

Template-based Monocular 3-D Shape Reconstruction And Tracking Using Laplacian Meshes

THÈSE N° 6891 (2016)

PRÉSENTÉE LE 23 MARS 2016

À LA FACULTÉ INFORMATIQUE ET COMMUNICATIONS
LABORATOIRE DE VISION PAR ORDINATEUR
PROGRAMME DOCTORAL EN INFORMATIQUE ET COMMUNICATIONS

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Tien Dat NGO

acceptée sur proposition du jury:

Prof. M. Pauly, président du jury
Prof. P. Fua, directeur de thèse
Prof. A. Bartoli, rapporteur
Prof. L. Smith, rapporteur
Prof. W. Jakob, rapporteur



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2016

To the memory of my beloved father

Ngô Đức Doanh

and to my wonderful mother

Nguyễn Thị Ngoan

Acknowledgements

This thesis could not have been completed without support and contributions of many people, to whom I am indebted.

First and foremost, I would like to express my deepest gratitude to my thesis supervisor Prof. Pascal Fua for his professional supervision and guidance. I would like to thank Pascal for accepting me as a member of his exceptional laboratory. Being a professor at EPFL is a very far from easy job, I admire his ability to efficiently direct the laboratory with his various talents.

I would like to send my sincere thanks to my thesis jury members: Prof. Mark Pauly, Prof. Adrien Bartoli, Prof. Lloyd Smith, and Prof. Wenzel Jakob for kindly accepting to evaluate my work and providing insightful comments.

I give my special thanks to Dr. Anne Jorstad without whose support and encouragement I would not have completed my Ph.D. study. I thank Anne for our numerous discussions and her sincere advice on both academic and social issues. Whenever I need someone to talk to, Anne is always available with her calmness listening to me. I consider her as a great colleague as well as an elder sister.

I would like to sincerely thank everyone in Computer Vision Laboratory, past and present, for their kindness and friendship. I am fortunate to have a chance to work with them. They are exceptional researchers from whom I have learned a lot. Thank you all for sharing the knowledge with me and for the good time we have had. Thank you for the Friday night beers we had together: Carlos, Roger, Amaury, Pablo, Kwang, Timur, Agata, Przemek, Artem, Amos, Bugra, Pol, RK ...

I especially thank the collaborators and colleagues with whom I have worked closely. To Dr. Aydin Varol for his valuable advice at the start of my Ph.D. study. To Jonas Ostlund from whose research I inherited a lot. To Alberto Crivellaro for being such an enthusiastic colleague and a good friend. To Sanghyuk Park for the nice work we did together. To Dr. Stephane Magnenat and Mattia Ryffel for our interesting AR Coloring Book project. To Dr. Mathieu Salzmann for proofreading my thesis draft. To Dr. Julien Pilet for setting up various connections that have helped me a lot in my career.

I would like to send my sincere thanks to our secretary Josiane Gisclon for her kindness

Acknowledgements

and great help whenever I need it.

I would like to thank my Vietnamese friends in Lausanne, who enrich my Ph.D. life and keep me balanced. My special thanks go to the group of Vietnamese students and researchers at EPFL with whom we often have lunch at Le Vinci. To Nguyễn Mạnh Hùng for kindly helping me to translate the abstract of this thesis into French.

I dedicate this thesis to the memory of my beloved father Ngô Đức Doanh and my wonderful mother Nguyễn Thị Ngoan for their measureless support, continuous encouragement, and eternal unconditional love that have sustained me through difficult times. They have sacrificed so much to raise me and my sisters and to give us better lives today.

I must acknowledge my wife, Phạm Thị Vân Anh, without whose love and care, I would not have finished my Ph.D. journey.

Last but not least, I cannot fully list the people who indirectly lead to this thesis. I would like to thank you all, especially my extended family, my sisters, my brothers, and my dear friends, for having supported me through my time.

Lausanne, February 2016

Ngo Tien Dat

Abstract

This thesis addresses the problem of recovering the 3-D shape of a deformable object in single images, or image sequences acquired by a monocular video camera, given that a 3-D template shape and a template image of the object are available. While being a very challenging problem in computer vision, being able to reconstruct and track 3-D deformable objects in videos allows us to develop many potential applications ranging from sports and entertainments to engineering and medical imaging. This thesis extends the scope of deformable object modeling to real-world applications of fully 3-D modeling of deformable objects from video streams with a number of contributions.

We show that by extending the Laplacian formalism, which was first introduced in the Graphics community to regularize 3-D meshes, we can turn the monocular 3-D shape reconstruction of a deformable object given correspondences with a reference image into a much better-posed problem with far fewer degrees of freedom than the original one. This has proved key to achieving real-time performance while preserving both sufficient flexibility and robustness. Our real-time 3-D reconstruction and tracking system of deformable objects can very quickly reject outlier correspondences and accurately reconstruct the object shape in 3D. Frame-to-frame tracking is exploited to track the object under difficult settings such as large deformations, occlusions, illumination changes, and motion blur.

We present an approach to solving the problem of dense image registration and 3-D shape reconstruction of deformable objects in the presence of occlusions and minimal texture. A main ingredient is the pixel-wise relevancy score that we use to weigh the influence of the image information from a pixel in the image energy cost function. A careful design of the framework is essential for obtaining state-of-the-art results in recovering 3-D deformations of both well- and poorly-textured objects in the presence of occlusions.

We study the problem of reconstructing 3-D deformable objects interacting with rigid ones. Imposing real physical constraints allows us to model the interactions of objects in the real world more accurately and more realistically. In particular, we study the problem of a ball colliding with a bat observed by high speed cameras. We provide quantitative measurements of the impact that are compared with simulation-based methods to evaluate which simulation predictions most accurately describe a physical quantity of interest and to improve the models.

Abstract

Based on the diffuse property of the tracked deformable object, we propose a method to estimate the environment irradiance map represented by a set of low frequency spherical harmonics. The obtained irradiance map can be used to realistically illuminate 2-D and 3-D virtual contents in the context of augmented reality on deformable objects. The results compare favorably with baseline methods.

In collaboration with Disney Research, we develop an augmented reality coloring book application that runs in real-time on mobile devices. The app allows the children to see the coloring work by showing animated characters with texture lifted from their colors on the drawing. Deformations of the book page are explicitly modeled by our 3-D tracking and reconstruction method. As a result, accurate color information is extracted to synthesize the character's texture.

Key words: deformable surfaces, monocular shape recovery, Laplacian formalism, augmented reality, template matching, image registration, occlusions, minimal texture, spherical harmonics, interactive books

Résumé

Cette thèse aborde le problème de reconstruction de la forme 3-D d'un objet déformable dans des images séparées, ou des séquences d'images qui sont acquises par une caméra vidéo monoculaire, en disposition du modèle de la forme 3-D et de l'image de modèle de l'objet. Bien que la reconstruction et le suivi des objets 3-D déformables dans les vidéos soient un problème très difficile dans la vision par ordinateur, leurs applications sont très potentiellement variées dans le domaine du sport, de le divertissement, de l'ingénierie ainsi que de l'imagerie médicale. Cette thèse contribue à l'extension de la modélisation 3-D des objets déformables de flux de vidéo afin d'appliquer dans le monde réel.

Nous trouvons que par l'extension du formalisme de Laplace, qui a été introduit dans la communauté graphique pour régulariser des maillages 3-D, la reconstruction monoculaire 3-D d'un objet déformable en sachant sa correspondance avec une image de référence devient un problème mieux posé avec beaucoup moins de degrés de liberté par rapport à celui d'origine. Cette contribution est la clé pour atteindre la performance en temps-réel, tout en préservant suffisamment de flexibilité et de robustesse. Notre système de reconstruction et suivi en 3-D des objets déformables en temps réel peut ainsi très rapidement rejeter les fausses et reconstruire précisément la forme 3-D de l'objet. L'exploitation du suivi image par image permet suivre l'objet sous des conditions difficiles telles que de larges déformations, d'occlusions, des changements d'éclairage et de flou au mouvement.

Nous présentons une approche pour résoudre le problème du recalage dense d'images et la reconstruction de la forme 3-D des objets déformables en présence des occlusions et la texture minimale. Nous utilisons principalement le score sur la pertinence de pixels pour évaluer l'influence des informations données par un pixel dans la fonction du coût de l'image. Un système est bien conçu afin d'obtenir les résultats sur l'état de l'art dans la reconstruction de déformations 3-D quel que soit les objets bien ou mal-texturés en présence d'occlusions.

Nous étudions le problème de reconstruction de la forme 3-D des objets déformables en interaction avec ceux rigides. En imposant des contraintes physiques, nous arrivons à modéliser les interactions entre les objets réels de façons plus précise et plus réaliste. Concrètement, nous étudions la collision entre une balle et une batte acquise par des caméras à haute vitesse. Nous fournissons des mesures quantitatives de l'impact. Ces

Résumé

mesures quantitatives sont comparées avec les méthodes basées sur la simulation pour évaluer et décider quelles sont les prédictions de simulation décrivant mieux précisément une quantité physique et améliorer les modèles.

Basé sur la diffusibilité des objets déformables suivis, nous proposons une méthode d'estimation d'une carte irradiant environnementale qui est représentée par un ensemble des harmoniques sphériques en basse fréquence. La carte irradiant obtenue peut être utilisée pour illuminer de façon réaliste des contenus virtuels 2-D et 3-D dans le contexte de réalité augmentée sur des objets déformables. Les résultats sont plus favorables par rapport aux méthodes de référence.

Dans le cadre d'une collaboration avec Disney Research, nous avons développé une application de coloriage des livres de réalité augmentée sur les appareils mobiles. L'application permet aux enfants de visualiser leur coloriage en affichant les personnages animés dont la texture est prise à partir des couleurs sur le dessin. Les déformations des pages du livre sont explicitement modélisées par notre méthode de suivi et reconstruction 3-D. En conséquence, les informations précisées des couleurs sont extraites pour synthétiser la texture du personnage.

Mots clefs : surfaces déformables, reconstruction monoculaire de la forme, formalisme de Laplace, réalité augmentée, reconnaissance du modèle, recalage d'images, occlusions, texture minimale, harmoniques sphériques, livres interactifs

Contents

Acknowledgements	i
Abstract (English/Français)	iii
List of figures	xi
List of tables	xv
1 Introduction	1
1.1 Overview	1
1.2 Applications	3
1.2.1 Sports	3
1.2.2 Engineering	3
1.2.3 Entertainment and Marketing	5
1.2.4 Medical Imaging	6
1.3 Contributions	7
1.3.1 Real-time Reconstruction and Tracking	7
1.3.2 Dense Image Registration and 3-D Reconstruction.	8
1.3.3 Applications	9
1.4 Thesis Outline	11
1.5 Publications	12
2 Related Work	15
2.1 Non-rigid Structure from Motion	15
2.2 Template-based 3-D Deformable Object Reconstruction	18
2.3 Dense Image Registration	21
3 Real-time 3-D Reconstruction and Tracking	25
3.1 Foreword	25
3.2 Linear Problem Formulation	26
3.3 Laplacian Formulation	28
3.3.1 Regularization Matrix	30
3.3.1.1 Planar Reference Shape	30
3.3.1.2 Non-Planar Reference Shape	32

3.3.1.3	Properties of the Regularization Term	33
3.3.2	Linear Parameterization	34
3.4	Rejecting Outliers in 3D	35
3.5	Quantitative Results on Real Data	36
3.5.1	Sequences, Templates, and Validation	36
3.5.2	Robustness to Erroneous Correspondences	40
3.5.3	Control Vertex Configurations	41
3.5.4	Comparative Accuracy of the Four Methods	42
3.5.5	Analysis	43
3.6	Real-time Deformable Object Tracking	47
3.6.1	Fast Robust Outlier Rejection in 2D	47
3.6.2	Surface Reconstruction by Detection	51
3.6.3	Surface Reconstruction by Tracking	51
3.6.3.1	Temporal Consistency	52
3.6.3.2	KLT-based Feature Point Tracking	52
3.6.3.3	Active Search	52
3.6.4	Real-time Deformable Object Tracking Evaluation	53
3.6.4.1	Robustness	53
3.6.4.2	Reconstruction Accuracy	55
3.6.4.3	Performance	55
3.7	Conclusion	56
4	Dense image registration and 3-D shape reconstruction	59
4.1	Foreword	59
4.2	Proposed Framework	60
4.2.1	Template Matching	61
4.2.2	Robust Optimization	62
4.2.2.1	Optimization Scheme	62
4.2.2.2	Feature Selection	63
4.2.2.3	Similarity Function Selection	63
4.2.3	Handling Occlusions with a Relevancy Score	65
4.2.4	Handling Sparsely Textured Surfaces	67
4.3	Experiments and Results	67
4.3.1	Basin of Convergence	69
4.3.2	Well-Textured Surfaces	69
4.3.3	Sparsely-Textured Surfaces	72
4.3.4	Additional Results	76
4.4	Conclusion	80
5	Applications	81
5.1	Deformable Object Reconstruction in Interactions	81
5.1.1	Tracking Rigid Object Pose	82
5.1.1.1	Edge-based Tracking	82

5.1.1.2	Template Matching based Tracking	95
5.1.2	Ball-Bat Study	97
5.1.2.1	Method	98
5.1.2.2	Results	100
5.1.2.3	Comparisons to Simulations	105
5.1.3	Additional Qualitative Results	114
5.2	Real-time Template Selection	116
5.3	Lighting Estimation	117
5.3.1	Method	118
5.3.2	Results	122
5.4	Coloring Book Application	126
5.4.1	Content Creation	127
5.4.2	Image Processing	129
5.4.3	Tracking	130
5.4.4	Texture Generation	131
5.4.5	Augmentation	131
5.4.6	Results	131
5.5	Conclusion	133
6	Concluding Remarks	135
6.1	Summary	135
6.2	Future Work	136
	Bibliography	139
	Curriculum Vitae	151

List of Figures

1.1	Examples of the objects to which we applied our reconstruction methods .	2
1.2	Many sport design tasks could benefit from 3-D shape recovery of deformable objects	4
1.3	An impact of a baseball colliding with a rigid cylinder serving as a bat was captured using a very high speed camera	4
1.4	Image-based deformation measurements could help engineering design tasks	5
1.5	Demonstration of Augmented Reality on a deformable object in 2D	5
1.6	Video-based 3-D surface reconstruction can be applied to surgeries	6
1.7	Real-time deformable surface reconstruction and tracking of a paper and a cushion.	7
1.8	Tracking a sparsely textured surface in the presence of occlusion	8
1.9	Studying the deformation behaviors of a ball colliding with a rigid cylindrical object	10
1.10	Augmented reality with our environment illumination estimation	10
1.11	Coloring book application scenarios with different drawings and colorings.	11
3.1	Linear parameterization of the mesh using control vertices	28
3.2	Conditioning of the regularization matrix	29
3.3	Building the regularization matrix \mathbf{A}	31
3.4	Paper and Apron datasets with a planar template	37
3.5	Cushion and banana leaf datasets with a non-planar template.	38
3.6	Sail dataset with a non-planar template	39
3.7	Probability of success as a function of the number of inlier matches and proportion of outliers	41
3.8	Templates and control vertices	42
3.9	Accuracy results when using planar templates and different numbers of control vertices for the paper and apron sequences	45
3.10	Accuracy results when using non-planar templates and different numbers of control vertices for the cushion, banana leaf, and sail sequences	46
3.11	Influence of the regularization parameter	47
3.12	Condition numbers and singular values of the matrix \mathbf{M}_{w_r}	48
3.13	Unconstrained vs constrained optimization results	49
3.14	Additional unconstrained and constrained results for the paper sequence .	50

List of Figures

3.15	Probability of having at least 90% of mesh vertices re-projected within 2 pixels of the solution as a function of the number of inlier matches and proportion of outliers	54
3.16	Real-time deformable surface reconstruction and tracking on a laptop. . .	56
3.17	Real-time deformable surface reconstruction and tracking on an iPad . . .	57
4.1	Tracking a sparsely textured surface in the presence of occlusion	60
4.2	An image warping function maps a pixel from the template image onto the deforming surface in the input image.	61
4.3	The relevancy score results using various methods on the cloth dataset and the sparsely textured paper dataset	65
4.4	Relevancy scores for the well-textured paper dataset.	67
4.5	Robustness of various alignment functions <i>w.r.t.</i> translations between a template and an input image, showing the basin of convergence of the alignment costs around the correct position	70
4.6	Reconstruction results computed using the vertex-to-vertex error metric on the well-textured paper dataset without occlusions	71
4.7	Reconstruction results computed using the vertex-to-cloud error metric on the well-textured paper dataset without occlusions.	71
4.8	Reconstruction results computed using the vertex-to-vertex error metric on the well-textured paper dataset, with occlusions	72
4.9	Reconstruction results computed using the vertex-to-cloud error metric on the well-textured paper dataset, with occlusions.	72
4.10	Output for a single frame showing relative reconstruction accuracies . . .	73
4.11	Tracking a rotating deformable surface	74
4.12	Sample reconstructions from the [Salzmann et al., 2008b] dataset	75
4.13	Reconstruction results computed using the vertex-to-vertex error metric on the sparsely textured paper dataset	75
4.14	Reconstruction results computed using the vertex-to-cloud error metric on the sparsely textured paper dataset	76
4.15	Reconstruction results on the same single frame of the sparsely-textured dataset	77
4.16	Reconstruction results computed using the vertex-to-vertex error metric on the t-shirt dataset.	77
4.17	Reconstruction results computed using the vertex-to-cloud error metric on the the t-shirt dataset.	78
4.18	Our representative reconstructions on the t-shirt dataset with artificial occlusions added	78
4.19	Image registration and surface reconstruction on the sparsely textured sail surface	79
4.20	Surface reconstruction of an animation capture from a monocular camera stream.	79

4.21 Results on the self occlusion dataset	80
5.1 Surface deformations caused by another object	82
5.2 The algorithm rotates and translates the CAD model in 3D so that its perspective camera projection matches to what is observed in the input image	83
5.3 Select edge profile and control points	84
5.4 Using the current pose estimate \mathbf{R} and \mathbf{t} , the projection of the CAD model roughly aligns with the input image	85
5.5 After applying the pose correction $\Delta\mathbf{R}$ and $\Delta\mathbf{t}$, we expect the projection of the control point to locate in its corresponding image edge	86
5.6 Pose estimation of a machinery part from a single view	90
5.7 Pose estimation of a half cylinder from a single view	91
5.8 Projections of the half cylinder onto the top view in its initial and final pose estimates using the side view image	92
5.9 Pose estimation of the half cylinder from two views	93
5.10 Pose estimation of the ball from two views	94
5.11 Tracking the 3-D pose of a small object using direct image alignment is an attractive approach that globally exploits most of the image information.	95
5.12 The image warping function that aligns the template image to the input image [Crivellaro and Lepetit, 2014]. It transforms a pixel location \mathbf{x} on the template image T to a new pixel location on the input image J	96
5.13 Results of tracking the 3-D pose of a small pen using direct image alignment and render-based template generation	97
5.14 Experiment setup of the ball-bat study. The ball is launched by a ball canon against a half cylinder. Two high speed cameras are used to capture the impact.	98
5.15 Template-based 3-D shape reconstruction of the ball	99
5.16 An experiment setup of the ball-bat study with stereo cameras seen from different viewpoints.	101
5.17 Reconstruction of a ball colliding with a cylinder in a video sequence	102
5.18 Median distance in each frame between our monocularly reconstructed surface and a 3-D point cloud obtained from stereo	103
5.19 Video-based 3-D reconstruction of the ball colliding with the cylinder	104
5.20 Views of a mesh of a softball impacting a solid cylinder, showing two planes of symmetry.	106
5.21 Comparison of the experimental ball diameter along Oy normal directions with the finite element models.	106
5.22 Comparison of the experimental ball diameter along Ox normal directions with the finite element models.	107
5.23 Comparison of the experimental ball diameter along Oz normal directions with the finite element models.	107

List of Figures

5.24	Simulation results with the visco-elastic material MAT 006a	108
5.25	Simulation results with the visco-elastic material MAT 006b	109
5.26	Simulation results with the visco-elastic material MAT 006c	110
5.27	Simulation results with the foam-based material MAT 057	111
5.28	Simulation results with the foam-based material MAT 083	112
5.29	Simulation results with the foam-based material MAT 181	113
5.30	Tracking a rigid pen and reconstructing the cushion with and without imposing non-penetration constraints during their interactions	115
5.31	Scalability of the template detection algorithm	117
5.32	Irradiance map computation by [Pilet et al., 2006]	120
5.33	Visualization of the first nine spherical harmonics projected on <i>Oxy</i> plane	121
5.34	Comparisons of methods estimating the environment irradiance map . . .	123
5.35	Estimating environment irradiance map of an office. There is a large window on the right side of the camera that illuminates the scene.	124
5.36	Comparison of augmented reality with and without environment illumination estimation.	125
5.37	Two examples of our augmented reality coloring book algorithm showing the colored input drawings and the captured texture applied to both visible and occluded regions of the corresponding 3-D characters.	127
5.38	The static content creation and the in-App live surface tracking, texturing, and rendering pipelines.	128
5.39	Image processing of the input image of the colored drawing	130
5.40	Coloring book application scenarios with different drawings and colorings	132
5.41	Screenshots from the App showing the robustness of our outlier rejection algorithm.	132

List of Tables

3.1	The five datasets used for quantitative evaluation. Number of frames in each sequence and specifics of the corresponding reference shape.	39
3.2	Computational time in milli-seconds of major steps of our tracking system measured on a first generation iPad Air	55
4.1	Reconstruction errors over a range of weighting coefficient values using the well-textured paper dataset.	69

1 Introduction

Recovering 3-D scene structure using ordinary video cameras has long been a fundamental problem in Computer Vision. In principle, it is possible to perform such 3-D reconstruction using a pair of images taken from different viewpoints, but only if the two images are acquired simultaneously or if the scene does not deform between the two captures. The problem becomes much more difficult when non-rigidity comes into play and a deformable object is only imaged by a single camera. This has been an active area of research for many years and the one we have focused on.

Deformable models have long been used to model the non-rigidity of target objects. They have been shown to be effective in applications as diverse as automated delineation of 2-D features, cartographic modeling, or 3-D modeling of human organs in the context of medical imaging. This thesis aims at extending their reach to applications of 3-D modeling of deformable objects from video streams, a domain in which they had not yet reached their full potential. They include modeling well-textured deformable objects in real-time, tracking and reconstructing poorly-textured objects in the presence of occlusions, and capturing the behaviors of deformable objects interacting with rigid ones.

In the remainder of this chapter, we will first describe in more detail the problem we address in this thesis and discuss a few potential practical applications. We will then discuss our contributions to the field.

1.1 Overview

Our goal in this work is to recover the 3-D shape of a deformable object either from single images or image sequences in a tracking scenario acquired by a single video camera, given that a 3-D template shape and a template image of the object are available. We systematically assume that the camera is calibrated and that its parameters do not change during acquisition. We use a 3-D triangle mesh to represent the object of interest in its rest shape as well as when it deforms. We also assume that the mesh topology

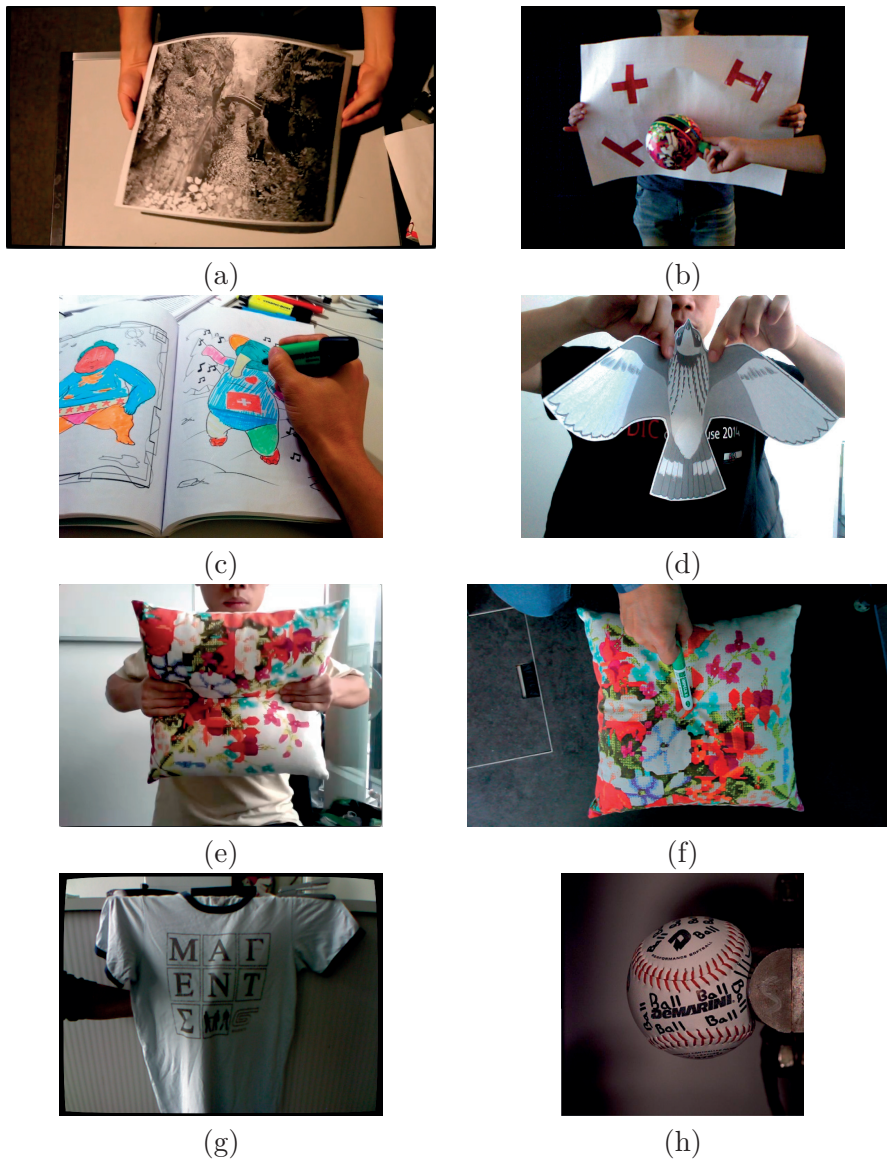


Figure 1.1: Examples of the objects to which we applied our reconstruction methods. (a) A well-textured sheet of paper. (b) A poorly-textured sheet of paper being partially occluded. (c) A drawing being colored by a child. (d) A bird-shape piece of paper used for animation capture. (e,f) A cushion being deformed by hands or by a pen. (g) A partially-textured t-shirt with non-uniform lighting. (h) A baseball colliding with a cylinder captured by a high speed camera.

does not change during deformations, meaning that the object does not tear or merge. The template image is used to extract the texture information of the deformable object of interest. We exploit salient feature points of well-textured objects and pixel-level information of poorly-textured ones. Example objects we worked with are presented in Fig. 1.1. They include a well-textured sheet of paper, a poorly-textured sheet of paper that is partially occluded, a page in a book being colored by a child, a t-shirt under

non-uniform lighting, a cushion being deformed by a rigid object, and a baseball colliding with a rigid cylinder at a very high speed.

1.2 Applications

Not only is working with all these different kinds of objects a challenging and open Computer Vision problem, but it also has many potential applications, which we discuss below.

1.2.1 Sports

Many sports could benefit from a system that reconstructs non-rigid 3-D shapes from videos and models their interactions with their environments. For example, sailors, especially those who race, want to analyze the effect of their maneuvers on the shape of their sails and on the resulting speed. These shape measurements help them better understand the behaviors of the boat, adjust their maneuvers, and potentially improve sail design. They also have a keen interest in the shape of their competitors' sails. As shown in Fig. 1.2 (a,b), video presents a clear advantage over other sensors that have to be placed on the sail itself, thus changing its behavior. Similarly, analyzing the deformations of any sports structure, such as the skis in Fig. 1.2 (c), in realistic situations could help understanding its performance and improving its design. Knowing the shape of a parachute in real-time could also help to automatically maneuver its course as demonstrated in Fig. 1.2 (d).

As another example, an accurate description of sports balls is needed to design protective equipments, such as helmets and pads, and striking equipment, such as bats, clubs and rackets. It would also help regulating agencies better understand equipment performance through experimental testing and numerical modeling. Given the geometric and material non-linearities inherent to ball impacts, finite element analysis is often used to simulate them. However, existing models are far from perfect. Therefore, there is a need for comparing impact simulations against reality not only to assess which predictions most accurately describe a physical quantity of interest but also to improve the models. Our video-based 3-D modeling of deformations can be used in these scenarios as depicted in Fig. 1.3.

1.2.2 Engineering

Along very similar lines, video-based 3-D measurements of deformable objects have also potential applications in many engineering fields. Such measurements could help mechanical engineers to better understand how complex structures truly behave under realistic working conditions. For example, plane wings in Fig. 1.4 (a) bend significantly

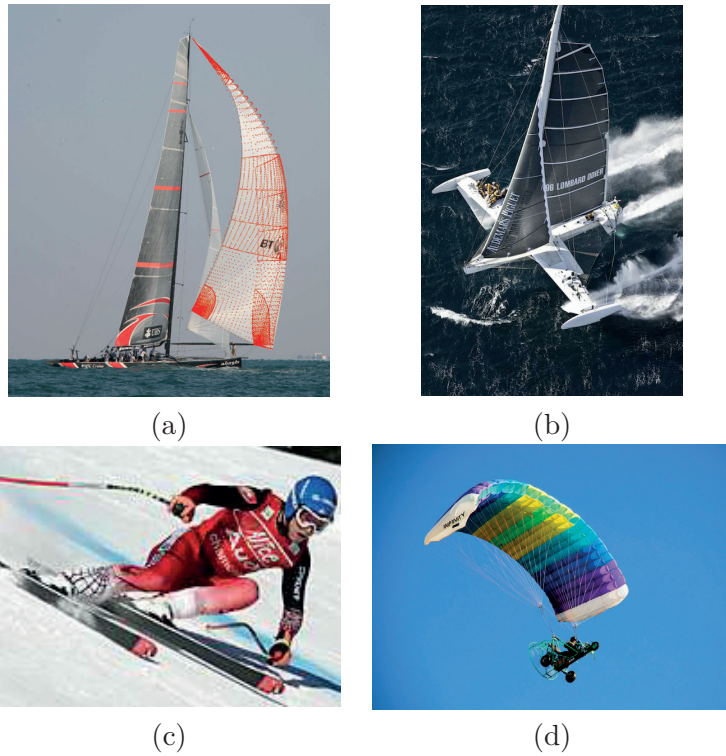


Figure 1.2: Many sport design tasks could benefit from 3-D shape recovery of deformable objects. (a,b) Sailors are interested in analyzing the shape of their own sails and that of their opponents. (c) Recovering the true deformations of skis during a race could help improve their design. (d) Knowing the shape of the parachute could help to maneuver it automatically.



Figure 1.3: An impact of a baseball colliding with a rigid cylinder serving as a bat was captured using a very high speed camera. Video-based 3-D modeling of ball deformations could help improving simulation results and designing protective equipments.

on take off and when encountering air turbulence. These deformations can be captured using a video camera mounted on-board of the plane and provide useful feedback to aerospace engineers in tailoring their designs. Similarly, the idea naturally extends to wind mills in Fig. 1.4 (b) and jet turbine blades such as those shown in Fig. 1.4 (c). As in the case of the baseball discussed above, this will allow the engineers to refine their simulation models to the point where their predictions exactly match reality.

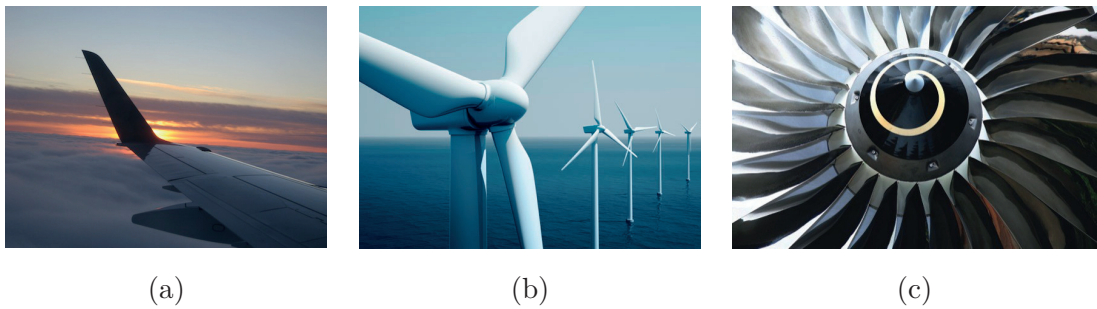


Figure 1.4: Image-based deformation measurements could help engineering design tasks. Recovering the true deformations of plane wings, wind mills, or jet turbine blades during flight could help improve their designs.

1.2.3 Entertainment and Marketing

Improved techniques for video-based modeling of deformable objects could bring great benefit to the entertainment and marketing industry. As illustrated by Fig. 1.5, there are already effective techniques [Pilet et al., 2008, White and Forsyth, 2006] for handling 2-D surface deformations for Augmented Reality purposes. They provide Augmented Reality solutions that can be used across a variety of applications ranging from live stage presentations to museums and theme parks attractions. They could also be used to draw virtual advertisement logos on athletes' or fashion models' clothes, thus avoiding the need to physically print them and making it easy to change them as necessity dictates. However, before that can happen, these techniques must be extended to full 3-D to better account for phenomena such as self-occlusions and self-collisions that become prevalent as the deformations become larger. Modeling object deformations in 3D is also required to obtain 3-D augmented reality in which 3-D virtual objects are placed in the real scene with respect to the deformable objects.

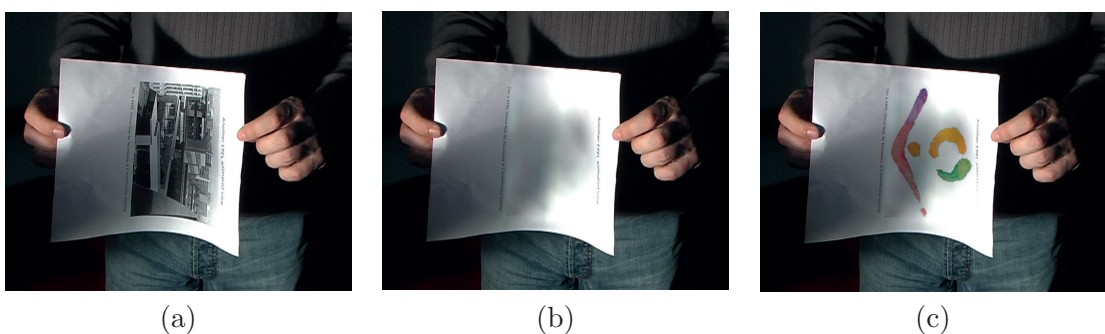


Figure 1.5: Demonstration of Augmented Reality on a deformable object in 2D. (a) A deformed piece of paper. (b) The paper texture has been virtually removed. (c) The paper appearance has been replaced by the properly deformed and re-shaded logo of a conference. Images are courtesy of Pilet et al. [2008].

The Computer Graphics community has spent a considerable effort on simulating and generating realistic object deformations for use in animation movies or video games. However, a lot of time would be saved if we could capture the animations directly from videos and transfer these deformations onto the animated objects. For example, to animate a graphics character and her clothes as realistically as possible, it would greatly help if the deformations of the clothes could be obtained by simply filming a real person performing some motions, reconstructing her clothes in 3-D, and re-applying the resulting deformations to the animated character.

1.2.4 Medical Imaging

More speculatively, 3-D modeling of deformable objects could have promising applications in the medical imaging field. As the techniques for surgery get more and more advanced, the current trend is to make surgery ever less invasive. Laparoscopic surgery is a such technique that only requires small cuts in the patient's skin. The small incisions only leave enough space for small cameras to be introduced into the patient's body. Hence, the surgeons do not have a direct view of their work. In such conditions, the resulting images are of poor quality and make the surgeons' work much harder. The 3-D shape reconstruction methods discussed in this thesis could be useful for the surgeons in several ways as illustrated in Fig. 1.6. Having a full 3-D representation of the organ's surface recovered from the images, or an augmented view of the organs could help the surgeons orient themselves more easily and improve their perception of where their surgery tools are with respect to the relevant surfaces [Bartoli et al., 2012, Maier-Hein et al., 2013]. Such shape measurements could also be useful for automated surgery robots, which are likely to be used widely in the next generation of surgical procedures.

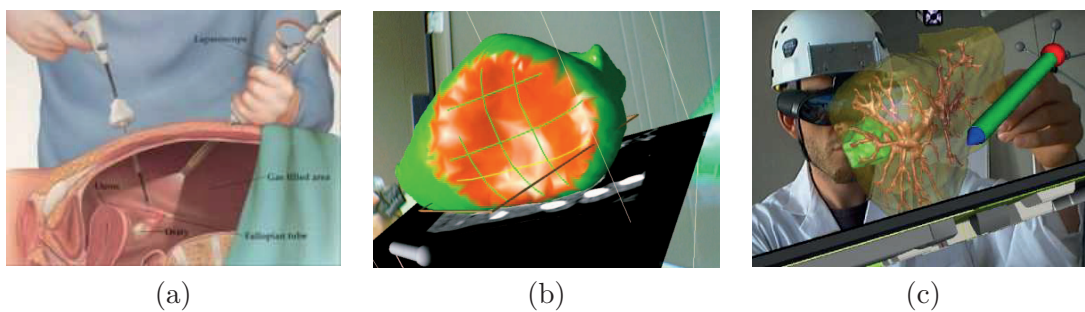


Figure 1.6: Video-based 3-D surface reconstruction can be applied to surgeries. (a) Schematic representation of a non-invasive laparoscopy surgery. (b) The 3-D reconstruction of the organ could be used to assist surgeons to orient themselves more easily and improve their perceptions. (c) Augmenting a scene with the reconstruction of a virtual liver. Images are courtesy of Reitingner et al. [2006].

1.3 Contributions

The goal of this work is to extend the scope of deformable object modeling to applications of 3-D modeling of deformable objects from video streams. Below we list our main contributions to the field.

1.3.1 Real-time Reconstruction and Tracking

To the best of our knowledge, prior to the work presented in this thesis, there had been no reports of real-time deformable object tracking system on mobile devices using nothing else than their built-in video cameras. The two main obstacles were the high dimensionality of the problem and the difficulty of dealing with noisy image information in such a high dimensional space. Many state-of-the-art approaches to template-based 3-D modeling of deformable objects require access to training data or material properties, which may be difficult to obtain or unknown. These methods may also require an estimate of the rigid transformation with respect to the template shape, which introduces extra variables and complicates the optimization. Other approaches tend to undesirably flatten non-planar objects in areas with only limited image information.

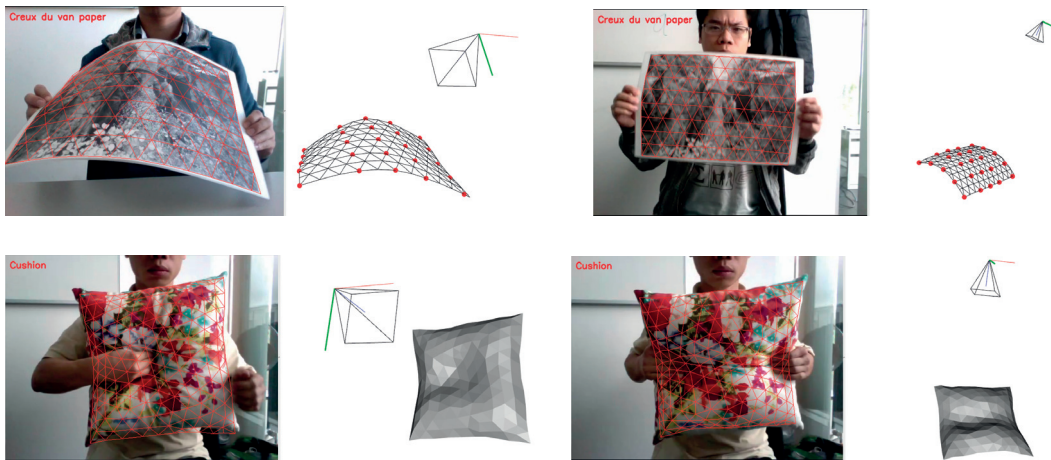


Figure 1.7: Real-time deformable surface reconstruction and tracking of a paper and a cushion.

Along with Jonas Ostlund, we therefore introduced a novel approach to regularizing potentially non-planar shapes and reducing the dimensionality of the reconstruction problem by extending the Laplacian formalism, which was first introduced in the Graphics Community. We could turn the monocular 3-D shape reconstruction of a deformable object given correspondences with a reference image into a much better-posed problem with far fewer degrees of freedom than the original one and without having to rigidly align the template shape to the shape to be recovered. This has proved key to achieving real-time performance while preserving both sufficient flexibility and robustness. We

further developed a real-time 3-D reconstruction and tracking system of deformable objects on mobile devices, that can very quickly reject outlier correspondences and accurately reconstruct the object shape in 3D. Frame-to-frame tracking is extensively exploited to track the object under difficult settings such as large deformations, occlusions, illumination changes, and motion blur. Some tracking scenarios are demonstrated Fig. 1.7.

1.3.2 Dense Image Registration and 3-D Reconstruction.

Depth ambiguities make monocular shape recovery highly under-constrained. Moreover, when the surface is partially occluded or has minimal texture, the problem becomes even more challenging because there is little or no useful information about large parts of it. When the surface is well-textured, correspondence-based methods have proved effective at solving this problem, even in the presence of occlusions. In contrast, when the surface lacks texture, dense pixel-level template matching should be used instead. Unfortunately, existing methods either are hampered by a narrow basin of attraction or require supervised learning to enhance robustness. Instead, we advocate template matching over robust dense features that relies on a pixel-wise relevancy score pre-computed for each frame. As shown in Fig. 1.8, our approach can handle occlusions and lack of texture simultaneously and obtain state-of-the-art results in recovering 3-D deformations of both well- or poorly-textured objects in the presence of occlusions.

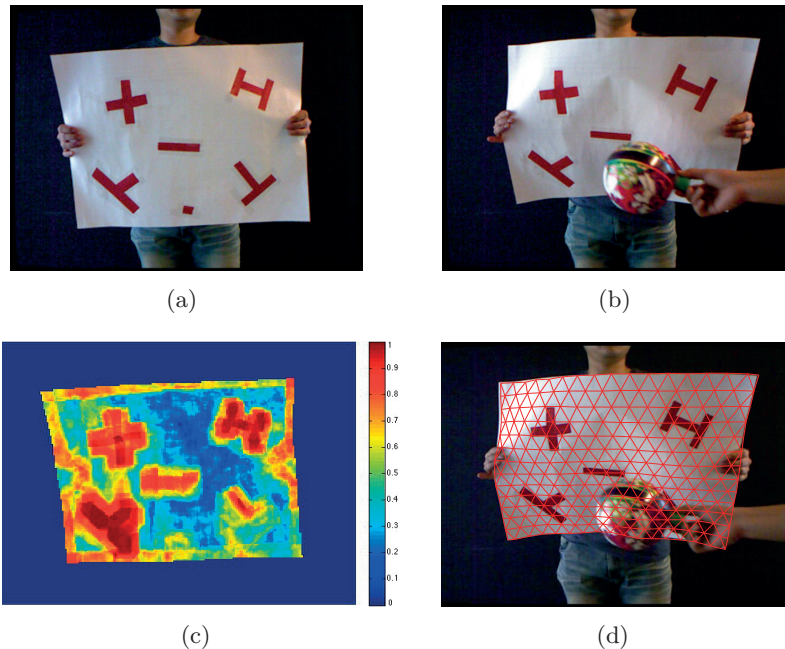


Figure 1.8: Tracking a sparsely textured surface in the presence of occlusion: (a) template image, (b) input image, (c) relevancy score, (d) surface tracking result with proposed framework.

Our contribution is a carefully-designed framework for tracking both well textured and sparsely textured deforming surfaces in videos in the presence of occlusions. Our method computes a relevancy score for each pixel, which is then used to weigh the influence of the image information from that pixel in the image energy cost function based on how informative and reliable that pixel is. In combination with a modified gradient-based pixel descriptors, our approach is robust to illumination change and has a much larger basin of convergence in the tracking context. The presented method favorably compares to standard cost functions used for handling occlusion, such as Mutual Information and M-estimators.

1.3.3 Applications

To study the extent of the utility of the proposed methods, we incorporated them into a number of real world applications of video-based 3-D modeling of deformable objects.

Existing algorithms for deformable object tracking and reconstruction usually focus on the deformable object alone. They can reliably handle deforming 3-D objects but do not take their environment into account. This is a severe limitation because, in the real world, rigid and deformable objects do not exist in isolation. Instead, they interact with each other and properly modeling these interactions is key to accurate reconstruction and understanding of the physical phenomena at play. Examples include balls being hit by rackets, bats, and clubs at ballgames and organs being prodded by surgical tools during operations. We explicitly attempt to address this problem by imposing real physical constraints between deformable and rigid objects. This results in 3-D reconstruction algorithms that are more accurate and can truly be deployed in real-world applications involving objects that interact with each other. For example, Fig. 1.9 demonstrates our results of video-based 3-D modeling of an impact between a ball and a rigid cylinder. We provide quantitative measurements for comparing impact simulations against video-based measurements to assess which physics-based predictions most accurately describe a physical quantity of interest and to improve the models.

Together with geometric registration, being able to estimate environment illumination would allow us to realistically render virtual objects with appropriate lighting into a real scene. It would drastically improve augmented reality experiences because virtual objects seamlessly blend with real objects under the real lighting condition. We introduce a non-invasive method to compute the environment illumination for augmented reality purposes on deformable objects. We rely on our markerless real-time deformable object tracking system to collect lighting cues and estimate the environment illumination, also in real-time, represented by a set of spherical harmonics. As a result, we obtain real-time augmented reality experiences on deformable objects with realistic lighting, demonstrated in Fig. 1.10.

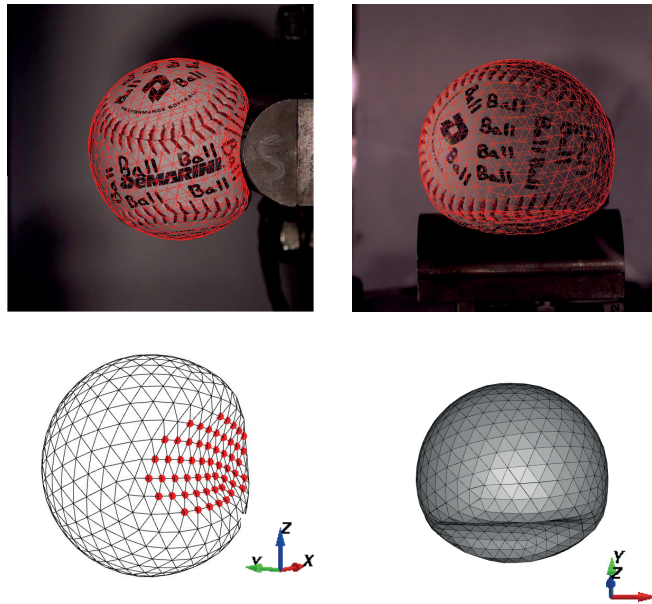


Figure 1.9: Studying the deformation behaviors of a ball colliding with a rigid cylindrical object. Top row: Reprojection of the reconstructed mesh on the side and top views. Bottom row: The reconstructed mesh seen from different viewpoints rotated slightly from the side and top views, respectively, to show the impact surface. The red dots denote vertices touching the cylinder.

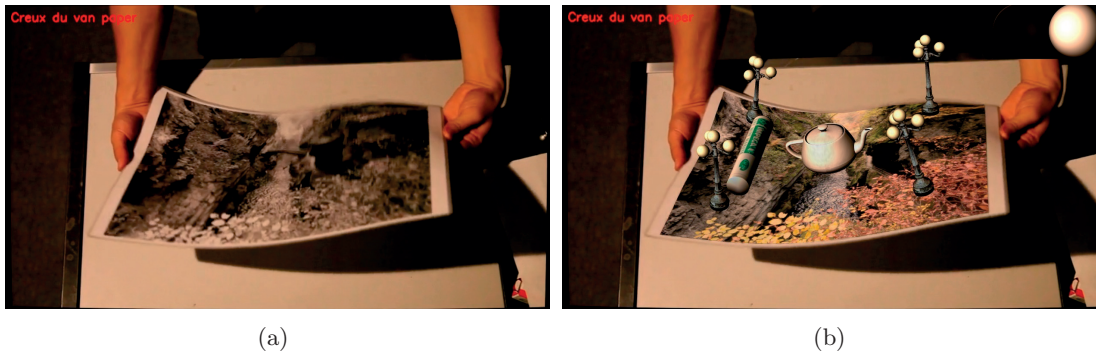


Figure 1.10: Augmented reality with environment illumination estimation. (a) An original input frame in the sequence. (b) The surface is re-textured and virtual objects are illuminated realistically using the recovered illumination.

We present an Augmented Reality coloring book App, see Fig. 1.11, that provides a bridge between animated cartoon characters and their colored drawings. Children color characters in a printed coloring book and inspect their work using a consumer-grade mobile device, such as a tablet or a smart phone. The drawing is detected and tracked, and the video stream is augmented with an animated 3-D version of the character that is textured according to the child's coloring. Accomplishing this goal requires addressing several challenges. First, the 2-D colored drawing provides texture information only about

the visible portions of the character. Texture for the occluded regions, such as the back side of the character, must be generated. Second, the pages in an actual printed coloring book are not flat but exhibit curvature due to the binding of the book. As a result, tracking algorithms and texture capture must be robust to page deformation in order to properly track the drawings and lift texture from the appropriate 2-D regions. Our coloring book App addresses each of these technical challenges. We present a texturing process that applies the captured texture from a 2-D colored drawing to both the visible and occluded regions of a 3-D character in real time. We use our deformable surface tracking and 3-D shape recovery for the drawings. We also present a content creation pipeline to efficiently create the 2-D and 3-D content used in our coloring book App.

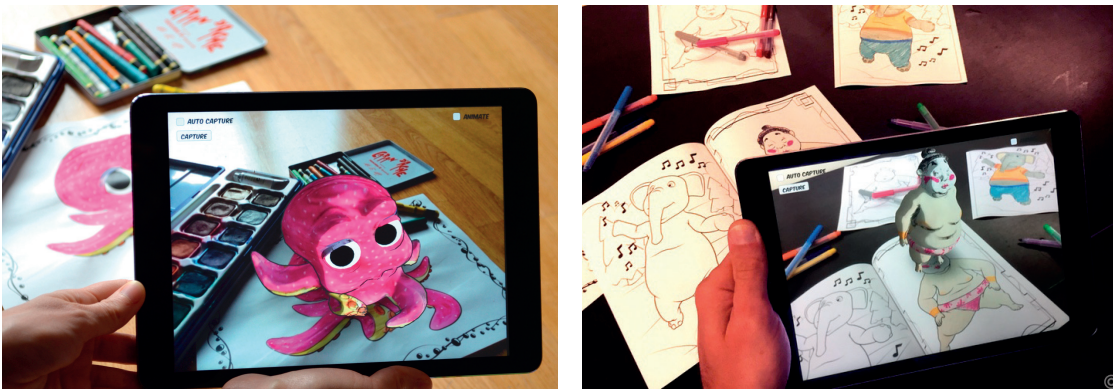


Figure 1.11: Coloring book application scenarios with different drawings and colorings.

1.4 Thesis Outline

Chapter 2 discusses the literature on recovering 3-D shape of deformable objects in monocular images and videos. To be more general, we first review the work in Non-Rigid Structure-from-Motion (NRSfM) in which a template image and a template shape of the deformable object are not required. We then discuss the template-based approaches to 3-D reconstruction of deformable objects from monocular images. We also talk about the dense image registration literature that relates to our framework of simultaneous dense image registration and 3-D reconstruction of poorly-textured and occluded deformable objects.

Chapter 3 introduces our linear formulation of the template-based monocular 3-D shape recovery of deformable objects along with a Laplacian approach to regularizing the solution and linearly parameterizing the mesh with respect to a set of a few control points. We present our reconstruction results on datasets of both planar and non-planar deformable objects and compare them to the state-of-the-art methods in the same category. We then introduce our fast and efficient 2-D outlier rejection strategy and a simplified formulation of the reconstruction problem, which together allows us to reconstruct and track deformable objects in real-time. We discuss our tracking techniques to make the

Chapter 1. Introduction

algorithm more robust, more efficient, and be able to deal with large deformations and occlusions.

Chapter 4 introduces our dense registration and reconstruction framework for poorly-textured and occluded surfaces. We first present our template matching based approach for simultaneous deformable surface image registration and 3-D shape reconstruction. We then discuss a number of methods to make the registration robust to sparse textures and occlusions, such as pixel descriptors and similarity functions. Next, we introduce our relevancy scores which we use to weigh the image energy per pixel, allowing the registration to deal with occlusions and poorly-textured objects. We finally compare our method with other related feature-based and dense matching-based methods on existing datasets and newly created datasets.

Chapter 5 discusses some real world applications of our methods in video-based 3-D modeling of deformations. We first present our approach to reconstructing deformable objects in interactions with rigid ones. To enforce the presence of rigid objects in reconstructing deformable ones, we first compute the 6-D pose of the rigid objects by using a modified edge-based or render-based dense matching object pose estimation. We present in detail our study of ball-bat collisions whose results are compared quantitatively to simulation-based methods. Next, we introduce our real-time object recognition algorithm which automatically selects the template for our template-based tracking and reconstruction algorithm. After that, we present an efficient environment lighting estimation modeled by spherical harmonics. The chapter concludes with our coloring book application on mobile devices. We discuss our content creation pipeline, tracking, texture generation, and augmentation of 3-D animated characters.

Chapter 6 concludes this thesis, summarizes what has been achieved, and discusses potential future research directions.

1.5 Publications

This thesis covers the following peer-reviewed accepted or pending publications:

- Lloyd Smith, Derek Nevins, Dat Tien Ngo, Pascal Fua. Measuring the accuracy of solid sport ball impact simulations. Submitted to Journal of Sports Engineering, 2015.
- Dat Tien Ngo, Sanghyuk Park, Anne Jorstad, Alberto Crivellaro, Chang Yoo, Pascal Fua. Dense image registration and deformable surface reconstruction in presence of occlusions and minimal texture. International Conference on Computer Vision (ICCV), 2015, Santiago, Chile.
- S. Magnenat, T.D. Ngo*, F. Zund, M. Ryffel, G. Noris, G. Rothlin, A. Marra,

M. Nitti, P. Fua, M. Gross, R. Sumner*. Live Texturing of Augmented Reality Characters from Colored Drawings. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, Proceedings ISMAR 2015. * *Corresponding authors. Best Paper Award Honorable Mention.*

- Dat Tien Ngo, Jonas Ostlund, Pascal Fua. Template-based Monocular 3D Shape Recovery using Laplacian Meshes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.
- Jonas Ostlund, Aydin Varol, Dat Tien Ngo, Pascal Fua. Laplacian meshes for monocular 3D shape recovery. *European Conference on Computer Vision (ECCV)*, 2012, Florence, Italy.

2 Related Work

Recovering the 3-D shape of deformable objects from monocular images and videos has been an active area of research in Computer Vision for many years. Although humans can reasonably infer the shape of deformable objects, it remains a difficult problem for computers because of ambiguities inherent to monocular shape recovery, especially for deformable objects. This chapter briefly discusses existing approaches to video-based deformable object 3-D modeling and points out the similarities and differences with our methods presented in this thesis.

Techniques in video-based 3-D modeling of deformable objects can be roughly classified into two categories: Non-Rigid Structure-from-Motion approaches and template-based methods. We first briefly overview the methods in Non-Rigid Structure from Motion (NRSfM) in which a template image and a template shape of the deformable object are not required. The reconstruction is performed by solely using a sparse or dense set of pixels tracked across a video sequence. We then review more related methods in template-based 3-D shape recovery of deformable objects, in which a 3-D template shape and a template image of the object are available. We also discuss image registration algorithms which relate to our methods of densely registering images and reconstructing deformable objects in 3D.

2.1 Non-rigid Structure from Motion

Non-Rigid Structure-from-Motion methods do not rely on a template image and work on a sequence of images of the same surface with different deformations in a video sequence. They rely on 2-D tracking results of a sparsely- or densely-sampled set of points in the whole video sequences. The goal is to recover both camera poses and 3-D locations of image features in every frame of the image sequence. In many practical applications in which a template cannot be acquired in advance, these methods become essential. In this section, we briefly discuss their common formulations and the most representative

existing attempts to solve the problem.

A pioneering approach to NRSfM was initially introduced by [Bregler et al., 2000] extending the Tomasi-Kanade factorization for rigid reconstruction [Tomasi and Kanade, 1992] for weak projective cameras. The fundamental assumption in [Bregler et al., 2000] is that non-rigidly deforming shapes can be linearly modeled by a low dimensional subspace of unknown smooth basis shapes. This assumption is particularly true for articulated objects since object parts move rigidly and connect to each other at joints. This assumption is also reasonable for general deformable objects because they do not deform arbitrarily from one frame to the next and motions of 3-D points are highly correlated in space and time. Hence, the solution should lie in a much lower dimensional space than the one spanning all possible 3-D shapes in each frame. The shape at frame t can be expressed as:

$$\mathbf{S}^{(t)} = \sum_{k=1}^K \alpha_k^{(t)} \mathbf{S}_k \quad , \quad (2.1)$$

in which \mathbf{S}_k are a set of fixed 3-D shapes. Provided a number of outlier-free 2-D tracked correspondences across a video sequence, one can build a linear system encoding the camera projections of 3-D locations of image features.

$$\mathbf{W} = \mathbf{R} \cdot \mathbf{\Omega} \cdot \mathbf{S} \quad , \quad (2.2)$$

where \mathbf{W} is the image observation matrix whose elements are 2-D image locations, \mathbf{R} is a matrix containing camera pose parameters, $\mathbf{\Omega}$ contains linear deformation weights $\alpha_k^{(t)}$ to be estimated for each frame, and fixed but unknown basis shapes are stacked in matrix \mathbf{S} . The NRSfM becomes a trilinear matrix factorization problem over camera poses, shape coefficient vectors, and shape basis vectors. In fact, the shape coefficient factors are usually incorporated with the camera poses into a motion matrix $\mathbf{M} = \mathbf{R} \cdot \mathbf{\Omega}$, making the problem become a bilinear matrix factorization. The rank of the observation matrix \mathbf{W} is at most $3K$ and singular value decomposition can be used to recover it up to a multiplication with a $3K \times 3K$ invertible matrix \mathbf{Q} . Bregler’s formulation has inspired numerous approaches to NRSfM that try to resolve the ambiguities and degeneracies by proposing different optimization schemes [Paladini et al., 2009, Delbue et al., 2010, Akhter et al., 2008, Dai et al., 2012, Garg et al., 2013a] and the use of generic priors [Bartoli et al., 2008, Torresani et al., 2008, Delbue, 2008, Fayad et al., 2010, Taylor et al., 2010].

When the matrix \mathbf{W} is factorized into two matrices \mathbf{M} and \mathbf{S} , there are two types of constraints needing to be imposed in the metric upgrade \mathbf{Q} . The first type of constraints ensures that every rotation matrix \mathbf{R}_i in \mathbf{M} is a Stiefel matrix. The second type of constraints enforces the repetitive block structure of the same camera matrix \mathbf{R}_i in \mathbf{M} . There are no general closed-form solutions for both constraints. Paladini et al. [2009]

proposed an iterative method which is capable to enforce both constraints. They solve the matrix factorization of \mathbf{W} as an alternating least-squares problem where at each step the motion \mathbf{M} and basis shapes \mathbf{S} matrices are optimized separately keeping the other one fixed. The solution is iteratively projected onto a metric motion manifold to satisfy the constraints until convergence.

While most NRSfM approaches represent the shape with a linear subspace models, Akhter et al. [2008] introduced a different formulation. They proposed a dual view on the orthographic factorization problem, in which the roles of shapes and trajectories are swapped. The observation is that the trajectory of each 3-D point over time often lies in a low dimensional space. Instead of reconstructing the whole shape at each time instant, the trajectory over the whole sequence of each 3-D point is estimated. These trajectories can be described as a linear combination of K basis trajectories. In practice, the method assumes that the basis trajectories are known and can be generated from the Discrete Cosine Transforms. Given this formulation, NRSfM can be re-written as the factorization problem $\mathbf{W} = \mathbf{R} \cdot \Theta \cdot \mathbf{A}$ in which fewer unknowns need to be determined because the trajectory bases Θ are fixed.

By contrast, Taylor et al. [2010] proposed a piecewise approach that uses the local rigidity assumption of the deforming shape, in which neighboring points are assumed to be close enough to justify a rigid model. Delaunay triangulations of the image observations are performed to identify a set of candidate triangles. Each of these rigid triangles is reconstructed independently using a linear algorithm to form a triangle soup. If the rigid reconstruction obeys the rigidity assumption within a pre-defined threshold on the reprojection errors and if the triangle is sufficiently regular, it is accepted as part of the total reconstruction. The triangles in the resulting triangle soup are then aligned together to provide a smooth 3-D surface. A disambiguation step is needed to resolve the translation and reflection ambiguities in a greedy way. This step allows the method to reconstruct changing-topology shapes such as a tearing paper. On the other hand, as the triangles are minimal, the reconstruction could be relatively noisy.

In a similar spirit, Fayad et al. [2010] proposed to use a more descriptive deformation model within a piecewise framework. Local patches are represented by a quadratic deformation model involving three modes of deformation: *linear* which allows shearing and stretching, *quadratic* which allows bending, and *mixed term* which allows twisting. The quadratic model for each patch is optimized over the sequence independently using temporal smoothness priors. The patches are then assembled to give a smooth 3-D surface using overlapping points to enforce global consistency.

Recently, Dai et al. [2012] showed that other than using Bregler’s basic low rank shape condition together with camera orthonormality constraints, their method can unambiguously solve the factorization problem of NRSfM. Instead of using a pre-defined number of basis shapes and time-varying coefficients, they directly imposed the low rank shape

constraints on the matrix of time-varying shapes by minimizing its trace norm, which is a tight relaxation of matrix rank minimization.

Garg et al. [2013a] adopted the trace norm minimization from [Dai et al., 2012] to estimate the low rank non-rigid shapes. It, however, can densely compute a depth map of the non-rigid shape given a dense optical flow across the video as the input. It formulates the problem in a variational framework with total variation regularization. The optimization is solved by using alternation between motion matrix, parameterized by quaternions, and shape matrix with proximal splitting techniques to decouple the trace norm and TV regularization. Strong results were obtained for faces, beating hearts, moving torsos. Although the algorithm reconstructs dense per-pixel deforming models, it is a batch process that requires multi-frame optical flow over the entire sequence as an input. Thus, it has a low frame rate, even using parallel processing on the GPU.

The NRSfM methods discussed in this section do not require a template of the deforming object and usually operate in a batch mode on a whole video. The problem becomes a lot easier if a template is assumed to be known. We have briefly reviewed the NRSfM methods to give a broad overview of video-based deformable object 3-D modeling. In the next section, we review more closely related methods to ours in template-based 3-D modeling of deformable objects from images and videos.

2.2 Template-based 3-D Deformable Object Reconstruction

Even when a template image of the object in a different but known configuration is available, reconstructing the 3-D shape of a non-rigid object from monocular images is still a severely under-constrained problem. This is the problem we discuss in this section, as opposed to recovering the 3-D shape from sequences as in monocular Non-Rigid Structure from Motion methods discussed in the previous section.

Given a template image, one can establish point correspondences between the input and template images, or in other words, the correspondences between the input image points and 3-D points on the object surface. However, by only enforcing the image reprojection constraints, the problem is severely ill-posed because there are many 3-D candidate shapes having the same camera projection. When image noises are present, it becomes more problematic and is impossible to identify the deforming shape without additional priors or constraints on the object deformations.

One class of the approaches suggested to alleviate the ambiguities and improve the reconstruction accuracy is to impose temporal consistency of 3-D deforming shapes in a tracking framework. They assume that the object deforms smoothly from the current frame to the next. By penalizing the displacements of 3-D reconstructed shapes between

2.2. Template-based 3-D Deformable Object Reconstruction

consecutive frames, the reconstruction problem can be made well-posed [Salzmann et al., 2007b]. Another way to impose temporal consistency is to prevent the orientation of mesh edges to change excessively from one frame to the next [Salzmann et al., 2007a]. These constraints can handle generic object deformations with different material properties and without introducing unnecessary smoothness. However, these constraints are limited to a tracking context, which requires an accurate initialization and a sequence of images. Alternative geometric constraints such as smoothness [Salzmann et al., 2008b, Shaji et al., 2010], developable [Gumerov et al., 2004, Perriollat and Bartoli, 2007, Taddei and Bartoli, 2008, Perriollat and Bartoli, 2013], or isometry [Salzmann et al., 2008a, Bartoli et al., 2012, Salzmann and Fua, 2011, Ostlund et al., 2012, Brunet et al., 2014, Collins and Bartoli, 2015, Parashar et al., 2015] have been proposed to solve the reconstruction for single input images. We discuss the most related methods below.

Given point correspondences established between the template and input images, [Bartoli et al., 2012, Chhatkuli et al., 2014] proposed a two-step algorithm to solve for the deforming shape in the input image. The method first computes a 2-D warp between the images and then infers a 3-D shape from it exploiting the isometric deformation constraint. Bartoli et al. [2012] were the first who derived an analytical solution to the problem of template-based deformable surface 3-D reconstruction. It forms a powerful tool and allows one to prove the existence of solutions. The reconstruction can be done in closed form and pointwise, and could potentially run in real-time. However, the quality of the recovered 3-D shape then depends on the quality of the 2-D warp, which does not necessarily account for the 3-D nature of the deformations and the constraints it imposes.

An alternative is therefore to go directly from correspondences to 3-D shape by solving an ill-conditioned linear-system [Salzmann and Fua, 2010], which requires the introduction of additional constraints to make it well-posed. The most popular ones involve preserving Euclidean or Geodesic distances as the surface deforms and are enforced either by solving a convex optimization problem [Brunet et al., 2014, Ecker et al., 2008, Salzmann et al., 2007a, Shen et al., 2009, Moreno-Noguer et al., 2010, Perriollat et al., 2011, Salzmann and Fua, 2011] or by solving in closed form sets of quadratic equations [Salzmann et al., 2008a, Moreno-Noguer et al., 2009]. The latter is typically done by linearization, which results in very large systems and is no faster than minimizing a convex objective function, as is done in [Salzmann and Fua, 2011] which has been shown to be an excellent representative of this class of techniques. The results can then be improved by imposing appropriate physics-based constraints [Malti et al., 2013] via non-linear minimization, but that requires some knowledge of the physical properties that may not be available.

The complexity of the problem can be reduced using a dimensionality reduction technique such as Principal Component Analysis (PCA) to create morphable models [Cootes et al., 1998, Blanz and Vetter, 1999, Dimitrijević et al., 2004], modal analysis [Moreno-Noguer et al., 2009, 2010], Free Form Deformations (FFDs) [Brunet et al., 2010, 2014], or 3-D warps [Delbue and Bartoli, 2011]. One drawback of PCA and modal analysis is

that it requires either training data or sufficient knowledge of the surface properties to compute a stiffness matrix, neither of which may be forthcoming. Another is that the modal deformations are expressed with respect to a reference shape, which must be correctly positioned. This makes it necessary to introduce additional rotation and translation parameters into the computation. This complicates the computations because the rotations cannot be treated as being linear unless they are very small. The FFD approach [Brunet et al., 2010, 2014] avoids these difficulties and relies on parameterizing the surface in terms of control points. However, its affine-invariant curvature-based quadratic regularization term is designed to preserve local structures and planarity. For non-planar surfaces, it tends to flatten the surface in areas with only limited image information. By contrast, our approach presented in Chapter 3 naturally handles non-planar reference surfaces, tends to preserve curvature, and its control vertices can be arbitrarily placed. This makes it closer to earlier ones to fitting 3-D surfaces to point clouds that also allow arbitrary placement of the control points by using Dirichlet Free Form Deformations [Ilić and Fua, 2002] or, more recently, sets of affine transforms [Jacobson et al., 2011]. These approaches, however, also require regularization of the control points if they are to be used to fit surfaces to noisy data.

None of these dimensionality reduction methods allows both orientation-invariant and curvature-preserving regularization. To do both simultaneously, we took our inspiration from the Laplacian formalism presented in [Sorkine et al., 2004] and the rotation invariant formulation of [Sumner and Popovic, 2004], which like our method in Chapter 3 involves introducing virtual vertices. In both these papers, the mesh Laplacian is used to define a regularization term that tends to preserve the shape of a non-planar object.

In [Botsch et al., 2006], it is shown that explicitly adding the virtual vertices is not necessary to achieve similar or even better results. In our work in Chapter 3, we go one step further by not only introducing a rotation invariant regularization term expressed as a linear combination of the vertex coordinates but also showing that these coordinates can themselves be written as a linear function of those of a subset of control vertices while preserving rotation invariance. Furthermore, in [Sumner and Popovic, 2004], the regularization term involves minimizing the magnitude of the local deformations, which favors small deformations. By contrast, we propose in Chapter 3 an approach that only penalizes curvature changes and can accommodate arbitrarily large deformations.

Recently, [Collins and Bartoli, 2015] proposed a real-time shape from template framework in a deformable object tracking context. They proposed various techniques to make the tracking fast and robust. At its core is the render-based block matching algorithm that establishes highly dense correspondences. These dense correspondences in a large number allow the deformable object tracking algorithm to deal with difficult image acquisition conditions. The method obtains very interesting results in tracking non-planar deformable objects such as boxes, bottles, rugby balls. In fact, this method shares some similarities with our real-time deformable object tracking system presented in Chapter 3 such as the

linear image energy and the Laplacian-based regularization term. It extensively uses parallel processing on GPUs and works, however, in a tracking context only.

Another interesting work worth to be mentioned is [Cashman and Fitzgibbon, 2013] that builds a 3-D morphable model of a deformable object class from a collection of 2-D images. Given a rough 3-D rigid template shape of the object as an initial estimate of the mean shape, a few manually provided 3D-2D correspondences for each image, and possibly a rough rigid pose of the template shape in each image, the method could fit unknown 3-D morphable shapes to the silhouettes of different object instances while learning the basis shapes, camera parameters, and morphable shape coefficients.

2.3 Dense Image Registration

The most successful current approaches to template-based deformable object 3-D reconstruction generally rely on finding feature point correspondences as discussed in the previous section. It is because they are robust to lighting changes and occlusions. Unfortunately, these methods tend to break down when attempting to reconstruct sparsely or repetitively textured surfaces, since they rely on a fairly high number of correct matches.

Pixel-based techniques are able to overcome some limitations of local feature matching, since they reconstruct surfaces based on a global, dense comparison of images. On the other hand, some precautions must be taken to handle occlusions, lighting changes, and noise.

Since the Lucas-Kanade image registration algorithm [Lucas and Kanade, 1981] was proposed, dense image alignment has become a widely-used techniques in Computer Vision. Especially, it has recently undergone a regain of interest for the accuracy and robustness, thanks to the growing power of computing devices. Its applications range from optical flow, tracking, and medical image registration to mosaic construction, face coding, and rigid object pose estimation.

In the context of 2-D image dense registration, Garg et al. [2013b] recently proposed an approach to computing optical flow from a template to each of the images in a long video sequence of a non-rigid object. It relies on a low-rank motion basis of 2-D point trajectories to significantly reduce the dimensionality of the problem and to impose temporal consistency of multi-frame optical flow. The subspace constraint is formulated as an additional soft constraint within a variational framework with a total energy functional consisting of a brightness constancy term and a spatial regularization term. The optimization can be solved efficiently by using a decoupling scheme of the data and regularization terms.

Braux-Zin et al. [2013] later introduced an optical flow estimation method that combines direct and feature-based matching costs in a unified variational framework. It makes use

of a set of putative matches of well- or weakly-localized features to drive optical flow computation out of local minima. Mismatches are filtered by Geman-McClure estimator on the fly. External occlusions are handled by simple thresholding and self-occlusions are detected to further improve the optical flow estimation.

In the context of registering 2-D images of deformable surfaces, [Pizarro and Bartoli, 2012, Pilet et al., 2008] proposed robust methods based on global or local smoothness motion field priors for detecting a well-textured deformed surface in an image using a set of wide-baseline feature point correspondences. The image warping function initialized by feature points is further refined by a pixel-based direct registration [Pizarro and Bartoli, 2012]. Similarly, [Gay-Bellile et al., 2010] densely registers images of deformable surfaces in 2D and shrinks the image warps in self-occluded areas. However, since the registration is in 2D, only part of the self occlusion is recovered.

In the context of deformable object modeling, [Malti et al., 2011] uses a direct photometric cost and estimates a visibility mask on the reconstructed surface, but only textured surfaces and self-occlusions are handled. [Bartoli et al., 2012] proves that an analytical solution to the 3-D surface shape can be derived from a 2-D warp. However, the surface shape in self-occluded areas is undefined. [Collins and Bartoli, 2014] registers local image patches of feature point correspondences to estimate their depths, and geometric constraints are imposed to classify incorrect feature point correspondences. In contrast to these local depth estimations, we want to develop a method that reconstructs surfaces globally in order to be more robust to noise and outliers.

Other recent approaches employ supervised learning for enhancing performance [Salzmann, 2013, Varol et al., 2012]. In [Salzmann et al., 2008b], strong results are achieved with poorly textured surfaces and occlusion by employing trained local deformation models, a dense template matching framework using Normalized Cross Correlation (NCC) [Scandaroli et al., 2012] and contour detection. Our proposed framework in Chapter 4 manages to achieve similar performance without requiring any supervised learning step, while the use of robust, gradient-based dense descriptors recently proposed in [Crivellaro and Lepetit, 2014] avoids the need to explicitly detect contours.

Other techniques employed for dealing with occlusions and noise, such as Mutual Information (MI) [Damen et al., 2012, Dowson and Bowden, 2006, Panin and Knoll, 2008, Viola and Wells, 1997] and robust M-estimators [Arya et al., 2007] are studied explicitly in our context, and found to be successful only up to a point in the context of deformable object modeling.

Our method presented in Chapter 4 is in the same spirit as that of [Pilet et al., 2007], where a template matching approach is employed and a visibility mask is computed for the pixels lying on the surface. However, in this earlier work a very good initialization from a feature point-based method is required in order for its EM algorithm to converge.

In addition to the geometrical degrees of freedom of the surface, local illumination parameters are explicitly estimated in [Hager and Belhumeur, 1998, Silveira and Malis, 2007]. This requires a reduced deformation model for the surface to keep the size of the problem tractable.

In our proposed framework presented in Chapter 4, we achieve good performance without the need to explicitly estimate any illumination model, so that an accurate geometric model for the surface can be employed. Furthermore, rather than estimating a simple visibility mask as is often done in many domains such as stereo vision [Strecha et al., 2006], face recognition [Zhou et al., 2009], or pedestrian detection [Wang et al., 2009], we employ a real-valued pixel-wise relevancy score, penalizing at the same time pixels with unreliable information originating both from occluded and low-textured regions. Our method has a much wider basin of convergence and we can track both well and poorly textured surfaces without requiring initialization by a feature point-based method.

A recent work [Yu et al., 2015] can capture the deformations of generic shapes with the depth being estimated densely, per-pixel, and by a using direct method. Their dense and direct formulation is similar to ours with a photometric data term, a regularization term, and a deformation constraint term. However, they use sensitive pixel intensity for photometric data term and do not explicitly handle occlusions and poorly-textured objects.

3 Real-time 3-D Reconstruction and Tracking

Rigid object tracking in 3D has reached the maturity level to be widely used for real-time augmented reality purposes. Tools such as Vuforia [Vuforia, 2015] or Metaio [Metaio, 2015] provide readily-usable solutions to accelerate the process of creating augmented reality applications based on natural rigid markers. By contrast, tracking non-rigid objects still lags behind, partly due to the high number of degrees of freedom to be recovered, which limits the range of objects that can be handled. This limitation motivates the research in developing a real-time tracking system for deformable objects. In this chapter, we present our method to build a real-time deformable surface tracking system on mobile devices. It opens the door to a wide range of real-world applications.

3.1 Foreword

To the best of our knowledge, prior to the work presented in this chapter, there had been no reports describing a real-time deformable object tracking system on mobile devices using nothing else than their built-in video cameras. The two main difficulties were the high dimensionality of the problem and how to deal with noisy image information in such a high dimensional space of solutions. Along with Jonas Ostlund, we contribute a novel approach to regularizing and reducing the dimensionality of the reconstruction problem by extending the Laplacian formalism, which was first introduced in the Graphics Community [Sorkine et al., 2004, Sumner and Popovic, 2004, Botsch et al., 2006].

Our novel regularization term is designed to minimize curvature changes from the potentially non-planar reference shape. Using this regularization term, we can express all vertex coordinates as linear combinations of those of a small number of control vertices. The monocular 3-D shape reconstruction problem could be turned into a much better-posed one with far fewer degrees of freedom than the original problem. This considerably increases computational efficiency at no loss in reconstruction accuracy. We further propose a new robust outlier rejection mechanism which is formulated in 2D to

avoid depth ambiguities. To gain speed while not sacrificing accuracy, we reformulate the reconstruction energy function with the use of soft isometric deformation constraints instead of complicated inextensibility constraints. Frame-to-frame tracking is extensively exploited to gain frame rate and robustness. We only apply the feature detection and matching or active search periodically to retrieve back lost tracked points and accumulate good correspondences. The proposed techniques together allow robust real-time tracking of deformable objects.

In the following sections, we first formulate the 3-D reconstruction problem as an ill-posed linear system. We then introduce our Laplacian-based rotation-invariant regularization term for both planar and non-planar objects. We then present our linear parameterization of the mesh with respect to a small set of control points. Next, we discuss our quantitative reconstruction results and finally our problem formulation that allows it to run in real-time on mobile devices.

3.2 Linear Problem Formulation

As in [Salzmann and Fua, 2010], we start by showing that given point correspondences between a reference image in which the 3-D shape is known and an input image, recovering the new shape in this image amounts to solving a linear system.

Let \mathbf{v}_i be the 3-D coordinates of the i^{th} vertex of the N_v -vertex triangulated mesh representing the surface, \mathbf{K} be the intrinsic camera matrix, \mathbf{p} be a 3-D point lying on facet f . One can represent \mathbf{p} in the barycentric coordinates of f :

$$\mathbf{p} = \sum_{i=1}^3 b_i \mathbf{v}_{f,i}, \tag{3.1}$$

where $\{\mathbf{v}_{f,i}\}_{i=1,2,3}$ are the three vertices of f . The fact that \mathbf{p} projects to the 2-D image point (u, v) can be expressed by

$$\mathbf{K}(b_1 \mathbf{v}_{f,1} + b_2 \mathbf{v}_{f,2} + b_3 \mathbf{v}_{f,3}) = k \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \tag{3.2}$$

where k is the homogeneous component. Since k can be expressed in terms of the vertex coordinates using the last row of the above equation, we can rewrite Eq. 3.2 to be

$$[b_1 \mathbf{H} \quad b_2 \mathbf{H} \quad b_3 \mathbf{H}] \begin{bmatrix} \mathbf{v}_{f,1} \\ \mathbf{v}_{f,2} \\ \mathbf{v}_{f,3} \end{bmatrix} = 0, \tag{3.3}$$

with

$$\mathbf{H} = \mathbf{K}_{2 \times 3} - \begin{bmatrix} u \\ v \end{bmatrix} \mathbf{K}_3, \quad (3.4)$$

where $\mathbf{K}_{2 \times 3}$ are the first two rows, and \mathbf{K}_3 is the third one of \mathbf{K} .

Given n correspondences between 3-D reference surface locations and 2-D image points, we obtain $2n$ linear equations which can be jointly expressed by a linear system

$$\mathbf{M}\mathbf{x} = \mathbf{0}, \text{ where } \mathbf{x} = \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{N_v} \end{bmatrix}. \quad (3.5)$$

and \mathbf{M} is a matrix obtained by concatenating the $[b_1\mathbf{H} \ b_2\mathbf{H} \ b_3\mathbf{H}]$ matrices. In practice, the sum of squares of successive elements of the vector $\mathbf{M}\mathbf{x}$ are squared distances in the direction parallel to the image-plane between the line-of-sight defined by a feature point and its corresponding 3-D point on the surface. A solution of this system defines a surface such that 3-D feature points that project to specific locations in the reference image project at corresponding locations in the input image. Solving this system in the least-squares sense therefore yields surfaces, up to a scale factor, for which the overall reprojection error is small.

The difficulty comes from the fact that, for all practical purposes, \mathbf{M} is rank deficient as shown in Fig. 3.2(a), with at least one third of its singular values being extremely small with respect to the other two thirds even when there are many correspondences. This is why the inextensibility constraints, as mentioned in Section 3.1, will be introduced.

A seemingly natural way to address the issue of high dimensionality is to introduce a linear subspace model and to write surface deformations as linear combinations of relatively few basis vectors. This can be expressed as

$$\mathbf{x} = \mathbf{x}_0 + \sum_{i=1}^{N_s} w_i \mathbf{b}_i = \mathbf{x}_0 + \mathbf{B}\mathbf{w}, \quad (3.6)$$

where \mathbf{x} is the coordinate vector of Eq. 3.5, \mathbf{B} is the matrix whose columns are the \mathbf{b}_i basis vectors typically taken to be the eigenvectors of a stiffness matrix, and \mathbf{w} is the associated vectors of weights w_i . Injecting this expression into Eq. 3.5 and adding a regularization term yields a new system

$$\begin{bmatrix} \mathbf{M}\mathbf{B} & \mathbf{M}\mathbf{x}_0 \\ \lambda_r \mathbf{L} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ 1 \end{bmatrix} = \mathbf{0}, \quad (3.7)$$

which is to be solved in the least squares sense, where \mathbf{L} is a diagonal matrix whose elements are the eigenvalues associated to the basis vectors, and λ_r is a regularization

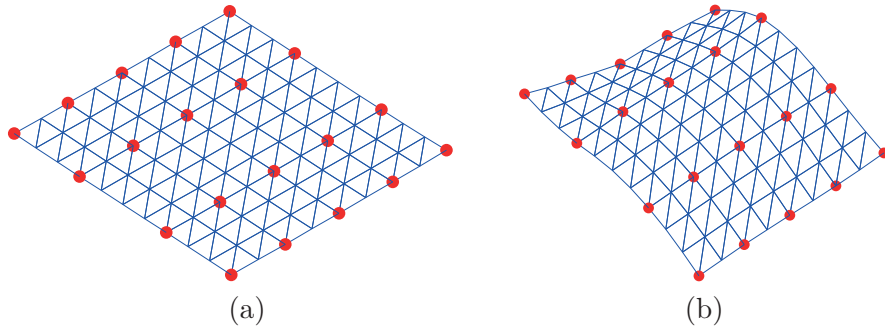


Figure 3.1: **Linear parameterization of the mesh using control vertices.** (a) Reference shape and (b) Deformed shape. Every vertex is a linear combination of the control vertices, shown in red. In this case, the reference shape is planar.

weight. This favors basis vectors that correspond to the lowest-frequency deformations and therefore enforces smoothness.

In practice, the linear system of Eq. 3.7 is better posed than the one of Eq. 3.5. But, because there are usually several *smooth* shapes that all yield virtually the same projection, its matrix still has a number of near zero singular values. As a consequence, additional constraints still need to be imposed for the problem to become well-posed. An additional difficulty is that, because rotations are strongly non-linear, this linear formulation can only handle small ones. As a result, the reference shape defined by \mathbf{x}_0 must be roughly aligned with the shape to be recovered, which means that a global rotation must be computed before shape recovery can be attempted. In the following section, we will show that we can reduce the dimensionality in a different and rotation-invariant way.

3.3 Laplacian Formulation

In the previous section, we introduced the linear system of Eq. 3.5, which is so ill-conditioned that we cannot minimize the reprojection error by simply solving it. In this section, we show how to turn it into a well-conditioned system using a novel regularization term and how to reduce the size of the problem for better computational efficiency.

To this end, let us assume we are given a reference shape as in Fig. 3.1(a), which may or may not be planar and let \mathbf{x}_{ref} be the coordinate vector of its vertices. We first show that we can define a regularization matrix \mathbf{A} such that $\|\mathbf{Ax}\|^2 = 0$, with \mathbf{x} being the coordinate vector of Eq. 3.5, when $\mathbf{x}_{\text{ref}} = \mathbf{x}$ up to a rigid transformation. In other words, $\|\mathbf{Ax}\|^2$ penalizes non-rigid deformations away from the reference shape but not rigid ones. The ill-conditioned linear system of Eq. 3.5 is augmented with the regularization term $\|\mathbf{Ax}\|^2$ to obtain a much better-conditioned linear system

$$\min_{\mathbf{x}} \quad \|\mathbf{Mx}\|^2 + w_r^2 \|\mathbf{Ax}\|^2, \text{ s. t. } \|\mathbf{x}\| = 1, \quad (3.8)$$

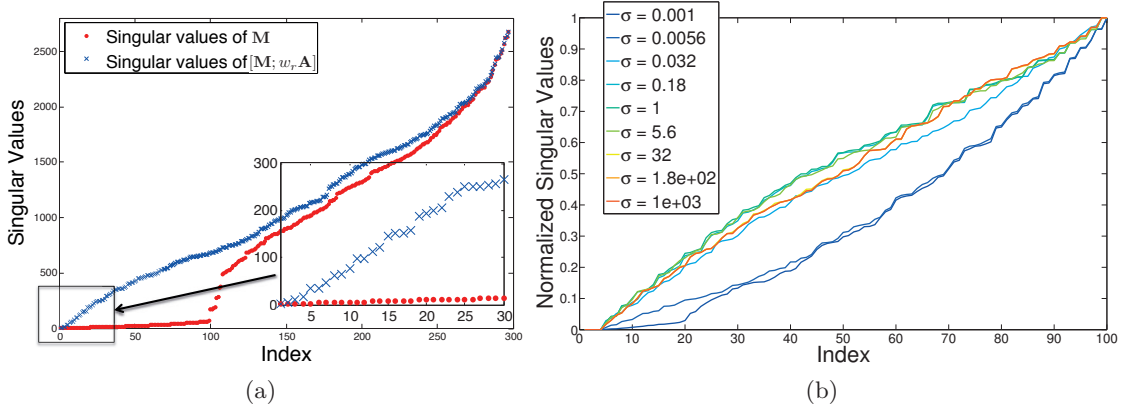


Figure 3.2: **Conditioning of the regularization matrix.** (a) Singular values of \mathbf{M} and $\mathbf{M}_{w_r} = [\mathbf{M}; w_r \mathbf{A}]$, in red and blue respectively, for a planar model and a given set of correspondences. \mathbf{M}_{w_r} has no zero singular values and only a few that are small whereas the first N_v singular values of \mathbf{M} are much closer to zero than the rest. (b) Singular values of regularization matrix \mathbf{A} for a non-planar model corresponding to one specific coordinate divided by the largest one. Each curve corresponds to a different value σ introduced in Section 3.3.1.2. Note that the curves are almost superposed for σ values greater than one. The first 4 singular values being 0 indicates that affine transformations are not penalized.

where w_r is a scalar coefficient defining how much we regularize the solution. Fig. 3.2(a) illustrates this for a specific mesh. In the result section, we will see that this behavior is generic and that we can solve this linear system for all the reference meshes we use in our experiments.

We then show that, given a subset of N_c mesh vertices whose coordinates are

$$\mathbf{c} = \begin{bmatrix} \mathbf{v}_{i_1} \\ \vdots \\ \mathbf{v}_{i_{N_c}} \end{bmatrix}, \quad (3.9)$$

if we force the mesh both to go through these coordinates and to minimize $\|\mathbf{A}\mathbf{x}\|^2$, we can define a matrix \mathbf{P} that is independent of the control vertex coordinates \mathbf{c} and such that

$$\mathbf{x} = \mathbf{P}\mathbf{c}. \quad (3.10)$$

In other words, we can linearly parameterize the mesh as a function of the control vertices' coordinates, as illustrated in Fig. 3.1, where the control vertices are shown in red and (a) is the reference shape and (b) is a deformed shape according to Eq. 3.10. Injecting this

parameterization into Eq. 3.8 yields a more compact linear system

$$\min_{\mathbf{c}} \quad \|\mathbf{MPc}\|^2 + w_r^2 \|\mathbf{APc}\|^2, \text{ s. t. } \|\mathbf{c}\| = 1, \quad (3.11)$$

which similarly can be solved in the least-square sense up to a scale factor by finding the eigenvector corresponding to the smallest eigenvalue of the matrix $\mathbf{M}_{w_r}^T \mathbf{M}_{w_r}$, in which

$$\mathbf{M}_{w_r} = \begin{bmatrix} \mathbf{MP} \\ w_r \mathbf{AP} \end{bmatrix}. \quad (3.12)$$

We fix the scale by making the average edge-length be the same as in the reference shape. This problem is usually sufficiently well-conditioned. Its solution is a mesh whose projection is very accurate but whose 3-D shape may not be because our regularization does not penalize affine deformations away from the reference shape. In practice, we use this initial mesh to eliminate erroneous correspondences. We then refine it by solving

$$\min_{\mathbf{c}} \quad \|\mathbf{MPc}\|^2 + w_r^2 \|\mathbf{APc}\|^2, \text{ s. t. } C(\mathbf{Pc}) \leq 0, \quad (3.13)$$

where $C(\mathbf{Pc})$ are inextensibility constraints that prevent Euclidean distances between neighboring vertices to grow beyond a bound, such as their geodesic distance in the reference shape. We use inequality constraints because, in high-curvature areas, the Euclidean distance becomes smaller than the geodesic distance. These are exactly the same constraints as those used in [Salzmann and Fua, 2011], against which we compare ourselves below. The inequality constraints are reformulated as equality constraints with additional slack variables whose norm is penalized to prevent lengths from becoming too small and the solution from shrinking to the origin [Fua et al., 2010]. This makes it unnecessary to introduce the depth constraints of [Salzmann and Fua, 2011]. As shown in Section 3.5.5, this non-linear minimization step is important to guarantee that not only are the projections correct but also the actual 3-D shape.

3.3.1 Regularization Matrix

We now turn to building the matrix \mathbf{A} such that $\mathbf{Ax}_{\text{ref}} = \mathbf{0}$ and $\|\mathbf{Ax}\|^2 = \|\mathbf{Ax}'\|^2$ when \mathbf{x}' is a rigidly transformed version of \mathbf{x} . We first propose a very simple scheme for the case when the reference shape is planar and then a similar, but more sophisticated one, when it is not.

3.3.1.1 Planar Reference Shape

Given a planar triangular mesh that represents a reference surface in its reference shape, consider every pair of facets that share an edge, such as those depicted by Fig. 3.3(a).

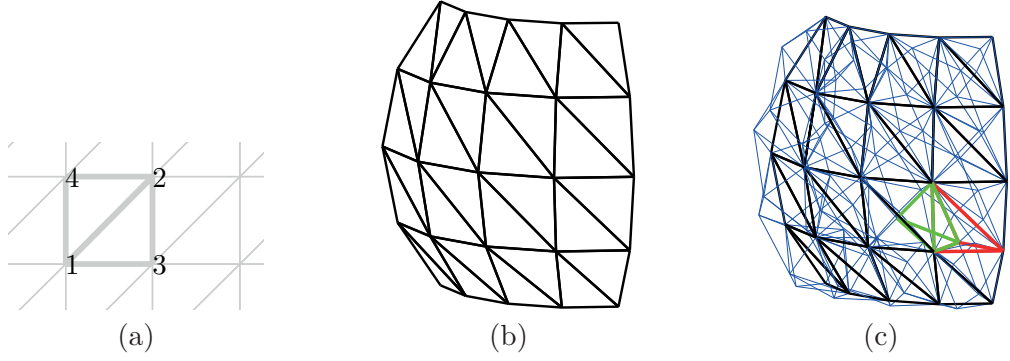


Figure 3.3: Building the regularization matrix \mathbf{A} of Section 3.3.1. (a) Two facets that share an edge. (b) Non-planar reference mesh. (c) Non-planar reference mesh with virtual vertices and edges added. The virtual vertices are located above and below the center of each facet. They are connected to vertices of their corresponding mesh facet and virtual vertices of the neighbouring facets on the same side. This produces pairs of blue tetrahedra, one of those is shown in green and the other in red.

They jointly have four vertices $\mathbf{v}_{1\text{ref}}$ to $\mathbf{v}_{4\text{ref}}$. Since they lie on a plane, whether or not the mesh is regular, one can always find a unique set of weights w_1, w_2, w_3 and w_4 such that

$$\begin{aligned} \mathbf{0} &= w_1 \mathbf{v}_{1\text{ref}} + w_2 \mathbf{v}_{2\text{ref}} + w_3 \mathbf{v}_{3\text{ref}} + w_4 \mathbf{v}_{4\text{ref}}, \\ 0 &= w_1 + w_2 + w_3 + w_4, \\ 1 &= w_1^2 + w_2^2 + w_3^2 + w_4^2, \end{aligned} \tag{3.14}$$

up to a sign ambiguity, which can be resolved by simply requiring the first weight to be positive. The first equation in Eq. 3.14 applies for each of three spatial coordinates x, y, z . To form the \mathbf{A} matrix, we first generate a matrix \mathbf{A}' for a single coordinate component. For every facet pair k , let i_1, i_2, i_3, i_4 be the indices of four corresponding vertices. The values at row k , columns i_1, i_2, i_3, i_4 of \mathbf{A}' are taken to be w_1, w_2, w_3, w_4 . All remaining elements are set to zero and the matrix \mathbf{A} is $\mathbf{A} = \mathbf{I}_3 \otimes \mathbf{A}'$, that is the Kronecker product with a 3×3 identity matrix.

The first equality of Eq. 3.14 is designed to enforce planarity while the second guarantees invariance. The third equality is there to prevent the weights from all being zero. We show in Section 3.3.1.3 that $\mathbf{Ax} = \mathbf{0}$ when \mathbf{x} is an affine transformed version of the reference mesh and that $\|\mathbf{Ax}\|^2$ is invariant to rotations and translations. The more the mesh deviates from an affine transformed version of the reference mesh, the more the regularization term penalizes.

3.3.1.2 Non-Planar Reference Shape

When the reference shape is non-planar, there will be facets for which we cannot solve Eq. 3.14. As in [Sumner and Popovic, 2004], we extend the scheme described above by introducing virtual vertices. As shown in Fig. 3.3(c), we create virtual vertices at above and below the center of each facet as a distance controlled by the scale parameter σ . Formally, for each facet \mathbf{v}_i , \mathbf{v}_j and \mathbf{v}_k , its center $\mathbf{v}_c = \frac{1}{3}(\mathbf{v}_i + \mathbf{v}_j + \mathbf{v}_k)$ and its normal $\mathbf{n} = (\mathbf{v}_j - \mathbf{v}_i) \times (\mathbf{v}_k - \mathbf{v}_i)$, we take the virtual vertices to be

$$\mathbf{v}_{ijk}^+ = \mathbf{v}_c + \sigma \frac{\mathbf{n}}{\sqrt{\|\mathbf{n}\|}} \quad \text{and} \quad \mathbf{v}_{ijk}^- = \mathbf{v}_c - \sigma \frac{\mathbf{n}}{\sqrt{\|\mathbf{n}\|}}, \quad (3.15)$$

where the norm of $\mathbf{n}/\sqrt{\|\mathbf{n}\|}$ is approximately equal to the average edge-length of the facet's edges. These virtual vertices are connected to vertices of their corresponding mesh facet and virtual vertices of the neighbouring facets on the same side, making up the blue tetrahedra of Fig. 3.3(c).

Let \mathbf{x}^V be the coordinate vector of the virtual vertices only, and $\mathbf{x}^U = [\mathbf{x}; \mathbf{x}^V]$ the coordinate vector of both real and virtual vertices. Let $\mathbf{x}_{\text{ref}}^U$ be similarly defined for the reference shape. Given two tetrahedra that share a facet, such as the red and green ones in Fig. 3.3(c), and the five vertices $\mathbf{v}_{1\text{ref}}^U$ to $\mathbf{v}_{5\text{ref}}^U$ they share, we can now find weights w_1 to w_5 such that

$$\begin{aligned} \mathbf{0} &= w_1 \mathbf{v}_{1\text{ref}}^U + w_2 \mathbf{v}_{2\text{ref}}^U + w_3 \mathbf{v}_{3\text{ref}}^U + w_4 \mathbf{v}_{4\text{ref}}^U + w_5 \mathbf{v}_{5\text{ref}}^U, \\ 0 &= w_1 + w_2 + w_3 + w_4 + w_5, \\ 1 &= w_1^2 + w_2^2 + w_3^2 + w_4^2 + w_5^2. \end{aligned} \quad (3.16)$$

The three equalities of Eq. 3.16 serve the same purpose as those of Eq. 3.14. We form a matrix \mathbf{A}^U by considering all pairs of tetrahedra that share a facet, computing the w_1 to w_5 weights that encode local linear dependencies between real and virtual vertices, and using them to add successive rows to the matrix, as we did to build the \mathbf{A} matrix in the planar case. It is also shown in Section 3.3.1.3 below that $\mathbf{A}^U \mathbf{x}^U = \mathbf{0}$ when \mathbf{x}^U is an affine transformed version of $\mathbf{x}_{\text{ref}}^U$ and that $\|\mathbf{A}^U \mathbf{x}^U\|^2$ is invariant to rigid transformations of \mathbf{x}^U . In our scheme, the regularization term can be computed as

$$C = \|\mathbf{A}^U \mathbf{x}^U\|^2 = \|\hat{\mathbf{A}} \mathbf{x} + \tilde{\mathbf{A}} \mathbf{x}^V\|^2, \quad (3.17)$$

if we write \mathbf{A}^U as $[\hat{\mathbf{A}} \tilde{\mathbf{A}}]$ where $\hat{\mathbf{A}}$ has three times as many columns as there are real vertices and $\tilde{\mathbf{A}}$ as there are virtual ones. Given the real vertices \mathbf{x} , the virtual vertex

coordinates that minimize the C term of Eq. 3.17 is

$$\begin{aligned} \mathbf{x}^V &= -\left(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}\right)^{-1} \tilde{\mathbf{A}}^T \hat{\mathbf{A}} \mathbf{x} \\ \Rightarrow C &= \left\| \hat{\mathbf{A}} \mathbf{x} - \tilde{\mathbf{A}} \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}\right)^{-1} \tilde{\mathbf{A}}^T \hat{\mathbf{A}} \mathbf{x} \right\|^2 = \|\mathbf{A} \mathbf{x}\|^2, \end{aligned} \quad (3.18)$$

where $\mathbf{A} = \hat{\mathbf{A}} - \tilde{\mathbf{A}} \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}\right)^{-1} \tilde{\mathbf{A}}^T \hat{\mathbf{A}}$. In other words, this matrix \mathbf{A} is the regularization matrix we are looking for and its elements depend only on the coordinates of the reference vertices and on the scale parameter σ chosen to build the virtual vertices.

To study the influence of the scale parameter σ of Eq. 3.15, which controls the distance of the virtual vertices from the mesh, we computed the \mathbf{A} matrix and its singular values for a sail shaped mesh and many σ values. For each σ , we normalized the singular values to be in the range $[0, 1]$. As can be seen in Fig. 3.2(b), there is a wide range of σ for which the distribution of singular values remains practically identical. This suggests that σ has limited influence on the numerical character of the regularization matrix. In all our experiments, we set σ to 1 and were nevertheless able to obtain accurate reconstructions of many different surfaces with very different physical properties.

3.3.1.3 Properties of the Regularization Term

Here we prove the claim made in Section 3.3.1.1 that $\mathbf{A} \mathbf{x} = \mathbf{0}$ when \mathbf{x} represents an affine transformed version of the reference mesh and that $\|\mathbf{A} \mathbf{x}\|^2$ is invariant to rotations and translations.

The first equation in Eq. 3.14 applies for each of three spatial coordinates x, y, z and can be rewritten in a matrix form as

$$[w_1 \ w_2 \ w_3 \ w_4] \begin{bmatrix} \mathbf{v}_{1\text{ref}x} & \mathbf{v}_{1\text{ref}y} & \mathbf{v}_{1\text{ref}z} \\ \mathbf{v}_{2\text{ref}x} & \mathbf{v}_{2\text{ref}y} & \mathbf{v}_{2\text{ref}z} \\ \mathbf{v}_{3\text{ref}x} & \mathbf{v}_{3\text{ref}y} & \mathbf{v}_{3\text{ref}z} \\ \mathbf{v}_{4\text{ref}x} & \mathbf{v}_{4\text{ref}y} & \mathbf{v}_{4\text{ref}z} \end{bmatrix} = \mathbf{0}^T. \quad (3.19)$$

It can be seen from Eq. 3.19 and the second equation in Eq. 3.14 that the three vectors of the x, y, z components of the reference mesh and the vector of all 1s lie in the kernel of the matrix \mathbf{A}' . It means our regularization term $\|\mathbf{A} \mathbf{x}\|^2$ does not penalize affine transformation of the reference mesh. Hence, $\mathbf{A} \mathbf{x} = \mathbf{0}$ as long as \mathbf{x} represents an affine transformed version of the reference mesh.

Let $\mathbf{v}'_i = \mathbf{R} \mathbf{v}_i + \mathbf{t}$ be the new location of vertex \mathbf{v}_i under a rigid transformation of the mesh, where \mathbf{R} is the rotation matrix and \mathbf{t} is the translation vector. Since $\mathbf{R}^T \mathbf{R} = \mathbf{I}_3$

and $w_1 + w_2 + w_3 + w_4 = 0$, we have

$$\begin{aligned}
 & \|w_1 \mathbf{v}'_{i_1} + w_2 \mathbf{v}'_{i_2} + w_3 \mathbf{v}'_{i_3} + w_4 \mathbf{v}'_{i_4}\|^2 \\
 = & \left\| \mathbf{R}(w_1 \mathbf{v}_{i_1} + w_2 \mathbf{v}_{i_2} + w_3 \mathbf{v}_{i_3} + w_4 \mathbf{v}_{i_4}) + \mathbf{t}(w_1 + w_2 + w_3 + w_4) \right\|^2 \\
 = & \|w_1 \mathbf{v}_{i_1} + w_2 \mathbf{v}_{i_2} + w_3 \mathbf{v}_{i_3} + w_4 \mathbf{v}_{i_4}\|^2 .
 \end{aligned} \tag{3.20}$$

Hence, $\|\mathbf{A}\mathbf{x}'\|^2 = \|\mathbf{A}\mathbf{x}\|^2$ when \mathbf{x}' is a rigidly transformed version of \mathbf{x} . In other words, $\|\mathbf{A}\mathbf{x}\|^2$ is invariant to rotations and translations. Similar results are obtained for non-planar reference shapes by applying the same proof.

3.3.2 Linear Parameterization

We now show how to use our regularization matrix \mathbf{A} to linearly parameterize the mesh using a few control vertices. As before, let N_v be the total number of vertices and $N_c < N_v$ the number of control points used to parameterize the mesh. Given the coordinate vector \mathbf{c} of these control vertices, the coordinates of the minimum energy mesh that goes through the control vertices and minimizes the regularization energy can be found by minimizing

$$\|\mathbf{A}\mathbf{x}\|^2 \text{ subject to } \mathbf{P}_c \mathbf{x} = \mathbf{c} , \tag{3.21}$$

where \mathbf{A} is the regularization matrix introduced above and \mathbf{P}_c is a $3N_c \times 3N_v$ matrix containing ones in columns corresponding to the indices of control vertices and zeros elsewhere. We prove below that, there exists a matrix \mathbf{P} whose components can be computed from \mathbf{A} and \mathbf{P}_c such that

$$\forall \mathbf{c} , \mathbf{x} = \mathbf{P}\mathbf{c} , \tag{3.22}$$

if \mathbf{x} is a solution to the minimization problem of Eq. 3.21.

To prove this without loss of generality, we can order the coordinate vector \mathbf{x} so that those of the control vertices come first. This allows us to write all coordinate vectors that satisfy the constraint as

$$\mathbf{x} = \begin{bmatrix} \mathbf{c} \\ \boldsymbol{\lambda} \end{bmatrix} , \tag{3.23}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{3(N_v - N_c)}$. Let us rewrite the matrix \mathbf{A} as

$$\mathbf{A} = [\mathbf{A}_c | \mathbf{A}_\lambda] , \tag{3.24}$$

where \mathbf{A}_c and \mathbf{A}_λ have $3N_c$ and $3(N_v - N_c)$ columns, respectively. Solving the problem

of Eq. 3.21, becomes equivalent to minimizing

$$\begin{aligned} \|\mathbf{A}_c \mathbf{c} + \mathbf{A}_\lambda \boldsymbol{\lambda}\|^2 &\Rightarrow \boldsymbol{\lambda} = -(\mathbf{A}_\lambda^T \mathbf{A}_\lambda)^{-1} \mathbf{A}_\lambda^T \mathbf{A}_c \mathbf{c} \\ &\Rightarrow \mathbf{x} = \begin{bmatrix} \mathbf{I} \\ -(\mathbf{A}_\lambda^T \mathbf{A}_\lambda)^{-1} \mathbf{A}_\lambda^T \mathbf{A}_c \end{bmatrix} \mathbf{c}. \end{aligned} \quad (3.25)$$

Therefore,

$$\mathbf{P} = \begin{bmatrix} \mathbf{I} \\ -(\mathbf{A}_\lambda^T \mathbf{A}_\lambda)^{-1} \mathbf{A}_\lambda^T \mathbf{A}_c \end{bmatrix}, \quad (3.26)$$

is the matrix of Eq. 3.22 under our previous assumption that the vertices were ordered such that the control vertices come first.

3.4 Rejecting Outliers in 3D

Our reconstruction algorithm relies on the wide-baseline correspondences established between the template image and the input image. However, these wide-baseline correspondences usually contain errors, which negatively influence the reconstruction. To handle outliers in this putative set of correspondences, we iteratively perform the unconstrained optimization of Eq. 3.11 starting with a relatively high regularization weight w_r and gradually reducing it during iterations.

The solution \mathbf{c} to Eq. 3.11 with a current value w_r is the eigenvector corresponding to the smallest eigenvalue of the following matrix:

$$(\mathbf{MP})^T \cdot (\mathbf{MP}) + w_r^2 \cdot (\mathbf{AP})^T \cdot (\mathbf{AP}). \quad (3.27)$$

This solution gives a current shape estimate $\mathbf{x} = \mathbf{Pc}$. We project it on the input image and disregard the correspondences with higher reprojection error than a pre-set inlier confidence radius, whose initial value is large and gradually decreases during iterations. We reduce both the regularization weight w_r and the inlier confidence radius by half for the next iteration. Repeating this procedure a fixed number of times results in an initial shape estimate and provides inlier correspondences for the more computationally demanding constrained optimization that follows.

The scale of the initial estimate \mathbf{c} returned by the procedure above does not match the scale of the mesh because we constraint $\|\mathbf{c}\| = 1$. Hence, to be able to initialize the constrained optimization problem in Eq. 3.13, we have fix the scale of \mathbf{c} by making the average edge-length of the resulting mesh be the same as in the reference one.

Even though the minimization problem we solve is similar to that of [Salzmann and Fua,

2011], this two-step outlier rejection scheme brings us a computational advantage. In the earlier approach, convex but nonlinear minimization had to be performed several times at each iteration to reject the outliers whereas here we repeatedly solve a linear least squares problem instead. Furthermore, the final non-linear minimization of Eq. 3.13 involves far fewer variables since the control vertices typically form a small subset of all vertices, that is, $N_c \ll N_v$.

3.5 Quantitative Results on Real Data

In this section, we first compare our approach to surface reconstruction from a single input image given correspondences with a reference image against the recent methods of [Brunet et al., 2010, Salzmann and Fua, 2011, Bartoli et al., 2012], which are representative of the current state-of-the-art.

To provide quantitative results both in the planar and non-planar cases, we use five different sequences for which we have other means besides our single-camera approach to estimate 3-D shape. We first describe them and then present our results. Note that we process every single image *individually* and do not enforce any kind of temporal consistency to truly compare the single-frame performance of all four approaches.

3.5.1 Sequences, Templates, and Validation

We acquired four sequences using a Kinecttm, whose RGB camera focal length is about 528 pixels. We show one representative image from each dataset in Figs. 3.4 and 3.5. We performed the 3-D reconstruction using individual RGB images and ignoring the corresponding depth images. We used these *only* to build the initial template from the first frame of each sequence and then for validation purposes. The Kinect random error in depth measurements is about 2 mm in our settings [Khoshelham and Elberink, 2012]. The specifics for each dataset are provided in Table 3.1.

For the paper and apron of Fig. 3.4, we used the planar templates shown in the first two rows of Fig. 3.8 and for the cushion and banana leaf of Fig. 3.5 the non-planar ones shown in next two rows of the same figure. For the paper, cushion, and leaf, we defined the template from the first frame, which we took to be our reference image. For the apron, we acquired a separate image in which the apron lies flat on the floor so that the template is flat as well, which it is not in the first frame of the sequence, so that we could use the method of [Brunet et al., 2010] in the conditions it was designed for.

To create the required correspondences, we used either SIFT [Lowe, 2004] or Ferns [Ozuysal et al., 2010] to establish correspondences between the template and the input image. We ran our method and three others [Brunet et al., 2010, Bartoli et al., 2012, Salzmann and Fua, 2011] using these correspondences as input to produce results such as those

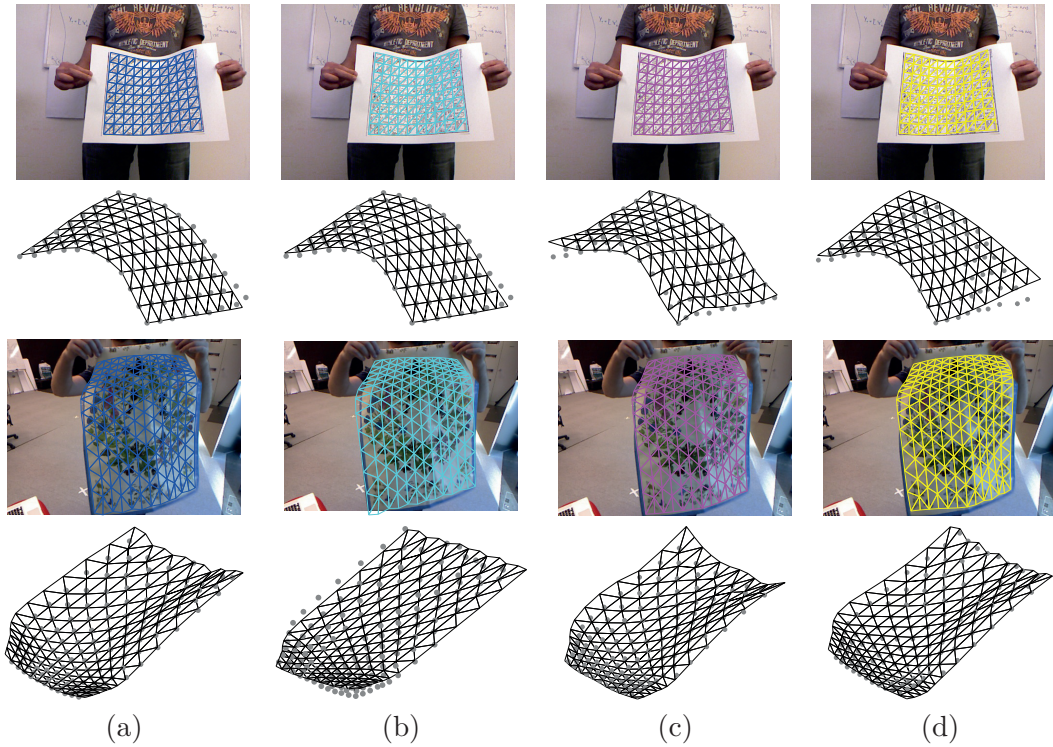


Figure 3.4: **Paper and Apron using a planar template.** In the first and third rows, we project the 3-D surfaces reconstructed using different methods into the image used to perform the reconstruction. The overlay colors correspond to those of the graphs in Fig. 3.9. In the second and fourth rows, we show the same surfaces seen from a different viewpoint. The gray dots denote the projections of the mesh vertices onto the ground-truth surface. The closer they are to the mesh, the better the reconstruction. (a) Our method using regularly sampled control vertices. (b) Brunet et al. [Brunet et al., 2010] (c) Bartoli et al. [Bartoli et al., 2012] (d) Salzmann et al. [Salzmann and Fua, 2011].

depicted by Figs. 3.4 and 3.5.

As stated above, we did not use the depth-maps for reconstruction purposes, only for validation purposes. We take the reconstruction accuracy to be the average distance of the reconstructed 3-D vertices to their projections onto the depth images, provided they fall within the target object. Otherwise, they are ignored. Since we do the same for all four methods we compare, comparing their performance in terms of these averages and corresponding variances is meaningful and does not introduce a bias.

The last dataset is depicted by Fig. 3.6. It consists of 6 pairs of images of a sail, which is a much larger surface than the other four and would be hard to capture using a Kinect-style sensor. The images were acquired by two synchronized cameras whose focal lengths are about 6682 pixels, which let us accurately reconstruct the 3-D positions of the circular black markers by performing bundle-adjustment on each image pair. As before we took the first image captured by the first camera to be our reference image

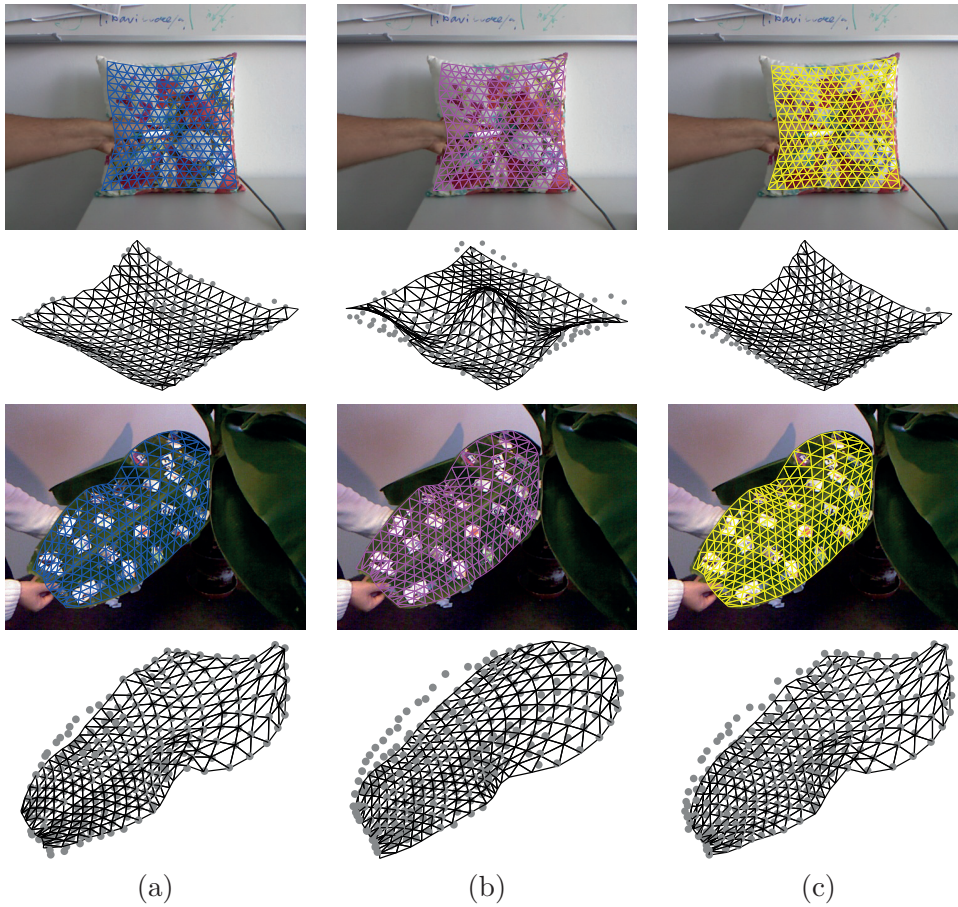


Figure 3.5: **Cushion and banana leaf using a non-planar template.** As in Fig. 3.4, we overlay the surfaces reconstructed using the different methods on the corresponding images and also show them as seen from a different viewpoint. The gray dots denote the projections of the mesh vertices onto the ground-truth surface and the overlay colors correspond to those of the graphs in Fig. 3.10(a) Our method using regularly sampled control vertices. (b) Bartoli et al. [Bartoli et al., 2012] (c) Salzmann et al. [Salzmann and Fua, 2011].

and established correspondences between the markers it contains and those seen in the other images. We also established correspondences between markers within the stereo pairs and checked them manually. We then estimated the accuracy of the monocular reconstructions in terms of the distances of the estimated markers' 3-D positions to those obtained from stereo.

The 3-D positions of the markers used to evaluate the reconstruction error are estimated from the stereo images up to an unknown scale factor: This stems from the fact that both the relative pose of one stereo camera to the other *and* the 3-D positions of the markers are unknown and have to be estimated simultaneously. However, once a stereo reconstruction of the markers has been obtained, fitting a mesh to those markers lets us

3.5. Quantitative Results on Real Data

Table 3.1: The five datasets used for quantitative evaluation. Number of frames in each sequence and specifics of the corresponding reference shape.

Dataset	# frames	# vertices	# facets	Planar
Paper	192	99	160	Yes
Apron	160	169	288	Yes
Cushion	46	270	476	No
Leaf	283	260	456	No
Sail	6	66	100	No

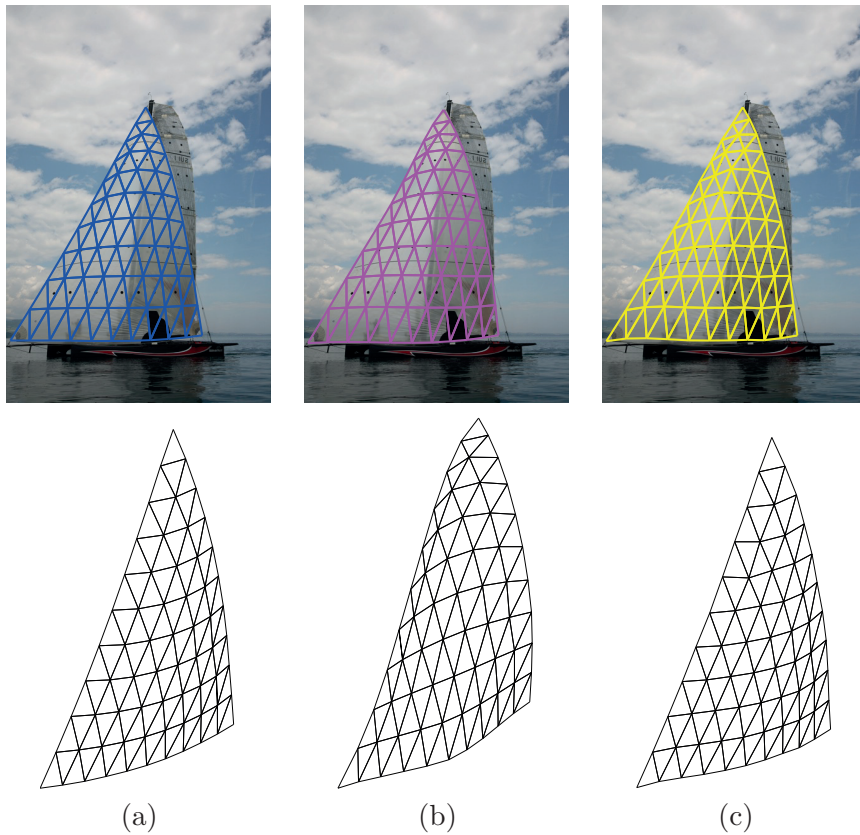


Figure 3.6: **Sail using a non-planar template.** As in Figs. 3.4 and 3.5, we both overlay the surfaces reconstructed using the different methods on the image used to perform the reconstruction and show them as seen from a different viewpoint. (a) Our method using regularly sampled control points. (b) Bartoli et al. [2012] (c) Salzmann and Fua [2011].

estimate an approximate scale by assuming the total edge length of the fitted mesh to be that of the reference mesh. However it is not accurate and, for evaluation purposes, we find a scale factor within the range $[0.8, 1.25]$ that we apply to each one of the monocular reconstructions so as to minimize the mean distance between the predicted 3-D marker positions of the monocular reconstructions and the 3-D positions obtained from stereo.

3.5.2 Robustness to Erroneous Correspondences

We demonstrate the robustness of our outlier rejection scheme described in Section 3.4 for both planar and non-planar surfaces. We also compare our method against two recent approaches to rejecting outliers in deformable surface reconstruction [Pizarro and Bartoli, 2012, Tran et al., 2012] whose implementations are publicly available. Alternatives include the older M-estimator style approaches of [Zhu and Lyu, 2007, Pilet et al., 2008], which are comparable in matching performance on the kind of data we use but slower [Tran et al., 2012]. A more recent method [Collins and Bartoli, 2014] involves detecting incorrect correspondences using local isometry deformation constraints by locally performing intensity-based dense registration. While potentially effective, this algorithm requires several seconds per frame, which is prohibitive for real-time implementation purpose.

We used one image from both the paper and the cushion datasets in which the surfaces undergo large deformations, as depicted by the first row of Figs. 3.4 and 3.5. We used both the corresponding Kinect point cloud and SIFT correspondences to reconstruct a mesh that we treated as the ground truth. We then synthesized approximately 600'000 sets of correspondences by synthetically generating 10 to 400 inlier correspondences spread uniformly across the surface, adding a zero mean Gaussian noise with standard deviation of 1 pixel to the corresponding pixel coordinates, and introducing proportions varying from 0 to 100% of randomly spread outlier correspondences.

We ran our algorithm with 25 regularly sampled control vertices on each one of these sets of correspondences. To ensure a fair comparison of outlier rejection capability in the context of deformable surface reconstruction, also because [Tran et al., 2012] was designed only for outlier rejection, and the implementation of [Pizarro and Bartoli, 2012] only performs outlier rejection, we used the set of inlier correspondences returned by these methods to perform surface registration using our method assuming that all input correspondences are inliers. For [Pizarro and Bartoli, 2012, Tran et al., 2012], we used the published parameters except for the threshold used to determine if a correspondence is an inlier or an outlier and the number of RANSAC iterations. We tuned the threshold manually and used 5 times more RANSAC iterations than the suggested default value for [Tran et al., 2012] to avoid a performance drop when dealing with a large proportion of outliers. Note that choosing the threshold parameter for [Tran et al., 2012] is non trivial because the manifold of inlier correspondences is not exactly a 2-D affine plane, as assumed in [Tran et al., 2012].

We evaluated the success of these competing methods in terms of how often at least 90% of the reconstructed 3-D mesh vertices project within 2 pixels of where they should. Fig. 3.7 depicts the success rates according to this criterion over many trials as a function of the total number of inliers and the proportion of outliers. Our method is comparable to [Tran et al., 2012] on the cushion dataset and better on the paper dataset. It is much better than [Pizarro and Bartoli, 2012] on both datasets. Our algorithm requires

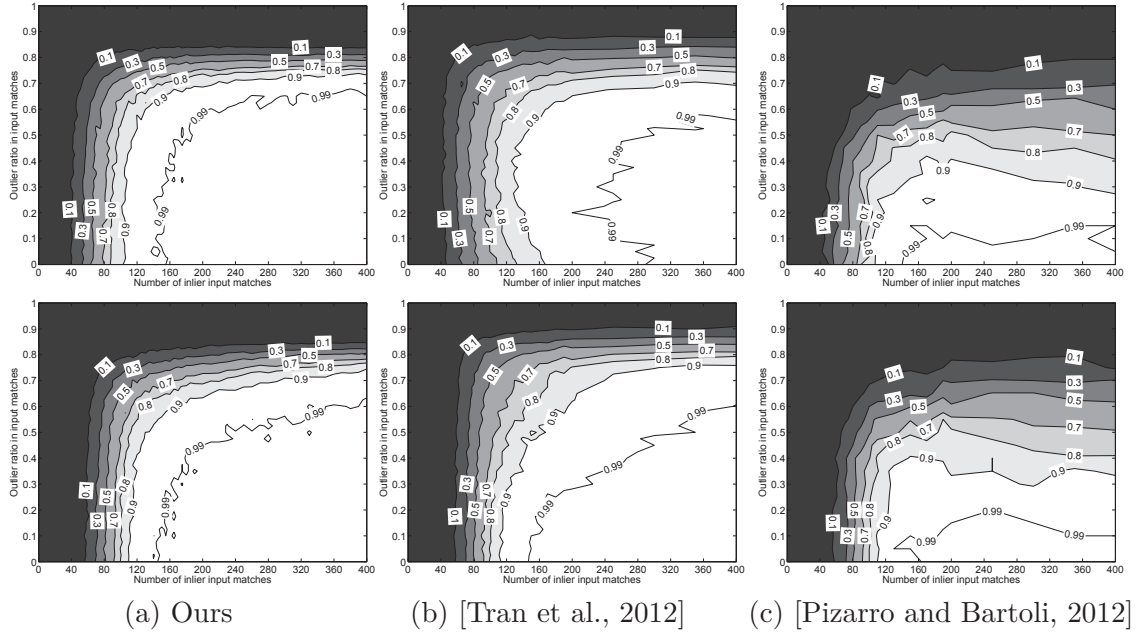


Figure 3.7: Probability of success as a function of the number of inlier matches and proportion of outliers, on the x-axis and y-axis respectively. The lines are level lines of the probability of having at least 90% mesh vertices reprojected within 2 pixels of the solution. **Top row:** paper dataset. **Bottom row:** cushion dataset.

approximately 200 inlier correspondences to guarantee the algorithm will tolerate high ratios of outliers with 0.99 probability. Rejecting outliers and registering images on approximately 600'000 set of synthesized correspondences on the paper and cushion dataset, respectively, took our method 11.73 (14.80) hours i.e. 0.070 (0.089) second per frame, took [Tran et al., 2012] 44.02 (48.83) hours i.e. 0.26 (0.29) second per frame, and took [Pizarro and Bartoli, 2012] 197.46 (201.23) hours i.e. 1.18 (1.21) second per frame.

3.5.3 Control Vertex Configurations

One of the important parameters of our algorithm is the number of control vertices to be used. For each one of the five datasets, we selected four such numbers and hand-picked regularly spaced sets of control vertices of the right cardinality. We also report the results using all mesh vertices as control vertices. The left side of Fig. 3.8 depicts the resulting configurations.

To test the influence of the positioning of the control vertices, for each sequence, we also selected control vertex sets of the same size but randomly located, such as those depicted on the right side of Fig. 3.8. We did this six times for each one of the four templates and each possible number of control vertices.

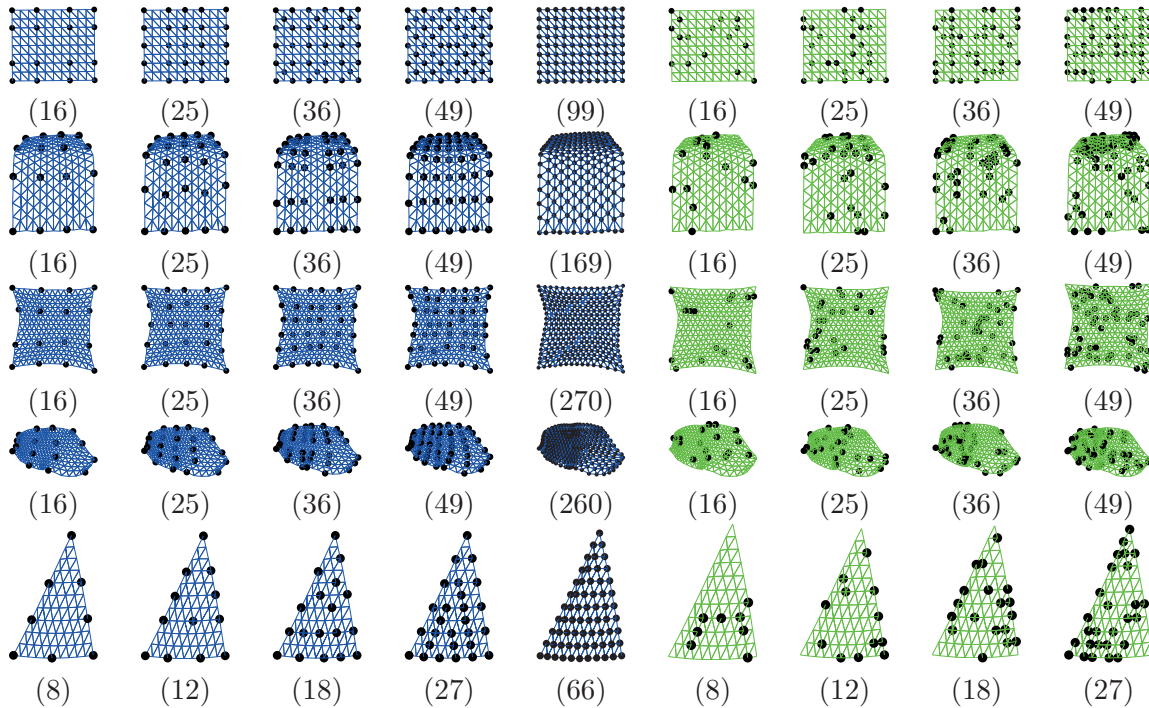


Figure 3.8: **Templates and control vertices.** Each row depicts the template and control vertex configuration used for the paper, apron, cushion, banana leaf, and sail sequences. The number below each figures denotes the number of control vertices used in each case. **Leftmost four columns:** Manually chosen control vertices. **Fifth column:** All vertices as control vertices. **Rightmost four columns:** Examples of randomly chosen control vertices.

In other words, we tested 29 different control vertex configurations for each sequence.

3.5.4 Comparative Accuracy of the Four Methods

We fed the same correspondences, which contain erroneous ones, to implementations, including the outlier rejection strategy, provided by the authors of [Brunet et al., 2010, Salzmann and Fua, 2011, Bartoli et al., 2012] to compare against our method. In the case of [Bartoli et al., 2012], we computed the 2D image warp using the algorithm of [Pizarro and Bartoli, 2012] as described in the section *Experimental Results* of [Bartoli et al., 2012].

As discussed above, for the kinect sequences, we characterize the accuracy of their respective outputs in terms of the mean and variance of the distance of the 3-D vertices reconstructed from the RGB images to their projections onto the depth images. For the sail images, we report the mean and variance of the distance between 3-D marker positions computed either monocularly or using stereo, under the assumption that the latter are more reliable.

In Fig. 3.9, we plot these accuracies when using planar templates to reconstruct the paper and apron of Fig. 3.4. The graphs are labeled as

- **Ours Regular** and **Ours Random** when using our method with either the regular or randomized control vertex configurations of Fig. 3.8. They are shown in blue and green respectively. In the case of the randomized configurations, we plot the average results over six random trials. We denote our results as **Ours All** when all mesh vertices are used as control vertices.
- **Brunet 10**, **Bartoli 12**, and **Salzmann 11** when using the methods of [Brunet et al., 2010], [Bartoli et al., 2012], and [Salzmann and Fua, 2011]. They are shown in cyan, purple, and yellow respectively.

The numbers below the graphs denote the number of control vertices we used. For a fair comparison, we used the same number of control vertices to run the method of [Brunet et al., 2010] and to compute the 2D warp required by the method of [Bartoli et al., 2012]. The algorithm of [Salzmann and Fua, 2011] does not rely on control vertices since it operates directly on the vertex coordinates. This means it depends on far more degrees of freedom than all the other methods and, for comparison purposes, we depict its accuracy by the same yellow bar in all plots of any given row.

Fig. 3.10 depicts the results similarly in the non planar case of the cushion, banana leaf, and sail of Figs. 3.5 and 3.6. The only difference is that we do not show the “Brunet 10” results because this method was not designed to handle non-planar surfaces.

3.5.5 Analysis

Our approach performs consistently as well or better than all the others when we use enough control vertices arranged in one of the regular configurations shown on the left side of Fig. 3.8. This is depicted by the blue bars in the four right-most columns of Figs. 3.9 and 3.10. Even when we use too few control vertices, we remain comparable to the other methods, as shown in the first columns of these figures.

More specifically, in the planar case, our closest competitor is the method of [Brunet et al., 2010]. We do approximately the same on the paper but much better on the apron. In the non-planar case that [Brunet et al., 2010] is not designed to handle and when using enough control vertices, we outperform the other two methods on the cushion and the leaf and perform similarly to [Salzmann and Fua, 2011] on the sail.

As stated above, these results were obtained using regular control vertex configurations and are depicted by blue bars in Figs. 3.9 and 3.10. The green bars in these plots depict the results obtained using randomized configurations such as those shown on the right side of Fig. 3.8. Interestingly, they are often quite similar, especially for larger numbers

of control vertices. This indicates the relative insensitivity of our approach to the exact placement of the control vertices. In fact, we attempted to design an automated strategy for optimal placement of the control vertices but found it very difficult to consistently do better than simply spacing them regularly, which is also much simpler.

Note that our linear subspace parameterization approach to reducing the dimensionality of the problem does not decrease the reconstruction accuracy when enough control vertices are used. We interpret this to mean that the true surface deformations lie in a lower dimensional space than the one spanned by all the vertices' coordinates.

All these experiments were carried out with a regularization weight of $w_r = 1$ in Eq. 3.13. To characterize the influence of this parameter, we used the control vertex configuration that yielded the best result for each dataset and re-ran the algorithm with the same input correspondences but with 33 different logarithmically distributed values of w_r ranging from $0.00833 = 1/120$ to 120. Fig. 3.11 depicts the results and indicates that values ranging from 1 to 10 all yield similar results.

Fig. 3.12 illustrates how well- or ill-conditioned the linear system of Eq. 3.11 is depending on the value of w_r . We plot condition numbers of \mathbf{M}_{w_r} in Eq. 3.12, averaged over several examples, as a function of w_r for all the regular control-vertex configurations we used to perform the experiments described above on the four kinect sequences. We also show the singular values of \mathbf{M}_{w_r} , also averaged over several examples, for two different configurations of the control vertices in the apron case. Note that the best condition numbers are obtained for values of w_r between 1 and 10, which is consistent with the observation made above that these are good default values to use for the regularization parameter. This can be interpreted as follows: For very small values of w_r , the **MP** term dominates and there are many shapes that have virtually the same projections. For very large values of w_r , the **AP** term dominates in Eq. 3.11 and many shapes that are rigid transformations of each other have virtually the same deformation energy. The best compromise is therefore for intermediate values of w_r .

Fig. 3.13 illustrates the importance of the non-linear constrained minimization step of Eq. 3.13 that refines the result obtained by solving the linear least-squares problem of Eq. 3.11. In the left column we show the solution of the linear least-squares problem of Eq. 3.11. It projects correctly but, as evidenced by its distance to the ground-truth gray dots, its 3-D shape is incorrect. By contrast, the surface obtained by solving the constrained optimization problem of Eq. 3.13 still reprojects correctly while also being metrically accurate. Fig. 3.14 depicts similar situations in other frames of the sequence.

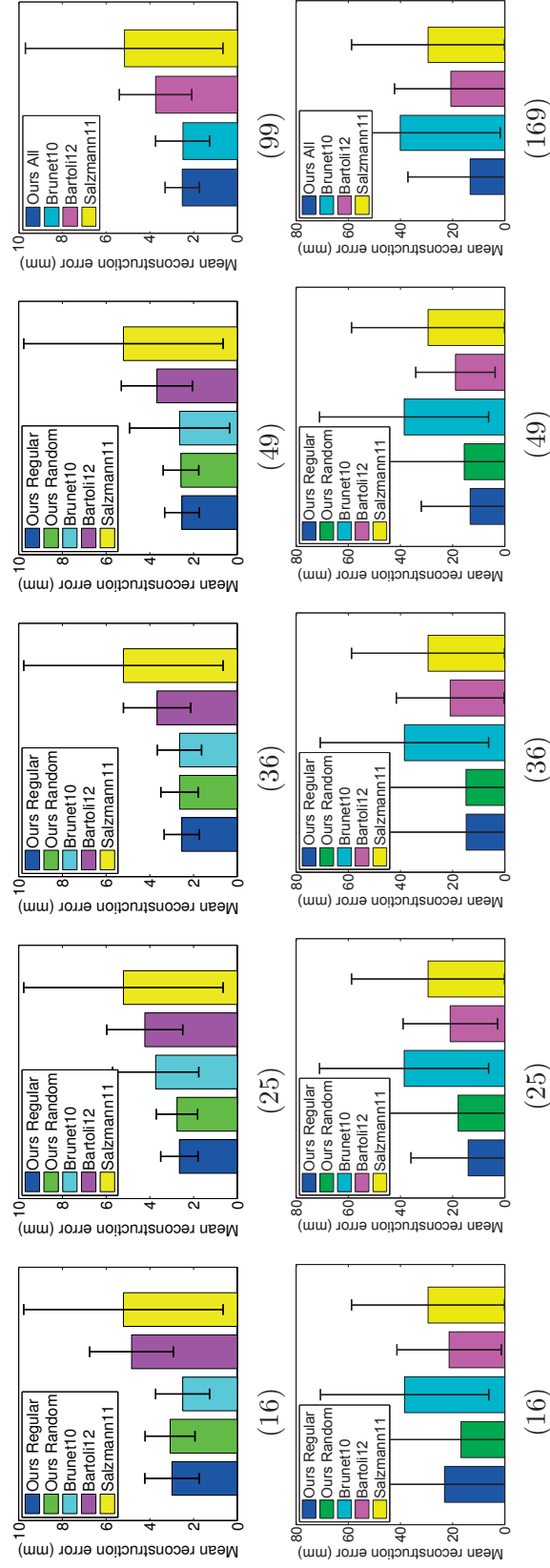


Figure 3.9: Accuracy results when using planar templates and different numbers of control vertices for the paper (top row) and apron (bottom row) sequences of Fig. 3.4.

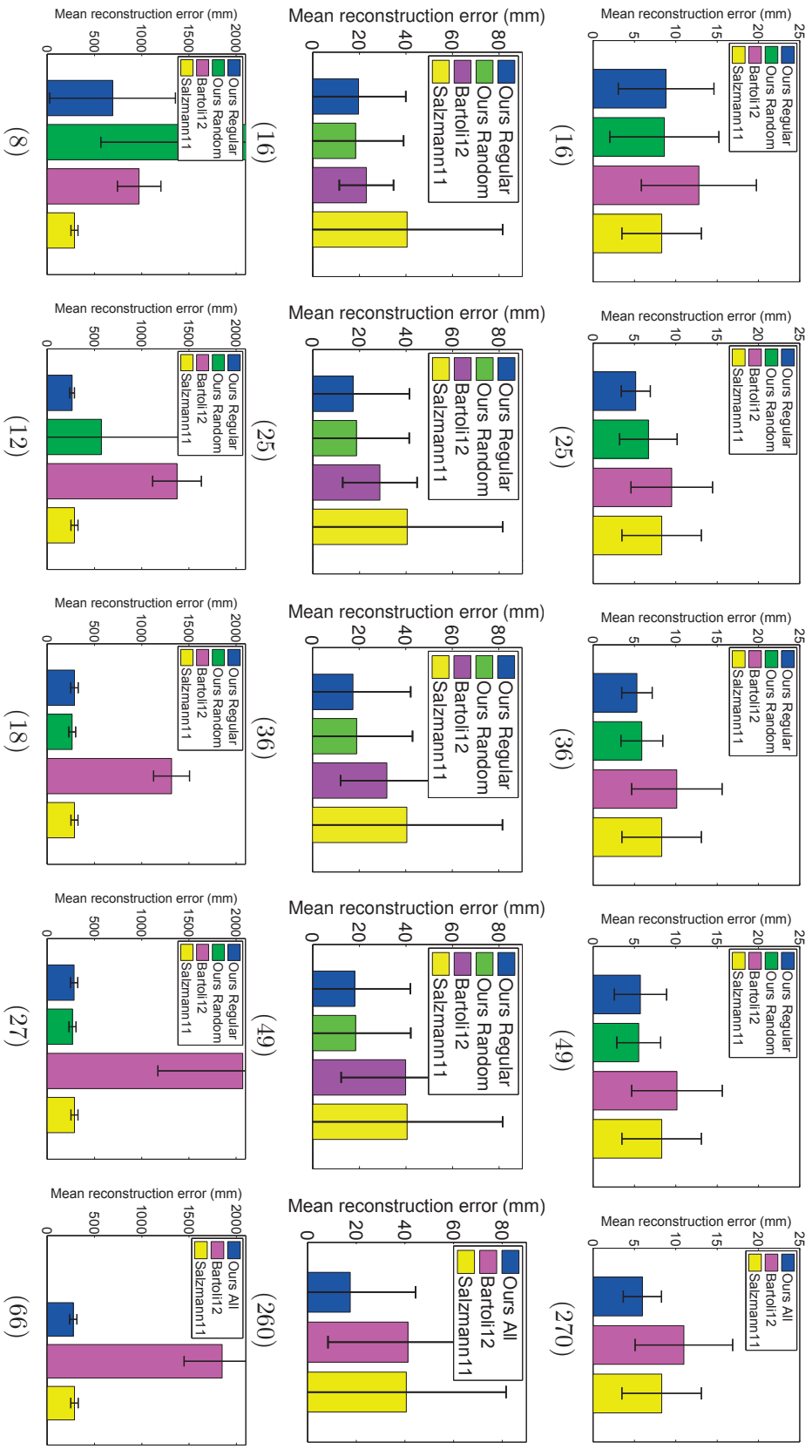


Figure 3.10: Accuracy results when using non-planar templates and different numbers of control vertices for the cushion (top row), banana leaf (middle row), and sail (bottom row) sequences of Figs. 3.5 and 3.6.

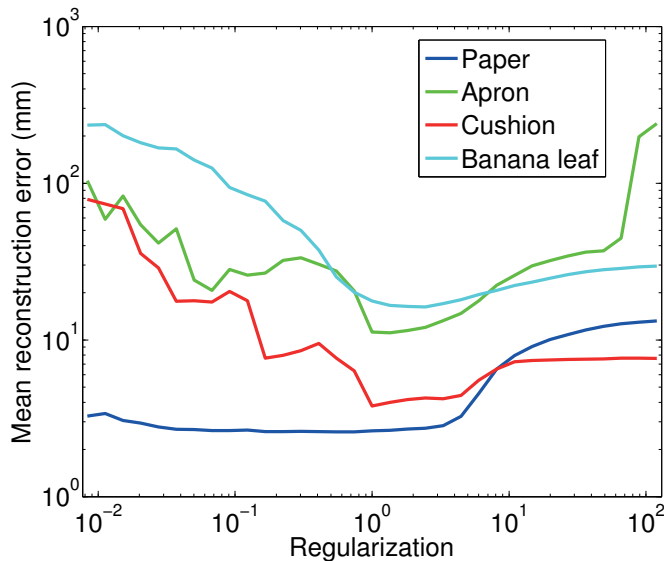


Figure 3.11: **Influence of the regularization parameter.** Mean reconstruction errors for the four kinect sequences as a function of the regularization weight. For each sequence, we used the control vertex configuration that gave the best result in the experiment of Section 3.5.4.

3.6 Real-time Deformable Object Tracking

In this section, we will present our method to create a real-time deformable object tracking system that is fast enough to run on mobile devices. We will propose a fast robust outlier rejection algorithm and a simplified problem formulation, which together allows real-time performance. We will then discuss various techniques to exploit the frame-to-frame tracking to make the system faster and more robust.

3.6.1 Fast Robust Outlier Rejection in 2D

The method in Section 3.4 eliminates erroneous correspondences by iteratively solving Eq. 3.11. One of its main limitations is that it solves for a 3-D mesh, which involves depth ambiguity and a larger number of variables compared to a 2-D mesh. We propose to perform outlier rejection in 2D similarly to [Pilet et al., 2008] but our algorithm can work with both regular and irregular meshes and is much faster thanks to the linear subspace parameterization.

We represent the 2-D deformable surface by a 2-D triangular mesh and use the regularization matrix \mathbf{A} mentioned in Section 3.3.1 on the x and y components to regularize the mesh.

Unlike [Pilet et al., 2008], which requires a regular 2-D mesh and uses the squared

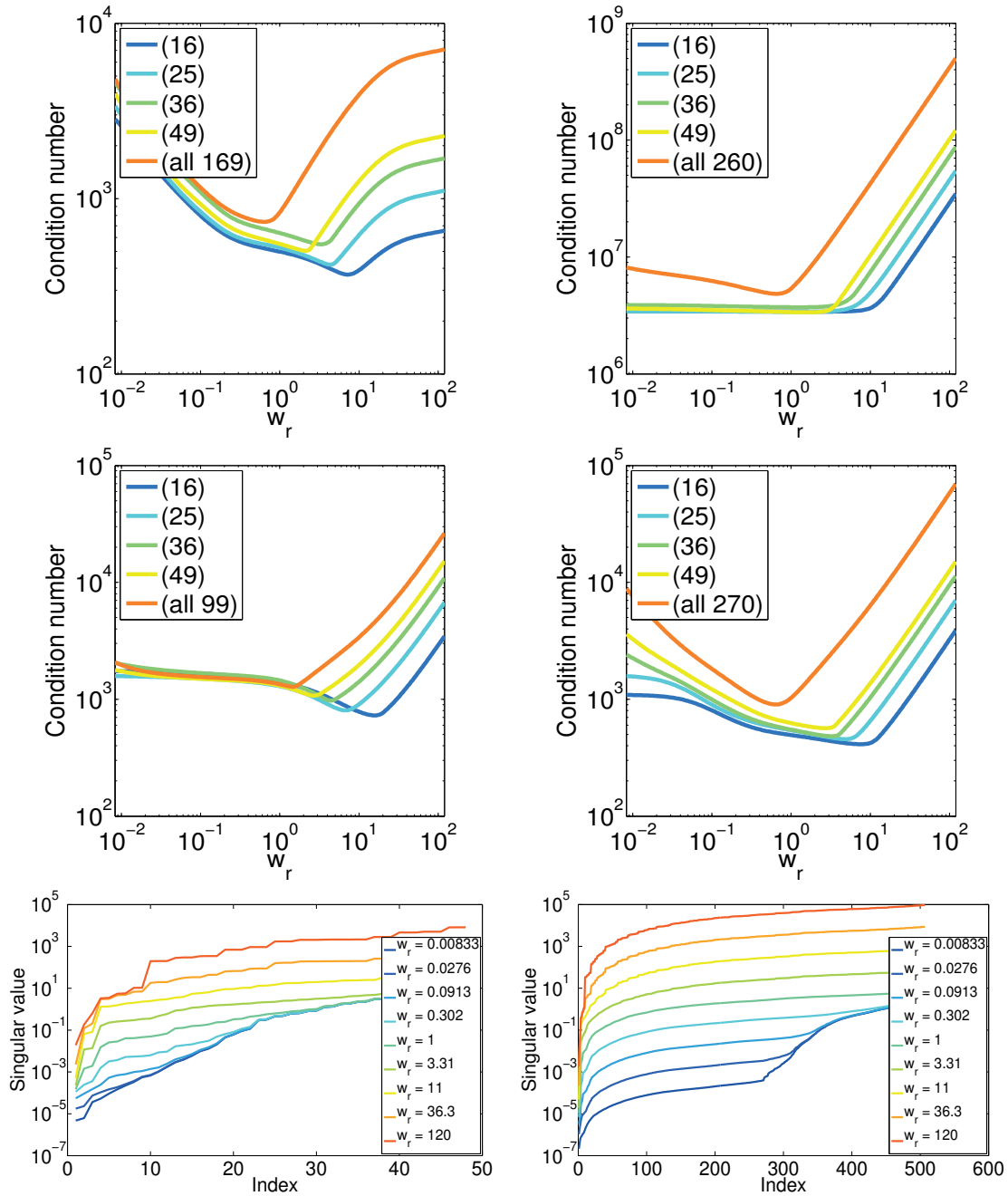


Figure 3.12: **Condition numbers and singular values.** **First two rows:** Plots of the condition number of the matrix \mathbf{M}_{w_r} of Eq. 3.12 as a function of w_r for each one of the four kinect sequences used to perform the experiments of Section 3.5.4. In each plot, we show one curve for each regular control vertex configuration of Fig. 3.8. Their respective labels denote the number of control vertices used. **Bottom row:** Singular values of \mathbf{M}_{w_r} in the apron case for two different configurations of the control vertices and several values of w_r . The curves for the three other cases look very similar.

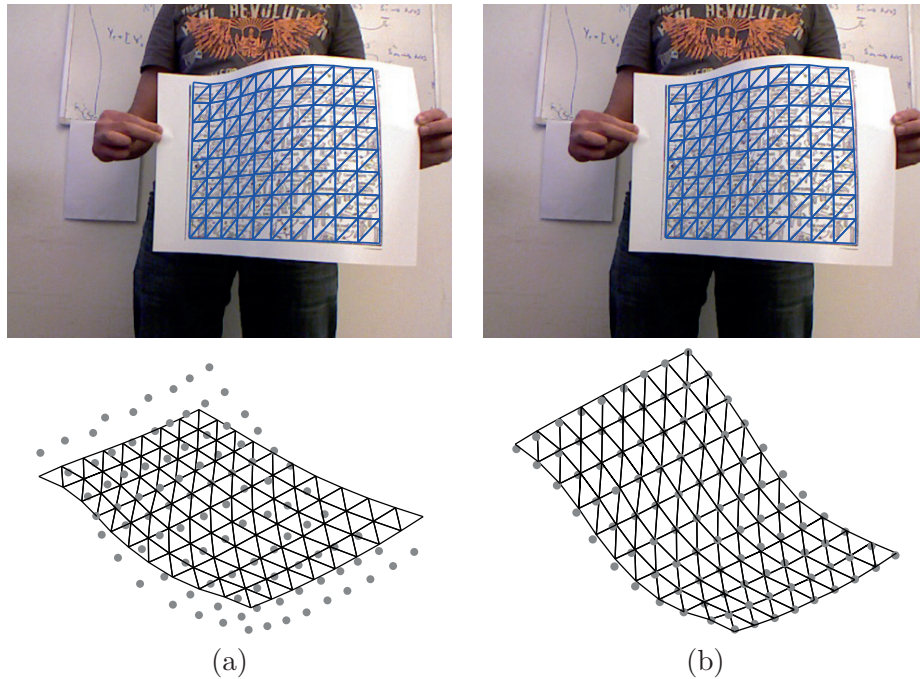


Figure 3.13: **Unconstrained vs constrained optimization results.** (a) The surface obtained by solving the unconstrained minimization problem of Eq. 3.11 and rescaling the result. It is projected on the original image at the top and shown from a different angle at the bottom. (b) The surface obtained by solving the constrained minimization problem of Eq. 3.13. The projections of both surfaces are almost identical but only the second has the correct 3-D shape.

directional curvature of the surface as the smoothing term, our regularization term can work on both regular and irregular meshes. We solve for a 2-D mesh which is smooth and matches the input image. The linear subspace parameterization $\mathbf{x} = \mathbf{P}\mathbf{c}$ still works on a 2-D triangular mesh and is used to reduce the complexity of the problem. We solve the following optimization problem:

$$\underset{\mathbf{c}}{\text{minimize}} \quad -\rho(\mathbf{B}\mathbf{P}\mathbf{c} - \mathbf{U}, r) + \lambda_s^2 \|\mathbf{A}\mathbf{P}\mathbf{c}\|^2, \quad (3.28)$$

where \mathbf{c} represents 2-D control vertices, \mathbf{A} is the regularization matrix, \mathbf{B} represents the barycentric coordinates of the feature points in matrix form, and \mathbf{U} encodes the feature point locations in the input image. Further, ρ is a robust estimator whose radius of confidence is r and is defined as

$$\rho(\delta, r) = \begin{cases} \frac{3(r^2 - \delta^2)}{4r^3} & -r < \delta < r \\ 0 & \text{otherwise} \end{cases} \quad (3.29)$$

Instead of introducing a viscosity parameter α and iteratively solving two coupled

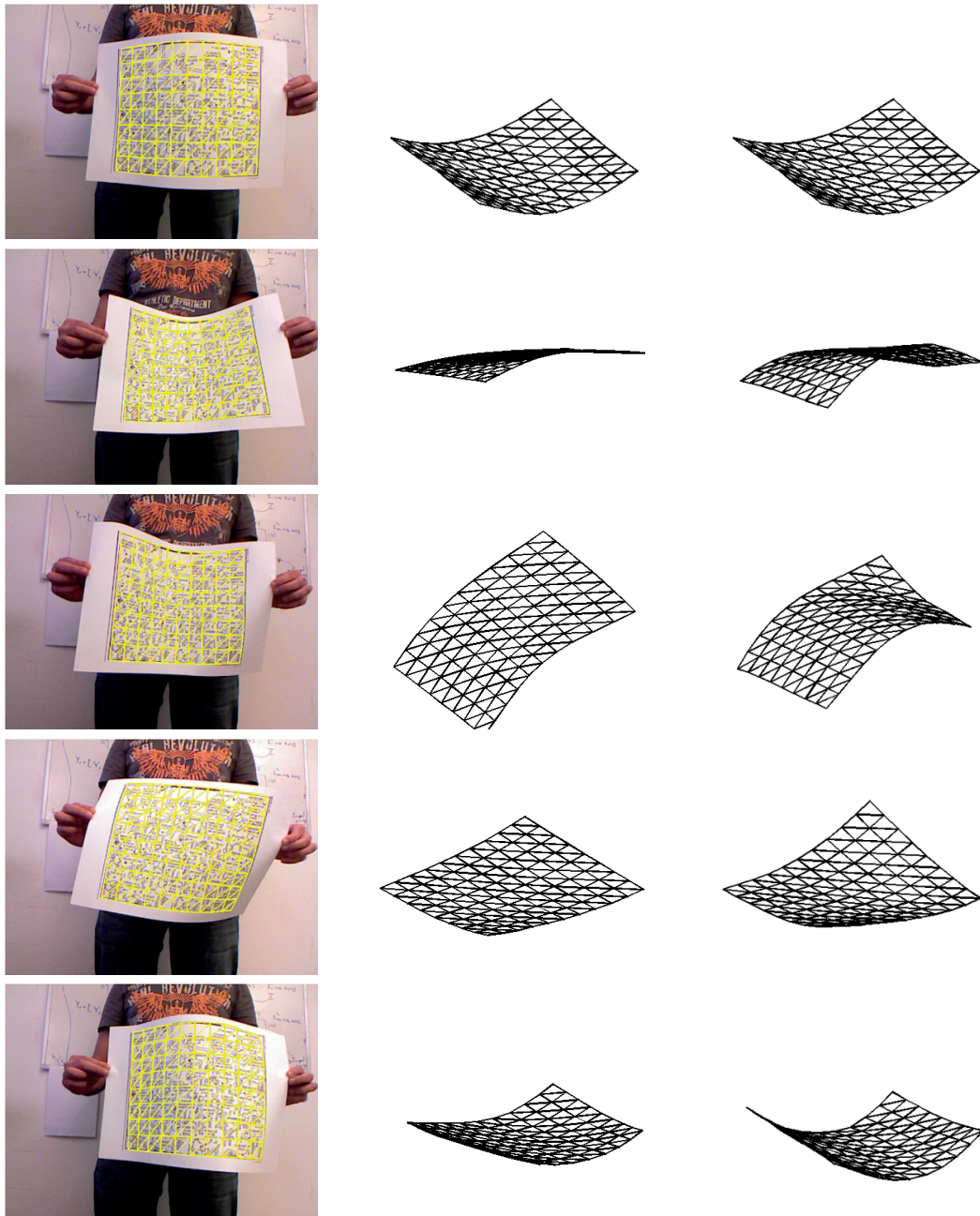


Figure 3.14: **Additional unconstrained and constrained results for the paper sequence.** **First column:** Reprojection on the input images of the final reconstructions obtained by solving the constrained minimization problem of Eq. 3.13. **Middle column:** Intermediate reconstructions obtained by solving the unconstrained minimization problem of Eq. 3.11 and seen from a different view-point. **Last row:** Final reconstructions seen from the same view-point as the intermediate ones. Note that the 3-D shapes can be quite different, even though their image projections are very similar.

equations with a random initial solution as in [Pilet et al., 2008], we solve Eq. 3.28 directly using a linear least squares approach with a big starting radius of confidence and reduce it by half at each iteration. The result of this iterative process is a both robust and very fast outlier rejection algorithm.

Note that our 2-D outlier rejection does not prevent us from tracking the surface in 3D. We use the obtained set of inlier correspondences to track and reconstruct the surface fully in 3D.

3.6.2 Surface Reconstruction by Detection

Once outlier correspondences are eliminated, we solve Eq. 3.11 only once and scale the solution to give an initialization for the constrained optimization in Eq. 3.13. We earlier formulate the inequality constraints $C(\mathbf{Pc}) \leq 0$ as equality constraints with additional slack variables whose norm is penalized to prevent lengths from becoming too small and the solution from shrinking to the origin. We solve a complex optimization problem involving extra variables and hard constraints:

$$\begin{aligned} \min_{\mathbf{c}, \mathbf{s}} \quad & \|\mathbf{MPc}\|^2 + w_r^2 \|\mathbf{APc}\|^2 + \mu^2 \|\mathbf{s}\|^2, \\ \text{s.t.} \quad & C(\mathbf{Pc}) + \mathbf{s}^2 = 0. \end{aligned} \tag{3.30}$$

We reformulate the problem by using soft constraints that allow the edge lengths to slightly vary around their reference lengths. By doing so, we remove the need to optimize the slack variables, thus obtain a simpler optimization problem with fewer variables and still arrive at sufficiently accurate reconstructions for Augmented Reality purposes. We solve the following optimization problem:

$$\min_{\mathbf{c}} \quad \|\mathbf{MPc}\|^2 + w_r^2 \|\mathbf{APc}\|^2 + \lambda^2 \|C(\mathbf{Pc})\|^2, \tag{3.31}$$

This is a least squares problem and we solve it iteratively using Gauss-Newton algorithm within a few iterations. The term $(\mathbf{AP})^T \mathbf{AP}$ is pre-computed only once for each deformable object. The term $(\mathbf{MP})^T \mathbf{MP}$ is also pre-computed for each input frame. These pre-computations greatly improve the efficiency of the optimization.

3.6.3 Surface Reconstruction by Tracking

In the tracking mode, we make use of the fact that both the surface shape and the camera pose change only slightly between two consecutive frames. We exploit this information to temporally regularize the solution and increase the pool of potential input-template feature point correspondences using KLT tracker or active search.

3.6.3.1 Temporal Consistency

We further use a motion model to temporally regularize the solution. Since the tracking frame rate is high, a linear motion model is enough. We solve

$$\min_{\mathbf{c}} \|\mathbf{M}\mathbf{P}\mathbf{c}\|^2 + w_r^2 \|\mathbf{A}\mathbf{P}\mathbf{c}\|^2 + \lambda^2 \|C(\mathbf{P}\mathbf{c})\|^2 + \gamma^2 \|\mathbf{c}_{t-2} - 2\mathbf{c}_{t-1} + \mathbf{c}\|^2, \quad (3.32)$$

in which \mathbf{c}_{t-1} and \mathbf{c}_{t-2} are solutions to previous two frames.

We also use this linear motion model to predict the shape for the current frame and use the result to initialize the reconstruction. Using the linear motion model, we can also predict the 3-D pose of the object in the next frame and create an occlusion mask where the surface projection should be in the next input image. This technique helps to speed up the feature point detection and matching. It also improves the robustness of the system because gross outliers are limited.

3.6.3.2 KLT-based Feature Point Tracking

Due to small camera movement, small change in object pose, and slowly changing lighting conditions, the object appearance changes very slightly between consecutive frames. We track inlier correspondence points on the input image from frame to frame on grayscale images by calculating an optical flow for a sparse set of inlier feature points using the iterative Lukas-Kanade algorithm with image pyramids [Bouguet, 1999]. The linear motion model is used to initialize the optical flow to obtain a better performance. We maintain the correspondences of these points to the template image to get a set of template-input image correspondences. This step brings a great advantage that allows the system to track under extreme tilts and large deformations, successfully. This step can also be done in parallel using GPU. The speed up can be a factor of 10 [Fassold et al., 2009].

As the point tracker algorithm progresses over time, points can be lost due to out of plane rotation, lighting variation, or occlusions, we may need to reacquire these points periodically. We will present a technique to deal with this problem in the next section.

3.6.3.3 Active Search

Wide-baseline feature points detection and matching can be used to retrieve lost tracked points of the KLT point tracker. Thanks to the robustness of feature points detection and description, matching feature points between the template and input images does not require any prior knowledge about the object pose and allows full 3-D reconstruction at every frame. However, using the expected locations of model points in the currently processed frame would reduce the required computation effort and increase the inlier

rate. Similar to [Klein and Murray, 2007, Wagner et al., 2008] in the context of rigid object tracking, we use the template image as source information to actively search for correspondences. For a feature point on the template, we know its appearance and want to find its corresponding location on the input image. However, due to object deformations and viewing change, its appearance on the input image may be very different. We exploit the tracking context in which the object deformation from one frame to the next is very small. Using the current reconstructed 3-D shape, we affinely warp the template to the view of the input image.

We define a warping function W that warps the template image to the input image by back-projection and perspective projection. The affine transform is computed using finite differences, with pre-computation of back-projection.

$$A = \begin{bmatrix} \frac{\partial W_u}{\partial u} & \frac{\partial W_v}{\partial u} & t_u \\ \frac{\partial W_u}{\partial v} & \frac{\partial W_v}{\partial v} & t_v \end{bmatrix} \quad (3.33)$$

Determinant of matrix A indicates the scale change of appearance of the feature point. We build a image pyramid of the template to account for aliasing effect when seeing the template from different distances. We extract a 9×9 template patch and search its corresponding patch by moving NCC in a 16×16 window. Sub-pixel accuracy can be optionally obtained by fitting a quadratic function over NCC responses at neighbouring pixels.

By combining point correspondences from both frame-to-frame KLT point tracker and active search, we get a set of correspondences with a high inlier rate, without drift, being robust to illumination, view point changes, and motion blurs. It allows the tracker to robustly work under various difficult settings.

3.6.4 Real-time Deformable Object Tracking Evaluation

In this section, we evaluate the accuracy, robustness, and timing of our proposed real-time deformable surface tracking system on mobile devices.

3.6.4.1 Robustness

We demonstrate the robustness of our outlier rejection scheme described in Section 3.6.1 for planar surfaces and compare it to our previous method in Section 3.4 and [Pilet et al., 2008]. To evaluate our approach, we used the same image from the standard paper dataset as in Section 3.5.2, in which the surface undergoes large deformations. We used both the corresponding Kinect point cloud and SIFT correspondences to reconstruct

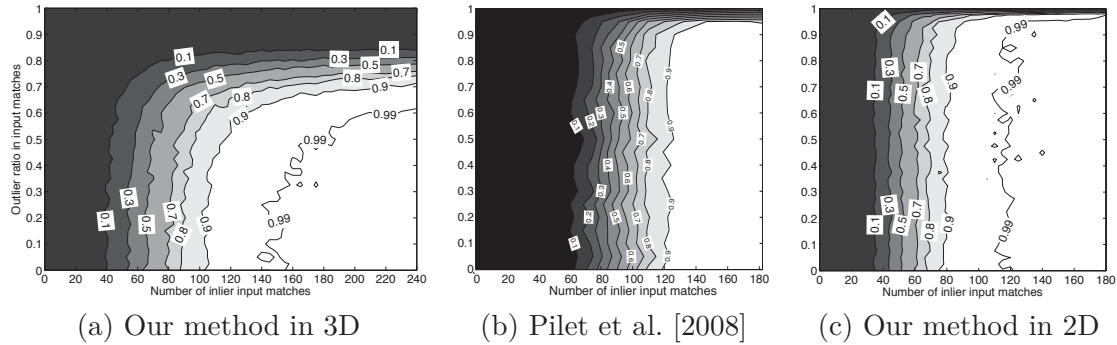


Figure 3.15: Probability of having at least 90 % of mesh vertices re-projected within 2 pixels of the solution as a function of the number of inlier matches and proportion of outliers, on the x-axis and y-axis respectively. The lines are level lines of the probability.

a mesh that we treated as the ground truth. Similar to [Pilet et al., 2008], we then produced approximately one million sets of correspondences by synthetically generating 10 to 180 inlier correspondences spread uniformly across the surface, adding a zero mean Gaussian noise ($\sigma = 1$ pixel) to the corresponding pixel coordinates to simulate feature point localization errors, and introducing proportions varying from 0 to 100 % of the randomly spread outlier correspondences.

We ran our proposed outlier rejection algorithm with 25 regularly sampled control vertices on each one of these sets of correspondences and defined a criteria to assess the effectiveness of our outlier rejection scheme. The criterion for success is that at least 90 % of the reconstructed 3-D mesh vertices project within 2 pixels of where they should. Fig. 3.15 depicts the success rates according to this criteria as a function of the total number of inliers and the proportion of outliers. The results reported in [Pilet et al., 2008] and the result of our earlier method in Section 3.4, which have similar experiment settings as ours, are included for comparisons. Note that unlike us, [Pilet et al., 2008] does not add Gaussian noise to the input correspondences. Nevertheless, it takes our method approximately only 80 inlier correspondences to guarantee that the algorithm will tolerate up to 0.95 ratio of outliers with 0.9 probability. The algorithm decays nicely with respect to the number of inlier input matches. It is still 50 % successful given only 50 inlier matches and a 0.95 outlier ratio. Compared to [Pilet et al., 2008] and method in Section 3.4, our proposed outlier rejection algorithm in 2D requires fewer inlier correspondences to detect the surface and can handle a larger portion of outliers.

Pilet et al. [2008] reports the total execution time of 100 ms per frame in a 2.8 GHz PC including feature point detection and matching time. Our outlier rejection algorithm is similar except that we solve a much smaller least squares problem. For the outlier rejection algorithm alone, on a MacBook Pro 2014 2.6 GHz laptop, it takes our earlier method in Section 3.4 about 25 ms per frame while it only takes our method about 2 ms per frame on the same input correspondences.


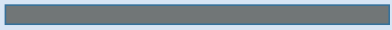

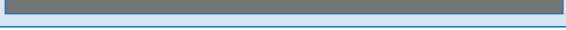

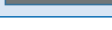

3.6.4.2 Reconstruction Accuracy

We compare the reconstruction accuracy of our method described in Section 3.6.2 with soft isometric constraints against the method described in Section 3.3 with exact inextensibility constraints. We used the same dataset as for the robustness evaluation (see the preceding section). Both methods use 25 regularly-placed control vertices. We used the Kinect point cloud and SIFT correspondences to build ground truth meshes, and compute the average vertex-to-vertex distance from the reconstructed mesh to the ground truth mesh as the measure of reconstruction accuracy. The average reconstruction accuracy of method in Section 3.3 is 2.83 mm while our new average reconstruction accuracy is 2.74 mm, which is comparable. However, on a MacBook Pro 2014 2.6 GHz laptop, it takes method in Section 3.3 about 70 ms per frame to solve the optimization problem in Eq. 3.30, while it takes our new method only 3.8 ms to solve the optimization problem in Eq. 3.32.

3.6.4.3 Performance

We measure the timing performance of our tracking system running on a first generation iPad Air tested with a deformable paper represented by a 11×13 triangle mesh and a deformable cushion represented by a 157-vertices triangle mesh. The timing patterns of the both cases are similar. We report the computation time for the main steps of our algorithm averaged over many frames in Table 3.2. It includes BRISK feature point detection, BRISK feature point description, BRISK feature point matching [Leutenegger et al., 2011], outlier rejection in 2-D, KLT point tracking between successive frames, active search for correspondences, and constrained optimization for 3-D shape reconstruction. We detect about 400 BRISK feature points lying on the deformable object on the template image. Feature points are described by binary BRISK descriptors. Brute-force search is used to match feature points between the template image and the input image accordingly to the Hamming metric.

Table 3.2: Computational time in milli-seconds of major steps of our tracking system measured on a first generation iPad Air

BRISK feature point detection		19 ms
BRISK feature point description		14 ms
BRISK feature point matching		15 ms
KLT inlier point tracking		19 ms
Active search		21 ms
Outlier rejection in 2-D		5 ms
Constrained optimization		12 ms

Chapter 3. Real-time 3-D Reconstruction and Tracking

Because the KLT point tracker gives good enough correspondences in a short sequence, we only apply active search once every 5 frames to gain frame rate. Overall, the tracking and 3-D reconstruction system runs at 25 frames per second, which is fast enough to allow real-time augmented reality applications on mobile devices. We believe further code optimization would significantly improve the frame rate.

Some screenshots of the real-time tracking system on a laptop and an iPad are depicted in Fig. 3.16 and 3.17.

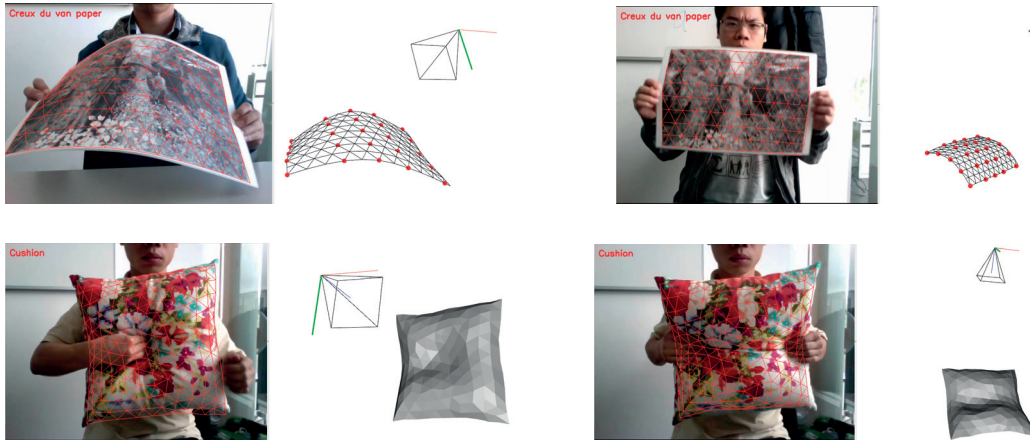


Figure 3.16: Real-time deformable surface reconstruction and tracking on a laptop.

3.7 Conclusion

This chapter has presented a novel approach to parameterizing the vertex coordinates of a mesh as a linear combination of a subset of them. In addition to reducing the dimensionality of the monocular 3-D shape recovery problem, it yields a rotation-invariant curvature-preserving regularization term that produces good results without training data or having to explicitly handle global rotations. Furthermore, it is particularly useful for reconstructing non-planar deformable objects, which tend to be flattened by previous methods.

We further applied the regularization and parameterization techniques to 2-D triangle meshes and obtained with a very fast way to eliminate erroneous correspondences, an essential step to make our deformable object 3-D reconstruction and tracking fast and robust. We have also presented our various techniques to build a real-time deformable surface tracking system that can run in real-time on mobile devices. It allows us to develop a wide range of real-world applications.

3.7. Conclusion

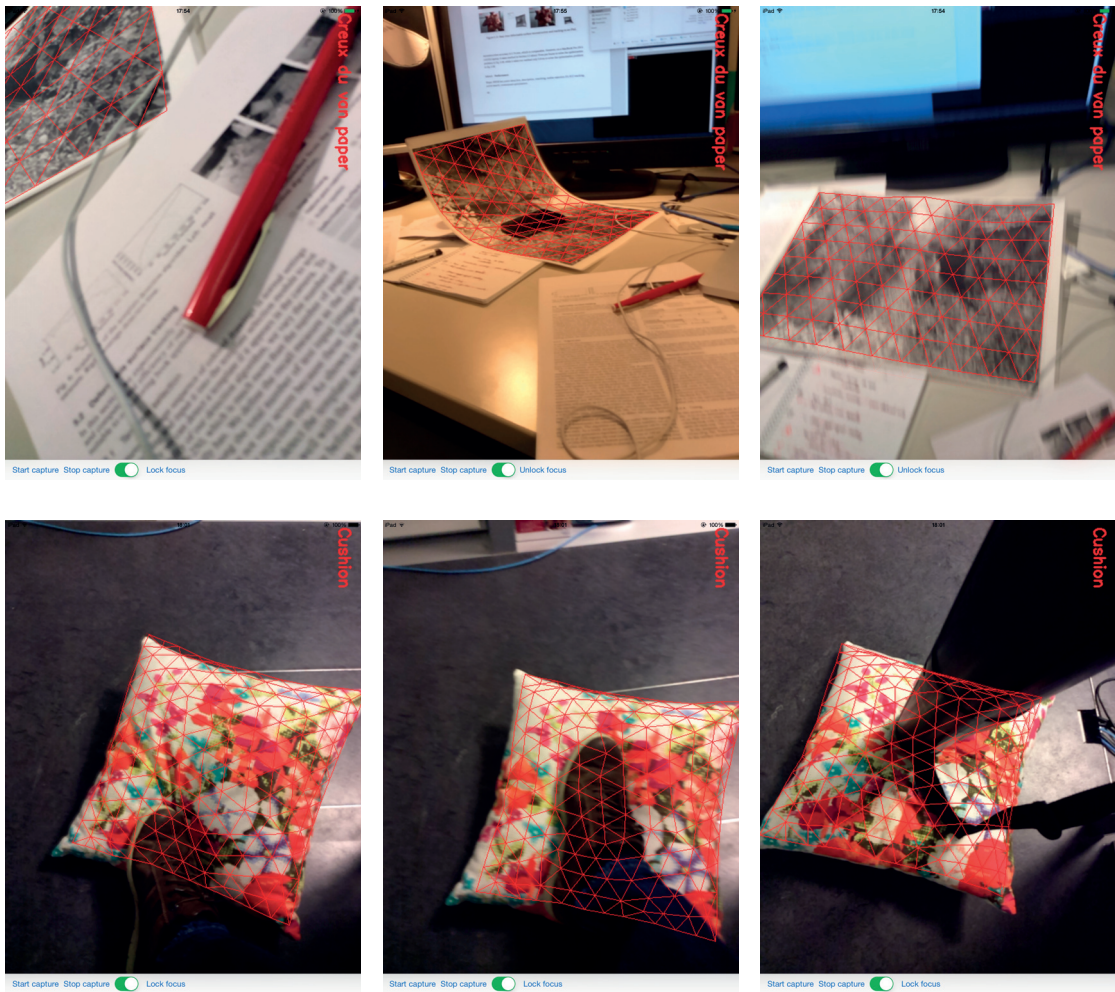


Figure 3.17: Real-time deformable surface reconstruction and tracking on an iPad. Our tracking algorithm can handle motion blurs and a reasonable amount of occlusion.

4 Dense image registration and 3-D shape reconstruction

Deformable surface tracking from monocular images is well-known to be under-constrained. Occlusions often make the task even more challenging, and can result in failure if the surface is not sufficiently textured. In this chapter, we explicitly address the problem of 3-D reconstruction of poorly textured, occluded surfaces, proposing a framework based on a template-matching approach that scales dense robust features by a relevancy score. Our approach is extensively compared to current methods employing both local feature matching and dense template alignment. We test on standard datasets as well as on new datasets of sparsely textured, occluded surfaces. Our framework achieves state-of-the-art results for both well- and poorly-textured, occluded surfaces.

4.1 Foreword

Depth ambiguities make monocular 3-D shape recovery highly under-constrained. Moreover, when the surface is partially occluded or has minimal texture, the problem becomes even more challenging because there is little or no useful information about large parts of it. When the surface is well-textured, correspondence-based methods have proved effective at solving this problem, even in the presence of occlusions [Bartoli et al., 2012, Bronte et al., 2014, Brunet et al., 2014, Chhatkuli et al., 2014, Perriollat et al., 2011, Pizarro and Bartoli, 2012, Vicente and Agapito, 2012, Ostlund et al., 2012, Salzmann and Fua, 2011]. In contrast, when the surface lacks texture, dense pixel-level template matching should be used instead. Unfortunately, many methods such as [Malti et al., 2011, Salzmann et al., 2008b] either are hampered by a narrow basin of attraction, which means they must be initialized from interest points correspondences, or require supervised learning to enhance robustness. Using Mutual Information has often been claimed [Dame and Marchand, 2012, Dowson and Bowden, 2006, Panin and Knoll, 2008, Viola and Wells, 1997] to be effective at handling these difficulties but our experiments do not bear this out. Instead, we advocate template matching over robust dense features that relies on a pixel-wise relevancy score pre-computed for each frame. As shown in Fig. 4.1,

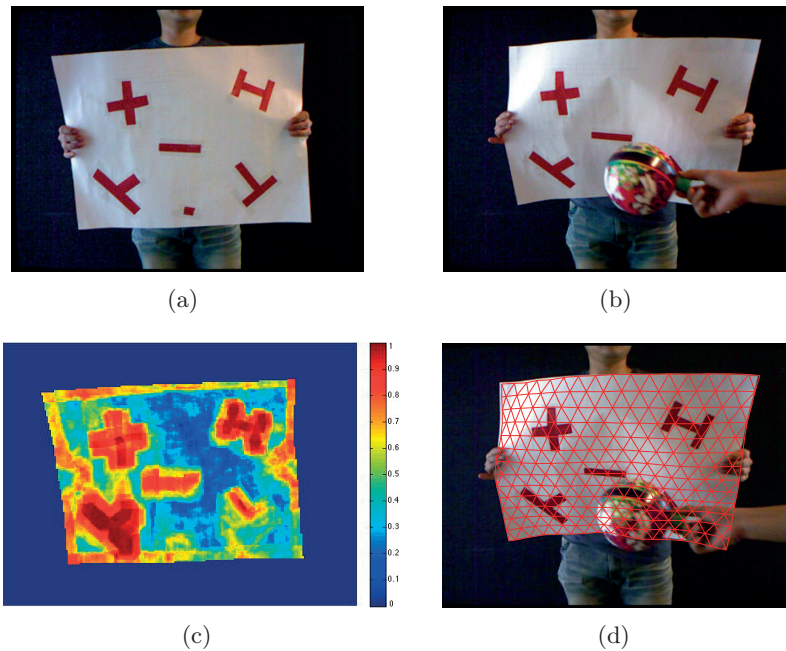


Figure 4.1: Tracking a sparsely textured surface in the presence of occlusion: (a) template image, (b) input image, (c) relevancy score, (d) surface tracking result with proposed framework.

our approach can handle occlusions and lack of texture simultaneously. Moreover, no training step is required in contrast to [Salzmann et al., 2008b], which we consider to be an advantage because this obligates either collecting training data or having sufficient knowledge of the surface physical properties, neither of which may be forthcoming.

Our main contribution is therefore a robust framework for image registration and monocular 3-D reconstruction of deformable surfaces in the presence of occlusions and minimal texture. A main ingredient is the pixel-wise relevancy score we use to achieve the robustness. We made the code publicly available, and released the dataset we used to validate our approach, which contains challenging sequences of sparsely-textured deforming surfaces and the corresponding ground truth.

4.2 Proposed Framework

In this work, we demonstrate that a carefully designed dense template matching framework can lead to state-of-the-art results in monocular reconstruction of deformable surfaces, without the need for any statistical learning step. In this section we describe our framework, based on a recently introduced gradient-based pixel descriptors [Crivellaro and Lepetit, 2014] for robust template matching and the computation of a relevancy score for outlier rejection.

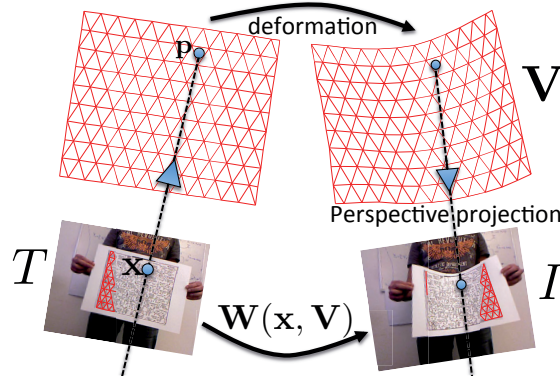


Figure 4.2: An image warping function maps a pixel from the template image onto the deforming surface in the input image.

4.2.1 Template Matching

We assume we are given both a template image T and the rest shape of the corresponding deformable surface, which is a triangular mesh defined by a vector of N_v vertex coordinates in 3D, $\mathbf{V}_T \in \mathbb{R}^{N_v \times 3}$. To recover the shape of the deformed surface in an input image I , the vertex coordinates \mathbf{V}_T of the 3-D reference shape must be adjusted so that their projection onto the image plane aligns with I .

We assume the internal parameters of the camera are known and, without loss of generality, that the world reference system coincides with the one of the camera. All RGB color images are converted to grayscale before feature descriptors are extracted. In order to register each input image, a pixel-wise correspondence is sought between the template and the input image. Each pixel $\mathbf{x} \in \mathbb{R}^2$ on the template corresponds to a point $\mathbf{p} \in \mathbb{R}^3$ on the 3-D surface. This 3-D point is represented by fixed barycentric coordinates which are computed by back-projecting the image location \mathbf{x} onto the 3-D reference shape.

The camera projection defines an image warping function $\mathbf{W} : \mathbb{R}^2 \times \mathbb{R}^{3 \times N_v} \rightarrow \mathbb{R}^2$ which sends pixel \mathbf{x} to a new image location based on the current surface mesh \mathbf{V} as illustrated in Fig. 4.2. The optimal warping function should minimize the difference between $T(\mathbf{x})$ and $I(\mathbf{W}(\mathbf{x}; \mathbf{V}))$, according to some measurement of pixel similarity. Traditionally, image intensity has been used, but more robust pixel feature descriptors $\phi_I(\mathbf{x})$ will lead to more meaningful comparisons, as discussed in Section 4.2.2.2.

The image energy cost function is a comparison between $\phi_T(\mathbf{x})$ and $\phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V}))$ at every image point \mathbf{x} defining the quality of their alignment

$$E_{\text{image}}(\mathbf{V}) = \sum_{\mathbf{x}} d(\phi_T(\mathbf{x}), \phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V}))). \quad (4.1)$$

There are many possible choices for the function d comparing the descriptor vectors, such

as Sum of Squared Differences (SSD), NCC, MI, and others. We will discuss more in detail about the choice of d in Section 4.2.2.3.

Since monocular 3-D surface reconstruction is an under-constrained problem and there are multiple 3-D shapes having the same reprojection on the image plane, minimizing the image energy in Eq. (4.1) alone is ill-posed. Additional constraints must be added, such as isometric deformation constraints enforcing that the surface should not stretch or shrink. A change in the length between vertex \mathbf{v}_i and vertex \mathbf{v}_j as compared to the template rest length l_{ij} from \mathbf{V}_T is penalized as

$$E_{\text{length}}(\mathbf{V}) = \sum_{i,j} (\|\mathbf{v}_i - \mathbf{v}_j\| - l_{ij})^2. \quad (4.2)$$

To encourage physically plausible deformations, the Laplacian mesh smoothing proposed in the previous chapter is used. This rotation-invariant curvature-preserving regularization term based on Laplacian smoothing matrix \mathbf{A} penalizes non-rigid deformations away from the reference shape, based on the preservation of affine combinations of neighboring vertices.

$$E_{\text{smooth}}(\mathbf{V}) = \|\mathbf{A}\mathbf{V}\|^2. \quad (4.3)$$

To reconstruct the surface, we therefore seek the mesh configuration \mathbf{V} that minimizes the following total energy:

$$\arg \min_{\mathbf{V}} E_{\text{image}}(\mathbf{V}) + \lambda_L E_{\text{length}}(\mathbf{V}) + \lambda_S E_{\text{smooth}}(\mathbf{V}), \quad (4.4)$$

for relative weighting parameters λ_L and λ_S .

4.2.2 Robust Optimization

4.2.2.1 Optimization Scheme

To make the optimization more robust to noise and wide pose changes, we employ a multi-scale approach, iteratively minimizing $E^\sigma = E_{\text{image}}^\sigma + \lambda_L E_{\text{length}} + \lambda_S E_{\text{smooth}}$ for decreasing values of a scale parameter σ , with:

$$E_{\text{image}}^\sigma = \sum_{\mathbf{x}} d(G^\sigma * \phi_T(\mathbf{x}), G^\sigma * \phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V}))), \quad (4.5)$$

where G^σ is a low-pass Gaussian filter of variance σ^2 . In our experiments we solve the alignment at three scales, using the final result of each coarser scale to initialize the next set of iterations, and initializing the coarsest scale with the final position found for the previous frame. The first frame of each image sequence is taken as the template, and we

employ a standard Gauss-Newton algorithm for minimization.

4.2.2.2 Feature Selection

The image information compared in Eq. (4.1) comes from pixel-based image features. Previous approaches [Malti et al., 2011, Pilet et al., 2007, Salzmann et al., 2008b] employ image intensity as a local descriptor, $\phi_I(\mathbf{x}) = I(\mathbf{x})$. More robust results can be obtained with other features, such as the lighting-insensitive image gradient direction (GD) [Gopalan and Jacobs, 2010], where

$$\phi_I(\mathbf{x}) = \tan^{-1} \frac{I_y(\mathbf{x})}{I_x(\mathbf{x})} \quad (4.6)$$

with $\bmod 2\pi$ differencing. Based on its strong previous performance we also consider the Gradient Based Descriptor Fields (GBDF) recently proposed in [Crivellaro and Lepetit, 2014]:

$$\phi_I(\mathbf{x}) = \left[\left[\frac{\partial I}{\partial x}(\mathbf{x}) \right]^+, \left[\frac{\partial I}{\partial x}(\mathbf{x}) \right]^-, \left[\frac{\partial I}{\partial y}(\mathbf{x}) \right]^+, \left[\frac{\partial I}{\partial y}(\mathbf{x}) \right]^- \right]^\top, \quad (4.7)$$

where the $[\cdot]^+$ and $[\cdot]^-$ operations respectively keep the positive and negative values of a real-valued signal. These descriptors are robust under light changes, and remain discriminative after the Gaussian smoothing employed in Eq. (4.5). However, as originally proposed in [Crivellaro and Lepetit, 2014], they are not rotation invariant. To achieve in-plane rotation invariance, in our final framework we employ a modified version of GBDF. In order to compare pixel descriptors in the same, unrotated coordinate system, the reconstruction of the previous frame is used to establish a local coordinate system for each mesh facet. Each pixel descriptor on the template is then rotated in accordance with its corresponding mesh facet, to be directly comparable to the points in the input image. We show in Section 4.3 that this modification indeed increases registration accuracy by being able to successfully track a rotating deformable surface.

4.2.2.3 Similarity Function Selection

Choosing the correct comparison function d for Eq. (4.1) also significantly affects the robustness of the tracking. Common choices include the SSD of the descriptors, and the NCC of image intensities [Lewis, 1995], which is invariant under affine changes in lighting.

$$E_{\text{imageSSD}}(\mathbf{V}) = \sum_{\mathbf{x}} \|\phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V})) - \phi_T(\mathbf{x})\|^2, \quad (4.8)$$

$$E_{\text{image}_{\text{NCC}}}(\mathbf{V}) = \frac{\sum_{\mathbf{x}} (\phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V})) - \bar{\phi}_I) \cdot (\phi_T(\mathbf{x}) - \bar{\phi}_T)}{\sqrt{\sum_{\mathbf{x}} (\phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V})) - \bar{\phi}_I)^2} \sqrt{\sum_{\mathbf{x}} (\phi_T(\mathbf{x}) - \bar{\phi}_T)^2}} \quad (4.9)$$

Mutual Information: MI [Viola and Wells, 1997] is a similarity function that measures the amount of information shared between two variables, and it is known to be robust to outliers such as noise and illumination changes [Dame and Marchand, 2012]. It has been repeatedly claimed to be robust to occlusion, for example in [Dame and Marchand, 2012, Dowson and Bowden, 2006, Panin and Knoll, 2008, Viola and Wells, 1997]. Where occlusions occur, the shared information between occluded pixels and the template image is low or none, and its variation does not cause significant change in the image entropy; therefore, the MI obtains an accurate maximum value at the position of the correct alignment, in spite of the occlusion. However, an MI-based cost function is limited in application. MI generally provides a non-convex energy function with a very strong response at the optimum, but a very narrow basin of convergence, as shown in Fig. 4.5. This makes it unsuited for direct numerical optimization, while smoothing leads to a significantly degraded energy function. Numerical experiments reported in Section 4.3 show that, in our context, MI leaves room for improvement.

Robust Statistics: M-estimators are a popular method for handling outliers in a template matching framework. Let $e_i = \phi_I(\mathbf{W}(\mathbf{x}_i; \mathbf{V})) - \phi_T(\mathbf{x}_i)$ be the residual at pixel \mathbf{x}_i ; then instead of minimizing the sum of squared residuals $\sum_i e_i^2$, a modified loss function ρ of the residuals is considered, instead minimizing $\sum_i \rho(e_i)$, in order to reduce the influence of outliers.

In Section 4.3, tests are performed using two of the most commonly employed M-estimators, the Huber [Huber, 1981] and the Tukey [Hoaglin et al., 1983] estimators. The Huber loss function is given by:

$$\rho^{\text{Hu}}(e) = \begin{cases} \frac{1}{2}e^2 & \text{for } |e| \leq \tau, \\ \tau(|e| - \frac{1}{2}\tau) & \text{otherwise.} \end{cases} \quad (4.10)$$

This function is quadratic for small values of e , but only increases linearly for large error residuals, reducing their influence.

The Tukey bisquare loss function is defined by:

$$\rho^{\text{Tu}}(e) = \begin{cases} \frac{\tau^2}{6}(1 - (1 - (e/\tau)^2)^3) & \text{for } |e| \leq \tau, \\ \tau^2/6 & \text{otherwise.} \end{cases} \quad (4.11)$$

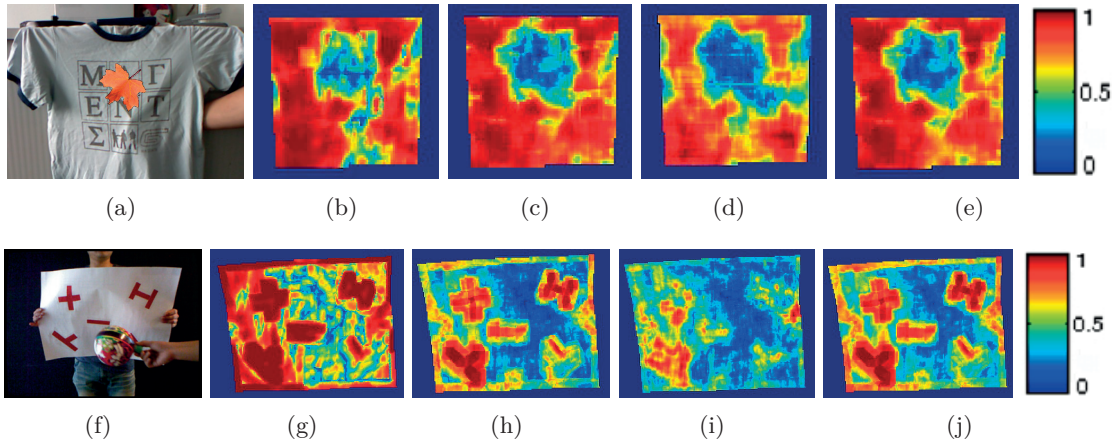


Figure 4.3: The relevancy score results using various methods on the (a) cloth dataset and (f) sparsely textured paper dataset using (b)(g) Intensity (c)(h) GBDF (d)(i) Gradient direction (e)(j) GBDF+Intensity. GBDF gives a better relevancy map than intensity and gradient direction on the first dataset while intensity is better than GBDF and gradient direction on the second. We therefore combine both intensity and GBDF in our proposed relevancy score.

The Tukey loss function suppresses the influence of points with large error residuals by re-assigning their residuals to a constant weight. Notice that Tukey loss function is not convex, but it reduces the effect of outliers more significantly than the Huber loss.

In our context, M-estimators show moderate efficacy, likely because part of the useful information is rejected as outliers. This problem becomes particularly significant when dealing with low-textured surfaces, where the amount of information available for alignment is low.

4.2.3 Handling Oclusions with a Relevancy Score

Our experiments suggest that selecting a robust similarity function is not enough to deal with the oclusions and image variability encountered when attempting to track a deforming surface in real-world imagery.

Inspired by the effectiveness of the occlusion masks developed in works such as [Strecha et al., 2006, Wang et al., 2009, Zhou et al., 2009], we derive a more robust method to handle occlusions by pre-computing a relevancy score for each pixel of the current frame, which is then used to weight the pixels during the alignment. Since we would like to handle occlusions and sparsely textured surfaces together, rather than designing a binary occlusion prediction mask, we develop a continuous-valued score that will raise or lower the importance of pixels depending on their relevancy. This pre-processing step can greatly improve the quality of the image information handed to the cost function in

Eq. (4.4).

Given the estimated configuration \mathbf{V}_{t-1}^* of the deformable surface from the previous frame, a thin-plate spline-based warping function [Bookstein, 1989] is used to un-warp image I_t to closely align with the template T . A relevancy score is then computed between each pixel \mathbf{x} on the synthetically back-warped image \hat{I}_t and the same pixel on the template T , with a sliding-window approach.

It has been verified repeatedly in the literature that NCC is a reliable choice for measuring patch-based image similarity, and so we compute the NCC over the images as an efficient prediction of relevancy. In one such approach [Klein and Murray, 2007] that performs Simultaneous Localization and Mapping (SLAM), local image patches are affinely warped based on the predicted camera pose, and sliding NCC windows are then used to look for correspondences of map points in the input image. Our approach is somewhat different, as we use sliding NCC to measure the relevancy of template pixels on the input image. We average the NCC of both the image intensity and the GBDF features, as it was found that both descriptors provide relevant and often complementary information at this predictor stage, (see Fig. 4.3 for a qualitative comparison).

The sliding relevancy score is computed as the maximum NCC value over a range of patches near image location \mathbf{x} :

$$\omega(\mathbf{x}) = \max_{\delta} \text{NCC}(\mathcal{P}_T(\mathbf{x}), \mathcal{P}_{\hat{I}}(\mathbf{x} + \delta)), \quad (4.12)$$

where $\mathcal{P}_T(\mathbf{x})$ and $\mathcal{P}_{\hat{I}}(\mathbf{x})$ are patches of size 26×26 centered at \mathbf{x} , $\delta = [\delta_x, \delta_y]^T$, and δ_x, δ_y vary over $[-30, 30]$ in all our experiments. Allowing the patch to be compared to nearby patches accounts for some of the variability between the surface position \mathbf{V}_{t-1}^* and the desired position \mathbf{V}_t^* to be recovered.

The similarity scores are then normalized to lie in $[0, 1]$, and outlier data is also limited at this stage in a process similar to an M-estimator. The mean μ and standard deviation σ of the NCC scores are found for each frame, and all values further than 3σ from the mean are clamped to the interval $\mu \pm 3\sigma$. These values are then linearly rescaled to lie between 0 and 1, and the normalized weights $\hat{\omega}$ are applied to the data in the image energy term of Equation (4.4):

$$E_{\text{image}}(\mathbf{V}) = \sum_{\mathbf{x}} \hat{\omega}(\mathbf{x}) d(\phi_T(\mathbf{x}), \phi_I(\mathbf{W}(\mathbf{x}; \mathbf{V}))), \quad (4.13)$$

where the sum here is extended to all the pixels of the template. Relevancy scores for the well-textured paper dataset are shown in Fig. 4.4.

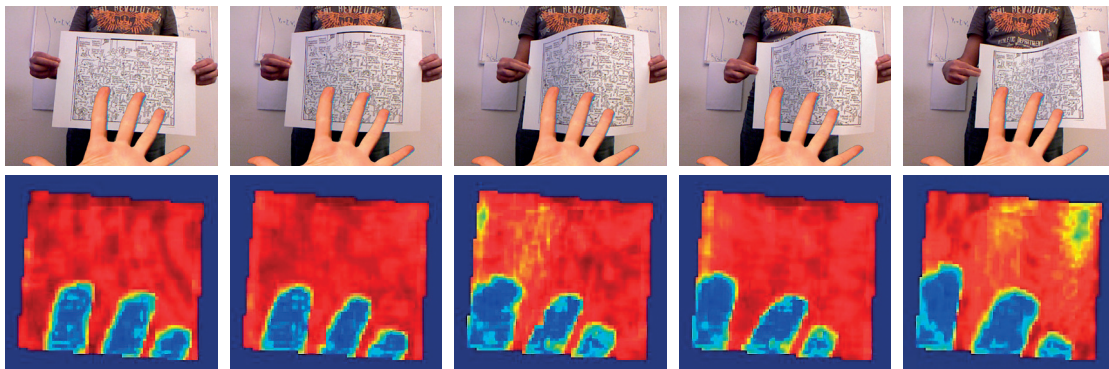


Figure 4.4: Relevancy scores for the well-textured paper dataset.

4.2.4 Handling Sparsely Textured Surfaces

The relevancy score described in Section 4.2.3 is also able to handle sparsely textured surfaces. Image regions containing little or no texture have low relevancy scores, so these pixels will not negatively influence the image alignment. For example, see Fig. 4.1. Using the proposed relevancy score to weight the utility of the image information coming from each pixel in the image allows the optimization to be driven by the most meaningful available information.

4.3 Experiments and Results

3-D surface reconstructions are computed with and without occlusion on both well and poorly textured deforming surfaces. We compare recent methods described in Section 2.2, which are representative of the current state-of-the-art, against our dense template matching-based reconstruction methods using the various similarity measures and occlusion handling techniques described in Section 4.2.

In particular, we report detailed results of comparisons with the following methods: “Bartoli12” [Bartoli et al., 2012], that reconstructs the surface by analytically solving a system of PDEs starting from an estimated 2D parametric warp between images; “Chhatkuli14” [Chhatkuli et al., 2014], that infers the surface shape exploiting the depth gradient non-holonomic solution of a PDE; “Brunet14” [Brunet et al., 2014], that reconstructs a smooth surface imposing soft differential constraints of isometric deformation; “Ostlund12” [Ostlund et al., 2012], that introduces the Laplacian mesh smoothing we employ; and “Salzmann11” [Salzmann and Fua, 2011], that uses pre-learned linear local deformation models.¹

As for pixel-based template matching techniques, comparing pixel intensity values “Intensity” and gradient direction values “GD” are done using SSD. We also compare

¹Code provided by the authors of these papers was used for all comparisons.

standard “NCC” and “MI” over intensity values. The “GBDF” features are compared using SSD, and were seen to be the strongest feature descriptor, so it is these values that we test in the M-estimator framework using the “Huber” and “Tukey” loss functions. Our proposed framework from Section 4.2.3 is labeled “GBDF+Oc” in the figures. We see that it achieves state-of-the-art performances on a standard, well-textured dataset, and it achieves optimal reconstruction performance in all datasets with occlusions and low texture.

Image sequences were acquired using a Kinect camera, and ground truth surfaces were generated from the depth information. The template is constructed from the first frame, and 3-D reconstruction is performed for the rest of the sequence using the image information alone. The initial mesh coordinates for each frame are set to the locations of the final reconstruction of the previous frame in the sequence.

We consider two different metrics to define the reconstruction accuracy. Many previous methods compare the average distance of the reconstructed 3-D mesh vertices to their closest projections onto the depth images. This metric ignores the correspondences between the mesh points and the point cloud. As a more meaningful metric, we use the Kinect point cloud to build ground truth meshes, and compute the average vertex-to-vertex distance from the reconstructed mesh to the ground truth mesh. We report results using both of the vertex-to-vertex and vertex-to-point-cloud metrics.

To ensure a fair comparison, all results are presented using the best parameter values found for each method, tuned separately. To ensure that our results are not overly sensitive to the selection of parameters λ_L and λ_S , we performed the full reconstruction on the well-textured paper dataset over a wide range of values, as presented in Table 4.1. It can be observed that increasing or decreasing these parameters by a factor of two around $\lambda_L = 1$ and $\lambda_S = 0.25$ results in very little change in the final reconstruction accuracy, implying that the method is sufficiently insensitive to these parameters as long as they are within a reasonable range.

The surface rest shape is modeled by a 10×13 triangle mesh in the well-textured dataset, 14×17 in the sparsely textured dataset, and 15×14 on the t-shirt dataset. The σ s used in the hierarchical procedures were $\{15, 7, 3\}$ and $\{5, 3, 2\}$.

Our approach relies on frame-to-frame tracking and thus requires a sufficiently good initialization. However, because the method has a wide basin of convergence, a rough initialization suffices. Our method can fail when the initialization is too far from the solution, when frame-to-frame deformations are so large that the relevancy scores stop being reliable, or when large changes in surface appearance and severe occlusions cause the image energy term to become uninformative. If the tracking is lost, it must be reinitialized, for example by using a feature point based method. However, this did not prove to be necessary to obtain any of the results shown below.

Table 4.1: Reconstruction errors over a range of weighting coefficient values using the well-textured paper dataset.

error (mm)		λ_L							
		10	2	1	0.5	0.25	0.1	0.05	0.01
λ_S	10	7.46	6.46	5.73	5.59	5.31	18.45	93.07	N.A
	2	4.60	1.58	2.41	3.68	4.49	5.17	17.23	319.09
	1	1.93	1.39	1.20	1.97	3.44	4.61	5.94	147.96
	0.5	2.02	1.73	1.43	1.08	1.80	3.76	4.77	61.87
	0.25	2.01	1.86	1.67	1.45	1.10	2.17	3.76	26.18
	0.1	2.05	1.91	1.75	1.62	1.57	1.20	1.68	240.66
	0.05	2.07	2.03	1.96	1.86	1.82	5.62	14.86	185.47
	0.01	18.23	15.44	6.25	6.45	6.31	10.44	14.11	342.12

4.3.1 Basin of Convergence

To understand the limitations of the various cost functions, we conducted a simple alignment experiment to test their respective sensitivities to initial position and image distractors; results are presented in Fig. 4.5. The red image window in the input image is translated in x and y about the known best alignment to the green template window, and the cost to compare each window pair is plotted, to allow the basins of convergence to be inspected visually. MI and NCC both reach a maximum value close to 1 at the point of best alignment, but we invert these functions so that a minimum cost of all functions is expected at the point of best alignment.

The GBDF descriptors from Eq. 4.7 are seen to have a strong minimum at the point of best alignment, with a reasonably wide, smooth basin of convergence, the desired property of a good cost function. However, Intensity, MI, and NCC all have several nearby local minima. Mutual Information is further seen to have a very narrow basin of convergence around the correct point of best alignment, meaning that it is very likely to converge to an incorrect alignment given an imperfect initial position.

This experiment only tests translation sensitivity because this is the variation best understood visually, but the similar results are likely from other types of misalignment.

4.3.2 Well-Textured Surfaces

We performed experiments using the well-textured paper dataset presented in [Varol et al., 2012] consisting of 193 consecutive images, for example see Fig. 4.10. Quantitative results are presented in Fig. 4.6 and Fig.4.7. For this well-textured dataset, all the feature point-based methods work well and dense matching methods are only slightly better. The biggest errors are due to lighting changes, where intensity features using SSD occasionally fail to track part of the surface, and hence have a higher error.

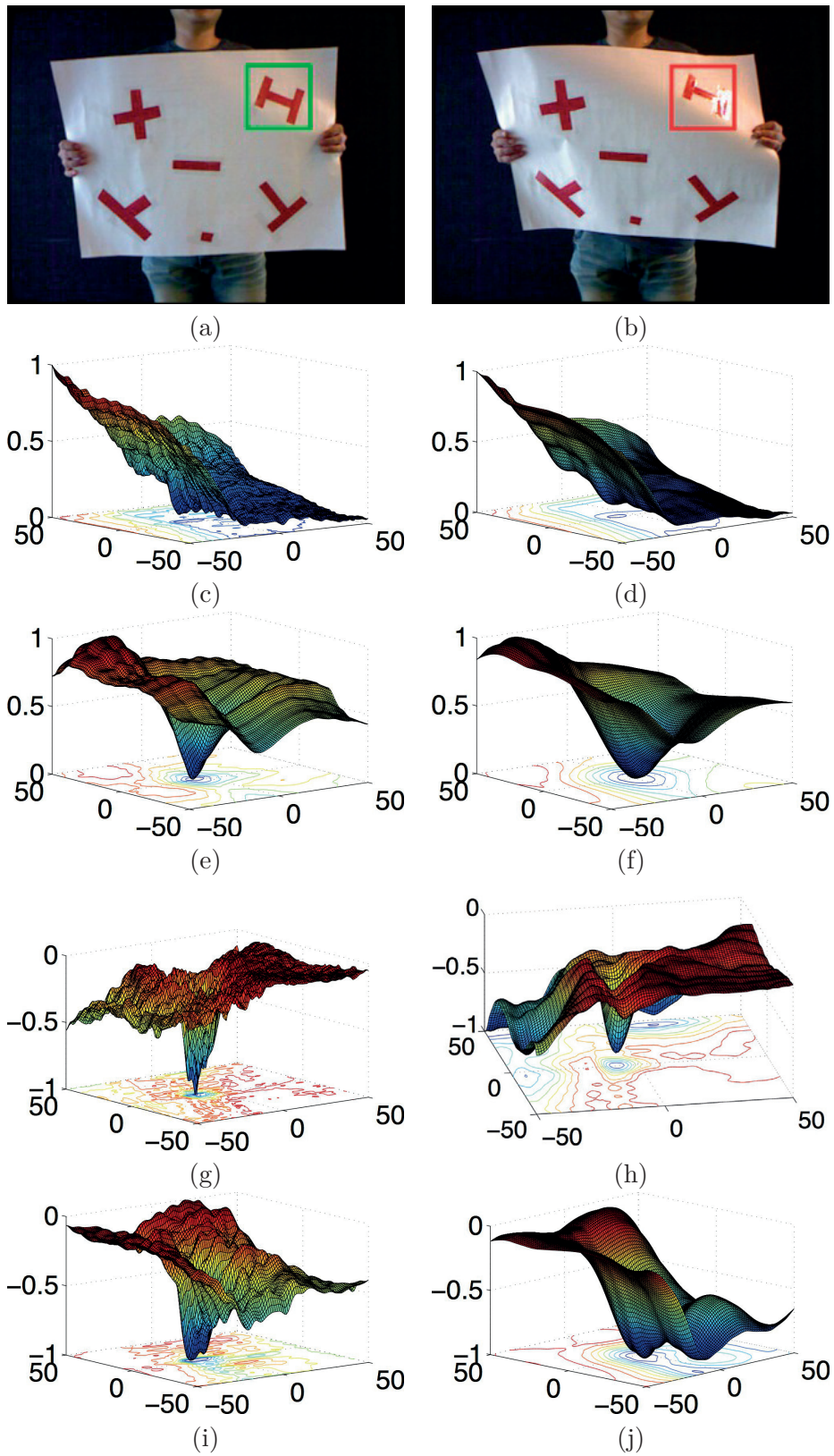


Figure 4.5: Robustness of alignment functions *w.r.t.* translations between (a) template, and (b) input image, showing the basin of convergence of the alignment costs around the correct position using **Left column:** weak Gaussian smoothing, **Right column:** strong Gaussian smoothing, over (c)(d) Intensity, (e)(f) GBDF, (g)(h) MI, and (i)(j) NCC.

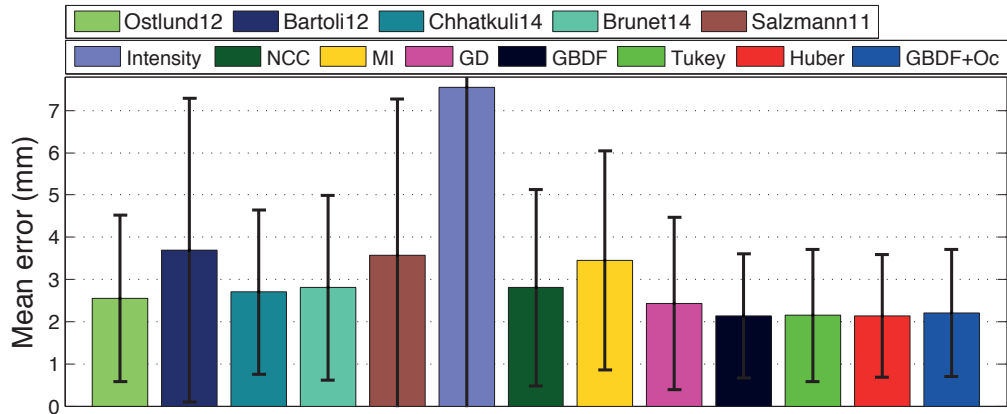


Figure 4.6: Reconstruction results computed using the vertex-to-vertex error metric on the well-textured paper dataset without occlusions. All feature-points based methods work reasonably well.

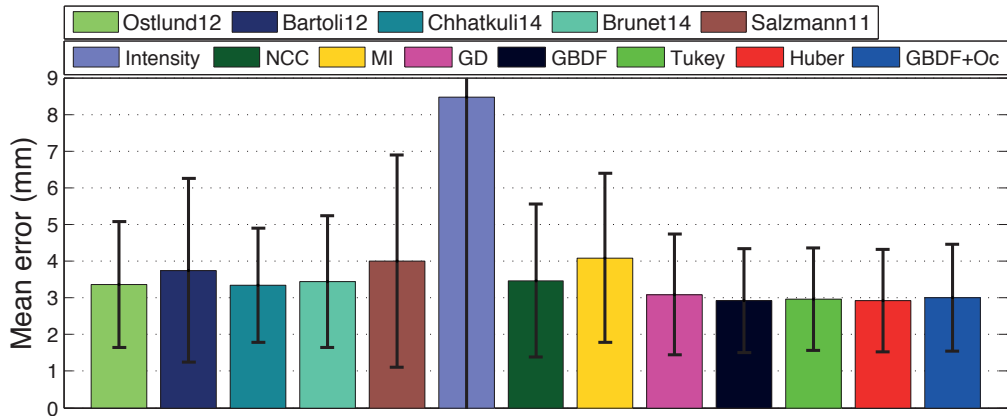


Figure 4.7: Reconstruction results computed using the vertex-to-cloud error metric on the well-textured paper dataset without occlusions.

To evaluate the robustness of each method to occlusion, we add artificial hand image occlusions to the image sequence. The reconstruction results are presented in Fig. 4.8 and Fig. 4.9. Feature-based methods still produce reasonably good 2D reprojection results in this dataset, but the recovered depths under the occlusion are not very accurate. Fig. 4.10 provides the output for a single frame where it can be seen that the reconstruction fails when using the strong GBDF without occlusion handling and also when using M-estimators to attempt to handle occlusion, while the proposed framework is still able to track the surface accurately. In this situation, Mutual Information and both the Tukey and Huber M-estimators are confused by the edges created by the finger and converge to incorrect locations.

We also demonstrate that the proposed rotation handling technique described in Section 4.2.2.2 that overcomes the rotation sensitivity of the GBDF descriptors can suc-

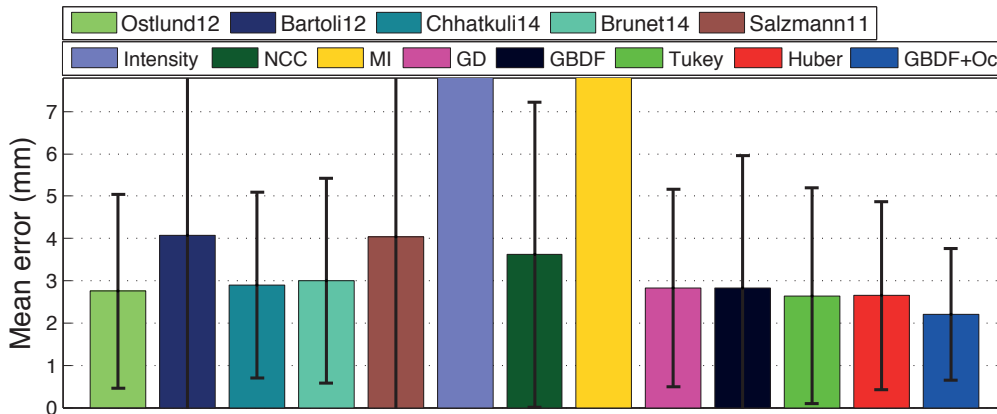


Figure 4.8: Reconstruction results computed using the vertex-to-vertex error metric on the well-textured paper dataset, with occlusions. Feature-based methods are largely robust to occlusion, however the overall depths recovered are not as accurate as the proposed framework that includes occlusion handling.

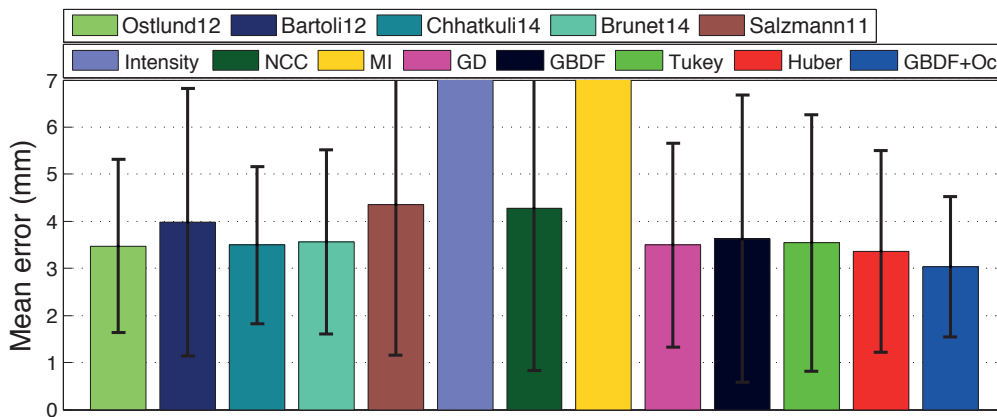


Figure 4.9: Reconstruction results computed using the vertex-to-cloud error metric on the well-textured paper dataset, with occlusions.

cessfully track a rotating deformable object. We ran our tracking algorithm with and without rotation handling on the well-textured paper dataset with images rotated 5 degree every frame. This rotation in addition to original surface deformations introduce large frame-to-frame motions. Fig. 4.11 shows that without rotation handling, the original GBDF descriptors can only track up to 50 degrees of rotation, while the proposed rotation handling technique can track the whole sequence including a full 360 degrees of rotation.

4.3.3 Sparsely-Textured Surfaces

To understand the performance of the methods in a realistic, sparsely textured setting, a different dataset is required. A less textured paper dataset exists, as published in [Salzmann et al., 2008b], but no ground truth information is available for this dataset

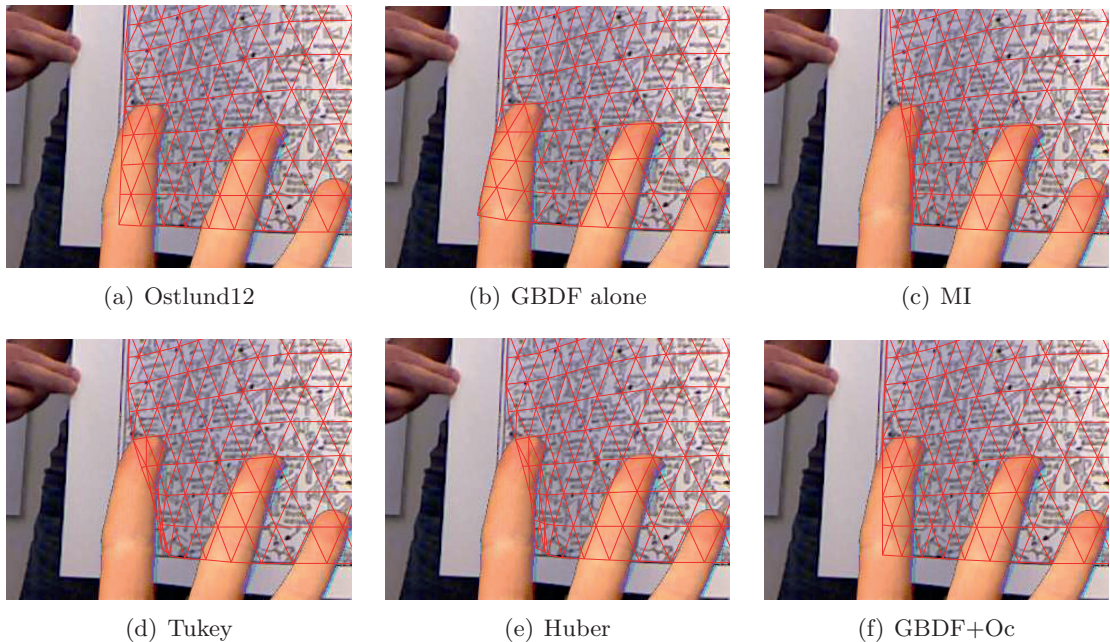


Figure 4.10: Output for a single frame showing relative reconstruction accuracies. Mutual Information and M-estimators fail to correctly handle the occlusion, while the proposed framework is successful.

in 3D, and so it is not suitable for numerical comparisons. Nevertheless, for qualitative comparison purposes, we ran the proposed framework on this dataset, and our reconstructions align very well to the image information. Example frames are provided in Fig. 4.12, and the entire video is provided as supplementary material. The best known published results on this dataset are found in [Salzmann and Urtasun, 2012], which uses an algorithm that requires training data in addition to explicitly delineating the edges of the surface. Our proposed framework is seen to perform as well as this previous method, qualitatively, while requiring no learning.

In order to be able to perform more meaningful numerical comparisons, we constructed a new dataset along with ground truth in 3D using a Kinect sensor, example images are provided in Fig. 4.15. This new sparsely textured paper dataset contains various deformations and large lighting changes along with occlusions.

Quantitative results using the new dataset are presented in Fig. 4.13 and Fig. 4.14. Feature-based methods that fail to reconstruct plausible surface shapes are indicated by high error bars that exceed y-axis range. Fig. 4.15 provides a representative reconstruction on a single frame. NCC and MI can track the surface fairly accurately, however they fail to capture fine details at the surface boundaries and hence the recovered depths in 3D are not very precise. Without occlusion handling, dense matching with gradient-based descriptors often fails near occlusions. The M-estimators are inconsistent near occlusions.

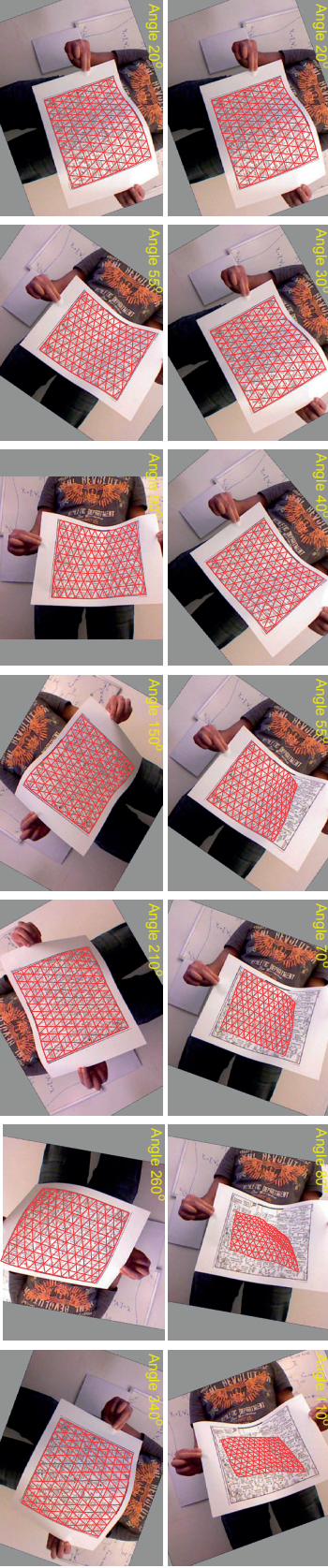


Figure 4.11: Tracking a rotating deformable surface. **Top row:** Without rotation handling, tracking with the original gradient-based descriptors eventually breaks down. **Bottom row:** The proposed rotation handling technique can track the whole sequence with up to 360 degrees of rotation.

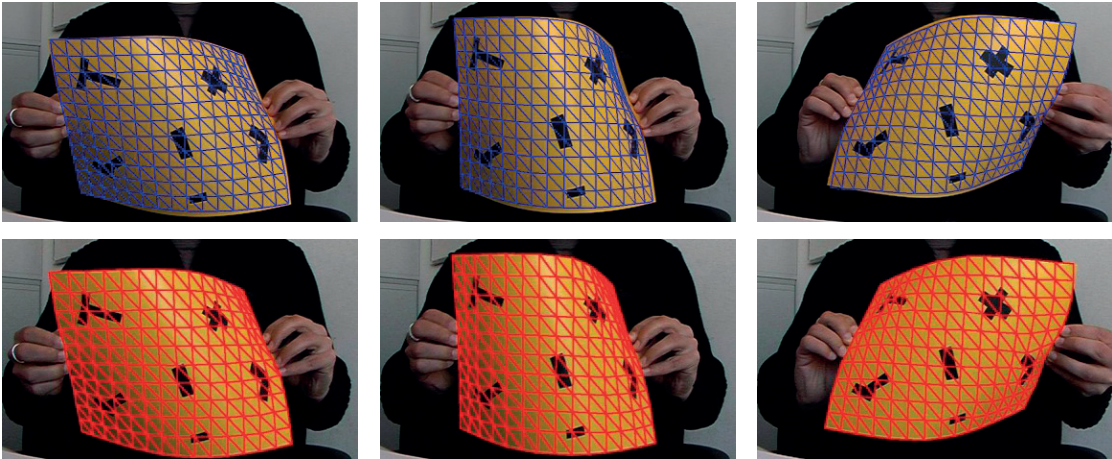


Figure 4.12: Sample reconstructions from the [Salzmann et al., 2008b] dataset. While no ground truth is available in 3D, our results (top row) are qualitatively observed to be very accurate; the best published results on this dataset are [Salzmann et al., 2008b] (bottom row), which has to extract the image edges explicitly, and involves learning, while our method does not. We do not have access to a reference image where the surface is in its planar rest shape, as our mesh assumes, causing some misalignment at the surface boundary.

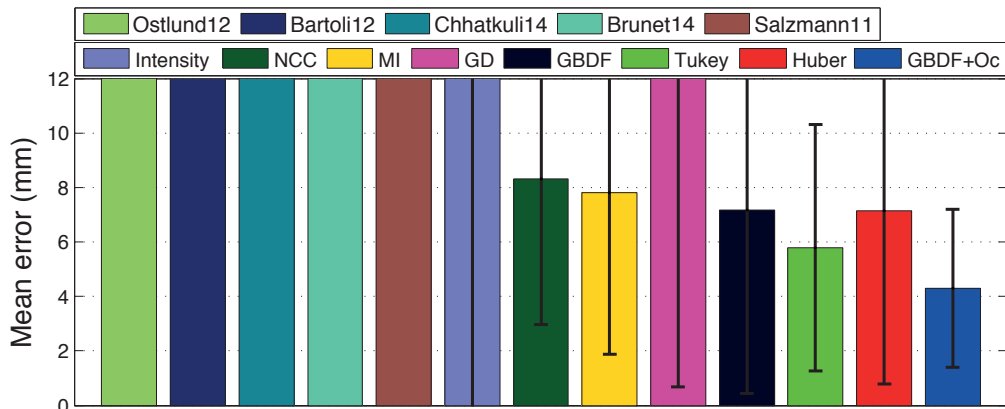


Figure 4.13: Reconstruction results computed using the vertex-to-vertex error metric on the sparsely textured paper dataset. Feature-based methods fail to reconstruct plausible surfaces, as indicated by the out-of-range error bars on the left.

However, the proposed framework is seen to be able to accurately track the surface throughout the entire sequence.

It is interesting to note that while the occlusions are cleanly delineated in the relevancy score over a textured surface, they are much less obviously visible in the relevancy score over a sparsely textured surface. This is expected, because well-textured un-occluded regions have consistently high correlation values with the template, and so it is only the occluded regions that are assigned low relevancy scores. However, image regions of little

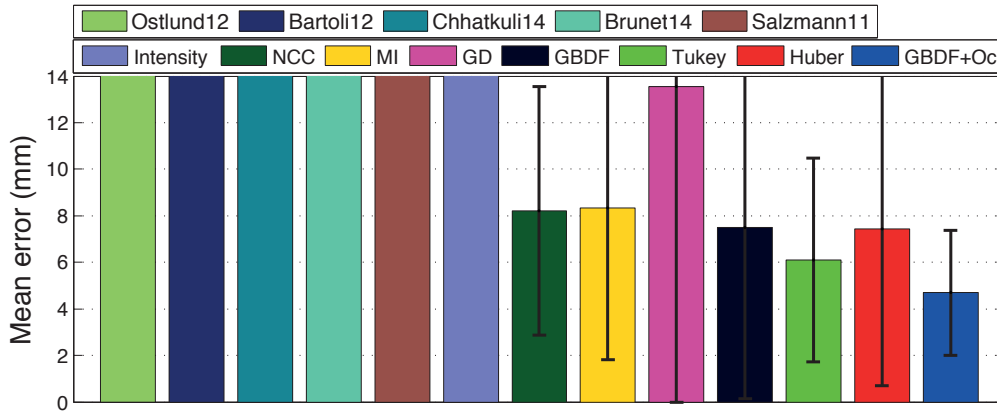


Figure 4.14: Reconstruction results computed using the vertex-to-cloud error metric on the sparsely textured paper dataset. Feature-based methods fail to reconstruct plausible surfaces, as indicated by the out-of-range error bars on the left.

texture have low and noisy correlation with a template, so occluded regions of similarly low correlation are assigned similarly low relevancy scores, and an occluded region is not as obviously distinct from the low textured regions in the relevancy score map. This is one of the strengths of the proposed framework, because only the truly meaningful image regions are allowed to strongly influence the image energy cost.

4.3.4 Additional Results

We demonstrate the robustness of our method in a variety of real-world applications.

First, we provide results on a cloth surface undergoing a different type of deformation than the one studied in the paper datasets. We created a new dataset along with ground truth in 3D using a Kinect sensor, as before, to which artificial occlusions were added. The t-shirt surface is represented by a 15×15 triangle mesh. Example images and our reconstructions are shown in Fig. 4.18. In the same figure, we also show a tracking failure case when occlusions appear at surface areas with large deformations and significant illumination changes. Quantitative results averaged over the whole dataset are presented in Fig. 4.16 and Fig. 4.17.

The strength of our approach is also demonstrated on a sparsely-textured sail surface with a few dot markers, shown in Fig. 4.19. Thanks to the large basin of convergence of our algorithm, we can simply initialize the registration from a very rough initial estimate without having first to establish correspondences. Our algorithm naturally exploits line features, which feature point-based methods usually do not.

Fig. 4.20 depicts another application of our method for animation capture from a monocular camera stream. In this setting, we capture the animations of a bird whose

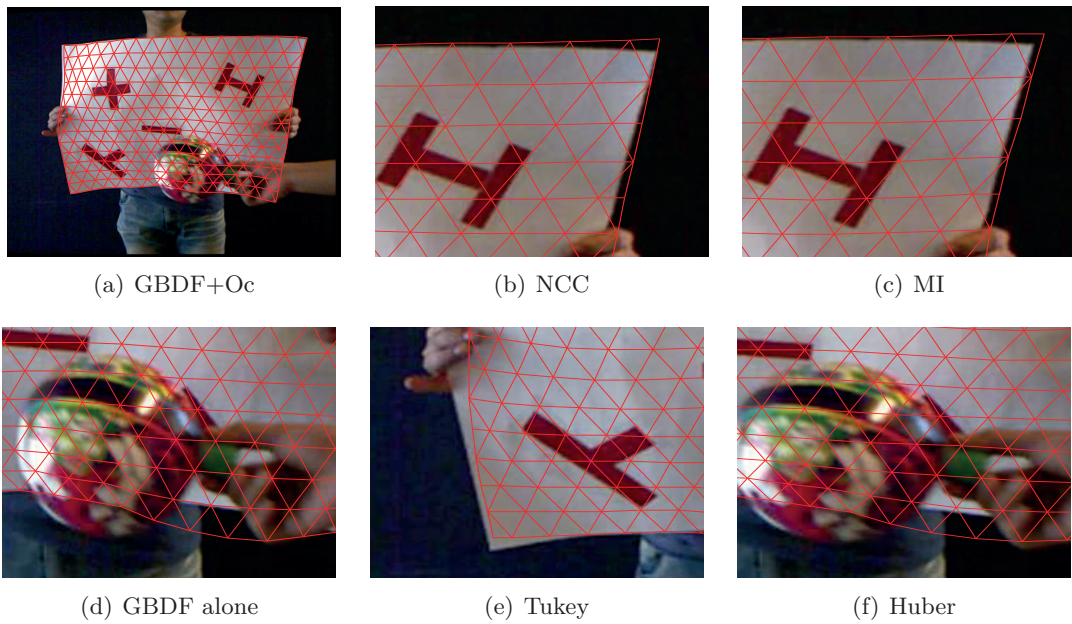


Figure 4.15: Reconstruction results on the same single frame. The proposed framework can track the whole sequence accurately, while other methods are seen to have trouble handling occlusions on top of the sparsely textured surface.

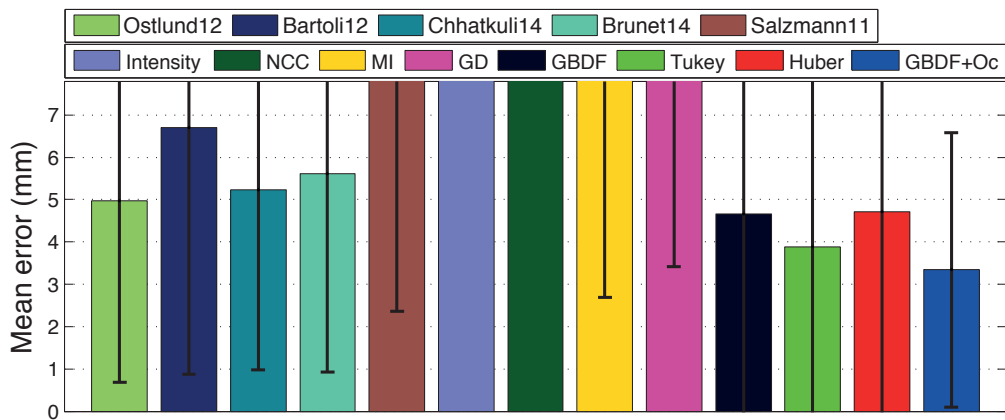


Figure 4.16: Reconstruction results computed using the vertex-to-vertex error metric on the t-shirt dataset.

animations can be transferred to another character. The video of captured animations is provided in the supplementary material.

Our tracking framework can be further extended to handle self-occlusions by incorporating a z-buffer. Based on the reconstruction from the previous frame or a motion model, self-occluded template pixels can be detected and excluded from the image energy computation. We used a z-buffer to densely register images of the self-occlusion dataset in [Gay-Bellile et al., 2010] and reconstruct the surface in the presence of large self-occlusions. The

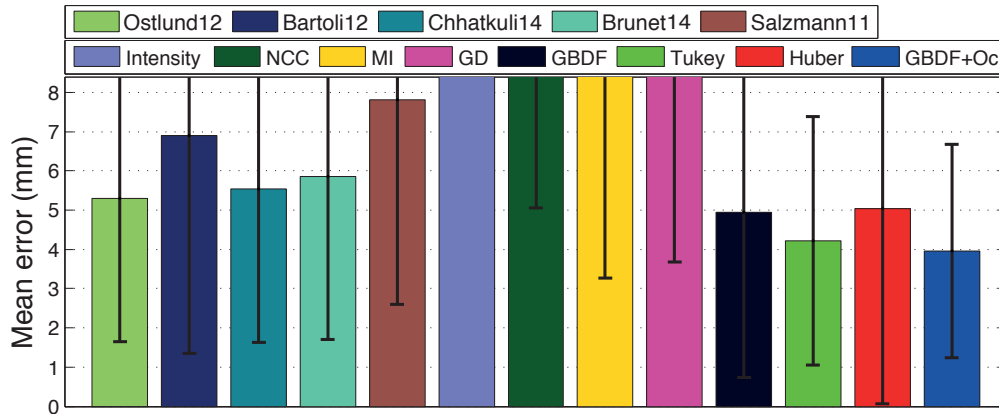


Figure 4.17: Reconstruction results computed using the vertex-to-cloud error metric on the the t-shirt dataset.



Figure 4.18: Our representative reconstructions on the t-shirt dataset with artificial occlusions added. Bottom right: a tracking failure case when occlusions appear at areas with large deformations.

qualitative results are provided in Fig. 4.21. With no additional modifications, our method can reliably predict the shape of the surface in the areas of self-occlusion. Our future work would be combine z-buffer and our relevancy scores to simultaneously handle self-occlusions and external occlusions.

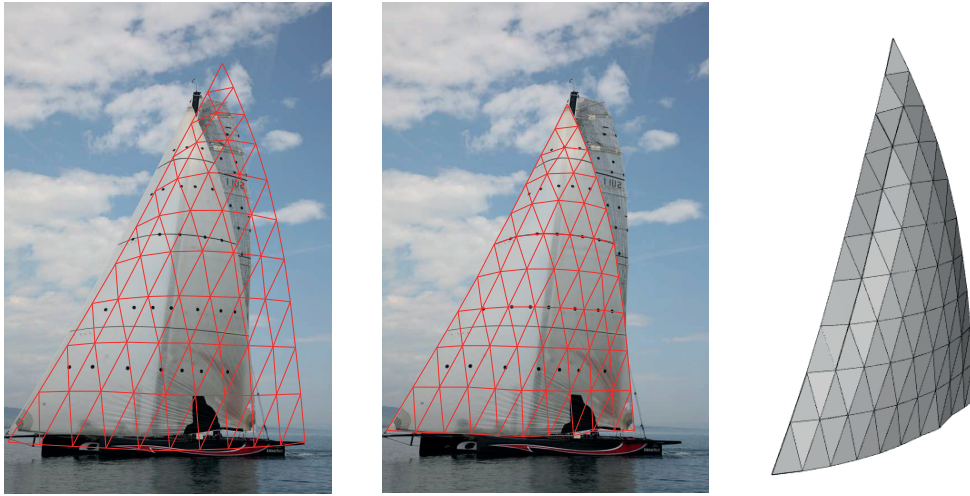


Figure 4.19: Image registration and surface reconstruction on the sparsely textured sail surface. From left to right: input image and initialization; final registration and reconstruction; the sail shape seen from a different viewpoint.

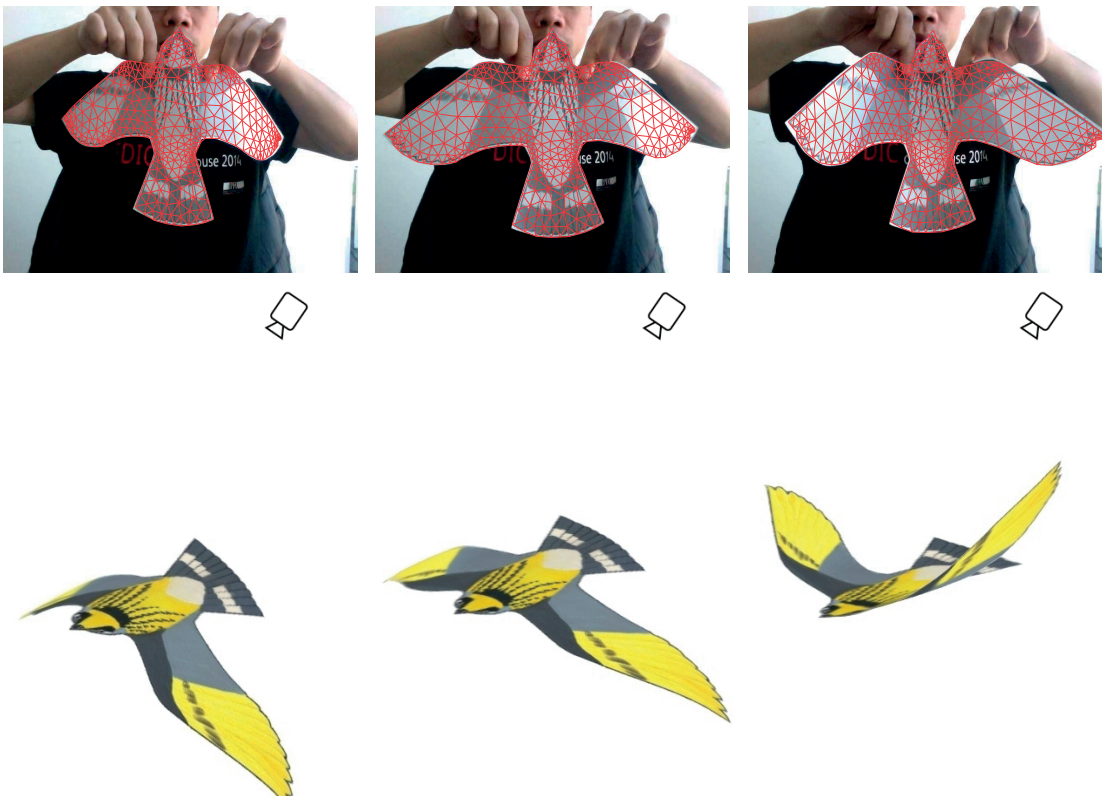


Figure 4.20: Surface reconstruction of an animation capture from a monocular camera stream.

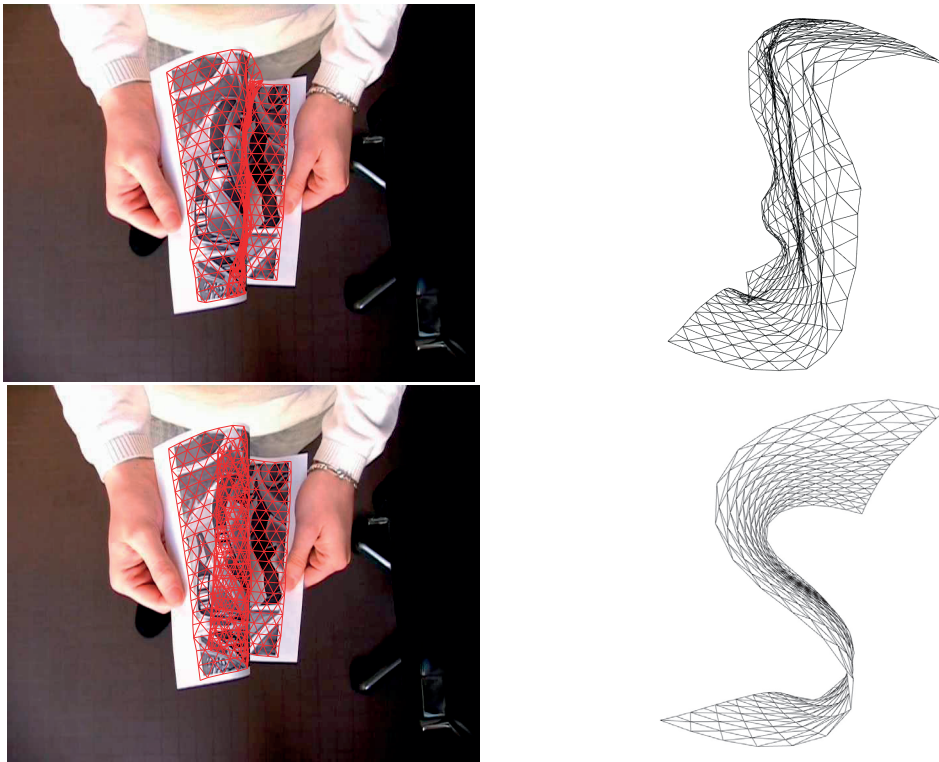


Figure 4.21: **Top row:** without handling self-occlusions, dense image registration and shape reconstruction give an erroneous result. **Bottom row:** A z-buffer is incorporated into our method to handle self-occlusions. Image registration is still successful in presence of 50% self-occlusions. **Left column:** Projections of the recovered shapes on image plane. **Right column:** the recovered 3-D surface shapes seen from a different viewpoint.

4.4 Conclusion

In this chapter, we have presented a framework for tracking both well textured and sparsely textured deforming surfaces in videos in the presence of occlusions. Our framework computes a relevancy score for each pixel, which is then used to weight the influence of the image information from that pixel in the image energy cost function. The presented method favorably compares to standard cost functions used for handling occlusion, such as Mutual Information and M-estimators.

5 Applications

In this chapter, we present a number of practical applications that rely on our deformable surface tracking and reconstruction algorithms. We first discuss the problem of reconstructing the 3-D shape of deformable objects as they interact with rigid objects. An application for studying ball-bat collisions is discussed to demonstrate that our video-based reconstruction could be potentially used to tune the parameters of mechanical simulation models so that their simulation results fit to what is observed in the input videos. Next, we introduce our real-time object recognition algorithm which automatically selects the template for our template-based tracking and reconstruction method. We then discuss some augmented reality applications that exploit our tracking and reconstruction technology. We can re-texture the object appearance or put virtual 3-D objects into the scene so that they are integrated seamlessly into the background. We introduce our lighting estimation algorithm that can retrieve environment lighting and realistically illuminate the virtual contents. Finally, we talk about our coloring book mobile application that can lift the colors from the drawing to paint 3-D virtual animated characters.

5.1 Deformable Object Reconstruction in Interactions

Our algorithms as described in the previous chapters can reliably handle deforming 3-D objects but do not take their environment into account. This is a severe limitation because, in the real world, rigid and deformable objects do not exist in isolation. Instead, they interact with each other and properly modeling these interactions is key to accurate reconstruction and understanding of the physical phenomena at play. Examples include balls being hit by rackets, bats, and clubs at ballgames and organs being prodded by surgical tools during operations, as depicted by Fig. 5.1. We will address this topic in this section. This will result in 3-D surface reconstruction algorithms that are more robust, more accurate, and can truly be deployed in real-world applications involving objects that interact with each other.

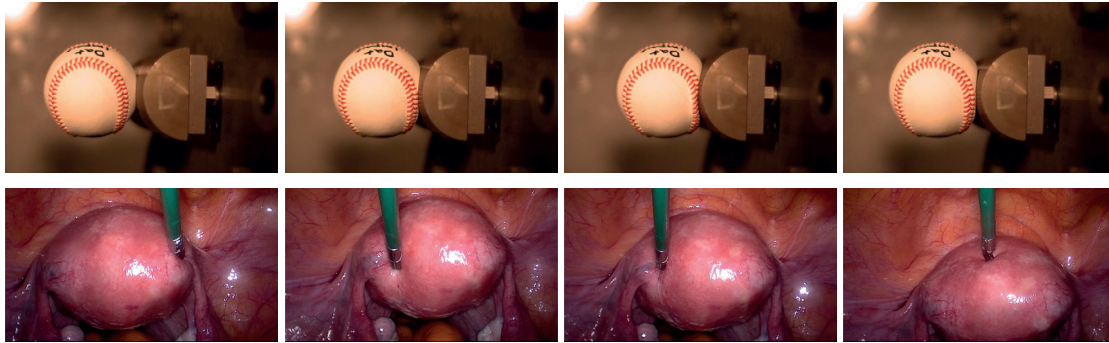


Figure 5.1: Surface deformations caused by another object. Top row: Baseball hitting a bat, courtesy of L. Smith, Washington State University. Bottom row: Surgical tool prodding the outside of a uterus during endoscopic surgery, courtesy of A. Bartoli, University of Clermont Ferrand.

5.1.1 Tracking Rigid Object Pose

We first present algorithms to compute the pose of the rigid object, which is a rotation and a translation that transform the rigid object in its world coordinates so that its projection matches to what is observed in an input image. We discuss an edge-based and a dense matching based method. Our edge-based method is an improvement upon [Harris and Stennett, 1990] extended to deal with erroneous correspondences and to work with more than one view. Our dense matching based algorithm modifies [Crivellaro and Lepetit, 2014] to track small objects and to deal with the rotation variant characteristic of the gradient-based descriptor fields that we use for dense image registration.

5.1.1.1 Edge-based Tracking

We describe in this section an edge-based algorithm that exploits edge cues on an input image to compute the 3-D pose of a known rigid object with respect to the camera, given that a 3-D CAD model of the object is available. Fig. 5.2 shows two examples in which there is a camera imaging a rigid object, for instance, a machinery part being inspected on a conveyor belt or a half-cylindrical object. The algorithm rotates and translates the CAD model in 3D, as shown in Fig. 5.2(a,b), so that its perspective camera projection matches the input image in Fig. 5.2(c,d).

Our algorithm is an improvement upon [Harris and Stennett, 1990] extended to deal with erroneous correspondences and to extract image information from more than one view. The overall idea of the algorithm is to use a set of pre-selected control points on high contrast edges or occluding contours of the 3-D CAD model, project them on the image plane using the current pose estimate, search for their corresponding edge points on the input image, formulate a set of equations that solves for a small rotation and a small translation adjustments that decrease the distance between the projected

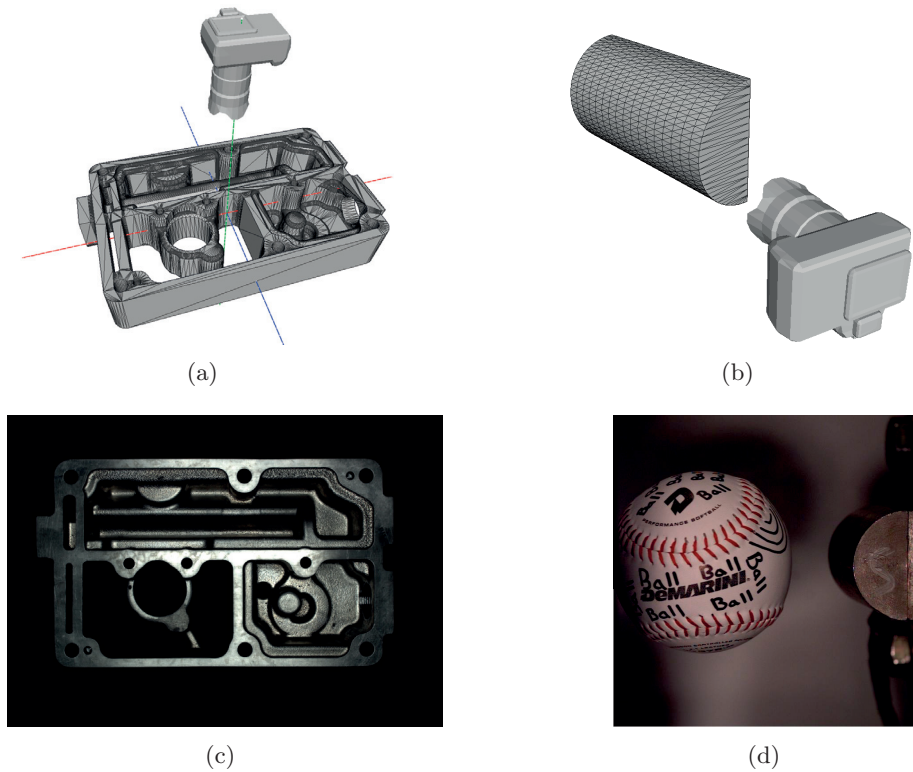


Figure 5.2: The algorithm rotates and translates the CAD model in 3D so that its perspective camera projection matches to what is observed in the input image. (a,b) Camera setup. (c,d) Input image captured by the camera.

control points and their corresponding edges on the input image. This procedure is performed iteratively until convergence. The method is robustified to deal with erroneous correspondences by using a robust estimator.

The 3-D CAD model is first pre-processed to select a set of high contrast edges and a set of control points lying on them. Occluding contours, on the other hand, will be computed in every iteration of an online process. The control points are selected so that they regularly sample the selected edge profile. This regular sampling guarantees that long edges play a more important role than short edges. It helps the optimization to avoid bias. An example of selected edges and control points is depicted in Fig. 5.3, in which only the front part of the CAD model is considered. If we allow an arbitrary pose of the object, we have to select edges and control points on all sides of the CAD model. In this case, hidden edges and hidden control points removals are required during iterations of the pose estimation.

For an input image, a rough initial pose estimate of the object is given. It could be estimated by other means such as generalized Hough transform [Ballard, 1981], template matching [Hinterstoisser et al., 2012], or machine learning based techniques [Crivellaro

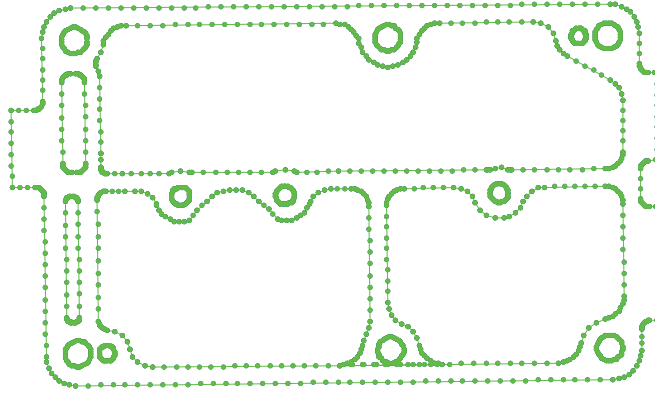


Figure 5.3: Select edge profile and control points. Line segments represent the edge profile, dots represent control points.

et al., 2015]. An iterative algorithm is then executed to gradually adjust the pose of the CAD model. It starts with projecting control points on the image using the current pose estimate. For each control point, in the directions of its normal we search for its one or more potential corresponding image edge points. From these correspondences, we formulate a set of equations that is solved for a small rotation and a small translation corrections that decrease the distance between the projected control points and their corresponding edges. To deal with erroneous correspondences, we use a robust estimator which becomes increasingly selective during iterations. Given the small rotation and the small translation, which are just found, polar matrix decomposition is then applied to update the current pose estimate of the object. This process repeats a number of iterations until convergence. We describe these steps in detail below.

Camera projection and correspondences Given an initial estimate of the rotation matrix \mathbf{R} and the translation vector \mathbf{t} , which transforms the 3-D CAD model from the world coordinates to the camera coordinates, the CAD model is perspectively projected on the image plane. Its projection already aligns roughly with the input image as demonstrated in Fig. 5.4. The transformation of a control point \mathbf{P}_{world} from the world coordinates to the camera coordinates is given by:

$$\mathbf{P}_{cam} = \mathbf{R} \cdot \mathbf{P}_{world} + \mathbf{t}. \quad (5.1)$$

Perspective projection on the image plane of the point \mathbf{P} in the camera coordinates is given by:

$$\begin{aligned} u &= f_u \cdot \frac{P_x}{P_z} + p_u \\ v &= f_v \cdot \frac{P_y}{P_z} + p_v \end{aligned} \quad (5.2)$$

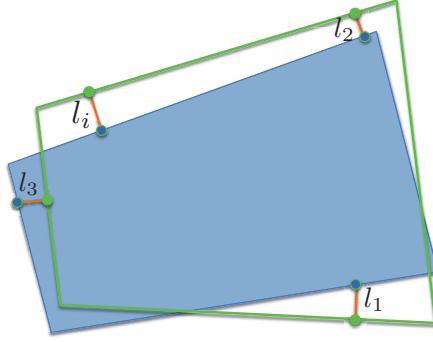


Figure 5.4: Using the current pose estimate \mathbf{R} and \mathbf{t} , the projection of the CAD model roughly aligns with the input image. The green lines and green dots represent the projection of the selected edge profile and the control points, respectively. The blue shape represents the object in the input image. The blue dots represent the corresponding edge points of the control points in the normal direction of the control edges.

The projection of these control points \mathbf{P} on image plane and their corresponding edge points in the input image in the normal direction of the control edges are demonstrated in Fig. 5.4. These corresponding edge points can be quickly searched for by intersecting a Bresenham line segment and the binary edge image (using Canny edge detector or a more sophisticated technique such as texture boundary [Shahrokhni et al., 2004]). The search is limited within a certain radius r . Each control point will result in a perpendicular distance l_i . We will use the set of these distances to find a small change in the object pose that minimizes these perpendicular distances after a pose correction.

Small pose adjustment From the current pose estimate, we consider rotating the CAD model about the world coordinate origin by a small rotation $\Delta\mathbf{R}$, and translating it by a small translation $\Delta\mathbf{t}$. A small rotation $\Delta\mathbf{R}$ can be linearized with respect to small rotation angles $\theta_x, \theta_y, \theta_z$ in radians about O_x, O_y, O_z .

$$\Delta\mathbf{R} = \begin{bmatrix} 1 & -\theta_z & \theta_y \\ \theta_z & 1 & -\theta_x \\ -\theta_y & \theta_x & 1 \end{bmatrix} \quad (5.3)$$

$$\Delta\mathbf{t} = \begin{bmatrix} \Delta t_x \\ \Delta t_y \\ \Delta t_z \end{bmatrix} \quad (5.4)$$

We write these small parameters as a six-vector $\mathbf{p} = [\Delta t_x, \Delta t_y, \Delta t_z, \theta_x, \theta_y, \theta_z]^T$, which is to be found to better align the projected model and the input image. After applying these small rotation and translation, the control point \mathbf{P} in the world coordinates will

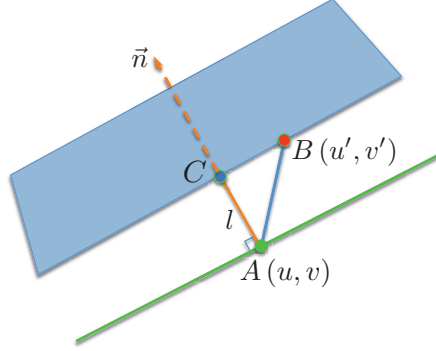


Figure 5.5: After applying the pose correction $\Delta \mathbf{R}$ and $\Delta \mathbf{t}$, we expect the projection of the control point to locate in its corresponding image edge. The blue shape represent the object seen in the input image. The green line and green dot represent the projection of the control edge and the control point using the current pose estimate. The distance between the projected control point and its corresponding image edge point is l . The red dot represents the expected location of the projection of the control point after the pose correction.

move to the new location \mathbf{Q} in the camera coordinates:

$$\mathbf{Q} = \Delta \mathbf{R} \cdot \mathbf{R} \cdot \mathbf{P} + \mathbf{t} + \Delta \mathbf{t}. \quad (5.5)$$

We assume that the orientations of the control edge and the image edge are nearly the same. Given a rough initialization, this assumption is reasonably and gradually satisfied while the algorithm is iterating. After applying the pose correction $\Delta \mathbf{R}$ and $\Delta \mathbf{t}$, we expect the projection of \mathbf{Q} to locate in its corresponding image edge (see Fig. 5.5). We have an equation:

$$\langle \vec{n} \cdot \overrightarrow{AB} \rangle = l \quad (5.6)$$

Denote $\mathbf{S} = \mathbf{R} \cdot \mathbf{P}$, and expand Eq. 5.5, we get:

$$\mathbf{Q} = \begin{bmatrix} S_x + t_x + \Delta t_x + \theta_y S_z - \theta_z S_y \\ S_y + t_y + \Delta t_y + \theta_z S_x - \theta_x S_z \\ S_z + t_z + \Delta t_z + \theta_x S_y - \theta_y S_x \end{bmatrix} \quad (5.7)$$

\mathbf{Q} will project on the image plane at:

$$\begin{aligned} u' &= f_u \cdot \frac{S_x + t_x + \Delta t_x + \theta_y S_z - \theta_z S_y}{S_z + t_z + \Delta t_z + \theta_x S_y - \theta_y S_x} + p_u \\ v' &= f_v \cdot \frac{S_y + t_y + \Delta t_y + \theta_z S_x - \theta_x S_z}{S_z + t_z + \Delta t_z + \theta_x S_y - \theta_y S_x} + p_v \end{aligned} \quad (5.8)$$

5.1. Deformable Object Reconstruction in Interactions

We use the first order Taylor series of u' and v' with respect to $\mathbf{p} = [\Delta t_x, \Delta t_y, \Delta t_z, \theta_x, \theta_y, \theta_z]^T$. We get:

$$\begin{aligned} u' &= u + \frac{f_u \Delta t_x - (u - p_u) \Delta t_z - (u - p_u) S_y \theta_x + ((u - p_u) S_x + S_z f_u) \theta_y - S_y f_u \theta_z}{S_z + t_z} \\ v' &= v + \frac{f_v \Delta t_y - (v - p_v) \Delta t_z - ((v - p_v) S_y + S_z f_v) \theta_x + (v - p_v) S_x \theta_y + S_x f_v \theta_z}{S_z + t_z} \end{aligned} \quad (5.9)$$

Replace u', v' into Eq. 5.6, we obtain a linear equation with respect to \mathbf{p} :

$$\mathbf{n}^T \cdot \mathbf{M} \cdot \mathbf{p} = l, \quad (5.10)$$

in which \mathbf{M} is a 2×6 matrix:

$$\begin{bmatrix} f_u & 0 & -(u - p_u) & -(u - p_u) S_y & (u - p_u) S_x + S_z f_u & -S_y f_u \\ 0 & f_v & -(v - p_v) & -(v - p_v) S_y - S_z f_v & (v - p_v) S_x & S_x f_v \end{bmatrix} \quad (5.11)$$

For each control point, we have one linear equation. We will get an over-determined linear system if we consider a set of $N > 6$ control points. This linear system can be solved in the least squares sense.

$$\mathbf{H} \cdot \mathbf{p} = \mathbf{l} \quad \Leftrightarrow \quad \mathbf{p} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{l} \quad \Leftrightarrow \quad \mathbf{p} = \mathbf{H} \backslash \mathbf{l}. \quad (5.12)$$

Pose update Once we have found $\mathbf{p} = [\Delta t_x, \Delta t_y, \Delta t_z, \theta_x, \theta_y, \theta_z]^T$ from Eq. 5.5, we can easily update the translation vector $\mathbf{t} \leftarrow \mathbf{t} + \Delta \mathbf{t}$. Conceptually, we can also update the rotation matrix as $\mathbf{R} \leftarrow \Delta \mathbf{R} \cdot \mathbf{R}$. However, due to the linearization, the small rotation matrix $\Delta \mathbf{R}$ is not exactly orthonormal. This update will gradually break the orthonormality of the rotation matrix \mathbf{R} . To prevent this, we apply the polar decomposition on the matrix $\Delta \mathbf{R}$ to find a unitary matrix \mathbf{U} that is closest to $\Delta \mathbf{R}$ in the Frobenius norm. The matrix \mathbf{U} is found by applying the singular value decomposition on the matrix $\Delta \mathbf{R}$: $[\mathbf{D}, \mathbf{\Sigma}, \mathbf{E}] = \text{svd}(\Delta \mathbf{R})$. We get $\mathbf{U} = \mathbf{D} \cdot \mathbf{E}^T$

The rotation matrix is finally updated by $\mathbf{R} \leftarrow \mathbf{U} \cdot \mathbf{R}$. It is guaranteed to be an orthonormal matrix, i.e. a valid rotation matrix.

Robust estimation The set of correspondences found above in Fig. 5.4 may contain errors due to misleading image edges. We need to suppress these outlier correspondences in the pose computation. M-estimators are a popular method for handling outliers. Let e_i be the residual at a control point \mathbf{P}_i , then instead of minimizing the sum of squared residuals $\sum_i e_i^2$, a modified loss function ρ of the residuals is considered, instead minimizing $\sum_i \rho(e_i)$, in order to reduce the influence of outliers. We use the Tukey

bi-square loss function defined by:

$$\rho^{\text{Tu}}(e) = \begin{cases} \frac{r^2}{6} (1 - (1 - (\frac{e}{r})^2)^3) & \text{for } |e| \leq r, \\ r^2/6 & \text{otherwise.} \end{cases} \quad (5.13)$$

The Tukey loss function suppresses the influence of points with large error residuals by re-assigning their residuals to a constant weight. Instead of solving the Eq. 5.12, we solve a weighted least squares:

$$\mathbf{W} \cdot \mathbf{H} \cdot \mathbf{p} = \mathbf{W} \cdot \mathbf{l} \Leftrightarrow \mathbf{p} = (\mathbf{H}^T \mathbf{W}^2 \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W}^2 \mathbf{l} \Leftrightarrow \mathbf{p} = (\mathbf{H}^T \mathbf{W}^2 \mathbf{H}) \setminus \mathbf{H}^T \mathbf{W}^2 \mathbf{l} \quad (5.14)$$

in which \mathbf{W} is a diagonal Tukey weighting matrix of input \mathbf{l} (i.e. distance measurements).

We repeat these steps above until the algorithm converges. Initially, we set a large radius to look for edge point correspondences and a large support radius for the Tukey estimator. This radius is gradually decreased to eliminate outliers and to help the algorithm converge to a correct solution.

Multi-views edge-based pose estimation We extend the algorithm above to utilize image information from more than one view. Assume that we have a calibrated system of two cameras (generalization to three or more cameras is straightforward). Our aim is to extend our above-described algorithm to two views. We select one view as the reference coordinate system. We try to find a small rotation and a small translation in this coordinate system that minimize the image discrepancy in this view as well as the other view.

Transformation of a 3-D point in the first camera coordinate system to the second camera coordinate system is given by

$$\begin{aligned} \mathbf{P}_{\text{cam}_2} &= \mathbf{R}_{\text{stereo}} \cdot \mathbf{P}_{\text{cam}_1} + \mathbf{t}_{\text{stereo}} \\ &= \mathbf{R}_{\text{stereo}} \cdot (\mathbf{R} \cdot \mathbf{P} + \mathbf{t}) + \mathbf{t}_{\text{stereo}} \\ &= \mathbf{R}_{\text{stereo}} \cdot \mathbf{R} \cdot \mathbf{P} + \mathbf{R}_{\text{stereo}} \cdot \mathbf{t} + \mathbf{t}_{\text{stereo}} \quad . \end{aligned} \quad (5.15)$$

Applying this stereo transformation to the point \mathbf{P} after a small rotation and a small translation in Eq. 5.5, we get

$$\begin{aligned} \mathbf{Q}_{\text{cam}_2} &= \mathbf{R}_{\text{stereo}} \cdot (\Delta \mathbf{R}_{\text{cam}_1} \cdot \mathbf{R} \cdot \mathbf{P} + \mathbf{t} + \Delta \mathbf{t}_{\text{cam}_1}) + \mathbf{t}_{\text{stereo}} \\ &= \mathbf{R}_{\text{stereo}} \cdot \Delta \mathbf{R}_{\text{cam}_1} \cdot \mathbf{R} \cdot \mathbf{P} + \mathbf{R}_{\text{stereo}} \cdot \mathbf{t} + \mathbf{R}_{\text{stereo}} \cdot \Delta \mathbf{t}_{\text{cam}_1} + \mathbf{t}_{\text{stereo}} \quad . \end{aligned} \quad (5.16)$$

Equivalent to applying a small rotation $\Delta \mathbf{R}$ and a small translation $\Delta \mathbf{t}$ to the object in the camera one coordinate system, we apply a small rotation $\Delta \mathbf{R}_{\text{cam}_2}$ and a small

5.1. Deformable Object Reconstruction in Interactions

translation $\Delta \mathbf{t}_{\text{cam}_2}$ to the object in the camera two coordinate system in Eq. 5.15. We obtain:

$$\mathbf{Q}_{\text{cam}_2} = \Delta \mathbf{R}_{\text{cam}_2} \cdot \mathbf{R}_{\text{stereo}} \cdot \mathbf{R} \cdot \mathbf{P} + \mathbf{R}_{\text{stereo}} \cdot \mathbf{t} + \mathbf{t}_{\text{stereo}} + \Delta \mathbf{t}_{\text{cam}_2} \quad . \quad (5.17)$$

Equate Eq. 5.16 and Eq. 5.17, we get the transformation of $\Delta \mathbf{R}$ and $\Delta \mathbf{t}$ from camera view one to camera view two as followed:

$$\begin{aligned} \Delta \mathbf{R}_{\text{cam}_2} &= \mathbf{R}_{\text{stereo}} \cdot \Delta \mathbf{R}_{\text{cam}_1} \cdot \mathbf{R}_{\text{stereo}}^T \\ \Delta \mathbf{t}_{\text{cam}_2} &= \mathbf{R}_{\text{stereo}} \cdot \Delta \mathbf{t}_{\text{cam}_1} \quad . \end{aligned} \quad (5.18)$$

Eq. 5.18 shows that $\Delta \mathbf{R}_{\text{cam}_2}$ is a valid small rotation matrix and that the parameters of $\Delta \mathbf{R}_{\text{cam}_2}$ and $\Delta \mathbf{t}_{\text{cam}_2}$ are linear with respect to $\mathbf{p}_{\text{cam}_1} = [\Delta t_x, \Delta t_y, \Delta t_z, \theta_x, \theta_y, \theta_z]^T$.

$$\mathbf{p}_{\text{cam}_2} = \mathbf{C} \cdot \mathbf{p}_{\text{cam}_1} \quad , \quad (5.19)$$

in which \mathbf{C} is a 6×6 matrix, composing of $\mathbf{R}_{\text{stereo}}$ coefficients, that linearly transforms the parameters in the camera one view to the camera two view.

Combining image information from the two views, we want to find small rotations $\Delta \mathbf{R}_{\text{cam}_1}$, $\Delta \mathbf{R}_{\text{cam}_2}$ and small translations $\Delta \mathbf{t}_{\text{cam}_1}$, $\Delta \mathbf{t}_{\text{cam}_2}$ in each view so that they decrease the distances between the projected control points and their corresponding edges on the input images subject to the constraint in Eq. 5.19. We solve the following optimization problem for the parameters \mathbf{p} in the camera one view:

$$\text{minimize}_{\mathbf{p}} \quad \|\mathbf{H}_{\text{cam}_1} \cdot \mathbf{p} - \mathbf{l}_{\text{cam}_1}\|^2 + \|\mathbf{H}_{\text{cam}_2} \cdot \mathbf{C} \cdot \mathbf{p} - \mathbf{l}_{\text{cam}_2}\|^2 \quad . \quad (5.20)$$

The robust estimator can still be used to reject outlier correspondences in each view and the pose update is performed in the camera one coordinate system.

Results We use the method described above to compute the pose of the machinery part in Fig. 5.2(c) from a single view and the result is depicted in Fig. 5.6. During iterations, the projected control edges gradually align better with the image edges.

The estimated pose of the half cylinder from a single view in Fig. 5.2(d) is shown in Fig. 5.7. In this setup, there is also another camera capturing the top view of the half cylinder. These two views are calibrated and the stereo transformation between them is known. The half cylinder in its initial and final pose estimates from the side view in Fig. 5.7(a,f) is projected onto the top view and the results are depicted in Fig. 5.8. It shows that the single view pose estimate using the side view alone is not accurate in estimating the depth. It can be explained by the fact that in this setup the side view

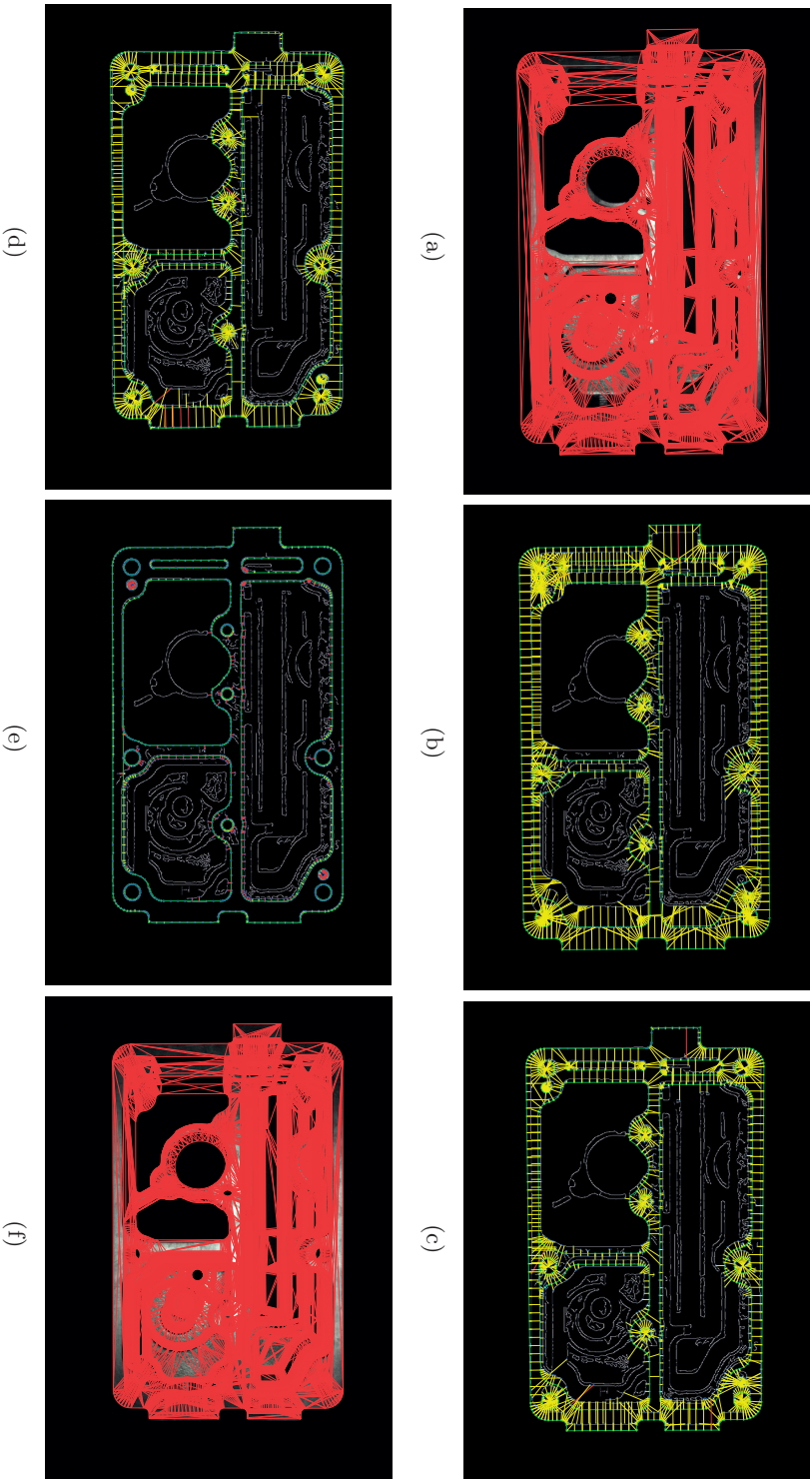


Figure 5.6: Pose estimation of a machinery part from a single view. (a) The CAD model is projected on the image plane using the initial pose estimate. (b,c,d,e) Iterations 1, 3, 5, 15. Green lines and green dots represent control edges and control points projected on the image plane. Blue points represent the corresponding image edge (in white) points of the projected control points. Yellow line segments represent distance l . Red line segments indicate outlier correspondences. (f) The CAD model is projected on the image plane using the final pose estimate.

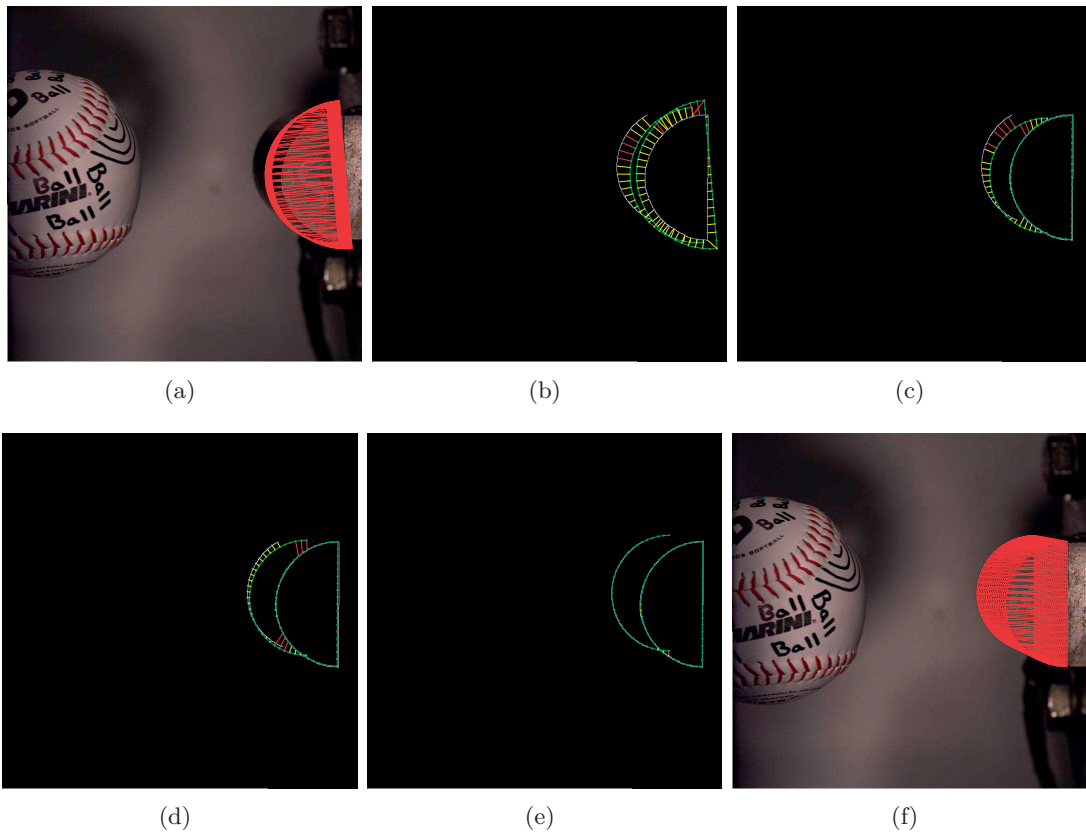


Figure 5.7: Pose estimation of a half cylinder from a single view. (a) Initial pose estimate. The cylinder 3-D model is projected on the image plane using the initial pose estimate. (b,c,d,e) Iterations 1, 3, 5, 15. Green lines and green dots represent control edges and control points projected on the image plane. Blue points represent the corresponding image edge points (in white) of the projected control points. Yellow line segments represent distance l . Red line segments indicate outlier correspondences. (c) Final pose estimate. The cylinder 3-D model is projected on the image plane using the final pose estimate.

camera focal length is so large that it makes the depth estimation imprecise. We can use both views to find the pose of the half cylinder that is consistent with both views. The results of this binocular pose estimation is shown in Fig. 5.9. In this scenario, besides high contrast edges, occluding contours can also be used.

In Fig. 5.10, to compute the location of the ball in an image when it is not deformed yet, we model the ball as a sphere with a known radius and use its occluding contours in both views. The results are depicted in Fig. 5.10 that shows good alignments of the ball and the two views.

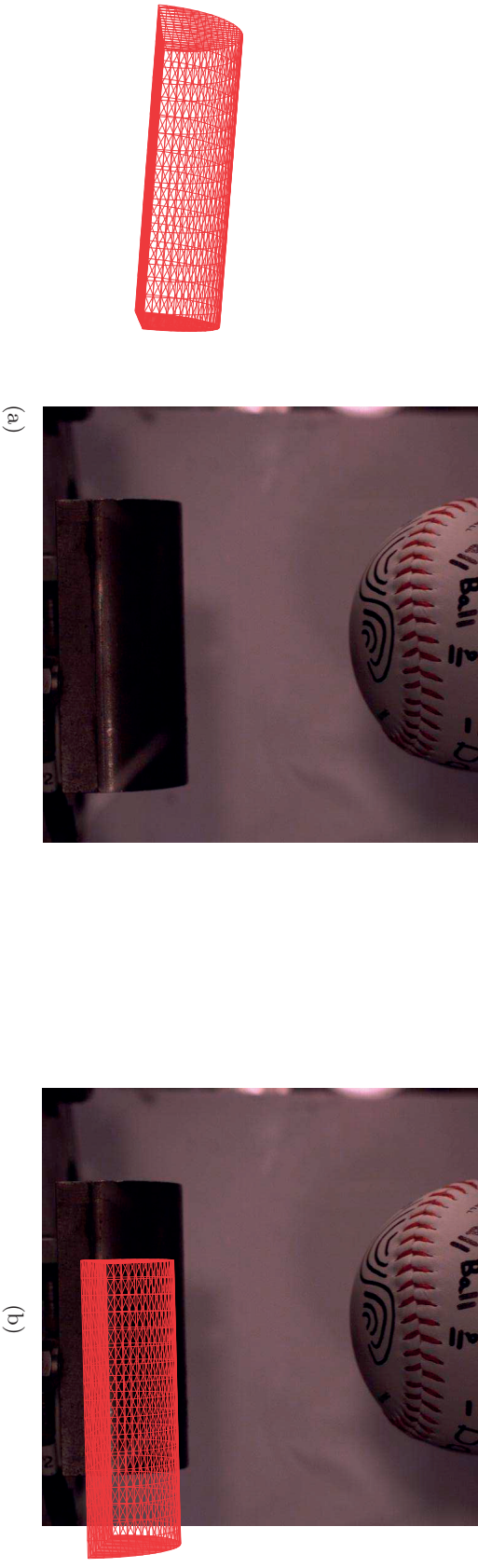


Figure 5.8: Projections of the half cylinder onto the top view in its initial and final pose estimates using the side view image in Fig. 5.7. (a) Initial pose. The half cylinder is projected outside the top view image. (b) Final pose. The half cylinder projection on the image plane does not align well with the input image.

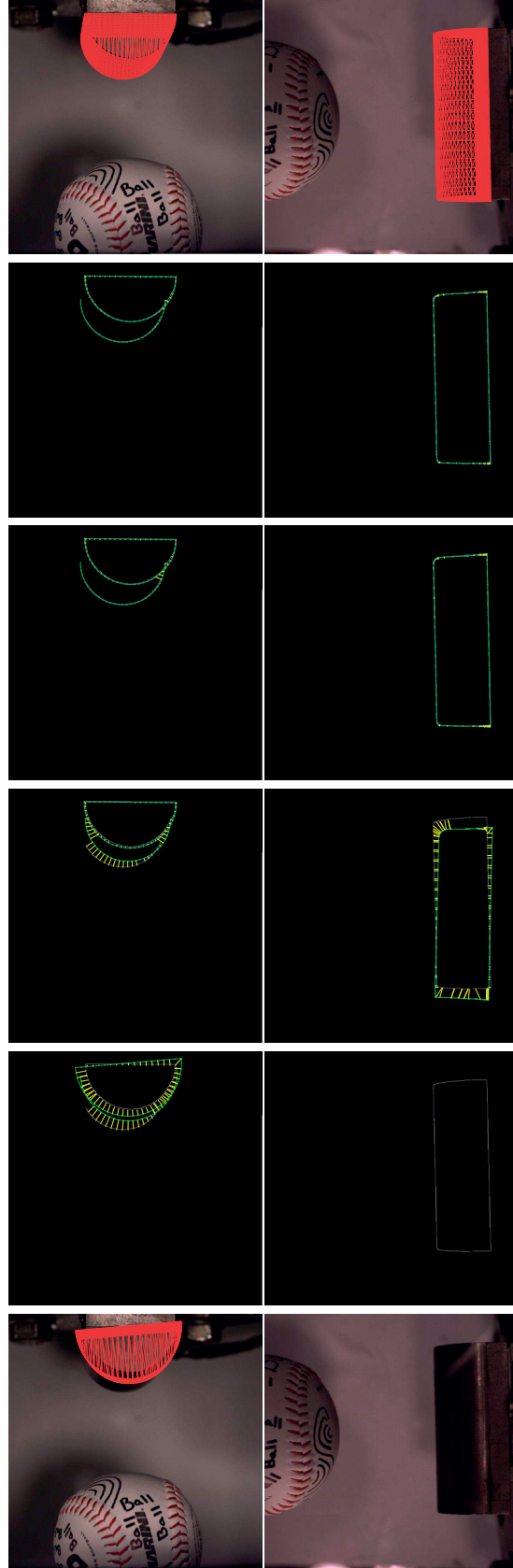


Figure 5.9: Pose estimation of the half cylinder from two views. Top row: side view. Bottom row: top view. Leftmost column: initial pose. In its initial pose, the half cylinder is projected outside the top view. Middle columns: iterations 1, 2, 5, 15. Green lines and green dots represent control edges and control points projected on the image plane. Blue points represent the corresponding image edge points (in white) of the projected control points. Yellow line segments represent distance l . Red line segments indicate outlier correspondences. Rightmost column: final pose estimate.

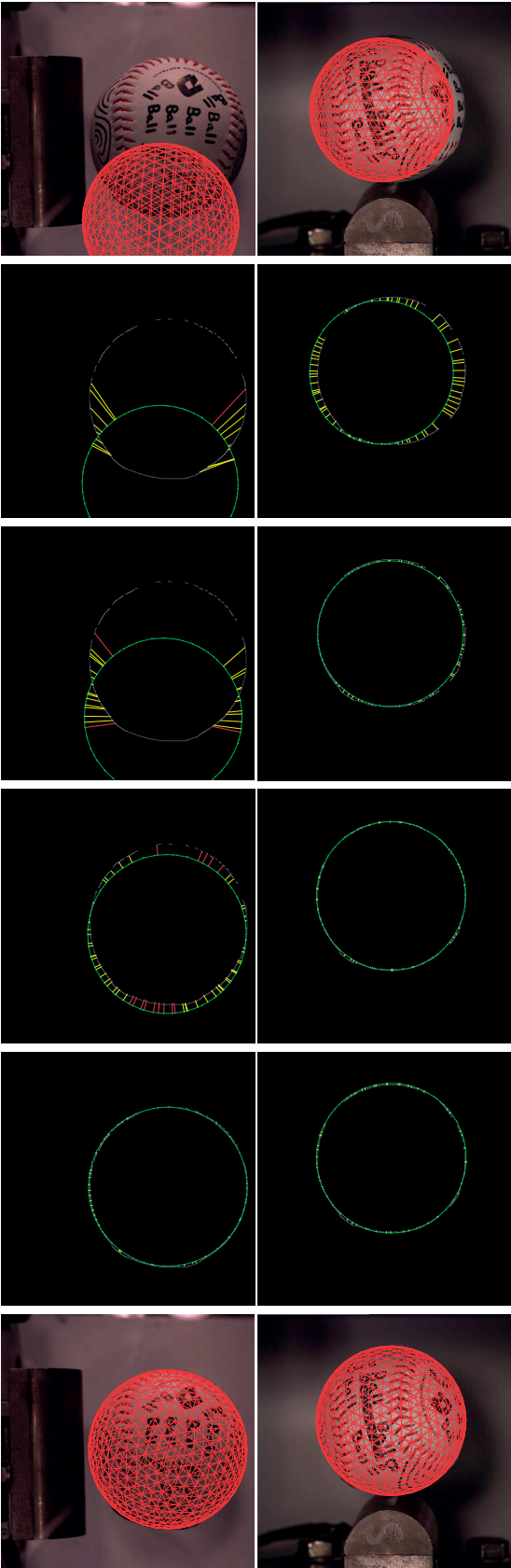


Figure 5.10: Pose estimation of the ball from two views. Top row: side view. Bottom row: top view. Leftmost column: initial pose. In its initial pose, the ball is projected outside the top view. Middle columns: iterations 1, 5, 9, 15. Green lines and green dots represent control edges and control points projected on the image plane. Blue points represent the corresponding image edge points (in white) of the projected control points. Yellow line segments represent distance l . Red line segments indicate outlier correspondences. Rightmost column: final pose estimate.

5.1.1.2 Template Matching based Tracking

Another class of 3-D rigid object pose estimation is to rely on object texture. Given a template image and the 3-D shape of a textured rigid object, we want to estimate the pose of the rigid object with respect to the camera in a new input image J . We can rely on feature points detected and matched between the template image and the input image. The problem is casted as a PnP problem, that is estimating the pose of a calibrated camera from a set of n 3D-to-2D feature point correspondences [Lepetit et al., 2009, Kneip et al., 2014]. This approach works well when the object is well-textured. Thanks to the properties of feature points, this approach is robust to large view changes, illumination changes, occlusions, and cluttered backgrounds.

However, when the object is not well-textured or appears small in the image, for example the pen in Fig. 5.11, this approach tends to fail. Dense image alignment approach is an attractive alternative because it globally exploits most of the image information, even when local image features such as interest points or edges are ambiguous.



Figure 5.11: Tracking the 3-D pose of a small object using direct image alignment is an attractive approach that globally exploits most of the image information.

The first template matching algorithm was introduced in [Lucas and Kanade, 1981] that computes the optical flow at a sparse set of image locations. It was then extended to general purpose image alignment. It has been used to register a 2-D template to an input image under a family of transformations such as affinity or homography. It has also been used to track 3-D objects. To track the pen in Fig. 5.11, we base on the recently proposed method [Crivellaro and Lepetit, 2014] that performs dense image alignment with robust gradient-based descriptor fields rather than illumination changes sensitive pixel intensities.

Let \mathbf{p}_T denote the object pose corresponding to the template image T . The image warping function $W(\mathbf{x}, \mathbf{p}_T, \mathbf{p})$ defines a transformation from a pixel location \mathbf{x} on the template to a new pixel location on the input image J . As shown in Fig. 5.12, this image warping function back-projects the template image location \mathbf{x} onto the 3-D model using the pose \mathbf{p}_T . The intersection defines the corresponding 3-D location on the object surface. The perspective projection of this 3-D point on input image plane defines the corresponding input image point. We solve for a six degrees of freedom camera pose \mathbf{p} that maximizes the similarity between every pixel \mathbf{x} and its corresponding pixel $W(\mathbf{x}, \mathbf{p}_T, \mathbf{p})$.

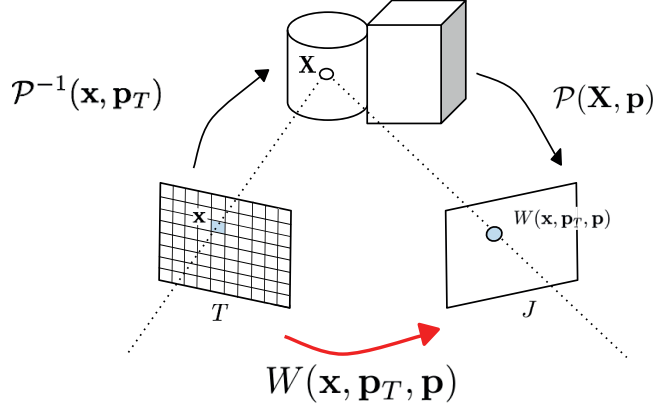


Figure 5.12: The image warping function that aligns the template image to the input image [Crivellaro and Lepetit, 2014]. It transforms a pixel location \mathbf{x} on the template image T to a new pixel location on the input image J .

Mathematically, we minimize the following image matching energy function:

$$E(\mathbf{p}) = \sum_{\mathbf{x}} \|d(J, W(\mathbf{x}, \mathbf{p}_T, \mathbf{p})) - d(T, \mathbf{x})\|^2 \quad (5.21)$$

in which $d(I, \mathbf{x})$ is a function that computes descriptors for pixel \mathbf{x} on image I . Normally, $d(I, \mathbf{x})$ is taken as image intensity at pixel \mathbf{x} . However, [Crivellaro and Lepetit, 2014] instead proposed to use gradient-based descriptor fields in place of the image intensity.

$$d(I, \mathbf{x}) = \left[\left[\frac{\partial I}{\partial x}(\mathbf{x}) \right]^+, \left[\frac{\partial I}{\partial x}(\mathbf{x}) \right]^-, \left[\frac{\partial I}{\partial y}(\mathbf{x}) \right]^+, \left[\frac{\partial I}{\partial y}(\mathbf{x}) \right]^- \right]^\top, \quad (5.22)$$

where the $[\cdot]^+$ and $[\cdot]^-$ operations respectively keep the positive and negative values of a real-valued signal. These descriptors are robust under lighting changes, and remain discriminative after the Gaussian smoothing which is usually employed in a multi-scale approach of image alignment.

One limitation of [Crivellaro and Lepetit, 2014] is that the descriptors are not invariant under rotations or when the object is seen from different viewpoints. As a consequence, the tracking performance will degrade if the object rotates or there are large viewing changes between the input image and the template image. To be able to successfully track the pen in a video shown in Fig. 5.11, we obtained a full textured model of the pen. When tracking at time t , we generate the template image by rendering the 3-D model to the view of the previous frame at time $t-1$. A motion model could also be used to predict ahead the pose in the current frame. This rendering makes the appearance between the template and the input image very similar and thus enhances the convergence of the image alignment. The tracking results of the small pen in Fig. 5.11 are shown in Fig. 5.13.



Figure 5.13: Results of tracking the 3-D pose of a small pen using direct image alignment and render-based template generation. Top row: Reprojection of the pen 3-D model on the input image (in red). Bottom row: Template generation for the next frame by rendering the 3-D model on the current view.

We will use these pose estimation results to impose the constraints in recovering the deformations of a cushion in interactions with the pen in the following section.

5.1.2 Ball-Bat Study

An accurate description of sports balls is needed to design protective equipment, such as helmets and pads, and striking equipment, such as bats, clubs and rackets. It would also help regulating agencies better understand equipment performance through experimental testing and numerical modeling. Given the geometric and material non-linearities inherent to ball impacts, finite element analysis is often used to simulate them. There is a need to compare impact simulations against reality to assess which predictions most accurately describe a physical quantity of interest. For example, when testing protective equipment or durability, peak impact force is key. By contrast, when testing equipment performance in the case of bat, club, or racket impacts, energy dissipation and coefficient of restitution from impact matter most.

To study the ability of numeric models to describe sport ball response, while impacting a rigid cylindrical surface, we set up an impact experiment that was designed to simulate a bat-ball impact and was recorded using high-speed video cameras. The resulting images and the ball geometry before impact were used as input to our computer vision algorithm, which then produced a quantitative description of the deformation during impact. This observed deformation will be compared to the deformations computed using simulation models. Foam-based material models were observed to match this observed deformation better than visco-elastic material models. The measured ball deformation, afforded through video analysis, has shown that foam material models are better able to describe ball impacts, involving large energy dissipation.

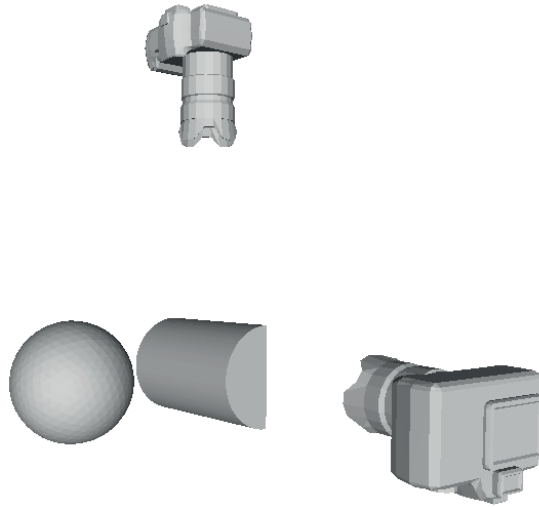


Figure 5.14: Experiment setup of the ball-bat study. The ball is launched by a ball canon against a half cylinder. Two high speed cameras are used to capture the impact.

5.1.2.1 Method

The set up of our experiments is shown in Fig. 5.14. The ball is projected at 153 km/h by a ball canon against a half cylinder, which is securely mounted on the wall and serves as the bat. Ball impacts were recorded using two high-speed video cameras, the first was aligned with the long axis of the impact cylinder and the second was mounted above it. The cameras recorded the impact at 13'000 frames per second with an exposure time of $15 \mu\text{s}$ and a resolution of 704×704 pixels. Prior to impact testing, a checkered board was rotated through several positions within the capture volume for calibrating the two-camera system. The 3-D shape of the deforming ball was estimated using image cues from both views. As illustrated by Fig. 5.15, we infer the deformed 3-D shape of the object using a template-based method. We take the template image to be the first one where the ball is undeformed and can be represented by a spherical triangulation of diameter 97.02 mm. The half cylinder has the diameter of 57.15 mm and the length of 101.6 mm.

We applied our pose estimation algorithm presented in Section 5.1.1.1 using image edge cues from both views. We computed the location and orientation of the half cylinder and the location of the ball corresponding to its template images. Our objective is to deform the ball's rest shape such that its perspective projections match to image observations in both the side view and the top view. The recovered shape must also obey physical constraints that the ball must not penetrate the half cylinder. To account for noise in image measurement, we regularize the solution using the Laplacian-based regularization term for a non-planar object presented in Section 3.3.1.2. We allow the ball surface to slightly stretch or shrink around its reference configuration. We also enforce the symmetry constraints of the ball with respect to the vertical plane that goes through the

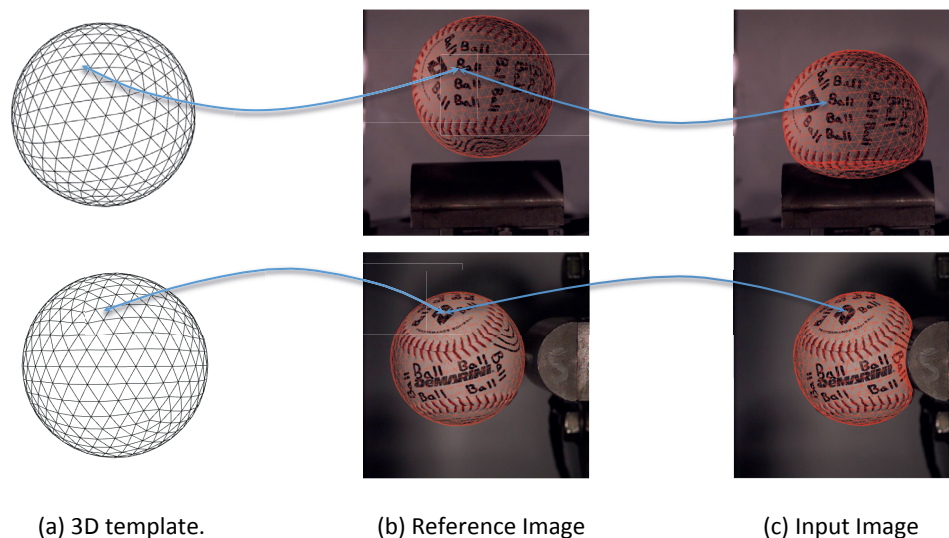


Figure 5.15: Template-based 3-D shape reconstruction. Given a 3-D template (a) representing the object’s 3-D shape in a reference image (b), we can match feature points on the surface as seen in the reference and input images (c), that is, measure how much they have shifted between the 2 images. These shifts can be used in turn to infer the 3-D surface deformation. **First row:** top view camera. **Bottom row:** side view camera.

ball center and is perpendicular to the half cylinder axis. Overall, we solve the following optimization problem:

$$\begin{aligned}
 & \underset{\mathbf{x}}{\text{minimize}} && E_{\text{side image}}(\mathbf{x}) & + & E_{\text{top image}}(\mathbf{x}) & + \\
 & && E_{\text{side contour}}(\mathbf{x}) & + & E_{\text{top contour}}(\mathbf{x}) & + \\
 & && E_{\text{reg}}(\mathbf{x}) & + & E_{\text{iso}}(\mathbf{x}) & \\
 & \text{subject to} && C_{\text{cylinder}}(\mathbf{x}) \leq \mathbf{0} & \text{ and } & C_{\text{symmetry}}(\mathbf{x}) = \mathbf{0} & .
 \end{aligned} \tag{5.23}$$

The term $E_{\text{side image}}(\mathbf{x})$ represents the geometric errors using feature point correspondences on the side view, $E_{\text{top image}}(\mathbf{x})$ represents the geometric errors using feature point correspondences on the top view. It is in contrast to the image term $\|\mathbf{M}\mathbf{x}\|^2$ presented in Chapter 3 which minimizes the linear least squares algebraic errors. This algebraic error term is useful for real-time performance purposes. However, to maximize the accuracy of the reconstruction, in this study, true geometric errors are minimized:

$$E_{\text{side image}}(\mathbf{x}) = \sum_{i=1}^N d(\mathbf{p}_i, \hat{\mathbf{p}}_i)^2, \tag{5.24}$$

$$E_{\text{top image}}(\mathbf{x}) = \sum_{j=1}^M d(\mathbf{p}_j, \hat{\mathbf{p}}_j)^2, \tag{5.25}$$

in which $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$ are the perspective projections of feature points $\mathbf{P}_i, \mathbf{P}_j$ on the ball surface on the image plane of the side view and the top view, respectively. The image points

$\mathbf{p}_i, \mathbf{p}_j$ are the corresponding points of $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$ on the side view image and the top view image, respectively, indicated by feature points correspondences found between the input images and the template images.

The terms $E_{\text{side contour}}(\mathbf{x}), E_{\text{top contour}}(\mathbf{x})$ represent the geometric errors between the projections of the ball's occluding contours and the ball's image contours on the side view and the top view. The occluding contours are computed online during optimization iterations. We detect edges in the input images by applying a simple Canny edge detector [Canny, 1986]. A more sophisticated technique implementing texture boundary detection [Shahrokni et al., 2005] could also be used.

We use our outlier rejection methods presented in Chapter 3 to reject erroneous correspondences. We also use the Tukey robust estimator with an automatically-chosen threshold to suppress small localization errors in feature points and occluding contours correspondences.

Because the two views' coordinate systems are related by a rigid transformation $[\mathbf{R}_{\text{stereo}}, \mathbf{t}_{\text{stereo}}]$, the variables vector \mathbf{x}_{top} in the top view can be expressed as a linear transformation of the variables vector in the side view: $\mathbf{x}_{\text{top}} = \mathbf{P} \cdot \mathbf{x}_{\text{side}} + \mathbf{q}$. The optimization is done with respect to the variables vector \mathbf{x} expressed in the side view coordinate system.

$E_{\text{reg}}(\mathbf{x})$ regularizes the solution using the Laplacian-based regularization matrix for non-planar objects. $E_{\text{iso}}(\mathbf{x})$ is the soft isometric deformation energy that only encourages the object surface to shrink or stretch around its reference configuration.

Inequality constraints $C_{\text{cylinder}}(\mathbf{x}) \leq \mathbf{0}$ do not allow the reconstructed ball shape to penetrate the half cylinder. Mathematically, they enforce the distance between every mesh vertex and the half cylinder axis to be greater than the half cylinder radius. We also enforce the symmetry property of the ball with respect to the vertical plane that goes through ball center and perpendicular to the half cylinder axis. Mathematically, $C_{\text{symmetry}}(\mathbf{x}) = \mathbf{0}$ requires the lines connecting symmetric points on the ball surface to be parallel to the cylinder axis.

We solve the optimization problem above using sequential quadratic programming with inequality and equality constraints. Gurobi optimization toolbox [Gurobi, 2012] is used for optimization.

5.1.2.2 Results

To quantify our reconstruction results, we set up an additional experiment with stereo cameras like in Fig. 5.16. We obtained 7000 frames-per-second videos such as the one of Fig. 5.17, which shows the very strong deformation that occurs at the moment of impact. The camera focal length is about 2850 pixels. We only use one view to compute

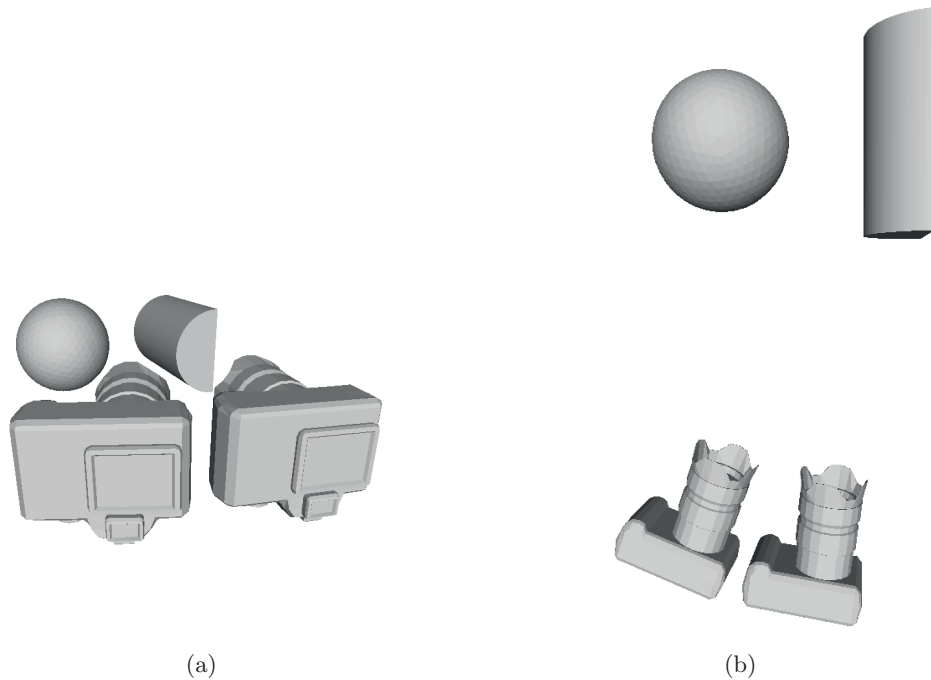


Figure 5.16: An experiment setup of the ball-bat study with stereo cameras seen from different viewpoints.

the shape of the ball by solving the optimization problem in Eq. 5.23 with one image term and with an additional term $E_{\text{traj}}(\mathbf{x})$ to account for depth ambiguity. This term encourages the ball to move in a straight line computed from frames in which the ball is undeformed. Since the horizontal speed of the ball is approximately 140 mph before impact and still 70 mph afterwards, we neglect gravity over this very short sequence and assume the ball keeps moving in a straight line. We therefore fit a line to the center of the ball in the frames in which it is undeformed and take $E_{\text{traj}}(\mathbf{x})$ to be the distance of the center of gravity from this line.

The results are shown in Fig. 5.17. We use the second sequence for validation purposes. To this end, we performed dense stereo reconstruction [Furukawa and Ponce, 2009] and show in Fig. 5.18 the median distance between the resulting 3-D point cloud and our monocular result. The undeformed ball diameter is 73.52 mm and the median distance hovers around 1% of that value except at the very beginning of the sequence when the ball is still undeformed but on the very left side of the frame. In fact, in those first few frames, it is the stereo algorithm that is slightly imprecise, presumably because the line of sight is more slanted. This indicates that it might not be perfectly accurate in the remaining frames either, thus contributing to some of the disagreement between the two algorithms.

To compute quantitative video-based reconstruction results used for comparisons with

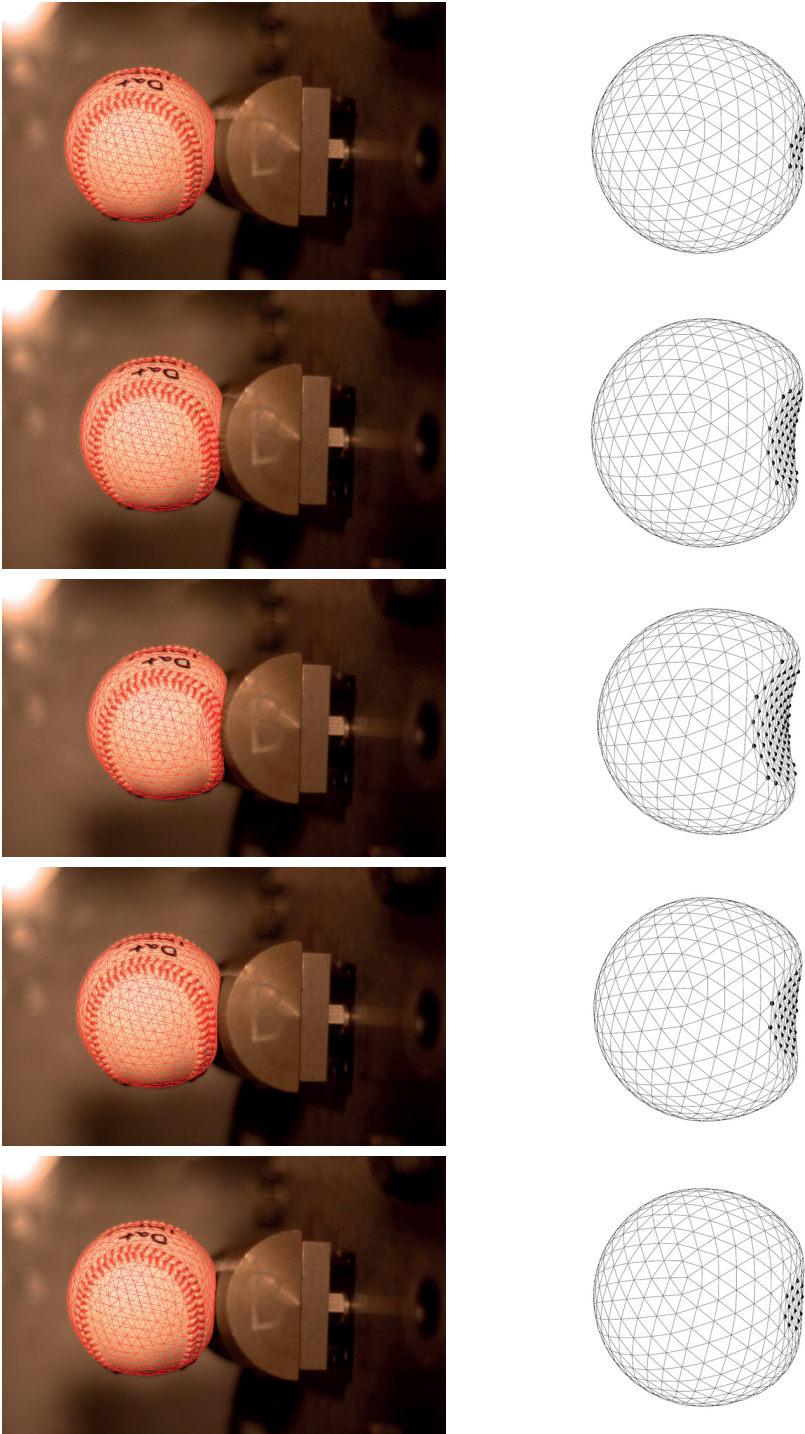


Figure 5.17: **Reconstruction of a ball colliding with a cylinder in a video sequence.** **Top row:** Reprojection of the reconstructed meshes for different frames in the sequence. **Bottom row:** The reconstructed meshes seen from a different view point. Black dots represent on-contact vertices.

5.1. Deformable Object Reconstruction in Interactions

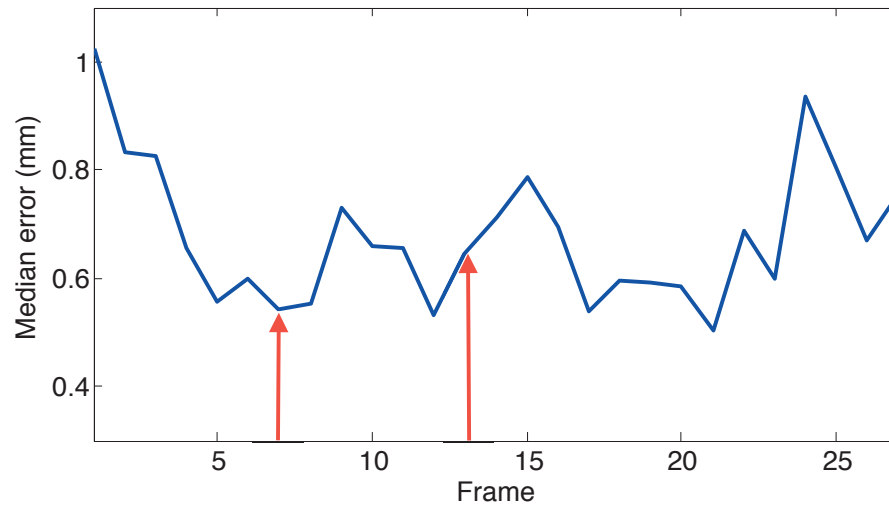


Figure 5.18: Median distance in each frame between our monocularly reconstructed surface and a 3-D point cloud obtained from stereo [Furukawa and Ponce, 2009]. The two red arrows indicate the frames in which the ball first and last touches the bat.

simulation-based results, we used image cues in both the side view and the top view as described in the previous section. Fig. 5.19 depicts our reconstruction results in which the ball reprojections match very well to the images in both views.

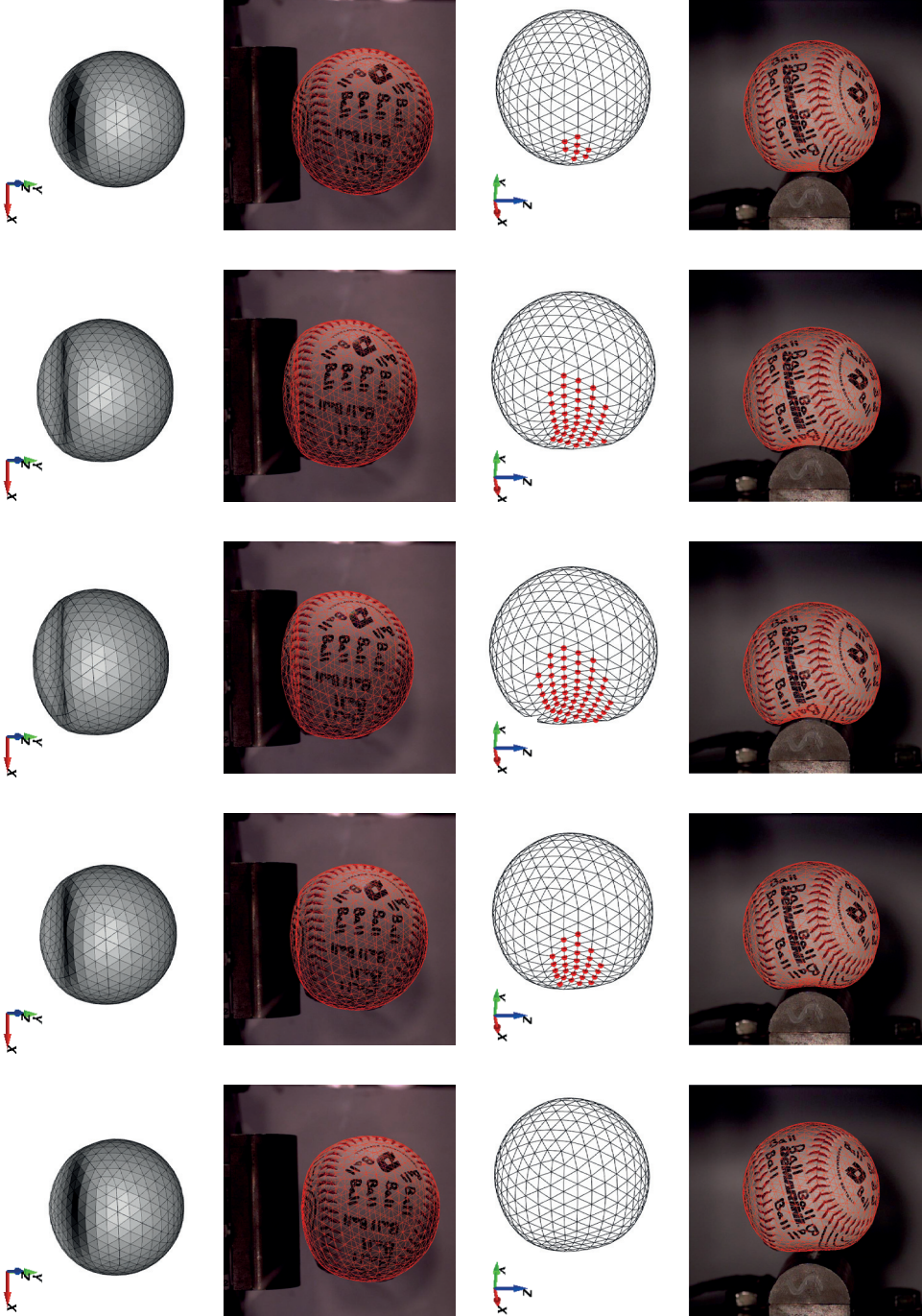


Figure 5.19: Video-based 3-D reconstruction of the ball colliding with the cylinder. First and third rows: Reprojection of the reconstructed mesh in different frames of the sequence in the side and top views. Second and fourth rows: The reconstructed meshes seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface. The red dots denote vertices touching the cylinder.

5.1.2.3 Comparisons to Simulations

The balls used for this work were solid adult slow-pitch softballs measuring 97.02 mm in diameter and weighing 190 gram. Softballs are made from high density polyurethane foam with a 1.5 mm thick leather covering. Since their response is dominated by the polyurethane foam they may be approximated as a homogeneous sphere comprised of a single material. This makes them ideal for evaluating finite element simulations, where their response is dominated by a single material behavior, and not influenced by material interactions as is the case with golf balls and baseballs. Thus, differences between the model and experiment may be directly attributed to material response.

Ball models were developed in the LS-DYNA [LS-DYNA, 2015] finite element code by mechanical and material researchers at Washington State University [Smith et al., 2015]. The ball was modeled using 7168 eight-noded solid brick elements. The solid steel cylinder was modeled with 4864 8-noded solid brick elements. A “surface to surface” contact was defined between the objects to prevent penetration during impact. The model included two symmetry planes to reduce solution time (see Fig. 5.20). Ball response was described using one of four material models: a linear visco-elastic model (MAT 006) with three variants [Smith and Duris, 2009, Nathan et al., 2011, Bryson and Smith, 2010], a low density foam material model (MAT 057) [Burbank and Smith, 2012], a medium density foam material model (MAT 083) [Burbank and Smith, 2012], and a third foam material model (MAT 181) developed for this work [Smith et al., 2015]. Parameters for the MAT 181 material model were identified phenomenologically using an iterative training process which attempted to minimize the difference between the simulated and experimentally observed force-displacement response and energy dissipation.

Video analysis of the impacts allows comparison of ball geometric features as a function of time. The experimental ball diameter in the direction parallel to the ball path, D_y (y direction in Fig. 5.20), the experimental ball diameter transverse to the ball path (D_x and D_z), are compared with the finite element models in Figs. 5.21, 5.22, and 5.23. It is apparent from the figures that the foam material models match the experimental deformation more closely than the visco-elastic models. The visco-elastic MAT 006 models exceed the observed diameter change, in some cases by large factors. Overall, the foam material model MAT 181 captures the deformation phase of D_x , D_y , and D_z best. This is also visually validated in Figs. 5.24, 5.25, 5.26, 5.27, 5.28, and 5.29.

The computer vision algorithms we used have provided unprecedented quantitative geometric comparison between computer simulations and experimental observation. Video-based analysis has thus proved an effective tool to validate computational models.

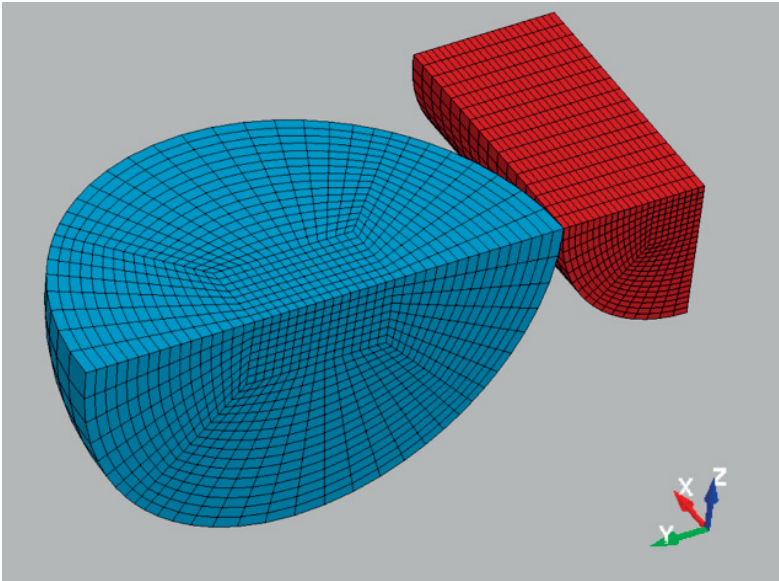


Figure 5.20: Views of a mesh of a softball impacting a solid cylinder, showing two planes of symmetry.

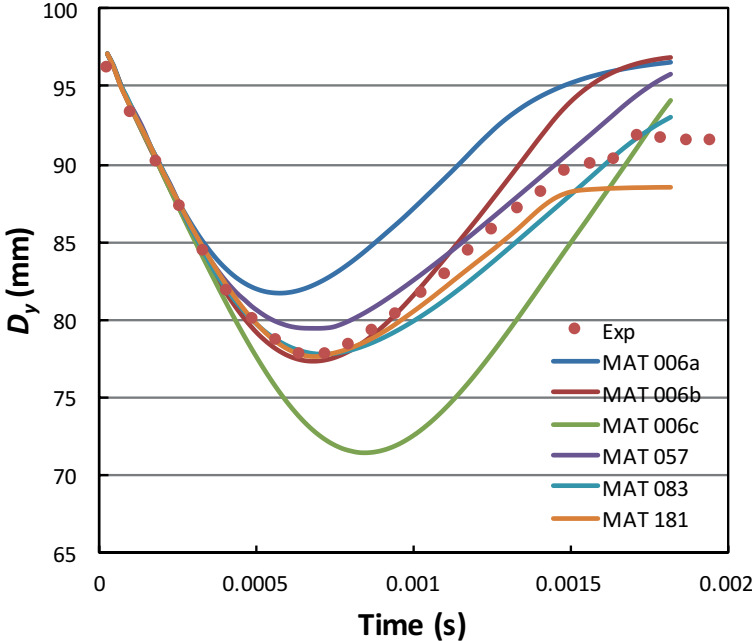


Figure 5.21: Comparison of the experimental ball diameter along O_y normal directions with the finite element models.

5.1. Deformable Object Reconstruction in Interactions

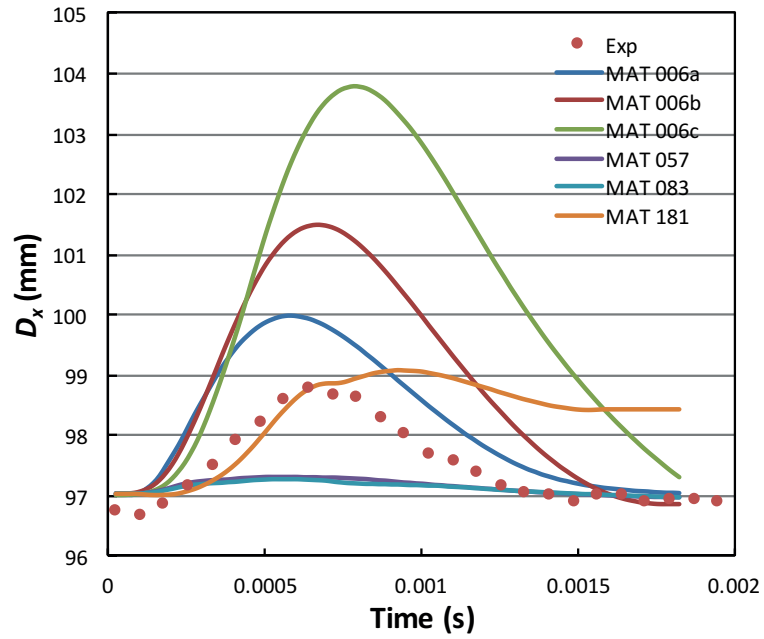


Figure 5.22: Comparison of the experimental ball diameter along O_x normal directions with the finite element models.

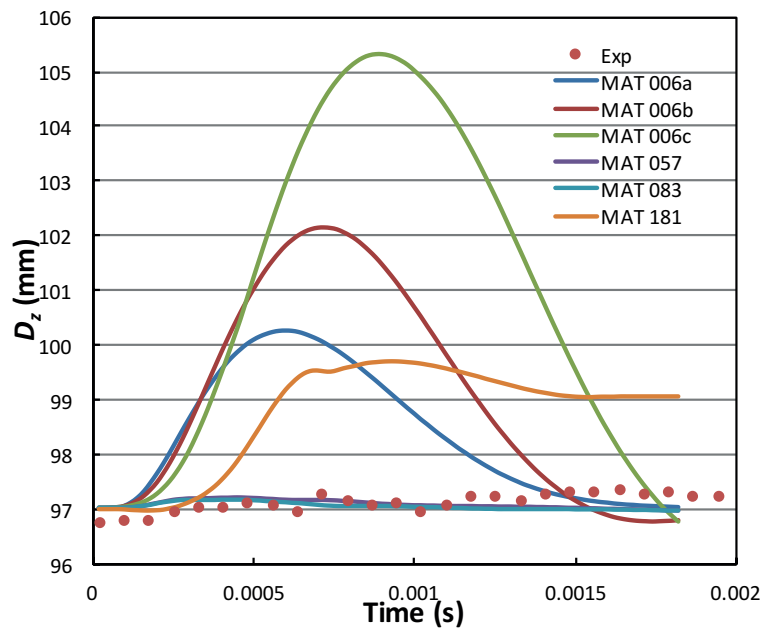


Figure 5.23: Comparison of the experimental ball diameter along O_z normal directions with the finite element models.

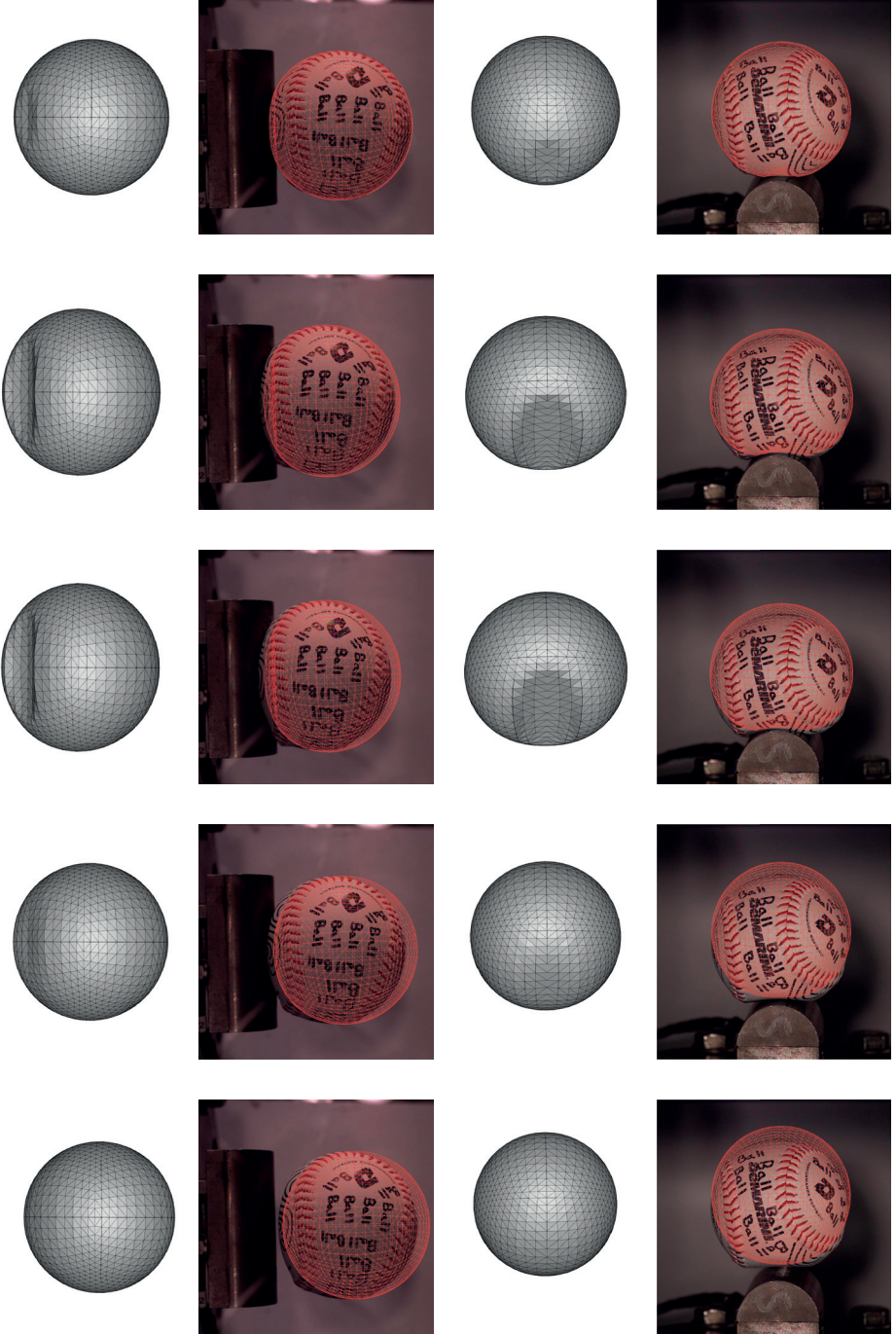


Figure 5.24: Simulation results with the visco-elastic material MAT 006a. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface.

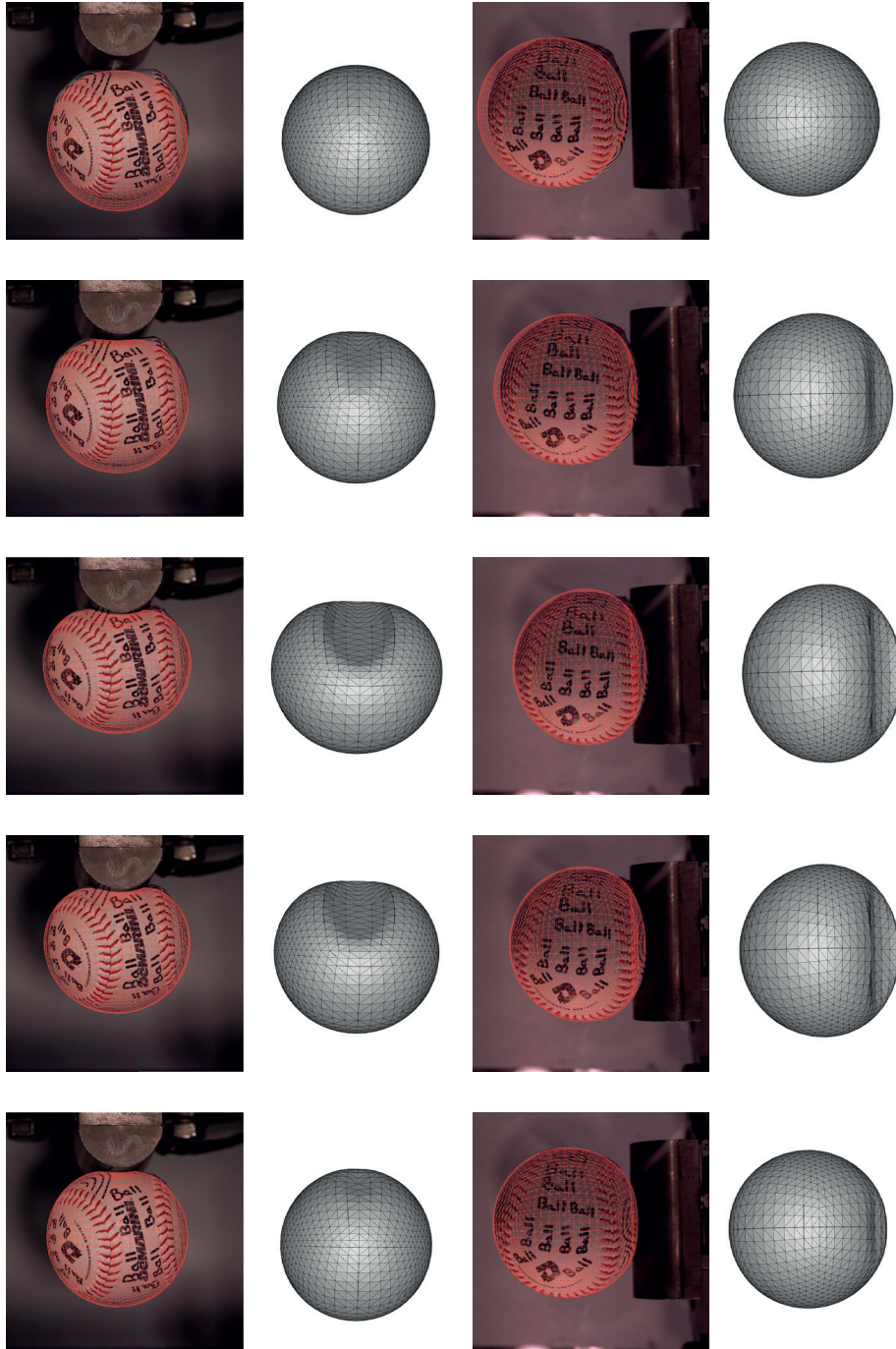


Figure 5.25: Simulation results with the visco-elastic material MAT 006b. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface.

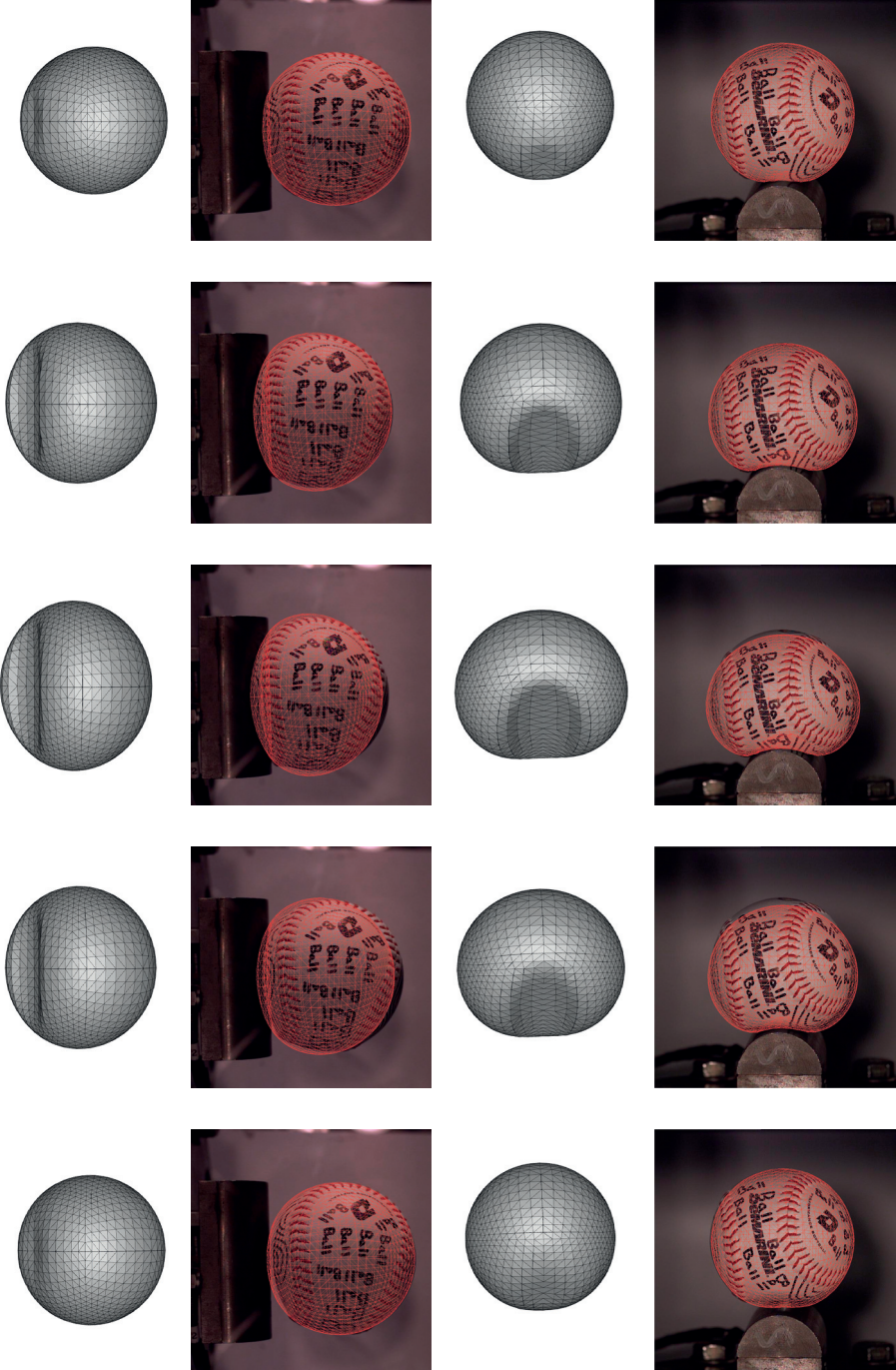


Figure 5.26: Simulation results with the visco-elastic material MAT 006c. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface.

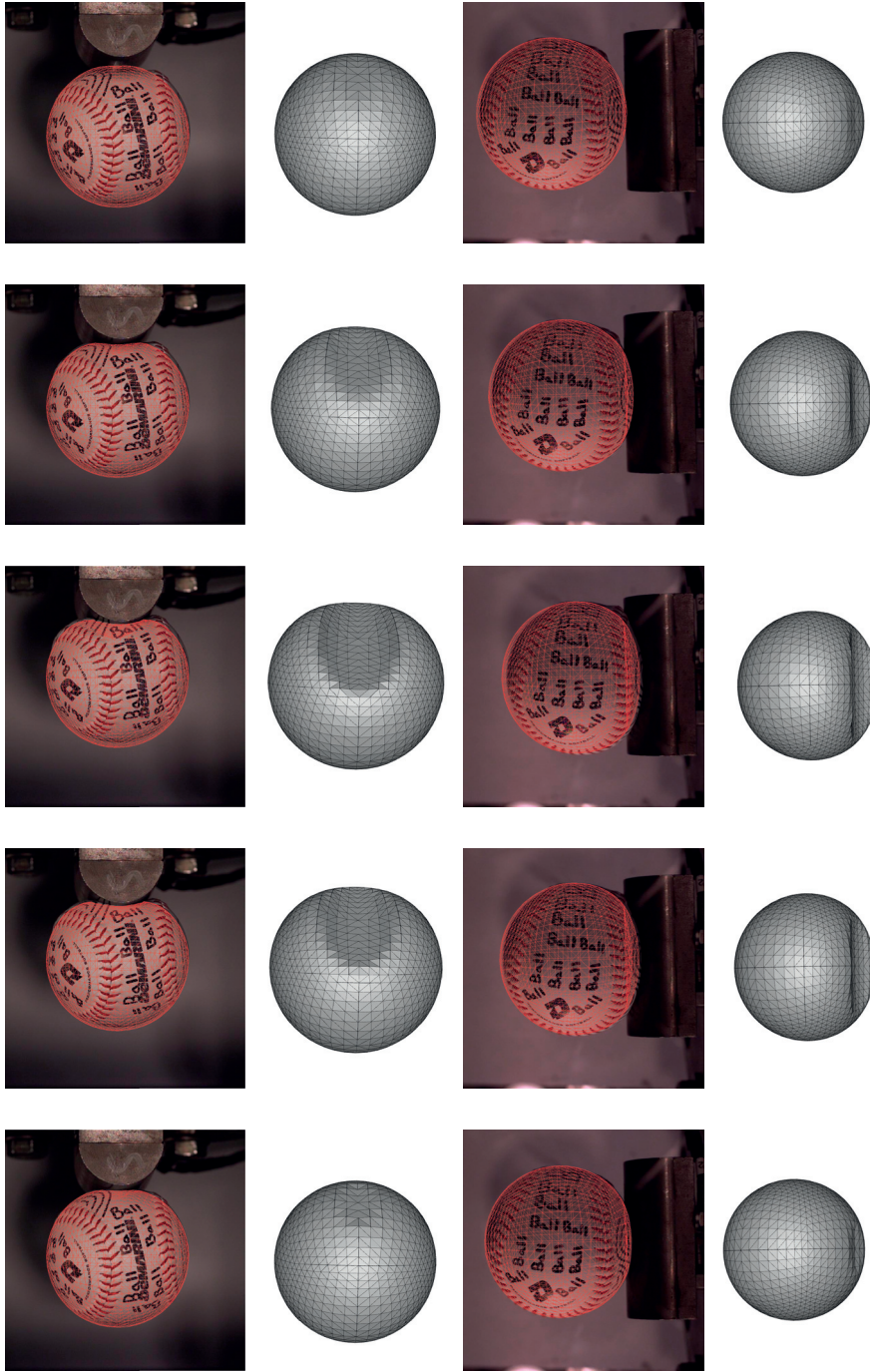


Figure 5.27: Simulation results with the foam-based material MAT 057. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface.

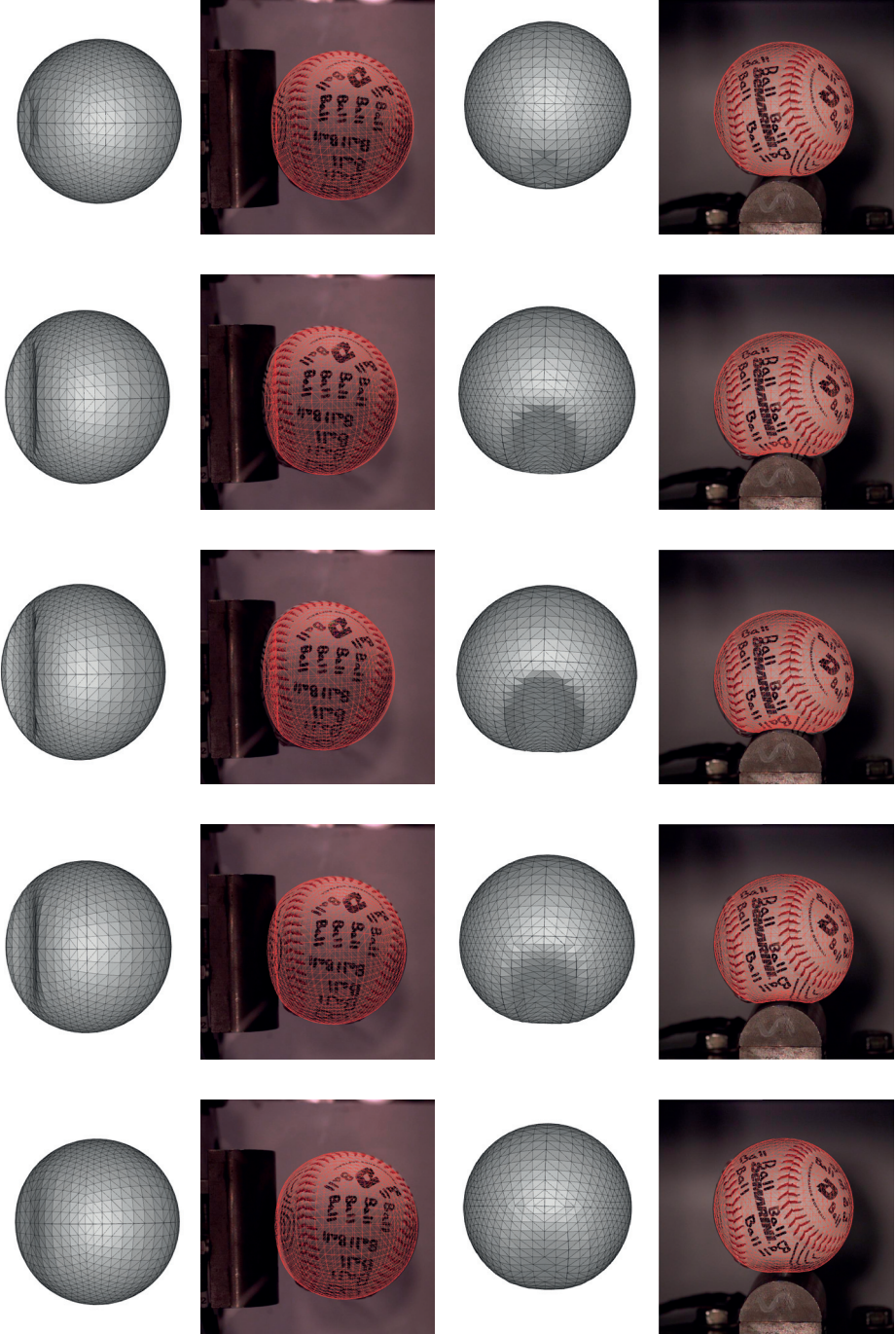


Figure 5.28: Simulation results with the foam-based material MAT 083. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface.

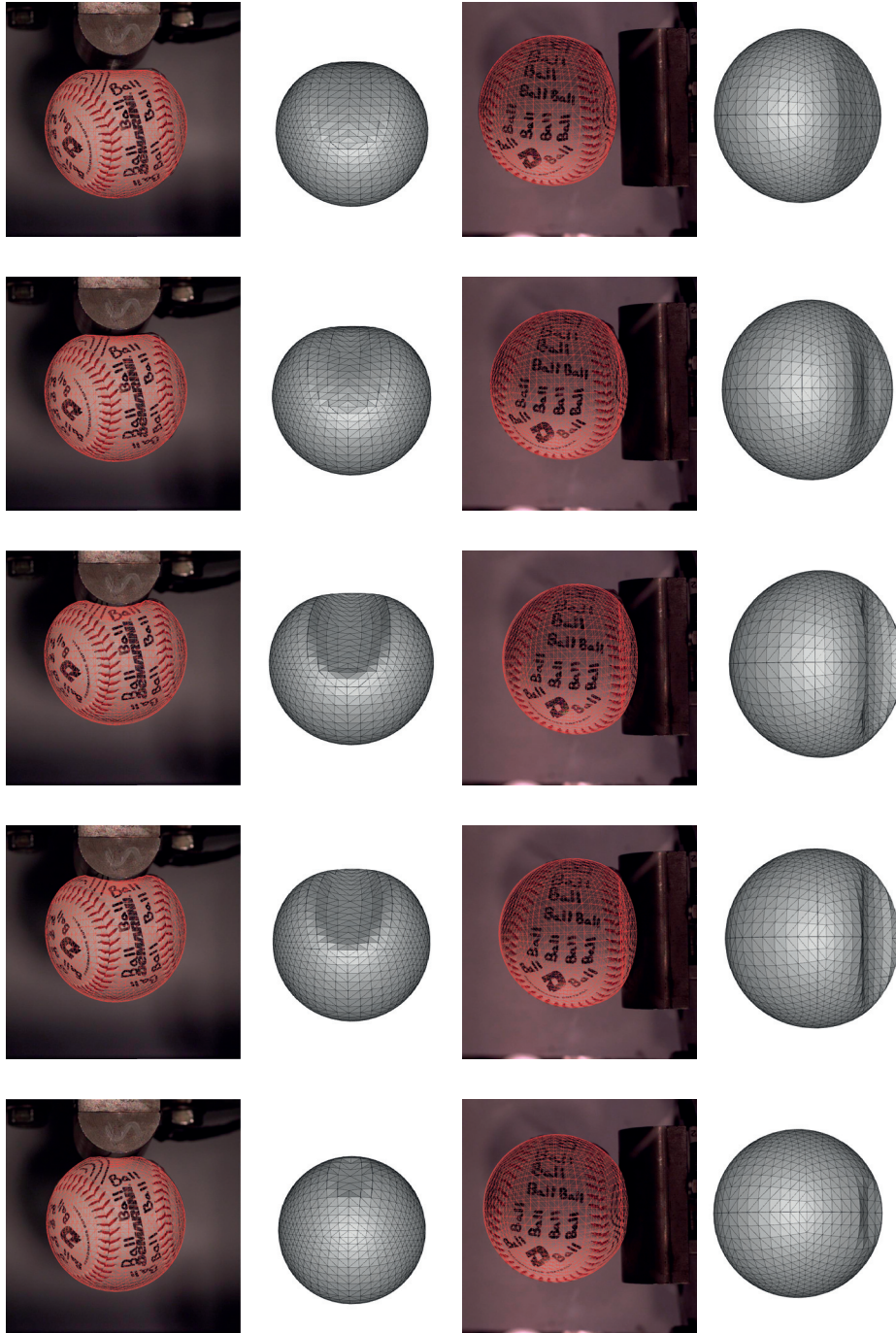


Figure 5.29: Simulation results with the foam-based material MAT 181. First and third rows: Reprojection of the simulation in different frames of the sequence in the side and top views. Second and fourth rows: The simulation seen from different viewpoints, rotated slightly about the y and x axis, respectively, to show the impact surface. This material model best matches with the observed videos.

5.1.3 Additional Qualitative Results

In this section, we present our qualitative results on modeling the interactions between a deformable cushion and a rigid pen in a video sequence. We used the dense matching based rigid object pose estimation approach presented in Section 5.1.1.2 to track the rigid pen and impose the constraints in recovering 3-D shape of the cushion. We enforce that the pen must not penetrate the cushion. We solve the following optimization problem:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && E_{\text{image}}(\mathbf{x}) + E_{\text{reg}}(\mathbf{x}) + E_{\text{iso}}(\mathbf{x}) \\ & \text{subject to} && C_{\text{pen}}(\mathbf{x}) \leq \mathbf{0} \quad , \end{aligned} \tag{5.26}$$

in which, like in Section 5.1.2, $E_{\text{image}}(\mathbf{x})$ is the geometric error term, $E_{\text{reg}}(\mathbf{x})$ is the regularization term, $E_{\text{iso}}(\mathbf{x})$ is the soft isometric deformation term, and $C_{\text{pen}}(\mathbf{x}) \leq \mathbf{0}$ enforces the cushion not to be penetrated by the pen.

To not allow the pen to penetrate the cushion, it is equivalent to enforce that all the mesh vertices of the pen reside in one certain side of the cushion. Since the cushion is represented by a triangle mesh, the problem becomes constraining each vertex of the pen mesh to be on one certain side of the plane that contains a facet of the cushion mesh. The signed 3-D distance between a 3-D point and a 3-D plane is well-defined.

To find which facet of the cushion mesh correspondences to a vertex of the pen mesh, we approximately use the *aabb* tree data structure [Bergen, 1997] to quickly find the intersection between the cushion mesh and a ray casted from camera center to the vertex of the pen mesh. This intersection defines the cushion mesh facet that corresponds to a 3-D plane that constraints each pen mesh vertex. We perform the search for these correspondences in every iteration of the optimization.

We solve the optimization problem above using sequential quadratic programming with inequality constraints. Gurobi optimization toolbox [Gurobi, 2012] is used for optimizing the total energy with respect to the constraints.

The qualitative results of the interactions between the cushion and the rigid pen are demonstrated in Fig. 5.1.3. Without imposing the pen-cushion non-interpenetration constraints, the pen goes through the reconstructed cushion, which violates the reality. Imposing the pen-cushion constraints gives us realistic results of the interactions.

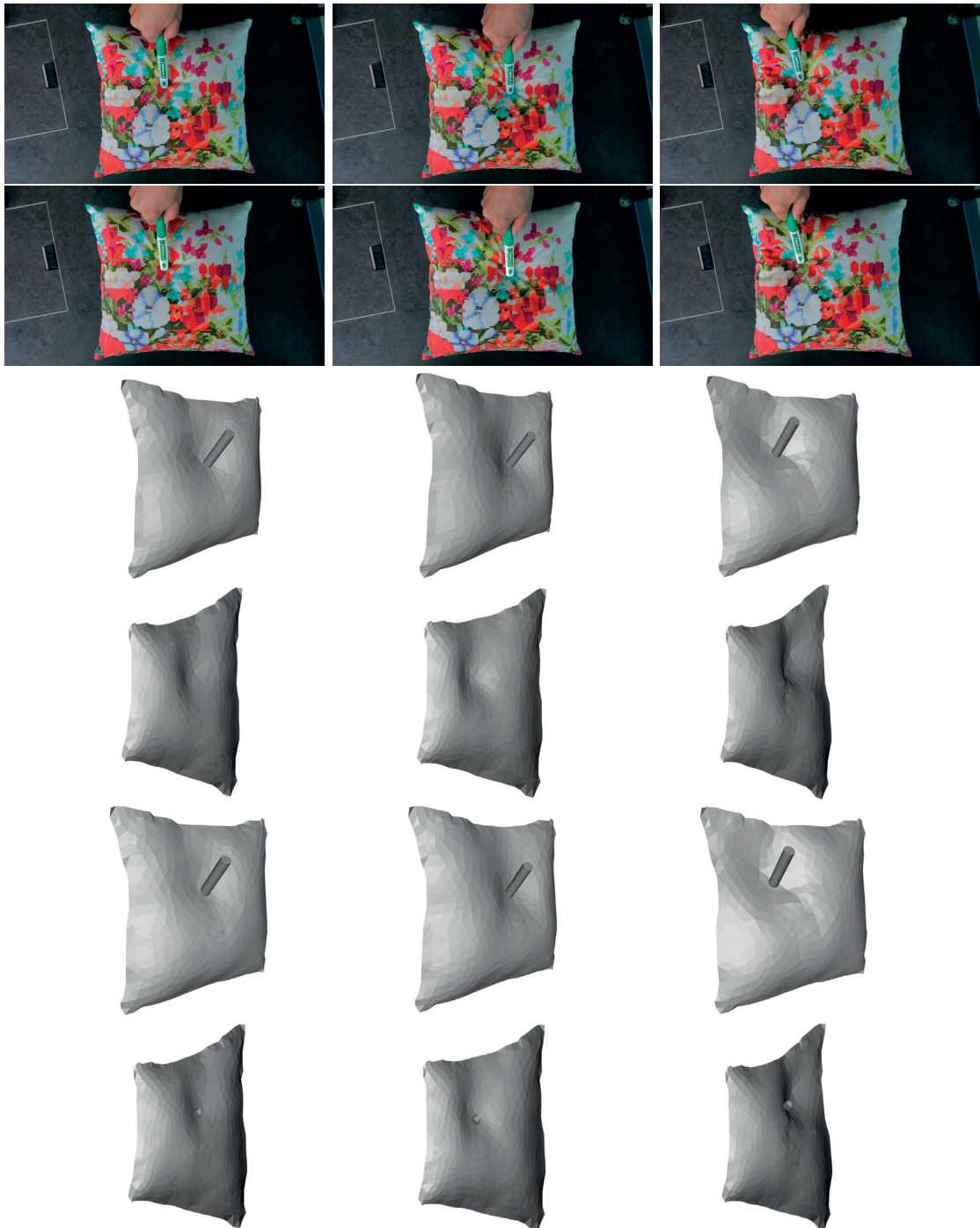


Figure 5.30: Tracking a rigid pen and reconstructing the cushion with and without imposing non-penetration constraints during their interactions. First row: input frames. Second row: OpenGL rendering of the pen on the input image, which is used as the template image for computing the pose of the pen in the next frame. Third and fourth rows: interaction of the cushion and the pen seen from front and back with imposing non-penetration constraints. Notice that the pen does not penetrate the cushion. Last two rows: interaction of the cushion and the pen seen from front and back without imposing non-penetration constraints. The pen does go through the cushion.

5.2 Real-time Template Selection

In our context of template-based deformable object tracking and reconstruction, we need to detect which template, among a set of them, appears in the camera. To do so, we compare the input image with a set of known templates. We detect a sparse set of feature points and use a voting scheme to determine which template is currently visible and should be selected for our template-based deformable object tracking and reconstruction.

Later in the shape reconstruction phase, we also need to establish feature point correspondences between the selected template and the input image. In our problem context, it is necessary to choose a suitable feature point detection, description, and matching method that has reasonable memory and CPU consumption, and provides good quality matches. We have tested a number of methods including Ferns [Ozuysal et al., 2010], BRIEF [Calonder et al., 2010], ORB [Rublee et al., 2011], FREAK [Alahi et al., 2012], BRISK [Leutenegger et al., 2011], SURF [Bay et al., 2006], and found BRISK to be the most suitable choice for our application. BRISK detects scale and rotation invariant feature points, the descriptors are binary vectors, which can be compared very quickly using NEON instructions widely available in mobile processors nowadays, and the matching quality is good enough for our purpose. Clark and Dunser [2012] use SURF in their augmented book application, which we found too computationally intensive. Our earlier work [Ngo et al., 2015] extracts scale and rotation invariant interest points as maxima of the Laplacian and then uses the Ferns classifier [Ozuysal et al., 2010] to do the matching. However, Ferns is so memory intensive that it is not suitable for our real-time tracking on mobile devices.

We detect about 500 BRISK feature points on each template image, and extract their binary BRISK descriptors. Doing so allows all templates to have an equal chance of being selected in the voting scheme. We automatically adjust parameters to extract between 300–2000 features points on the processed input image. The idea of our voting scheme is that each feature descriptor in the input image will vote for one template which has the closest descriptor according to the Hamming metric.

Although comparing two binary descriptors can be done very quickly using NEON instructions, performing brute-force search for the nearest neighbour in the template descriptor database is prohibitive. We instead project all binary descriptors to a low d -dimension space ($d = 7$ in our settings) using a pre-generated random projection matrix. K nearest neighbours ($K = 100$ in our settings) in this low dimension space can be searched very quickly using k-d trees. It also scales well with respect to the number of templates. We use the *libnabo* library [Elseberg et al., 2012] to perform K -nearest-neighbour search. We then go back to the original binary space and select the nearest descriptors among these K candidates. A Hamming threshold is used to filter truly good matches. The template with the highest vote is selected.

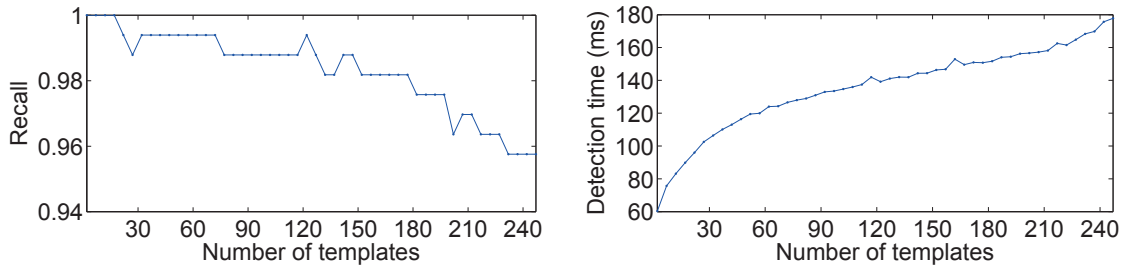


Figure 5.31: Scalability of the template detection algorithm. Left: recall measure. Right: average detection time.

To evaluate our template selection algorithm, we captured an image sequence of colored drawings in normal settings of book coloring and input it into the template selection module to test how accurately we can detect a template if it appears in the camera. To study the scalability of our detection algorithm, we vary the number of templates from 2 to 250, which is much more than the number of pages a coloring book usually has. We define the recall measure as the ratio of correct detection among the total number of test input images. In our case, false positives are not very severe because these false alarms will be rejected in the outlier rejection step. As shown in Fig. 5.31, our template selection algorithm can detect the template accurately and its computational performance scales well with the number of templates. Since we detect the template only once before passing the result to the tracking module, this detection can be run across several frames.

5.3 Lighting Estimation

Together with geometric registration, being able to estimate environment illumination would allow us to realistically render virtual objects with appropriate lighting into a real scene. It would drastically improve augmented reality experiences because virtual objects seamlessly blend with real objects under the real lighting condition. In this section, we describe our non-invasive method to compute the environment illumination for augmented reality purposes on deformable objects. We rely on our existing markerless real-time deformable object tracking system presented in Chapter 3 to collect lighting cues and estimate the environment illumination, also in real-time. As a result, we obtain real-time augmented reality experiences on deformable objects with realistic lighting.

Different approaches exist for acquiring real-world illumination. One such approach is to directly measure the incoming light by using artificial calibration objects such as reflective spheres or by using high dynamic range camera with fish-eye lenses. As a pre-processing step, an image of a reflective sphere or a high dynamic range image of the environment is captured and the environment illumination is pre-computed [Debevec, 1998]. This estimated illumination can be used to render virtual objects assuming that the illumination remains unchanged. It can also be done in real-time by adding a

mirror sphere into the scene at a known location, for example, on top of a 2-D marker [Kanbara and Yokoya, 2004]. However, the mirror sphere will obscure the real scene and negatively affect the Augmented Reality experiences. Similarly, Sato et al. [1999] relied on omni-directional stereo cameras, which requires specialized hardware instead of the ordinary camera that we use.

Aittala [2010] captured incoming lighting using a diffuse spherical light probe. He represents the incoming illumination by a set of regularly sampled point light sources on a sphere. The powers of these light sources are then solved for by relating an over-complete diffuse lighting basis and the pixel intensity observations from the camera images using L_1 -regularized least squares minimization. However, our experiments showed that the problem is very ill-posed and the solution varies a lot with different regularization weights. Gruber et al. [2012] presented an approach to inverse lighting for arbitrary scene geometry without relying on special light probes. It, however, requires an RGB-D camera in the KinectFusion framework [Newcombe et al., 2011] and limits the light color to be white. Recently, Knorr and Kurz [2014] proposed a method to estimating the real-world illumination based on human face images. The approach relies on learning a face-appearance model from a dataset of faces under known lighting conditions. At run-time, it recovers the most plausible real-world lighting conditions for measured reflections on a face, represented by spherical harmonic functions.

In contrast to previous methods, we take advantage of the fact that our deformable object effectively samples the space of surface normals. These normals together with observed image intensities allow us to reconstruct an environment irradiance map, which we will use to illuminate diffuse virtual objects. The illumination estimated by our photometric calibration may not be as accurate as those obtained with such lighting probes but it is amply sufficient to synthesize realistic augmented images while being much more light-weight and non-invasive.

Our method is related to [Pilet et al., 2006], which tracks a rotating planar matte object and computes irradiance values in a set of surface normals. [Pilet et al., 2006] uses an interpolation scheme to infer irradiance between normals. By contrast, we use spherical harmonics to represent the environment irradiance map. In this way, we can compute the irradiance map more realistically and much faster, while the algorithm is more robust and scales better with the number of observations.

5.3.1 Method

We assume that the deformable object we track, such as a book page, exhibits a diffuse reflectance, meaning that it reflects lights equally in all directions. As a result, the image intensity of the object at a given surface point only depends on the surface normal \vec{n} at that point and camera shutter speed or aperture, but not on the camera pose. Given

arbitrary L directional light sources, their corresponding radiosity, or power, $\Psi_{l=\{1..L\}}$, and their corresponding direction \vec{d}_l , the irradiance ε at the surface point p and time t can be written as a function of the light intensity Ψ_l , the surface normal \vec{n}_t , and the light direction \vec{d}_l :

$$\varepsilon_{p,t} = \sum_{l=1}^L \Psi_l \cdot \max(\langle \vec{n}_t, \vec{d}_l \rangle, 0) \quad . \quad (5.27)$$

We assume that the camera responds linearly with respect to the irradiance. The image intensity observed at the surface point p is:

$$I_{p,t} = g \cdot \alpha_p \cdot \varepsilon_{p,t} + \beta \quad , \quad (5.28)$$

in which α_p is the surface albedo at surface point p , g is the camera gain, and β is the camera bias. In a multi-camera setup, the camera gain is used to account for different shutter speeds and apertures of different cameras. In our single camera settings, we can just set the camera gain to be one because it can be represented together by the surface albedo α_p . The surface albedo can be taken as an image of the object captured in an ambient lighting condition or its texture image.

To be more robust to small localization error and image intensity noise, instead of using a single image point, we measure the average intensity of a small circular patch π around the image point of interest. Assuming that the patch is planar and thus all the patch pixels share the same surface normal vector, Eq. 5.28 becomes:

$$I_{\pi,t} = \alpha_{\pi} \cdot \varepsilon_{\pi,t} + \beta \quad . \quad (5.29)$$

Recovering the radiosity of all L directional light sources given only image observations of a diffuse surface is a severely ill-posed problem because the diffuse surface acts as a low pass filter for the incoming light. Instead, we are interested in estimating irradiance $\varepsilon_{\vec{n}}$ in all possible direction of surface normal vectors \vec{n} . To do so, Pilet et al. [2006] linearize Eq. 5.28 by changing variables. Let

$$\begin{aligned} g' &= \frac{1}{g} , \\ b' &= \frac{b}{g} . \end{aligned} \quad (5.30)$$

For each image patch of the object, Pilet et al. [2006] rewrites Eq. 5.28 as:

$$-I_{\pi,t} \cdot g' + \alpha_{\pi} \cdot \varepsilon_{\vec{n}} + b' = [-I_{\pi,t}, \alpha_{\pi}, 1] \begin{bmatrix} g' \\ \varepsilon_{\vec{n}} \\ b' \end{bmatrix} = 0 \quad . \quad (5.31)$$

Given many image observation of patches at different normals, Pilet et al. [2006] put all those equations above together to yield a large but sparse linear system whose solution is the eigenvector associated to the smallest eigenvalue. By setting camera gain to one, camera bias and irradiance in the direction of sampled normals \vec{n} are computed. A smooth interpolation scheme is used to infer the irradiance in the direction between sampled normal vectors. One such irradiance map obtained by [Pilet et al., 2006] is shown in Fig. 5.32.

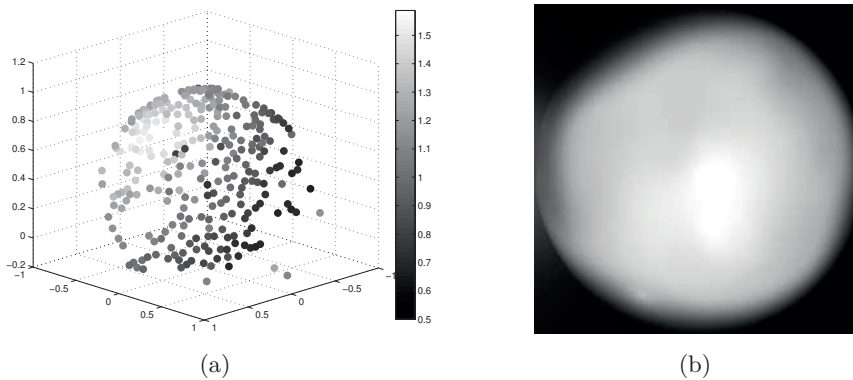


Figure 5.32: Irradiance map computation by [Pilet et al., 2006]. (a) The irradiance values $\varepsilon_{\vec{n}}$ obtained by solving Eq. 5.31. (b) Interpolating these values yields this irradiance map. Images courtesy of Pilet et al. [2006].

[Pilet et al., 2006] ignores the inherent relationship between $\varepsilon_{\vec{n}}$ and has to solve a very large linear system with a lot of unknowns. It does not scale well with the number of observations because one irradiance value has to be solved for each observation.

We were inspired by the work of Ramamoorthi and Hanrahan [2001], who presented a theoretical analysis of the relationship between incoming radiance and irradiance, both of which are represented using a basis of spherical harmonics. They showed that the irradiance can be viewed as a convolution of the incident radiance and a clamped cosine transfer function. As a result, the irradiance is insensitive to high frequency modes and can be well approximated using only nine parameters, achieving average errors of only 1%. Similarly, we represent the irradiance map in our problem context as a linear combination of the first nine spherical harmonic functions.

Analogous to the Fourier basis on the line, spherical harmonics Y_l^m , with $l \geq 0$ and $-l \leq m \leq l$, form an orthogonal basis on the surface of a sphere. The first nine spherical harmonics are simply constant ($l = 0$), linear ($l = 1$), and quadratic ($l = 2$) polynomials of Cartesian coordinates (x, y, z) . A visualization of these spherical harmonics is depicted

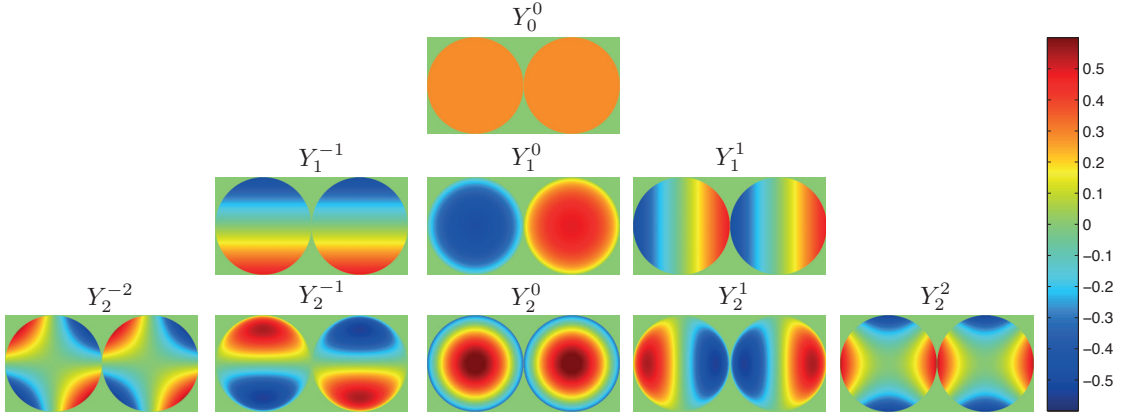


Figure 5.33: Visualization of the first nine spherical harmonics projected on Oxy plane. The left half of each sub-figure corresponds to negative z . The right half of each sub-figure correspond to positive z .

in Fig. 5.33. More specifically, they are given in the numerical form:

$$\begin{aligned}
 (x, y, z) &= (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta) \\
 Y_0^0(\theta, \phi) &= \frac{1}{2\sqrt{\pi}} \\
 Y_1^{-1}(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{3}{\pi}} y \\
 Y_1^0(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{3}{\pi}} z \\
 Y_1^1(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{3}{\pi}} x \\
 Y_2^{-2}(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{15}{\pi}} xy \\
 Y_2^{-1}(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{15}{\pi}} yz \\
 Y_2^0(\theta, \phi) &= \frac{1}{4}\sqrt{\frac{5}{\pi}} (3z^2 - 1) \\
 Y_2^1(\theta, \phi) &= \frac{1}{2}\sqrt{\frac{15}{\pi}} zx \\
 Y_2^2(\theta, \phi) &= \frac{1}{4}\sqrt{\frac{15}{\pi}} (x^2 - y^2)
 \end{aligned} \tag{5.32}$$

We write the irradiance map E , which encodes irradiance values $\varepsilon_{\vec{n}}$ in all possible directions \vec{n} , as a linear combination of these nine spherical harmonics.

$$E(\theta, \phi) = \sum_{l,m} e_{lm} Y_{lm}(\theta, \phi) \quad \text{with } 0 \leq l \leq 2, -l \leq m \leq l \tag{5.33}$$

And we want to find the values of the nine coefficients e_{lm} and camera bias β such that they explain best our set of image observations in Eq. 5.29. Rewriting Eq. 5.29 with respect to variables e_{lm} and β , we get:

$$I_{\pi,t} = \alpha_{\pi} \cdot \sum_{l,m} e_{lm} Y_{lm}(\theta, \phi) + \beta \quad . \quad (5.34)$$

Stacking all image observations and re-writing them in a matrix form, we get an over-determined linear system with respect to the vector of variables $\mathbf{x} = [e_0^0, e_1^{-1}, e_1^0, e_1^1, e_2^{-2}, e_2^{-1}, e_2^0, e_2^1, e_2^2, \beta]$

$$\mathbf{D}\mathbf{x} = \mathbf{b} \quad , \quad (5.35)$$

which can be solved in a least squares sense. However, in our single camera settings, our deformable surfaces' normals can usually cover less than a hemisphere corresponding to negative n_z in the camera coordinate system. We do not have image data to compute the irradiances corresponding to positive n_z . The linear system above is thus ill-posed with its solution varying given different sets of image observations. To regularize the solution, we instead solve the following linear least squares problem:

$$\underset{\mathbf{x}}{\text{minimize}} \quad \|\mathbf{D}\mathbf{x} - \mathbf{b}\|^2 + \lambda \|\mathbf{R}\mathbf{x}\|^2 \quad . \quad (5.36)$$

in which

$$\mathbf{R} = \begin{bmatrix} \mathbf{I}_{9 \times 9} & 0 \\ 0 & 0 \end{bmatrix} \quad (5.37)$$

The intuition behind the regularization term is to prefer a simple explanation of the lighting. The idea is similar to preferring small coefficients of deformation modes in recovering deformable shapes using linear shape models [Salzmann et al., 2008b].

The linear least squares problem above can be solved very efficiently because there are only 10 variables. In case of color images, the irradiance map is solved for each color channel independently. They are then assembled to create a color irradiance map. We can use the environment irradiance map to illuminate virtual objects by using a short OpenGL Shading Language (GLSL) script that specifies a color for each mesh vertex. Potentially, diffuse soft shadows can also be created with a ray tracer, making the augmented contents look more realistic.

5.3.2 Results

In this section, we evaluate our irradiance map estimation algorithm, compare it against [Pilet et al., 2006], and present some augmented reality results in which diffuse virtual

objects are illuminated realistically.

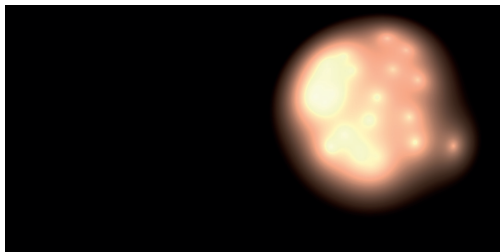
Because [Pilet et al., 2006] only works with planar rigid objects, we acquired a video sequence in which a rigid planar object is rotated in the camera field of view sampling the space normals. An example frame in the sequence is shown in Fig. 5.34(b). In Fig. 5.34(a), an object was put into the scene to demonstrate the lighting direction indicated by the casted shadow. We computed the irradiance map using Pilet et al. [2006]’s method and our method on the same sets of 50 and 300 images. The comparisons are given in Fig. 5.34. It is seen that our method requires less image data to be able to estimate a reasonable irradiance map. Our irradiance maps are also smoother, which is consistent with the fact that diffuse surfaces act as low pass filters of incoming light.



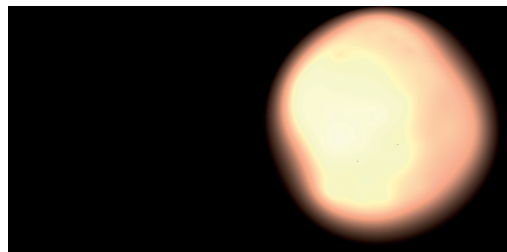
(a) The casted shadow indicates the lighting direction. The world coordinate system coincides with the camera coordinate system



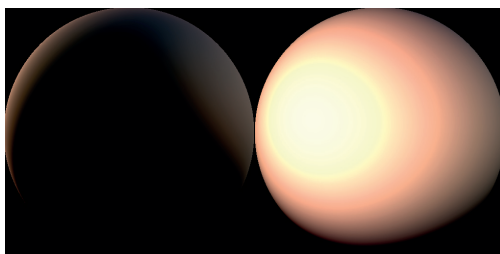
(b) An example frame in the sequence. [Pilet et al., 2006] can only work on planar rigid objects.



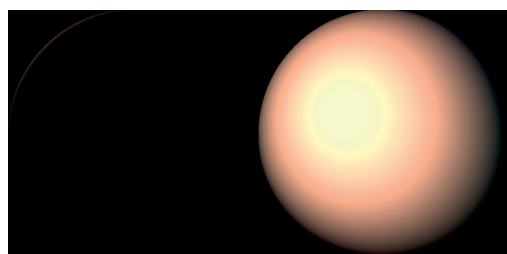
(c) [Pilet et al., 2006] with 50 planar poses



(d) [Pilet et al., 2006] with 300 planar poses



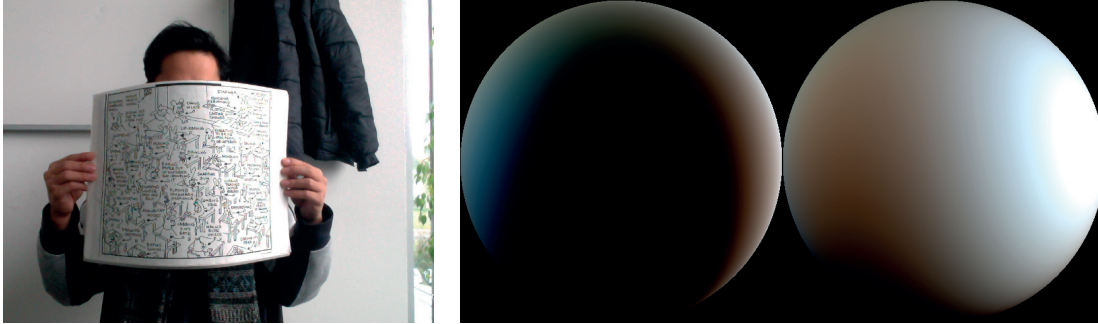
(e) Our method with 50 planar poses



(f) Our method with 300 planar poses

Figure 5.34: Comparisons of methods estimating the environment irradiance map (second and third rows) on the same set of image observations. The left (right) half of the irradiance map corresponds to positive (negative) n_z of the normal vector \vec{n} .

In another setting, we estimated the irradiance map on a sequence in which the scene was illuminated by a large window on the right side of the camera. The result is shown in Fig. 5.35. The recovered irradiance map with a bright region on the right is consistent with the real lighting.



(a) An example frame in the sequence.

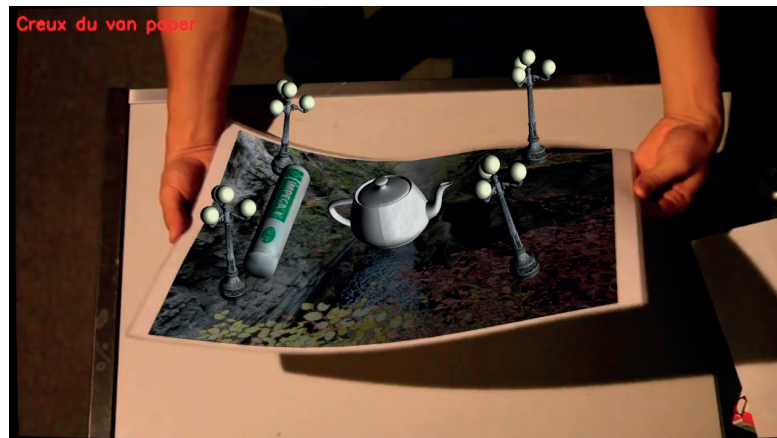
(b) Our recovered irradiance map.

Figure 5.35: Estimating environment irradiance map of an office. There is a large window on the right side of the camera that illuminates the scene.

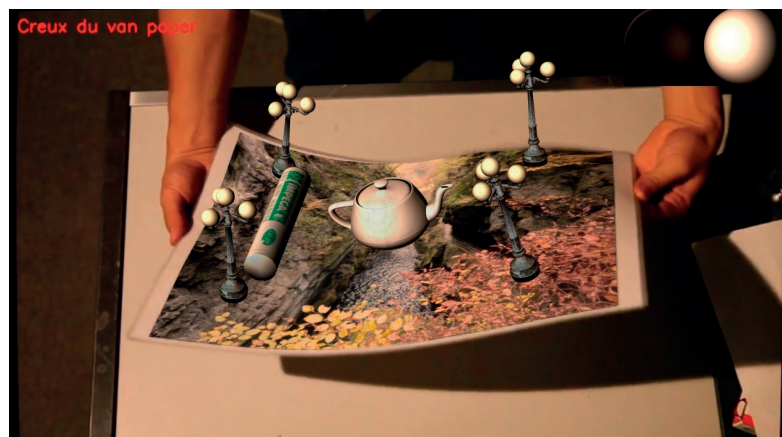
Fig. 5.36 demonstrates our augmented reality scenario in which diffuse virtual objects are illuminated properly accordingly to the recovered environment irradiance map. As a future work, we could use a ray tracer to create diffuse soft shadows. It would make the virtual objects look more realistic.



(a) An original input frame in the sequence.



(b) The surface is re-textured and virtual objects are illuminated by a random illumination. The shading of the teapot is not consistent with the casted shadow.



(c) The surface is re-textured and virtual objects are illuminated realistically by the recovered illumination.

Figure 5.36: Comparison of augmented reality with and without environment illumination estimation.

5.4 Coloring Book Application

Coloring books capture the imagination of children and provide them with one of their earliest opportunities for creative expression. However, the results in traditional coloring books look dead, static, and uninteresting. Given the popularity of television and digital devices, this activity seems unexciting in comparison. Augmented Reality (AR) holds unique potential to impact this situation by providing a bridge between real-world activities and digital enhancements. AR allows us to use the full power and popularity of digital devices in order to direct renewed emphasis on traditional activities like coloring.

In this section, we present an AR coloring book App that provides a bridge between animated cartoon characters and their colored drawings. Children color characters in a printed coloring book and inspect their work using a consumer-grade mobile device, such as a tablet or a smart phone. The drawing is detected and tracked, and the video stream is augmented with an animated 3-D version of the character that is textured according to the child's coloring. Fig. 5.37 shows two 3-D characters textured based on the input 2-D colored drawings.

Accomplishing this goal requires addressing several challenges. First, the 2-D colored drawing provides texture information only about the visible portions of the character. Texture for the occluded regions, such as the back side of the character, must be generated. Naive approaches, such as mirroring, produce poor results since features like the character's face may be mirrored to the back of their head. In addition, without special treatment, texture mapping will exhibit visible seams where different portions of the parameterization meet. Second, we target live update, so that colored changes are immediately visible on the 3-D model as the child colors. Thus, the texturing challenges must be solved with a very limited computation budget. Third, the pages in an actual printed coloring book are not flat but exhibit curvature due to the binding of the book. As a result, tracking algorithms and texture capture must be robust to page deformation in order to properly track the drawings and lift texture from the appropriate 2-D regions.

Our coloring book App addresses each of these technical challenges. We present a texturing process that applies the captured texture from a 2-D colored drawing to both the visible and occluded regions of a 3-D character in real time while avoiding mirroring artifacts and artifacts due to parameterization seams. We use the deformable surface tracking and 3-D shape recovery presented in Chapter 3 for the drawings. We also present a content creation pipeline to efficiently create the 2-D and 3-D content used in App.

Our method for live texturing an AR character from a colored drawing updates the texture of the 3-D character at every frame by copying pixels from the drawing. To do so, we create a UV lookup map which, for every pixel of the texture, indicates a pixel coordinate in the drawing. As the drawing lies on a deformable surface, the latter procedure operates on a rectified image of the drawing. We split this process into two

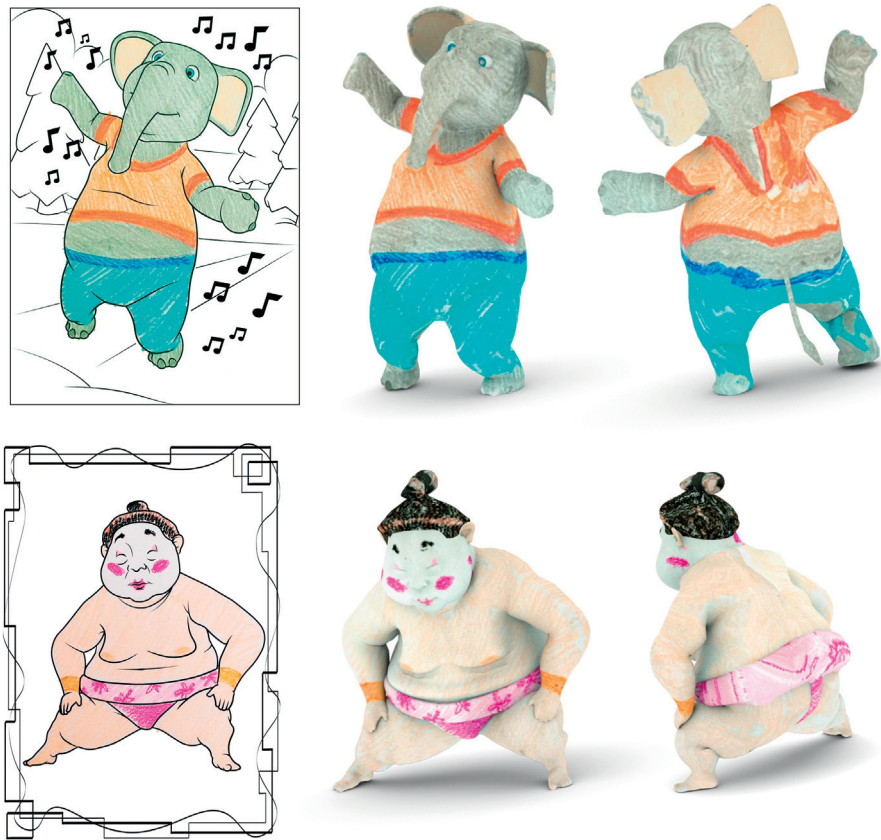


Figure 5.37: Two examples of our augmented reality coloring book algorithm showing the colored input drawings and the captured texture applied to both visible and occluded regions of the corresponding 3-D characters.

separate pipelines, as shown in Fig. 5.38. A static *content creation* pipeline creates a lookup map from the work of artists. This pipeline needs to be run only once. A *live* pipeline tracks the drawing and overlays the augmented character on top of it, with a texture created from the drawing in real time using the lookup map. This work was done in collaboration with Disney Research Zurich, who is mainly responsible for the lookup map generation pipeline.

5.4.1 Content Creation

This process aims at finding suitable lookup coordinates on the drawing for the parts of the mesh that are not visible in the drawing, given a UV-mapped mesh and its projection as a drawing. We want to find a lookup map that generates a texture that is continuous over the boundary between hidden and visible parts of the mesh in the drawing and over seams. We also want to fill hidden parts with patterns similar to the ones on visible parts, with minimal distortions between them.

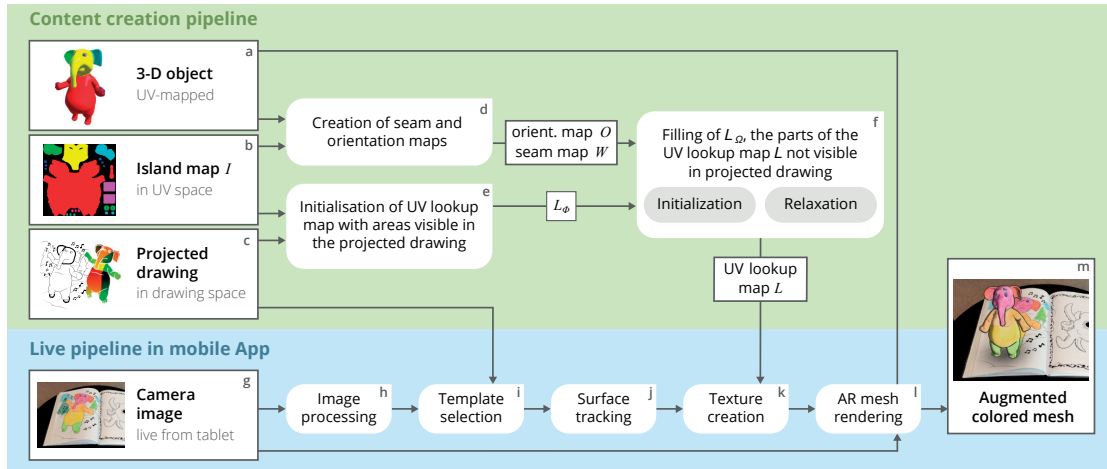


Figure 5.38: The static content creation and the in-App live surface tracking, texturing, and rendering pipelines.

The artist produces:

1. a UV-mapped mesh (Fig. 5.38a)
2. an island map I in UV space that defines regions that shall receive similar coloring (Fig. 5.38b)
3. a drawing, which is a combination of a view of the mesh through an edge shader – for printing – and a mapping from coordinates on this drawing to points in the UV map (Fig. 5.38c).

Based on these three items, the following objects are created:

1. the wrapping seams W^i for island I^i , mapping some pixels of I^i together
2. the orientation map O in the UV space, giving a local orientation of the projected mesh structure
3. the lookup map L , indicating for every pixel of the character texture which drawing pixel to read. $L = L_\Phi \cup L_\Omega$, where L_Φ are regions visible in the drawing (source regions) and L_Ω are regions not visible in the drawing (target regions).

The most challenging part of the content creation pipeline is the creation of L , based on the mesh, the projected drawing, and the island map I . To do so, W , O (Fig. 5.38d), and L_Φ (Fig. 5.38e) are first created. Then, for every island I^i , a first approximation of L_Ω^i is generated by copying coordinates from L_Φ^i . This approximation is very rough and

violates the continuity desideratum, in particular across seams. Therefore, the next step is to enforce this continuity (Fig. 5.38f).

We can frame this problem in a similar way as the one of energy minimization in a spring system. Assuming that every pair of neighboring points of L —including across seams—are connected by a virtual spring, relaxing all springs will lead to fulfill the continuity constraints. To do so, we have to minimize the total energy of the system:

$$\sum_{(p,q) \in N_L} k_{p,q} (\|L[q] - L[p]\| - 1)^2 \quad (5.38)$$

for all pairs p, q which are either direct neighbors or seam neighbors in L ; $k_{p,q}$ being a constant specific to the pair (p, q) that compensates the texel density in the UV map, which is not constant compared to the density in the model space. This correction is necessary for the generated texture motifs to be unstretched.

5.4.2 Image Processing

In the first step of the live pipeline, we apply image processing techniques to the input image so that the appearance of the colored drawing becomes as close to the original template as possible. Because in the next step of the pipeline we describe feature points by binary descriptors using intensity comparisons, this step is especially important to be able to better detect which drawing template is selected and to provide more inlier wide-baseline feature point correspondences between the input image and the reference image.

In the coloring book context, we want to remove colors and keep the black lines in the image of the colored drawing. Therefore, we transform the input image from RGB to HSV color space, in which only the luminance channel is used because it captures most information about the original black line draws. Line draws are more visible in the HSV luminance channel than in the gray scale image as shown in Fig. 5.39. We then apply adaptive thresholding to the luminance image to get a binary line draw image. Small noisy connected components are removed and Gaussian smoothing with standard deviation $\sigma = 1$ pixel is used to remove the staircase effect of binarization. This process is demonstrated in Fig. 5.39.

Our color removal procedure is completely automatic without having to adjust parameters for different capturing scenarios, in contrast to [Clark and Dunser, 2012], where the thresholds must be adjusted for different lighting conditions and for different cameras.

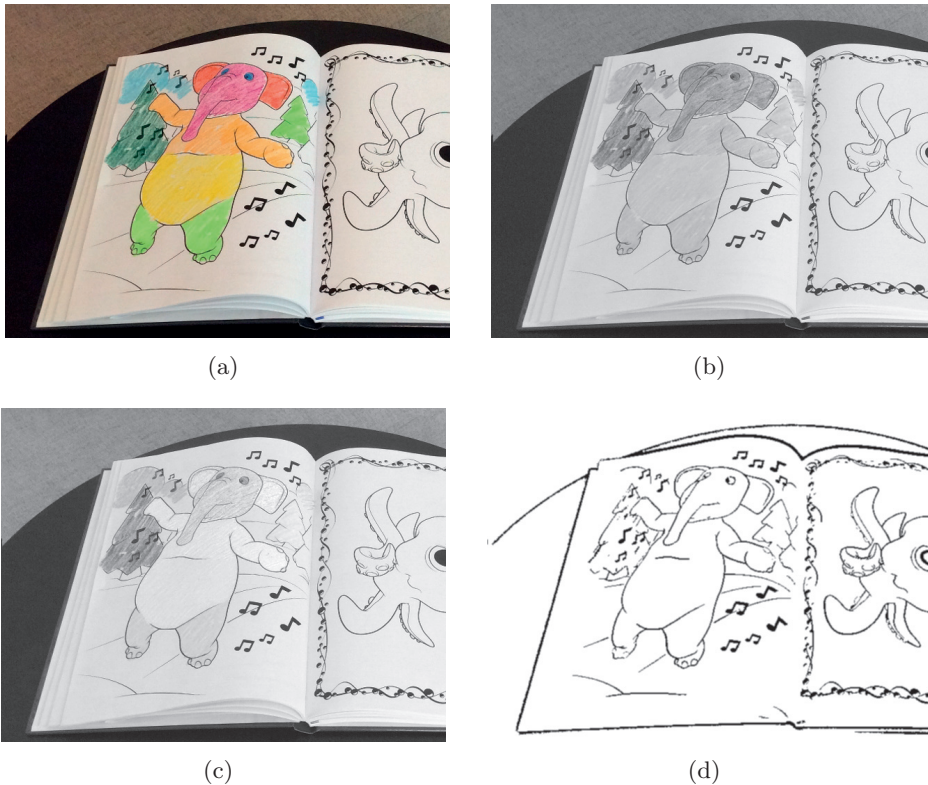


Figure 5.39: Image processing: (a) Input image of the colored drawing. (b) Gray scale input image. (c) Luminance channel in HSV color space. (d) Resulting line draw image.

5.4.3 Tracking

Through an edge shader, a 2-D drawing template for each 3-D coloring character is generated for the printing purpose. We also use these drawings as the template images with flat rectangular 3-D shapes in our template-based tracking algorithm.

We first have to detect which drawing among a set of them appears in the camera images. We use our real-time template selection algorithm presented in Section 5.2. After that, our template-based deformable object tracking and 3-D reconstruction method presented in Chapter 3 is used to track and reconstruct the shape of the drawing page. We rely on frame-to-frame tracking to gain frame rate and only apply the active search for correspondences once every $N = 5$ frames to retrieve back lost tracked points and accumulate good correspondences. This simple technique turns out to be efficient for our problem and to increase frame rate while not degrading the tracking quality. We perform frame-to-frame inlier feature points KLT tracking on the gray-scale image of the original input image. This feature point tracking is therefore robust to appearance changes caused by coloring. On the other hand, active search for correspondences using the template image as the source is done on the line-draw processed image.

5.4.4 Texture Generation

Once the 3-D shape of the colored drawing has been recovered in the view of camera, the mesh is re-projected onto the image plane. This re-projection defines a direct mapping between the pixels on the original drawing template and the pixels on the image of the colored drawing. We then can generate the texture for the character mesh using this the mapping and the lookup map L (Fig. 5.38k). In practice, we rectify the input image of the colored drawing in the canonical fronto-parallel pose by rendering a rectangular mesh with texture coordinates taken from the mesh reprojection. The color lookup is then performed on this rendered image.

5.4.5 Augmentation

To show an augmented view of the coloring book, we use the live view as the background image for the 3-D scene, and use proper parameters for the virtual camera to render the 3-D animating character on top of the drawing page with the generated texture from the drawing (Fig. 5.38 l). The world coordinate system is defined to be at the middle of the drawing page with Oz axis aligning with the surface normal at that point. The character and its animations are positioned in relative to this world coordinate system and surface deformations. The virtual camera orientation and location are also expressed in this world coordinate system. In this way, the virtual animated character appears seamlessly as a real one standing on the drawing when being seen from the augmented view.

5.4.6 Results

The interactive coloring book App is built using the Unity¹ game engine and runs on iOS. It uses Unity to access the camera through the `WebCamTexture` object, and fetches the pixels into the main memory. These are then passed to a C++ library implementing the deformable surface tracking algorithm. This library tells Unity whether a drawing template is detected and if so returns the 3-D shape of the possibly non-flat colored drawing in the camera coordinates. The App renders the camera image using a screen-aligned quad, and overlays the corresponding animated 3-D character. Some scenarios of our coloring book application are depicted in Fig. 5.40.

Fig. 5.41 demonstrates the robustness of our outlier rejection algorithm on real scenarios. The algorithm can filter 37 inlier matches out of 422 putative ones. As a result, accurate color information can be retrieved for the texture generation algorithm (see the virtual characters in Fig. 5.41).

¹<http://unity3d.com>



Figure 5.40: Coloring book application scenarios with different drawings and colorings. Images courtesy of Disney Research Zurich.

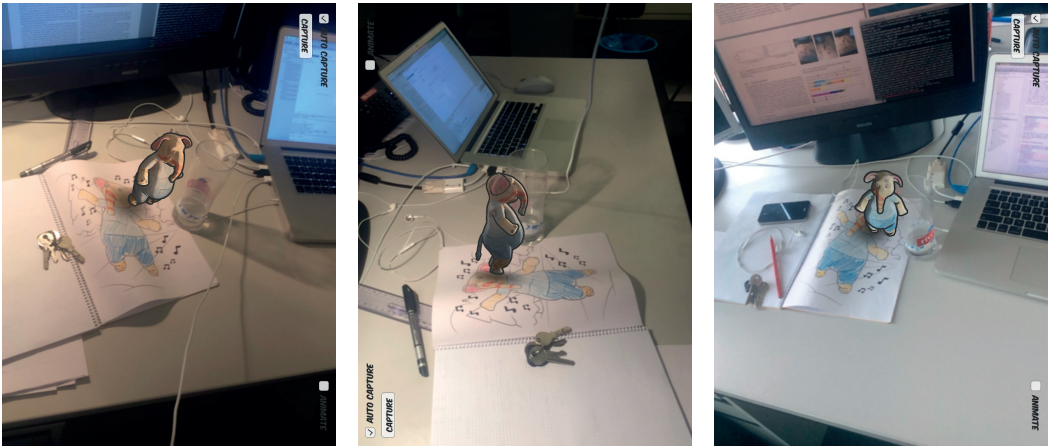


Figure 5.41: Screenshots from the App showing the robustness of our outlier rejection algorithm.

5.5 Conclusion

In this chapter, we have presented a number of real world applications that use our proposed deformable object 3-D reconstruction and tracking approaches. We first presented our method to reconstructing deformable objects interacting with rigid ones. To enforce the presence of rigid objects, we used the modified edge-based or render-based dense matching object pose estimation to compute the 6-D pose of the rigid objects. We discussed in detail our study of ball-bat collisions whose results are compared quantitatively to simulation-based methods.

We also introduced our real-time object recognition algorithm which automatically selects the template for our template-based tracking and reconstruction algorithm. By using a simple voting scheme, the algorithm obtained a high accuracy and a real-time performance.

Our efficient environment lighting estimation modeled by spherical harmonics was proposed for augmented reality purposes. Its recovered irradiance maps compared favorably with existing methods while being faster and more scalable.

Finally, we introduced an Augmented Reality coloring book App that runs on an iPad. It detects and tracks the drawing, and then augments the video stream with an animated 3-D version of the character that is textured according to the child's coloring. Our tracking algorithm has been used in various projects at Disney.

6 Concluding Remarks

This thesis has presented our algorithms to recover the 3-D shape of a deformable object in single images, or image sequences acquired by one or more cameras, given that a 3-D template shape and a template image of the object are available. We have also presented a number of real-world applications that exploit our template-based deformable object 3-D reconstruction and tracking algorithms.

6.1 Summary

In Chapter 3, we presented our linear formulation of the template-based monocular 3-D shape recovery of a deformable object using a Laplacian approach to regularizing the solution and parameterizing the mesh with respect to a small set of control points. We showed that we could turn the monocular 3-D shape reconstruction into a much better-posed problem with far fewer degrees of freedom than the original one formulated in terms of all mesh vertices. This has proved key to achieving real-time performance while preserving both sufficient flexibility and robustness. Our real-time 3-D reconstruction and tracking system of deformable objects can very quickly reject outlier correspondences and accurately reconstruct the object shape in 3D. The frame-to-frame tracking was extensively exploited to track the object under difficult settings such as large deformations, occlusions, illumination changes, and motion blur.

Texture-based methods, such as the one discussed above, are partially effective in surface reconstruction tasks for well-textured surfaces, which give rise to many interest points. However, they are ill-equipped to handle sparsely-textured ones. In Chapter 4, we therefore introduced an approach to solving the problem of dense image registration and 3-D shape reconstruction of deformable objects in the presence of occlusions and minimal texture. We developed the pixel-wise relevancy scores that are used to weigh the influence of the pixel image information in the image energy cost function based on how informative and reliable that pixel is. A careful design of the framework was essential

for obtaining state-of-the-art results in recovering 3-D deformations of both well- and poorly-textured objects in the presence of occlusions.

In Chapter 5, we studied the problem of reconstructing 3-D deformable objects as they interact with rigid objects. Imposing real physical constraints allowed us to model the interactions of objects in the real world more accurately and more realistically. In particular, we studied the problem of a ball colliding with a bat observed by high speed cameras. We provided quantitative measurements of the impact that were compared with simulation-based methods to evaluate which simulation predictions most accurately describe a physical quantity of interest and to improve the models.

We also proposed an efficient method to estimate the environment irradiance map represented by a set of low frequency spherical harmonics. The obtained irradiance map could be used to realistically illuminate 2-D and 3-D virtual contents in the context of augmented reality on deformable objects. The results compared favorably with the baseline methods.

In collaboration with Disney Research, we developed an augmented reality coloring book application that runs in real-time on mobile devices. The app allows children (or anybody) to view their illustrations as animated characters with texture lifted from their colors on the drawing. Deformations of the book page were explicitly modeled by our 3-D tracking and reconstruction method. As a result, accurate color information was extracted to synthesize the character's texture.

6.2 Future Work

We believe we have advanced the state-of-the-art in video-based deformable 3-D shape recovery. However, our algorithms are not without limitations. A number of research directions could be pursued to further improve them.

In Chapter 3, our linear parameterization of a triangle mesh with respect to a few control point yields a Jacobian of the energy function that is dense. This is because the coordinates of a mesh vertex are expressed with respect to all control vertices. As a consequence, we lose the sparsity property of the reconstruction problem. One possible research direction is to impose the sparsity constraints in the linear parameterization matrix by enforcing local influences of control points to only their nearby vertices. It would help reduce the dimensionality of the problem while preserving sparsity.

There is also space for improving the method we presented in Chapter 4. An important image cue that we have neglected in this work is shading. When the amount of image information is small, shading cues would significantly help to resolve ambiguities. In combination with our dense-matching energy, shading information should help to better constrain the reconstruction problem.

Our methods in Chapter 4 do not handle self-occlusions and external occlusion at the same time. The way we compute the relevancy scores discards self-occluded pixels as well as visible ones near occlusion boundaries. Since these visible pixels convey important shape information, this would reduce accuracy. A next step would be to include a Z-buffer in the computation of the relevancy scores to handle both external and self-occlusions simultaneously.

More difficult scenarios of 3-D modeling of deformable objects were not addressed in this thesis. As a future work, we would like to track a turning book page or a folding flag. Solving these problems would truly enlarge the application scope of deformable object modeling.

Bibliography

- [Aittala, 2010] Aittala, M. (2010). Inverse lighting and photorealistic rendering for augmented reality. *The Visual Computer*, 26(6-8):669–678.
- [Akhter et al., 2008] Akhter, I., Sheikh, Y., Khan, S., and Kanade, T. (2008). Nonrigid Structure from Motion in Trajectory Space. In *Advances in Neural Information Processing Systems*.
- [Alahi et al., 2012] Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). FREAK: Fast Retina Keypoint. In *Conference on Computer Vision and Pattern Recognition*.
- [Arya et al., 2007] Arya, K., Gupta, P., Kalra, P., and Mitra, P. (2007). Image Registration Using Robust M-Estimators. *Pattern Recognition*, 28(15):1957–1968.
- [Ballard, 1981] Ballard, D. H. (1981). Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2):111–122.
- [Bartoli et al., 2008] Bartoli, A., Gay-bellile, V., Castellani, U., Peyras, J., Olsen, S., and Sayd, P. (2008). Coarse-To-Fine Low-Rank Structure-From-Motion. In *Conference on Computer Vision and Pattern Recognition*.
- [Bartoli et al., 2012] Bartoli, A., Gérard, Y., Chadebecq, F., and Collins, T. (2012). On Template-Based Reconstruction from a Single View: Analytical Solutions and Proofs of Well-Posedness for Developable, Isometric and Conformal Surfaces. In *Conference on Computer Vision and Pattern Recognition*.
- [Bay et al., 2006] Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded Up Robust Features. In *European Conference on Computer Vision*.
- [Bergen, 1997] Bergen, G. v. d. (1997). Efficient collision detection of complex deformable models using aabb trees. *Journal of Graphics Tools*, 2(4):1–13.
- [Blanz and Vetter, 1999] Blanz, V. and Vetter, T. (1999). A Morphable Model for the Synthesis of 3D Faces. In *ACM SIGGRAPH*, pages 187–194.
- [Bookstein, 1989] Bookstein, F. (1989). Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585.

Bibliography

- [Botsch et al., 2006] Botsch, M., Sumner, R., Pauly, M., and Gross, M. (2006). Deformation Transfer for Detail-Preserving Surface Editing. In *Vision Modeling and Visualization*, pages 357–364.
- [Bouguet, 1999] Bouguet, J. (1999). Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the Algorithm. Technical report, Intel Microprocessor Research Labs.
- [Braux-Zin et al., 2013] Braux-Zin, J., Dupont, R., and Bartoli, A. (2013). A general dense image matching framework combining direct and feature-based costs. In *International Conference on Computer Vision*, pages 185–192.
- [Bregler et al., 2000] Bregler, C., Hertzmann, A., and Biermann, H. (2000). Recovering Non-Rigid 3D Shape from Image Streams. In *Conference on Computer Vision and Pattern Recognition*.
- [Bronte et al., 2014] Bronte, S., Paladini, M., Bergasa, L., Agapito, L., and Arroyo, R. (2014). Real-Time Sequential Model-Based Non-Rigid SFM. In *International Conference on Intelligent Robots and Systems*, pages 1026–1031.
- [Brunet et al., 2014] Brunet, F., Bartoli, A., and Hartley, R. (2014). Monocular Template-Based 3D Surface Reconstruction: Convex Inextensible and Nonconvex Isometric Methods. *Computer Vision and Image Understanding*, 125:138–154.
- [Brunet et al., 2010] Brunet, F., Hartley, R., Bartoli, A., Navab, N., and Malgouyres, R. (2010). Monocular Template-Based Reconstruction of Smooth and Inextensible Surfaces. In *Asian Conference on Computer Vision*.
- [Bryson and Smith, 2010] Bryson, A. and Smith, L. (2010). Impact response of sports materials. *Procedia Engineering*, 2(2):2961–2966.
- [Burbank and Smith, 2012] Burbank, S. and Smith, L. (2012). Dynamic characterization of rigid polyurethane foam used in sports balls. *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*.
- [Calonder et al., 2010] Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *European Conference on Computer Vision*.
- [Canny, 1986] Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6).
- [Cashman and Fitzgibbon, 2013] Cashman, T. J. and Fitzgibbon, A. W. (2013). What shape are dolphins? Building 3D morphable models from 2D Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):232–244.

- [Chhatkuli et al., 2014] Chhatkuli, A., Pizarro, D., and Bartoli, A. (2014). Stable Template-Based Isometric 3D Reconstruction in All Imaging Conditions by Linear Least-Squares. In *Conference on Computer Vision and Pattern Recognition*.
- [Clark and Dunser, 2012] Clark, A. and Dunser, A. (2012). An Interactive Augmented Reality Coloring Book. In *IEEE Symposium on 3D User Interfaces*, pages 7–10.
- [Collins and Bartoli, 2014] Collins, T. and Bartoli, A. (2014). Using Isometry to Classify Correct/incorrect 3D-2D Correspondences. In *European Conference on Computer Vision*, pages 325–340.
- [Collins and Bartoli, 2015] Collins, T. and Bartoli, A. (2015). Realtime Shape-from-Template: System and Applications. In *International Symposium on Mixed and Augmented Reality*, pages 116–119.
- [Cootes et al., 1998] Cootes, T., Edwards, G., and Taylor, C. (1998). Active Appearance Models. In *European Conference on Computer Vision*, pages 484–498.
- [Crivellaro and Lepetit, 2014] Crivellaro, A. and Lepetit, V. (2014). Robust 3D Tracking with Descriptor Fields. In *Conference on Computer Vision and Pattern Recognition*.
- [Crivellaro et al., 2015] Crivellaro, A., Rad, M., Verdie, Y., Yi, K., Fua, P., and Lepetit, V. (2015). A Novel Representation of Parts for Accurate 3D Object Detection and Tracking in Monocular Images. In *International Conference on Computer Vision*.
- [Dai et al., 2012] Dai, Y., Li, H., and He, M. (2012). A Simple Prior-Free Method for Non-Rigid Structure from Motion Factorization. In *Conference on Computer Vision and Pattern Recognition*.
- [Dame and Marchand, 2012] Dame, A. and Marchand, E. (2012). Second-Order Optimization of Mutual Information for Real-Time Image Registration. *IEEE Transactions on Image Processing*, 21(9):4190–4203.
- [Damen et al., 2012] Damen, D., Bunnun, P., Calway, A., and Mayol-cuevas, W. (2012). Real-Time Learning and Detection of 3D Texture-Less Objects: A Scalable Approach. In *British Machine Vision Conference*.
- [Debevec, 1998] Debevec, P. (1998). Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography. In *ACM SIGGRAPH*.
- [Delbue, 2008] Delbue, A. (2008). A Factorization Approach to Structure from Motion with Shape Priors. In *Conference on Computer Vision and Pattern Recognition*.
- [Delbue and Bartoli, 2011] Delbue, A. and Bartoli, A. (2011). Multiview 3D Warps. In *International Conference on Computer Vision*.

Bibliography

- [Delbue et al., 2010] Delbue, A., Xavier, J., Agapito, L., and Paladini, M. (2010). Bilinear Factorization via Augmented Lagrange Multipliers. In *European Conference on Computer Vision*.
- [Dimitrijević et al., 2004] Dimitrijević, M., Ilić, S., and Fua, P. (2004). Accurate Face Models from Uncalibrated and Ill-Lit Video Sequences. In *Conference on Computer Vision and Pattern Recognition*.
- [Dowson and Bowden, 2006] Dowson, N. and Bowden, R. (2006). A Unifying Framework for Mutual Information Methods for Use in Non-Linear Optimisation. In *European Conference on Computer Vision*, pages 365–378.
- [Ecker et al., 2008] Ecker, A., Jepson, A., and Kutulakos, K. (2008). Semidefinite Programming Heuristics for Surface Reconstruction Ambiguities. In *European Conference on Computer Vision*.
- [Elseberg et al., 2012] Elseberg, J., Magnenat, S., Siegwart, R., and Nüchter, A. (2012). Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics*, 3(1):2–12.
- [Fassold et al., 2009] Fassold, H., Rosner, J., Schallauer, P., and Bailer, W. (2009). Realtime klt feature point tracking for high definition video. *GraVisMa*.
- [Fayad et al., 2010] Fayad, J., Agapito, L., and Delbue, A. (2010). Piecewise Quadratic Reconstruction of Non-Rigid Surfaces from Monocular Sequences. In *European Conference on Computer Vision*.
- [Fua et al., 2010] Fua, P., Varol, A., Urtasun, R., and Salzmann, M. (2010). Least-Squares Minimization Under Constraints. Technical report, EPFL.
- [Furukawa and Ponce, 2009] Furukawa, Y. and Ponce, J. (2009). Accurate, Dense, and Robust Multi-View Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99.
- [Garg et al., 2013a] Garg, R., Roussos, A., and Agapito, L. (2013a). Dense Variational Reconstruction of Non-Rigid Surfaces from Monocular Video. In *Conference on Computer Vision and Pattern Recognition*.
- [Garg et al., 2013b] Garg, R., Roussos, A., and Agapito, L. (2013b). A variational approach to video registration with subspace constraints. *International Journal of Computer Vision*, 104(3):286–314.
- [Gay-Bellile et al., 2010] Gay-Bellile, V., Bartoli, A., and Sayd, P. (2010). Direct Estimation of Nonrigid Registrations with Image-Based Self-Occlusion Reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):87–104.

-
- [Gopalan and Jacobs, 2010] Gopalan, R. and Jacobs, D. (2010). Comparing and Combining Lighting Insensitive Approaches for Face Recognition. *Computer Vision and Image Understanding*, 114(1):135–145.
- [Gruber et al., 2012] Gruber, L., Richter-Trummer, T., and Schmalstieg, D. (2012). Real-time photometric registration from arbitrary geometry. In *International Symposium on Mixed and Augmented Reality*, pages 119–128.
- [Gumerov et al., 2004] Gumerov, N., Zandifar, A., Duraiswami, R., and Davis, L. (2004). Structure of Applicable Surfaces from Single Views. In *European Conference on Computer Vision*.
- [Gurobi, 2012] Gurobi (2012). Gurobi Optimizer. <http://www.gurobi.com/>.
- [Hager and Belhumeur, 1998] Hager, G. and Belhumeur, P. (1998). Efficient Region Tracking with Parametric Models of Geometry and Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039.
- [Harris and Stennett, 1990] Harris, C. and Stennett, C. (1990). RAPID-a Video Rate Object Tracker. In *British Machine Vision Conference*.
- [Hinterstoisser et al., 2012] Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., and Lepetit, V. (2012). Gradient Response Maps for Real-Time Detection of Textureless Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):876–888.
- [Hoaglin et al., 1983] Hoaglin, D., Mosteller, F., and Tukey, J. (1983). *Understanding Robust and Exploratory Data Analysis*. Wiley New York.
- [Huber, 1981] Huber, P. (1981). *Robust Statistics*. Wiley.
- [Ilić and Fua, 2002] Ilić, S. and Fua, P. (2002). Using Dirichlet Free Form Deformation to Fit Deformable Models to Noisy 3D Data. In *European Conference on Computer Vision*.
- [Jacobson et al., 2011] Jacobson, A., Baran, I., Popovic, J., and Sorkine, O. (2011). Bounded Biharmonic Weights for Real-Time Deformation. In *ACM SIGGRAPH*.
- [Kanbara and Yokoya, 2004] Kanbara, M. and Yokoya, N. (2004). Real-Time Estimation of Light Source Environment for Photorealistic Augmented Reality. In *International Conference on Pattern Recognition*.
- [Khoshelham and Elberink, 2012] Khoshelham, K. and Elberink, S. O. (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454.
- [Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel Tracking and Mapping for Small AR Workspaces. In *International Symposium on Mixed and Augmented Reality*.

Bibliography

- [Kneip et al., 2014] Kneip, L., Li, H., and Seo, Y. (2014). UPnP: An Optimal $O(n)$ Solution to the Absolute Pose Problem with Universal Applicability. In *European Conference on Computer Vision*.
- [Knorr and Kurz, 2014] Knorr, S. B. and Kurz, D. (2014). Real-time illumination estimation from faces for coherent rendering. In *International Symposium on Mixed and Augmented Reality*, pages 113–122.
- [Lepetit et al., 2009] Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). EPnP: An Accurate $O(n)$ Solution to the PnP Problem. *International Journal of Computer Vision*, 81(2).
- [Leutenegger et al., 2011] Leutenegger, S., Chli, M., and Siegwart, R. (2011). BRISK: Binary Robust Invariant Scalable Keypoints. In *International Conference on Computer Vision*.
- [Lewis, 1995] Lewis, J. (1995). Fast Normalized Cross-Correlation. In *Vision Interface*, pages 120–123.
- [Lowe, 2004] Lowe, D. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 20(2).
- [LS-DYNA, 2015] LS-DYNA (2015). LS-DYNA: An advanced general-purpose multiphysics simulation software package. <http://www.lstc.com/products/ls-dyna/>.
- [Lucas and Kanade, 1981] Lucas, B. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679.
- [Maier-Hein et al., 2013] Maier-Hein, L., Mountney, P., Bartoli, A., Elhawary, H., Elson, D., Groch, A., Kolb, A., Rodrigues, M., Sorger, J., Speidel, S., et al. (2013). Optical techniques for 3D surface reconstruction in computer-assisted laparoscopic surgery. *Medical Image Analysis*, 17(8):974–996.
- [Malti et al., 2011] Malti, A., Bartoli, A., and Collins, T. (2011). A Pixel-Based Approach to Template-Based Monocular 3D Reconstruction of Deformable Surfaces. In *International Conference on Computer Vision Workshops*.
- [Malti et al., 2013] Malti, A., Hartley, R., Bartoli, A., and Kim, J.-H. (2013). Monocular Template-Based 3D Reconstruction of Extensible Surfaces with Local Linear Elasticity. In *Conference on Computer Vision and Pattern Recognition*.
- [Metaio, 2015] Metaio (2015). Metaio SDK for Developers. <https://www.metaio.com/>.
- [Moreno-Noguer et al., 2010] Moreno-Noguer, F., Porta, J., and Fua, P. (2010). Exploring Ambiguities for Monocular Non-Rigid Shape Estimation. In *European Conference on Computer Vision*.

-
- [Moreno-Noguer et al., 2009] Moreno-Noguer, F., Salzmann, M., Lepetit, V., and Fua, P. (2009). Capturing 3D Stretchable Surfaces from Single Images in Closed Form. In *Conference on Computer Vision and Pattern Recognition*.
- [Nathan et al., 2011] Nathan, A. M., Smith, L. V., and Faber, W. (2011). Reducing the effect of the ball on bat performance measurements. *Sports Technology*, 4(1-2):13–18.
- [Newcombe et al., 2011] Newcombe, R., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *International Symposium on Mixed and Augmented Reality*.
- [Ngo et al., 2015] Ngo, D., Ostlund, J., and Fua, P. (2015). Template-Based Monocular 3D Shape Recovery Using Laplacian Meshes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Ostlund et al., 2012] Ostlund, J., Varol, A., Ngo, D., and Fua, P. (2012). Laplacian Meshes for Monocular 3D Shape Recovery. In *European Conference on Computer Vision*.
- [Ozuysal et al., 2010] Ozuysal, M., Calonder, M., Lepetit, V., and Fua, P. (2010). Fast Keypoint Recognition Using Random Ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):448–461.
- [Paladini et al., 2009] Paladini, M., Del Bue, A., Dodig, S., Xavier, J., and Agapito, L. (2009). Factorization for Non-Rigid and Articulated Structure Using Metric Projections. In *Conference on Computer Vision and Pattern Recognition*.
- [Panin and Knoll, 2008] Panin, G. and Knoll, A. (2008). Mutual Information-Based 3D Object Tracking. *International Journal of Computer Vision*, 78(1):107–118.
- [Parashar et al., 2015] Parashar, S., Pizarro, D., Bartoli, A., and Collins, T. (2015). As-Rigid-As-Possible Volumetric Shape-From-Template. In *International Conference on Computer Vision*, pages 891–899.
- [Perriollat and Bartoli, 2007] Perriollat, M. and Bartoli, A. (2007). A Quasi-Minimal Model for Paper-Like Surfaces. In *Conference on Computer Vision and Pattern Recognition*.
- [Perriollat and Bartoli, 2013] Perriollat, M. and Bartoli, A. (2013). A Computational Model of Bounded Developable Surfaces with Application to Image-Based Three-Dimensional Reconstruction. *Computer Animation and Virtual Worlds*, 24(5).
- [Perriollat et al., 2011] Perriollat, M., Hartley, R., and Bartoli, A. (2011). Monocular Template-Based Reconstruction of Inextensible Surfaces. *International Journal of Computer Vision*, 95.
- [Pilet et al., 2006] Pilet, J., Geiger, A., Lagger, P., Lepetit, V., and Fua, P. (2006). An All-In-One Solution to Geometric and Photometric Calibration. In *International Symposium on Mixed and Augmented Reality*.

Bibliography

- [Pilet et al., 2007] Pilet, J., Lepetit, V., and Fua, P. (2007). Retexturing in the Presence of Complex Illuminations and Occlusions. In *International Symposium on Mixed and Augmented Reality*.
- [Pilet et al., 2008] Pilet, J., Lepetit, V., and Fua, P. (2008). Fast Non-Rigid Surface Detection, Registration and Realistic Augmentation. *International Journal of Computer Vision*, 76(2):109–122.
- [Pizarro and Bartoli, 2012] Pizarro, D. and Bartoli, A. (2012). Feature-Based Deformable Surface Detection with Self-Occlusion Reasoning. *International Journal of Computer Vision*.
- [Ramamoorthi and Hanrahan, 2001] Ramamoorthi, R. and Hanrahan, P. (2001). An Efficient Representation for Irradiance Environment Maps. In *ACM SIGGRAPH*.
- [Reitinger et al., 2006] Reitinger, B., Bornik, A., Beichel, R., and Schmalstieg, D. (2006). Liver Surgery Planning Using Virtual Reality. *IEEE Computer Graphics and Applications*, 26(6):36–47.
- [Rublee et al., 2011] Rublee, E., Rabaud, V., Konolidge, K., and Bradski, G. (2011). ORB: An Efficient Alternative to SIFT or SURF. In *International Conference on Computer Vision*.
- [Salzmann, 2013] Salzmann, M. (2013). Continuous Inference in Graphical Models with Polynomial Energies. In *Conference on Computer Vision and Pattern Recognition*.
- [Salzmann and Fua, 2010] Salzmann, M. and Fua, P. (2010). *Deformable Surface 3D Reconstruction from Monocular Images*. Morgan-Claypool.
- [Salzmann and Fua, 2011] Salzmann, M. and Fua, P. (2011). Linear Local Models for Monocular Reconstruction of Deformable Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):931–944.
- [Salzmann et al., 2007a] Salzmann, M., Hartley, R., and Fua, P. (2007a). Convex Optimization for Deformable Surface 3D Tracking. In *International Conference on Computer Vision*.
- [Salzmann et al., 2007b] Salzmann, M., Lepetit, V., and Fua, P. (2007b). Deformable Surface Tracking Ambiguities. In *Conference on Computer Vision and Pattern Recognition*.
- [Salzmann et al., 2008a] Salzmann, M., Moreno-Noguer, F., Lepetit, V., and Fua, P. (2008a). Closed-Form Solution to Non-Rigid 3D Surface Registration. In *European Conference on Computer Vision*.
- [Salzmann and Urtasun, 2012] Salzmann, M. and Urtasun, R. (2012). Beyond Feature Points: Structured Prediction for Monocular Non-Rigid 3D Reconstruction. In *European Conference on Computer Vision*.

-
- [Salzmann et al., 2008b] Salzmann, M., Urtasun, R., and Fua, P. (2008b). Local Deformation Models for Monocular 3D Shape Recovery. In *Conference on Computer Vision and Pattern Recognition*.
- [Sato et al., 1999] Sato, I., Sato, Y., and Ikeuchi, K. (1999). Acquiring a Radiance Distribution to Superimpose Virtual Objects Onto a Real Scene. *IEEE Transactions on Visualization and Computer Graphics*.
- [Scandaroli et al., 2012] Scandaroli, G., Meilland, M., and Richa, R. (2012). Improving NCC-Based Direct Visual Tracking. In *European Conference on Computer Vision*.
- [Shahrokhni et al., 2004] Shahrokhni, A., Drummond, T., and Fua, P. (2004). Texture Boundary Detection for Real-Time Tracking. In *European Conference on Computer Vision*, pages 566–577.
- [Shahrokhni et al., 2005] Shahrokhni, A., Drummond, T., and Fua, P. (2005). Fast Texture-Based Tracking and Delineation Using Texture Entropy. In *International Conference on Computer Vision*.
- [Shaji et al., 2010] Shaji, A., Varol, A., Torresani, L., and Fua, P. (2010). Simultaneous Point Matching and 3D Deformable Surface Reconstruction. In *Conference on Computer Vision and Pattern Recognition*.
- [Shen et al., 2009] Shen, S., Shi, W., and Liu, Y. (2009). Monocular 3D Tracking of Inextensible Deformable Surfaces Under L2-Norm. In *Asian Conference on Computer Vision*.
- [Silveira and Malis, 2007] Silveira, G. and Malis, E. (2007). Real-Time Visual Tracking Under Arbitrary Illumination Changes. In *Conference on Computer Vision and Pattern Recognition*.
- [Smith et al., 2015] Smith, L., Nevins, D., Ngo, D. T., and Fua, P. (2015). Measuring the accuracy of solid sport ball impact simulations. *Journal of Sports Engineering*. Submitted for publication.
- [Smith and Duris, 2009] Smith, L. V. and Duris, J. G. (2009). Progress and challenges in numerically modelling solid sports balls with application to softballs. *Journal of sports sciences*, 27(4):353–360.
- [Sorkine et al., 2004] Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., and Seidel, H.-P. (2004). Laplacian Surface Editing. In *Symposium on Geometry Processing*, pages 175–184.
- [Strecha et al., 2006] Strecha, C., Fransens, R., and Van Gool, L. (2006). Combined Depth and Outlier Estimation in Multi-View Stereo. In *Conference on Computer Vision and Pattern Recognition*.
- [Sumner and Popovic, 2004] Sumner, R. and Popovic, J. (2004). Deformation Transfer for Triangle Meshes. *ACM Transactions on Graphics*, pages 399–405.

Bibliography

- [Taddei and Bartoli, 2008] Taddei, P. and Bartoli, A. (2008). Template-based paper reconstruction from a single image is well posed when the rulings are parallel. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–6. IEEE.
- [Taylor et al., 2010] Taylor, J., Jepson, A. D., and Kutulakos, K. N. (2010). Non-Rigid Structure from Locally-Rigid Motion. In *Conference on Computer Vision and Pattern Recognition*.
- [Tomasi and Kanade, 1992] Tomasi, C. and Kanade, T. (1992). Shape and Motion from Image Streams Under Orthography: A Factorization Method. *International Journal of Computer Vision*, 9(2):137–154.
- [Torresani et al., 2008] Torresani, L., Hertzmann, A., and Bregler, C. (2008). Nonrigid Structure-From-Motion: Estimating Shape and Motion with Hierarchical Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):878–892.
- [Tran et al., 2012] Tran, Q.-H., Chin, T.-J., Carneiro, G., Brown, M. S., and Suter, D. (2012). In Defence of RANSAC for Outlier Rejection in Deformable Registration. In *European Conference on Computer Vision*, pages 274–287. Springer.
- [Varol et al., 2012] Varol, A., Salzmann, M., Fua, P., and Urtasun, R. (2012). A Constrained Latent Variable Model. In *Conference on Computer Vision and Pattern Recognition*.
- [Vicente and Agapito, 2012] Vicente, S. and Agapito, L. (2012). Soft Inextensibility Constraints for Template-Free Non-Rigid Reconstruction. In *European Conference on Computer Vision*.
- [Viola and Wells, 1997] Viola, P. and Wells, W. (1997). Alignment by Maximization of Mutual Information. *International Journal of Computer Vision*, 24(2):134–154.
- [Vuforia, 2015] Vuforia (2015). Vuforia Developer. <https://developer.vuforia.com/>.
- [Wagner et al., 2008] Wagner, D., Reitmayr, G., Mulloni, A., Drummond, T., and Schmalstieg, D. (2008). Pose Tracking from Natural Features on Mobile Phones. In *International Symposium on Mixed and Augmented Reality*.
- [Wang et al., 2009] Wang, X., Han, T., and Yan, S. (2009). An HoG-LBP Human Detector with Partial Occlusion Handling. In *International Conference on Computer Vision*.
- [White and Forsyth, 2006] White, R. and Forsyth, D. (2006). Retexturing Single Views Using Texture and Shading. In *European Conference on Computer Vision*, pages 70–81.
- [Yu et al., 2015] Yu, R., Russell, C., Campbell, N., and Agapito, L. (2015). Direct, Dense, and Deformable: Template-Based Non-Rigid 3D Reconstruction from RGB Video. In *International Conference on Computer Vision*.

- [Zhou et al., 2009] Zhou, Z., Wagner, A., Mobahi, H., Wright, J., and Ma, Y. (2009). Face Recognition with Contiguous Occlusion Using Markov Random Fields. In *International Conference on Computer Vision*.
- [Zhu and Lyu, 2007] Zhu, J. and Lyu, M. (2007). Progressive Finit Newton Approach to Real-Time Nonrigid Surface Detection. In *International Conference on Computer Vision*.

Dat NGO

EPFL IC ISIM CVLAB, BC 301, Station 14, CH-1015 Lausanne, Switzerland
(+41) 21 69 38161 | dat.ngo@epfl.ch
<http://people.epfl.ch/tdngo>



EDUCATION

- **Ph. D. in Computer Science** 09/2011 - 03/2016
Computer Vision Laboratory
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
- **M. Sc. in Artificial Intelligence** 09/2009 - 08/2011
VU University Amsterdam, Netherlands.
- **B. Sc. in Information Technology** 09/2004 - 07/2008
Vietnam National University Hanoi.

PROFESSIONAL EXPERIENCE

- **Computer Vision Laboratory, EPFL, Switzerland, Research Assistant** 09/2011 - 03/2016
Designed and implemented various algorithms in computer vision: object recognition, object pose tracking, shape reconstruction, image registration
- **Disney Research Zurich, Switzerland, Software Engineer Internship** 08/2014 - 10/2014
Developed the first real-time highly optimized 3-D deformable surface tracking system on mobile devices for an augmented reality children coloring book application
Pending US patent: *Deformable-Surface Tracking based Augmented Reality Image Generation*
- **School of Computer and Communication Sciences, EPFL, Switzerland, Teaching Assistant** 09/2011 - 03/2016
Courses taught: CS-445 Foundation of Imaging Science, MATH-111 Linear Algebra
- **Florida State University, USA, Research Assistant** 02/2011 - 08/2011
Developed a computer-aided diagnosis system for non-mass breast tumor classification
- **Vietnam National University Hanoi, Lecturer** 06/2008 - 08/2009
Course taught: C++ programming

PUBLICATIONS

- Lloyd Smith, Derek Nevins, **Dat Tien Ngo**, Pascal Fua. Measuring the accuracy of solid sport ball impact simulations. Submitted to Journal of Sports Engineering, 2015.
- **Dat Tien Ngo**, Sanghyuk Park, Anne Jorstad, Alberto Crivellaro, Chang Yoo, Pascal Fua. Dense image registration and deformable surface reconstruction in presence of occlusions and minimal texture. International Conference on Computer Vision (ICCV), 2015, Santiago, Chile.
- S. Magnenat, **T.D. Ngo***, F. Zund, M. Ryffel, G. Noris, G. Rothlin, A. Marra, M. Nitti, P. Fua, M. Gross, R. Sumner*. Live Texturing of Augmented Reality Characters from Colored Drawings. IEEE Transactions on Visualization and Computer Graphics (TVCG), 2015. * *Corresponding authors. Best Paper Award Honorable Mention.*
- **Dat Tien Ngo**, Jonas Ostlund, Pascal Fua. Template-based Monocular 3D Shape Recovery using Laplacian Meshes. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2015.
- Jonas Ostlund, Aydin Varol, **Dat Tien Ngo**, Pascal Fua. Laplacian meshes for monocular 3D shape recovery. European Conference on Computer Vision (ECCV), 2012, Florence, Italy.

- **Dat Ngo**, Olmo Zavala, Jamie Shutler, Mark Lobbes, Maribel Lockwood, Anke Meyer-Base. Spatio-temporal feature extraction for differentiation of non-mass-enhancing lesions in breast MRI. SPIE Independent Component Analyses, Compressive Sampling, Wavelets, Neural Net, 2012.
- Claudia Plant, **Dat Ngo**, Felix Retter, Olmo Zavala, Thomas Schlossbauer, Marc Lobbes, Maribel Lockwood, Anke Meyer-Base. Computer-aided diagnosis of small lesions and non-masses in breast MRI. SPIE Smart Biomedical and Physiological Sensor Technology, 2012.
- Le Anh Cuong, **Ngo Tien Dat**, Nguyen Viet Ha. Isolated Handwritten Vietnamese Character Recognition with Feature Extraction and Classifier Combination. VNU Scientific Journal, 26 (3), 123-139, 2010.
- Bui The Duy, Nguyen Duy Khuong, **Ngo Tien Dat**. Supervising an unsupervised neural network. IEEE Asian Conference on Intelligent Information and Database Systems (ACIIDS), 2009.

PROFESSIONAL SKILLS

- Technical skills: Computer Vision, Numerical Optimization, Machine Learning, Pattern Recognition, Image Processing, Augmented Reality, Math-involved Algorithms, Software Design
- Programming skills: Extensive experience with C/C++, Matlab. Prior experience with Visual C++/C#, Java, PHP, SQL, JavaScript
- Language skills: Vietnamese (native), English (full working proficiency), French (basic)

SELECTED HONORS AND AWARDS

- Winner of €10,000 award presented by Qualcomm Innovation Fellowship, Austria 05/2014
- Merit EDIC fellowship for PhD study, École Polytechnique Fédérale de Lausanne, Switzerland 09/2011
- Merit Huygens scholarship for Master study, Dutch Government, 2009-2011 09/2009 - 08/2011
- Certificate of merit for outstanding student ranking first in the university graduation, People's Committee of Hanoi City, Vietnam 08/2008
- Certificate of merit for outstanding student coming first in the National University Entrance Examination (absolute mark 10/10 each subject: Mathematics, Physics, Chemistry), Vietnam National University Hanoi 09/2004
- Third Prize in "19th Student's Scientific Research Contest", Vietnam Ministry of Education and Training 03/2008
- First Prize in "Student's Scientific Research Contest", Vietnam National University Hanoi 02/2008
- Merit scholarship for outstanding students in Information Technology, Vietnam Ministry of Information and Telecommunication and Motorola Corporation. 01/2008
- Third Prize in "FIRA Simulated Soccer Robot Contest", Vietnam National University Hanoi 05/2006
- Various Prizes of Award in Mathematics and Physics contests for high school pupils

OTHER ACTIVITIES

- Member of Organization Board of National Informatics Contest for non-specialized High School Pupils, Hanoi, Vietnam 07/2009
- Founding member of Supporting Group (giving talks, helping freshmen on academic issues), Vietnam National University Hanoi 08/2007 - 08/2008
- Green belt member of Vietnam National University Hanoi Karatedo Martial art Club 09/2006 - 07/2009

