NOTE TO USERS

This reproduction is the best copy available.

uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Ting Yu

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Stereo-Based 3D Model Acquisition and Motion Detection

TITRE DE LA THÈSE / TITLE OF THESIS

Jochen Lang

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

Chris Joslin

Robert Langanière

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Stereo-Based 3D Model Acquisition and Motion Detection

by

Ting Yu

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

School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-74199-3
Our file Notre référence
ISBN: 978-0-494-74199-3

Abstract

Deformable models have a long tradition in computer graphics and computer vision. This thesis looks at the capture of surface deformation based on stereo vision. In recent years, 3D reconstruction and motion detection has attracted great attention. In this thesis a framework for 3D reconstruction from mutli-view images followed by isometry-based motion detection is proposed. For 3D reconstruction, the thesis proposes a multi-view stereo algorithm based on well-known window-based matching combined with fusion of multiple matching results. To improve the matching result, some low-level image processing algorithms, camera calibration and background detection are utilized. For window-based matching, a new hybrid matching method is introduced by combining both, a measure of intensity difference and intensity distribution difference. Multiple MVS pointclouds from different reference views are fused with two new fusion strategies to generate a better final reconstruction. To characterize the performance of our matching method and fusion strategies, an evaluation based on the quality of reconstruction is given in the thesis. Based on 3D pointclouds of object surface obtained with stereo, the deformation of the surface is captured. To generate dense motion vectors over a deformed surface, a simple window-based 3D flow method is applied by using isometry of the observed surface as its primary matching constraint. The method uses feature points as anchoring references of the surface deformation. Given a set of matched features no other intensity information is used and hence the method can tolerate intensity changes over time. The approach is shown to work well on two example scenes which capture non-rigid isometric and general deformations. The thesis also presents experiments demonstrating the stability of the geodesic approximation employed in the isometry-based matching when the 3D pointclouds are sparse.

Acknowledgements

First and foremost I would like to show my deepest gratitude to my supervisor Dr. Jochen Lang for his invaluable guidance and patience during my whole Master's study and his constructive criticism through the thesis writing.

Then, I would like to thank my father and my mother for their constant support from the other side of the earth. And I would like to offer special thanks to my girlfriend Jie Cheng for her understanding and patience during these years.

And I also would like to thank all my friends at the DISCOVER Lab for giving me an immensely enjoyable experience during this educational journey.

Last but not the least, I would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for the financial support of this research.

Contents

List of Tables

List of Figures

Glossary of Terms

2D Two Dimensional

3D Three Dimensional

MVS Multi-View Stereo

PC PointCould

SSD Sum of Squared Difference

SAD Sum of Absolute Difference

NCC Normalized Cross Correlation

KLT Kanade-Lucas-Tomasi

Chapter 1

Introduction

1.1 Background

In recent years, surface motion detection has attracted a lot of attention in many areas such as animation, physics-based modeling, interaction capture, etc. Its broad application areas range from monitoring of plant growth in biology, e.g., [64, 57], analyzing the deformation of human skin, e.g., [54] to advanced motion capture for the entertainment industry, e.g., [1]. In this thesis, we are interested in building a motion detection model by estimating a 3D flow over a deformed surface in unconstrained settings with dynamic boundary conditions. The estimation is fed by a set of tracked feature points during the motion. Combining the tracked feature points and the assumption of distance preservation, the 3D flow over an observed object surface is calculated by using a isometry-based matching. In more general terms, our goal is to track a 3D pointcloud during a deformation of a real world object. The 3D pointcloud is a reconstruction model of the real world object.

Deformable objects have been studied in computer vision and computer graphics for a long time [68] and 3D flow techniques with either intensity or range images have often been used to capture shape changes. The 3D flow is a 3D velocity field which reflects the 3D shape changes due to object motion in a time interval. The approach of 3D flow detection based on 3D optical flow or range flow techniques is applicable if the interframe magnitude of the flow vectors is not too large, e.g., a maximal flow of 10 pixels [5]. Another approach to track surfaces of deforming objects is by registration or parameterization of surfaces. Registration and parameterization are commonly applied to large changes in shape, much larger than what is typically addressed with 3D optical

flow or range flow techniques. Recently, registration and parameterization techniques for deformable surfaces have enjoyed great success and often these techniques involve either an as-rigid-as possible constraint or an isometry constraint on the deformation surface.

For an accurate detection of an object surface motion, a high quality reconstruction model of the real object needs to be generated first. Therefore, we are also interested in reconstruction methods which can generate a 3D model of a real world object realistically and efficiently. Computer-based 3D reconstruction of the real world object has become one of the core elements in many areas, such as the manufacturing industry, the film industry, the game industry, computer simulation, visual walkthrough, etc. Artist drawing and 3D scanning are two common methods of modeling 3D objects. In artist drawing, a lot of software is developed to assist the drawer, such as 3Dmax, Maya, AutoCAD, solidworks and etc. However drawing 3D model by artist is still labour-intensive and time-consuming. It has enjoyed great success in visual effects, such as creating a non-existent scene in a movie. But the exact modeling of the real world objecl is still hard to achieve. With 3D scanning, realistic models are achievable. But the weakness of modeling moving objects is the Achilles heel of the 3D scanning method. However, the recently-developed structured light scanner can be applied in real-time applications by projecting patterns onto the scanning surface. As well, in the last decade, the Multi-View Stereo (MVS) modeling method has attracted a great deal of attention. The main idea of the MVS modeling method is to extract 3D object structure from multiple object images. This method has the potential to overcome weaknesses of the traditional modeling methods. The data collecting device can be just a regular camera. Compared to artist drawing, the MVS method uses little or no labour in the whole reconstruction procedure covering from object images to a 3D realistic model. And the real-time 3D modeling of a moving object can be obtained with the MVS method, since the images can be easily captured in real-time.

In MVS modeling, the basic reconstruction idea is based on the triangulation principle. Every 3D point of an object surface can be reconstructed, if its projecting pixels are matched through object images from different views. With the intensity constancy assumption, various block matching strategies can be applied such as sum of absolute difference (SAD), sum of squared difference (SSD), normalized cross correlation (NCC), etc. Based on the classification of Seitz et al. [59], there are four kinds of different MVS reconstruction methods: MVS models computing and fusion, volumetric reconstruction, surface evolution, and surface growing. In the first method, all the other three reconstruction methods can be used to generate MVS pointclouds from different reference

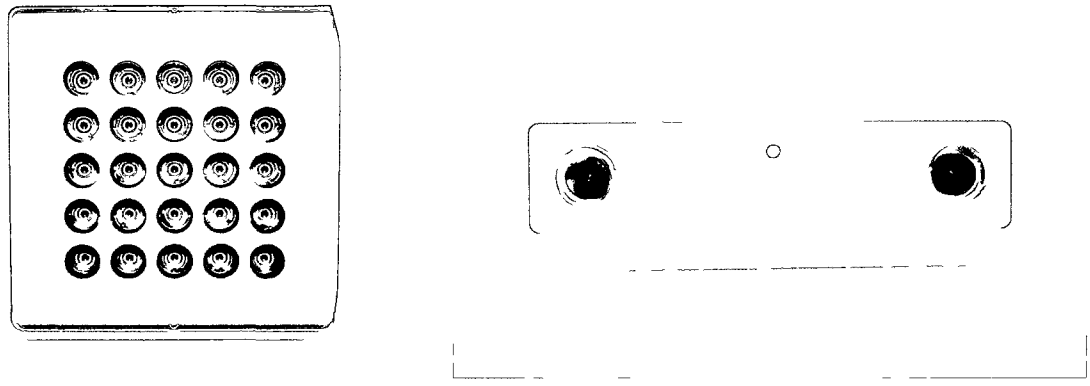


Figure 1.1: Image capture devices: ProFUSION25(left) and Bumblebee2(right)

views. In this thesis, we will focus on the MVS modeling method based on building and merging multiple reconstruction models.

1.2 Thesis Statement and Problem Definition

In this thesis, we are mainly concerned with two things: 3D model reconstruction from object images and motion detection between object models. In 3D model reconstruction, our goal is to reconstruct a 3D object model from the object images generated from a ProFUSION25 camera, as shown in Figure 1.1. The ProFUSION25 is a 5*5 camera array system. With the ProFUSION25, twenty-five black and white multi-view images are captured simultaneously. Its multiple image capture ability simplifies our image data collection step. But, because of the non-calibrated lenses of the ProFUSION25, the camera calibration needs to be done before the reconstruction procedure. And the distortion and vignetting effect should be removed from the raw object images. And also due to the lack of color information from the ProFUSION25 images (black and white), pixel matching and reconstruction are both more difficult.

With regard to motion detection, our goal is to detect the motion among a sequence of object models. Here, the motion between two object models is represented by a 3D flow on the object's surface. By using our CPU based MVS application, it costs 10 hours (CPU: Intel Q6660, Memory: 4GB) to generate a single 3D model from 25 ProFUSION stereo images. Therefore, the popular Point Grey Research Bumblebee2 stereo-head is used to generate the object models during the deformation of an object. The Bumblebee2

is a commercial stereo vision camera with two high quality and calibrated camera lenses. Using this commercial stereo camera, the reconstruction quality is improved during our motion detection procedure. But, there are still some holes in the reconstruction of the model. These holes are caused by intensity variations or lack of texture which is quite common in the real world. Holes present a good test of the robustness of our motion detection method.

1.3 Thesis Overview

To overcome the quality weakness of our ProFUSION25, camera calibration, distortion removal and vignetting removal are considered in our 3D reconstruction framework. Then, we focus on a two-step 3D reconstruction method. The first step is to generate twenty-four MVS pointclouds of the same object surface from different reference views. For each pointcloud, the same twenty-five multi-view images and different reference views are used. A robust pixel matching method will be applied to assign reasonable surface confidence value into each point in the pointcloud. The second step is to fuse multiple pointclouds into a final pointcloud which meets the consistency of all the input pointclouds. In the first step, we design a new hybrid matching method to calculate the surface confidence. And two new fusion strategies are provided to generate the final pointcloud in the second step. Because our 3D reconstruction mainly focuses on improving the pixel matching method and fusion strategy, the 3D model is refined in a post-processing step.

To detect the motion of a deforming object, we propose a local 3D flow method where window-based matches are found based solely on an isometry constraint anchored on the surface with a few reference matches. The geodesic distance is approximately calculated as the shortest path in a pixel graph which is built to embed the topological structure of the object model. Based on the assumption of distance preservation deformation, the 3D flow over the object surface is estimated in two phases. In the first phase, isometry-based matching is applied by using the reference matches generated from intensity-based matching. To overcome the insufficiency of intensity-based reference points, all points matched in the first phase are treated as reference points in the second phase.

1.4 Thesis Contributions

In this thesis, we have developed a framework of 3D reconstruction from the multi-view images generated from the ProFUSION25. And we also propose a robust 3D flow

method based on the isometry constraint. One advantage of our 3D reconstruction framework is that it doesn't restrict to refined stereo camera set. Camera calibration, distortion removal and vignetting removal are all considered in our 3D reconstruction framework. Based on applying the multiple pointclouds fusion step, the improvement of reconstruction quality is shown. And the evaluation of different matching methods and fusion strategies is given. In our isometry-based motion detection, the intensity images are no longer used after matching the feature points. This gives robustness to the intensity variance during the motion. And we show that our motion detection method can tolerate reconstruction noise and successfully find a dense set of motion vectors over the surface of an object. And the work of motion detection is published in the IEEE Conference on Computer and Robot Vision (CRV 2010) [77].

In summary, the key contributions arising from this thesis are:

- a robust hybrid pixel matching method which combines the intensity difference measurement (SAD) and intensity distribution measurement (NCC).
- two new fusion strategies. One applies both high and low confidence points in the fusion step. The other one is based a confidence value weighted dense measurement.
- a novel window-based matching technique for 3D flow based on isometric surface deformation.
- an evaluation of the degradation of the isometry constraint for surfaces which deform non-isometrically and in the presence of topological noise.

1.5 Thesis Organization

The rest of the thesis is organized as follows: In **Chapter 2**, we review related work and present background on 3D reconstruction and 3D flow detection. **Chapter 3** goes through the details of our 3D reconstruction framework including: calibration, raw image post-processing, hybrid matching method and two new reconstruction algorithms. Our 3D flow detection framework will be mentioned in **Chapter 4** including reference point generation, geodesic distance calculation and verification, a window-based matching technique and results for two sequences: an isometric and a non-isometric deformation. Finally, **Chapter 5** concludes this thesis and indicates the direction of the further work.

Chapter 2

Related Work

2.1 Multi-View Stereo

Multi-view stereo (MVS) focuses on the problem of extracting three-dimensional object structure from images captured from different viewpoints of a scene. Recently, MVS has become a very active area in computer vision benefiting from the increasing need for realistically models of the real world. In this section, the pinhole camera model, MVS principles and related MVS algorithms will be reviewed.

2.1.1 Pinhole Camera and Camera Projection Matrix

The pinhole camera is the most common camera model in computer graphics and computer vision. As shown in Figure 2.1, O is the camera optical center. 3D point $Q(x, y, z)^t$ has its projection pixel point $q(u, v)$ in the image plane. Based on camera projection matrix P, we have the Equation 2.1 which transforms the real world 3D point positions into 2D pixel positions in the image domain.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim P \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.1}$$

Here, the $(u, v, 1)^t$ and $(x, y, z, 1)^t$ are the positions of a 2D image point and a 3D point in the homogeneous coordinate. The symbol $\sim$ means the equation is equal up to a non-zero scaling number.

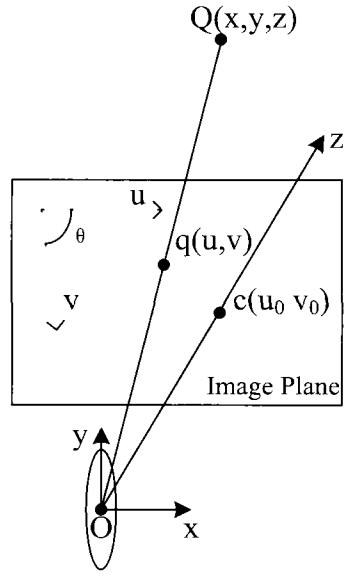


Figure 2 1 Pinhole camera and projection matrix

As shown in Equation 2 2, the projection matrix P can be expressed as the product of three matrices

$$P = CP_0[R|T] \tag{2 2}$$

The Matrix $[R|T]$, as shown in Equation 2 3, is the 4×4 spatial transformation matrix which maps world coordinates into camera coordinates

$$[R|T] = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{2 3}$$

As shown in Equation 2 4, P_0 is the standard projection matrix

$$P_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2 4}$$

C, as shown in Equation 2 5, is the camera calibration matrix In the calibration matrix C, a_u and a_v are the focal lengths in the image row and column directions respectively measured in units of pixels The angle between the image row and column is represented by θ which is typically very close to 90° for digital imaging The principle point (u_0, v_0) in the image domain is the intersection between the image plane and the

camera z-axis.

$$C = \begin{bmatrix} a_u & -a_u \cot\theta & u_0 \\ 0 & a_v \sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

2.1.2 Principle of Multi-View Stereo

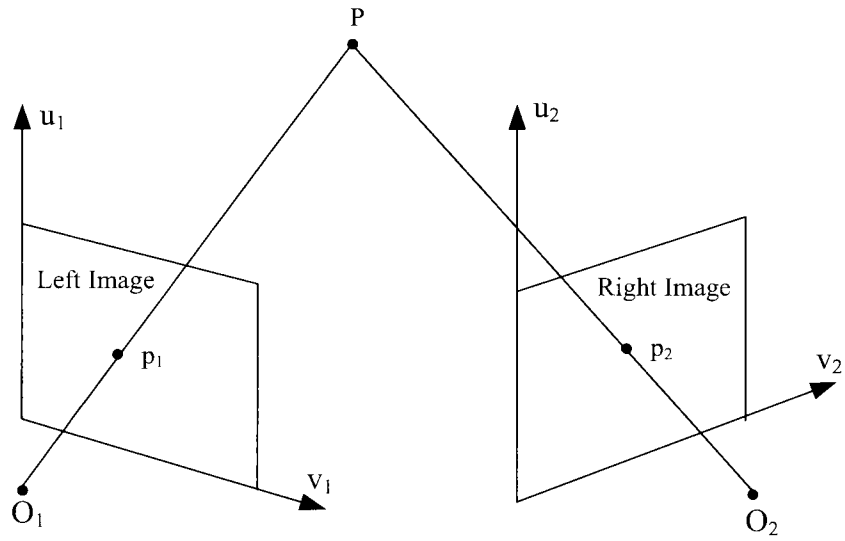


Figure 2.2: Triangulation

The principle which underlies multi-view stereo is called triangulation [53], as shown in Figure 2.2. Left and right images are captured from different view points O_1 and O_2 of the same scene. If pixel point p_1 and p_2 are corresponding to the same object point P, the point P must lie at the intersection of the ray $\overrightarrow{O_1p_1}$ and the ray $\overrightarrow{O_2p_2}$. Using camera positions, directions, and projection matrices, the position of point P can be calculated based on its two projeciton pixels p_1 and p_2. The core task of triangulation is to find the intersection point which represents the object point P.

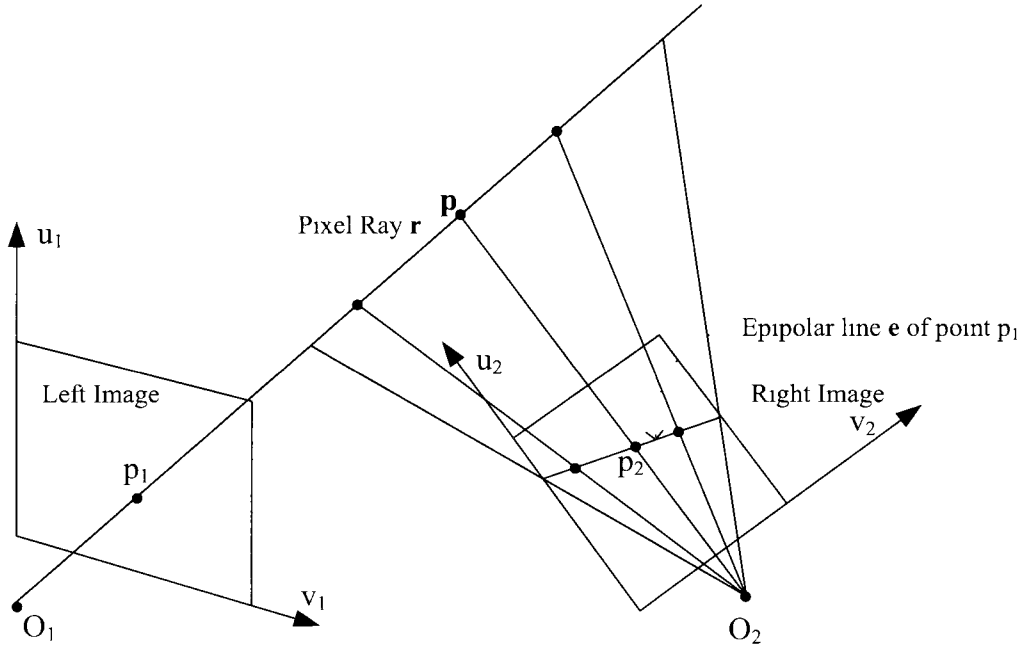


Figure 2 3 Epipolar line and pixel ray

Before calculating the 3D position of the object point P, the corresponding pixels which represent the point P in different images need to be determined Fortunately, the simple and powerful epipolar constraint can be used in the corresponding determination As shown in Figure 2 3, given a pixel p_1 in the left image, its corresponding pixel p_2 in the right image must lie on a line That line is called the epipolar line And the epipolar line **e** can be calculated by projecting the pixel ray **r** $\overrightarrow{O_1 p_1}$ into the right image Therefore, the object point P can be searched for along the pixel ray **r** and verified by its projection pixel in the right image

2.1.3 Multi-view Stereo Algorithms

As mentioned before, multi-view stereo (MVS) is a very active area in computer vision A lot of MVS algorithms and techniques have been developed in recent years In this section, we will review the multi-view reconstruction algorithms following the taxonomy introduced by Seitz et al [59] They classify the multi-view reconstruction algorithms into four classes volumetric reconstruction [62, 19, 60, 17, 36, 15, 76, 3, 24], MVS models computing and fusing [66, 26, 16, 50, 8], surface evolution [39, 41, 55, 10, 49] and surface growing [52, 48, 28, 35]

2.1.3.1 Volumetric Reconstruction

The volumetric reconstruction method is based on the idea of extracting a scene surface from a volume containing an object in 3D space. The volume is usually represented by voxels in a regular 3D grid. The task of generating a 3D model is to label each voxel as a surface voxel or not. A cost function is used to fill voxel with surface confidence value for surface extraction. There are two nice surveys about the volumetric reconstruction method proposed by Slabaugh et al. [62] and Dyer [19].

A voxel coloring algorithm is presented by Seitz and Dyer [60]. In their approach, a voxel is marked as surface voxel, if it has invariant color through multiple images. Snow et al. [17] mark the surface voxel based on minimizing an energy function which combines the observation and smoothness constraints. Kolmogorov and Zabih [36] compute the object shape based on minimizing an energy function based on three properties: photo-consistency, visibility and smoothness. In our MVS pointcloud generation step, a object volume is generated for each reference view. For each volume, every voxel v is represented by a candidate point and filled by a surface confidence value calculated from a combined matching score of its projecting pixels.

Using a sweeping plane is a common technique in volumetric algorithms to assist volume generation and mapping voxels into different image planes. Initially, plane sweeping was introduced by Collins [15] as a way to determine feature points correspondences across multiple images which are captured from different views of the same scene. Using the plane sweeping technique, image rectification is not required when performing pixel matching in multiple images. A GPU implementation of a plane sweeping stereo algorithm is proposed by Yang and Pollefeys [76]. Based on their work, three dimensional reconstruction can be performed in a real-time application. In the approach of Gallup et al. [24], the surface normal direction is estimated to guide the sweeping direction. Baillard and Zisserman [3] proposed a novel piecewise planar multi-view stereo algorithm to reconstruct urban roofs using aerial images. In their approach, 3D edges of a roof are generated based on matched 2D image lines from multiple images. Using a 3D edge as an axis, a set of hypothetic half planes is created by rotating around the axis. The valid half plane is verified based on the similarity among its projection regions in multiple images. Then, new 3D lines are created by half plane intersections. All 3D lines belonging to the same half plane will be grouped and used to cut the half plane into a planar facet. Finally, a piecewise planar building model is represented by facets. In our 3D reconstruction method, the plane sweeping technique is used to generate a

set of candidate points which sample the 3D space containing the object surface. For each MVS pointcloud generation, the sweeping plane is swept in the z-axis direction of the reference camera coordinate system. And the candidate points are the intersections between pixel rays and the sweeping plane.

2.1.3.2 MVS models Computing and Fusing

The main idea of computing and fusing MVS method is to reconstruct a 3D model using multiple depth maps or pointclouds which represent the same object surface from different views. Our MVS method is also applying the fusion step to enhance the reconstruction consistency. There are two steps in this approach. First, multiple depth maps or pointclouds are generated from different reference views of the same object. Second, the consistency of these depth maps or pointclouds is sought to generate a single refined object model.

Szeliski [66] proposed the first multi-view reconstruction framework from multiple depth maps. In his work, the final depth map recovery from multiple depth maps is formulated as a global optimization. Instead of global optimization, our MVS method is based on local optimization.

Goesele et al. [26] introduced a simple MVS algorithm which is divided into two steps: first, depth maps are generated from different reference views by using the window-based binocular stereo matching. Each reconstructed point is assigned a confidence value based on the NCC matching score of corresponding windows. Their final mesh is the result of fusing multiple depth maps using the volumetric reconstruction approach proposed by Curless and Levoy [16].

Bradley et al. [8] also proposed a MVS algorithm based on binocular stereo matching. Different from the method by Goesele et al. [26], a scaled matching window is combined in their block matching method. For the raw single view depth map, the visual hull and disparity ordering constraint [4] are used to verify the valid point. A median-rejection filter is used to remove outliers and a trilateral filter [71] is used to smooth the disparity image. In the fusion step, multiple pointclouds are merged into one dense pointcloud. Then, a hierarchical vertex clustering method is used for down-sampling of the dense pointcloud. An adaptive point-based filter is introduced to remove the noisy points. At the end, a lower dimensional triangulation method is used to generate the mesh.

Merrell et al. [50] proposed two approaches to fuse multiple MVS pointclouds. Each MVS pointcloud is generated by choosing the point with the highest confidence value on each pixel ray from its reference view. The confidence value is calculated from the SAD

matching score of corresponding windows. In the pointcloud fusion step, all points will be grouped if they belong to the same pixel ray from the fusion view. And all points also will be grouped by different reference views which are applied to generate multiple MVS models in the first step. The goal of their fusion step is to select the best point on each fusion view group from the fusion view. Their visibility-based fusion strategy is based on minimizing violations of a visibility constraints. An occlusion value and a violation value are calculated for each point in the fusion step. For a point p, the occlusion value is calculated as the total number of points in p's fusion view group with smaller depth value than p. p's violation value is calculated as the total number of points which have larger depth value than p in reference view groups containing the point p. For each point, its stability value is defined as its occlusion value minus its violation value. In each fusion view group, the surface point is selected as the point with the smallest depth and non-negative stability value. The other fusion strategy of them treats the point with the highest confidence value of each fusion view group as a surface candidate point. If the verification score of the candidate point is positive, it is marked as a real surface point. The verification score is initialized as zero. It is increased if there are points in a small neighbor area of the candidate point. And verification score is decreased if there are points which occlude or are violated by the candidate point. They found the second fusion strategy is more computational efficient than the visibility-based fusion strategy. But the first fusion strategy generates more stable results. The visibility-based fusion strategy is applied as a reference in our evaluation of different fusion strategies. In their fusion strategies, only the high confidence points are used to confirm the final surface point. The conflict between final surface point and low confidence point is not considered. In our high-and-low confidence fusion strategy, both high and low confidence points are used in the search of the object's surface.

2.1.3.3 Surface Growing

The surface growing method also has two steps. First, a set of feature points are extracted and matched into a set of 3D points which represent partial information of a 3D surface. Then, a complete surface is reconstructed and refined from the patial surface information calculated in the first step.

Morris and Kanade [52] address the surface growing problem by finding the best surface triangulation from a set of 3D points which lie on the surface of an object. The best surface triangulation is decided by minimizing the difference between images and reprojecting the surface triangulation back into image views. In the approach of Manessis

et al. [48], a surface triangulation is refined by recursively adding new feature points and using visibility constraints from additional images. The triangulation is generated in the 2D image plane and then mapped into 3D.

Instead of reconstructing 3D structure based on tessellating recovered 3D feature points, Taylor [67] proposed a reconstruction method to calculate the surface by considering the freespace volume returned from a set of reconstructed feature points. A Delaunay triangulation is created from recovered feature points in the image. The disparity of unrecovered areas is estimated by linear interpolation of the disparity of recovered feature points in the triangulation. Scene occupancy is detected from disparity measurement.

Furukawa and Ponce [22] reconstruct a 3D model based on local photometric consistency and global visibility constraints. Their approach also starts from a set of reconstructed 3D feature points. But, instead of generating a triangulation, 3D feature points are expanded into a set of rectangular patches based on photometric consistency. To enforce the visibility consistency, two step filtering is applied to remove the outline from the reconstructed patches. Finally, a polygonal surface is reconstructed by removing vertices which violate smoothness, photometric consistency or rim consistency.

Habbecke and Kobbelt [28] represent a surface based on oriented disks [35]. The surface is reconstructed by two alternating steps. First, the seed disk is computed and corrected by a plane fitting algorithm. Then, the surface is grown by adding new disks or increasing the original disk radius until all visible area is covered by disks.

2.1.3.4 Surface Evolution

The idea of surface evolution method is to estimate the object surface by minimizing a surface error function. Matusik et al. [49] represented an approach of computing and rendering a 3D object based on the visual hull [41] constraint. A visual hull is the intersection volume of multiple perspective projection volumes based on object silhouettes from difference view images. Kutulakos and Seitz [39] proposed a space carving algorithm to remove voxels from the object's volume based on photo consistency. The photo consistency provides background constraints [41] and radiance constraints to help deciding which voxel should be removed.

Pons et al. [55] proposed an image reprojection-based approach. It refines the hypothetic surface based on minimizing the difference between the input multi-view images and their predicted images projected from the hypothetic surface. Gargallo et al. [25] also reconstruct 3D models based on minimizing the error between predicted and observed pixels. But they give an energy function which represents the derivative of reprojection

error over the image plane. In their error function, the visibility constraint is considered. The energy function is minimized by gradient descent surface evolution. Instead of minimizing an error function via gradient descent surface evolution, Tran and Davis [72] minimize the error function with a graph cut algorithm using visual hull constraints and color consistency constraints. Appleton and Talbot [2] use a maximal flow algorithm to achieve a global minimization of the error function.

2.2 Motion Detection

In this thesis, our motion detection method mainly focuses on detecting the motion of a deformable object. Our goal is to obtain the 3D motion flow of each pointcloud which represents a object deformation over time. In this section, the principle and related algorithms of optical flow and 3D flow will be reviewed. After that, we will discuss the idea of a metric constraint and its applications.

2.2.1 Optical Flow

Optical flow is a 2D velocity field which reflects the image changes due to object motion in a time interval. Each velocity vector is represented by the motion of a pixel between two images. Optical flow techniques are widely used in the computer vision area, such as image matching for robot vision, motion analysis, and pattern recognition. Optical flow can be classified into global methods such as the classical approach by Horn-Schunck [32] and local methods such as the classical approach by Lucas-Kanade [46]. Most other optical methods are based on these two basic approaches [6, 13, 65].

2.2.1.1 Global Method

The original global optical flow method is proposed by Horn and Schunck [32]. In their approach, two constraints are considered: intensity constancy and spatial smoothness. The intensity constancy is based on the assumption that the surface of an object will have almost the same intensity during a small motion of an object over time. The spatial smoothness is based on the assumption that the nearby points on the object's surface will have similar image motion vectors during the object motion. A sequence of an object images can be represented as an intensity function $I(x, y, t)$. The variables, x and y, are the pixel positions. The variable t is the time index, which points to individual images.

For any given x, y and t, the function result is the intensity of a pixel at the position (x, y) in an image captured at the time t. Based on the intensity constancy, we have:

$$I(x + dx, y + dy, t + dt) = I(x, y, t) \tag{2.6}$$

Based on intensity constancy, we also can get:

$$\frac{dI}{dt} = 0 \tag{2.7}$$

Based on the chain rule of differentiation and the Equation 2.7, we have:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{2.8}$$

If we let

$$u = \frac{dx}{dt} \quad and \quad v = \frac{dy}{dt} \tag{2.9}$$

Then we can get a single linear equation with two unknown variables u and v,

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \tag{2.10}$$

One way to represent the spatial smoothness constraint is to minimize the square of the gradient magnitude of the motion vector (u, v) over the whole velocity field:

$$(\frac{\partial u}{\partial x})^2 + (\frac{\partial v}{\partial x})^2 \quad and \quad (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial y})^2 \tag{2.11}$$

Then combining these two constraints, we need to minimize the sum of the errors in the brightness equation

$$\epsilon_b = \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} \tag{2.12}$$

and minimize the square of the gradient magnitude of the motion vector (u, v) over the whole velocity field.

$$\epsilon_c^2 = (\frac{\partial u}{\partial x})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial y})^2 \tag{2.13}$$

Finally, the motion vector (u, v) can be solved by minimizing the total error ϵ over the whole velocity field

$$\epsilon = \int\int (a^2\epsilon_c^2 + \epsilon_b^2)dxdy \tag{2.14}$$

2.2.1.2 Local Method

The original local optical flow method was proposed by Lucas and Kanade [46]. Compared to Horn and Schunck's method, they use the spatial constancy constraint instead of the spatial smoothness constraint. And the optical flow is assumed to be a locally constant flow. In a small window which has $n \times n$ pixels, all pixels have the same motion vector (u, v). Then, we can get a set of equations for each window

$$\begin{aligned} &\frac{\partial I}{\partial x_1}u + \frac{\partial I}{\partial y_1}v + \frac{\partial I}{\partial t} = 0 \\ &\frac{\partial I}{\partial x_2}u + \frac{\partial I}{\partial y_2}v + \frac{\partial I}{\partial t} = 0 \\ &\cdots \\ &\frac{\partial I}{\partial x_{n^2}}u + \frac{\partial I}{\partial y_{n^2}}v + \frac{\partial I}{\partial t} = 0 \end{aligned} \tag{2.15}$$

Now, there are two unknowns with n^2 equations. Therefore, the motion vector (u, v) can be solved by applying a least squares method, if $n > 1$.

2.2.2 3D Flow

3D flow is a 3D velocity field which reflects the 3D scene changes due to object motion in a time interval. The 3D flow can be seen as an extension of optical flow. In the principle, all optical flow estimation methods can be extended to estimate the 3D flow in a scene. Combining with the 3D surface structure, 3D flow can be used to analyze the motion of a deformable surface. The 3D flow estimation can be divided by different motion environments: rigid environment [31, 47, 29, 43] and non-rigid environment [54, 74, 75, 64, 63, 57]. In our motion detection application, the main purpose is to detect the motion of a deformable surface. To serve the goal, the following 3D flow review will focus on the 3D flow estimation of non-rigid motion.

Nebel and Sibiryakov [54] calculate 3D flow by combining the stereo and temporal matching in the motion detection. The object surface is reconstructed by using pixel matching on stereo pairs of images. The temporal matching is used to generate the optical flow between successive images. For a pixel associated with a reconstruction surface point in the stereo matching process, its optical flow is converted into 3d flow by mapping the optical flow into two successive reconstruction surfaces.

Vedula et al. [74, 75] detect 3D flow from a deformable surface in a scene. They propose two 3D flow estimation algorithms based on pre-optical flow calculation on the image domain. Both of their algorithms assume that the scene structure is known. One is based on optical flow generated from a single camera. For single camera 3D flow estimation, noisy result can be easily caused by depth noise and discontinuities. To overcome the weakness of single camera, they also introduce a 3D flow estimation algorithm by applying multiple cameras. For each 3D point P, the multiple projections of P's 3D motion vector are calculated by using its pre-estimated optical flows in multiple camera views. The final 3D flow is calculated by minimizing the sum of least squares of the difference between its reprojecting optical flows and the pre-estimated optical flows in different views.

Spies et al. [64] proposed a 3D flow estimation algorithm as the extension of optical flow onto range sequences. The range flow motion constraint is based on a total derivative of a depth function $Z = Z(X, Y, t)$ with respect to time, as shown in Equation 2.16.

$$Z_X U + Z_X V + W + Z_t = 0 \tag{2.16}$$

Here, $f = (U, V, W)$ is the 3D motion flow. Z_X, Z_Y are spatial derivatives of the depth coordinate Z. And Z_t is the temporal derivative of the depth coordinate Z. Based on the range flow motiom constraint equation, three possible flows can be calculated on each 3D point respect to different neighborhood geometric structures. There are plane flow, line flow and full flow. The plane flow is also called as raw normal flow f_r which can be directly computed from the Z derivatives as Equation 2.17

$$f_r = \frac{-Z_t(Z_X, Z_Y, 1)}{{Z_X}^2 + {Z_Y}^2 + 1} \tag{2.17}$$

For points on a planar surface, only the plane flow can be computed from the range flow motion constraint. For point on the intersection line of two planes, the line flow can be computed based on range flow motion constraint which means only the motion along the line can not be solved. For the corner point, the full 3D flow can be directly generated by using range flow motion constraint. Barron and Spies [63] also compute the 3D flow by combining the intensity constraint and range constraint both in a regularization framework and a least squares framework.

The 3D flow estimation in varying illumination is studied by Schuchert et al. [57] High pass and homomorphic filters are mentioned to suppressing the intensity changes. Instead of the intensity constancy assumption, brightness changes are allowed in their 3D

flow estimation method. A physics-based model is proposed that applies a combination of gradient and intensity constraints.

2.2.3 Isometry Constraint

As we mentioned above, the intensity constraint and range flow motion constraint assist finding the points correspondences by approximating the total derivative. Therefore, for large deformation detection, the intensity constraint and range flow motion constraint will easily fail. Recently, the isometry constraint is widely used in mesh parameterization, registration and morphing which have in common the goal of finding the correspondence between two surfaces with large deformation.

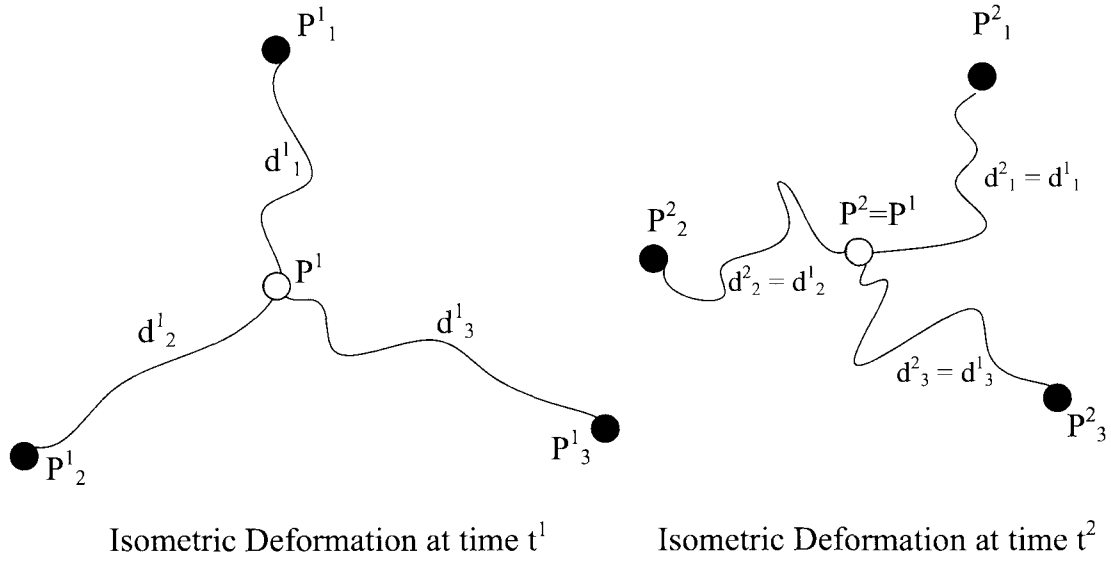


Figure 2.4: Isometry constraint

An isometry mapping between surfaces S^1 and S^2 is an angle preserve mapping, i.e., all angles generated by three points P_1, P_2 and P_3 on S^1 are equal to the angle generated by their mapping on S^2. An isometry mapping is also an area preserving mapping, i.e., the area A^1 on S^1 is a mapping of A^2 on surface S^2 only if A^1 and A^2 have the same size. This implies that the isometry mapping is also a length preserving mapping. And the length preservation can be applied to find correspondences between the isometrically deformed surfaces S^1 and S^2. Given three matched non-collinear reference points on surfaces S^1 and S^2, points on the surface can be tracked between S^1 and S^2. As we see in Figure 2.4, Given a point P^1 and three non-collinear reference points $P^1{}_1$, $P^1{}_2$ and $P^1{}_3$

on S^1 with geodesic distances ${d^1}_1$, ${d^1}_2$ and ${d^1}_3$ from P^1, under the isometry constraint, the corresponding point P^2 should have the same geodesic distances to the corresponding reference points ${P^2}_1$, ${P^2}_2$ and ${P^2}_3$ on S^2. And only the corresponding point P^2 can have the same geodesic distances to the corresponding reference points ${P^2}_1$, ${P^2}_2$ and ${P^2}_3$.

In this thesis, we use reference points to name the matched feature points. Theoretically, based on the isometry constraint, the correspondence between two isometrically deformed surfaces can be explored with more than three non-collinear reference points. In practical, the approximate isometry holds for many surfaces with little stretch during object deformation, such as human skin, most cloth material, etc. and is an attractive constraint for matching.

Ahmed et al. [1] propose to track the deformation of a triangular surface mesh with a sparse set of intensity-based reference features. They define harmonic functions which stay invariant with respect to the Laplace-Beltrami operator over time. These functions are iso-contours on the mesh and they fill-in the vertices not matched with features by intersecting ten contours for a vertex and moving the vertex to the intersection point with minimal error. We also use optical features as reference points but for our small-baseline application, we prefer KLT feature tracking [46, 70] instead of using SIFT features [45] as Ahmed et al. [1] and Pritchard and Heidrich [56]. We propose a direct approximation of geodesics on the surface and do not utilize a mesh but use the range image itself for neighbourhood information.

The method of Ahmed et al. [1] does not handle holes and other topological noise well. Later, Tevs et al. [69] specifically addressed this issue making both the feature correspondence step robust and dealing with location errors of features on the surface. Tevs et al. [69] use an isometry constraint for verification of the reference features in a RANSAC step and during their tangent optimization of the feature localization. The geodesics on the surface are approximated directly on the pointcloud obtained with a 3D scanner. The neighbourhood information on the pointcloud is established through k-nearest neighbours in a topology graph. In order to speed up finding geodesics, the graph is subsampled for point locations with a few thousands points (up to $3,196$) but distances are still calculated on the original dense graph. In addition to matching the reference points, additional secondary features are inserted based on an isometry constraint again. They report handling up to $7,740$ candidate correspondences. The problems of holes and noise are also handled in our motion detection framework. Instead of dealing with the matching in 3D, our motion deteciton method is working on a 2D pixel graph which embeds the topological structure of the sample points on the object surface. The

neighbourhood information is directly established through window-based neighbours in the reference images. And a dense 3D flow is achieved by applying additional matching based on isometry-based reference points which are derived from intensity-based reference points first.

Bradley et al. [9] register scans of garment in motion obtained from a multi-view stereo setup. They use the isometry constraint in their surface parameterization framework. For each garment surface, four off-surface anchor points of the cloth are selected as reference points, basically, the openings of the garment. By applying the isometry assumption, a uniform base mesh is generated from these four reference points. And a stretch-minimizing cross-parameterization [37, 38] is applied to map each surface mesh onto the base mesh. At the end, the garment surfaces are consistently parameterized based on combining mappings between the base mesh and surface meshes. The solution by Bradley et al. [9] is specialized to garment capture with multi-view stereo data. Instead of using meshes to generate a mapping between triangles, our motion detection method can be directly used on pointclouds without meshing.

Chapter 3

Stereo-Based 3D Model Acquisition

3.1 Overview

In this chapter, a 3D model reconstruction framework will be described. A multi-view camera system (ProFUSION25) is applied to generate object images from different views. As shown in Figure 3.1, the whole framework includes the aquisition setup, MVS pointcloud generation and fusion of multiple MVS pointclouds. In Section 3.2, camera calibration, distortion removal, vignetting removal and background detection will be discussed. A new matching method and two new fusion strategies will be introduced in Section 3.3. Finally, the evaluation and 3D reconstruction results of different matching methods and fusion strategies are given in Section 3.4.

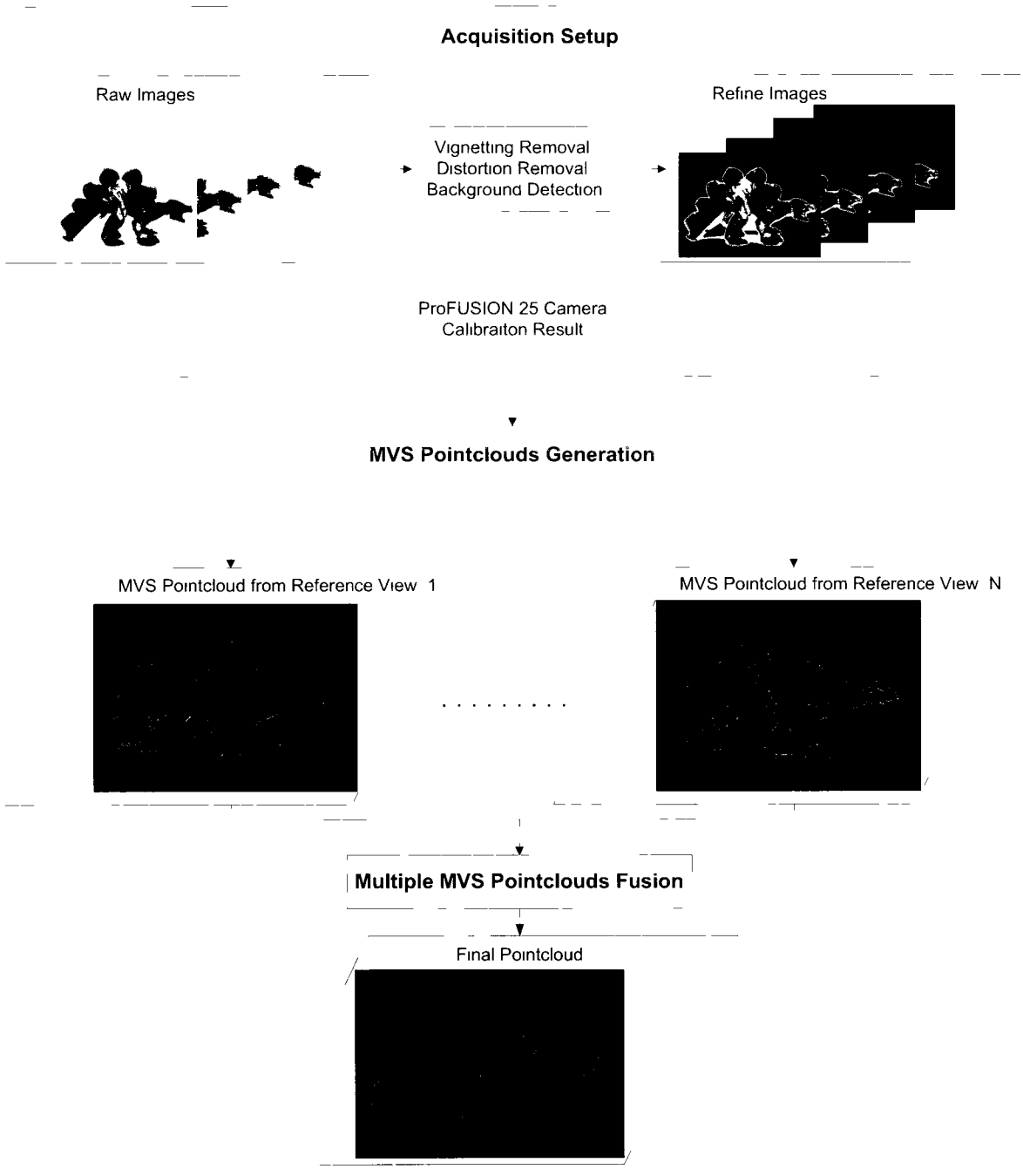


Figure 3.1: Overview of our 3D reconstruction framework

3.2 Acquisition Setup

3.2.1 Geometric Camera Calibration

Camera calibration is commonly treated as the first step in most multi-view 3D reconstruction approaches. The accuracy of the camera calibration directly determines the quality of the 3D reconstruction result. The goal of camera calibration is to estimate a geometric relationship between a scene point and an image point. This geometric relationship is formed by using internal and external camera parameters. As we mentioned in Section 2.1.1, the main internal camera parameters include focal length, optical center and the angle between image axes. Also, the camera distortion parameters are considered as a part of camera internal parameters. More detail of the camera distortion will be discussed in Section 3.2.3. Based on the internal camera parameters, the camera calibration matrix can be used to project 3D points into the image plane. The camera external parameters form the transformation matrix between the coordinate systems of a camera and the reference camera in the multi-view stereo system. Therefore, if the 3D position of a point P is known in the coordinate system of camera i, the projection pixel of P in camera j can be calculated by using internal parameters of camera j and the external camera parameters between camera i and camera j. To calculate the external parameters between any two cameras in the ProFUSION25, we only need to estimate the external parameters between one fixed camera to all other twenty-four cameras in the ProFUSION25. To estimate the external parameter, the central camera 12 of the ProFUSION25 is selected as the fixed camera. In each camera pair, the fixed camera is called the reference camera. The other camera is called the non-reference camera. For any 3D point $P(x, y, z)$ in the reference camera coordinate system, its 3D position $P_i(x_i, y_i, z_i)$ in a non-reference camera i can be calculated by Equation 3.1

$$P_i = R_i P + T_i \tag{3.1}$$

Here, R_i and T_i are defined as the rotation and translation matrices from the reference camera to the non-reference camera i. Therefore, the external parameters between camera n and camera m can be calculated by using their external parameters (R_n, T_n, R_m, T_m) related to the reference camera. The rotation matrix R_{nm} and translation matrix T_{nm} between camera n and camera m are given in Equation 3.2 and Equation 3.3. R'_n is the transpose of R_n and $R'_n = R_n^{-1}$.

$$R_{nm} = R_m R'_n \tag{3.2}$$

$$T_{nm} = T_m - R_m R_n' T_n \tag{3.3}$$

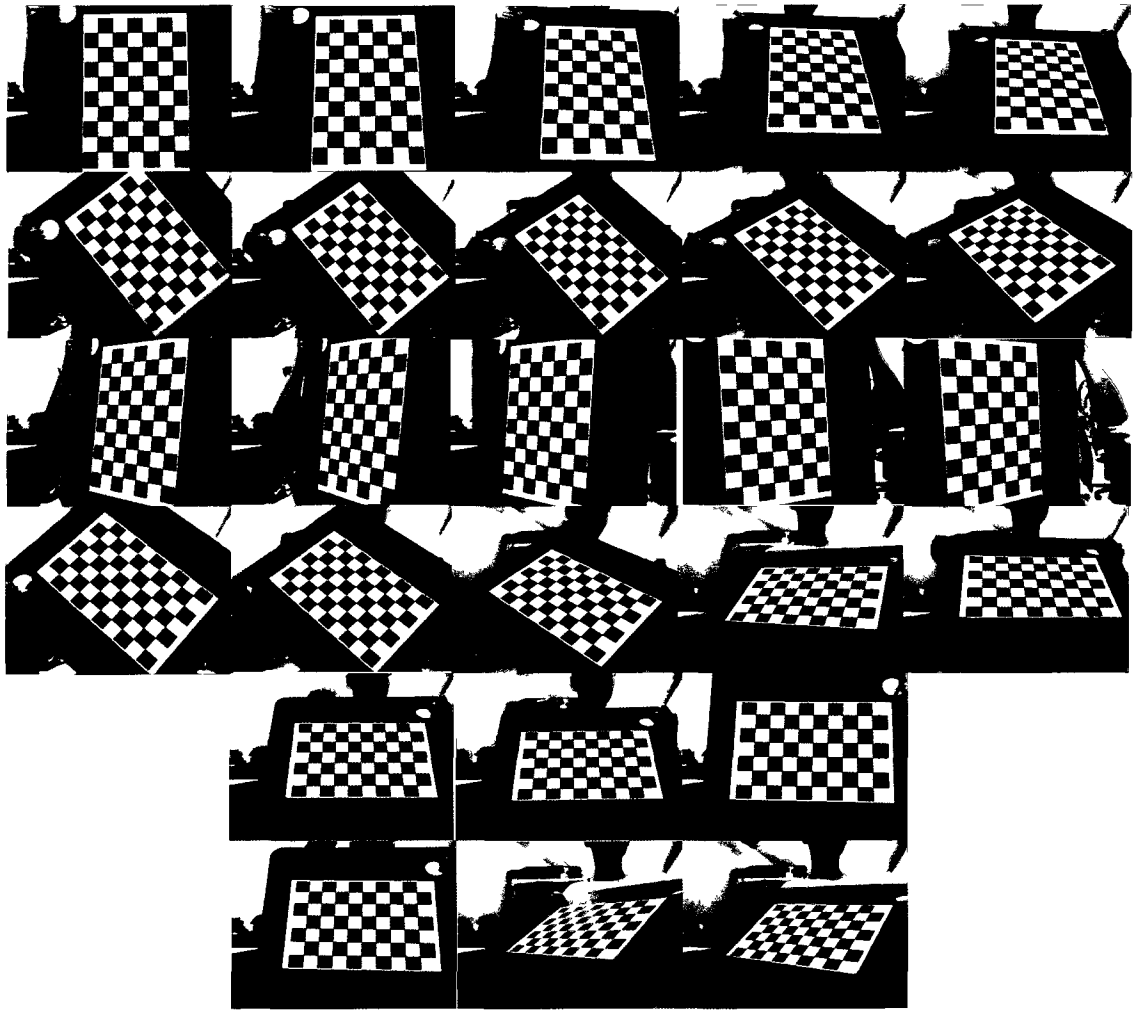


Figure 3.2: Calibration images

In off-line camera calibration methods, the camera parameters are estimated based on a known structure object which usually is a checkerboard pattern. To calibrate the ProFUSION25, a black and white checkerboard pattern is printed and fixed to a flat board. As shown in Figure 3.2, twenty-six calibration images showing different poses of the checkerboard pattern are captured for each camera. The internal camera parameters are computed by minimizing the error between the checkerboard pattern and its reprojection image. More recently, the idea of camera self-calibration has been introduced to estimate camera internal parameters without eculidean scene information. For further

information about camera calibration and self-calibration, we direct the reader to surveys by Clarke and Fryer [14] and Hemayed [30].

In our 3D model reconstruction framework, the Camera Calibration Toolbox [7] is used to calibrate the internal parameters of the ProFUSION25. The accuracy of the camera calibration result depends on experimental conditions such as the quality of calibration images, flatness of the checkerboard pattern picture, the number of calibration images and overall checkerboard pattern poses for each camera. In the Camera Calibration Toolbox, a pixel error is given for each single camera calibration and it measures the pixel difference between the projected scene point and the image point.

The external parameter of the ProFUSION25 can not be directly estimated from the Camera Calibration Toolbox. Because it only provides the function of estimating the external parameter between two cameras and the estimation of external parameters adjusts the initial internal parameters. Therefore, internal parameters of the same camera are varying in the external parameter estimation of different camera pairs. The main idea of optimizing the camera external parameters is to minimize the overall error between the observed and predicted image points. The predicted image points are calculated as the projections of scene points by applying the estimated internal and external parameters. To calculate the initial predicted image points, the internal and external parameters generated from Camera Calibration Toolbox are applied. To minimize parameters related reprojection error, the bundle adjustment (BA) [73] is used to find the optimized external parameters which lead to a reduced reprojection error. If we assume that n 3D points are seen in m views and X_{ij} is the projection of ith point in the image jth, BA minimizes the Equation 3.4

$$\min_{a_j, b_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(Q(a_j, b_i), X_{ij})^2 \tag{3.4}$$

Here, a_j is a vector which denotes the parameters of camera j. b_i is a vector which denotes the parameters of a 3D point i. v_{ij} is equal to 1, if point i is visible in camera j and 0 otherwise. $Q(a_j, b_i)$ is the predicted projection of point i in camera j. $d(X, Y)$ denotes the Euclidean distance between the pixel points represented by vector X and Y. To solve the minimization problem of the nonlinear function, we apply the sparse bundle adjustment (SBA) [44] to optimize the camera external parameters. SBA is a sparse variant of Levenberg-Marquardt (LM) algorithm [42] which is a classical solution of minimizing a nonlinear function. In our external parameter estimation, 35100 projections are provided in the SBA external parameters optimization. The initial internal

and external parameters of SBA are generated from the Camera Calibration Toolbox. For the SBA reprojection optimization, we set the optimal internal parameters as the initial internal parameters and only optimize the external parameters by minimizing the reprojection error. The average image error of 35100 projections is 0.1844 pixel based on our final calibration result of the ProFUSION25.

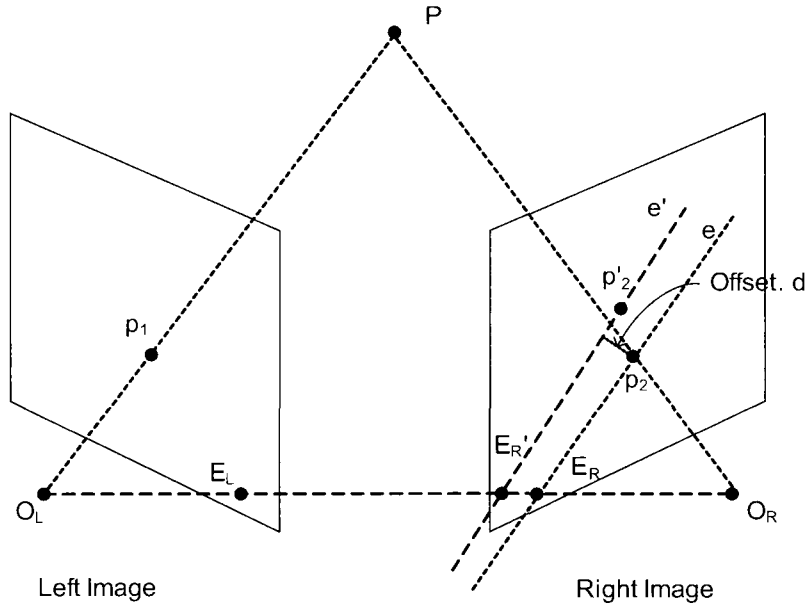


Figure 3.3: Epipolar line offset

To make sure the calibration result is accurate enough for our application, the affect of the calibration error related to the 3D reconstruction result is also measured here. As mentioned in Section 2.1, the position of a 3D surface point P can be calculated by using at least two of its projection pixels in different images. If we want to calculate the position of a 3D surface point P which has a projection pixel p_1 in the left image, we need to find the projection pixel p_2 of P in the right image, as shown in Figure 3.3. Based on the epipolar constraint and the intensity constancy assumption, we search for the matching pixel p_2 on p_1's epipolar line e in the right image. Usually, only the approximate epipolar line e' can be calculated instead of e by using the left and right camera calibration matrices. The difference between e' and e is caused by the error in the camera calibration. Therefore, the calibration error will lead to improper pixel matching. And it will lead to a reduction in quality of the reconstruction result. Here, the calibration result is verified by measuring the distance between the real matched

pixel p_2 and the estimated epipolar line e'. Finally, a good calibration result is chosen by having an overall offset distance less than one pixel. For more detail about calibration error estimation, we refer the reader to Appendix A.

3.2.2 Vignetting Removal

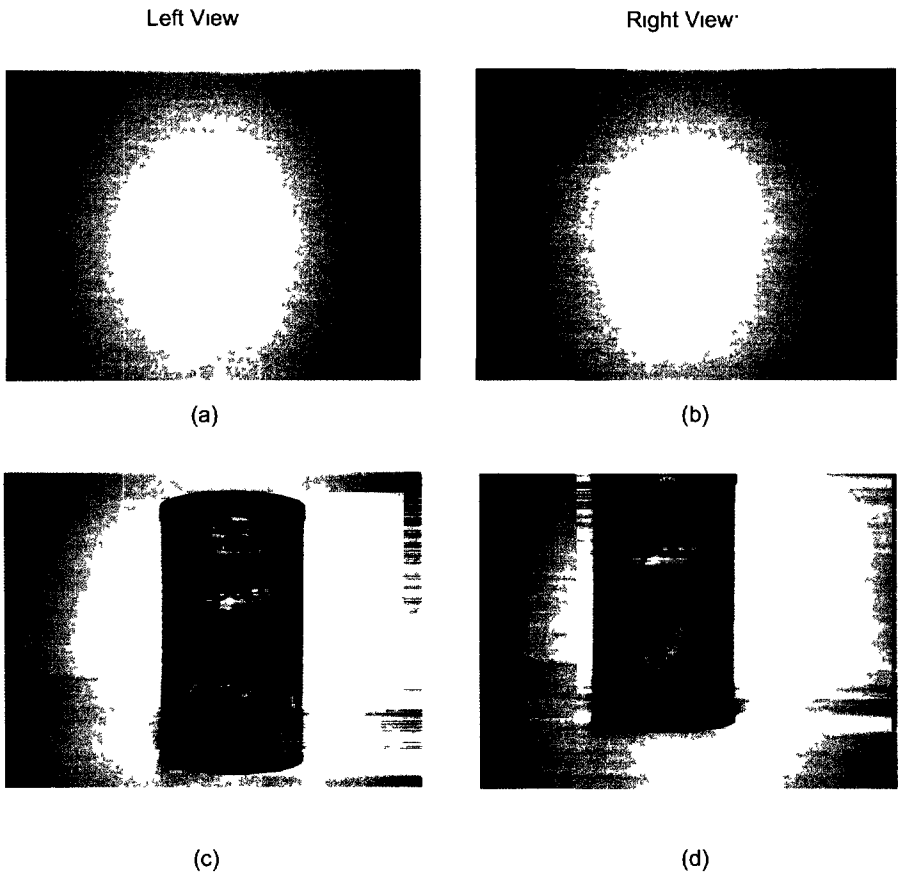


Figure 3.4: Vignetting effect. (a) and (b) are uniform white images with visible vignetting. (c) and (d) are images of an object with visible vignetting.

Vignetting is defined as the radial reduction of brightness or saturation from the image center. The effect can be observed well in images of light objects as gradual darkening towards the image boundary. As shown in Figure 3.4, (a) and (b) are two images for the same uniform white background under an ambient lighting from the left view and the right view. (c) and (d) are two object images from the same scene with the coffee can. The vignetting causes the different parts of the object imaged darker than the

ground truth in different images. Therefore, the effect of vignetting violates the intensity constancy assumption. Most multi-view reconstruction algorithms, as well as our reconstruction method, are based on the intensity constancy assumption and intensity-based pixel matching. This motivates us to remove the vignetting from the image in a preprocessing step.

Goldman and Chen [27] categorize image vignetting due to four main causes: natural vignetting, pixel vignetting, optical vignetting and mechanical vignetting.

1. Natural vignetting is caused by geometric optics that lead different pixels of the image receiving different irradiance. This intensity falloff can be modeled as fourth power of the cosine of the angle between the pixel ray and the optical axis.

2. Pixel vignetting is caused by angle-dependence of the digital sensor that leads the light at a right angle to the digital sensor producing stronger signal than other light at an oblique angle.

3. Optical vignetting is caused by lens diaphragm that blocks light through the lens. The optical vignetting can be modeled as a function of aperture width.

4. Mechanical vignetting is caused by camera element blocking light path such as filter or lens hood.

Fanaswala [20] finds that natural vignetting dominates the vignetting effect in the raw images from the ProFUSION25 camera. He proposed a prototype vignetting function to simplify the fourth power cosine function. The prototype vignetting function of a camera can be calculated based on a background image of a uniform white scene. Figure 3.5(a) and Figure 3.5(d) are the raw images from left and right view. Figure 3.5(c) and Figure 3.5(f) are images after applying the vignetting prototype removal kernels as shown in Figure 3.5(b) and Figure 3.5(e). In this thesis work, we use Fanaswala's implementation for images captured with the ProFUSION25.

To remove the vignetting from an image, we need to scale each pixel in the raw image by its vignetting factor as the inverse of its prototype value. As shown in the Equation 3.5, the Scaling factor $SF(p)$ for pixel p in image position (u, v) can be calculated using the intensity $I(q)$ of the pixel q with the same image position (u,v) from the white background image.

$$SF(p) = 255/(255 - I(q)) \tag{3.5}$$

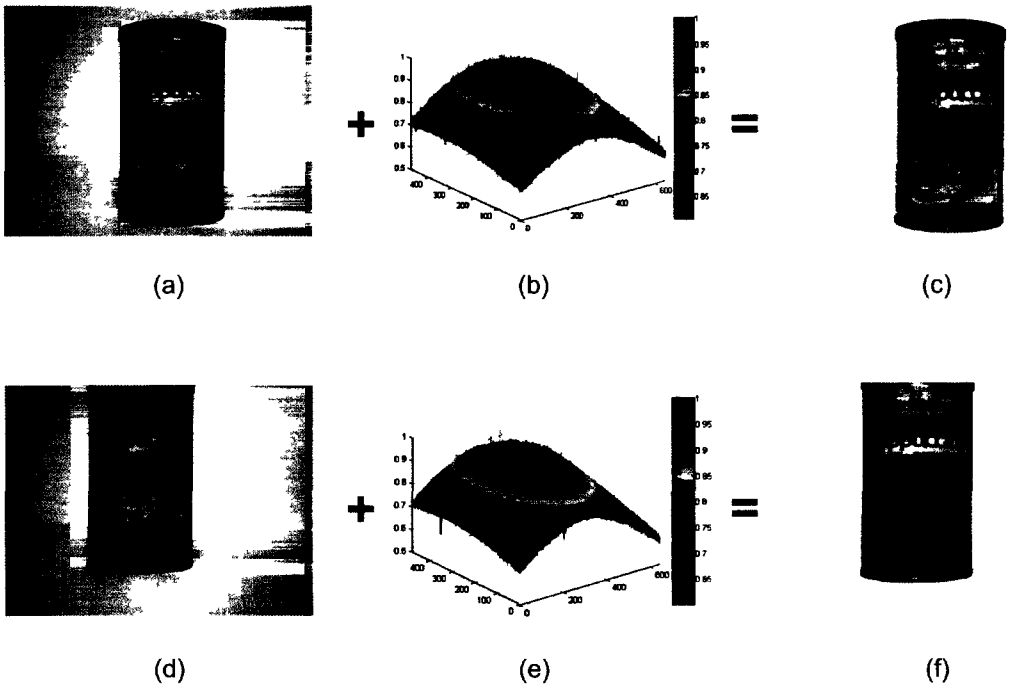


Figure 3.5: Vignetting removal. (a) and (d) are original images from left and right views, respectively. (c) and (f) are relative images without vignetting effect by applying vignetting kernels (d) and (e), respectively.

3.2.3 Distortion Removal

For a raw image with geometric distortion, a straight line in the scene will be projected as a non-straight line in the image. The distortion effect in the image may be caused for example by spherical camera lenses which don't exactly follow the projection principle of a pinhole camera model. In our 3D reconstruction framework, the pinhole camera model is applied to map the 3D point to its projections in different images. Therefore, the distortion effect should be removed from the raw image captured with the ProFUSION25. In Brown's distortion model [11, 21], the image distortion is mainly caused by two components: radial distortion and tangential distortion. The radial distortion is due to misalignment of the spherical lens which decreases or increases the image magnification with the distance from the optical center of the lens. In the image, the radial distortion will cause an outward or inward shift for all the pixels from their initial perspective projection positions. The tangential distortion is caused by imperfect centering of the camera lens. In the calibration toolbox, three radial distortion coefficients (K_1, K_2, K_5)

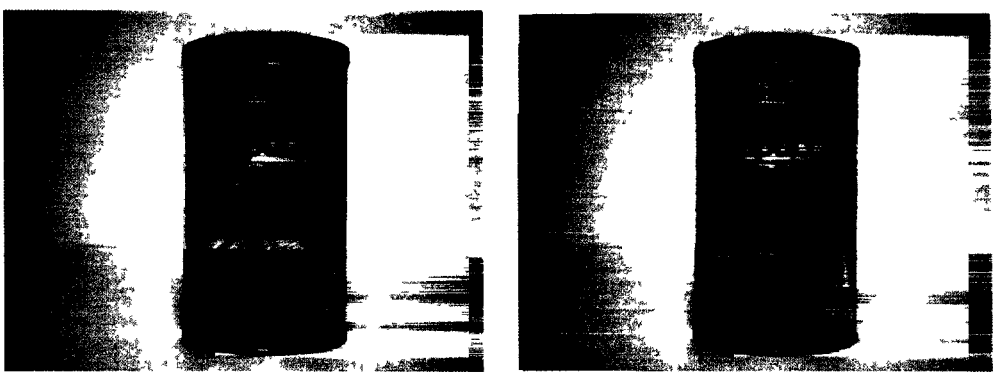
and two tangential distortion coefficients $(K_3, K4)$ for each camera are estimated under Brown's distortion model. Based on these distortion parameter, the normalized distorted image point $P_{distorted}(x_d, y_d)$ can be calculated from a normalized undistorted image point $P_{undistorted}(x_u, y_u)$ by the following equations:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = (1 + K_1 r^2 + K_2 r^4 + K_5 r^6) \begin{bmatrix} x_u \\ y_u \end{bmatrix} + dx \tag{3.6}$$

$$dx = \begin{bmatrix} 2K_3 x_u y_u + K_4(r^2 + 2{x_u}^2) \\ K_3(r^2 + 2{y_u}^2) + 2K_4 x_u y_u \end{bmatrix} \tag{3.7}$$

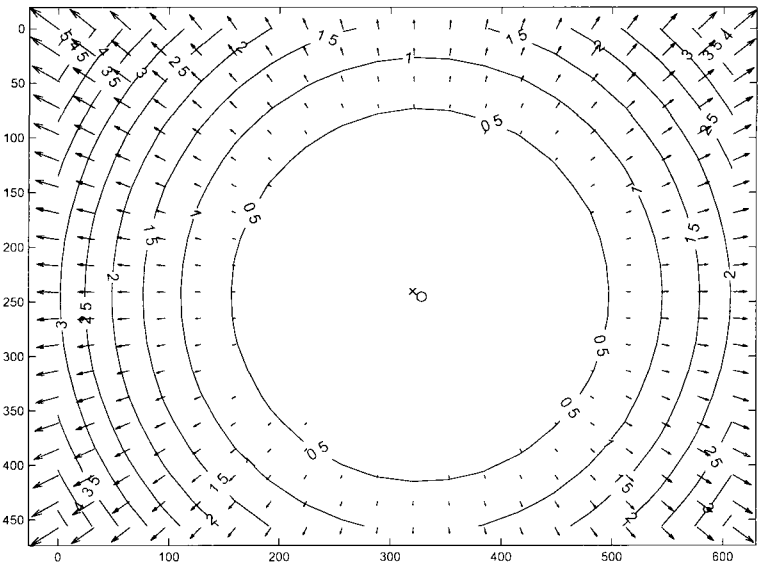
$$r^2 = {x_u}^2 + {y_u}^2 \tag{3.8}$$

Here, we apply the Camera Calibration Toolbox to remove the image distortion based on the estimated distortion components. In Figure 3.6, the distortion effect is removed from the raw image based on the radial and tangential distortion coefficients. And the Figure 3.7 generated from the Camera Calibration Toolbox shows a contour graph of pixel displacements caused by different distortion components. These pixel-based displacements show the differences between raw image pixels and their projections calculated by camera calibration matrix.

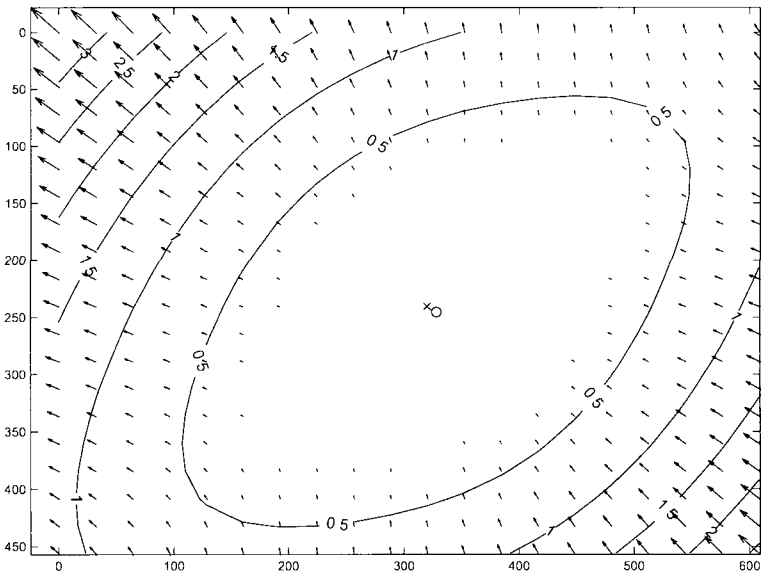


(a) (b)

Figure 3.6: Distortion removal. (a) is a image with distortion. (b) is the image after distortion removal.



(a)



(b)

Figure 3.7: Distortion effect. (a) and (b) show the distortion on each pixel related to radial distortion and tangential distortion on central camera in the ProFUSION25, respectively. Each arrow represents a pixel based shift of an image point caused by image distortion.

3.2.4 Background Removal

To simplify the background detection, the object image is captured under a uniform white background with ambient lighting. A background image is recorded from each camera without the object. To cut the background from the raw object image, a simple background threshold method is applied. For the object image, the background pixel can be identified as having less color difference between object image and its background image than a threshold h. But the correct threshold is hard to choose for single pixel comparison. A higher value of h can remove more background points, but it will also cut more object pixels. To achieve better result, we apply the sum of squared difference (SAD) window matching to compare similarity between two pixels. Based on SAD window matching, similarity of two pixels p and q are calculated as the average color difference d between two pixel squares centered at p and q, respectively. More detail about the window matching method will be covered in Section 3.3.3. In the background removal step, background pixels are detected if their d are less than h. Taking advantage of our simple background environment, the object pixel can be directly detected from the image by setting threshold h as 30.

Figure 3.8: Background removal. (a) is a background image taken from central camera of the ProFUSION25. (b) is the object image taken from the same camera in the background of (a). The background pixel of (b) is marked as black in (c) after the background removal procedure.

3.3 Multi-View Stereo

3.3.1 Overview

In this section, we will introduce the idea of our MVS method based on multi-view images taken from the ProFUSION25. A summary of terms mentioned in our MVS method is given in Section 3.3.2. In Section 3.3.3, a new matching method for the MVS pointcloud generation step is designed to measure the similarity of matching windows both, based on the intensity difference and the intensity distribution difference. In order to generate a better final pointcloud from multiple MVS pointclouds, two new fusion strategies are introduced in Section 3.3.4 and Section 3.3.5.

3.3.2 Terms

Before discussing the details of our MVS method, we give a summary of the terms used in the following explanation of the MVS method. Here, we assign a camera index to each single camera in the ProFUSION25 camera system, as shown in Figure 3.9

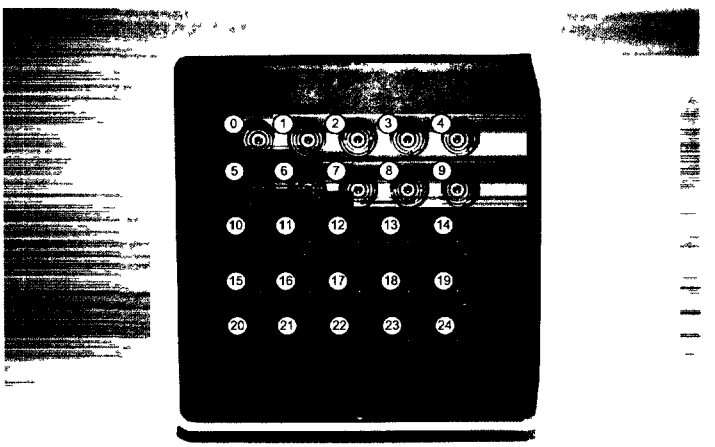


Figure 3.9: ProFUSION camera with index

- A **pixel ray** is related to a view point and a object pixel in the image from that view. It starts from a view point through a image pixel, as shown in Figure 3.10.

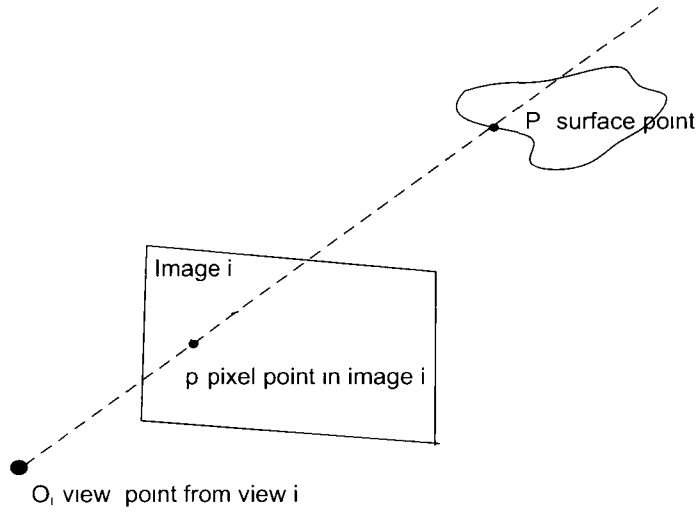


Figure 3.10: Pixel ray

- A **reference view** is a camera view which is used to generate the initial point-cloud. In our MVS algorithm, 24 intial pointclouds are generated based on using different reference views $[0, 1 \cdots 24, 25]$ expect view 12. View 12 will be used as a reference view in the multiple MVS pointclouds fusion step. In each initial point-cloud generation and surface confidence filling procedure, once a reference view is chosen, all other views will be defined as **non-reference views**.

- A **reference image** is the image generated from a reference view in the respective initial pointcloud generation and surface confidence filling procedure. Meanwhile, all the images generated from non-reference views are defined as non-reference images. Each pixel ray is generated by a pixel in a reference image. The pixel in a reference image is called a reference pixel.

- An **initial pointcloud** is the combination of all candidate points on all pixel rays generated from a reference image and a reference view. For each pixel ray, its candidate points are generated as the intersections between the pixel ray and a sweeping plane. The plane is swept in the depth direction of the relative reference view. A initial pointcloud gives a dense sampling of the 3D space which contains the real world object surface in the coordinate system of the reference view. Further, each candidate point P will be assigned a surface confidence value based on the intensity consistency of P's projection pixels in different views.

- A **matching windows** of pixel p is defined as a small neighboring region centered at p. To increase the robustness of the pixel matching result, the similarity between two pixels p and q is measured by the similarity between their matching windows.
- A **matching score** is a similarity metric of two matching windows. Based on different matching methods, the matching score can be different for the same pair of matching windows.

3.3.3 Matching Method

In 3D reconstruction, the real surface point causes a high similarity measure between its projection pixels in multiple images. In practice, a single pixel can not be matched, unless it has very different brightness value than all other pixels in the image. For robust matching results, the similarity of two pixels is generally calculated by comparing the similarity of their matching windows. Common matching methods can often be divided into two classes: matching based on intensity difference measurement and on intensity distribution difference measurement. One instance of the first class is the sum of squared difference (SSD) matching method which measures the direct intensity difference between two matching windows. Another popular intensity difference measuring method is the sum of absolute difference (SAD) which is frequently used for computational efficiency. The normalized cross correlation (NCC) is the classical matching method that measures the intensity distribution difference. It is used in most recent MVS algorithms because it is insensitive to radiometric bias and gain [12]. The definition of SSD, SAD and NCC are given in Equation 3.9, Equation 3.10 and Equation 3.11, respectively. Here, N^2 means the size of the matching window.

$$SSD(v_0, v_1) = \sum_{j=0}^{N^2} (v_0(j) - v_1(j))^2 \tag{3.9}$$

$$SAD(v_0, v_1) = \sum_{j=0}^{N^2} |v_0(j) - v_1(j)| \tag{3.10}$$

$$NCC(v_0, v_1) = \frac{\sum_{j=0}^{N^2} (v_0(j) - \overline{v}_0) \cdot (v_1(j) - \overline{v}_1)}{\sqrt{\sum_{j=0}^{N^2} (v_0(j) - \overline{v}_0)^2 \cdot (v_1(j) - \overline{v}_1)^2}} \tag{3.11}$$

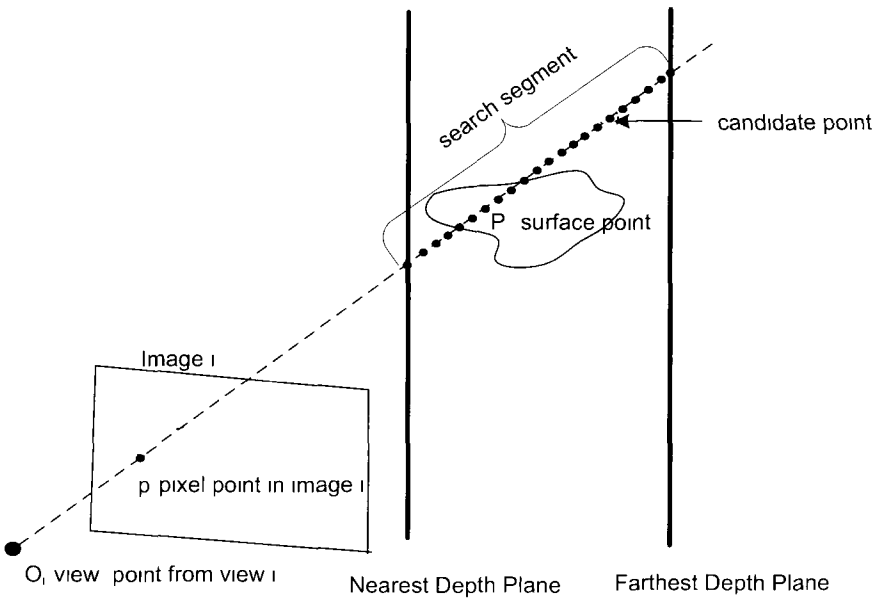


Figure 3.11: Search segment and candidate point

In our application, the search area is narrowed from the whole pixel ray to a search segment formed by the nearest depth plane and the farthest depth plane as shown in Figure 3.11. The reconstruction object is contained in the 3D space between these bounding planes. In our application, the appropriate nearest and farthest planes are determined by manual measurement. Ideally, the interval between two consecutive candidate points should be small enough so that their projection in the image isless or equal to one pixel. A larger interval will lead to matching error in modeling objects with high-frequency texture, while a smaller interval will lead to increased computational cost. To balance the computational cost and matching accuracy, each search segment will be symmetrically sampled by candidate points. Here, we set one millimeter as the interval depth of two consecutive candidate points. To find the surface point on each pixel ray, a surface confidence value is calculated for each candidate point based on the similarity among its projection pixels.

As shown in Figure 3.12, Figure 3.13 and Figure 3.14, the surface confidence of each candidate point is measured by three kinds of matching scores calculated by SSD, SAD and NCC matching methods. A matching score distribution curve is drawn for each matching method on the same search segment. The search segment is sampled by 150 candidate points. And the index of candidate point is marked from 1 to 150 reflecting

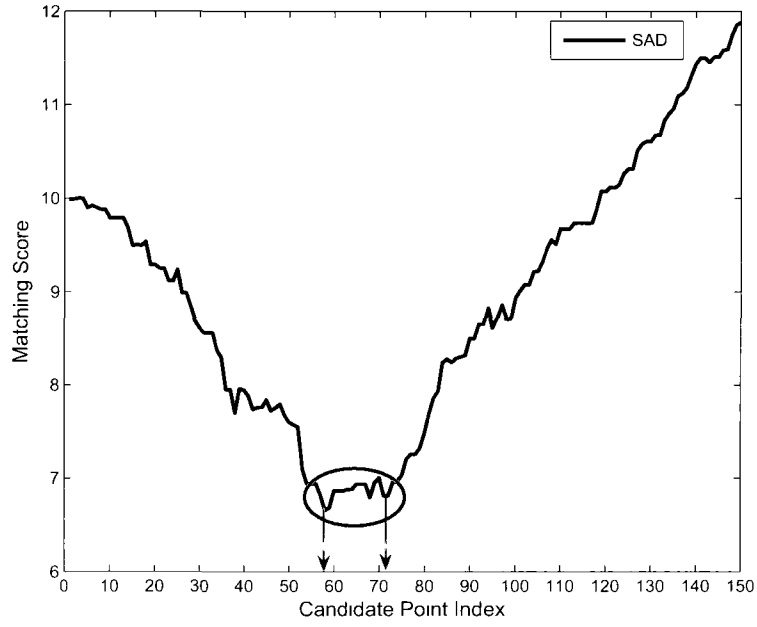


Figure 3.12: SAD measurement

its depth value. The higher index means larger depth value. Here, both SSD and SAD matching scores are divided by the matching window size N^2.

Because our input data are just black and white images, it causes the overall pixel intensity difference range to be much smaller than with color images. The lack of color leads to a flat bottom in the SAD and SSD matching score curves and a flat top in the NCC matching score curve. For the same search segment, the candidate point with the highest surface confidence value can be different depending on different similarity measures. Both, intensity difference and intensity distribution difference measurements have their own weaknesses. The intensity difference is sensitive to radiometric gain and bias [12] as we mentioned before. The intensity distribution difference measurement will fail, if the measuring windows have symmetrical color distribution, e.g., one window is all white and the other one is all black. For our small color difference range, these side effects can be more common than for full color images. Therefore, the surface point can not be found reliably by simply applying a single matching method.

Here, we use an ad-hoc matching method to enhance the surface confidence of the point with highest similarity score. The method combines the intensity difference measurement and intensity distribution difference measurement. We observe that the in-

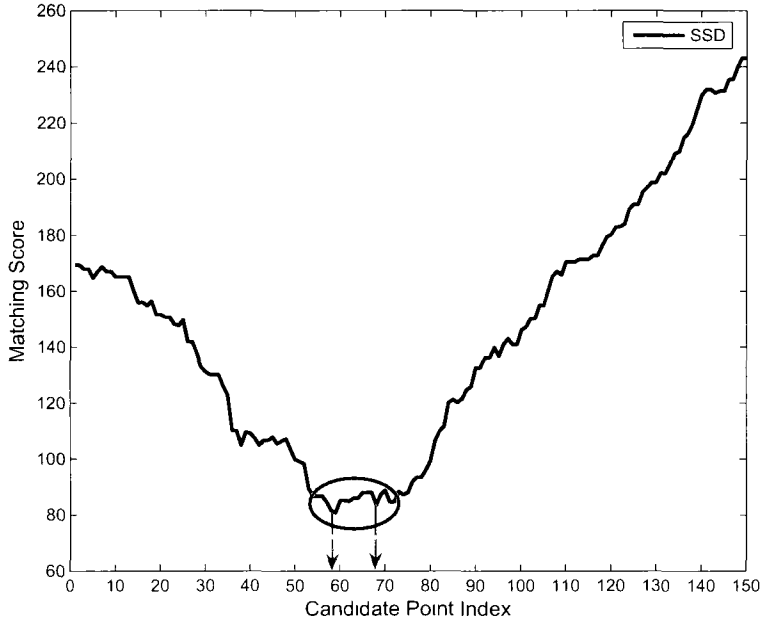


Figure 3.13: SSD measurement

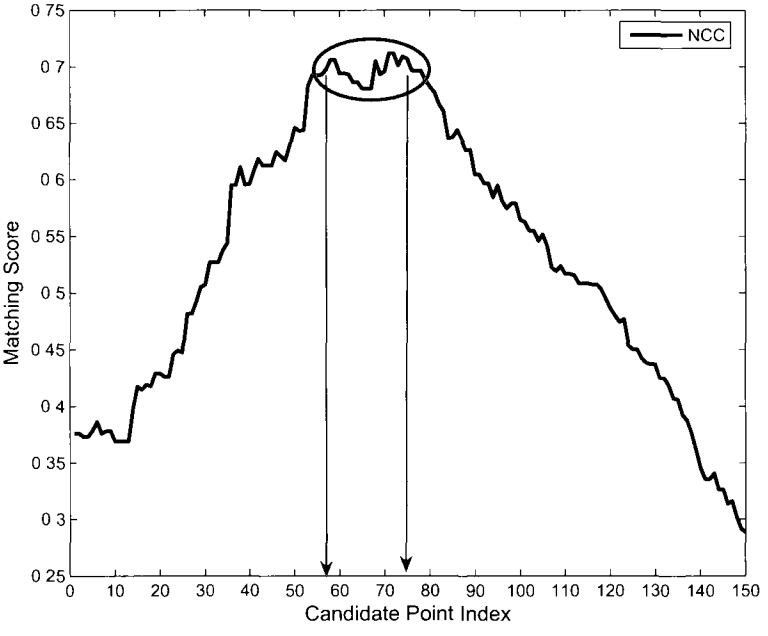


Figure 3.14: NCC measurement

tersection of high surface confidence points measured by different matching methods is non-empty. This phenomenon supports our intuition that real surface points will yield high surface confidence value among different matching methods. The intensity difference and intensity distribution difference matching methods measure the windows similarity in two different ways. Therefore, the combination of intensity difference and intensity distribution difference measurements will have the potential to generate more stable and reliable matching results. In our hybrid matching method, NCC will be applied to measure the intensity distribution difference. As shown in Figure 3.12 and Figure 3.13, SAD and SSD generate similar matching score curves in our application. To optimize computational efficiency, SAD will be used to measure the intensity difference in our hybrid matching method.

Though we decide to use SAD and NCC to build our hybrid matching method, there are still many possible ways to combine the scores. we define some basic properties of effective hybrid matching methods below.

Three basic properties of a valid hybrid method are defined based on the consideration that the hybrid matching score distribution should reflect both matching score distributions calculated by SAD and NCC on the same search segment.

1. For any two points p and q on the same pixel ray r, if p has lower SAD matching score and higher NCC matching score than q, p should have a higher hybrid matching score than q.

2. For any two points p and q on the same pixel ray r, if p has the same SAD matching score as q and has higher NCC matching score than q, p should have a higher hybrid matching score than q.

3. For any two points p and q on the same pixel ray r, if p has the same NCC matching score as q and has a lower SAD matching score than q, p should have a higher hybrid matching score than q.

To achieve all these goals, we use a linear combination of the scores of the two different matching methods. The general hybrid matching score function is given in Equation 3.12

$$Hybrid(p) = \alpha * SAD'(p) + \beta * NCC'(p) \tag{3.12}$$

with the two weighting factors α and β. In order to balance the effect of intensity difference and intensity distribution difference measurements, α and β are both set as 0.5 in our hybrid matching model. $SAD'(p)$ and $NCC'(p)$ are two hybrid matching score

components corresponding to the original SAD and NCC scores. In the SAD matching method, a lower matching score yields a higher surface confidence value but the NCC matching method works in the opposite way. The higher NCC matching score yields a higher surface confidence value. Therefore, we need to transform matching scores in different domains into the same score domain before combining the two matching methods together.

One way to map both SAD and NCC matching scores into the same score range is by scaling. After scaling, both original score curves are mapped into range $[0 \ldots 1]$. The point with the highest NCC score will be assigned 1 after scaling. And point with the lowest NCC score will be assigned 0. And the SAD score will be inversed first before scaling. The scaled SAD and NCC matching score are calculated with Equation 3.13 and Equation 3.14. The special case of a zero denominator will be handled separately. The scaling based hybrid matching score is calculated by Equation 3.15.

$$Scaled\_SAD(p) = 1 - (SAD(p) - SAD\_min)/(SAD\_max - SAD\_min) \tag{3.13}$$

$$Scaled\_NCC(p) = (NCC(p) - NCC\_min)/(NCC\_max - NCC\_min) \tag{3.14}$$

$$Scaled\_Hybrid(p) = 0.5 * Scaled\_SAD(p) + 0.5 * Scaled\_NCC(p) \tag{3.15}$$

The scaled SAD and NCC matching score curves are given in Figure 3.15. For the same pixel ray and search segment range, the score curve from the scaling based hybrid matching method is given in Figure 3.16. In this case, a single and visible peak is shown in the hybrid score curve. And the candidate point with the highest hybrid matching score has both high scaled SAD and NCC matching scores in the circled area of Figure 3.15. This scaled hybrid matching score holds all three basic hybrid method properties and works well for merging SAD and NCC measurements on a single pixel ray.

But the surface confidence value of two candidate points on different pixel rays can not be correctly compared by using the scaling based hybrid matching score. For example, it will fail in the following scenarios:

1. Assume that the pixel rays r_1 and r_2 have exactly the same scaled NCC based matching score curve and the scaled SAD based matching score curve shape of $r1$ is flatter than r_2. Then, for the candidate point p on r_1 and the candidate point q

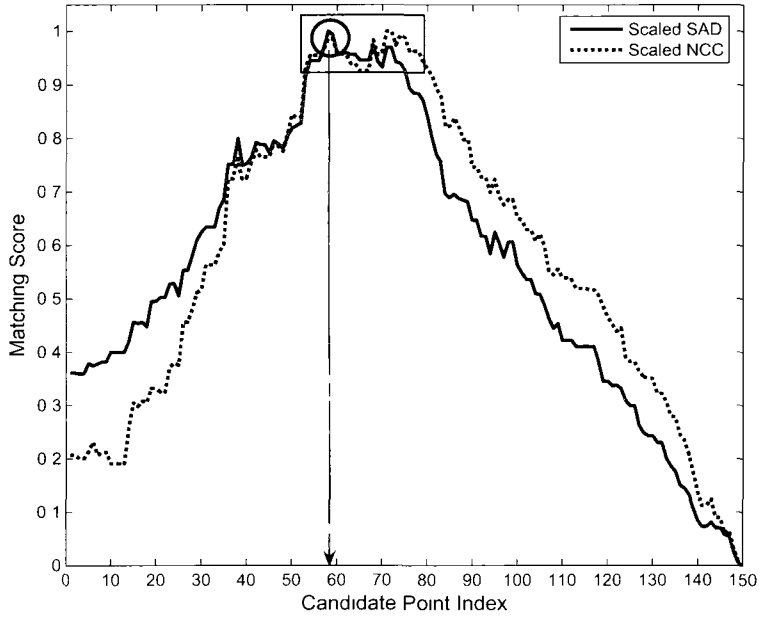


Figure 3.15: Scaled SAD and Scaled NCC

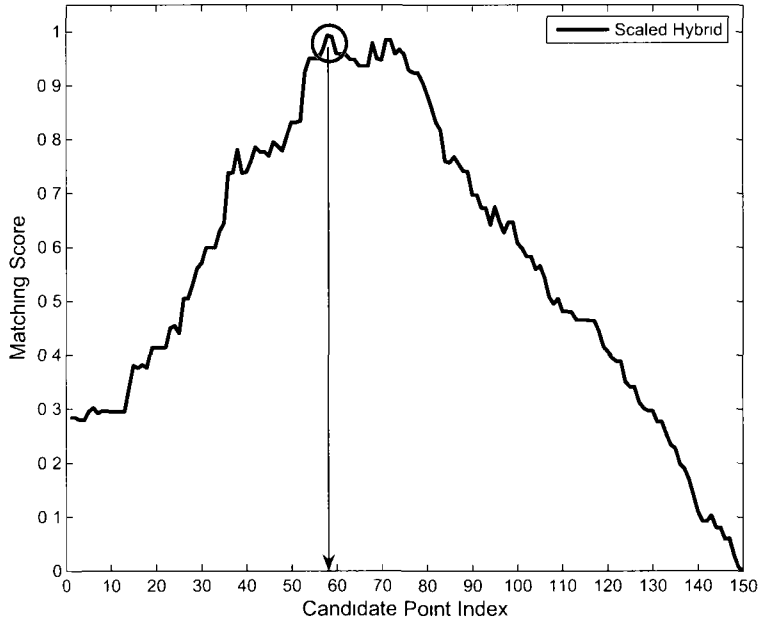


Figure 3.16: Scaling based hybrid matching

on r_2, p should have a lower larger hybrid matching score than q, if p and q have the same SAD and NCC matching score. But the scaling based hybrid model will lead to the opposite result.

2. Assume that pixel rays r_1 and r_2 have exactly the same scaled SAD based matching score curve and the same scaled NCC based matching score curve shape, and p and q are two separate candidate points on pixel ray r_1 and r_2 with the same SAD matching score. If p has a smaller scaled NCC based matching score than q, p should have the smaller hybrid matching score than q. But p and q may be assigned the same scaled hybrid matching score.

In our MVS method, we need to compare the surface confidence of candidate points from different pixel rays in the pointclouds fusion step. Therefore, this scaling based hybrid model is not sufficient for our MVS algorithm even though it works well in single pointcloud estimation without fusion.

Consequently, we use a different hybrid matching score to overcome the weakness of scaling based hybrid matching. In this modified hybrid matching method, each hybrid matching score component should reflect the overall shapes of the SAD and NCC matching score curves. We design a overall hybrid matching method to satisfy this requirement. In this hybrid matching method, the SAD and NCC matching score components are updated. Again, we need to transform the original SAD and NCC matching score into the same domain. For the intensity-based matching component, the original SAD matching score is first inversed so that the higher score of SAD based matching component will lead to the higher surface confidence value. To balance the effect of intensity difference and intensity distribution difference measurements, one is added to the original NCC matching score so that the NCC based matching component will not yield negative values. Then, for each candidate point, its SAD based matching component is calculated by its inverse SAD value over the sum of all inverse SAD values within the search range on the segment. A similar idea is used to calculate the NCC based matching component. The matching components of the overall hybrid matching model are calculated by Equation 3.16 and Equation 3.17. The overall hybrid matching score are calculated by Equation 3.18. Here, M means all candidate points in the search segment. In practice, to avoid zero denominators, a small value (0.001) is added to the denominator of both SAD and NCC matching components calculation.

$$Overall\_SAD(p_i) = \frac{1/SAD(p_i)}{\sum_{j=0}^{M}(1/SAD(p_j))} \tag{3.16}$$

$$Overall\_NCC(p_i) = \frac{NCC(p_i + 1)}{\sum_{j=0}^{M}(NCC(p_j) + 1)} \tag{3.17}$$

$$Overall\_Hybrid(p) = 0.5 * Overall\_SAD(p) + 0.5 * Overall\_NCC(p) \tag{3.18}$$

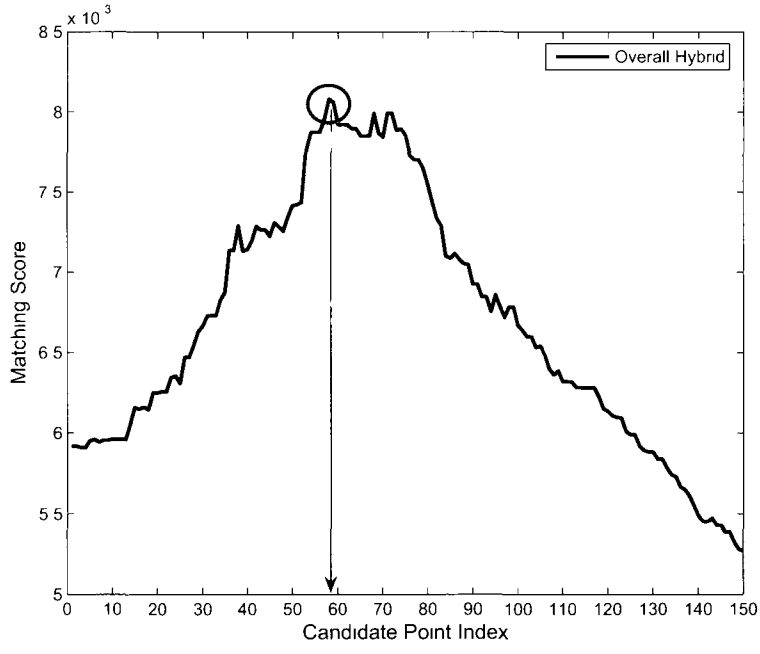


Figure 3.17: Overall hybrid matching which reflects the overall shapes of the SAD and NCC matching score curves in the search segment

In Figure 3.17, the overall hybrid matching score curve is drawn on the same search segment than the previous matching score estimation in Figure 3.15 and Figure 3.16. As we can see, it also leads to a visible peak with a high similarity score for our example. A stable hybrid matching method will help to match projection pixels of the 3D point in our MVS pointcloud generation. A quantative evaluation of different matching methods will be given in Section 3.4.

There is a well-known trade-off in setting the best window size for matching, e.g., Goesele et al. [26] point out that a larger window size causes smoothing in the reconstructions. It will also cause smoother matching score curves which will help selecting a peak and to remove low surface confidence points. But a larger window size will also lead to larger matching error when we reconstruct non-planar surfaces. Based on our experiments, we found a 5×5 window size to achieve to a good reconstruction result and we apply this window size in all our reconstructions.

3.3.4 Low-and-High Confidence Fusion

Up to now we discussed how multiple initial pointclouds can be generated from different reference views and the surface confidence of each candidate point can be calculated by the similarity between its projection pixels. Therefore, multiple MVS pointclouds can be extracted from the initial pointclouds with a confidence measure based on the hybrid matching score. Here, we present two new fusion strategies which can be used in the MVS method based on fusing multiple MVS pointclouds. In the MVS method with fusion, multiple MVS pointclouds are generated to represent the same object from different reference views if all points in each MVS pointcloud have a surface confidence value. Our method gives an indication of the surface evidence, i.e., whether the candidate point is on the object surface or not. Further, all the MVS pointclouds will be transformed into the coordinate system of the fusion reference view which is a reference view defined in the fusion step. Then, the surface evidence from different reference views will be combined to generate a final pointcloud by considering the consistency of the input pointclouds. In general, candidate points with high surface confidence value will be retained in a MVS pointcloud generated from a reference view. Later, those points with high surface confidence will be used to assist answering the question in the fusion step: where the real object surface should be? The novelty of our low-and-high confidence fusion strategy is to use both low and high confidence points to assist the final pointcloud extraction from multiple MVS pointclouds based on different reference views. Equal to points with high surface confidence, points with low surface confidence can also provide useful information about the structure of the object surface, just in the opposite way. Using the points with low surface confidence, we can estimate the area where the object surface should not be.

Our MVS method based on low-and-high confidence fusion strategy can be divided into two steps as shown in Figure 3.18. First, multiple MVS pointclouds are generated from different reference views by recording all the candidate points and their surface

confidence value. The surface confidence value reflects the surface likelihood of each candidate point. In the second step, an initial pointcloud IPC_{fusion} is generated from the fusion reference view. All candidate points in IPC_{fusion} are defined as fusion candidate points. The final pointcloud is generated by extracting the surface point from these fusion candidate points. All MVS pointclouds generated from different reference views are transformed into the fusion reference view. And all points from MVS pointclouds are marked as predefined candidate points. The surface confidence value of each fusion candidate point Q_{fusion} will be calculated by averaging the surface confidence value of all neighboring predefined candidate points of Q_{fusion}. The final pointcloud is extracted by choosing the fusion candidate point with the highest surface confidence on each pixel ray generated from an object pixel in the fusion reference image.

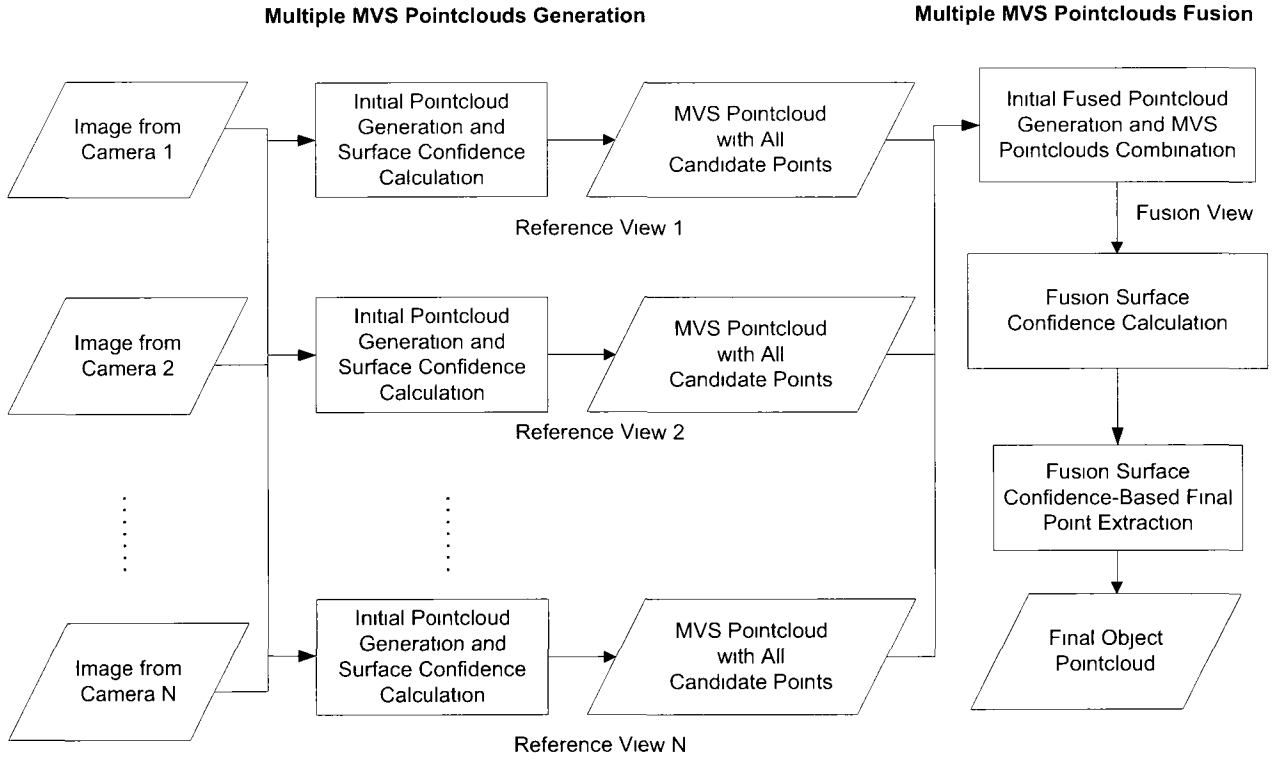


Figure 3.18: Confidence-based MVS procedure

In the ProFUSION25 based 3D model reconstruction, twenty-four MVS pointclouds will be generated by setting each camera as the reference view once except the center camera 12. For each MVS pointcloud generation, there will be one reference view and 25 black and white images including one reference image and twenty-four non-reference

images. To calculate a MVS pointcloud from a reference view i, an initial pointcloud will be built from reference view i first. Then, all the candidate points which map to background pixels in the reference image will be removed from the initial pointcloud. Using the camera projection matrix, each remaining candidate point will be projected to one object pixel in the reference image and twenty four pixels in the non-reference images. For each candidate point P, its surface confidence value $point\_confidence(P)$ is calculated as the average similarity between its projection pixel in the reference image and each of its projection pixel in the non-reference images as shown in the Equation 3.19. The similarity of two pixels is calculated by the similarity of their matching windows. And the similarity of matching windows is measured with our overall hybrid matching score. Here, K means the total number of all valid matching window pairs of the candidate point P between its projection pixel in reference image and its projection pixels in the non-reference images. And the matching window will be marked as valid, if all pixels in the matching window are object pixels.

$$point\_confidence(P) = \frac{\sum_{k=0}^{K} Overall\_Hybrid(P_k)}{K} \tag{3.19}$$

In the fusion step, a final pointcloud will be extracted from multiple MVS pointclouds generated from twenty-four different reference views. In our experiments with the ProFUSION25, the central camera 12 is defined as the fusion reference view. An initial fused pointcloud IPC_{fusion} will be generated in the coordinate system of the central camera (camera 12). Each fusion candidate point is projected into a pixel in the fusion reference image. All the fusion candidate points that map to a background pixel will be removed from IPC_{fusion}. All MVS pointclouds are transformed into the coordinate system of camera 12. After that, all transformed candidate points will be marked as predefined candidate points. A predefined candidate points $P_{predefined}$ is defined as a neighbor to a fusion candidate point Q_{fusion}, if $P_{predefined}$ and Q_{fusion} are projected into the same object pixel in the fusion reference image and the depth difference between $P_{predefined}$ and Q_{fusion} is smaller than the half of the interval depth between two consecutive candidates from the same pixel ray. In our experiments, the interval depth is set as 1 millimeter. As shown in the Equation 3.20, the surface confidence of each fusion candidate point Q_{fusion} is calculated as the average of surface confidence value of all predefined candidate points close to Q_{fusion}. Here, N is the point set of all predefined candidate points which are neighbors to Q_{fusion}. P_n is a predefined candidate point in N.

$$point\_confidence(Q_{fusion}) = \frac{\sum_{n=0}^{N} point\_confidence(P_n)}{N} \tag{3.20}$$

3.3.5 Density-Based Fusion

Our density-based fusion assumes that the 3D space containing the object's surface will have more high confidence candidate points than non-surface space, when we combine all high confidence points from different MVS pointclouds together. The procedure of our MVS method based on the density-based fusion strategy is shown in Figure 3.19

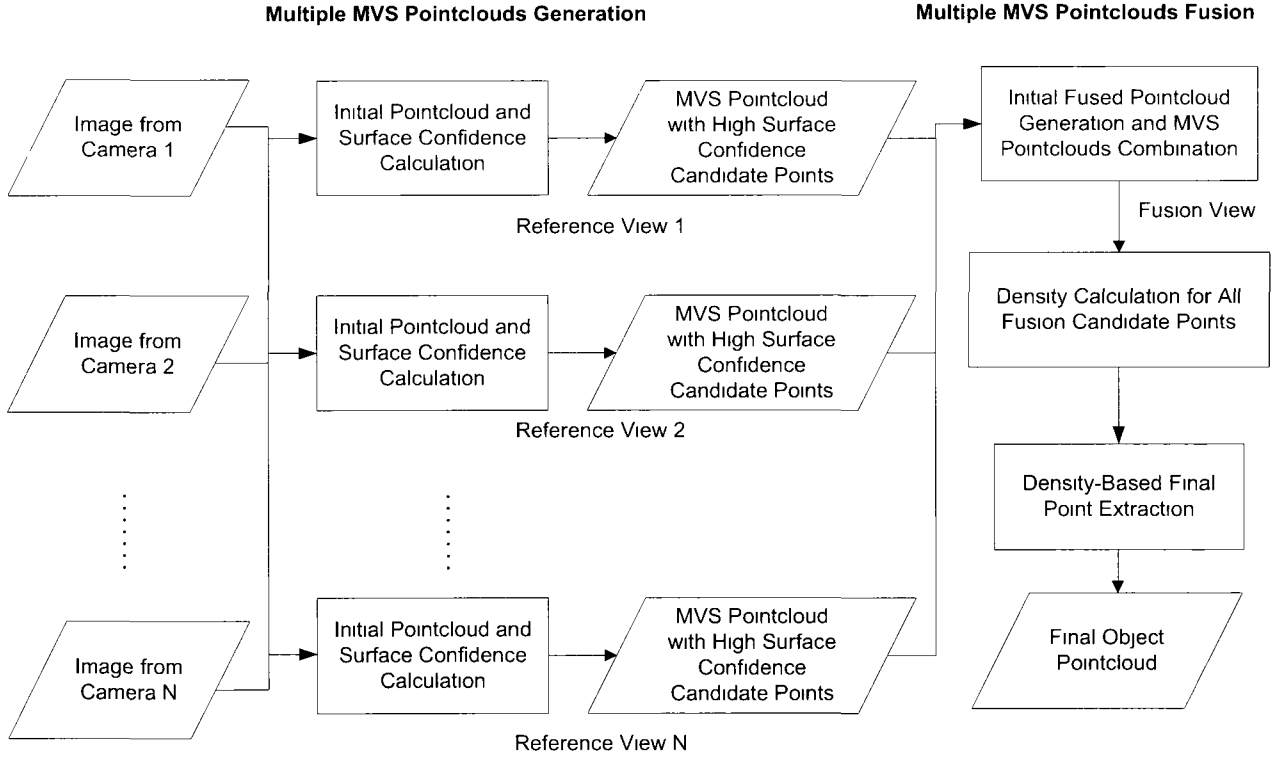


Figure 3.19: Density-based MVS procedure

In the MVS pointcloud generation step, our MVS method with density-based fusion is similar to the MVS method with low-and-high confidence fusion. First, initial pointclouds are defined by different reference views and candidate points which violate the visual hull constraint will be removed from the initial pointcloud. The surface confidence value will

be calculated for all remaining candidate points based on the similarity of their projection pixels. But, we will not record all candidate points into the MVS pointclouds. For each pixel ray r from a object pixel in the reference view i, only the candidate point with the highest surface confidence value will be recorded into the MVS pointcloud i.

In our density-based fusion, the final pointcloud is also generated by extracting one fusion candidate point from each pixel ray generated from an object pixel in the fusion reference image. For each fusion candidate point Q_{fusion}, a density value of predefined candidate points on Q_{fusion} will be calculated. And the density value of Q_{fusion} is measured by all predefined candidate points which map to the same object pixel as Q_{fusion} in the fusion reference image. Let $P = \{P_j | j = 1 \dots n\}$ be the point set of all predefined candidate points mapping to an object pixel p_{object} in the fusion reference image and let $Q = \{Q_i | i = 1 \dots m\}$ be the point set of all fusion candidate points mapping to the same object pixel p_{object}. For each fusion candidate point Q_i, its inverse density is calculated as the absolute sum of its weighted oriented distance to all pre-candidate points in P. The oriented distance between P_i and Q_i is calculated by Equation 3.21. The distance is positive, if P_i's depth value is smaller than or equal to Q_i's depth value, negative otherwise.

$$ori\_dist(Q_i, P_j) = \begin{cases} Euclidean\_dist(Q_i, P_j) & \text{if } P_j\text{'s depth} \leq Q_i\text{'s depth} \\ \text{-}Euclidean\_dist(Q_i, P_j) & \text{otherwise} \end{cases} \tag{3.21}$$

For the density value to reflect the surface confidence of each predefined candidate point, the predefined candidate point with higher surface confidence will have larger weight than predefined candidate points with lower surface confidence. Each orientation distance $ori\_dist(Q_i, P_j)$ will be weighted by the surface confidence value of the predefined candidate point P_j. The inverse density of each fusion candidate point is calculated by Equation 3.22. For each pixel ray, the fusion candidate point with the smallest inverse density value will be accepted as the surface point in the final pointcloud. For a smooth final pointcloud, the density value of a fusion candidate point Q_{fusion} can be calculated by using all predefined candidate points which map to the object pixel p_{object} and p_{object}'s neighboring pixels. Then, different additional weight should be assigned to predefined candidate points which map to different object pixels. To generate a sharp result, this kind of smoothing is not applied in our density-based fusion.

$$inverse\_density(Q_i) = |\sum_{j=0}^{m} point\_confidence(P_j) * ori\_dist(Q_i, P_j)| \tag{3.22}$$

3.4 Evaluation and Result

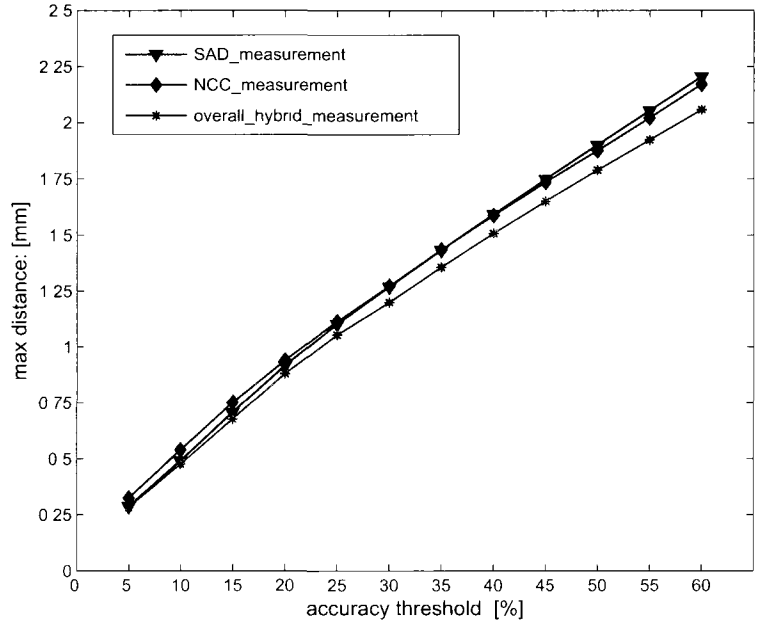
We evaluate our hybrid matching method and fusion strategies by comparing the quality of the 3D pointclouds generated from different matching methods and fusion strategies. To evaluate the quality of a reconstruction model, the difference between the 3D pointcloud and the ground truth is measured by applying the evaluation methodology introduced by Seitz et al. [59]. In our evaluation, the experimental multi-view images are generated from the ProFUISON25 instead of using the Middlebury dataset [58]. Therefore, the performance of 3D reconstruction based on the ProFUSION25 can also be estimated. Because the triangulation 3D laser scanner (VIVID 910) can generate a highly accurate model of 3D objects compared to MVS, the 3D model form scanning is chosen as the ground truth model in our quality evaluation. The accuracy of the VIVID 910 given by the manufacturer [33] is ± 0.22 mm on the x-axis, ± 0.16 mm on the y-axis and ± 0.1 mm on the z-axis. The scanner obtains up to 640×480 individual points per scan. In the acquisition of the ground truth model, the middle lens of the scanner with measurement distance from 600 mm to 2500 mm is used with auto focus. The distance between the scanner and the modeling object is around 700 mm. In the multi-view image acquisition, the distance between the ProFUSION25 and the object to be modelled is also around 700 mm.

In the quality estimation of the 3D reconstructions, two measurements will be used: one measures the accuracy of the reconstruction by how close the 3D pointcloud is to the model obtained with the scanner. The other measures the completeness of the reconstruction by how much of the 3D model generated by the 3D scanner is covered by the 3D pointcloud. Because the model from the scanner is generated with a high resolution sampling of the object surface, the distance between a reconstruction point P and the model from the scanner can be approximately calculated as the distance between the point P and the nearest vertex on the ground truth model. The accuracy of the reconstruction is evaluated by the distance $d_{accuracy}$ such that $X\%$ of points of the reconstruction have a smaller offset than $d_{accuracy}$ from the ground truth model. The parameter X is treated as the accuracy threshold. The completeness of the reconstruction model is evaluated by the rate of completeness $Y\%$ such that $Y\%$ of points on the reconstruction model have a smaller offset than $d_{completeness}$ from the ground truth. The parameter $d_{completeness}$ is treated as the completeness threshold.

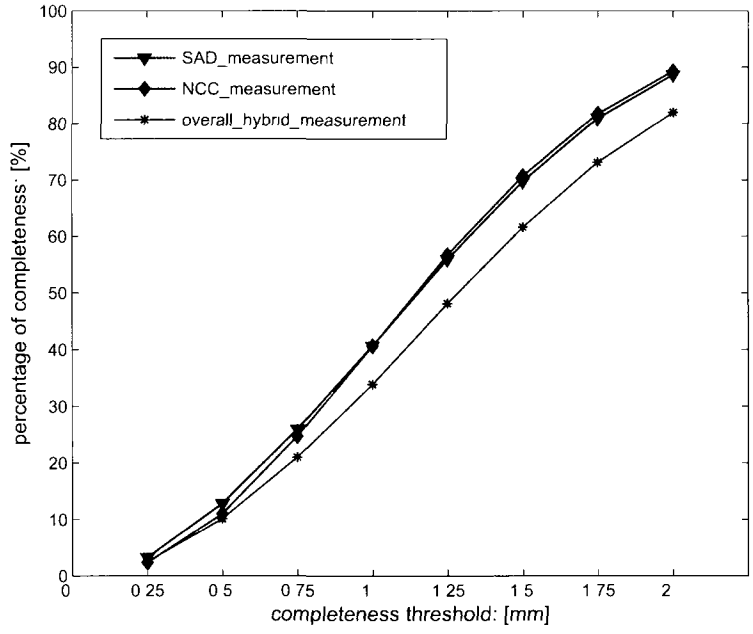
3.4.1 Evaluation of Hybrid Matching Method

In the evaluation of the matching method, three object pointclouds are generated by using twenty-five multi-view images of a coffee can taken from the ProFUSION25. These pointclouds are generated by using SAD, NCC and our hybrid matching method, respectively. The ground truth of the coffee can is generated from the scanner VIVID 910. In the generation of these pointclouds, the object point is simply chosen as the point with the highest surface confidence value on each pixel ray in the initial pointcloud generated from the reference view camera 12. To avoid very poor matches, a matching score threshold is used for an object point. We arbitrarely chose a SAD matching score of at most 10 and a NCC matching score of at least 0.5. For the object pointcloud based on the hybrid matching method, the same thresholds are used for the SAD and NCC matching scores of the chosen object point. Because the object pointcloud is reconstructed from a single reference view and without any post-processing, we set the maximum accuracy threshold as 60% and the maximum completeness threshold as 2 millimeter in the evaluation of matching methods. Further, the accuracy and completeness will be improved in the later step of fusing multiple MVS pointclouds generated from different reference views.

In Figure 3.20, the object pointcloud using our hybrid matching method shows a smaller maximum distance $d_{accuracy}$ than pointclouds generated by using SAD and NCC matching methods. For the completeness measure, the pointclouds using SAD and NCC matching method show higher completeness rate than the pointcloud using the hybrid matching method. But the higher completeness of SAD and NCC matching method may be caused partially by very noisy points. And the missing points could likely be recovered in a multiple pointclouds fusion procedure. Based on our experiments, the pointcloud generated with the hybrid matching method leads to a better reconstruction after applying the fusion procedure of multiple pointclouds.



(a) Accuracy evaluation



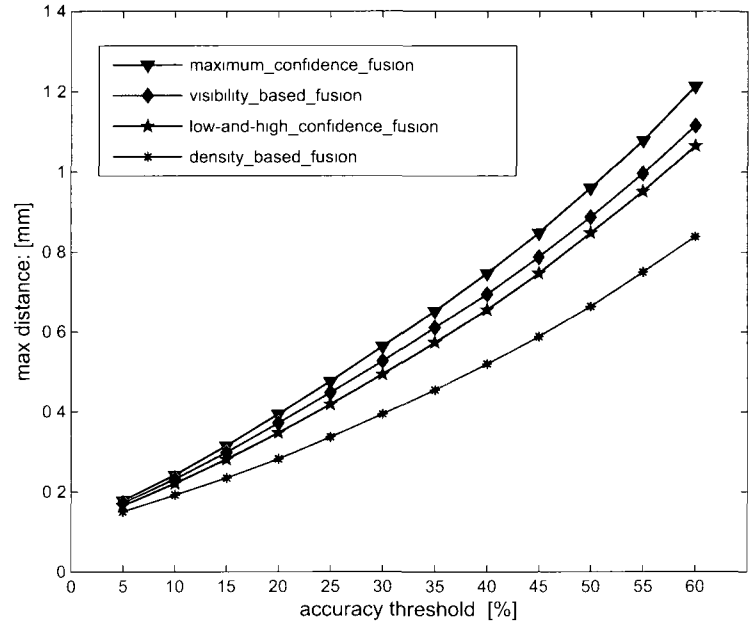
(b) Completeness evaluation

Figure 3.20: Matching method comparison

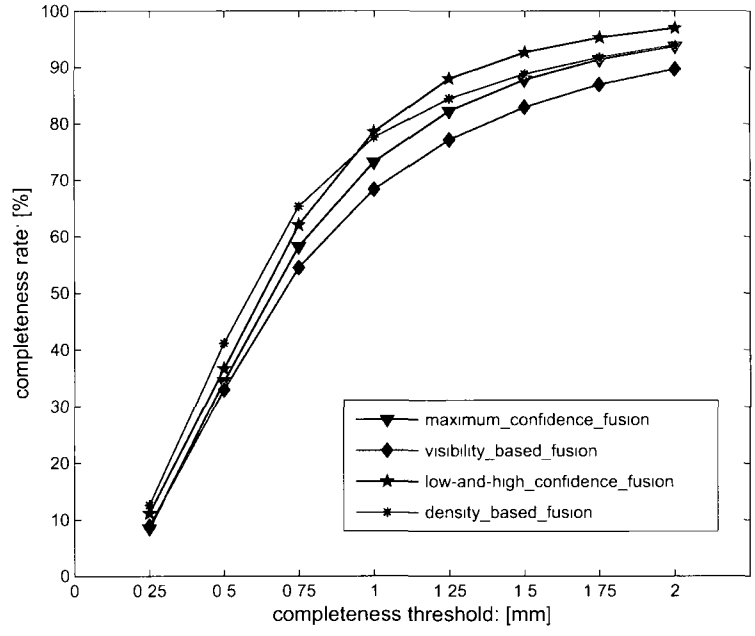
3.4.2 Evaluation of Fusion Strategies

To evaluate our low-and-high confidence and density-based fusion strategies two additional fusion strategies are included as references One additional fusion strategy is called maximum confidence fusion The only difference between maximum confidence fusion and low-and-high confidence fusion is choosing the candidate point with highest surface confidence on each pixel ray instead of choosing all candidate points in the MVS pointcloud generation The other fusion strategy is called visibility-based fusion It was introduced by Merrell et al [50] and it fuses pointclouds by minimizing violations of visibility constraints In the fusion strategies evaluation, the hybrid matching method is applied to calculate the surface confidence value for all candidate points in the MVS pointcloud The same multi-view image set and ground truth model which are used in the matching method evaluation are used in the evaluation of different fusion strategies Four raw object pointclouds are generated from these images by using different fusion strategies To purely evaluate the fusion strategies, raw object pointclouds are generated without any post-process, such as outlier removal, noise removal, hole filling etc The accuracy and completeness estimation based on these raw object pointclouds are given in Figure 3 21 In the experiments, the density-based fusion strategy leads to the most accurate model, and more complete model comparing to the visibility-based fusion strategy And our low-and-high confidence fusion strategy generates the most complete model, and a more accurate model comparing to the visibility-based fusion strategy and maximum confidence fusion strategy

Because the same multi-view images are used in the MVS pointcloud generation and final pointcloud generation, we can see that the fusion procedure achieves a significant improvement of the quality of the reconstruction by comparing Figure 3 20 and Figure 3 21 After fusing the pointclouds, the accuracy and the completeness of the reconstruction are both improved by about 40% and 10%, respectively Two object pointclouds and one scan model of the same coffee can are shown in Figure 3 22 These object pointclouds are generated by using our hybrid matching method One is a single MVS pointcloud generated from a reference view with twenty-five multi-view images The other one is calculated by fusing twenty-four MVS pointclouds which are generated by using the same twenty-five multi-view images with different reference views To fuse multiple pointclouds, the maximum confidence fusion strategy is applied As we can see, the pointcloud with the fusion step has better completeness and accuracy than the pointcloud from the single reference view



(a) Accuracy evaluation



(b) Completeness evaluation

Figure 3.21: Fusion strategy comparison: raw pointcloud

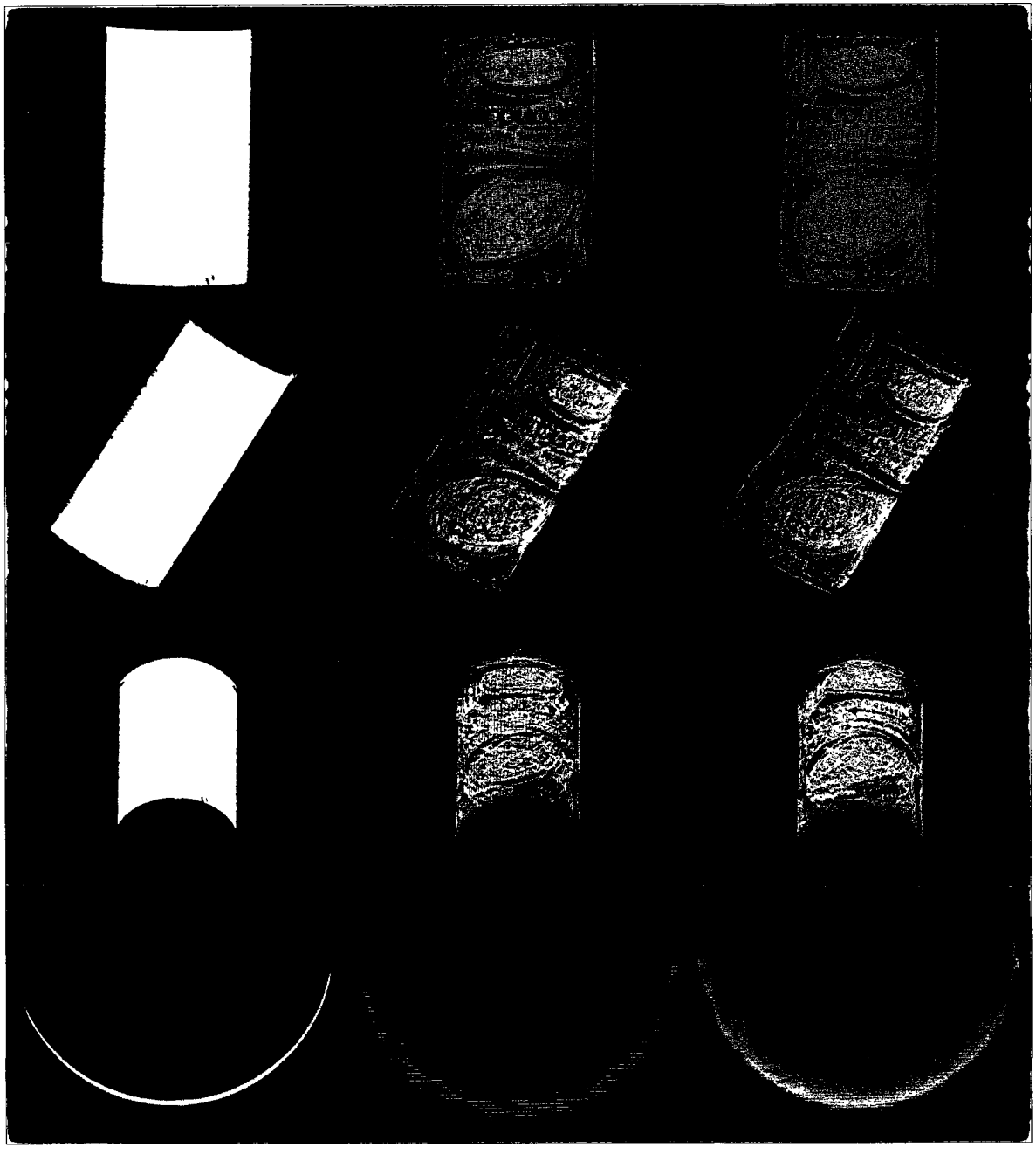
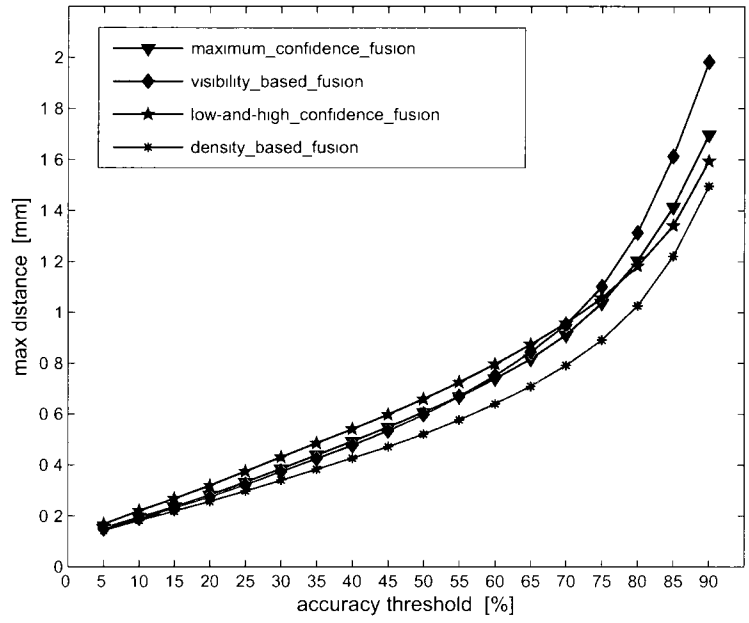


Figure 3.22: The advantage of fusion: from left to right, there are three models of the same object generated from 3D scanner, MVS reconstruction without fusion and MVS reconstruction with fusion, respectively.

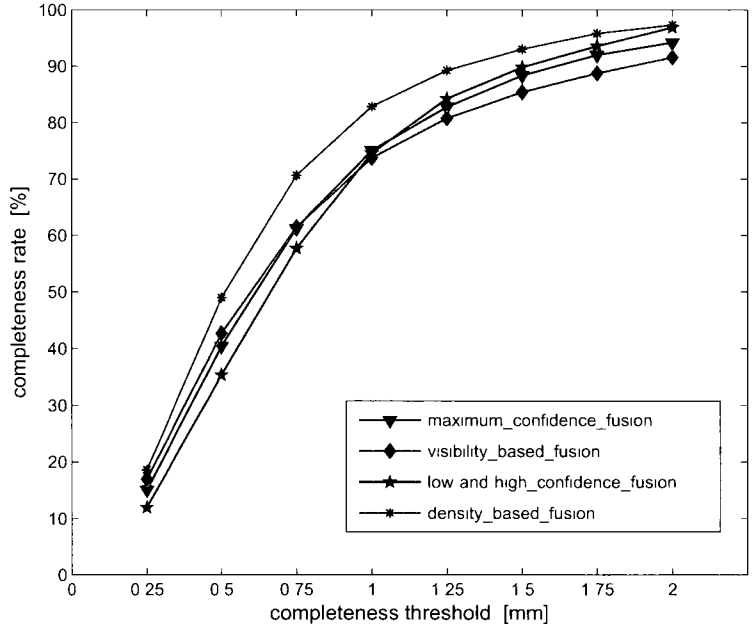
3.4.3 Reconstruction Results

A post-process is applied on the raw models generated from different fusion strategies. All the post-procedure functions which include outlier removal, noise removal, hole filling and mesh generation are provided by using the Geomagic Studio 9. Geomagic Studio is designed to process raw scan data. The accuracy and completeness estimation based on refined object pointclouds is given in Figure 3.23. It shows the density-based fusion strategy leads to the better reconstruction result than the other three strategies. And our low-and-high confidence fusion strategy achieves better reconstruction result with a threshold setting of high accuracy and completeness than the other two strategies. Noise points far from the true surface will be easily detected by using the low-and-high confidence fusion strategy, because there will be more low confidence candidate points in the area far from the true surface.

In Figure 3.24 and Figure 3.25, one ground truth and four MVS models of a coffee can and a plush dinosaur are given. Here, we code the MVS application based on the CPU programming, it costs 10 hours (CPU: Intel Q6660, Memory: 4GB) to generate each MVS model. Taking the advance of sweeping plane technique, the CPU programming can be replaced with the GPU programming. Therefore, the computational time can be much shorter. The ground truth models are generated from the 3D scanner VIVID 910. The other four MVS models are generated by using different fusion strategies. The models generated from the density-based and low-and-high confidence fusion strategy show clearer texture than the other models. They have more similar shape to the ground truth compared to the other models generated by maximum confidence and visibility-based fusion strategies. Based on our experiments, we conclude that our density-based and high-and-low confidence fusion strategies are more stable than the visibility-based and maximum confidence fusion strategies. To generate the best refined model, the density-based fusion strategy is used in our 3D reconstruction framework.



(a) Accuracy evaluation



(b) Completeness evaluation

Figure 3.23: Fusion strategy comparison: refined pointcloud

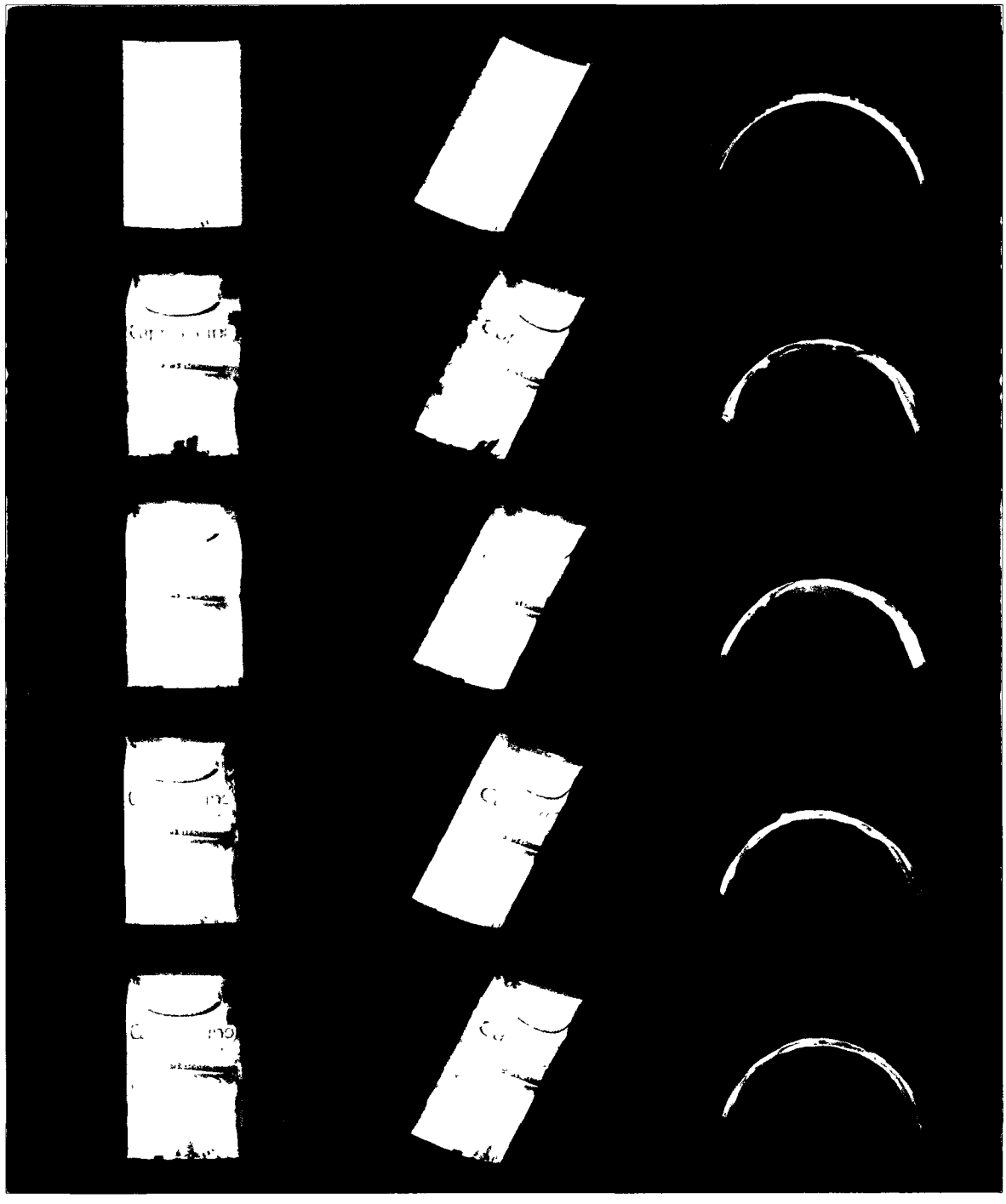


Figure 3.24: Fusion strategy comparison presented by a coffee can: from the top to the bottom, there are five models of the same object generated points obtained with 3D scanner, visibility-based fusion, maximum confidence fusion, low-and-high confidence fusion and density-based fusion, respectively.

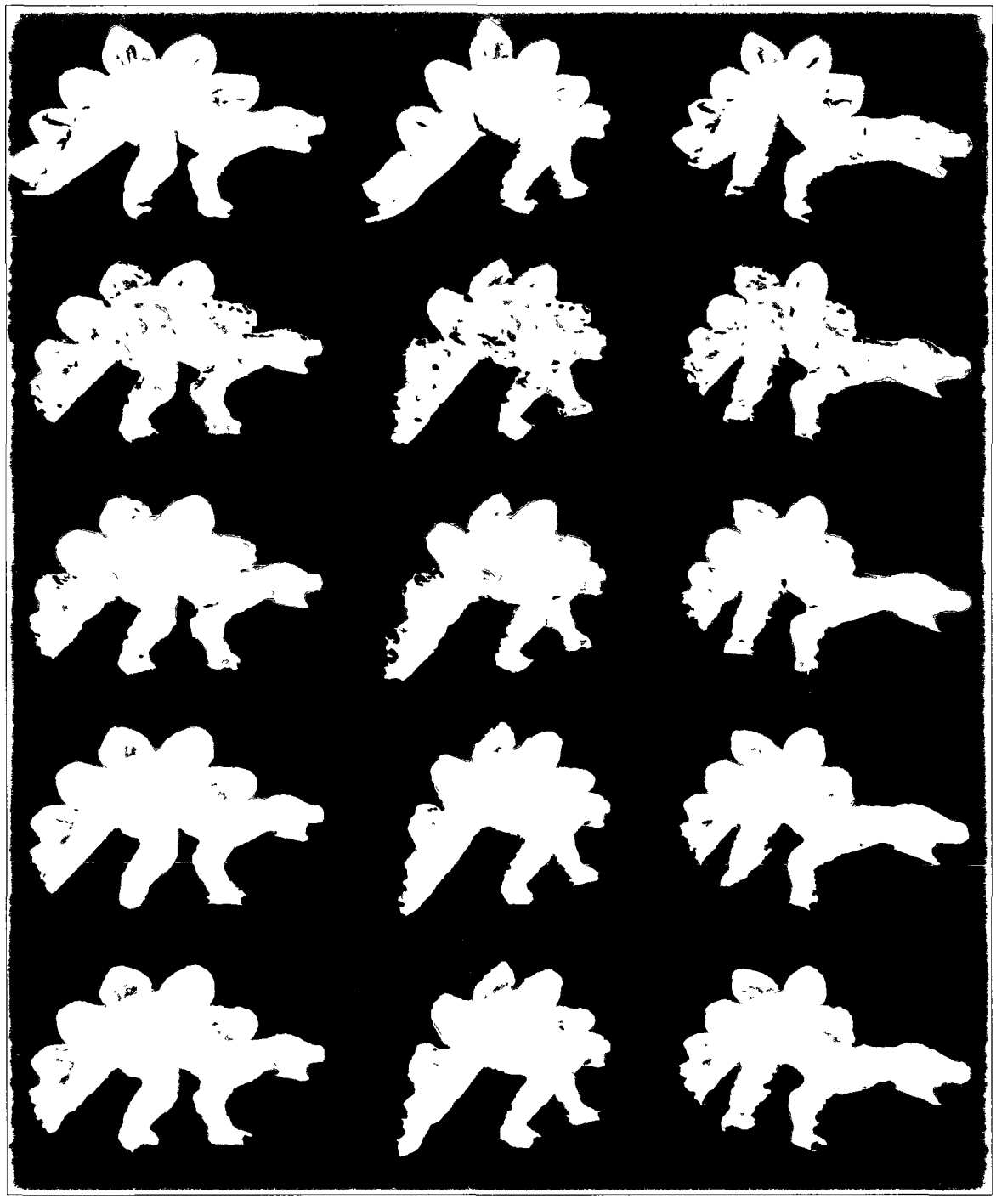


Figure 3.25: Fusion strategy comparison presented by a plush dinasour: from the top to the bottom, there are five models of the same object generated points obtained with 3D scanner, visibility-based fusion, maximum confidence fusion, low-and-high confidence fusion and density-based fusion, respectively.

Chapter 4

Motion Detection

4.1 Overview

In this section, we will track a surface deformation of an object through time consecutive stereo images. A sequence of pointclouds are generated from the stereo images and each pointcloud relates to a reference image. Therefore, each point in the pointcloud projects into a pixel in its reference image. Our goal is to detect the motion between two consecutive pointclouds PC^1 and PC^2 at time t^1 and t^2. Then, point P^1 in PC^1 will be assigned a motion vector which points to P^2 in PC^2 which is the location of P^1 at time t^2. PC^1 is a 3D pointcloud generated from a set of object images $[{I^1}_r, {I^1}_1, \ldots, {I^1}_n]$ at time t^1, while PC^2 is generated from a set of object images $[{I^2}_r, {I^2}_1, \ldots, {I^2}_n]$ at time t^2. Our motion detection framework is shown in Figure 4.1. The intensity-based tracking of feature points is performed in the reference images ${I^1}_r$ and ${I^2}_r$ and mapped to the pointcloud PC^1 and PC^2 to obtain 3D reference points. In the geometric matching step, the points tracked by intensity-based matching will be used as reference points for matching points by isometry-based matching.

In Section 4.2, reference points are generated from the intensity-based tracking. The geodesic distance calculation and validation is discussed in Section 4.3. Then, the window-based isometric matching method and its extension will be introduced in Section 4.4. Finally, the results of two sequences, an isometric and a non-isometric surface deformation, will be represented in Section 4.5.

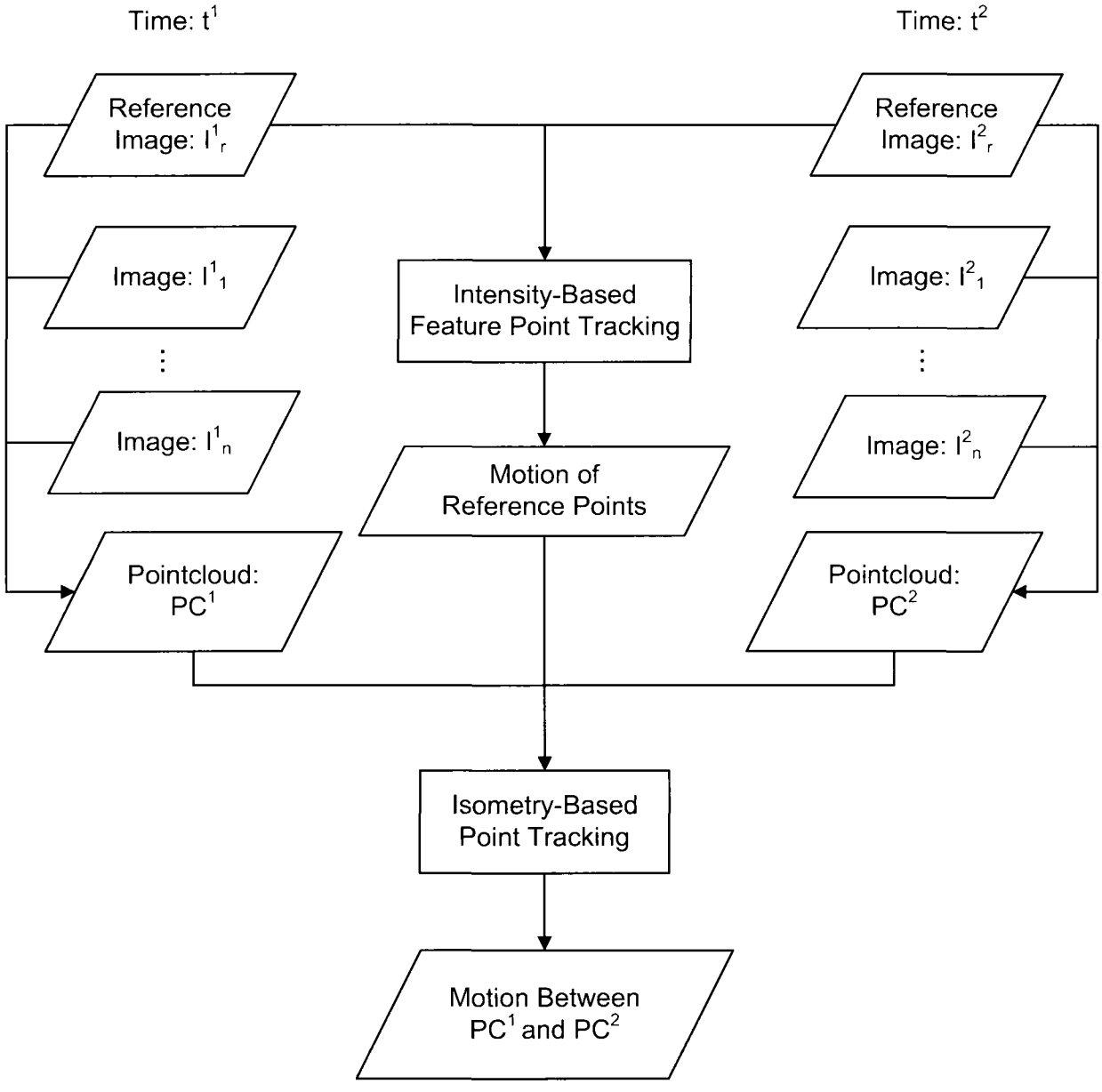


Figure 4.1: Overview of our motion detection approach

4.2 Feature Matching

As we mentioned in Subsection 2.2.3, to apply the isometry constraint in detecting the motion of a deformed surface, at least three non-collinear points of surface should be tracked already. Using the projection relationship between the pointcloud and it reference image, a 3D feature point can be tracked by tracking its projection pixels. In our motion detection framework, a intensity based feature matching algorithm is used to track reference points.

In this thesis, all feature pixels are selected and tracked by applying the KLT feature tracking algorithm [70, 61]. Instead of tracking a single pixel p, KLT feature tracking algorithm tracks an $N \times N$ pixel based feature window centered at the pixel p. To solve the problem of intensity invariance, it uses a residue monitoring strategy. Only the features with stable intensity will be considered in the tracking step. To solve the problem of windows warping, it applies the affine mapping between feature windows. Candidate feature points are selected, if its gradient matrix has two large eigenvalues. This feature point selection strategy rejects the feature windows with roughly constant intensity and unidirectional texture pattern. And it reserves the feature windows with salt-and-pepper textures and corners. In Figure 4.2, the feature points between reference image I^{1^r} and $I^2{}_r$ are tracked by the KLT feature tracking algorithm and motion vectors for these feature points are shown.

(a) Paper (b) Plush dinosaur

Figure 4.2: KLT reference points. The magnitude of the motion vectors is multiplied by 10.

4.3 Geodesic Distance Calculation and Approximation Validation

4.3.1 Geodesic Distance Calculation

As mentioned above, to track a surface point P based on an isometry constraint, the geodesic distance from P to at least three non-collinear matched reference points need to be known firstly. In practice, to calculate the exact geodesic distance between two points on a surface is time and space costly. Mitchell et al [51] presented an algorithm to calculate the exact geodesic distance on a polyhedral surface. Their algorithm cost $O(n^2 logn)$ in time and $O(n^2)$ in space. Here, n is the number of edges of the surface. More recent work is towards computing the approximations of exact geodesic distance. The advantage of the approximation are computational cheaper and more practical. Our geodesic distance calculation is based on an idea of Lanthier [40]. He approximates the geodesic distance of two vertices P and Q on a polyhedron surface S by using the shortest path in a corresponding graph G generated from S. The graph is constructed with all the vertices and edges of the polyhedral surface as graph nodes and graph edges of G, respectively. To increasing the accuracy of the approximation, additional points can be added onto the edges of G and generate a new denser graph G' by interconnecting these points.

In our approach, a sequence of dense pointclouds of the deformable object is generated based on images of the object. For each pointcloud PC, it represents a sampling of the object surface S. In each pointcloud PC, one traditional way of the geodesic distance calculation is to generate the surface mesh from PC, firstly. Then a graph can be extracted based on the topological structure of the surface mesh by mapping all the mesh vertices into graph nodes and mapping all the mesh edges into graph edges. Here, we call the graph generated from the topological structure of the sample points on the mesh of S as the surface graph of S, notated as G_S. For each two points P and Q in PC, If the sampling is dense enough, their geodesic distance can be approximately calculated as their shortest path length in the surface graph G_S.

In instead of using surface graph G_S, our geodesic distance calculation is based on a pixel graph G_P which is a 2D graph formed by the projected pixels in the reference image I_r of the pointcloud PC. The pixel graph G_P is built to embed the topological structure of the sample points on the object surface S. A pixel in I_r is marked as a valid pixel if it has a mapping 3D point in PC. All the valid pixels will be treated as nodes in G_P.

The connectivity of the G_P is generated from the triangulation of the valid pixels in the reference image I_r, as shown in Figure 4.3. For each valid pixel p_0, it will be connected to its valid neighbor pixels. Therefore, for each node in G_P, it will have at most eight degree. And only valid pixels will be treated as nodes in the G_P.

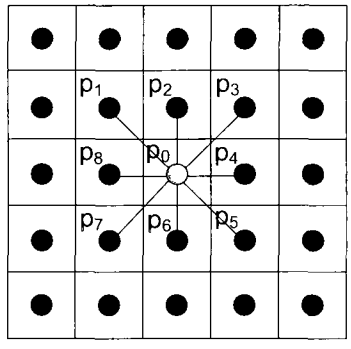


Figure 4.3: Connectivity of pixel graph

To embed the geodesic distance information in the G_P, all the edges of G_P are calculated as the mapping Euclidian distance in the pointcloud PC. As shown in Figure 4.4, e is the edge between two valid pixel p and q in G_S. 3D point P and Q are mapping points of p and q in the pointcloud PC. The length of e is calculated as the Euclidean distance of point P and Q in PC.

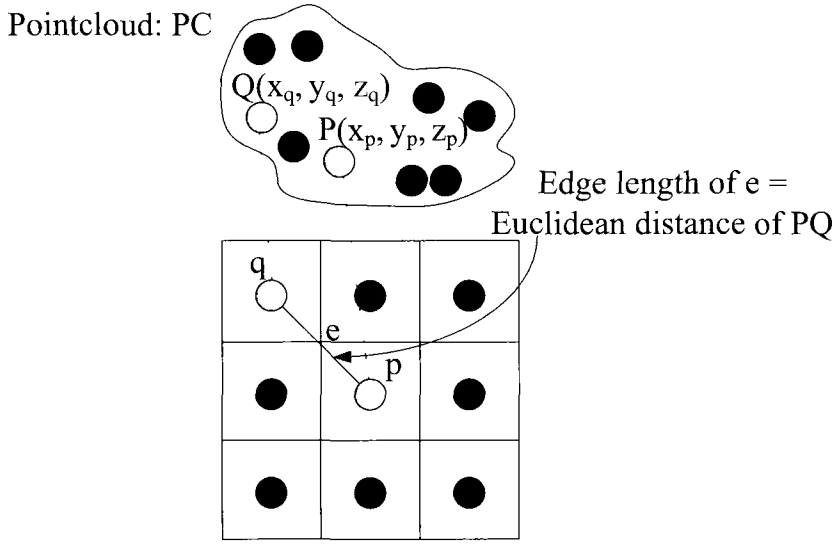


Figure 4.4: Edge length calculation

Once the pixel graph G_P is created from the pointcloud PC. The calculation of the

geodesic distance of two point P and Q is transformed into finding the shortest path between P and Q's projecting pixels in G_P Finding the shortest path in a graph G is a fundamental question in the graph theory The problem can be classifies by the number of source and destination nodes of G And it also can be classified by the weight and direction of the edge In our application, all the edges are weighted by the Euclidean distance calculated from the node's reprojeciton points And every edge is walkable from both directions in G_P To calculate the geodesic distance from a reference point to more than one points in pointcloud PC, we need to solve a one-to-all shortest path problem In our approach, the Dijkstra's algorithm [18] is applied to solve the shortest path problem in $O((n + m)logn)$ time Here, n means the number of vertices in G_P m means the number of edges in G_P The core idea of the Dijkstra's algorithm is to sequentially decrease the estimated shortest distance for each destination point until it finds the actual shortest path At the beginning, all the vertices will be added in an unvisited list Each time, the vertice with the closest distance to the source point will be removed from the unvisited list and marked as visited vertice The estimation of the shortest distance to its connected vertices will be updated The recurrence is stopped when all vertices are visited

4.3.2 Validation of Geodesic Distance Approximation

In our application, the approximate geodesic distance is calculated on a pixel graph G_P based on a reference image I_r and its related pointcloud PC G_P reflects the geometric structure of an object reconstruction surface The reconstruction surface is an approximation of the real object surface based on the sampling PC of the real object surface Therefore, there are two approximations in the geodesic distance calculation One is using the reconstruction surface to approximate the real surface The other approximation is using the pixel graph to approximate the reconstruction surface

In our image-based reconstruction, brightness variations and insufficient texture will lead to few KLT features and therefore few reference points This will increase the average distance of a pixel from the reference points Additionally, holes will be present in the stereo-based reconstruction which causes the shortest path on G_p to increase and hence the geodesic may become poorly approximated Figure 4 5 shows a hole h in an area where the stereo result is unreliable and hence no points are added to the pointcloud For any two points P and Q in PC with a geodesic which crosses the hole h in G_p, the calculated geodesic path d between P and Q will contain a part on the boundary of the

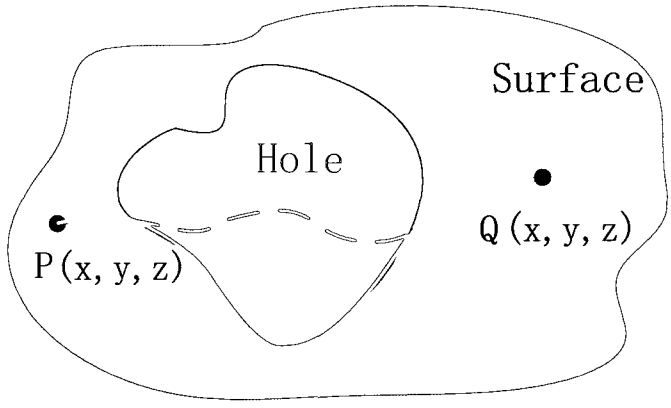


Figure 4.5: Geodesic on surface with hole

hole. Therefore, if the calculated shortest path contains vertices on the hole boundary, it is an indication that the calculated shortest path may be in error. Most holes are frequently changing location and size during object motion. The effect of hole boundary points on the geodesic distance calculation will be estimated in the following discussion.

Considering the use of the geodesics in isometry-based matching, we need an indication of the accuracy of the approximation by shortest path in G_p. Here the accuracy of the geometric distance calculation is estimated by how well it preserves the isometric property during motion. In our motion detection framework, the geodesic distance is calculated on a reconstruction surface which approximates the real object surface. There is no way to guarantee that the estimated shortest geometric path of two surface points P and Q will have exactly the same geodesic distance on the reconstruction surface during the deformation of real surface, even if the real surface deforms isometrically. But, the difference in the estimated geodesic distances between P and Q during a isometric deformation can still be a meaningful indication of the accuracy of the approximation. Let the distances d^1 and d^2 be the shortest path distances between the two points P and Q at time t^1 and t^2, then the absolute difference $|d^1 - d^2|$ reflects how well the isometry constraint has been preserved during the motion between times t^1 and t^2.

In the following experiments, the influence of boundary vertices in G_p on the stability of the geodesic approximation will be estimated in three examples: fronto-parallel rotation of paper, bending of paper and plush dinosaur. And the relationship between path length and error in geodesic distance calculation will also be studied.

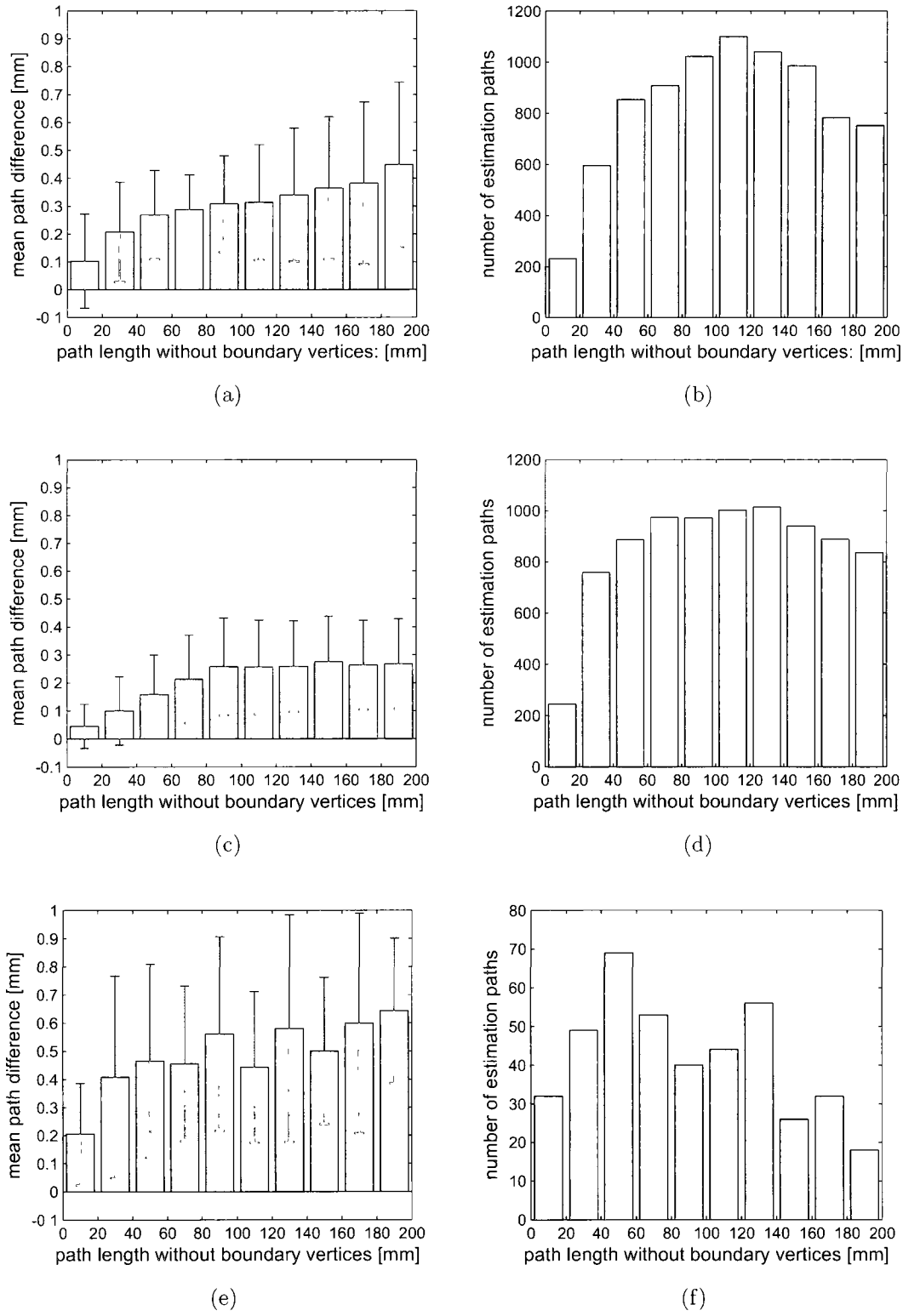


Figure 4.6: Influence of path length on geodesic approximation error. From top to botton: Approximate fronto-parallel rotation of paper (a, b), bending of paper (c, d), and plush dinosaur (e, f).

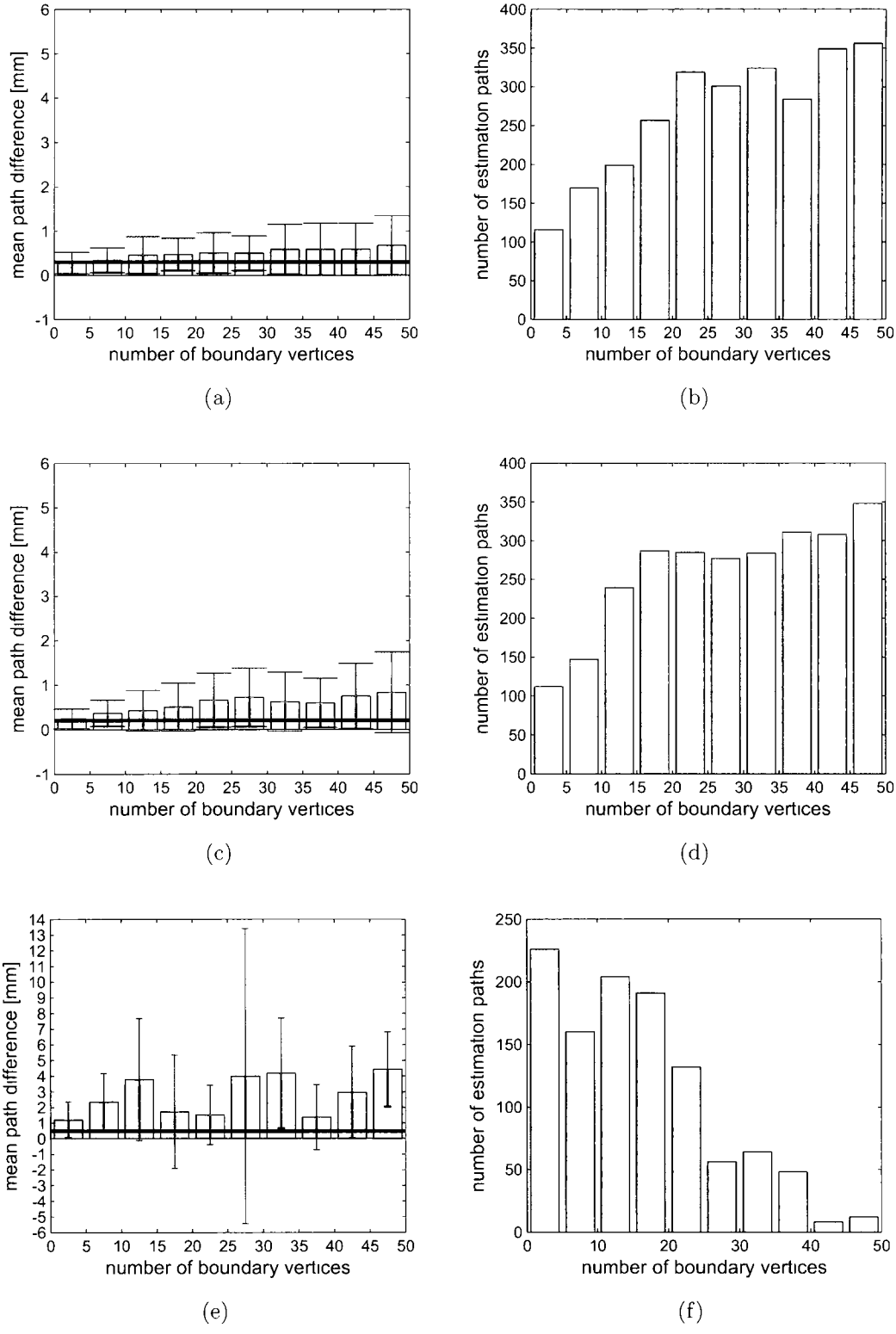


Figure 4.7: Influence of boundary vertice on geodesic approximation error. From top to botton: Approximate fronto-parallel rotation of paper (a, b), bending of paper (c, d), and plush dinosaur (e, f).

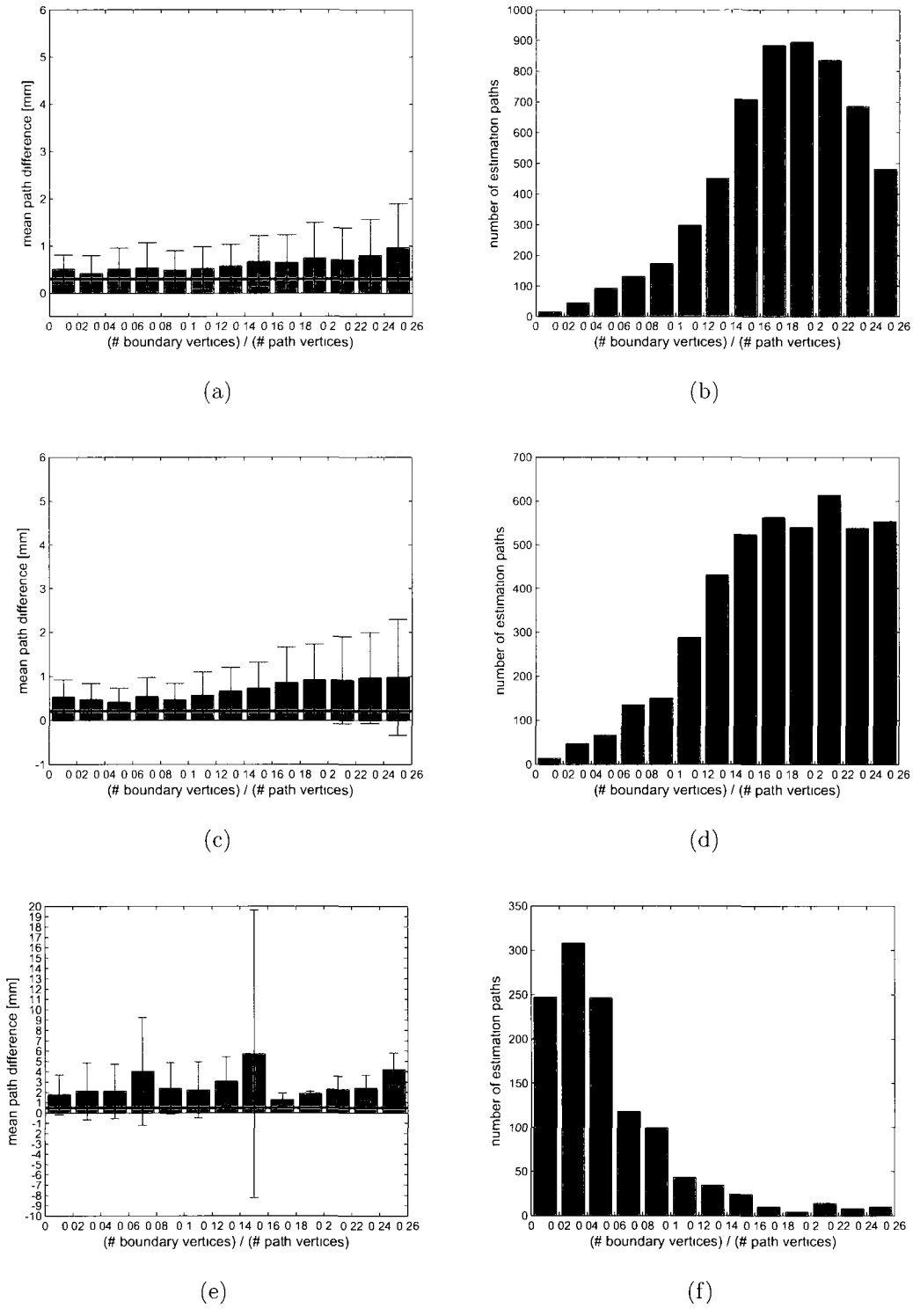


Figure 4.8: Influence of boundary vertice ratio on geodesic approximation error. From top to botton: Approximate fronto-parallel rotation of paper (a, b), bending of paper (c, d), and plush dinosaur (e, f).

In these experiments, the calculation error of the geodesic distance between two pre-matched points P and Q is measured as $|d^1 - d^2|$ through pixel graphs ${G_p}^1$ and ${G_p}^2$. For each pair of pre-matched points P and Q, there will be an estimation record $[(P,Q), e, d, h]$. e means the error of the geodesic distance calculation between P and Q. d is the average length of the shortest paths between P and Q through ${G_p}^1$ and ${G_p}^2$. h mean the total numbers of boundary vertices contained in shortest paths between P and Q both on ${G_p}^1$ and ${G_p}^2$. And in the experiments, all the pre-matched points are generated from KLT features. To estimation the relationship between path length and the error in geodesic distance calculation, we apply all the estimation records which have zero h. The average errors of geodesic distance calculation related to different range of geodesic length are given in the Figure 4.6. For boundary vertice estimation, only the path with boundary vertice will be applied. The relationship between the absolute number of boundary vertice and geodesic approximation are given in Figure 4.7. For boundary ratio estimation, the result is given in Figure 4.8.

Here, Figure 4.6 shows that shorter paths are more stable. This is true if the underlying motion is close to rigid, isometric or a more general small motion. However, the key observation is that all variations are in the sub-millimeter range. This approve the assumption that if the pixel graph is generated from a dense sampling of a object surface, the shortest path approximation of the geodesic distance is trustable and hence, we conclude that for our application the graph-based approximation of geodesics is sufficient. It is desirable to have small error in the geodesic distance calculation and as many geodesic distances as possible in the isometry-based matching. Therefore, all the approximation geodesic distance without the hole will be applied in our isometry-based matching.

Figure 4.7 and Figure 4.8 show that boundary vertices increase the variation in the geodesic approximation as expected. The variation increases with the number of boundary vertices but for the general small motion the trend is not so clear. In general, the variation is much higher than the average variation of paths without holes (horizontal line in Figure 4.7 and Figure 4.8), even if only a few vertices are on the boundary of a hole. This remains true even if we look at the ratio between boundary vertices and total number of vertices on the shortest path, as shown in Figure 4.8. Based on this study, we conclude that our isometry-based matching should not rely on geodesic approximations which are based on shortest paths with boundary vertices.

4.4 Isometry-Based Motion Detection

4.4.1 Isometry-Based Matching

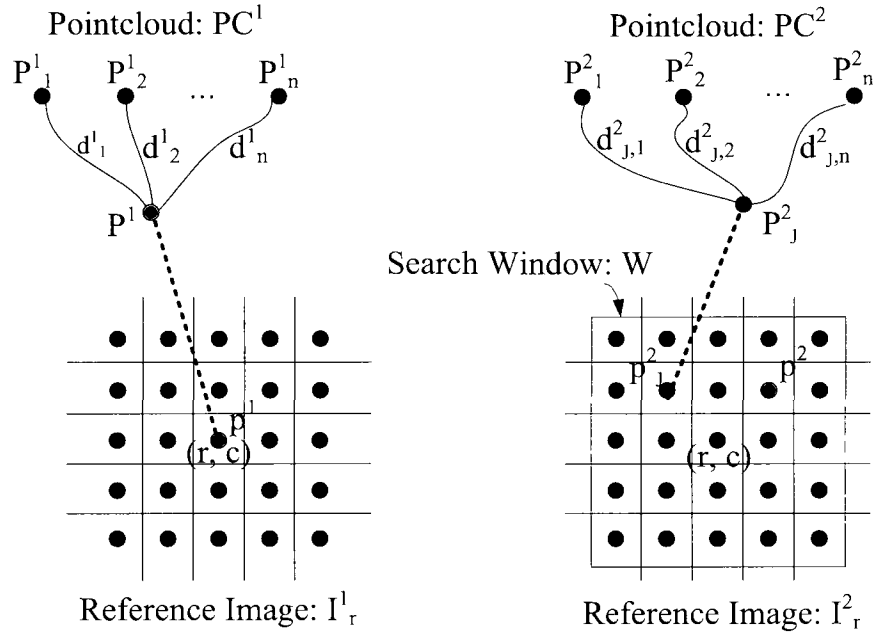


Figure 4.9: Isometry-based matching

In isometry-based matching, we match 3D points over time based on a local search over an image window. The matching criteria is based on the minimum difference in the shortest path to reference points over time. Our isometry-based matching is depicted in Figure 4.9. The 3D point P^1 is an unmatched point in the pointcloud PC^1 at time t^1 and the pixel p^1 is its projection into the reference image ${I_r}^1$. The pixel p^1 has a shortest path of length ${d^1}_1, {d^1}_2, \ldots, {d^1}_n$ from the reference points ${P^1}_1, {P^1}_2, \ldots, {P^1}_n$, respectively. The reference points are KLT features with depth values and hence their position ${P^2}_1, {P^2}_2, \ldots, {P^2}_n$ at t^2 are known. We can find the shortest path ${d^2}_{\jmath,\imath}$ between any 3D point ${P^2}_\jmath$ in PC^2 and the reference point ${P^2}_\imath$ at t^2 based on ${G_p}^2$. We search for a point P^2 which has the same distance to the reference points at t^2 than point P^1 had at t^1. We minimize the function

$$error_{Match}(P^1, {P^2}_\jmath) = \sum_{\imath=0}^{n} ({d^1}_\imath - {d^2}_{\jmath,\imath})/n. \tag{4.1}$$

and limit our search to a window W of $k \times k$ pixels centered at p^1 in ${I_r}^2$. In all our experiments $k = 5$ is chosen. The pixel with the lowest matching error in W will be selected as the corresponding pixel p^2 of pixel p^1.

Also, we refine this matching strategy in two ways. One, we add a maximum path length difference d_{max} in the calculation of Equation 4.1. Two, we require a minimum number n_{min} of shortest paths to the same respective feature points at t_1 and t_2. Although $n_{min} = 3$ is theoretically sufficient, we can expect that a larger number will increase robustness to errors in the approximation of the geodesics.

In the following experiment, we track KLT feature points between the two pointclouds PC^1 and PC^2. We test the effects of the thresholds by comparing the motion calculated with KLT and isometry-based matching. For each feature point in PC^1, we know its motion vector (x, y, z) from KLT tracking and we estimate its motion vector (x', y', z') with isometry-based matching with the 49 other feature points. Then, we can define a motion error

$$error_{Motion} = \sqrt{|x - x'|^2 + |y - y'|^2 + |z - z'|^2} \tag{4.2}$$

The motion error is reported by using different thresholds d_{max} and n_{min} during a general small deformation (50 KLT features), rotation of paper (100 KLT features) and bending of paper (100 KLT features) in Figure 4.10. A small value of d_{max} leads to a small motion error because points are only matched if the isometry constraints have a a small error. But the drawback of a small threshold d_{max} is a small number of matched points. It is also sensitive to errors since the relative weight $\frac{1}{n}$ of each path length difference is large. A small d_{max} is also undesirable since it does not use the isometry constraint to rule out matches. Large differences in shortest path over time indicate that a match is incorrect but a small threshold d_{max} disregards these path length differences. Figure 4.10 shows that a larger value of d_{max} can lead to a small motion error with large matching number of points. The worst setting of d_{max} is to a medium value because at this setting geodesic with some approximation error are included but the evidence of large path difference to rule out matches is still excluded. Based on our experiments, we set a large threshold $d_{max} = 8mm$ and require a minimum of four paths $n_{min} = 4$.

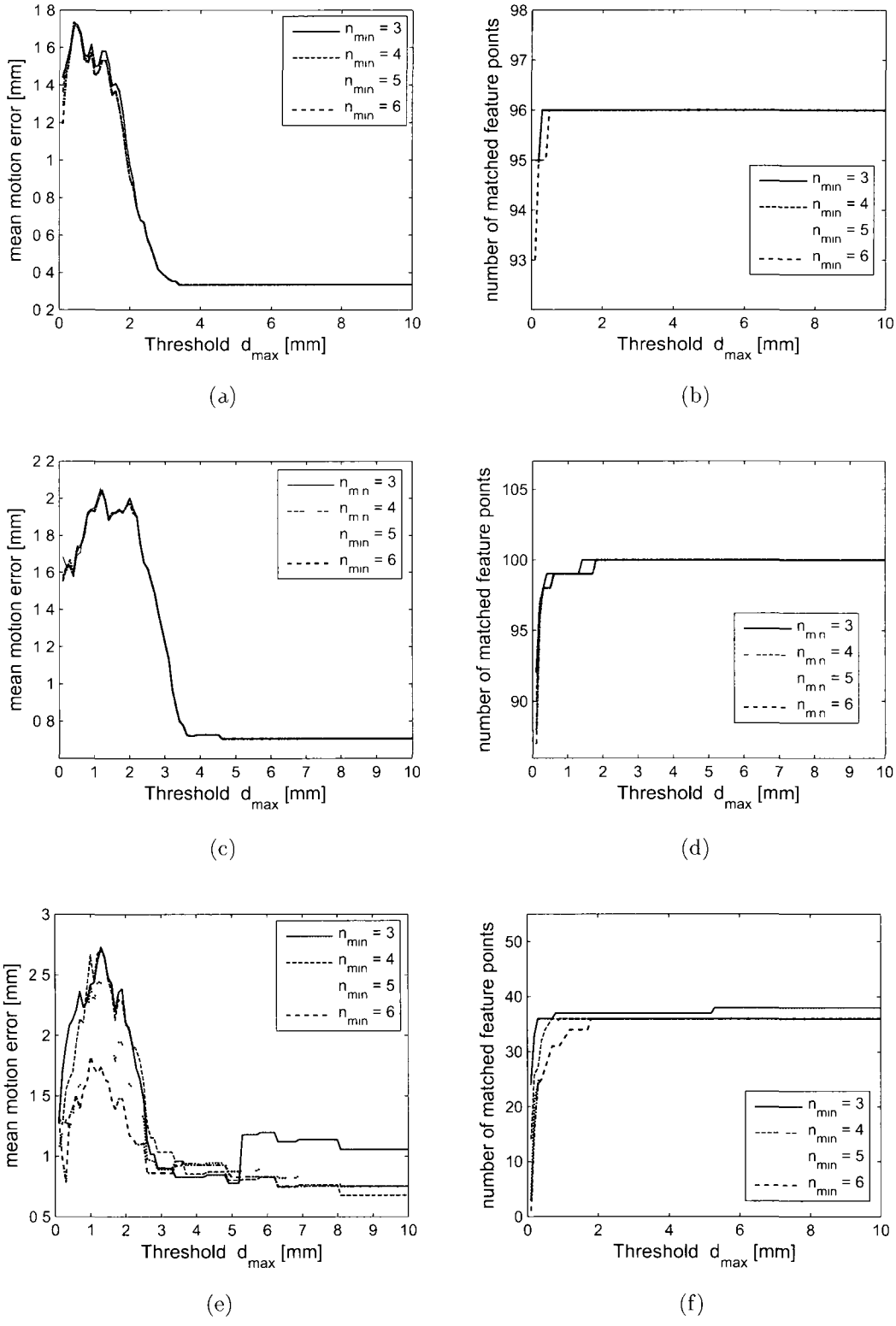


Figure 4.10: Influence of threshold d_{max} (maximum path length difference) and n_{min} (minimum number of shortest paths over time) on motion error. From top to botton Approximate fronto-parallel rotation of paper (a, b), bending of paper (c, d), and plush dinosaur(e, f)

4.4.2 Additional Reference Points

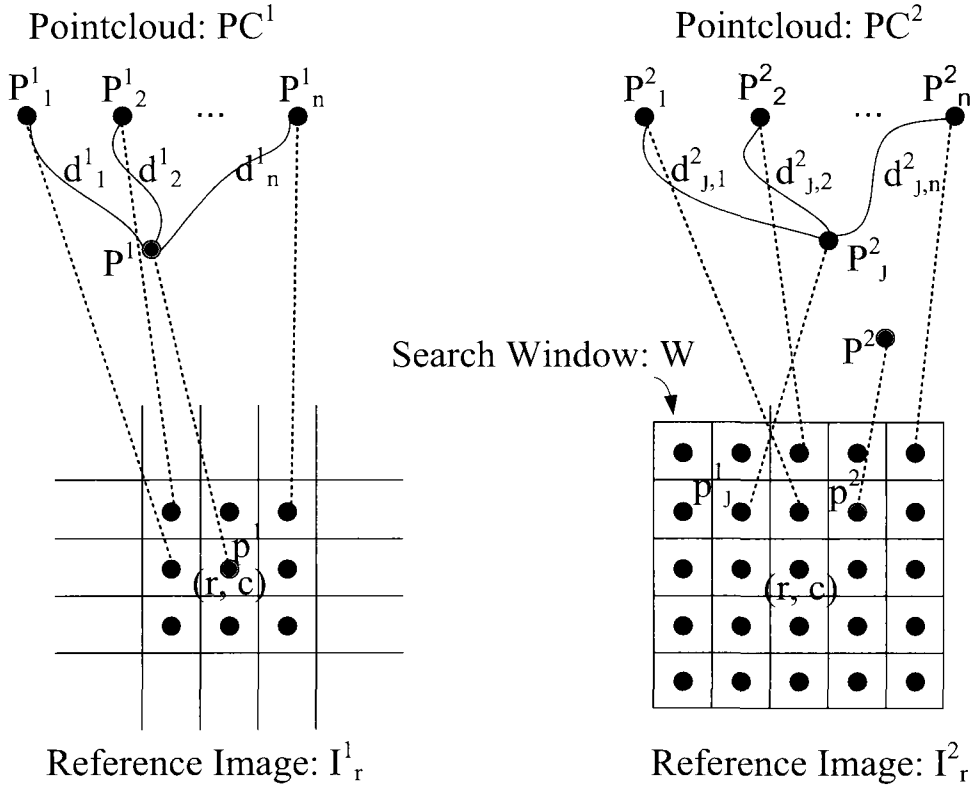


Figure 4.11: Isometry-based matching with additional reference points

In most cases, the reference points generated by feature tracking are sparse but additional reference points are desirable for isometry-based matching. Therefore, we use all the points matched in the first phase of isometry-based matching as additional potential reference points in the second phase. The second phase matches points which could not be matched by using reference points from feature tracking in phase one. Because we have a fairly dense set of matched points after isometry-based matching in phase one, we restrict the reference points for phase two to the 8-neighbours of pixel p^1 corresponding to the unmatched point P^1. Then, we only need to calculate the shortest path to the 8 neighbours and not to the whole G_p. Figure 4.11 shows that to find the matching point P^2 in PC^2 for P^1, a search window W centered at the pixel position of p^1 is used. Each point $P^1{}_j$ corresponding to pixel p^1_j in W will be evaluated for the isometric matching error with point P^1. The matching pixel is found as the pixel with minimum matching error. We iterate this procedure by adding newly matched points as potential reference

points until there is no more unmatched point in PC^1, or there is no new reference point added in the last iteration.

4.5 Results of Motion Detection

Here, we report results for two sequences: bending of a paper over 14 frames and deformation of a plush toy dinosaur over 8 frames. Calculating each 3D flow between two consecutive models, it costs about 20 minutes (CPU: Intel Q6660, Memory: 4GB). Both sequences were captured with a colour Point Grey Research Bumblebee2 camera at 24 fps at a resolution of 640×480 and the deforming object was covering a large portion of the images (see Figures 4.12 and 4.13 for the reference images). The object pointcloud was calculated with the window-based correlation method available in the Point Grey Research Triclops library with a window size of 11×11 and points filtered with "back-and-forth" and the "surface" method. The resulting pointclouds for the first and last frame of each sequence are shown in Figures 4.12 and Figure 4.13, respectively.

The 3D flow vectors calculated with our window-based isometric matching method over the whole sequence are shown in Figure 4.12 and Figure 4.13. The magnitude of the flow vectors is displayed with the Matlab jet colouring scheme (blue is smallest, red is largest). The direction of the motion vectors is shown with samples of the flow vectors (10% of the vectors, scaled by a factor of 4). We also show PC^1 transformed to PC^n with the help of the flow vectors where $n = 14$ and $n = 8$ for the paper and dinosaur sequences, respectively. The results show that our motion detection method produces a dense set of flow vectors during non-rigid deformation of objects. The paper deformation fits the assumptions of isometry in our method exactly. Also, dense stereo pointclouds are generated due to the pseudo-random texture. The dinosaur deforms non-isometrically because the cloth of the plush toy stretches but the interframe motion is small. The wrinkles in the surface and repetitive pattern cause holes in the pointclouds but despite these challenges reasonable results are produced by our method.

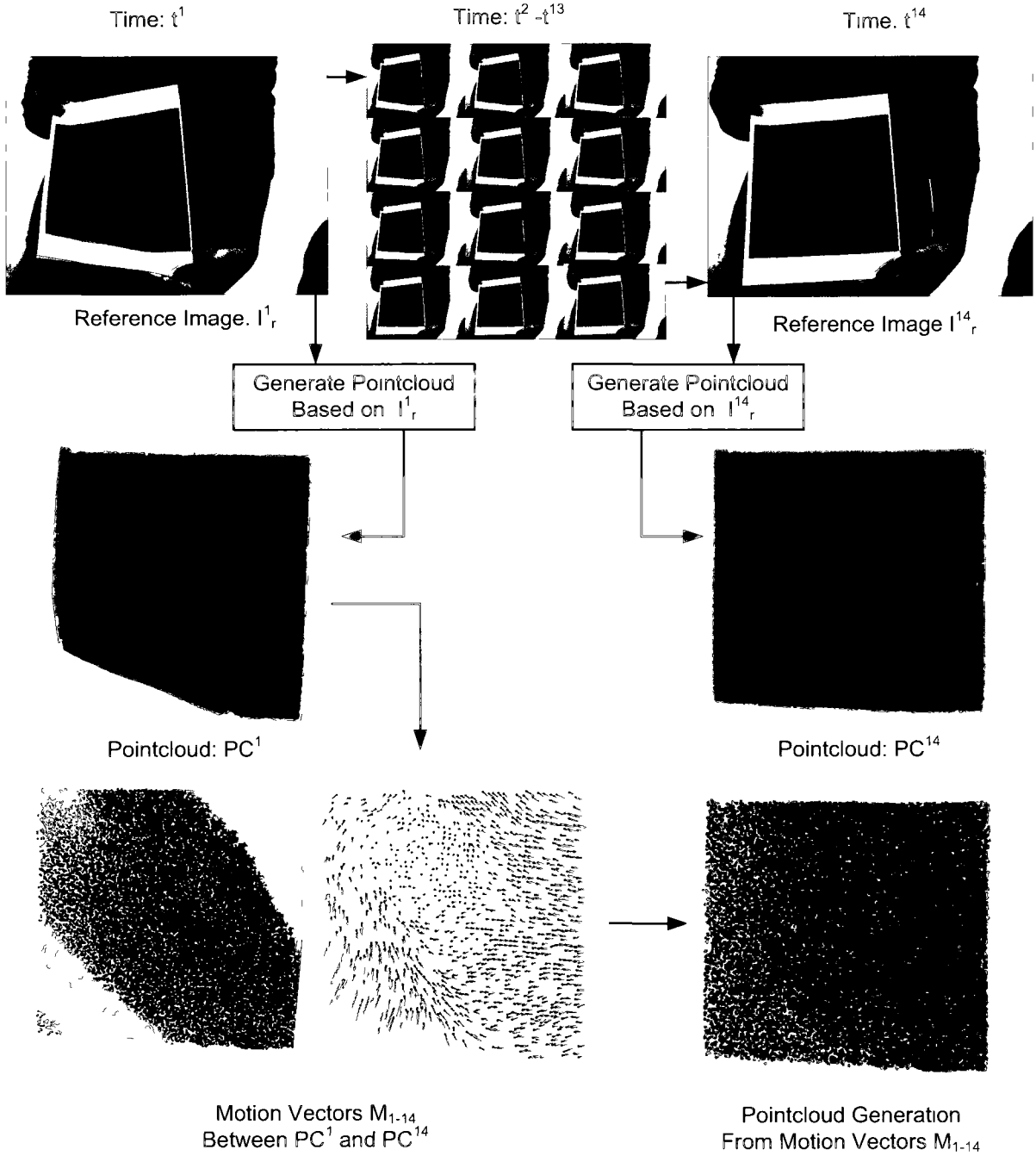


Figure 4.12: Paper deformation

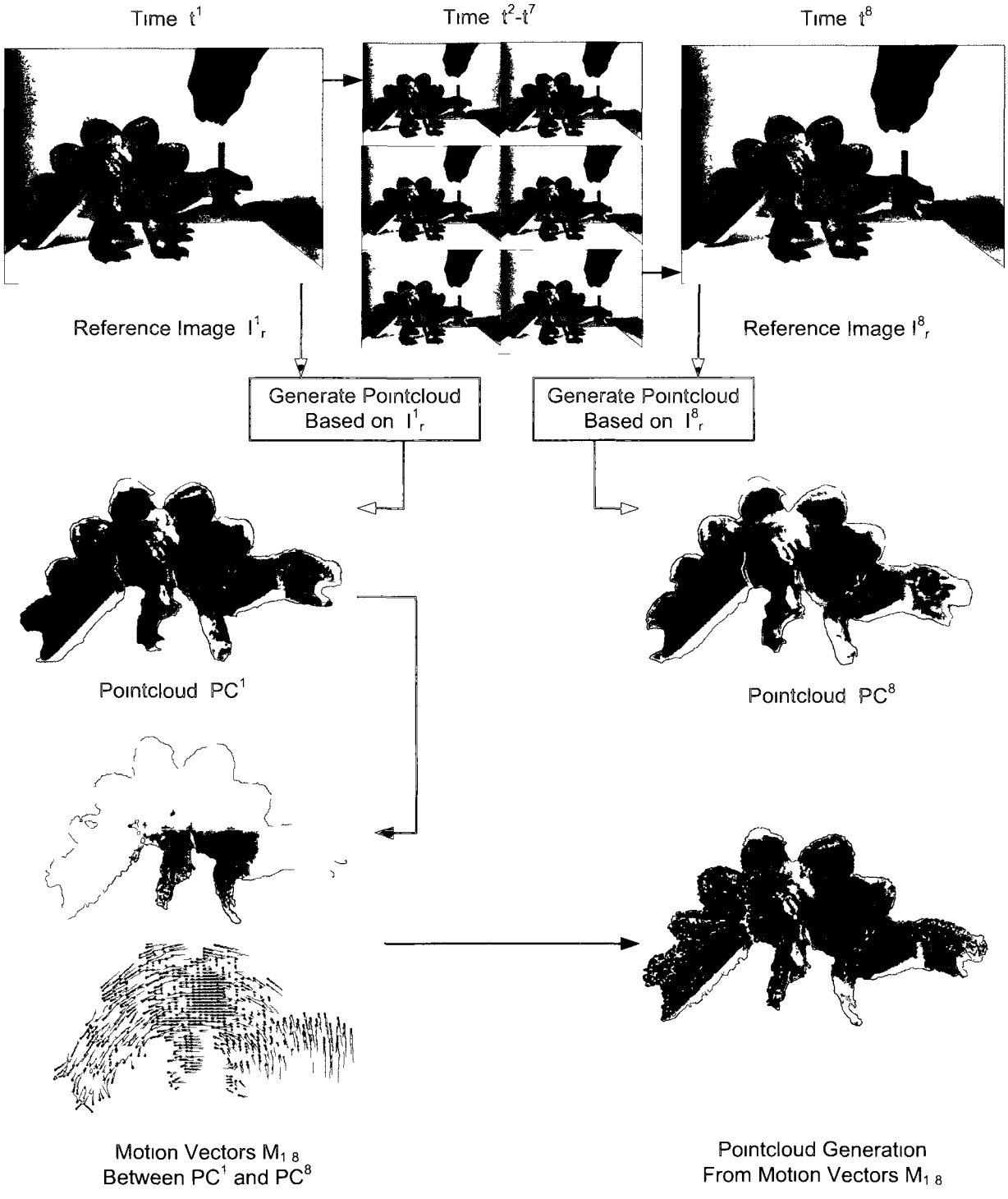
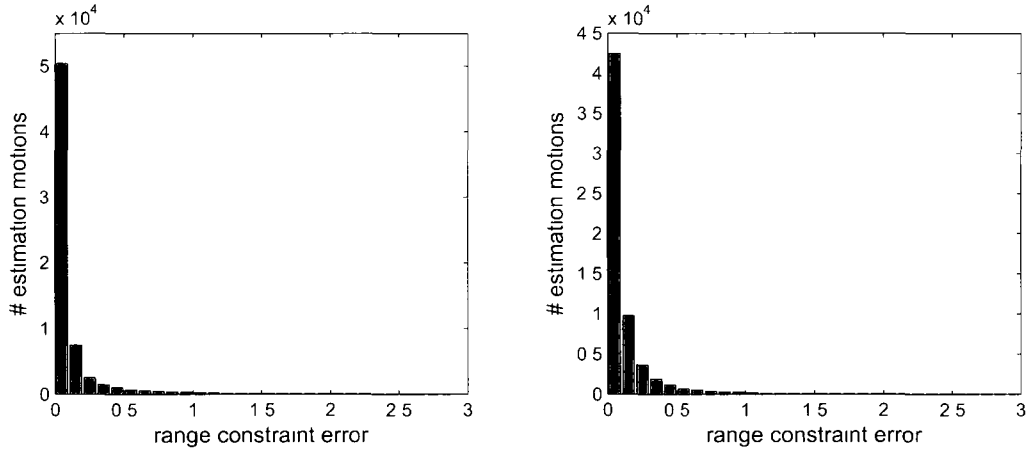


Figure 4 13 Dinosaur deformation

In order to give a more quantitative evaluation of our results, we add a filtering step to our method based on the range flow motion constraint Equation 2.16). We calculate the absolute error in the constraint equation for each flow vector with direct neighbours in the pointclouds. Because we use the range flow motion constraint only as a verification step, we use a simple central difference approximation to the spatial derivatives of the depth and a forward difference for the time derivative. The number of motions based on different maximum absolute range flow error is given in Figure 4.14. The mean absolute error in the constraint equation is $0.45mm$ and $0.59mm$ for the paper and dinosaur, respectively. This result can also be used as a filtering step, rejecting flow vectors which are in violation of the constraint.



(a) Paper (b) Dinosaur

Figure 4.14: Motion error estimation with range flow motion constraint

Chapter 5

Conclusions and Future Work

5.1 Summary

In this thesis, a 3D reconstruction framework is presented by applying the multi-view images taken from the ProFUSION25 camera array in Chapter 3. To calculate accurate projections from 3D points, the internal and external parameters of the ProFUSION25 camera are calibrated. Vignetting and distortion removal are performed to refine the raw images captured with the ProFUSION25. The background in the images is detected to increase the computational efficiency and apply the visual hull constraint. In the MVS reconstruction, multiple MVS pointclouds of the same object are generated by applying the same set of multi-view images but different reference views. A new combined matching method is applied to find projection pixels of the same 3D point through multi-view images. Two new fusion strategies are provided to generate refined points from multiple MVS pointclouds. In Chapter 4, an isometry-based motion detection technique is introduced to successfully obtain a dense set of motion vectors over the deforming surface of an object. Between two consecutive pointclouds, the matching feature points are detected by the KLT matching method. For computational efficiency, an approximate geodesic distance is applied instead of computing the exact geodesic distance. The definition and error estimation of the approximate geodesic distance is discussed. Finally, the motion vectors are estimated by using the isometry-based matching constraint.

5.2 Conclusions

In 3D reconstruction, the original contribution of this thesis is a novel framework of using multi-view images taken from the ProFUSION25 camera array. Multi-view images can be easily obtained without external synchronization by using the ProFUSION25. And the ProFUSION25 also provides the opportunity to study the 3D reconstruction based on low-quality, non-calibrated cameras similar to consumer cameras. To calibrate the ProFUSION25, the internal and external parameters are estimated by minimizing the overall error between predicted and observed image points. The external parameters are optimized by using the sparse bundle adjustment (SBA) to generate uniform internal parameters. To overcome the poor the camera lens quality, a vignetting removal kernel is generated by using a prototype vignetting function [20]. And the image distortion is detected under Brown's distortion model [11, 21]. The overall hybrid matching method is another contribution of our work. It combines the measurements of intensity difference and intensity distribution difference in the matching. As shown in the experiments of Section 3.4 for our data set, the new combined matching method leads to better reconstruction results compared to the matching methods purely based on SAD and NCC, alone. To achieve better reconstruction results, there are two new fusion strategies introduced in the fusion step. A refined pointcloud is estimated by fusing multiple MVS pointclouds generated from the same set of multi-view images and different reference views. The low-and-high confidence fusion strategy is based on combining both high and low confidence candidate points to indicate the real surface. The density-based fusion strategy is based on measuring the real surface area by applying the density of high confidence candidate points. Thanks to all the pre-processing and techniques in our reconstruction framework, an object model can be reconstructed by using twenty-five black and white images as shown in Section 3.4.

For motion detection, a novel window-based matching technique for 3D flow based on isometric surface deformation is proposed. The matching proceeds iteratively and is seeded with matched features in the intensity images. After the feature points have been matched, the intensity images are no longer used which gives robustness to emittance changes of the surface over time. To deal with the compute-intensity problem of the exact geodesic distance calculation, an approximate geodesic distance is used for computational efficiency. The approximate geodesic distance is calculated as the shortest path length based on the pixel graph generated from a pointcloud. The boundary points and path length are measured to discover their influence related to the accuracy of the geodesic

distance approximation. Based on the experiments of Section 4.3.2, the shortest path length doesn't show any significant impact on the accuracy of the geodesic distance approximation. However, shortest path with boundary vertices leads to significant geodesic approximation error. Therefore, all shortest paths will be used in our isometry-based matching except shortest paths with boundary vertices. To generate as many highly reliable matches as possible, the threshold of maximum path length difference and a minimum number of matched shortest paths are used to verify each matched point pair. Taking the benefit from our Euclidean distance embedded pixel graph, 3D points can be matched by matching their projecting pixel in the image domain. To overcome the lack of reference points, all the points matched in the first phase of isometry-based matching will be treated as additional reference points in the second phase. In Section 4.5, the motion vectors generated from our window-based isometric matching method is verified by using the range flow motion constraint equation. The experimental results show that our motion detection method is tolerant to the topological noise from the stereo range data and generates dense motion vectors over the object surface.

5.3 Future Work

In future work, the accuracy of the calibration result can be improved. Right now, the applied internal and external parameters of the ProFUSION25 will lead to at most 0.71 average pixel error in the estimated epipolar lines. The error of the calibration result causes inaccurate projection calculation. And the inaccurate projection calculation will lead to the low-quality reconstruction results. The calibration result can be improved in two steps. The first step is to improve the condition of the calibration data. A more precise calibration grid should be used. In practice, it should be consistently flat, so that the ground truth of the grid image is more precisely known during the camera calibration. To achieve more general optimized internal and external parameters, larger numbers of calibration images from different poses of the grid image should be used. In the second step, a idea similar to the approach of Furukawa and Ponce [23] could be used and adjusting the camera calibration with the result of the 3D reconstruction procedure. The initial 3D reconstruction can be generated with the initial camera parameters form multi-view images. Then, a set of 2D features can be matched through the multi-view images. A set of 3D features and their projection 2D features can be provided by using the mapping between the initial reconstruction and its relative images. Finally, the bundle adjustment can be used to optimize both the structure of the reconstruction model and

the camera parameters by minimizing the projection error.

The possible applications of the matching method are many. In this thesis, our hybrid matching method is applied to match the corresponding pixel through multi-view images in the MVS reconstruction. It also can be used in background segmentation, feature point matching, optical flow and 3D flow estimation. But the performance of the new hybrid matching method should be measured in different applications. A truly probability-based hybrid model should be designed by using similar ideas to the overall hybrid matching model. And the combination concept also can be used to generate a hybrid fusion strategy by combined surface confidence calculated from low-and-high confidence fusion strategy and density-based fusion strategy. In this thesis, our hybrid matching method is based on the assumption that the camera view directions are nearly the same and the object surface is nearly co-planar locally. Therefore, a matching window in one view will map into a matching window with the same size in another view. In the case that the object surface is non-coplanar or the view directions are varying, the matching window is distorted as seen from different views. To let the hybrid matching method overcome this drawback, a scaled window technique should be applied to remove the window distortions.

In this thesis, the range flow motion constraint equation is only applied in the final verification of the isometry-based matches. But the range flow motion constraint equation can also be applied to rule out the invalid matching candidate in the isometry-based matching step. Right now, the 3D flow is only detected on a single view since our geodesic approximation method relies on the neighbourhood information from the raster image. But the isometry-based motion detection can be extended to estimate the scene flow by applying other methods to find neighbourhoods (e.g., k-nearest neighbours). The computational efficiency of the 3D motion detection application is low. Real-time motion detection should be considered by applying more efficient methods of calculating the approximate geodesics (e.g., with the Fast Marching Method [34]) and converting the CPU program into a GPU program in the future. In isometry-based matching, corresponding 3D points are matched by seeking the similarity of two 3D points which have the same geodesic distances to reference points. It uses a similar idea as the intensity-based matching which seeks similar pixels that have the same intensity value. Therefore, the geodesic distance can be treated as pseudo-intensity for each 3D point. Taking the benefit of our window-based isometric matching, the pseudo-intensity can be assigned to projection pixels. Based on the geodesic distance embedding pseudo-intensity images, any intensity-based algorithm or technique can be combined with the isometry constraint

in the motion detection. To minimize the reconstruction noise, the 3D model applied in the motion detection phase is generated from the commercial stereo camera em Bumblebee2. A complete framework of 3D reconstruction and motion detection should be considered.

Appendix A

Appendix A: Camera Calibration Estimation

A.1 Estimation of Camera Parameters

The goal of the camera calibration procedure is to find the internal and external parameters of the ProFUSION25. The ProFUSION25 is a camera array system which has 25 VGA resolution cameras. For one shot of the ProFUSION25, twenty-five images can be captured for the same object from twenty-five different views. Higher accuracy in the camera calibration will lead to the better quality of the 3D model generated from the images captured by the ProFUSION25.

In our approach, we use the Camera Calibration Toolbox to estimate the internal and the initial external parameters of the ProFUSION25. The refined external parameters of the ProFUSION25 are estimated by the SBA of Lourakis and Argyros [44]. The opitmized external parameters are calculated by using fixed internal parameters and by minimizing the projection error of $35,100$ 3D points. The internal parameters include focal length, principal point and distortion parameters. For distortion parameters, two coefficients of radial distortion (K_1, K_2) and two coefficients of tangential distortion (K_3, K_4) are used. The calibration result for internal parameters is shown in Table A.1. For the refined external parameters estimation, the central camera 12 is defined as the left camera (or reference camera). All other cameras in the ProFUSION25 are defined as the right cameras. For each 3D point P, its 3D position is given in the reference camera coordinate system. And its two coordinate positions X_L and X_R in the left and right camera coordinate system are related through a rigid motion transformation as shown

in Equation A.1.

$$X_R = RX_L + t \tag{A.1}$$

The rotation matrix R and translation vector t are defined as the external parameters between the right and left cameras. The optimized external parameters between camera 12 and other cameras are given in Table A.2.

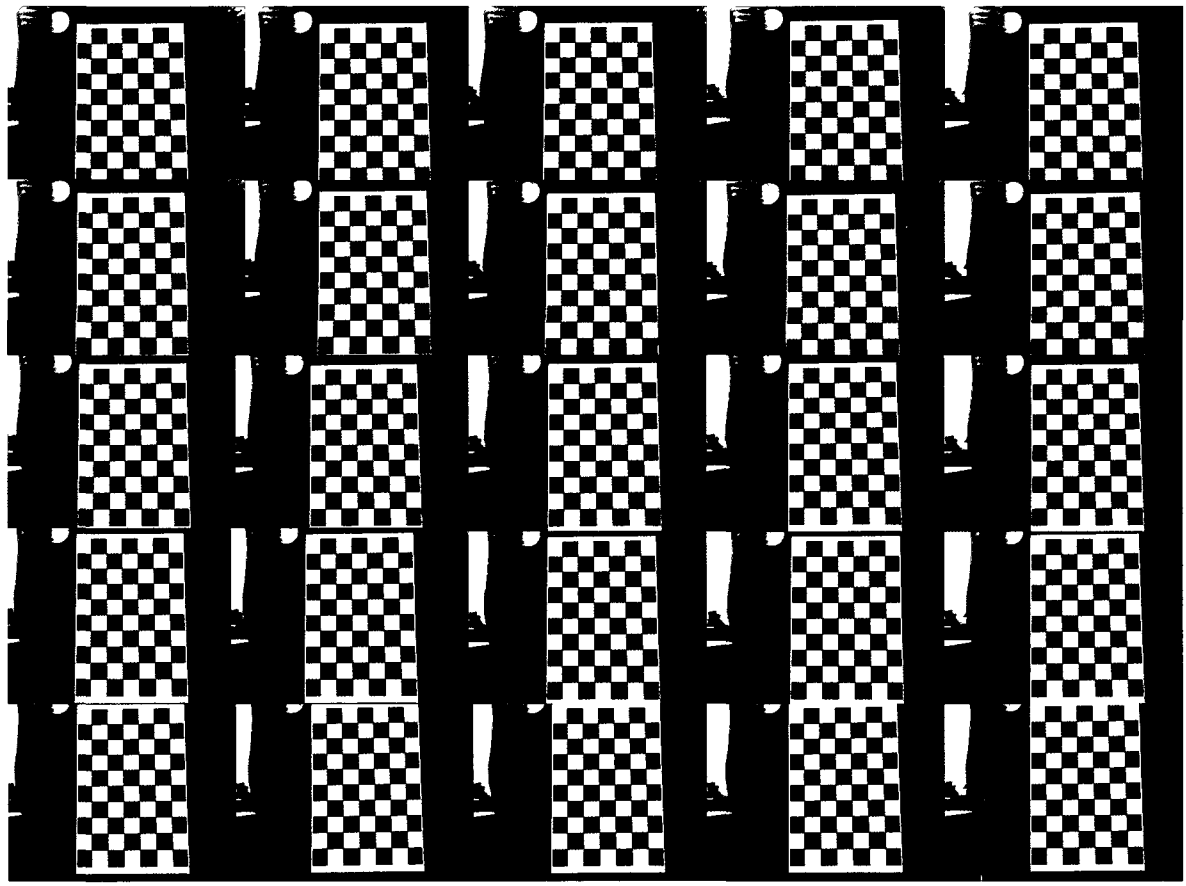


Figure A.1: Calibration images. The twenty-five images are taken from the ProFUSION25 for the same pose of the calibration board.

Camera number	Focal length	Principal Point	Distortion Parameters
Camera 0	901.371, 902.932	319.479, 239.752	0.0838 0.0352 0.0040 0.0038
Camera 1	900.450, 902.285	324.552, 237.762	0.0653 0.1708 -0.0074 -0.0058
Camera 2	900.362, 901.781	314.617, 238.108	0.0982 -0.2968 0.0045 -0.0087
Camera 3	900.918, 902.326	323.365, 229.032	0.1051 -0.1857 -0.0058 -0.0023
Camera 4	902.241, 904.305	312.865, 238.461	0.0942 -0.2217 -0.0072 0.0072
Camera 5	902.773, 904.105	324.827, 238.013	0.0848 -0.1917 -0.0036 -0.0069
Camera 6	904.555, 906.191	319.423, 231.425	0.0957 -0.2022 0.0066 -0.0046
Camera 7	904.467, 905.891	322.379, 238.839	0.0911 -0.0109 -0.0011 0.0047
Camera 8	901.058, 902.625	313.443, 240.174	0.0898 -0.1143 -0.0087 0.0038
Camera 9	905.724, 907.250	318.882, 240.175	0.0738 0.0213 -0.0055 0.0026
Camera 10	903.114, 904.088	328.965, 241.399	0.0982 -0.1311 0.0039 0.0041
Camera 11	894.472, 895.630	328.044, 242.943	0.1484 -0.5846 0.0081 0.0009
Camera 12	905.900, 906.716	327.505, 244.423	0.0856 -0.1095 -0.0052 -0.0040
Camera 13	897.559, 898.805	318.755, 238.986	0.1037 -0.4051 0.0060 0.0052
Camera 14	894.435, 895.768	316.670, 241.710	0.1289 -0.3452 0.0003 -0.0075
Camera 15	903.021, 904.380	319.103, 247.909	0.0887 -0.1124 0.0095 -0.0004
Camera 16	903.222, 904.622	322.199, 247.929	0.0780 -0.0292 0.0018 0.0082
Camera 17	904.340, 905.441	321.472, 249.761	0.1017 -0.2370 0.0012 0.0042
Camera 18	904.889, 906.109	324.029, 249.691	0.0969 -0.1858 -0.0048 -0.0009
Camera 19	902.949, 904.166	313.839, 242.971	0.0962 -0.2468 0.0044 -0.0016
Camera 20	903.037, 904.081	316.106, 244.481	0.0920 -0.0637 0.0068 -0.0020
Camera 21	899.240, 900.592	325.800, 242.731	0.1161 -0.3853 0.0011 -0.0095
Camera 22	902.637, 903.846	319.766, 248.662	0.1064 -0.2832 0.0042 -0.0084
Camera 23	904.967, 905.927	327.789, 244.777	0.1074 -0.3445 0.0010 0.0049
Camera 24	904.904, 905.751	326.318, 242.918	0.1100 -0.3906 -0.0071 -0.0027

Table A.1: ProFUSION25 internal parameters

Camera number	Rotation Quaternion	Translation Vector
Camera 0	1.0000, -0.0003, 0.0004, 0.0001	-23.9777, 23.9389, -0.4973
Camera 1	0.9999, -0.0019, -0.0012, 0.0002	-11.8761, 23.8868, 0.1101
Camera 2	0.9999, -0.0008, 0.0001, 0.0018	0.0281, 23.7153, -2.6756
Camera 3	0.9999, -0.0025, -0.0003, 0.0006	12.0461, 23.7844, -0.7974
Camera 4	0.9999, -0.0018, -0.0005, -0.0006	24.1264, 23.7071, -0.1519
Camera 5	0.9999, -0.0031, -0.0005, 0.0001	-23.9307, 12.0099, -0.1884
Camera 6	0.9999, -0.0025, 0.0013, -0.0001	-11.9003, 11.9286, -1.2867
Camera 7	0.9999, -0.0022, -0.0010, 0.0024	0.0123, 11.8778, 0.6816
Camera 8	0.9999, -0.0031, 0.0001, 0.0000	12.0385, 11.8835, 0.0050
Camera 9	0.9999, -0.0018, -0.0003, -0.0011	23.9915, 11.7181, -0.7974
Camera 10	0.9999, -0.0005, -0.0007, 0.0010	-23.9866, -0.0405, 0.2005
Camera 11	0.9999, -0.0001, -0.0019, -0.0010	-12.0848, -0.0370, -0.1528
Camera 12	1.0000, 0.0000, -0.0001, -0.0000	0.0046, 0.0044, -0.0090
Camera 13	0.9999, -0.0002, 0.0003, -0.0011	11.8178, -0.1111, -1.4202
Camera 14	0.9999, -0.0010, 0.0017, -0.0004	23.9260, -0.1316, -0.3353
Camera 15	0.9999, 0.0008, 0.0006, 0.0011	-23.9353, -12.0304, 0.0174
Camera 16	1.0000, 0.0003, 0.0004, 0.0006	-12.0334, -11.9464, 0.4609
Camera 17	0.9999, 0.0012, 0.0007, 0.0022	0.0946, -12.0245, 0.6559
Camera 18	0.9999, 0.0016, 0.0012, 0.0035	12.1469, -11.9880, 0.1850
Camera 19	0.9999, 0.0014, 0.0023, -0.0002	23.9732, -12.1373, -0.5939
Camera 20	0.9999, 0.0000, 0.0029, -0.0003	-23.9409, -23.8663, 0.2367
Camera 21	0.9999, 0.0004, 0.0009, -0.0021	-12.1023, -23.8913, 0.0651
Camera 22	0.9999, 0.0025, 0.0016, 0.0005	0.1085, -24.0189 0.4827
Camera 23	0.9999, 0.0023, 0.0014, -0.0011	11.7605, -24.0232 -0.8830
Camera 24	0.9999, 0.0022, -0.0005, -0.0012	23.9031, -24.2145 0.1851

Table A.2: ProFUSION25 external parameters

A.2 Calibration Error Measurement

A.2.1 Calibration Error

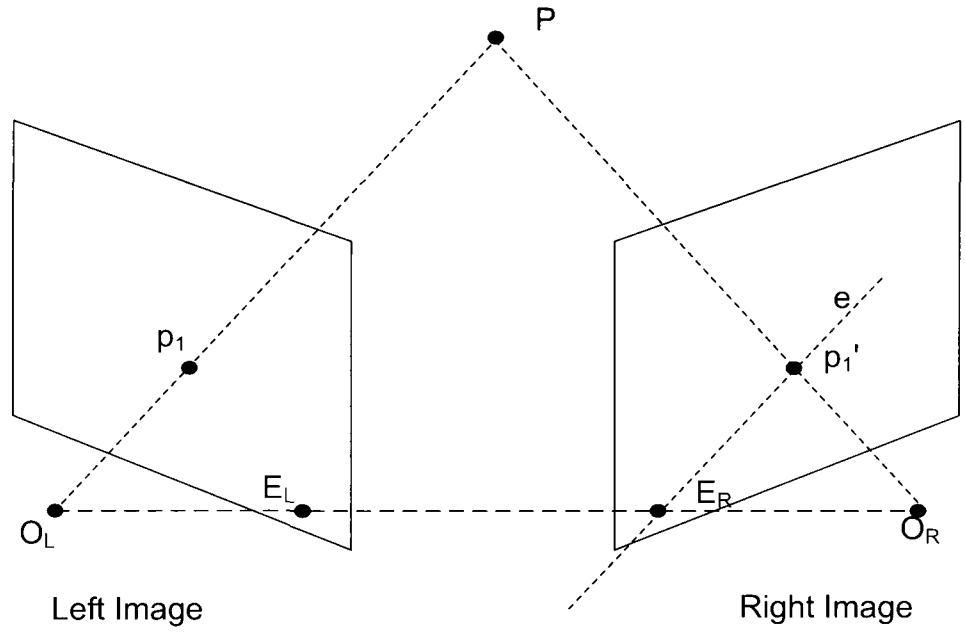


Figure A.2: Epipolar line

In our application, the 3D point is reconstructed based on its projections through different camera views. For each view, the projection of a 3D point is searched in P's epipolar line calculated by using the calibration result. To verify whether the calibration result is good enough, the accuracy of the epipolar line calculated by the calibration result is measured. The epipolar line based pixel matching is shown in Figure A.2. p_1 is a 2D pixel in the left image. Based on the internal and external parameters of the left camera and the right camera, we can draw the epipolar line e of pixel p_1 in the right image. If p_1 has a matched pixel p_1' in the right image, p_1' should be on the line e based on the epipolar line property. If the camera parameters are not accurate enough, the calculated epipolar line will be far from the real epipolar line e. In this case, the pixel matching procedure will fail.

If we use the previous example, the reconstruction task is to calculate the 3D point P which is projected into pixel p_1 in left image. p_2 is the real corresponding pixel of p_1 in the right image. e is the real epipolar line of p_1. In practice, the epipolar line of p_1 will be calculated as the e' due to the error in the calibration step. Therefore, the corresponding

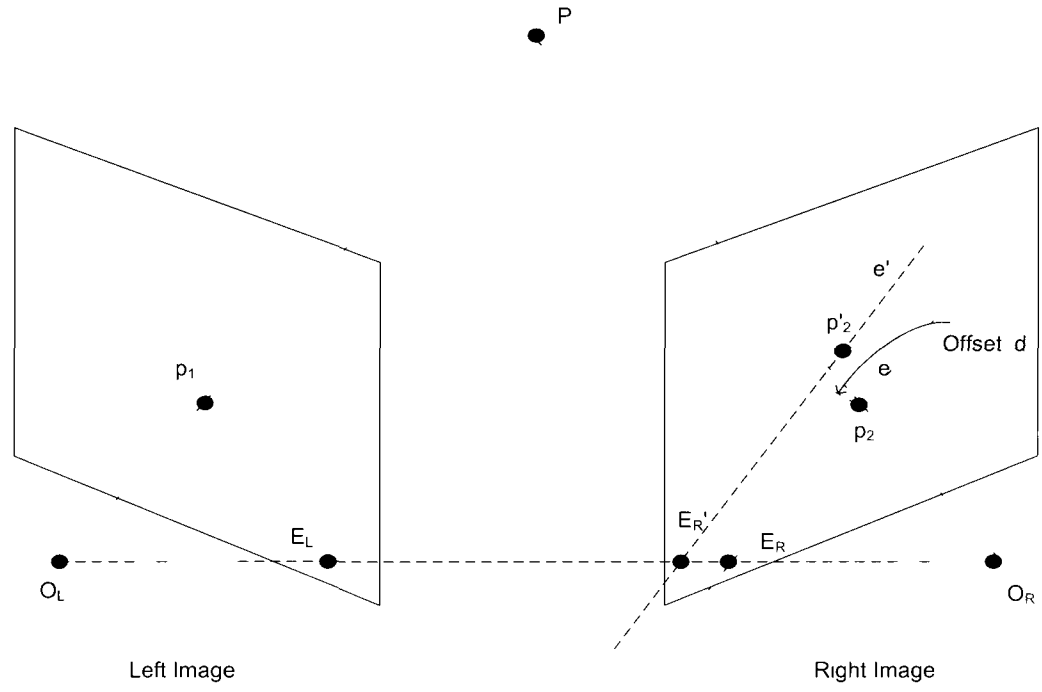


Figure A 3 Estimation of calibration error

pixel of p_1 will be calculated as p_2' on the line e' We hope the real matched point p_2 is not far from the epipolar line e' Then a high quality reconstruction of P can be achieved Here, we use the offset d to measure the calibration error, as shown in Figure A 3 The offset d is the distance between the corresponding pixel p_2 and the estimated epipolar line e' We find the real matched pixel p_2 by KLT feature tracking Finally, we use a calibration result which has the average calibration error with less than one pixel

A.2.2 Calculation of Calibration Error

There are 25 single cameras in the ProFUSION25 In order to estimate the calibration error of each single camera, 24 camera pairs are formed The central camera 12 is always set as the left camera which is also called as the reference camera The other camera is set as the right camera in each estimation camera set The approximation matching pixels are provided by KLT tracking algorithm

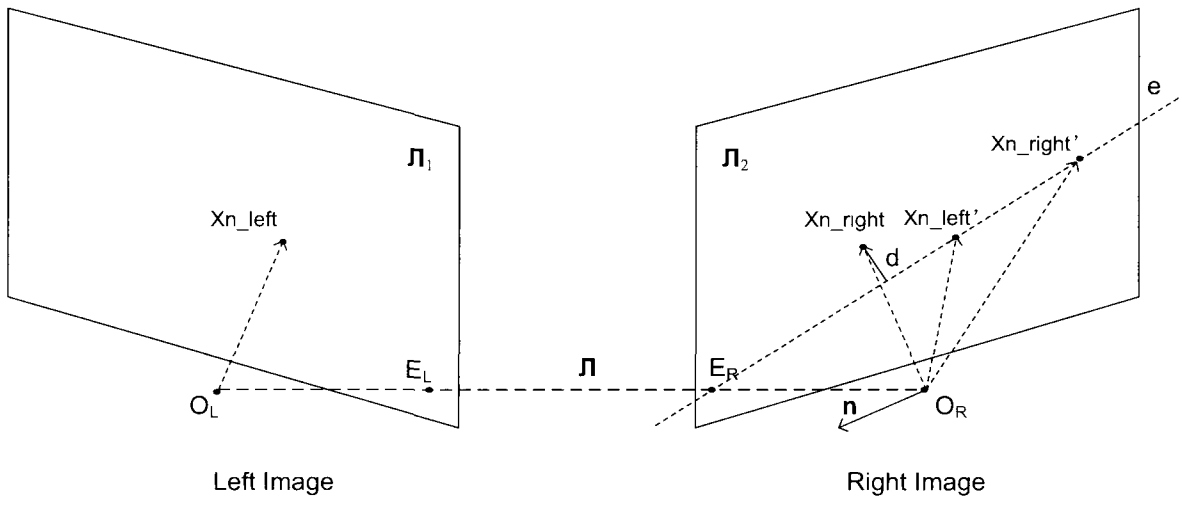


Figure A.4: Calculation of calibration error

As shown in Figure A.4, the error of the camera parameters can be estimated based on a pair of matched KLT features p_1 and p_2. p_1 is in the left image. p_2 is in the right image. Xn_{left} is the point with the normalized position of p_1 in the left camera's normalized coordinate system. Xn_{right} is the point with the normalized position of p_2 in right camera's normalized coordinate system. Point O_L and O_R are view points respective to left and right camera. Point E_L and E_R are epipolar points calculated from the calibration result. Π_1 is the image plane in the left camera. Π_2 is the image plane in the right camera. Π is the epipolar plane which contains points O_L, O_R and Xn_{left}. Line e is p_1's epipolar line in the plane Π_2. n is the normal vector of epipolar plane Π. Xn'_{right} is the intersection point of line e and vector $O_R Xn_{right}$'s projection line in plane Π. Xn'_{left} is the intersection of line e and vector $O_L Xn_{left}$'s parallel line which contains point O_R. The calibration error can be measured as the offset d calculated as shown in Table A.3. In Table A.4, the average offset distance d is measured for each right camera related to the reference camera (camera 12) based on the internal and external parameters generated by the camera calibration toolbox. Table A.5 shows the average offset distance d based on the internal parameters generated by camera calibration toolbox and the external parameters optimized by the SBA. As shown in the experiment, the accuracy of calibration result improves by applying the SBA optimization on the external parameters.

Definition
$v1{=}O_L X n_{left}$, $v2{=}O_R X n_{right}$, $v3{=}O_R X n'_{right}$, $v4{=}O_R X n'_{left}$
$v3'.z$ is the Z-axis component of $v3'$, $v4'.z$ is the Z-axis component of $v4'$
K is camera internal matrix, R is the rotation matrix, t is the translation matrix
Calculation
$n = (R \times v1) \times t$
$v3' = (n \times v1) \times n)$, $v3 = v3'/v3'.z$
$v4' = R \times v1$, $v4 = v4'/v4'.z$
$d = \|(K \times v2 - K \times v3) \times (K \times v4 - K \times v3)\|/\|v4 - v3\|$

Table A.3: Offset distance estimation

Camera0 1.445100	Camera1 0.590900	Camera2 0.405300	Camera3 0.733400	Camera4 0.809200
Camera5 1.618600	Camera6 0.789700	Camera7 0.936600	Camera8 0.477200	Camera9 0.497500
Camera10 0.619300	Camera11 1.119800	Camera12	Camera13 2.285200	Camera14 0.532500
Camera15 0.786300	Camera16 0.492200	Camera17 1.425700	Camera18 1.964700	Camera19 0.945600
Camera20 0.698200	Camera21 0.136600	Camera22 1.230500	Camera23 0.513600	Camera24 2.214300

Table A.4: Average offset distance d based on initial external parameters

The matched KLT feature pairs are used as input data in the estimation procedure of the calibration error. So far we assumed that there is no error in the KLT feature pairs, which means two feature points of each feature pair exactly map to the same 3D point. But to get a fair estimate of the calibration error, we also need to estimate the error of the KLT matching. To this end, we use the Bumblebee 2 to estimate the error in the KLT feature pairs. The Bumblebee2 is a stereo camera which contains two single cameras. The internal and external parameters of each single camera are given by the manufacturer. The root mean square error of its calibration result is under 0.1 pixels. Therefore, if we use the KLT feature pairs generated in the Bumblebee2 images

Camera0 0.445600	Camera1 0.310400	Camera2 0.400100	Camera3 0.391200	Camera4 0.570500
Camera5 0.534800	Camera6 0.342700	Camera7 0.261300	Camera8 0.287800	Camera9 0.459500
Camera10 0.538100	Camera11 0.187000	Camera12 0.000000	Camera13 0.174400	Camera14 0.296200
Camera15 0.711500	Camera16 0.301500	Camera17 0.246100	Camera18 0.296200	Camera19 0.379100
Camera20 0.708000	Camera21 0.111500	Camera22 0.713200	Camera23 0.156200	Camera24 0.513900

Table A.5: Average offset distance d based on optimized external parameters

to estimate the calibration error of Bumblebee2, the calibration error can be treated as an approximation error of the KLT feature pairs. In our test, the average error of KLT pairs is 0.5 pixels measured by the offset distance.

Bibliography

[1] N. Ahmed, C. Theobalt, C. Rossl, S. Thrun, and H.-P. Seidel. Dense correspondence finding for parametrization-free animation reconstruction from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 1–8, 2008.

[2] B. Appleton and H. Talbot. Globally minimal surfaces by continuous maximal flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:106–118, 2006.

[3] C. Baillard and A. Zisserman. A plane-sweep strategy for the 3D reconstruction of buildings from multiple images. *International Archives of Photogrammetry and Remote Sensing*, 32(2):56–62, 2000.

[4] H.H. Baker and T.O. Binford. Depth from edge and intensity based stereo. In *the 7th International Joint Conference on Artificial Intelligence*, pages 631–636, 1981.

[5] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *Technical Report MSR-TR-2009-179*, 2009.

[6] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 296–302, 1991.

[7] J.-Y. Bouguet. Camera calibration toolbox is available at:. `http://www.vision.caltech.edu/bouguetj/calib_doc/`, 2008.

[8] D. Bradley, T. Boubekeur, and W. Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 1–8, 2008.

[9] D. Bradley, T. Popa, A. Sheffer, W. Heidrich, and T. Boubekeur. Markerless garment ccapture. In *ACM Transaction on Graphics*, pages 1–9, 2008.

[10] A. Broadhurst, T. W. Drummond, and R. Cipolla. A probabilistic framework for space carving. In *International Conference of Computer Vision*, volume 1, pages 388–393, 2001.

[11] D.C. Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[12] M. Z. Brown, D. Burschka, G. D. Hager, and S. Member. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:993–1008, 2003.

[13] A. Bruhn, J. Weickert, and C. Schnoerr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005.

[14] T. A. Clarke and J. G. Fryer. The development of camera calibration methods and models. *The Photogrammetric Record*. 16:51–66, 1998.

[15] R.T. Collins. A space-sweep approach to true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 358–363, 1996.

[16] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *International Conference on Computer Graphics and Interactive Techniques*, pages 303–312, 1996.

[17] P. Viola D. Snow and R. Zabih. Exact voxel occupancy with graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 345–352, 2000.

[18] E. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[19] C. R. Dyer. Volumetric scene reconstruction from multiple views. In *Foundations of Image Understanding*, pages 469–489. Kluwer, 2001.

[20] M. H. Fanaswala. *Master Thesis: Regularized Super-Resolution of Multi-View Images*. Carleton University, 2009.

[21] J. Fryer and D.C. Brown. Lens distortion for close-range photogrammetry. *Photogrammetric Engineering and Remote Sensing*, 52(1):51–58, 1986.

[22] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[23] Y. Furukawa and J. Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. *IEEE Conference on Computer Vision and Pattern Recognition*, 0:1–8, 2008.

[24] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 1–8, 2007.

[25] P. Gargallo, E. Prados, and P. Sturm. Minimizing the reprojection error in surface reconstruction from images. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[26] M. Goesele, B. Curless, and S. M. Seitz. Multi-view stereo revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2402–2409, 2006.

[27] D. B. Goldman and J.-H. Chen. Vignette and exposure calibration and compensation. *IEEE International Conference on Computer Vision*, 1:899–906, 2005.

[28] M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 1–8, 2007.

[29] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill. 3D pose tracking with linear depth and brightness constraints. In *IEEE International Conference on Computer Vision*, volume 1, pages 206–213, 1999.

[30] E.E. Hemayed. A survey of camera self-calibration. In *IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 351–357, 2003.

[31] B. K. P. Horn and J. G. Harris. Rigid body motion from range image sequences. *Computer Vision Graphics and Image Processing: Image Understanding*, 53(1):1–13, 1991.

[32] B. K. P. Horn and B. G. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[33] Konica Minoca Minolta Sensing Inc. *Non-Contact 3D Digitizer VIVID 910/VI-910 Instruction Manual*. Japan.

[34] R. Kimmel and J. A. Sethian. Computing geodesic paths on manifolds. In *Processings of the National Academy of Science*, pages 8431–8435, 1998.

[35] L. Kobbelt and M. Botsch. A survey of point-based techniques in computer graphics. *Computers and Graphics Archive*, 28(6):801–814, 2004.

[36] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conference on Computer Vision*, pages 82–96, 2002.

[37] V. Kraevoy and A. Sheffer. Cross-parameterization and compatible remeshing of 3D models. In *ACM Transaction on Graphics*, pages 861–869, 2004.

[38] V. Kraevoy and A. Sheffer. Template-based mesh completion. In *the third Eurographics symposium on Geometry processing*, page 13, 2005.

[39] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *IEEE International Journal of Computer Vision*, 38(3):199–218, 2000.

[40] M. A. Lanthier. *PhD Thesis: Shortest Path Problem on Polyhedral Surfaces*. Carleton University, 1999.

[41] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994.

[42] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[43] Y. Liu and M. A. Rodrigues. Correspondenceless motion estimation from range images. In *IEEE International Conference on Computer Vision*, volume 1, pages 654–659, 1999.

[44] M. I. A. Lourakis and A. A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1):1–30, 2009.

[45] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[46] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[47] L. Lucchese, G. Doretto, and G. M. Cortelazzo. Frequency domain estimation of 3-d rigid motion based on range and intensity data. In *the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, page 107, 1997.

[48] A. Manessis, A. Hilton, P. Palmer, P. McLauchlan, and X. Shen. Reconstruction of scene models from sparse 3D structure. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, page 2666, 2000.

[49] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 369–374, 2000.

[50] 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 *International Conference on Computer Vision*, volume 0, pages 1–8, 2007.

[51] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM Journal of Computing*, 16(4):647–668, 1987.

[52] D. D. Morris and T. Kanade. Image-consistent surface triangulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 1332, 2000.

[53] V. S. Nalwa. *A Guided Tour of Computer Vision.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1993.

[54] J.-C. Nebel and A. Sibiryakov. Range flow from stereo-temporal matching: Application to skinning. In *IASTED International Conference on Visualization, Imaging, and Image Processing*, 2002.

[55] J.-P. Pons, R. Keriven, and O. Faugeras. Modelling dynamic scenes by registering multi-view image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 822–827, 2005.

[56] D. Pritchard and W. Heidrich. Cloth motion capture. *Computer Graphics Forum (Eurographics 2003)*, 22(3):263–271, 2003.

[57] T. Schuchert, T. Aach, and H. Scharr. Range flow for varying illumination. In *the 10th European Conference on Computer Vision*, pages 509–522, 2008.

[58] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. The middlebury datasets are available at:. `http://vision.middlebury.edu/mview/`, 2009.

[59] S. M. Seitz, B. C., J. D., D.Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528, 2006.

[60] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *IEEE Conference on Computer Vision and Pattern Recognition*. volume 0. pages 1067–1073, 1997.

[61] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[62] G. Slabaugh, B. Culbertson, T. Malzbender, and R. Schafe. A survey of methods for volumetric scene reconstruction from photographs. In *International Workshop on Volume Graphics*, 2001.

[63] H. Spies, B. Jähne, and J.L. Barron. Dense range flow from depth and intensity data. In *the International Conference on Pattern Recognition*, page 1131, 2000.

[64] H. Spies, B. Jähne, and J.L. Barron. Range flow estimation. *Computer Vision and Image Understanding*, 85(3):209–231, 2002.

[65] D. Sun, S. Roth, J. P. Lewis, and M. J. Black. Learning optical flow. In *the 10th European Conference on Computer Vision*, pages 83–97, 2008.

[66] R. Szeliski. A multi-view approach to motion and stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 157–163, 1999.

[67] C.J. Taylor. Surface reconstruction from feature based stereo. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 184–190, 2003.

[68] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *the 14th annual conference on Computer graphics and interactive techniques*, volume 21, pages 205–214, 1987.

[69] A. Tevs, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Isometric registration of ambiguous and partial data. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 0, pages 1185–1192, 2009.

[70] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, International Journal of Computer Vision, April 1991.

[71] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *the Sixth International Conference on Computer Vision*, pages 839–846, 1998.

[72] S. Tran and L. Davis. 3D surface reconstruction using graph cuts with surface constraints. In *European conference on computer vision*, pages 219–231, 2006.

[73] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *the International Workshop on Vision Algorithms*, pages 298–372, 2000.

[74] S. Vedula, S. Baker, R. Collins, T. Kanade, and P. Rander. Three-dimensional scene flow. In *the International Conference on Computer Vision*, volume 2, page 722, Washington, DC, USA, 1999.

[75] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 27(3):475–480, 2005.

[76] R. G. Yang and M. Pollefeys. A versatile stereo implementation on commodity graphics hardware. *Real-Time Imaging*, 11(1):7–18, 2005.

[77] T. Yu and J. Lang. Window-based range flow with an isometry constraint. In *IEEE Conference on Computer and Robot Vision (CRV 2010)*, 2010.