omnidirectional disparity layer reprojection, and is different from all of these methods.

Micusik and Kosecka [23] constructed piecewise-planar 3D models of city environments from sequences of streetview panoramic images. They employed a piecewise perspective panoramic camera model with a single virtual center of projection, which they leverage in their pose estimation stage. Following pose estimation is a sweeping stage that finds candidate projections for super-pixels from each segmented panorama. A labeling problem is solved to assign a plane equation to each super-pixel. The resulting depth maps are then fused using a viewpoint dependent approach that leverages the normals computed during the sweeping stage. While we also leverage panoramic field-of-view to aid the calibration process, we use a spherical camera model as opposed to piecewise perspective, and we do not restrict the scene to be piecewise planar.

Ambient point clouds [17] incorporate the uncertainty in the reconstruction process into the visualization of reconstructed scenes. They augment standard point clouds and textured meshes for view interpolation. However, the novel viewpoints are restricted to be near the line connecting any two camera centers, because the ambient points are distributed along the viewing ray and therefore the epipolar relationship between the input images and the novel view is linearly dependent on the epipolar relationship between the input images. We handle difficult to reconstruct areas by using multiple omnidirectional view-dependent proxies, without

restricting the novel viewpoint.

Li et al. [102] performed stereo reconstruction from multiperspective panoramas, deriving approximately horizontal epipolar geometry to allow standard stereo algorithms to be applied. Like Li et al., we sample uniformly in inverse radial distance, however, we derive different epipolar relationships for single-perspective panoramas. Kang and Szeliski [22] used feature-based structure-from-motion to estimate the pose of panoramic images and a sparse set of 3D points, and omnidirectional multi-baseline stereo to obtain denser 3D points for meshing. They noted that the pose of the panoramic viewpoints can be determined with greater accuracy than in the case of images with narrower fields-of-view.

A related family of methods enumerate depth values, but optimize the synthesized view rather than the geometry. These include image-based priors [18], in which a library of image patches is built from the input images, and constrains the synthesized view to locally look like the input images. Woodford et al. [19] modify this approach to be solved using a pairwise Markov random field. Li and Li [20] use a joint conditional random field with online learning, modeling pixel interactions with steerable filters. It is not clear that these methods can run in real-time, so we perform stereo reconstruction and render view-dependent proxies.

6.1.2 Overview

We model the geometry of a scene as a set of non-concentric spherical disparity and color layer pairs. We model the disparity layers as continuous piecewise smooth functions.

Section 6.2 describes our initial disparity estimation. We perform multi-scale stereo matching as described in Chapter 5 between pairs of spherical images to estimate an initial set of spherical disparity space images (DSI). We fuse the disparity maps using multi-view visibility constraints similar to those for depth map fusion from Merrell et al. [99], leverage the properties of our spherical disparity formulation, removing the outliers which cause the most egregious artifacts when using stereo for NVS. We also discuss our calibration technique based on spherical geometry.

Section 6.3 describes how we filter or smooth the fused disparity maps using an edge-avoiding redundant wavelet transform [51]. The transform takes color and the confidence of the fused disparity maps into account as side-information. This continuous smoothing technique is simple, fast and allows for non-fronto-parallel surface reconstruction.

Section 6.4 describes how we generate novel views given a set of spherical images, disparity layers and confidence maps. We generate meshes on-the-fly, texturing them with the input images and reprojecting them into the novel viewpoint. We blend the resulting novel views

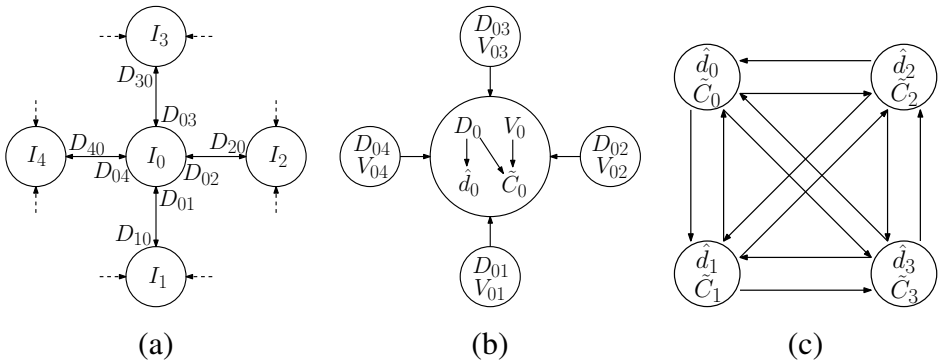

Figure 6.2: Stages in disparity estimation: pairwise matching (a), DSI merging (b), disparity map fusion (c).

using the robust confidence maps computed during disparity estimation.

Finally, in Section 6.5 we present implementation details and experimental results of our approach.

6.2 Disparity Estimation

To robustly estimate disparity we build upon the multi-scale disparity estimation framework from Chapter 5, extending it to multi-view reconstruction. We begin with the pairwise multi-scale matching as in Section 5.2, but since we can match each reference image against multiple matching images, we subsequently merge the resulting pairwise disparity maps for each reference image. We then perform full multi-view fusion, as opposed to the limited binocular fusion performed in Section 5.2.4. These stages are depicted graphically in Figure 6.2.

6.2.1 Initial Disparity Estimation

Once we have computed pairwise disparity maps and DSI, we perform cross-checking by comparing the disparity maps for pairs where the reference and matching images are reversed. From Figure 5.3, we see that if we switch the reference and matching images γ remains the same. Consider the normalized radial disparity map $\hat{d}_{ij}(\hat{\mathbf{u}}_i) = \arg\min_{\hat{d}} D_{ij}(\hat{\mathbf{u}}_i, \hat{d})$ that minimizes the DSI D_{ij} at each pixel $\hat{\mathbf{u}}_i$ for reference image I_i and matching image I_j. For a given pixel direction $\hat{\mathbf{u}}_i$, $\hat{d}_{ij}(\hat{\mathbf{u}}_i)$ gives us $\gamma_{ij}(\hat{\mathbf{u}}_i)$ using (5.8), which allows us to compute the corresponding location in I_j, $\hat{\mathbf{u}}_j$. We can then sample $d_{ji}(\hat{\mathbf{u}}_j)$ and compute $\gamma_{ji}(\hat{\mathbf{u}}_j)$. If $\gamma_{ji}(\hat{\mathbf{u}}_j) = \gamma_{ij}(\hat{\mathbf{u}}_i)$ then the point is visible in both images and $D_{ij}(\hat{\mathbf{u}}_i)$ can be considered reliable.

In practice, we compute a confidence based on the discrepancy between the two estimates,

$$V_{ij}(\hat{\mathbf{u}}_i) = \begin{cases} (\gamma_{ij} - \gamma_{lo})/(\gamma_{ij} - \gamma_{ji}) & \text{if } \gamma_{ji} < \gamma_{lo} \\ (\gamma_{hi} - \gamma_{ij})/(\gamma_{ji} - \gamma_{ij}) & \text{if } \gamma_{ji} > \gamma_{hi} \\ 1 & \text{otherwise} \end{cases} \tag{6.1}$$

where γ_{lo} and γ_{hi} are computed by plugging $(1-\epsilon)\hat{d}_{ij}$ and $(1+\epsilon)\hat{d}_{ij}$ into (5.8), respectively, where ϵ is a user parameter. We use $\epsilon = 0.01$. The equivalent confidence value for planar disparity is given by $V_{ij}(u) = \epsilon_{ij}(u)/\max(\epsilon_{ij}(u), |d_{ij} - d_{ji}|)$, where $\epsilon_{ij}(u) = \epsilon \max(d_{ij}(u), \Delta d)$ for disparity step Δd.

With these confidence maps in hand, we merge all disparity maps computed for the same reference image. This is done by selecting the disparity estimate for each pixel, from the set of pairwise disparity maps, which balances the amount of confidence attributed to disparity estimates greater than and less than itself. We first sort the pairwise disparity estimates at each pixel, and re-order the confidence measures accordingly, then find the first disparity where the cumulative confidence is greater than half the total confidence. A merged disparity map $\hat{d}_i$ is shown in Figure 6.6.

6.2.2 Disparity Map Fusion

We proposed a visibility-based spherical disparity map fusion approach similar to that of Merrell et al. [99] for planar depth maps. This approach exploits the properties of our formulation for efficient outlier removal. We do not have to reconstruct any 3D points in this process.

This stage of our pipeline takes as input a set of disparity maps, each with an accompanying confidence map, and produces a fused disparity map, with outliers removed, for each input. It also outputs a fused confidence map. We begin with the stability-based fusion algorithm from Merrell et al., adapted to spherical disparity. Each input disparity map is set as the reference disparity map in turn, and the other disparity maps are warped or reprojected into its viewpoint. Each of these disparity estimates for this viewpoint is evaluated using the notion of stability, which is the number of times a disparity estimate is occluded by another minus the number of times that disparity estimate violates the free-space of another estimate. The disparity estimate with the smallest stability greater or equal to zero is taken as the result. This is done independently for each pixel in the reference view.

We denote by $\hat{d}_i^j(\hat{\mathbf{u}}_i)$ the disparity map $\hat{d}_j$ warped to the viewpoint of $\hat{d}_i$, as in Figure 6.2c. This is done very efficiently using a forward-splatting technique. The $\hat{d}_i^j$ are used to check for occlusions along the rays of the reference disparity map ($\hat{d}_i^j(\hat{\mathbf{u}}_i) > \hat{d}_i(\hat{\mathbf{u}}_i)$), and

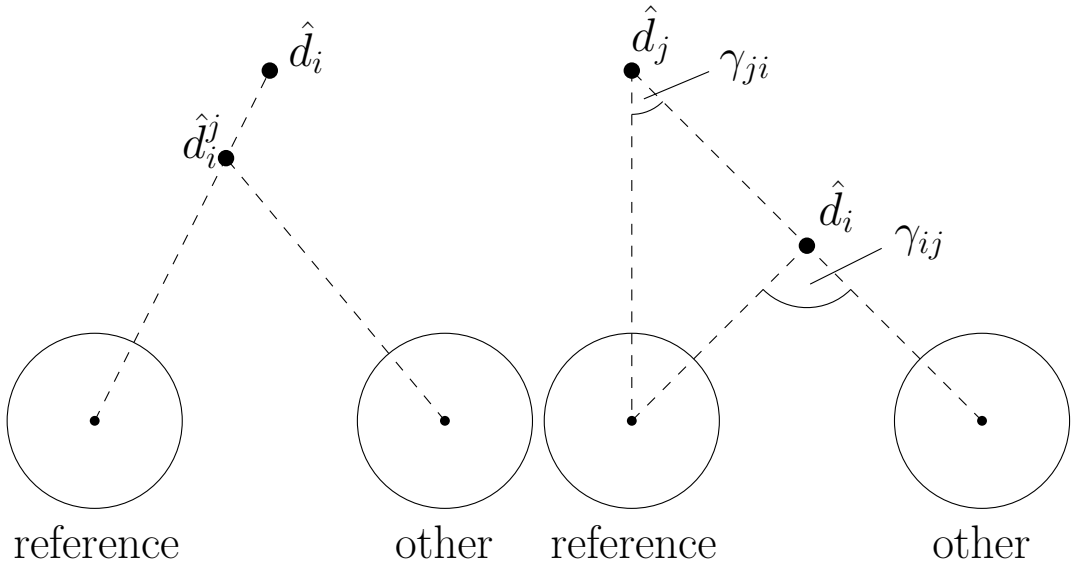

Figure 6.3: Illustrations of an occlusion (left) and a free-space violation (right). Note that although the points are labeled with, *e.g.*, $\hat{d}_i$, they actually show the points obtained by scaling the pixel direction by, *e.g.*, $1/\hat{d}_i$.

angular disparity is used to check for free-space violations along the rays of the other disparity maps ($\gamma_{ji}(\hat{\mathbf{u}}_j) < \gamma_{ij}(\hat{\mathbf{u}}_i)$). An occlusion occurs when the current disparity estimate, either the value from the reference disparity map or one of the warped disparity maps, is occluded by another, closer disparity estimate with respect to the same viewpoint. Again, this may be either the reference disparity or a warped disparity. A free-space violation occurs when the current disparity estimate occludes a disparity estimate with respect to the other viewpoint. So, checks for free-space violations examine the non-warped disparity maps. An example for both an occlusion and a free-space violation are shown in Figure 6.3. On the left, a warped disparity estimate from another view occludes the reference disparity estimate for a given pixel. On the right, the reference disparity estimate for given pixel, violates the free space of a disparity estimate from another view. If the other disparity estimate is in $\left[(1-\epsilon)\hat{d}_i, (1+\epsilon)\hat{d}_i\right]$, it is considered in agreement with the reference disparity estimate.

Following initial visibility constraint enforcement, we iteratively pull disparity estimates towards each other. This is inspired in part by the space-time fusion of Zhang et al. [100], and is designed to ensure that at a fine-scale the disparity estimates from different viewpoints agree with one another, to avoid blurring or double-edge in the synthesized view. We iteratively update each disparity map as follows, denoting the disparity map being updated as the reference disparity map $\hat{d}_i$. We warp all other nearby disparity maps into the reference point of view, and

at each pixel determine which of these estimates is close to the reference disparity estimate. That is, which warped disparities are within $\epsilon \max(\hat{d}_i(\hat{\mathbf{u}}), \Delta\hat{d})$ of $\hat{d}_i(\hat{\mathbf{u}})$. We then assign the average of these disparities to $\hat{d}_i(\hat{\mathbf{u}})$. We perform 32 iterations of this, as a trade-off between the time required and the fine-scale consistency of the disparity estimates. While this process helps incorporate multi-view consistency, it is not clear that it converges in the limit, so we choose a relatively low number of iterations that results in visually pleasing results.

We also generate a fused confidence map, which is the sum (for each pixel) of the confidence of the disparity maps that support that estimate. A disparity map supports the disparity estimate if the computed angular disparity values agree as in (6.1).

We apply a final pass through the disparity maps to remove any lingering outliers that cause free-space violations of the other disparity estimates. For each neighboring disparity map $\hat{d}_j$ of $\hat{d}_i$, we count by how many disparity steps $\hat{d}_i$ occludes $\hat{d}_j$. Specifically, for each pixel $\hat{\mathbf{u}}_i$ in disparity map $\hat{d}_i$, we compute the sum

$$\sum_j \left\lfloor (\hat{d}_j(\hat{\mathbf{u}}_j) - \hat{d}_j^i(\hat{\mathbf{u}}_j))/(\sqrt{\epsilon}\hat{d}_j(\hat{\mathbf{u}}_j)) \right\rfloor \tag{6.2}$$

where $\hat{\mathbf{u}}_j$ is the pixel direction in I_j mapped to by disparity $\hat{d}_i$ from $\hat{\mathbf{u}}_i$, and $\hat{d}_j^i$ is the warped disparity. For a particular j, the term in the summation is positive if the warped disparity is less than $\hat{d}_j$ and negative if it is greater. The use of $\sqrt{\epsilon}$ instead of ϵ is chosen because we only want to remove the most blatant outliers are this stage. Referring to Figure 6.6, this final pass is targeted at the high disparity values in the center of the disparity map, which correspond to pixels with no color information in the input panorama (*i.e.*, where there is no sensor coverage in the spherical camera, see Section 6.2.3). These are not removed in the stability-based fusion because similar artifacts occur in multiple disparity maps, so counting the occlusions positively allows these estimates to have non-negative stability. By only examining gross free-space violations we are able to remove them. However, this does remove some good disparity estimates along object boundaries. We designate all pixels $\hat{\mathbf{u}}_i$ for which this sum is negative as outliers and set the disparity and confidence values to 0.

We denote the fused disparity and confidence maps by $\hat{d}_i^{(f,0)}$ and $\tilde{C}_i^{(f)}$, respectively. A disparity result after the fusion stage is shown in Figure 6.6. As can be seen, we remove the vast majority of outliers at the cost of losing some legitimate disparity estimates.

6.2.3 Data Capture and Calibration

The preceding assumes we have not only calibrated, but rotationally aligned spherical images, although this latter condition is simply a matter of convenience and efficiency. We first discuss

the process and equipment we used to capture the omnidirectional images, then the procedure by which we calibrate and align those images.

The two datasets used in this chapter were captured on the University of Ottawa campus. One, shown in Figure 6.13 was captured in the VIVA lab. The other was captured in Tabaret Hall, shown in Figure 6.1 and Figures 6.8 through 6.12. Images were captured with the Point Grey Ladybug 2 spherical camera system[2], which is a multi-sensor omnidirectional camera with six 1024×768 CCDs with wide-angle lenses covering $> 75\%$ of the sphere. The camera was mounted on top of a 914 PC-Bot[3], a simple to use mobile robot. A path of capture locations was pre-programmed and remotely initiated from a laptop. The images from the separate sensors were stitched together using the manufacturer's calibration and software kip with a fixed parallax, which resulted in some seams and ghosting artifacts, especially in the lab data set. The resulting spherical images were then calibrated and aligned as described below.

The wide field-of-view of panoramic images allows more accurate structure-from-motion than for planar images [22, 23] due to the fact that correspondences in a wider variety of directions are used to estimate relative pose. Thus, we wish to calibrate using our stitched panoramic images, rather than calibrating individual planar images. This means we need a custom calibration method. Aly and Bouget [151] perform calibration of spherical images assuming planar motion, and do not apply bundle adjustment to refine pose estimates. We considered using sparse bundle adjustment (SBA) [152], however, this allows only a Euclidean error metric, whereas we wish to use an angular reprojection error to take advantage of our omnidirectional camera model. Using spherically projected images, our camera model consists only of a rotation and translation. No intrinsic parameters are required.

We perform calibration in an incremental, and hence scalable, way. We treat our set of panoramas as being in sequence, and calibrate image $i + 1$ relative to image i before chaining this onto the calibration of image i relative to the world coordinate system. This pairwise calibration is done using a method similar to that of Kangni and Laganière [153]. We first detect SURF features [154] in each panorama. Future work might include using SIFT features [98] for the sphere [114] to retain greater robustness to wider baselines. Such an approach could be accelerated by the GPU implementation of the spherical harmonic transform presented in Chapter 3. An initial pairwise rotation and translation are computed using RANSAC [155]. From the pairwise correspondences, 3D points in the scene are reconstructed by solving a linear system. Remaining outliers are removed using the "X84" outlier rejection rule [156].

Subsequently, a spherical bundle adjustment is applied to refine the pairwise calibration

[2] `http://www.ptgrey.com/products/ladybug2/ladybug2_360_video_camera.asp`
[3] `http://www.whiteboxrobotics.com/`

parameters and the positions of the 3D points by directly minimizing the reprojection error, as measured by the angle between the detected feature direction and that of the reprojected scene point. Specifically, we minimize the energy

$$E_r(\mathcal{C},\mathcal{U}) = \sum_i \sum_j v_{ij} \arccos(\hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}) \tag{6.3}$$

where $\hat{\mathbf{y}}_{ij}$ is the location of feature j as detected in image i, $\hat{\mathbf{u}}_{ij}$ is the projection of 3D scene point $\mathbf{u}_j \in \mathcal{U}$ into image i, and v_{ij} is a visibility term indicating whether feature j was detected in image i. In our camera model, the projection is given by $\hat{\mathbf{u}}_{ij} = (q_i u_j q_i^{-1} + \mathbf{t}_i) / \left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\|$ where q_i and $\mathbf{t}_i$ are a quaternion and a vector representing the rotation and translation, respectively, transforming a point in world coordinates into the coordinate system of image i, and u_j is the quaternion form of the vector $\mathbf{u}_j$.

This energy function yields an analytical gradient that is inexpensive to evaluate and can be efficiently minimized using a quasi-Newton method [157]. The energy function (6.3) is minimized over the set of camera parameters $\mathcal{C} = \{q_i, \mathbf{t}_i : i = 1, \ldots, N\}$ and 3D scene point positions $\mathcal{U}$. In both our data sets we use $N = 86$.

The gradient of this energy function with respect to the camera parameters and scene point positions is a $7N + 3|\mathcal{U}|$ dimensional vector, which can be broken down into a derivative per camera parameter and per point. We use a quaternion to represent the rotation of camera with respect to the world coordinate system by an angle of θ about axis $\hat{\mathbf{a}}$, with the following notation: $q = (q_w, q_x, q_y, q_z) = (q_w, \mathbf{q}_v) = (\cos(\theta/2), \hat{\mathbf{a}} \sin(\theta/2))$. The derivative of (6.3) with respect to a given camera rotation can be written as follows,

$$\begin{aligned}
\frac{\partial E_r}{\partial q_{iw}} &= \sum_j v_{ij} \frac{-1}{\sqrt{1 - \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}} \frac{\partial \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}{\partial q_{iw}} \\
\frac{\partial E_r}{\partial q_{ix}} &= \sum_j v_{ij} \frac{-1}{\sqrt{1 - \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}} \frac{\partial \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}{\partial q_{ix}} \\
\frac{\partial E_r}{\partial q_{iy}} &= \sum_j v_{ij} \frac{-1}{\sqrt{1 - \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}} \frac{\partial \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}{\partial q_{iy}} \\
\frac{\partial E_r}{\partial q_{iz}} &= \sum_j v_{ij} \frac{-1}{\sqrt{1 - \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}} \frac{\partial \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}{\partial q_{iz}}
\end{aligned} \tag{6.4}$$

where

$$\frac{\partial \hat{\mathbf{y}}_{ij} \cdot \hat{\mathbf{u}}_{ij}}{\partial q_{iw}} = \hat{\mathbf{y}}_{ij} \cdot \frac{\partial \hat{\mathbf{u}}_{ij}}{\partial q_{iw}} \tag{6.5}$$

$$\frac{\partial \hat{\mathbf{u}}_{ij}}{\partial q_{iw}} = \frac{\frac{\partial (q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{iw}} \left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\| - (q_i u_j q_i^{-1} + \mathbf{t}_i) \frac{\partial \left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\|}{\partial q_{iw}}}{\left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\|^2} \tag{6.6}$$

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{iw}} = 2\left(\mathbf{u}_j q_{iw} + \mathbf{q}_{iv} \times \mathbf{u}_j\right) \tag{6.7}$$

$$\frac{\partial \left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\|}{\partial q_{iw}} = \left((q_i u_j q_i^{-1} + \mathbf{t}_i) \cdot \frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{iw}} \right) / \left\| q_i u_j q_i^{-1} + \mathbf{t}_i \right\| \tag{6.8}$$

and similarly for q_{ix}, q_{iy} and q_{iz}

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{ix}} = \mathbf{q}_{iv} u_{jx} + \begin{bmatrix} \mathbf{q}_{iv} \cdot \mathbf{u}_j \\ -2q_{iw}u_{jz} + q_{iy}u_{jx} - 2q_{ix}u_{jy} \\ 2q_{iw}u_{jy} + q_{iz}u_{jx} - 2q_{ix}u_{jz} \end{bmatrix} \tag{6.9}$$

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{iy}} = \mathbf{q}_{iv} u_{jy} + \begin{bmatrix} 2q_{iw}u_{jz} + q_{ix}u_{jy} - 2q_{iy}u_{jx} \\ \mathbf{q}_{iv} \cdot \mathbf{u}_j \\ -2q_{iw}u_{jx} + q_{iz}u_{jy} - 2q_{iy}u_{jz} \end{bmatrix} \tag{6.10}$$

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial q_{iz}} = \mathbf{q}_{iv} u_{jz} + \begin{bmatrix} -2q_{iw}u_{jy} + q_{ix}u_{jz} - 2q_{iz}u_{jx} \\ 2q_{iw}u_{jx} + q_{iy}u_{jz} - 2q_{iz}u_{jy} \\ \mathbf{q}_{iv} \cdot \mathbf{u}_j \end{bmatrix} \tag{6.11}$$

are substituted into appropriate versions of (6.6). The derivatives with respect to t_{ix}, t_{iy} and t_{iz} are similar in form to (6.6), but much simpler in that

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial t_{ix}} = [1\ 0\ 0]^T \tag{6.12}$$

and similarly for t_{iy} and t_{iz}.

For completeness, we now give the derivatives (of the coordinate transform) with respect to the components of the scene points. They are,

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial u_{jx}} = q_{ix}\mathbf{q}_{iv} + \begin{bmatrix} q_{iw}^2 - q_{iy}^2 - q_{iz}^2 \\ 2q_{iw}q_{iz} + q_{ix}q_{iy} \\ -2q_{iw}q_{iy} + q_{ix}q_{iz} \end{bmatrix} \tag{6.13}$$

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial u_{jy}} = q_{iy}\mathbf{q}_{iv} + \begin{bmatrix} -2q_{iw}q_{iz} + q_{iy}q_{ix} \\ q_{iw}^2 - q_{iz}^2 - q_{ix}^2 \\ 2q_{iw}q_{ix} + q_{iy}q_{iz} \end{bmatrix} \tag{6.14}$$

$$\frac{\partial(q_i u_j q_i^{-1} + \mathbf{t}_i)}{\partial u_{jy}} = q_{iz}\mathbf{q}_{iv} + \begin{bmatrix} 2q_{iw}q_{iy} + q_{iz}q_{ix} \\ -2q_{iw}q_{ix} + q_{iz}q_{iy} \\ q_{iw}^2 - q_{ix}^2 - q_{iy}^2 \end{bmatrix} \tag{6.15}$$

where, again, these are plugged into appropriate versions of (6.6) and subsequently into appropriate versions of (6.5) and then (6.4) to get the derivative of the energy function.

Following pairwise calibration, the baseline between images i and $i + 1$ is estimated using matches in previous images by solving a linear system. Newly constructed 3D points are added to the global set, and bundle adjustment is run for the full set of $i + 1$ camera parameters and scene points.

6.3 Disparity Map Filtering

After the initial disparity estimation and disparity map fusion stages of our pipeline, near-viewpoint (high-disparity) outliers, which cause the worst visual artifacts in view synthesis, have been removed. However, there remain low-disparity outliers and small-scale noise in the disparity maps, which cause visually unpleasing distortions for close surfaces. We deal with these problems by applying a redundant wavelet transform and discarding the detail coefficients. The wavelet transform applies edge-preserving smoothing while considering the confidence of the disparity estimates.

We use an edge-avoiding redundant wavelet transform to regularize our fused disparity maps. This allows us to perform continuous smoothing, rather than discrete optimization. We incorporate color and confidence as side information to guide the regularization. After applying the wavelet transform, we are only interested in the scaling coefficients, as they contain the smoothed disparity values. We apply the wavelet transform iteratively for $t = 1, \ldots, T$; i.e. feeding the scaling coefficients back in as the source data. We use $T = 64$.

The wavelet transform we use is based on the algorithm à-trous, which uses filters with a fixed number of non-zero coefficients, but increasing step size at different resolutions. In contrast to Chapter 5, where we apply a form of median filter in the case of binocular stereo matching, here we have applied an extensive multi-view fusion stage in which many points of view have been used to remove a much higher proportion of outliers. Hence, we use a weighted mean filter (at multiple scales), instead of a weighted median filter. Rank statistics, such as median filters, are typically more robust to outliers than moment statistics, such as a weighted mean, which are based on summations. However, since we have more effectively removed outliers already with the multi-view fusion, we can use a mean filtering technique which provides better smoothing of small scale noise without introducing step-like artifacts in the resulting disparity maps. Further, because we apply a wavelet transform iteratively, after the initial iteration we are taking the weighted mean of the scaling coefficients from the previous wavelet transform, which have undergone large-scale, but edge-sensitive smoothing.

By using reasonably strong edge-avoidance weight, as described below, we avoid smoothing over depth discontinuities.

The wavelet decomposition of resolution level k into level $k-1$ goes by

$$c_{k-1}(\hat{\mathbf{u}}) = \frac{1}{W} \sum_{\hat{\mathbf{v}} \in N_k(\hat{\mathbf{u}})} w(\hat{\mathbf{v}}) w_u(\hat{\mathbf{v}}) h_k(\hat{\mathbf{v}}) c_k(\hat{\mathbf{v}}) \tag{6.16}$$

where $h_k(\hat{\mathbf{v}})$ is a B-spline filter kernel for level k, $w(\hat{\mathbf{v}})$ is a static weight for pixel $\hat{\mathbf{v}}$, $w_u(\hat{\mathbf{v}})$ is a bilateral weight for pixel $\hat{\mathbf{v}}$ relative to pixel $\hat{\mathbf{u}}$, c_k is the level k scaling coefficients, and $N_k(\hat{\mathbf{u}})$ is the level k neighborhood of pixel direction $\hat{\mathbf{u}}$. The leading factor normalizes the weights by their sum total

$$W = \sum_{\hat{\mathbf{v}} \in N_k(\hat{\mathbf{u}})} w(\hat{\mathbf{v}}) w_u(\hat{\mathbf{v}}) h_k(\hat{\mathbf{v}})$$

so that c_{k-1} have the same total energy as c_k. At the start of each iteration we set the highest resolution coefficients to the current disparity map estimate $c_K(\hat{\mathbf{u}}) = \hat{d}_i^{(f,t-1)}(\hat{\mathbf{u}})$, for image I_i. We then decompose K levels to get the new disparity estimate in the c_0 scaling coefficients. In subsequent iterations, we feed this result back into the highest resolution scaling coefficients, $\hat{d}_i^{(f,t)}(\hat{\mathbf{u}}) = c_0(\hat{\mathbf{u}})$. The result at the end of the iterations is our final disparity estimate $\hat{d}_i^{(f,T)}$

For the static weight, we use the confidence value from the fusion stage, $w(\hat{\mathbf{v}}) = \tilde{C}_i^{(f)}(\hat{\mathbf{v}})$. The bilateral weights are given by

$$w_u(\hat{\mathbf{v}}) = \exp(-\lambda_d \left| c_k(\hat{\mathbf{v}}) - c_k(\hat{\mathbf{u}}) \right|) \exp(-\lambda_c \left| I_i(\hat{\mathbf{v}}) - I_i(\hat{\mathbf{u}}) \right|) \tag{6.17}$$

where λ_d and λ_c are scaling parameters to penalize differences in disparity and color, respectively. This allows us to avoid smoothing over both color and disparity discontinuities. We use $K = 3$, $\lambda_d = 50$, $\lambda_c = 50$.

6.4 View Synthesis

We synthesize novel views by first generating a colored point cloud from the disparity maps and associated images. We generate a point for every pixel with disparity greater than zero. We subsequently filter the point cloud using a statistical outlier removal. Outliers are determined by computing the median distance of each point to its nearest neighbors, and examining the distribution of these mean distances over the entire point cloud. Points are rejected as outliers using the median distance to their nearest neighbors according to the so-called "X84" rule [156]. This rule uses the Median Absolute Deviation (MAD) as a statistic of the variation of data, which is simply the median of absolute differences from the median of the data. Points

with median nearest neighbor distances greater than k times the MAD of all these median distances over the point cloud are rejected as outliers. We use $k = 5.2$ as recommended by Fusiello et al. [158]. We use 60 nearest neighbors, computed using ANN[4]. This is similar to the statistical outlier removal used by Rusu et al. [159], which uses the mean distance to nearest neighbors and the standard deviation to identify outliers, however, we found the MAD statistic to work slightly better in our case.

Currently, following the removal of outliers from the point cloud, we simply render the point cloud as is, relying on the density from multiple disparity maps to provide good coverage. Future work includes developing a method for in-painting missing gaps in the novel view.

6.5 Experiments

6.5.1 Implementation

Offline Processing: We first apply the pairwise matching technique described in Chapter 5, implemented in CUDA. We simultaneously compute V_{ij} and V_{ji} in the same kernel launch since the epipolar geometry is simply mirrored. Merging the pairwise disparity maps is done on the CPU, as is computing the resulting confidence, because these computations are relatively fast, but .

We adapt visibility-based fusion to spherical disparity maps in CUDA. We begin by warping the other disparity maps to the reference viewpoint using an efficient forward point-splatting technique. Once the warping is done, tests for occlusions and free-space violations can be performed efficiently simply by sampling the warped and un-warped disparity maps.

Online Processing: We rendered the point clouds from OpenGL buffer objects into an off-screen framebuffer. From there gamma correction was applied for final display.

We performed our experiments on a workstation with an Intel Xeon 3.2 GHz, 12 GB of RAM, and an NVidia Quadro 4000 with 2 GB. Using this system, a point cloud with over 10 million points generated from four disparity maps easily renders in real-time (> 30Hz).

6.5.2 Results

In this Section we report results for our complete pipeline: spherical calibration, disparity estimation and view synthesis. We compare our disparity estimation to the graph cut technique of Zheng et al. [15].

[4] `http://www.cs.umd.edu/~mount/ANN/`

Figure 6.4: Two views of the layout of capture positions and novel views for the Tabaret Hall dataset. See text for explanation.

We validate the accuracy of our calibration using reprojection error, measured as the angle between the direction of the feature detected in the input image and the direction of the reconstructed scene point when reprojected into that image. We had a final average reprojection error of $0.021°$, or about a quarter of a pixel in the 3MP images we used. Figure 6.4 shows the layout of the panorama capture positions used for the reconstruction and the positions of the novel views synthesized. The capture locations are shown with red (x), green (y) and blue (z) coordinate axes, while the novel views are shown as viewing frustums (pyramids with the apex at the viewpoint). The cluster of perspective views with the same center of projection in the middle of the capture positions correspond to the novel views against a reference panorama not in the set used to reconstruct the point clouds, as shown in Figures 6.8 through 6.12. The novel view frustums shown in Figure 6.4 are used to generate a smooth path of interpolated views for our video. This path starts with the point of view shown in Figure 6.8. The baseline between capture locations varies, due to imprecision in the PC-bot path following, but is between 10 and 20cm for these capture locations.

The approach most similar to ours, in that it also estimates panoramic disparity layers to generate novel views, is LDP [15]. Zheng et al. use graph cut [74] with edge-sensitive pairwise potentials to estimate disparity layers for LDP. We implemented their disparity estimation as closely as possible for comparison. We use the same parameter values for the smoothing term

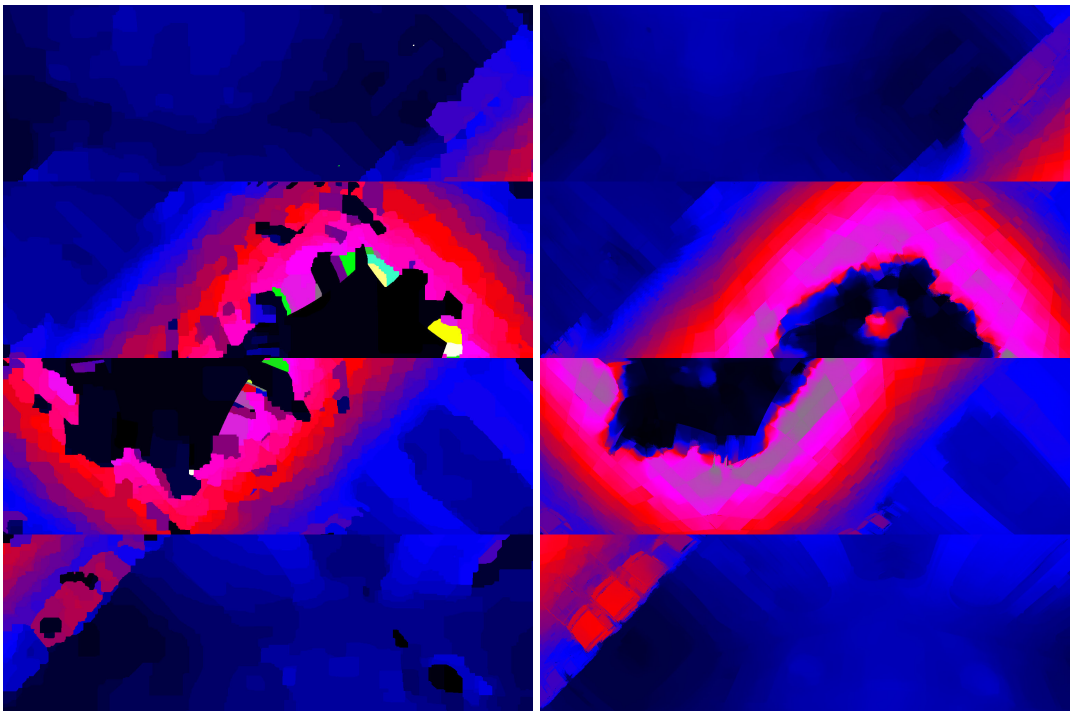

Figure 6.5: Comparison of disparity maps generated by graph cut approach and our method (right).

as in LDP. For the data term we use the robust DSI as in Section 6.2.1, and we adjusted other parameters to find a balance between outliers and over-smoothing. We recognize that the goal of LDP was to provide limited off-axis navigation of tourist-style panoramas, while our method is better suited to immersive scenes.

Figure 6.5 compares the disparity maps obtained with the graph cut approach and with our approach. As can be seen, the discrete optimization with piecewise constant smoothing prior causes over-smoothing on most of the sphere, but leaves abrupt steps in some places. These discontinuities are often more aligned with the sampling grid rather than the image contours despite the edge-sensitive smoothing prior. This is due to only using pairwise constraints. In contrast our multi-resolution filtering takes successively larger filter windows to smooth, while maintaining edge sensitivity. Figure 6.6 shows one of the disparity maps after four stages of our estimation algorithm: pairwise matching, merging, fusion and filtering. The disparity range and color-coding scale is given in Figure 6.1. A perspective close-up is shown in Figure 6.7, where the improvements of each stage can be more easily seen.

Figures 6.8, 6.9, 6.10, 6.11 and 6.12 compare a reference view (top left), taken from a nearby panorama not in the set used to estimate the disparity maps, to synthesized views from

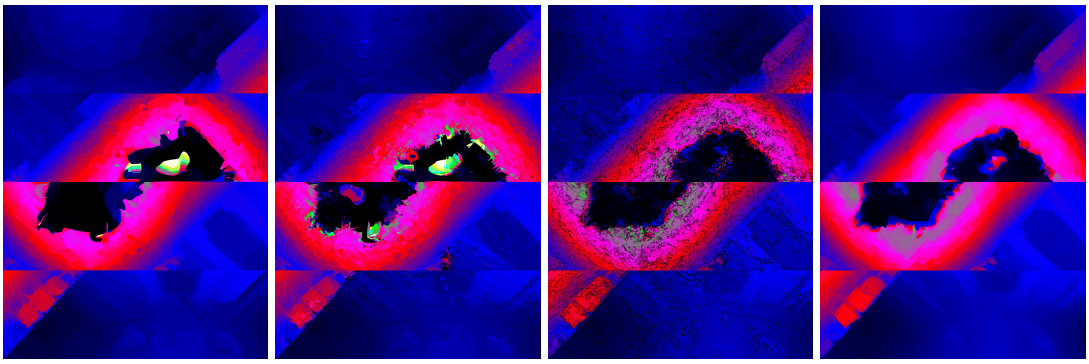

Figure 6.6: Disparity after different stages of our algorithm. Left to right: pairwise matching, merging, fusion, filtering.

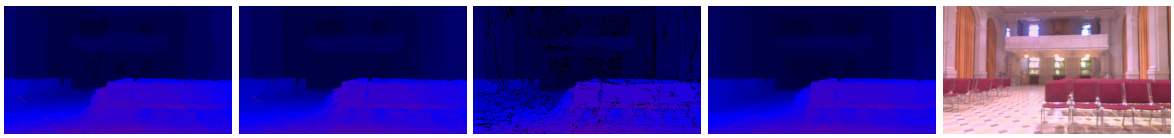

Figure 6.7: Close-up perspective views of disparity maps after different stages of our pipeline as shown in Figure 6.6. The same perspective view of the input image is shown on the right for context. Note how the pairwise disparity estimate is overall accurate, but blurs the disparity from the chairs on the background. This is reduced in both the merge and fusion steps. The filtering steps aligns disparity discontinuities with edge images and smooths over the disparity estimates that were zeroed-out by the fusion stage.

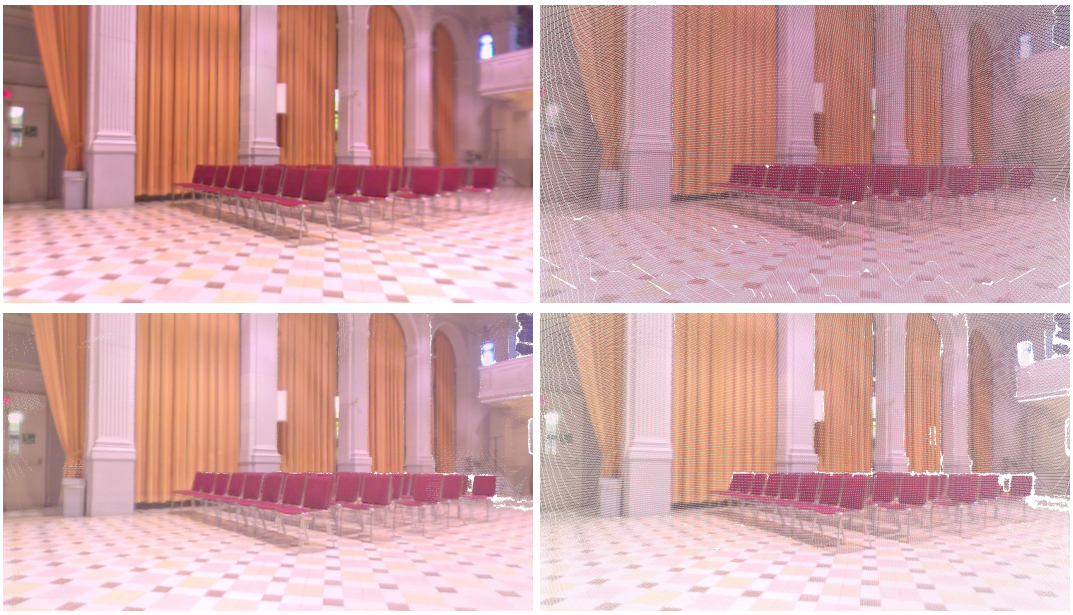

Figure 6.8: Comparison of synthesized novel views. Clockwise from top-left: reference view; point cloud from 1 disparity map computed with graph cuts; point cloud from 1 disparity map computed with our method; point cloud from 4 disparity maps computed with our method.

the same viewpoint. The top right is the view using a single disparity map estimated using graph cuts, bottom right is using a single disparity map estimated with our method, and bottom left shows the view synthesized using 4 disparity maps from our method. Note that the disparity map estimated with piecewise constant model produces a step-function effect resulting in cracks along surfaces slanting away from the camera. Also notice that with the use of the multi-view fusion, we remove many outliers, which could not be removed by applying an additional continuous smoothing step after the graph cut. While gaps in the novel views rendered from the graph cut-generated point clouds occur throughout the view, the gaps due to the point cloud generated from our method occur along object boundaries, and as such can be more easily improved by generating a denser point cloud from additional disparity maps. Note how, by using a point cloud generated from multiple disparity maps, gaps are filled in by disoccluded surfaces and overlapping areas are blended smoothly while preserving detail.

Figure 6.13 compares synthesized views for a different scene. Again, a point cloud generated from a single disparity map estimated using our method improves on a single disparity map point cloud using graph cut by relegating gaps to object boundaries, making them less distracting, and improving the reconstruction within individual objects. Using multiple disparity maps to generate the point cloud greatly reduces the gaps.

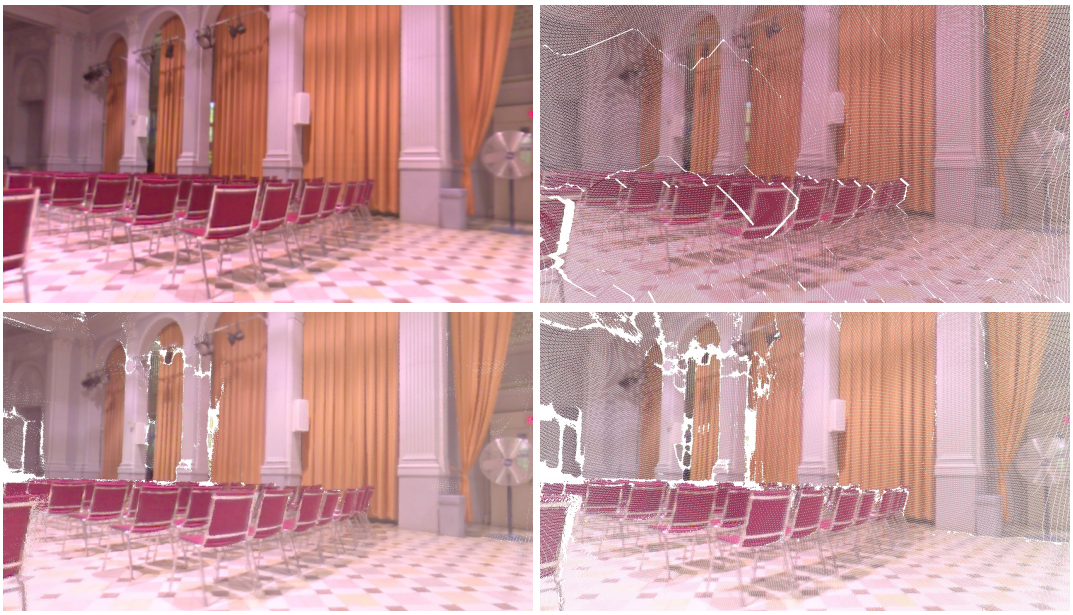

Figure 6.9: Comparison of synthesized novel views. Same ordering as Figure 6.8. This view has significant depth variation and discontinuities. Note how the discontinuities from the graph cut estimation do not correlate with actually depth discontinuities in the scene, where as our method does.

Figure 6.10: Comparison of synthesized novel views. Same ordering as Figure 6.8. This view has many slanting surfaces at different depths. Large gaps appear in the graph cut estimate, whereas our method incurs only small gaps which are corrected by using more disparity maps.

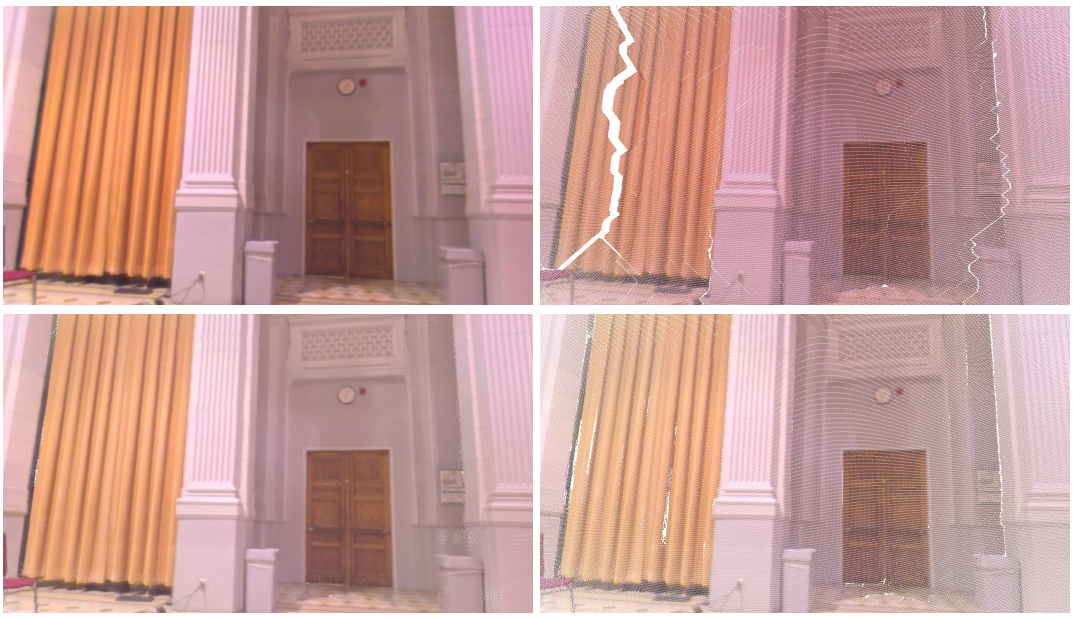

Figure 6.11: Comparison of synthesized novel views. Same ordering as Figure 6.8. Even from only one disparity map, our method produces only small gaps along object boundaries. With four disparity maps, the approximation of the reference view is very close.

Figure 6.12: Comparison of synthesized novel views. Same ordering as Figure 6.8. Aside from avoiding the obvious gaps from the discretely estimated disparity maps, our method better estimates the shape of fine-scale objects. Note especially how the chair legs appear straighter with our method.

Figure 6.13: Comparison of novel views for a different scene. Same ordering as Figure 6.8. Again, with our method, gaps are localized at depth discontinuities.

Both scenes are challenging; they contain lots of close and small occluders, bright lights, specularities and lens effects, and large areas with low texture. Still, we are able to generate convincing walkthroughs as shown in our video. The rendering would likely benefit from boundary matting [14], but we leave this for future work. While for the results presented here we used disparity maps with the same resolution as the input images, if necessary a trade-off can easily be made to improve frame rates on slower machines by reducing the resolution of the disparity maps.

6.6 Discussion

We have presented an effective and efficient system for real-time novel view synthesis using spherical disparity layers. We use the multi-scale stereo matching algorithm presented in Chapter 5 followed by an extension of visibility-based fusion [99] to the sphere. We have presented a novel disparity map filtering technique based on a multi-lateral wavelet decomposition that takes color and robust confidence information into account. We have demonstrated high-quality virtual walkthroughs of real environments, generating real-time views not captured in the input images. The primary future work is to in-paint the gaps in the synthesized views using a model based on the geometry and content of the input images. In the future, it would also be interesting to explore ways to make the estimation phase real-time or near real-time.

Chapter 7

Conclusions

I'm rather surprised that it happened in my lifetime...

Prof. Peter Higgs, July 4, 2012

In this thesis we have addressed several problems in multi-scale spherical image processing, planar and spherical binocular stereo, planar and spherical multi-view stereo, novel view synthesis from stereo-reconstructed point clouds, and multi-scale statistical shape analysis. We have addressed these problems through the common framework of multi-scale representations and algorithms, and demonstrated how these techniques can be adapted to both the planar and spherical domains. We have proposed novel solutions to these problems.

7.1 Thesis Summary

We now reiterate the specific contents of this thesis by chapter. Each chapter tackles a different problem, but they all fit within the themes of spherical representations, and multi-scale representations and algorithms in computer vision. All chapters are directly or indirectly related to the tasks of multi-view stereo reconstruction and real-time novel view synthesis. This thesis is comprised of the following:

- In Chapter 3, we developed a tool for spherical image and geometry processing in the form of a GPU implementation of fast spherical harmonic transforms and frequency-domain convolutions of spherical images. Our method was based on the semi-naive

algorithm of Healy et al. [31], and implemented in CUDA. We have shown how this tool can be used to quickly generate an overcomplete spherical wavelet representation of a spherical function. Such a representation has been shown to be useful for geometry processing as well, in the form of analyzing cortical folding in MRI scans of the brain [28]. We believe this technique can also be useful for panoramic images as well. A preliminary version of this chapter was published in the journal of graphics, gpu, and game tools [113].

- In Chapter 4, we learn the shape variation of the human face in the wavelet domain, and use the resulting statistical information to constrain stereo reconstruction, or to extract human face shapes from point clouds. The localizing and decorrelating properties of the spherical wavelet transform allows us to treat our shape parameters independently, which allows us to capture greater variation than exists in the training set and makes exhaustive global search practical in the optimization procedure when estimating a new face shape. While we demonstrate our method for human faces, it can be applied to any shape that can be mapped to a sphere. The reconstructed shapes are already in correspondence with the training set. A preliminary version of this chapter was published in 2011 proceedings of the Canadian Conference on Computer and Robot Vision [126].

- In Chapter 5 we develop a multi-scale binocular stereo matching approach based on a novel application of distance transforms to disparity space images (DSI) at different scales. This allows us to avoid making hard, greedy decisions at coarser scales that might prevent us from finding the correct disparity a finer scales, while retaining efficiency, both in terms of computation time and storage space. We further extended this approach from the planar images to spherical images with a novel disparity formulation, which, unlike previous disparity formulations for spherical images, allows geometrically correct smoothing and filtering of disparity values and DSI values, allows arbitrary sampling of the spherical images, and can be directly extend to multi-view stereo. We obtain nearly uniform sampling of the image domain and the disparity space, whereas previous formulations sample the images most densely near the epipoles. Our multi-scale matching framework operates independently for each pixel in the disparity map, and hence can be extended to any sampling of any image domain. Extensive evaluation was performed, including quantitative evaluations against a public benchmark for planar stereo, which demonstrated state-of-the-art performance. Further, the multi-scale framework could incorporate any single-scale DSI computation technique. A version of this chapter has been accepted and will appear at the International Conference on 3D Imaging and

Modeling, Processing, Visualization and Transmission (3DIMPVT) 2012.

- In Chapter 6 we extend the binocular stereo matching approach of Chapter 5 to multi-view stereo, and demonstrate the effectiveness of this approach for real-time novel view synthesis from point clouds generated from spherical disparity maps. This includes the extension of multi-view visibility-based fusion to spherical images and disparity, and the further development of a novel fusion and refinement stage. This chapter also discussed the development of a spherical bundle-adjustment based on a non-Euclidean reprojection error. We demonstrate real-time novel view synthesis with a smooth virtual walkthrough a real environment captured with a set of panoramic images. We plan to submit a version of this chapter to Eurographics 2013.

In summary, this thesis has made several contributions to multi-scale image processing and computer vision for both planar and spherical images, and has even made some contributions to statistical geometry analysis and processing. This thesis has proposed a number of strategies for reconstructing geometry from images that build upon the state-of-the-art, but incorporate new insights. This thesis has addressed problems in omnidirectional imaging and vision because this is a potentially very practical and efficient modality for capturing environments and reconstructing the geometry thereof. As previous research has noted, the wide field-of-view of omnidirectional images can provide better stabilization of structure-from-motion and therefore more robustly estimated camera positions and orientations. This accuracy propagates through to the stereo reconstruction stage. Throughout this thesis, a theme of structured multi-scale representations and algorithms has been followed. These techniques provide scalability as image sizes increase, and non-local regularization that does not necessarily depend on directly enforcing smoothing constraints.

7.2 Thesis Contributions

This thesis makes several contributions to the state-of-the-art in spherical image processing, computer vision and geometry processing. These contributions address some of the gaps in the existing literature of these fields as laid out in Chapter 2. The contributions of this thesis are, in chronological order:

- An efficient GPU implementation of the semi-naive algorithm [31] for spherical harmonic transforms, with a GPU implementation for frequency-domain convolution of spherical functions.

- A statistical spherical wavelet shape prior for object-specific point-cloud parameterization and model-based stereo.
- A sampling-based Bayesian framework for parameterizing point clouds using the wavelet shape prior for any class of shapes that can be meshed using a compatible subdivision scheme.
- A novel application of distance transforms for time- and space-efficient multi-scale pairwise image matching that avoids hard, greedy decisions at coarse levels that prevent finding the correct fine scale match.
- A novel formulation of disparity for spherical images based on two disparity quantities: radial disparity and angular disparity. By sampling in radial disparity and mapping to angular disparity we maintain greater flexibility than previous spherical disparity formulations. We can use any underlying sampling of the spherical images. By combining this disparity formulation with the RD-map sampling scheme, we obtain a nearly uniform sampling of the spherical disparity space. Previous spherical disparity formulations actually sample the images and disparity maps most densely in directions with the least parallax.
- A novel multi-view stereo algorithm for both planar and spherical images. This algorithm is based on multi-scale pairwise stereo matching, efficiently and robustly merging pairwise disparity maps, an extension of multi-view visibility constraints to spherical disparity maps, and a novel fusion and filtering stage.
- A novel spherical bundle adjustment, based on spherical (angular) reprojection error rather than the standard Euclidean error.
- A system for generating real-time virtual walkthroughs of real environments from a set of panoramic images of the environments, based on the other contributions of this thesis. This includes an offline preprocessing stage to calibrate the images, estimate disparity and reconstruct point clouds, and an online stage in which novel views are synthesized in real-time (*i.e.*, at video rates).

While the work in this thesis has been motivated by specific applications, such as face capture or real-time novel view synthesis, the contributions of this thesis imply a broader range of applications, including object-specific point cloud parameterization, human-computer interaction, passive geometry sensing, robot navigation, and even autonomous driving.

7.3 Future Work

We can build upon the work done in this thesis in the following ways. The GPU implementation of the spherical harmonic transform, while already efficient and offering significant acceleration over serial algorithms, can be further improved using the concept of a computational plan, wherein threads are assigned to coefficients more judiciously. Some work has already been done on improving the performance of the wavelet shape prior, but this can be additionally improved by a better re-meshing stage, using, *e.g.*, a subdivision shrink-wrapping approach [160]. Additionally, expression variation can be included in the model using, *e.g.*, a multi-linear model [161]. The multi-scale image matching can be made more efficient by not considering disparity values that have no hope of minimizing the multi-scale energy function for a given pixel. Such an optimization might make the approach suitable for optical flow or motion estimation. Another dissimilarity measure, such as mutual information [71], could also be used within the multi-scale framework to compute the DSI at a single scale. This has the potential to make the matching more robust, especially under illumination and exposure changes. It would also be interesting to combine the fast spherical harmonic transform of Chapter 3 with existing methods for computing SIFT features [98] for the sphere [114] and the calibration method of Chapter 6 for a fast, wide-baseline spherical calibration method. The most interesting and important future work involves improving the novel view synthesized with the point clouds reconstructed in Chapter 6. This includes more sophistical particle-splatting techniques, possibly along the lines of layered relief textures [162], and filling in the gaps via some form of in-painting. Ideally, such an approach would construct a localized generative model of the input images that incorporates the multi-view camera geometry, and sample from this model to fit to the rendered point cloud.

Bibliography

[1] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," in *IEEE International Conference on Computer Vision and Pattern Recongition (CVPR)*, 2007.

[2] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[3] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen, "On benchmarking camera calibration and multi-view stereo for high resolution imagery," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[4] E. Adelson and J. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, M. Landy and J. Movshon, Eds. MIT Press, 1991, pp. 3–20.

[5] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1995, pp. 39–46.

[6] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006, pp. 519–526.

[7] S. Chen and L. WIlliams, "View interpolation for image synthesis," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, August 1993, pp. 279–288.

[8] S. M. Seitz and C. M. Dyer, "View morphing," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, August 1996, pp. 21–30.

[9] H.-Y. Shum and L.-W. He, "Rendering with concentric mosaics," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, August 1999, pp. 299–306.

[10] D. Aliaga and I. Carlbom, "Plenoptic stitching: A scalable method for reconstructing 3D interactive walkthroughs," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001, pp. 443–450.

[11] D. Aliaga, T. Funkhouser, D. Yanovsky, and I. Carlbom, "Sea of images," in *IEEE Visualization*, 2002, pp. 331–338.

[12] J. Kopf, B. Chen, R. Szeliski, and M. Cohen, "Street slide: Browsing street level imagery," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 29, no. 3, 2010.

[13] J. Shade, S. J. Gortler, L. wei He, and R. Szeliski, "Layered depth images," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1998.

[14] C. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 23, no. 3, 2004.

[15] K. Zheng, S. Kang, M. Cohen, and R. Szeliski, "Layered depth panoramas," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[16] A. Hornung and L. Kobbelt, "Interactive pixel-accurate free viewpoint rendering from images with silhouette aware sampling," *Computer Graphics Forum*, vol. 28, no. 8, pp. 2090–2103, 2009.

[17] M. Goesele, J. Ackermann, S. Fuhrmann, C. Haubold, R. Klowsky, D. Steedly, and R. Szeliski, "Ambient point clouds for view interpolation," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 29, no. 3, 2010.

[18] A. Fitzgibbon, Y. Wexler, and A. Zisserman, "Image-based rendering using image-based priors," in *IEEE International Conference on Computer Vision (ICCV)*, 2003.

[19] O. Woodford, I. Reid, and A. Fitzgibbon, "Efficient new-view synthesis using pairwise dictionary priors," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[20] W. Li and B. Li, "Joint conditional random field of multiple views with online learning for image-based rendering," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[21] H. Ishigura and S. Tsuji, "Omni-directional stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, 1992.

[22] S. Kang and R. Szeliski, "3D scene data recovery using omnidirectional multi-baseline stereo," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1996.

[23] B. Micusik and J. Kosecka, "Piecewise planar city 3D modeling from streetview panoramic sequences," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 2906–2912.

[24] S. Li, "Real-time spherical stereo," in *International Conference on Pattern Recognition*, 2006.

[25] H. Kim and A. Hilton, "3D environment modelling using spherical cross-slits stereo imaging," in *IEEE International Conference on 3-dimensional Digtial Imaging and Modeling (3DIM) (ICCV Workshop)*, 2009.

[26] ——, "3D modelling of static environments using multiple spherical stereo," in *Reconstruction and Modeling of Large-Scale 3D Virtual Environments (RMLE) (ECCV Workshop)*, 2010.

[27] A. Pagani, C. Gava, Y. Cui, B. Krolla, J.-M. Hengen, and D. Stricker, "Dense 3D point cloud generation from multiple high-resolution spherical images," in *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, 2011.

[28] P. Yu, B. T. T. Yeo, P. E. Grant, B. Fischl, and P. Golland, "Cortical folding development study based on over-complete spherical wavelets," in *IEEE Internation Conference on Computer Vision (ICCV)*, 2007.

[29] R. Szeliski and D. Scharstein, "Sampling the disparity space image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, pp. 419–425, March 2004.

[30] *CUDA Programming Guide*, 2010.

[31] D. Healy, Jr., D. Rockmore, P. J. Kostelec, and S. S. B. Moore, "FFTs for the 2-sphere – improvements and variations," *Journal of Fourier Applications*, vol. 9, no. 4, 2003.

[32] M. Bertram, M. Duchaineau, B. Hamann, and K. I. Joy, "Generalized B-Spline subdivision-surface wavelets for geometry compression," *IEEE Transactions on Visualization and Graphics*, vol. 10, no. 3, pp. 326–338, 2004.

[33] C. Jekeli, J. K. Lee, and J. H. Kwon, "On the computation and approximation of ultra-high-degree spherical harmonic series," *Journal of Geodesy*, vol. 81, pp. 603–615, 2007.

[34] Y. Li, T.-S. Tan, I. Volkau, and W. L. Nowinski, "Model-guided segmentation of 3D neuroradiological image using statistical surface wavelet model," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[35] R. Green, "Spherical harmonic lighting: The gritty details," in *Game Development Conference*, January 2003.

[36] R. Ramamoorthi and P. Hanrahan, "Frequency space environment map rendering," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 21, no. 3, pp. 527–536, 2002.

[37] J. Kautz, P.-P. Sloan, and J. Snyder, "Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics," in *Eurographics Workshop on Rendering*, 2002.

[38] P.-P. Sloan, J. Kautz, and J. Snyder, "Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 21, no. 3, pp. 527–536, 2002.

[39] L. Zhang and D. Samaras, "Face recognition from a single training image under arbitrary unknown lighting using spherical harmonics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 351–363, 2006.

[40] Q. Wang, O. Ronneberger, and H. Burkhardt, "Rotational invariance based on Fourier analysis in polar and spherical coordinates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, 2009, accepted.

[41] J. R. Driscoll and D. M. Healy, Jr., "Computing Fourier transforms and convolutions on the 2-sphere," *Advances in Applied Mathematics*, vol. 15, pp. 202–250, 1994.

[42] "Spherical harmonics," in *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[43] Gradshteyn and Ryzhik, *Table of integrals, series and products*. New York: Academic Press, 1980.

[44] P.-P. Sloan, "Stupid spherical harmonics (SH) tricks," in *Game Development Conference*, 2008.

[45] B. Yeo, W. Ou, and P. Golland, "On the construction of invertible filter banks on the 2-sphere," *IEEE Transactions on Image Processing*, vol. 17, no. 3, 2008.

[46] P. Schroder and W. Sweldens, "Spherical wavelets: Efficiently representing functions on the sphere," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1996.

[47] C. Lessig and E. Fiume, "SOHO: Orthogonal and symmetric haar wavelets on the sphere," *ACM Transactions on Graphics*, vol. 27, no. 1, 2008.

[48] R. Fattal, "Edge-avoiding wavelets and their applications," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, pp. 22:1–22:10, 2009.

[49] P. Dutilleux, "An implementation of the "algorithme à trous" to compute the wavelet transform," in *Wavelets: time-frequency methods and phase space : proceedings of the international conference*, J.-M. Combes, A. Grossman, and P. Tchamitchian, Eds. Springer-Verlag, 1989, pp. 298–304.

[50] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform," in *Wavelets: time-frequency methods and phase space : proceedings of the international conference*, J.-M. Combes, A. Grossman, and P. Tchamitchian, Eds. Springer-Verlag, 1989, pp. 289–297.

[51] H. Dammertz, D. Sewtz, J. Hanika, and H. Lensch, "Edge-avoiding $\grave{a}$-trous wavelet transform for fast global illumination filtering," in *ACM SIGGRAPH Symposium on High Performance Graphics*, 2010.

[52] J. Hanika, H. Dammertz, and H. Lensch, "Edge-optimized $\grave{a}$-trous wavelets for local contrast enhancement with robust denoising," in *Pacific Graphics*, 2011.

[53] F. Losasso, H. Hoppe, S. Schaefer, and J. Warren, "Smooth geometry images," in *Eurographics Symposum on Geometry Processing (SGP)*, 2003, pp. 138–145.

[54] C. Peirce, "A quincuncial projection of the sphere," *American Journal of Mathematics*, vol. 2, no. 4, pp. 394–396, 1879.

[55] C.-W. Fu, L. Wan, T.-T. Wong, and C.-S. Leung, "The rhombic dodecahedron map: An efficient scheme for encoding panoramic video," *IEEE Transactions on Multimedia*, vol. 11, no. 4, June 2009.

[56] P. Dutre, K. Bala, and P. Bekaert, *Advanced Global Illumination*. AK Peters, 2006.

[57] S. Mallat, "Zero-crossings of a wavelet transform," *IEEE Transactions On Information Theory*, vol. 37, no. 4, pp. 1019–1033, 1991.

[58] E. Simoncelli, W. Freeman, E. Adelson, and D. Heeger, "Shiftable multi-scale transforms," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 587–607, 1992.

[59] J.-W. Hsieh, H.-Y. Liao, K.-C. Fan, M.-T. Ko, and Y.-P. Hung, "Image registration using a new edge-based approach," *Computer Vision and Image Understanding*, vol. 67, no. 2, pp. 112–130, 1997.

[60] J. Magarey and A. Dick, "Multiresolution stereo image matching using complex wavelets," in *International Conference on Pattern Recognition (ICPR)*, vol. 1, 1998.

[61] M. Shim, "Wavelet-based stereo vision," *Lecture Notes in Computer Science Biologically Motivated Computer Vision*, vol. 1811/2000, pp. 335–385, 2000.

[62] F. Shi, N. Hughes, and G. Roberts, "SSD matching using shift-invariant wavelet transform," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2001.

[63] G. Caspary and Y. Zeevi, "Wavelet-based multiresolution stereo vision," in *International Conference on Pattern Recognition (ICPR)*, vol. 3, 2002.

[64] C. Liu, W. Pei, S. Niyokindi, J. Song, and L. Wang, "Micro stereo matching based on wavelet transform and projective invariance," *Measurement Science and Technology*, vol. 17, 2006.

[65] A. Bhatti and S. Nahavandi, *Stereo Correspondence Estimation based on Wavelets and Multiwavelets Analysis*, 2008.

[66] J. Li, H. Zhao, X. Zhou, and C. Shi, "Robust stereo image matching using a two-dimensional monogenic wavelet transform," *Optics Letters*, vol. 34, pp. 3514–3516, 2009.

[67] W. Miled, J. Pesquet, and M. Parent, "Wavelet-constrained regularization for disparity map estimation," in *European Signal and Image Processing Conference*, 2006.

[68] I. Geys and L. V. Gool, "Hierarchical coarse to fine depth estimation for realistic view interpolation," in *IEEE International Conference on 3-dimensional Digital Imaging and Modeling (3DIM)*, 2005.

[69] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision (ACCV)*, 2010.

[70] J. Kim and T. Sikora, "Gaussian scale-space dense disparity estimation with anisotropic disparity-field diffusion," in *IEEE International Conference on 3-dimensional Digital Imaging and Modeling (3DIM)*, 2005.

[71] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[72] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross, "High-quality single-shot capture of facial geometry," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 29, no. 3, 2010.

[73] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, 2003.

[74] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, 2001.

[75] P. Felzenzwalb and D. Huttenlocher, "Efficient belief propagation for early vision," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[76] P. Debevec, C. Taylor, and J. Malik, "Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1996.

[77] B. Amberg, A. Blake, A. Fitzgibbon, S. Romdhani, and T. Vetter, "Reconstructing high quality face-surfaces using model based stereo," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[78] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3D faces," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1999, pp. 187–194.

[79] Y. Sun, P. Kohli, M. Bray, and P. Torr, "Using strong shape priors for stereo," in *Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, 2006, pp. 882–893.

[80] F. Romeiro and T. Zickler, "Model-based stereo with occlusions," in *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, 2007.

[81] M. Tonko and H.-H. Nagel, "Model-based stereo-tracking of non-polyhedral objects for automatic disassembly experiments," *International Journal of Computer Vision*, vol. 37, pp. 99–118, 2000.

[82] W. Zhao, N. Nandhakumar, and P. Smith, "Model-based interpretation of stereo imagery of textured surfaces," *Machine Vision and Applications*, vol. 10, pp. 201–213, 1997.

[83] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer, "High resolution passive facial performance capture," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 29, no. 3, 2010.

[84] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W. Sumner, and M. Gross, "High-quality passive facial performance capture using anchor frames," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 30, pp. 75:1–75:10, 2011.

[85] Y. Furukawa and J. Ponce, "Dense 3D motion capture for human faces," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[86] L. Williams, "Performance-driven facial animation," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1990, pp. 235–242.

[87] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin, "Making faces," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1998, pp. 55–66.

[88] I.-C. Lin and M. Ouhyoung, "Mirror mocap: Automatic and efficient capture of dense 3D facial motion parameters from video," *The Visual Computer*, vol. 21, no. 6, pp. 355–372, 2005.

[89] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross, "Multi-scale capture of facial geometry and motion," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 26, no. 3, 2007.

[90] S. Seitz and C. Dyer, "Photorealistic scene reconstruction by voxel coloring," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 151–173, 1999.

[91] K. Kutulakos and S. Setiz, "A theory of shape by space carving," *International Journal of Computer Vision*, vol. 38, no. 3, pp. 199–218, 2000.

[92] G. Vogiatzis, P. Torr, and R. Cipolla, "Multi-view stereo via volumetric graph cuts," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 391–398.

[93] O. Faugeras and R. Keriven, "Variational principles, surface evolution, PDE's, level set methods and the stereo problem," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 336–344, 1998.

[94] J.-P. Pons, R. Keriven, and O. Faugeras, "Modelling non-rigid dynamic scenes from multi-view image sequences," in *Handbook of Mathematical Models in Computer Vision*, N. Paragios, Y. Chen, and O. Faugeras, Eds. Springer, 2006.

[95] H.-H. Vu, R. Keriven, P. Labatut, and J.-P. Pons, "Towards high-resolution large-scale multi-view stereo," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[96] M. Goesele, B. Curless, and S. Seitz, "Multi-view stereo revisited," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[97] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. Seitz, "Multi-view stereo for community photo collections," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[98] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[99] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nister, and M. Pollefeys, "Real-time visibility-based fusion of depth maps," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[100] G. Zhang, J. Jia, T.-T. Wong, and H. Bao, "Consistent depth maps recovery from a video sequence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 6, pp. 974–988, June 2009.

[101] D. Bradley, T. Boubekeur, and W. Heidrich, "Accurate multi-view reconstruction using robust binocular stereo and surface meshing," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[102] Y. Li, H.-Y. Shum, C.-K. Tang, and R. Szeliski, "Stereo reconstruction from multiperspective panoramas," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, 2004.

[103] H.-Y. Shum, M. Han, and R. Szeliski, "Interactive construction of 3D models from panoramic mosaics," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998.

[104] J. Fujiki, A. Torii, and S. Akaho, "Epipolar geometry via rectification of spherical images," in *MIRAGE*, 2007.

[105] L. Bagnato, P. Frossard, and P. Vanderhyesnst, "A variational framework for structure from motion in omnidirectional image sequences," *Journal of Mathematical Imaging and Vision*, vol. 41, no. 3, 2011.

[106] H.-Y. Shum, S. B. Kang, and S.-C. Chan, "Survey of image-based representations and compression techniques," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 11, pp. 1020–1037, November 2003.

[107] M. Levoy and P. Hanrahan, "Light field rendering," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1996.

[108] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen, "The lumigraph," in *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1996, pp. 43–54.

[109] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall, "Mosaicing new views: The crossed-slits projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, 2003.

[110] C. Lei, X. Chen, and Y. Yang, "A new multiview spacetime-consistent depth recovery framework for free viewpoint video rendering," in *IEEE International Conference on Computer Vision (ICCV)*, 2009.

[111] T. Takahashi, H. Hawasaki, K. Ikeuchi, and M. Sakauchi, "Arbitrary view position and direction rendering for large-scale scenes," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[112] Z. Zhu, G. Xu, and X. Lin, "Panoramic EPI generation and analysis of video from a moving platform with vibration," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[113] A. Brunton, J. Lang, and E. Dubois, "Spherical harmonic transforms and convolutions on the gpu," *journal of graphics, gpu, and game tools*, vol. 15, no. 1, pp. 13–27, 2010.

[114] J. Cruz-Mota, I. Bogdanova, B. Paquier, M. Bierlaire, and J.-P. Thiran, "Scale invariant feature transform on the sphere: Theory and applications," *International Journal of Computer Vision*, vol. 98, no. 2, pp. 217–241, 2012.

[115] K. Grski, E. Hivon, A. Banday, B. Wandelt, F. Hansen, M. Reinecke, and M. Bartelmann, "Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere," *The Astrophysical Journal*, vol. 622, no. 2, pp. 759–771.

[116] J. C. Adams and P. N. Swarztrauber, "Spherepack 2.0: A model development facility," NCAR, Tech. Rep. NCAR/TN-436-STR, September 1997.

[117] M. J. Mohlenkamp, "A fast transform for spherical harmonics," *Journal of Fourier Analysis and Applications*, vol. 5, no. 2/3, 1999.

[118] R. Suda and M. Takami, "A fast spherical harmonics transform algorithm," *Mathematics of Computation*, vol. 71, no. 238, pp. 703–715, November 2001.

[119] Shtools. [Online]. Available: http://www.ipgp.jussieu.fr/~wieczor/SHTOOLS/SHTOOLS.html

[120] Spharmonickit. [Online]. Available: http://www.cs.dartmouth.edu/~geelong/sphere/

[121] J. Keiner, S. Kunis, and D. Potts, "Using NFFT 3 - a software library for various nonequispaced fast Fourier transforms," *ACM Transactions on Mathematical Software*, vol. 36, no. 4, pp. 19:1–19:30, 2009.

[122] ccsht. [Online]. Available: http://crd.lbl.gov/~cmc/ccSHTlib/doc/

[123] M. Vetterli and H. Nussbaumer, "Simple FFT and DCT algorithms with reduced number of operations," *Signal Processing*, vol. 6, pp. 267–278, 1984.

[124] N. Govindaraju, B. Lloyd, Y. Dotsenko, B. Smith, and J. Manferdelli, "High performance discrete Fourier transforms on graphics processors," in *ACM/IEEE Conference on Supercomputing*, 2008.

[125] A. Nukada, Y. Ogato, T. Endo, and S. Matsuoka, "Bandwidth intensive 3-d FFT kernel for GPUs using CUDA," in *ACM/IEEE Conference on Supercomputing*, 2008.

[126] A. Brunton, C. Shu, J. Lang, and E. Dubois, "Wavelet model-based stereo for fast, robust face reconstruction," in *Canadian Conference on Computer and Robot Vision*, 2011.

[127] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal Computer Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[128] S. Koterba, S. Baker, I. Matthews, C. Hu, J. Xiao, J. Cohn, and T. Kanade, "Multi-view AAM fitting and camera calibration," in *IEEE International Conference on Computer Vision (ICCV)*, 2005.

[129] S. Essafi and G. Langs, "Hierarchical 3D diffusion wavelet shape priors," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 1717–1724.

[130] Y. Chen and R. Cipolla, "Learning shape priors for single view reconstruction," in *International Conference on 3-dimensional Digital Imaging and Modeling (3DIM) (ICCV Workshop)*, 2009, pp. 1425–1432.

[131] M. Levine and Y. Yu, "State-of-the-art of 3D facial reconstruction methods for face recognition based on a single 2d training image per person," *Pattern Recognition Letters*, vol. 30, pp. 908–913, 2009.

[132] I. Dryden and K. Mardia, *Statistical Shape Analysis*. Wiley, 2002.

[133] N. Snavely, S. Seitz, and R. Szeliski, "Photo tourism: Exploring image collections in 3D," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 25, no. 3, pp. 835–846, 2006.

[134] V. Kolmogorov, "Graph based algorithms for scene reconstruction from two or more views," Ph.D. dissertation, Cornell University, 2003.

[135] K. Robinette, H. Daanen, and E. Paquet, "The CAESAR project: A 3-D surface anthropometry survey," in *IEEE International Conference on 3-dimensional Digital Imaging and Modeling (3DIM)*, 1999, pp. 180–186.

[136] L. Yin, X. We, Y. Sun, J. Wang, and M. Rosato, "A 3D facial expression database for facial behavior research," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 2006, pp. 211–216.

[137] A. Salazar, S. Wuhrer, S. Chu, and F. Prieto, "Fully automatic expression-invariant face correspondence," arXiv:1202:1444, Tech. Rep., 2012.

[138] A. Patel and W. Smith, "Exploring the identity manifold: Constrained operations in face space," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 112–125.

[139] A. Savran, N. Alyuz, H. Dibeklioglu, O. Celiktutan, B. Gökberk, B. Sankur, and L. Akarun, "Bosphorus database for 3D face analysis," in *Workshop on Biometrics and Identity Management*, 2008, pp. 47–56.

[140] A. Brunton, J. Lang, and E. Dubois, "Efficient multi-scale stereo of high-resolution planar and spherical images," in *IEEE International Conference on 3D Digital Imaging, Modeling, Processing, Visualization and Transmission (3DimPVT)*, 2012.

[141] M. Lew and T. Huang, "Optimal multi-scale matching," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[142] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[143] P. Felzenzwalb and D. Huttenlocher, "Distance transforms of sampled functions," Cornell University, Tech. Rep., 2004.

[144] T. Yu, R.-S. Lin, B. Super, and T. Tang, "Efficient message representations for belief propagation," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[145] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[146] K. He, J. Sun, and X. Tang, "Guided image filtering," in *European Conference on Computer Vision (ECCV)*, 2010.

[147] K. Yoon and S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 650–656, 2006.

[148] H. Hirschmüller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[149] J. Kostkova, "Stratified dense matching for stereopsis in complex scenes," in *British Machine Vision Conference*, 2003.

[150] K. Yamaguchi, T. Hazan, D. McAllester, and R. Urtasun, "Continuous markov random fields for robust stereo estimation," in *European Conference on Computer Vision (ECCV)*, 2012.

[151] M. Aly and J.-Y. Bouget, "Street view goes indoors: Automatic pose estimation from uncalibrated unordered spherical panoramas," in *IEEE Workshop on the Applications of Computer Vision*, 2012.

[152] M. A. Lourakis and A. Argyros, "SBA: A Software Package for Generic Sparse Bundle Adjustment," *ACM Transactions on Mathematical Software*, vol. 36, no. 1, pp. 1–30, 2009.

[153] F. Kangni and R. Laganière, "Orientation and pose estimation of panoramic imagery," *Machine Graphics and Vision*, vol. 19, no. 3, pp. 339–363, 2010.

[154] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[155] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[156] F. Hampel, E. Ronchetti, P. Rousseeuw, and W. Stahel, *Robust Statistics: The Approach Based on Influence Functions*, ser. Wiley Series in Probability and Statistics. John Wiley & Sons, 1986.

[157] D. C. Liu and J. Nocedal, "On the limited memory method for large scale optimization," *Mathematical Programming*, vol. 45, no. 3, pp. 503–528, 1989.

[158] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto, "Improving feature tracking with robust statistics," *Pattern Analysis and Applications*, vol. 2, no. 4, pp. 312–320, 1999.

[159] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems Journal (Special Issue on Semantic Knowledge)*, vol. 56, no. 11, pp. 927–941, 2008.

[160] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel, "A shrink wrapping approach to remeshing polygonal surfaces," *Computer Graphics Forum (Proc. Eurographics)*, vol. 18, no. 3, 1999.

[161] D. Vlasic, M. Brand, H. Pfister, and J. Popović, "Face transfer with multilinear models," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 24, no. 3, 2005.

[162] S. Parilov and W. Stuerzlinger, "Layered relief textures," *Journal of WSCG*, 2002.