UFR DE MÉDECINE
ET DES PROFESSIONS PARAMÉDICALES
Université Clermont Auvergne

2018

ECOLE DOCTORALE
DES SCIENCES POUR L'INGENIEUR

THÈSE

Présentée à l'Université Clermont Auvergne
pour l'obtention du grade de **Docteur**
(Décret du 5 juillet 1984)

Specialité
VISION PAR ORDINATEUR

Soutenue le
20 Septembre 2018

Mathias Gallardo

Contributions to Monocular Deformable 3D Reconstruction: Curvilinear Objects and Multiple Visual Cues

---

Rapporteur, Président du jury	Vincent Lepetit, Professeur, Université de Bordeaux
Rapporteur	Peter Sturm, Directeur de Recherche, INRIA Rhône-Alpes
Examinatrice	Sylvie Chambon, Maître de Conférences, ENSEEIHT Toulouse
Directeur de thèse	Adrien Bartoli, Professeur, Université Clermont Auvergne
Co-encadrant	Toby Collins, Directeur de Recherche, IRCAD, Strasbourg

EnCoV Institut Pascal
EnCoV, Institut Pascal, UMR 6602 CNRS,
Université Clermont Auvergne, SIGMA
Faculté de Médecine
28 place Henri Dunant, Clermont-Ferrand
Tel: +33 4 73 17 81 23

To my family, Sarah and her family.

S.D.G

Acknowledgements

I would like to express my deepest and sincerest gratitude to my supervisors, Toby Collins and Adrien Bartoli, for giving me this opportunity and for their guidance, inspiration, ingenuity, kindness, patience, support and responsiveness at any time of the day and at any day of the week. They encouraged me a lot to pursue a career in academic research. I feel truly blessed to be their student. I also want to thank Daniel Pizarro for his guidance during my first steps in the academic research.

I would like to sincerely thank all the jury members, Vincent Lepetit, Peter Sturm and Sylvie Chambon, for reading my work. I believe that their remarks were very insightful and valuable for my further research.

My PhD was funded by the EU's FP7 through the ERC research grant 307483 FLEXABLE and hosted by the Université Clermont Auvergne through the Ecole Doctorale des Sciences pour l'Ingénieur. I am very thankful for the support that I received for my PhD.

I would like to thank Julia, Natalia and Isabelle for making my PhD easier by taking care of all administrative procedures with a friendly attitude. I would also like to thank the IT service department for providing good working conditions and their quick responsiveness during technical glitches.

This PhD has become a beautiful part of my life due to the amazing people I met and the friendships I developed with them. I am thankful to all these people for welcoming me to the lab, sometimes they supported me even when they had left the lab. These warm and amazing people supported me with words, technical help and tremendous fun at the parties. They added beautiful colors to these PhD years. In addition, I am thankful to some people outside the lab, especially the people at the FEU and my church. I grew a lot, lived beautiful things and met edifying people at these places. I also want to thank the Breton family for their support, prayers and all the weekends where we shared good food and relaxing moments.

My biggest pillars of support during my PhD were Sarah and my family. I want to thank Sarah for encouraging me to do this PhD even if it meant for us to stay apart. I am deeply grateful for her support, her understanding, her patience, our regular and lovely reunions in the train stations, our trips, our daily calls, her prayers and her love. Thanks for raising my spirits when needed. Merci, chérie!

In the end, I want to thank my parents and my aunt for their very long emotional and financial support, true love, wisdom and teachings of hard work. This journey so far has been made possible only due to their dedication towards me since my birth. Gracias!

S.D.G

Abstract

Monocular deformable 3D reconstruction is the general problem of recovering the 3D shape of a deformable object from monocular 2D images. Several scenarios have emerged: the Shape-from-Template (SfT) and the Non-Rigid Structure-from-Motion (NRSfM) are two approaches intensively studied for their practicability. The former uses a single image depicting the deforming object and a template (a textured 3D shape of this object in a reference pose). The latter does not use a template, but uses several images and recovers the 3D shape in each image. Both approaches rely on the motion of correspondences between the images and deformation priors, which restrict their use to well-textured surfaces which deform smoothly. This thesis advances the state-of-the-art in SfT and NRSfM in two main directions. The first direction is to study SfT for the case of 1D templates (*i.e.* curved, thin structures such as ropes and cables). The second direction is to develop algorithms in SfT and NRSfM that exploit multiple visual cues and can solve complex, real-world cases which were previously unsolved. We focus on isometric deformations and reconstruct the outer part of the object. The technical and scientific contributions of this thesis are divided into four parts.

The first part of this thesis studies the case of a curvilinear template embedded in 2D or 3D space, referred to Curve SfT. We propose a thorough theoretical analysis and practical solutions for Curve SfT. Despite its apparent simplicity, Curve SfT appears to be a complex problem: it cannot be solved locally using exact non-holonomic partial differential equation and is only solvable up to a finite number of ambiguous solutions. A major technical contribution is a computational solution based on our theory, which generates all the ambiguous solutions.

The second part of this thesis deals with a limitation of SfT methods: reconstructing creases. This is due to the sparsity of the motion constraint and regularization. We propose two contributions which rely on a non-convex energy minimization framework. First, we complement the motion constraint with a robust boundary contour constraint. Second, we implicitly model creases with a dense mesh-based surface representation and an associated robust smoothing constraint, which deactivates curvature smoothing automatically where needed, without knowing *a priori* the crease location.

The third part of this thesis is dedicated to another limitation of SfT: reconstructing poorly-textured surfaces. This is due to correspondences which cannot be obtained so easily on poorly-textured surfaces (either sparse or dense). As shading reveals details on poorly-textured surfaces, we propose to combine shading and SfT. We have two contributions. The first is a cascaded initialization which estimates sequentially the surface's deformation, the scene illumination, the camera response and then the surface albedos from deformed monocular images. The second is to integrate shading to our previous energy minimization framework for simultaneously refining deformation and photometric parameters.

The last part of this thesis relaxes the knowledge of the template and addresses two limitations of NRSfM: reconstructing poorly-textured surfaces with creases. Our major contribution is an extension of the second framework to recover jointly the 3D shapes of all

input images and the surface albedos without any template.

Keywords: 3D Curves, 3D Surfaces, Creases, Non-Smooth, Poorly-Textured, Shape-from-Template, Non-Rigid Structure-from-Motion, Shading, Contour, Isometry, Photometric Calibration, M-estimators

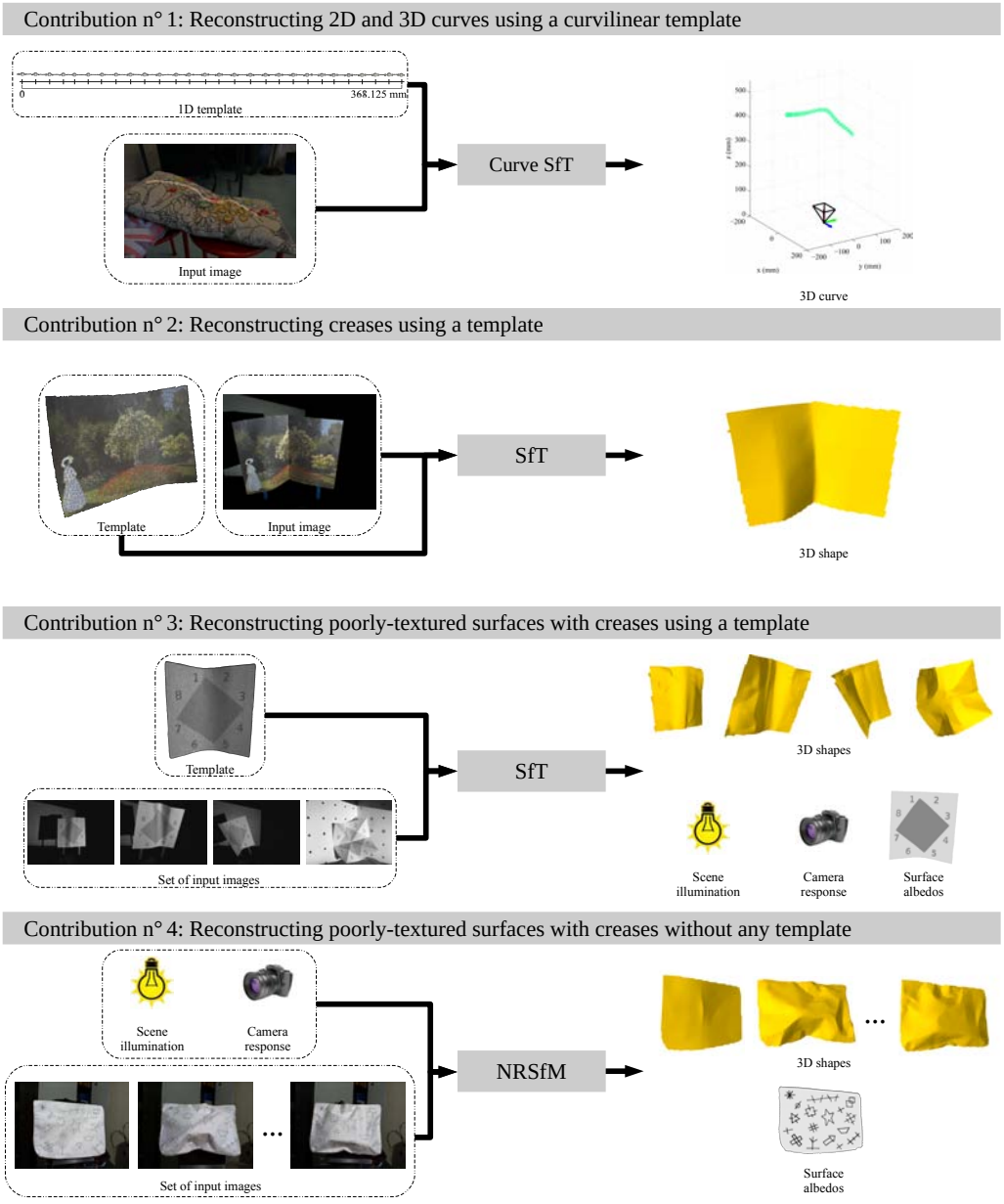


Figure 1: Illustration of our four contributions to monocular deformable 3D reconstruction.

Résumé

La reconstruction 3D monoculaire déformable est le problème général d'estimation de forme 3D d'un objet déformable à partir d'images 2D. Plusieurs scénarios ont émergé : le Shape-from-Template (SfT) et le Non-Rigid Structure-from-Motion (NRSfM) sont deux approches qui ont été grandement étudiées pour leur applicabilité. La première utilise une seule image qui montre un objet se déformant et un patron (une forme 3D texturée de l'objet dans une pose de référence). La seconde n'utilise pas de patron, mais utilise plusieurs images et estime la forme 3D dans chaque image. Les deux approches s'appuient sur le mouvement de points de correspondances entre les images et sur des *a priori* de déformations, restreignant ainsi leur utilisation à des surfaces texturées qui se déforment de manière lisse. Cette thèse fait avancer l'état de l'art du SfT et du NRSfM dans deux directions. La première est l'étude du SfT dans le cas de patrons 1D (*c-à-d.* des courbes comme des cordes et des câbles). La seconde direction est le développement d'algorithmes de SfT et de NRSfM qui exploitent plusieurs indices visuels et qui résolvent des cas réels et complexes non-résolus précédemment. Nous considérons des déformations isométriques et reconstruisons la partie extérieure de l'objet. Les contributions techniques et scientifiques de cette thèse sont divisées en quatre parties.

La première partie de cette thèse étudie le SfT curvilinéaire, qui est le cas du patron curvilinéaire plongé dans un espace 2D ou 3D. Nous proposons une analyse théorique approfondie et des solutions pratiques pour le SfT curvilinéaire. Malgré son apparente simplicité, le SfT curvilinéaire s'est avéré être un problème complexe : il ne peut pas être résolu à l'aide de solutions locales non-holonomes d'une équation différentielle ordinaire et ne possède pas de solution unique, mais un nombre fini de solutions ambiguës. Une contribution technique majeure est un algorithme basé sur notre théorie, qui génère toutes les solutions ambiguës.

La deuxième partie de cette thèse traite d'une limitation des méthodes de SfT : la reconstruction de plis. Cette limitation vient de la parcimonie de la contrainte de mouvement et de la régularisation. Nous proposons deux contributions qui s'appuient sur un cadre de minimisation d'énergie non-convexe. Tout d'abord, nous complétons la contrainte de mouvement avec une contrainte robuste de bord. Ensuite, nous modélisons implicitement les plis à l'aide d'une représentation dense de la surface basée maillage et d'une contrainte robuste de lissage qui désactive automatiquement le lissage de la courbure sans connaître *a priori* la position des plis.

La troisième partie de cette thèse est dédiée à une autre limitation du SfT : la reconstruction de surfaces peu texturées. Cette limitation vient de la difficulté d'obtenir des correspondances (parcimonieuses ou denses) sur des surfaces peu texturées. Comme l'ombrage révèle les détails sur des surfaces peu texturées, nous proposons de combiner l'ombrage avec le SfT. Nous présentons deux contributions. La première est une initialisation en cascade qui estime séquentiellement la déformation de la surface, l'illumination de la scène, la réponse de la caméra et enfin les albédos de la surface à partir d'images monoculaires où la surface se déforme. La seconde est l'intégration de l'ombrage à notre précédent cadre de minimisation d'énergie afin de raffiner simultanément les paramètres photométriques et de déformation.

La dernière partie de cette thèse relâche la connaissance du patron et aborde deux limitations du NRSfM : la reconstruction de surfaces peu texturées avec des plis. Une contribution majeure est l'extension du second cadre d'optimisation pour la reconstruction conjointe de la forme 3D de la surface sur toutes les images d'entrée et des albédos de la surface sans en connaître un patron.

Mots-clés : Courbes 3D, Surfaces 3D, Plis, Non-lisses, Surfaces peu texturées, Shape-from-Template, Non-Rigid Structure-from-Motion, Ombrage, Contour, Isométrie, Calibration photométrique, M-estimateurs

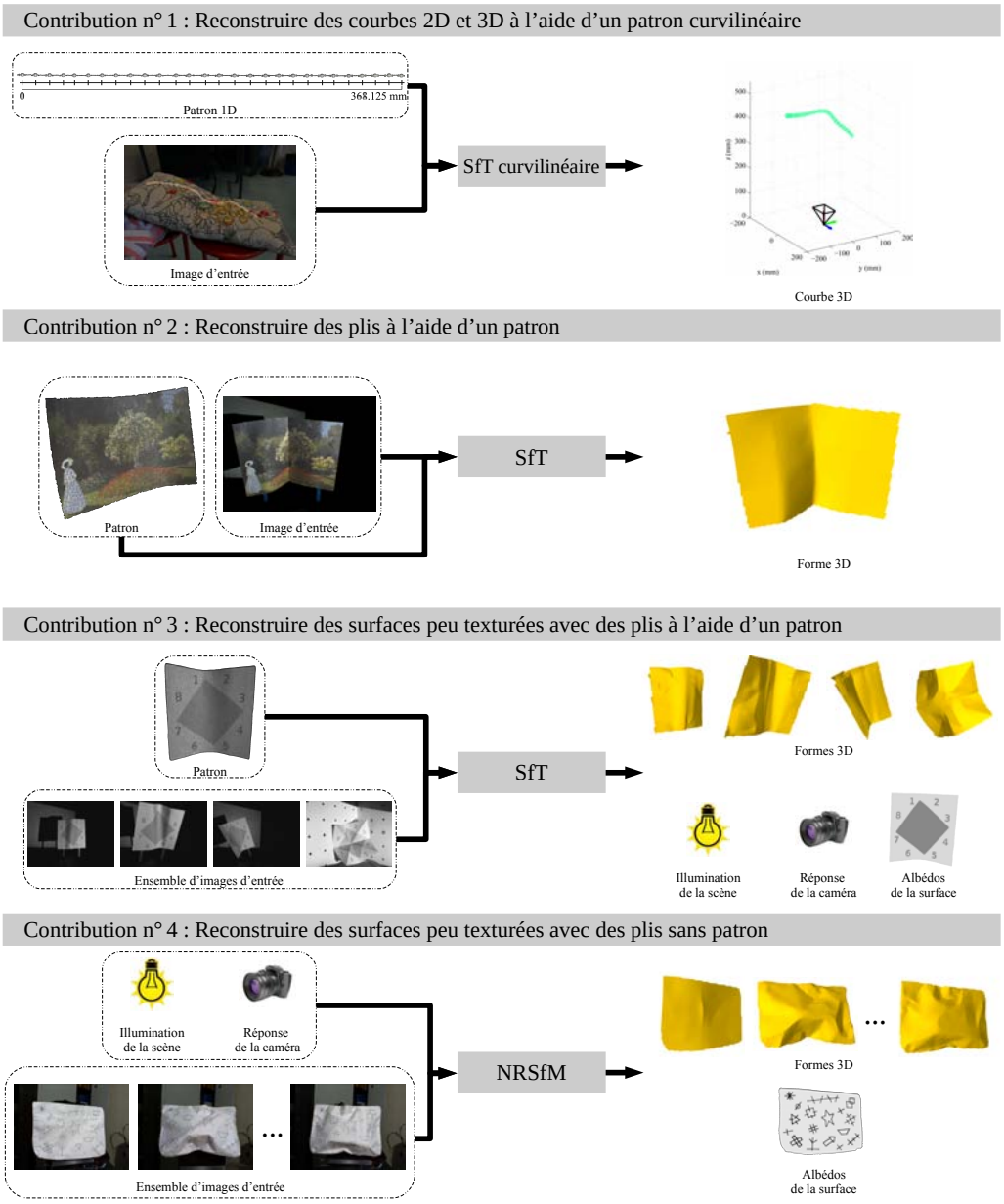


Figure 2: Illustration de nos quatre contributions à la reconstruction 3D monoculaire déformable.

Contents

List of Abbreviations

AR	Augmented Reality
ARAP	As-Rigid-As-Possible
CAD	Computer Aided Design
CNN	Convolutional Neural Networks
CT	Computed Tomography
DCT	Discrete Cosinus Transform
DoF	Degree-of-Freedom
DRBM	Deformable Render-based Block Matching
EKF	Extended Kalman Filter
EXIF	Exchangeable Image File Format
FEM	Finite Element Method
GN	Gauss-Newton
GT	Ground-truth
HMM	Hidden Markov Model
ICP	Iterative Closest Point
IRLS	Iteratively Reweighted Least Squares
IVP	Initial Value Problem
MDH	Maximum Depth Heuristic
MIS	Minimally Invasive Surgery
MIVP	Multiple Initial Value Problem

MRI	Magnetic Resonance Imaging
MVS	Multi-View Stereo
NRSfM	Non-Rigid Structure-from-Motion
NRSfMS	Non-Rigid Structure-from-Motion-and-Shading
NURBS	Non-Uniform Rational Basis Spline
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
RANSAC	RANdom SAmple Consensus
SDP	Semi-Definite Programming
SfM	Structure-from-Motion
SfS	Shape-from-Shading
SfT	Shape-from-Template
SfTS	Shape-from-Template-and-Shading
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SOCP	Second-Order Cone Programming
SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
TPS	Thin-Plate Splines

Chapter 1

Introduction

Summary

This chapter provides the thesis's scientific context and an overview of the main approaches to monocular deformable 3D reconstruction and registration. We first present the main applications of monocular deformable 3D reconstruction and registration. We then explain the current limitations of two main approaches and the contributions of the thesis: the use of multiple visual cues with surface models and the study of curvilinear models for monocular deformable 3D reconstruction and registration. We close this chapter by giving the structure of the thesis and the notations and symbols used in the following chapters.

1.1 Computer Vision, 3D Reconstruction and Registration

The discipline of computer vision looks at the theories and systems for extracting information from images and videos, and inferring higher level understandings of the visual content. The fundamental problems of computer vision can be encapsulated by '3Rs': Recognition, Registration and Reconstruction. Recognition is about determining which objects, activities or events are present in the image. It has numerous sub-problems such as object detection, image classification, image retrieval and semantic segmentation. Registration is about determining which points across images correspond to the same physical point, either with respect to another image, or some other reference domain. Reconstruction is about recovering the 3D shape of an object from one or more images. This thesis focuses on solving registration and 3D reconstruction problems for deforming objects.

3D reconstruction methods can be roughly distinguished by six components: the acquisition hardware (active or passive sensor), whether the object or scene is rigid or deformable (non-rigid), the type of image information used (also called visual cues), the type of prior knowledge used, the problem modeling and the numerical approach for solving the problem. 3D reconstruction for rigid objects has been studied extensively. The multi-view geometry with rigid objects is well understood; mature and stable passive solutions have been proposed. Many solutions have even been commercialized [3Dflow, 2017; Agisoft, 2014]. However, passive 3D reconstruction and registration of non-rigid objects are still open challenges. Fundamentally, these problems are much less constrained than the rigid case. Taking up these challenges is important because many objects of interest are deformable, including faces, bodies, organs, clothes and fabrics. Good solutions have many applications in a range of domains including entertainment, medical imaging and mechanics. Another important point is that deformable 3D reconstruction currently lacks strong theoretical understanding. The extension of the multi-view geometry of rigid objects to the non-rigid objects is not straightforward and progress on this aspect has been still limited.

The recent development of inexpensive real-time active depth sensors, such as the Kinect, has facilitated many important applications concerning deformable objects. However, solving deformable 3D reconstruction with monocular passive methods remains very important and relevant. This comes from the inherent limitations of active depth sensors: they have a restricted range (they cannot capture the 3D when the object is too far or too close to the sensors), have a higher power consumption than RGB cameras, and are often strongly affected by outdoor illumination conditions. There may also be physical restrictions preventing their use for specific applications, such as endoscopic imaging. Finally, there are billions of RGB cameras used every day on mobile devices, which yields a huge potential for usage and commercialization and underlines the need of solving the problem of monocular deformable 3D reconstruction and registration.

1.2 The Main Paradigms for Deformable Object 3D Reconstruction

There are four main paradigms in deformable 3D reconstruction from monocular images. These go by the names of Shape-from-Template (SfT), Non-Rigid Structure-from-Motion (NRSfM), Shape-from-Shading (SfS) and learning-based monocular 3D reconstruction methods. Figure 1.1 illustrates the main differences between them, by considering their inputs and outputs. We now give an overview of each paradigm.

1.2.1 Preliminary Definitions

As the notion of crease is fundamental in our work, we give here its precise definition and two related terms.

Definition 1 (*Crease*)**.** *We define a crease as a discontinuity of the first derivative of the surface.*

Definition 2 (*Creasable surface*)**.** *We define a creasable surface as a surface which may crease.*

Definition 3 (*Creased surface*)**.** *We define a creased surface as a surface which presents at least one crease.*

1.2.2 Shape-from-Template

The objective of SfT is to reconstruct the 3D shape of a deformed object using a single image and a textured 3D model of the object in a reference position [Bartoli et al., 2015; Salzmann and Fua, 2011]. SfT works by registering the object to the input image and deforming the template of the object: SfT performs simultaneously 3D reconstruction and dense registration. From the first works of [Gumerov et al., 2004; Perriollat et al., 2008; Salzmann et al., 2007a] to the more recent real-time methods [Collins and Bartoli, 2015; Ngo et al., 2016], SfT is one the paradigm which has been the most rapidly developed. SfT methods are also called template-based or model-based reconstruction methods. Figure 1.1 (row n°1) gives a general description of SfT. A template can be broken down into a shape model, an appearance model usually with a texture-map and a deformation model. The shape model is a 3D model of the object in a known reference position and can be built in various ways depending on the application. For example, the template can be generated from a Computer Aided Design (CAD) model of the object, or reconstructed from data such as Structure-from-Motion (SfM), with a set of rigid views. Nearly all SfT methods are named 'Surface' SfT methods because the template is a thin-shell model without volume. There also exist some 'Volume' SfT methods [Collins and Bartoli, 2015; Parashar et al., 2015]. These model the object with a volumetric deformable model, using either continuous models, such as 3D splines [Parashar et al., 2015] or discrete models, such as tetrahedral meshes [Collins

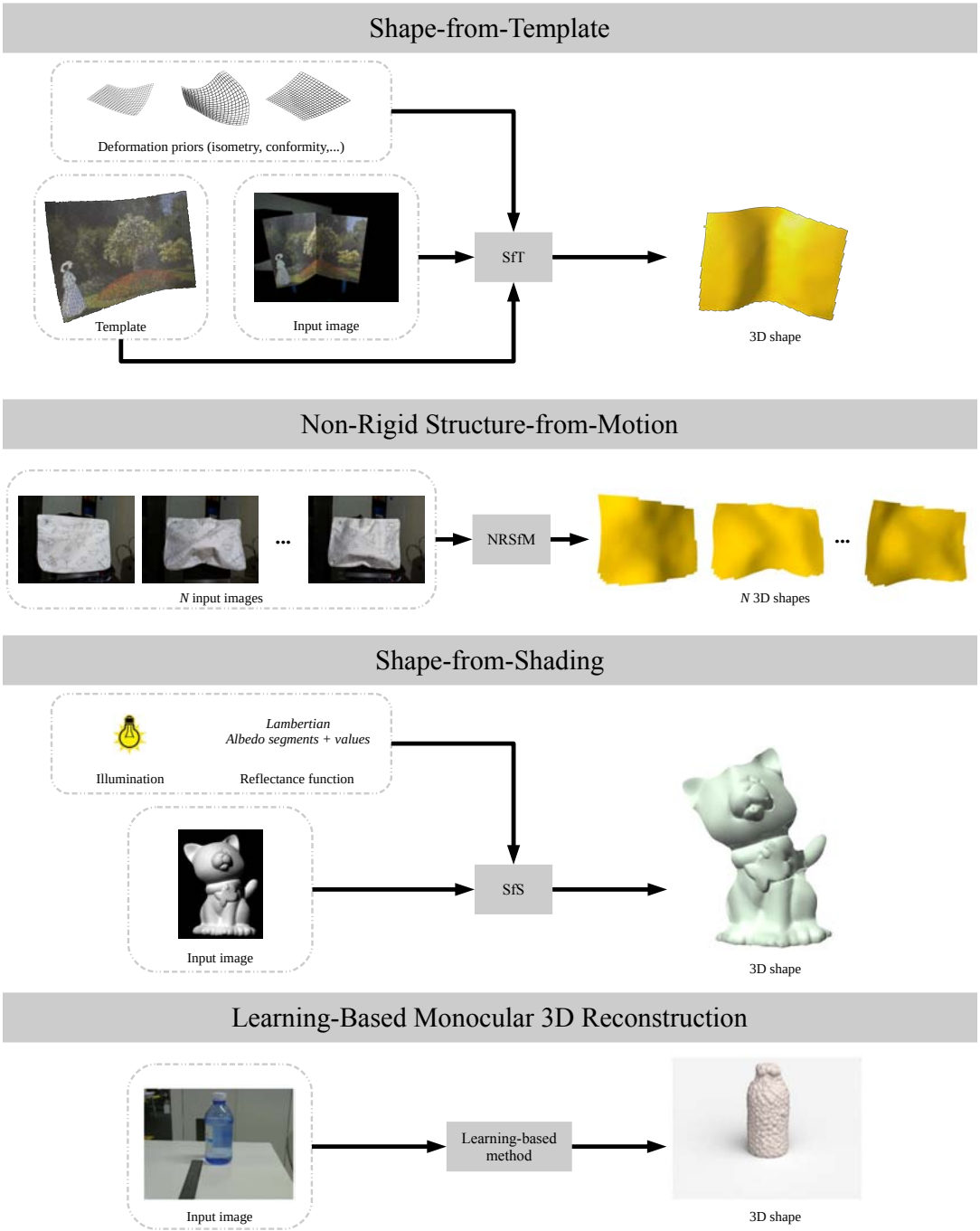


Figure 1.1: Four monocular deformable 3D reconstruction techniques: SfT (row n°1), NRSfM (row n°2), SfS (row n°3) and learning-based monocular 3D reconstruction from deep neural networks (row n°4). **Row n°1:** the SfT method used here is [Chhatkuli et al., 2017]; it gives the best result for the shown dataset. **Row n°2:** the NRSfM method used here is [Chhatkuli et al., 2014]; it gives the best result for the shown dataset. **Row n°3:** the SfS method used here is [Xiong et al., 2015], which is one of the state-of-the-art methods. Result extracted from [Xiong et al., 2015]. **Row n°4:** the learning-based monocular 3D reconstruction method used here is [Fan et al., 2017], which is one of the state-of-the-art methods. Result extracted from [Fan et al., 2017].

and Bartoli, 2015]. The development of SfT with 1D curve templates has not been reported in the literature. We introduce the special case of 'Curve' SfT in chapter 3 and reveal its practical uses and hidden complexity.

Most SfT methods use apparent motion of the object (also called motion cues). In practice, motion can use features matches, such as Scale-Invariant Feature Transform (SIFT) [Lowe, 2004] or Speeded Up Robust Features (SURF) [Bay et al., 2008] to construct sparse matches between the template and the input image. Alternatively, dense matching such as [Collins and Bartoli, 2015; Malti et al., 2011; Yu et al., 2015] can be used. However, motion cues are often insufficient to infer the 3D shape of a deforming object, because an image can be generated by possibly infinitely many deformations (the so-called depth ambiguity). To make SfT (and more globally, all deformable 3D reconstruction problems) well-constrained, deformation priors are required. This idea can be found in rigid 3D reconstruction, where a rigidity prior is considered. The most studied deformation prior is isometry, which means that distances on the object's surface are preserved during deformation. The success of isometry comes from the fact that it is simple to model mathematically and approximates well the behavior of most objects under near-isometric deformations, such as clothes and fabrics. Also, it has been shown to make SfT well-posed [Bartoli et al., 2015] assuming dense correspondences.

Several existing SfT methods give stable and accurate solutions, but only for well-textured surfaces under smooth deformations. These methods may fail in two cases: when the object is poorly-textured or when it deforms non-smoothly. At poorly-textured regions, features are very sparse and may not be reliable, and dense matching cannot perform well. Fundamentally, motion is insufficient in these regions to neither accurately recover the deformed shape nor accurately register the template. The incapability of these methods to handle non-smooth deformations comes from two reasons. First, motion information is not usually sufficient to infer precisely where discontinuity such as creases occurs, if it happens in a textureless region. Second, most existing methods model deformations with a low-dimensional set of bases or reduce its dimensionality using an ℓ_2 smoothing prior, which is done in nearly all previous approaches to reduce the problem's dimensionality and provide sufficient regularization. The problem however is that this prior generates smooth rather than discontinuous solutions. Figure 1.1 (row n°1) illustrates the incapability of a state-of-the-art method [Chhatkuli et al., 2017] to reconstruct complex deformations such as the crease in the middle of a sheet of paper. Some methods have tackled the problem of reconstructing 'sharp folds' [Salzmann and Fua, 2009] and poorly-textured surfaces [Ngo et al., 2015; Varol et al., 2012b]. However, folds are not accurately modeled in [Salzmann and Fua, 2009], a full photometric calibration to use shading is required in [Varol et al., 2012b], and the template has to be registered to the first input image in [Ngo et al., 2015].

1.2.3 Non-Rigid Structure-from-Motion

The objective of NRSfM is to reconstruct the 3D shapes of a deformable object using only a set of 2D images. Unlike SfM, in NRSfM the object can undergo deformation in each image [Bregler et al., 2000]. NRSfM methods are also called template-free or model-free reconstruction methods. Figure 1.1 (row n°2) illustrates the different components of NRSfM. This problem is much harder than SfT because no template is available and consequently, the physical structure of the object is unknown. NRSfM considers a set of monocular images instead of a single image as SfT, and aims to simultaneously reconstruct a template (or a subspace describing the template's deformation space) and compute the template's deformation with respect to each image. The significant increase in difficulty explains why SfT solutions have evolved faster than NRSfM solutions and why mature and stable NRSfM implementations have not been proposed yet.

Similarly to SfT methods, most existing NRSfM methods use apparent motion, using feature-correspondences or optical flow [Garg et al., 2013]. They necessarily integrate deformation priors to make NRSfM solvable. There are two categories: the statistics-based [Akhter et al., 2009; Bregler et al., 2000; Dai et al., 2014; Garg et al., 2013; Gotardo and Martinez, 2011; Torresani et al., 2008a] and physics-based priors [Agudo et al., 2016; Chhatkuli et al., 2014, 2016; Varol et al., 2009; Vicente and Agapito, 2012; Wang et al., 2016]. Statistics-based priors use dimensionality reduction and thus do not explicitly model the physical behavior of the object. Common physics-based priors are isometry [Chhatkuli et al., 2014; Collins and Bartoli, 2010; Taylor et al., 2010; Varol et al., 2009; Vicente and Agapito, 2012], inextensibility [Chhatkuli et al., 2016] and linear elasticity [Agudo et al., 2016]. Because they model the real behavior of the object, they usually result in better constrained problems, and consequently more accurate results, compared to statistics-based priors. However, a majority of existing NRSfM methods suffer from the same limitations as SfT methods: they break down when the object to be reconstructed is poorly-textured or deforms non-smoothly. Figure 1.1 (row n°2) gives the output of a state-of-the-art method [Chhatkuli et al., 2014] which is not capable of reconstructing complex deformation such as the creases. [Wang et al., 2016] has proposed a method which handles poorly-textured surfaces, however, it only reconstructs very smooth surfaces.

1.2.4 Shape-from-Shading

The objective of SfS is to recover the surface depth at each pixel of a single image using shading information. Importantly, it does not assume any template of the surface known *a priori* and does not perform any registration. It works exclusively with shading information through the photometric relationship between the surface, the surface reflectance, the scene illumination and the pixel intensity [Horn, 1970]. The surface reflectance tells us how the light is reflected by the surface. Figure 1.1 (row n°3) shows the different components of SfS. Unlike motion used in SfT and NRSfM, shading is the most important visual cue for inferring high-frequency shape details at poorly-textured regions [Pentland, 1988].

However, SfS is fundamentally an ill-posed problem [Belhumeur et al., 1997; Pentland, 1984] and then requires a photometric calibration of the scene to be known *a priori*. A photometric calibration involves knowing surface reflectance (mainly diffuse and specular reflections) and scene illumination. The non-uniqueness of the solution to SfS is often presented with the concave/convex ambiguities [Belhumeur et al., 1997]. Furthermore, SfS cannot reconstruct self-occluded surfaces. This makes SfS quite unpractical and this is why most methods work in very controlled environments (illumination and reflectance known) and/or simulated data.

SfS methods differ mainly by the way they solve the problem. Five categories of method exist: *(i)* propagation approaches [Ahmed and Farag, 2007; Kimmel and Bruckstein, 1994; Prados and Faugeras, 2005; Rouy and Tourin, 1992], *(ii)* local approaches [Pentland, 1984; Xiong et al., 2015], *(iii)* linear approaches [Pentland, 1988; Tsai and Shah, 1994], *(iv)* minimization approaches [Barron and Malik, 2015; Ikeuchi and Horn, 1981; Lee and Kuo, 1993] and *(v)* learning-based approaches [Richter and Roth, 2015]. In general, categories *(iv)* and *(v)* methods are more robust, while the other categories are faster.

1.2.5 Learning-Based Monocular 3D Reconstruction

The objective of learning-based methods is to predict the 3D shape of an object or the depth-map of a scene from a single image using a training dataset. Similarly to SfS, they do not perform any registration. Most of these methods use neural networks, but works previous to the recent development of neural networks have proposed to learn to predict depth-maps using for instance graphical models [Saxena et al., 2009]. This paradigm poses the monocular deformable 3D reconstruction problem as a supervised learning. This is a recently emerging category [Choy et al., 2016; Fan et al., 2017; Godard et al., 2017; Kar et al., 2015; Laina et al., 2016; Richardson et al., 2016] and has shown to work for common object classes with very large datasets available. Examples include cars and furniture [Choy et al., 2016; Fan et al., 2017; Kar et al., 2015], particular indoor scenes [Godard et al., 2017; Laina et al., 2016] such as bedrooms, offices and roads, and faces [Richardson et al., 2016] using specific low-dimensional deformation models. This category has not shown to enable 3D reconstruction of objects under very high dimensional deformations, which notably limits for now its applicability. Two other shortcomings are that the training and the test data domains have to match very closely and reasoning about well-posedness and ambiguities with neural networks is extremely hard. Indeed, no clear answer has been proposed, however such analysis is very important for diagnosing when the problem can and cannot be solved. Figure 1.1 (row n°4) gives an example of 3D reconstruction from [Fan et al., 2017].

1.2.6 3D Reconstruction and Registration with Multiple Cues, and Consistent Naming Convention

Nearly all SfT and NRSfM methods use motion as the main visual cue, but some works combine motion with other cues to overcome difficult scenarios such as poorly-textured sur-

faces. Some methods also differ by the additional parameters they estimate, such as the surface reflectance. For the sake of clarity, we complement the two main paradigms, SfT and NRSfM, by the suffix 'S' when shading is used. Using this notation, we propose in table 1.1 a systematic definition of the different general SfT and NRSfM problems. For each method abbreviation, we give some previous works.

Abbreviation	Description	Key references
SfT	SfT using motion	[Salzmann and Fua, 2009] [Bartoli et al., 2015]
SfTS	Shape-from-Template-and-Shading using motion and shading	[Malti and Bartoli, 2014] [Liu-Yin et al., 2016]
NRSfM	NRSfM using motion	[Bregler et al., 2000]
NRSfMS	Non-Rigid Structure-from-Motion-and-Shading using motion and shading	×

Table 1.1: Definition of the different general SfT and NRSfM problems.

We refer to the silhouette of a 3D object with a plane topology as boundary contours. In the SfT literature, the two main complementary visual cues are the boundary contour [Salzmann et al., 2007b; Vicente and Agapito, 2013] and shading [Liu-Yin et al., 2016; Malti and Bartoli, 2014], which form Shape-from-Template-and-Shading (SfTS).

In the NRSfM literature, only one work combines NRSfM with the boundary contour constraint [Wang et al., 2016] to handle poorly-textured surfaces, which is related to one of our contributions. It uses a simple boundary contour constraint which reduces the distance in the input images between the projected contours and the image edges which are detected by a Canny filter. [Wang et al., 2016] uses a brightness constancy constraint which gives dense correspondences. There is no previous work on Non-Rigid Structure-from-Motion-and-Shading (NRSfMS).

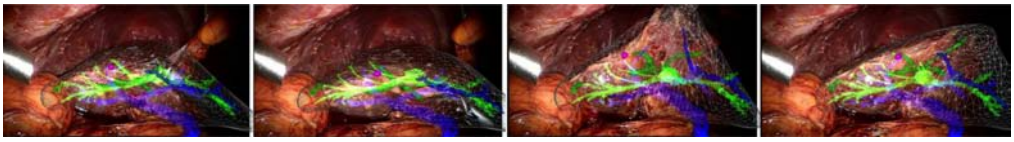
[Liu-Yin et al., 2016; Malti and Bartoli, 2014; Salzmann et al., 2007b; Vicente and Agapito, 2013] and [Wang et al., 2016] are respectively specific instances of the general problem of SfT and NRSfM. There are lots of problem components which must be specified when defining a problem (SfT, SfTS, NRSfM or NRSfMS) and then a specific instance. Many papers are not clear and systematic about all these problem components. We present a complete characterization of a problem instance in terms of models, scene assumptions, and known and unknown parameters. To denote a particular instance of the general SfT or NRSfM problem, we complement SfT or NRSfM by the suffix '-Y', with Y a positive integer, and write it in *italic*. We solve important and general instances of SfT, SfTS and NRSfMS. We denote these with the nomenclature *SfT-1*, *SfTS-1* and *NRSfMS-1*. The main characteristics of *SfT-1* are to solve the problem using motion constraints while simultaneously handling templates that can deform in complex, non-smooth ways such as creases. This has not been done successfully

before. The complete definition of *SfT-1* is described in table 4.2.2. The main characteristics of *SfTS-1* are to solve the problem when the reflectance and camera functions are unknown *a priori*. This has not been achieved before, and unlike SfT, it requires performing inference simultaneously over multiple images (four or more). The complete definition of *SfTS-1* is described in table 5.2.2. The main characteristics of *NRSfMS-1* are to solve the problem when the reflectance function is unknown *a priori*, and the object can have different deformations in all views. This has not been achieved before. The complete definition of *NRSfMS-1* is described in table 6.2.2.

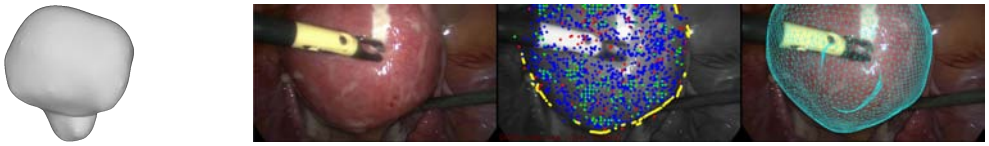
1.3 Motivation and Applications of Deformable Object 3D Reconstruction and Registration

Research on deformable 3D reconstruction and registration has raised considerable interest for many applications including medical imaging, facial performance capture and editing, interactive deformation transfer, Augmented Reality (AR) games and mechanical analytics. Some examples are shown in figure 1.2.

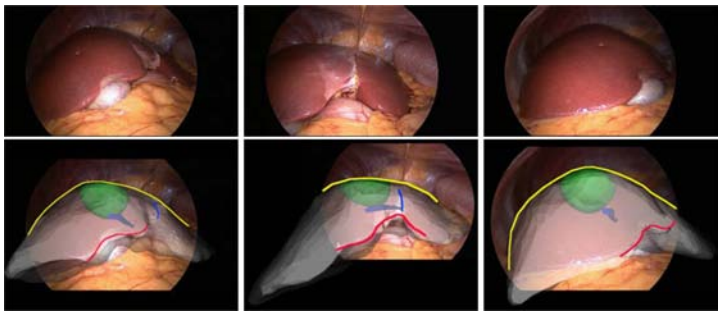
An important application is in medical augmented reality with Minimally Invasive Surgery (MIS) and more precisely with laparoscopic surgeries. This advanced surgery technique is performed by inserting through small incisions small surgical instruments and a laparoscope (thin, tube-like instrument with a light and a lens for viewing). The surgeon uses the live video stream from the laparoscope to perform the surgery. The main advantages of MIS are that the patient's trauma is strongly reduced and the recovery time is shorter, compared to open surgeries. However, during MIS, surgeons face three main problems: the viewpoint is limited, the localization in 3D and the perception of depth become harder. AR appears to be a very suitable way to give a real-time feedback during MIS. This is done by augmenting the live video stream with the organs' shape and/or the internal structures from a pre-operative image, such as Magnetic Resonance Imaging (MRI). For instance, an SfT method is proposed on liver surgery by [Haouchine et al., 2013]. As figure 1.2 shows, it registers into the images of a stereo laparoscope the tumors (in purple) and the internal structures of the liver, such as the hepatic veins (in blue) and portal veins (in green), using a 3D template built from a Computed Tomography (CT) scan. It shows encouraging results, but considerable work remains to achieve a reliable and robust solution. Using a monocular laparoscope, a deformable registration of a pre-operative template of a liver (obtained from CT) was presented in [Koo et al., 2017]. This permits one to register at the same time the tumor (in green) and the internal structures of the liver such as veins (in blue). This is done using silhouette and shading cues. The Volume SfT real-time method of [Collins and Bartoli, 2015] was applied by [Collins et al., 2016] to deformable tracking of organs such as kidneys or uterus with monocular laparoscopic videos. It also permits one to update on demand the texture-map of the organ, which is helpful because the organ texture may change during surgery.



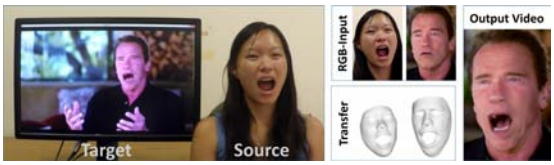
Applications to medical AR: real-time augmentation of vascular network and tumors for MIS [Haouchine et al., 2013]



Applications to medical AR: real-time deformable tracking of organs for MIS [Collins et al., 2016]

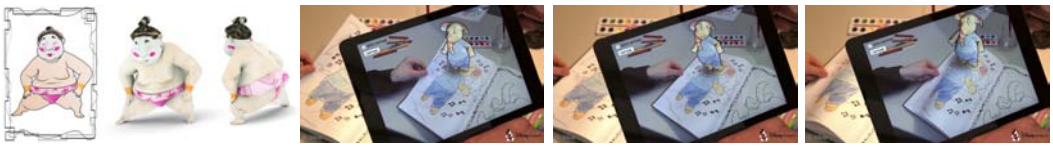


Applications to medical AR: deformable 3D registration of liver and some of its internal structure for MIS [Koo et al., 2017]

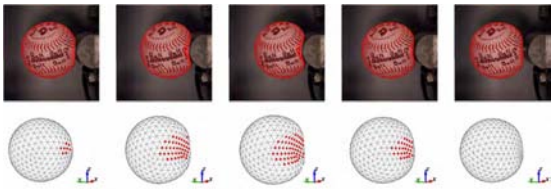


Applications for real-time transfer of facial expression: [Thies et al., 2016]

Applications for real-time generic deformation transfer: [Collins and Bartoli, 2015]



Applications for AR games: an AR coloring book application from [Magnenat et al., 2015]



Applications for mechanical model analytics: a soft-ball impact analysis [Smith et al., 2016]

Figure 1.2: Applications of deformable 3D reconstruction and registration.

The 3D reconstruction and the registration of non-rigid objects have also considerable applications in the movie post-production. Editors are often required to edit movies after the recording, by removing, introducing or modifying content. When the content is deformable, this can be highly labour-intensive. However, most movies are not recorded with depth sensors, which makes monocular methods extremely valuable. A real-time technique of facial performance capture and editing was proposed in [Thies et al., 2016]. It works by reconstructing the 3D faces of a source actor and of a target actor, and transfer the facial expression of the source actor to the target actor. Any special setup can be used to acquire the source actor, including a simple RGB camera. More generally, an interactive and real-time deformation transfer method was presented in [Collins and Bartoli, 2015]. This can be used in computer graphics animation in order to give more easily a desired shape to a digital object. [Collins and Bartoli, 2015] uses a real-time SfT solution and works for any generic object with a known template.

A vast domain of application is AR gaming. The idea is to offer to players new gameplay experiences and a different game environment since it uses the real world environment. Nearly all AR games assume the scene to be rigid. For instance, an AR coloring book application is presented in [Magnenat et al., 2015]: children can see in 3D the characters which they color in a printed deformable coloring book. The application runs in any device which has a camera, such as a tablet. A deformable 3D reconstruction algorithm is used to reconstruct the book page in real-time in order to render the 3D colored character over the book page.

Another application domain is mechanical analytics. Relevant experimental data is required to analyze soft body behavior and reduce the gap between the simulation modeling and the real behavior of the materials. For some products, obtaining such data may demand deformable 3D reconstruction and registration. This is the case of a soft ball presented by [Smith et al., 2016]. It uses the SfT solution of [Ngo et al., 2016] to reconstruct in 3D and register the surface of a soft ball during an impact. This allows one to evaluate the accuracy of its simulation framework.

1.4 Thesis Organization and Contribution

We illustrate the organization of this thesis in figure 1.3. We have advanced state-of-the-art in four main directions by considering four of its fundamental limitations. We first enumerate our different contributions and then give further details for each of them.

1) Curve SfT. The SfT literature lacked solutions and a theoretical understanding of the special case of a 1D template, *i.e.* where the shape is a 1D curve embedded in a 2D or a 3D space.

2) Surface SfT for creasable surfaces. Most of the existing SfT methods were built to be capable of reconstructing only smooth surfaces. However, non-smooth surfaces are very common in practice, such as a creased sheet of paper. The limitations of these methods originate from the insufficiency of motion constraint to infer where creases occur, and the use of dimensionality reduction and smoothing priors, which limit application to only smoothly

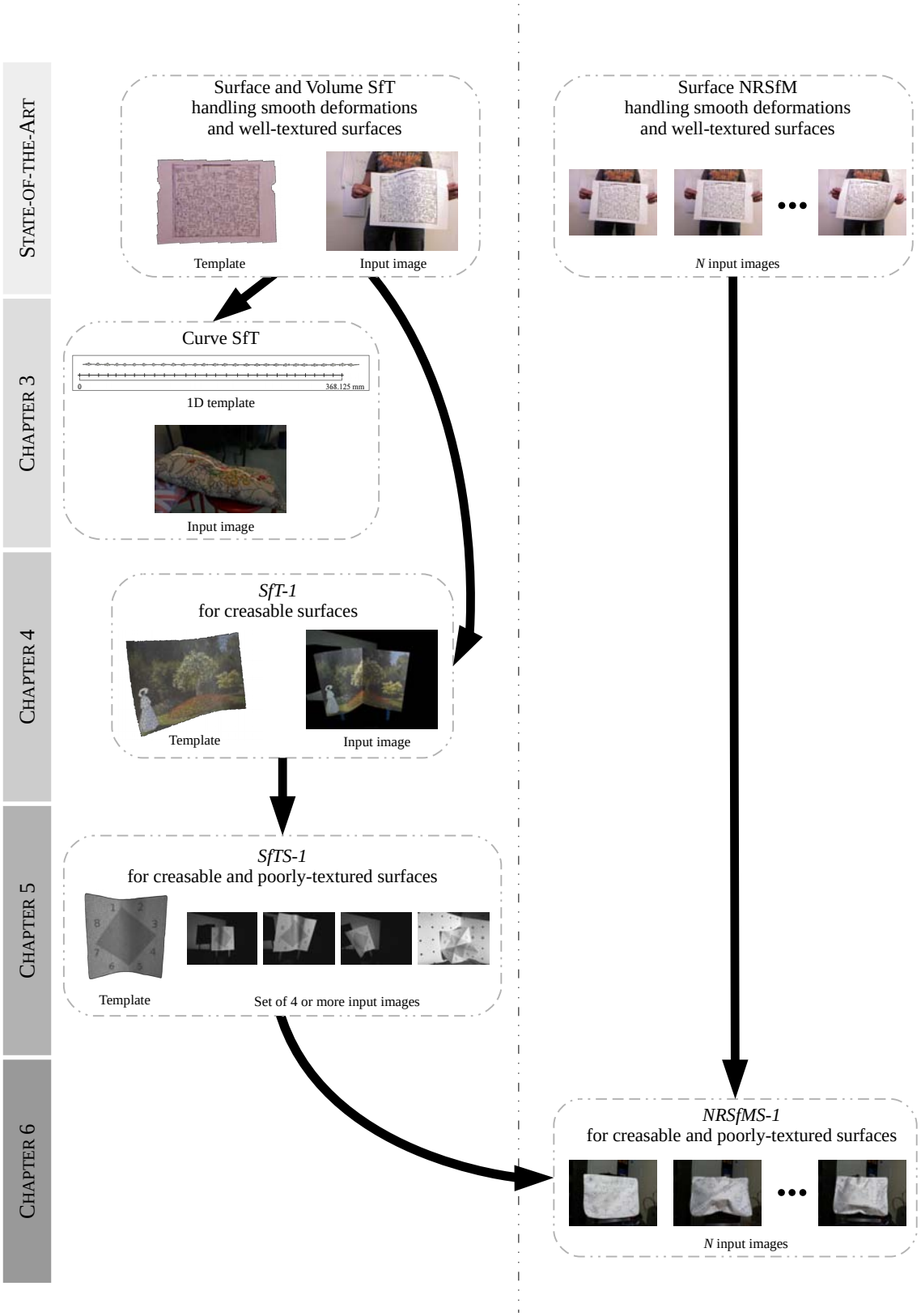


Figure 1.3: Overview of the main contributions of this thesis.

deforming objects.

3) Surface SfT for creasable and poorly-textured surfaces. Most of existing SfT methods give good results for well-textured surfaces, and almost all of them use feature-correspondences constraints. However, in practice, a lot of objects comprise well-textured surfaces with significant poorly-textured regions. Since correspondences cannot be obtained so easily on poorly-textured surfaces (either sparse or dense), other visual cues have to be used.

4) Surface NRSfM for creasable and poorly-textured surfaces. For the same reasons as SfT, most of existing NRSfM methods cannot reconstruct accurately poorly-textured surfaces under complex deformations. Solving this fundamentally changes the NRSfM paradigm because we cannot separate correspondence estimation from 3D reconstruction and we must introduce non-convex photometric constraints into the problem, which complexifies significantly the optimization process.

We now provide details on how we have contributed to solving the above four limitations.

1.4.1 Contribution to Curve SfT

We propose a thorough theoretical study and practical solutions of Curve SfT. As with all SfT cases, Curve SfT requires the use of deformation priors because of the loss of shape information from camera projection. We use the isometry prior. We consider in this thesis two main cases of Curve SfT. The first case is when the template is a curve embedded in the 3D space and observed by a regular 2D camera. A practical example is to reconstruct a thin necklace around a person's neck, given a template of the necklace, as shown at the bottom of figure 1.4. The second case is similar to the first case, but the camera is 1D. It may be created from an orthogonal view of the ground plane, for instance. At first glance, the use of 1D templates in Curve SfT may seem to make SfT simpler compared with a 2D template in Surface SfT. However, we found that *Curve SfT has fundamental theoretical differences concerning degeneracies, well-posedness and solution uniqueness.* These differences motivate us to propose new theoretic and algorithmic solutions.

Theoretical contributions to Curve SfT. We use continuous differential geometry to analyze and derive local solutions, problem well-posedness and ambiguities. We show that the two sub-cases of Curve SfT, which are two problems with different dimensions, can both be written as the same first-order Ordinary Differential Equation (ODE) and solved through an Initial Value Problem (IVP). However, the initial condition required to solve the IVP is a known depth at one point. At first glance, this additional information is generally unavailable. We propose a strategy to solve the IVP by giving an initial condition which is directly obtained from the ODE. This initial condition uses special points of the curve, called the *critical points*. Through the IVP with the critical points, the mathematical formulation of Curve SfT gives several solutions which we call *candidate solutions*. We prove the following results:

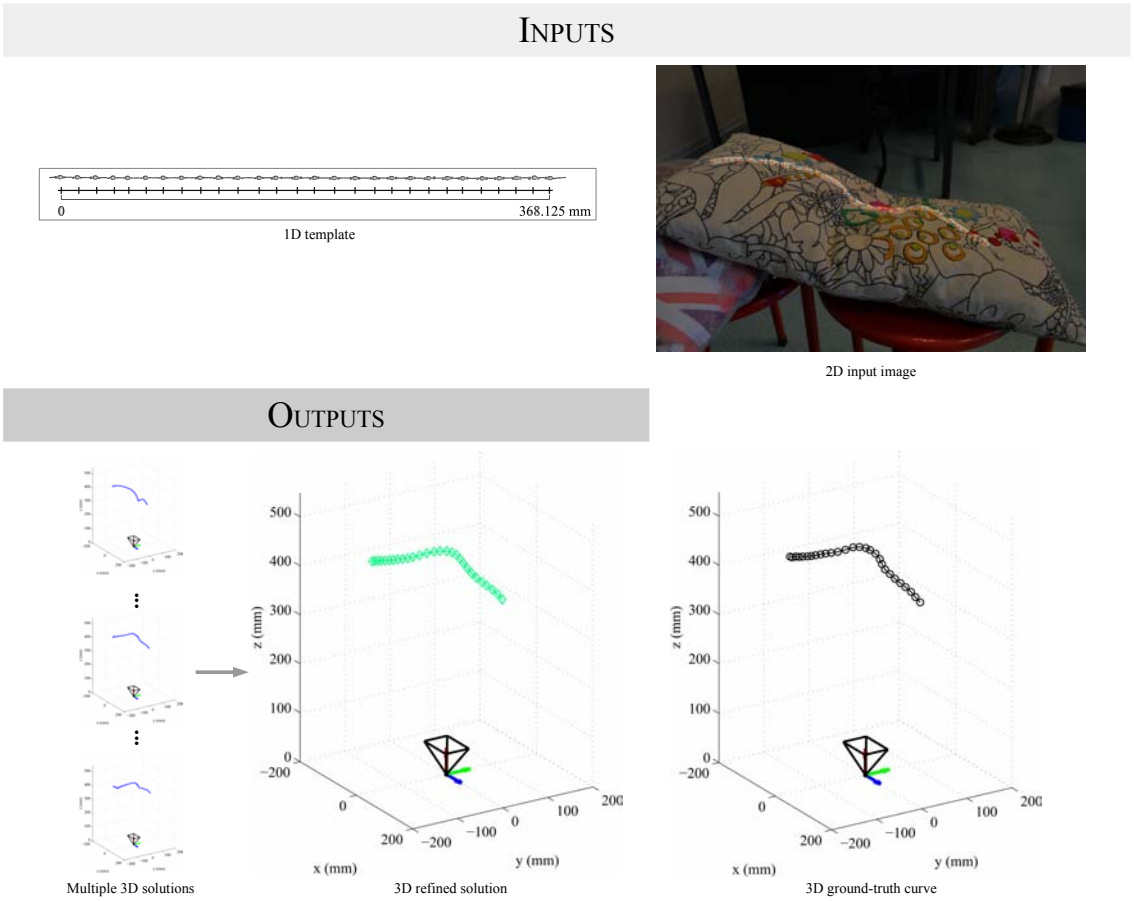


Figure 1.4: An example of **3D** curve reconstruction from a **2D** input image and a **1D** template using our refinement solution of Curve SfT. It uses the *necklace* dataset. In order to give a better visualization of the necklace on the pillow, we have brightened the region near the necklace by dimming the rest of the image. For our method, inputs are correspondences between the 1D template and the input image, which are the center of mass of the pearls. We show several candidate solutions obtained by our category *(iv)* method and give the refined version of the best solution. To detect the best solution, we use the ground-truth curve and select the one with the smallest reconstruction error.

1. In Curve SfT, the depth of a point is uniquely recoverable if and only if it is a critical point.
2. Curve SfT is solvable up to a finite number of solutions if and only if there exists at least one critical point.
3. A section of template falling between two critical points is recoverable up to a two-fold ambiguity.
4. A Curve SfT problem with N_s critical points has 2^{N_s+1} discrete candidate solutions.

We also study the solvability of Curve SfT with a method using local non-holonomic solution to our Partial Differential Equation (PDE). We prove the following results:

1. Unlike Surface SfT, Curve SfT cannot be solved exactly using local non-holonomic solution to our PDE.

2. By neglecting curvature, it is possible to solve Curve SfT using local non-holonomic solution to our PDE.

Technical contributions to Curve SfT. In the literature, there exist three categories of methods to solve Surface and Volume SfT: *(i)* local analytical solutions, *(ii)* convex optimization and *(iii)* non-convex iterative optimization. We give a computational solution for each category for solving Curve SfT. For the category *(i)* method, we first proceed to a similar differential analysis as [Bartoli et al., 2015] and then consider non-holonomic solutions under the assumption of infinitesimal linearity. For the category *(ii)* method, we adapt a convex formulation of Surface SfT, called Maximum Depth Heuristic (MDH) [Perriollat et al., 2011; Salzmann and Fua, 2011]. It uses the inextensibility constraint, a relaxation of the isometry constraint, which we review in chapter 2. For the category *(iii)* method, we propose a non-convex continuous formulation that can be optimized efficiently using gradient-based minimization. We achieve this with a novel angle-based parameterization which implicitly models isometric deformations. We also introduce a new category *(iv)* of SfT method, which uses a discrete graphical model. Our method models SfT with a discrete graphical model *without relaxing isometry* and *without requiring an initial estimate*. This makes it very different to the three categories of methods used previously for solving Surface and Volume SfT. Importantly, our category *(iv)* method generates all candidate solutions, as figure 1.4 illustrates. We emphasize that all existing isometric methods for solving Surface and Volume SfT and perspective cameras are only able to generate one solution, which cannot be used to solve Curve SfT.

1.4.2 Contribution to Surface SfT: Creasable Surfaces

As mentioned in §1.2.2, almost all of existing Surface and Volume SfT methods are not capable of reconstructing and registering accurately complex deformations such as creases. The closest work for handling creased surfaces is [Salzmann and Fua, 2009]. However, this does not model creases because the reconstructed creases are a by-product of the way the isometry prior is relaxed. Figure 1.5 shows that in practice there is no accurate Surface and Volume SfT method which reconstructs creases, such as found on a paper aeroplane model.

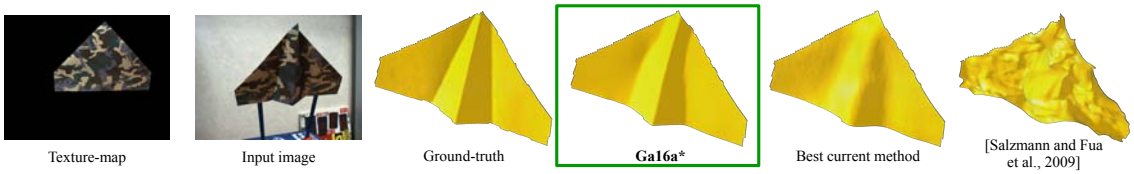


Figure 1.5: Visual results of our solution to the *SfT-1* problem for creasable surfaces and previous methods. It shows creased surface reconstructed by our method, denoted **Ga16a***, and the best previous method [Bartoli et al., 2015] and the closest work [Salzmann and Fua, 2009]. Unlike our method, [Bartoli et al., 2015] and [Salzmann and Fua, 2009] fail to reconstruct the creases.

Reconstructing and registering complex deformations is a challenging problem for two rea-

sons. First, we must deal with very high-dimensional deformation spaces, which are needed to model non-smooth deformations such as surface creases that can form in arbitrary places. One cannot therefore approximate the problem using a globally smooth surface representation (as is common in previous SfT methods), which both increases the search space dramatically, and leads to a highly non-convex energy landscape. Second, we do not know *a priori* the crease locations. This makes it very difficult to employ existing parametric crease representations used in models such as the B-splines, because we do not know *a priori* where to modify the spline to permit high changes in curvature or discontinuities.

To solve this problem, we optimize the deformation through a minimization of image data constraint, feature correspondences and boundary contour, and deformation priors. We refer to this problem as *SfT-1* and give a complete definition in table 4.2.2. We summarize the two main contributions as follows.

Modeling implicitly creases through an adaptive smoothing term acting on a high-resolution non-parametric surface mesh. We propose to not enforce globally smooth deformations and to not apply dimensionality reduction. Instead, we use a so-called non-parametric approach where the surface is modeled by a dense triangulated mesh. We have found that creased surfaces such as folded paper can be recovered using mesh resolutions of $\mathcal{O}(10^4)$ vertices. We are able to work with such high resolution meshes because the constraints we apply on the mesh are very sparse (each constraint only applies to a small number of vertices), and this allows us to solve the resulting system iteratively with sparse linear solvers. This solution does not require knowing anything *a priori* about the crease location, and they emerge as the lowest energy points at convergence. This is based on M-estimators and is inspired by discontinuity-preserving optical flow [Black and Anandan, 1993; Zach et al., 2007]. The idea is to deactivate automatically the smoothing where needed in order to prevent creases from being smoothed. M-estimators come in two main types: non-redescending and redescending. We have systematically compared two of the most common non-redescending M-estimators ($(\ell_1\text{-}\ell_2)$ and Huber) and the most common redescending M-estimator (Tukey). We found that by setting the Huber hyper-parameter correctly the results with the non-redescending M-estimators are comparable, and both allow us to convincingly reconstruct creased surfaces. However, we found that the redescending M-estimator does not work well. We expect that this result generalizes to most other common redescending M-estimators.

A robust boundary contour constraint. We complement feature correspondences constraints with a boundary contour constraint for two reasons. First, feature correspondences constraints are often not sufficient data constraints. Second, as crease location is not known *a priori*, the surface should be well registered in order to make creases emerge at the correct location. The boundary contour constraint encourages the boundary of the surface to project to strong intensity edges in the image. This is a powerful constraint and should be used wherever possible. One main challenge is that we must ensure that the boundary is attracted to correct image edges, which is not trivial. To deal with this we use statistical color models

to help disambiguate non-boundary edges (*e.g.* from background clutter or texture). In the broader context of SfT, this is the first time that segmentation models have been exploited to solve SfT problems.

1.4.3 Contribution to Surface SfT: Creasable and Poorly-Textured Surfaces

As figure 1.6 shows, poorly-textured surfaces are the second limitation of nearly all state-of-the-art SfT methods, particularly with creases. We want to use shading to constrain the surface densely at poorly-textured regions. As said earlier in §1.2.4, shading works on textureless regions and can be used also to infer fine surface details. We want to combine motion (from textured regions) and shading (from poorly-textured regions) constraints with the physical constraints from the template. Therefore, we propose to solve it by jointly registering and reconstructing the template and applying shading constraints densely over the template's surface. We refer to this problem as *SfTS-1* and give a complete definition in table §5.2.2. This is a novel and challenging problem. To apply shading constraints, we must model the surface reflectance (albedos for Lambertian surfaces), the scene illumination and the radiometric response of the camera. In some specific situations, surface reflectance may be known *a priori*, but this is not very common. For instance, arbitrary templates, from CAD models such as [TurboSquid, 2016; Warehouse, 2016] or 3D acquisition systems such as [David 3D Scanner, 2014], do not usually have it. Similarly, scene illumination and camera responses are usually not known *a priori*. We thus propose to solve all these unknowns (shapes, surface reflectance, scene illumination and camera responses) together as one joint system. We do this using at least four images. Our solution also allows us to upgrade the template into a template with the reflectance function of the surface. In figure 1.7, we show our general process and, in figure 1.6, we show a rendering of the 3D shape given by our method.

The outputs of our algorithm include the template's 3D shape for each view and a surface reflectance function. Given the surface reflectance function, it is then possible to use existing SfTS methods which require the surface reflectance function as a known input [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. We illustrate how these algorithms can be chained together in figure 1.8.

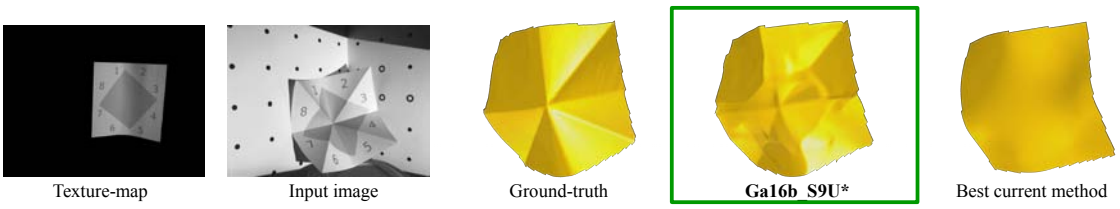


Figure 1.6: Visual results of our solution to the *SfTS-1* problem for creasable and poorly-textured surfaces and the best current method. It shows a creased and poorly-textured surface reconstructed by our method, denoted **Ga16b_S9U***, and the best current method [Chhatkuli et al., 2017]. Unlike our method, [Chhatkuli et al., 2017] fails to reconstruct the creases on the poorly-textured regions.

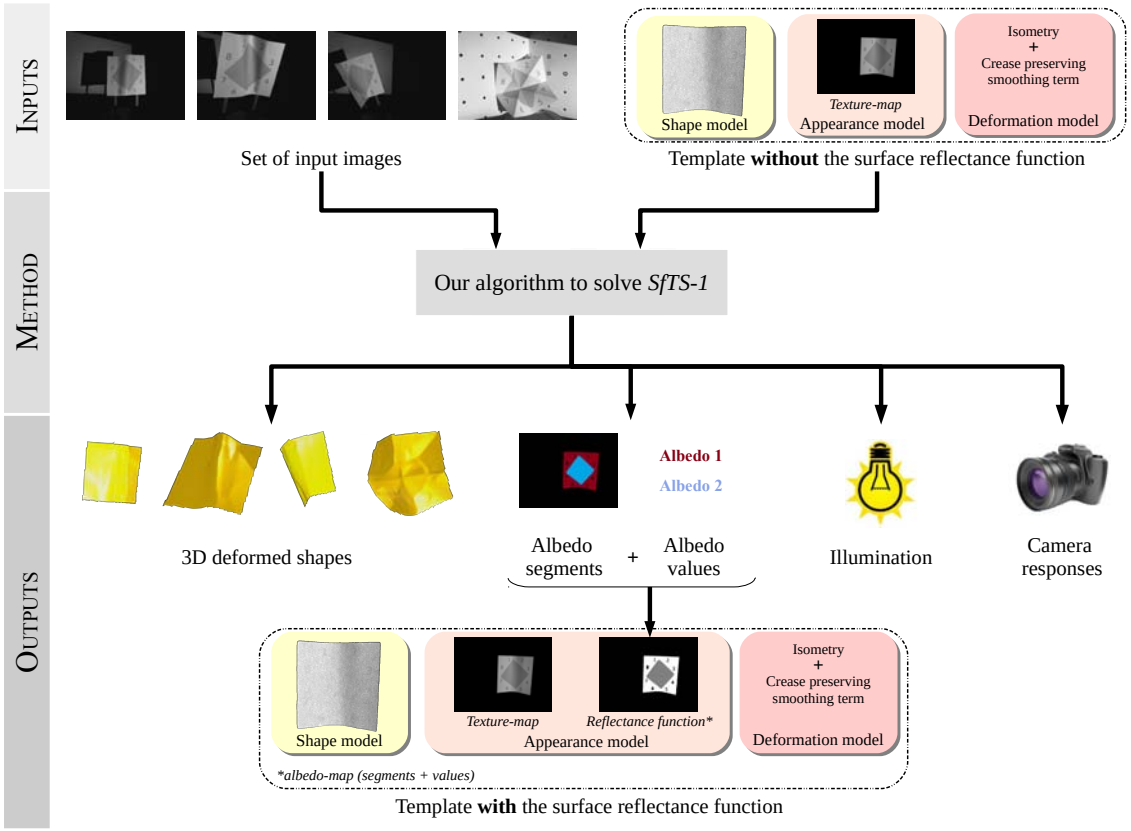


Figure 1.7: Illustration of our solution to the *SfTS-1* problem for creasable and poorly-textured surfaces. The inputs of our method are a set of four or more input images and a template without a surface reflectance function. The outputs are the 3D shape of the deformed object for each of the input images, the albedo-map, the illumination and the camera responses. Under the Lambertian assumption, our method gives an estimation of the surface reflectance function thanks to the segmentation of the texture-map into piecewise-constant albedo regions and the estimation of albedo values. Our method also allows us to integrate the surface reflectance function into the template.

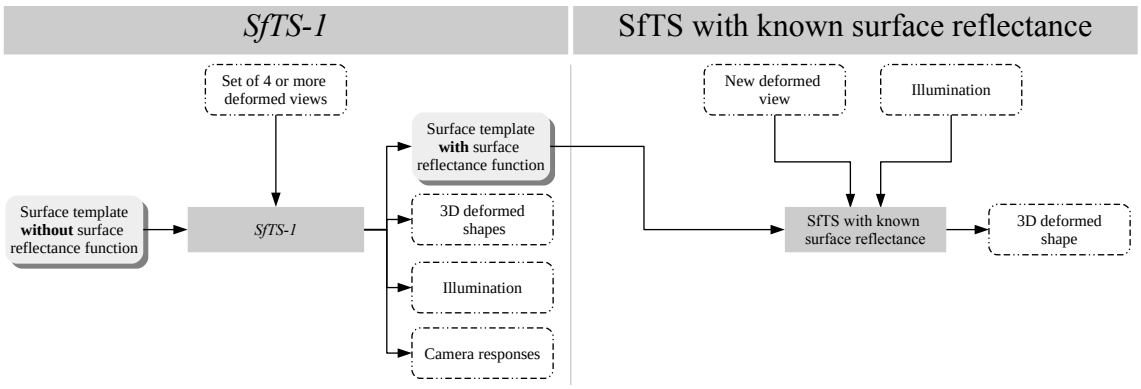


Figure 1.8: Connection between *SfTS-1* and SfTS with known surface reflectance.

Modeling the problem as joint deformation estimation and photometric calibration, using three complementary visual cues. We propose a novel, fully-integrated approach to solve *SfTS-1* for creasable and poorly-textured surfaces. It uses the adaptive

smoothing constraint proposed in our solution to SfT for creasable surfaces and it is designed to combine all the advantages of SfS and SfT. As with SfT, we use the template to provide strong physical constraints on the problem. As with SfS we use shading constraints to reveal the complex deformations. The use of the adaptive smoothing constraint is extremely important in order to allow one to use the full power of shading. The problem is solved by optimizing the deformation using these visual cues and physical deformation constraints, whilst jointly performing photometric auto-calibration, required to use shading. Our approach shows that it is possible to jointly reconstruct a surface with complex, non-smooth deformations at all visible regions (both textured and textureless) and to estimate the reflectance function of the surface (the albedos), without any *a priori* photometric calibration. Unlike previous approaches, ours works with generic object templates from a CAD database or a scanned model, and does not require a rigid observation video.

Efficient inference with cascaded initialization. The inference problem is large-scale ($\mathcal{O}(10^4)$ unknowns) and non-convex. We propose to solve it through an efficient initialization and refinement process. The initialization process works by sequentially estimating deformation, illumination, camera responses and then the albedo-map using at least four images. We solve the refinement problem by an iterative minimization, where the energy function consists of three data terms (for shading, motion and boundary contour constraints), and prior terms for isometry and smoothing constraints.

1.4.4 Contribution to Surface NRSfM: Creasable and Poorly-Textured Surfaces

NRSfM methods are limited to using only motion constraints (sparse or dense). Similarly to SfT, this limits the ability to reconstruct creased and well-textured surfaces, as figure 1.9 shows. We propose to overcome this by introducing shading constraints and a deformation model able to reconstruct creases. We refer to this problem as *NRSfMS-1* and give a complete definition in table 6.2.2. For the same reasons as in *SfTS-1*, we propose to recover simultaneously the 3D shape of the surface on each input image and its reflectance function. However, *NRSfMS-1* is a more difficult problem than the *SfTS-1* problem because no template of the surface is available. This is why we restrict our study to the case where the scene illumination and the camera responses are known. In figure 1.10 we show our general process and in figure 1.9 we show a rendering of the 3D shape obtained by our method. The *NRSfMS-1* problem *has never been solved before in the literature* and is a crucial missing component for densely reconstructing images in unconstrained settings.

Similarly to *SfTS-1*, figure 1.11 shows that the *NRSfMS-1* problem can be linked to SfTS with known surface reflectance. From a set of input images, one can construct a template of the surface and upgrade it with its surface reflectance function by solving *NRSfMS-1*. Then, this template can be used to recover the shapes of the same surface visible in a new image or video captured under different acquisition conditions.

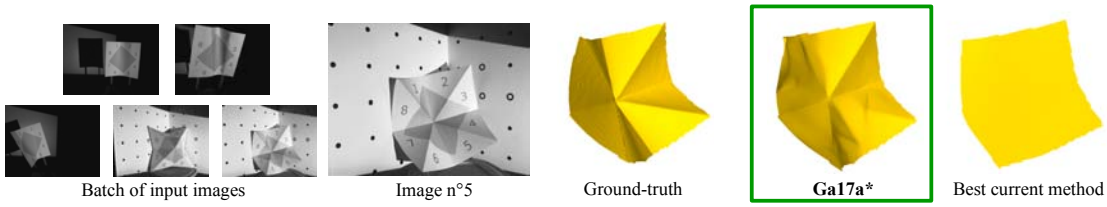


Figure 1.9: Visual results of our solution to the *NRSfMS-1* problem for creasable and poorly-textured surfaces and the best current method. It shows a creased and poorly-textured surface reconstructed by our method, denoted **Ga17a***, and the best current method [Parashar et al., 2016]. Unlike our method, [Parashar et al., 2016] fails to reconstruct the creases.

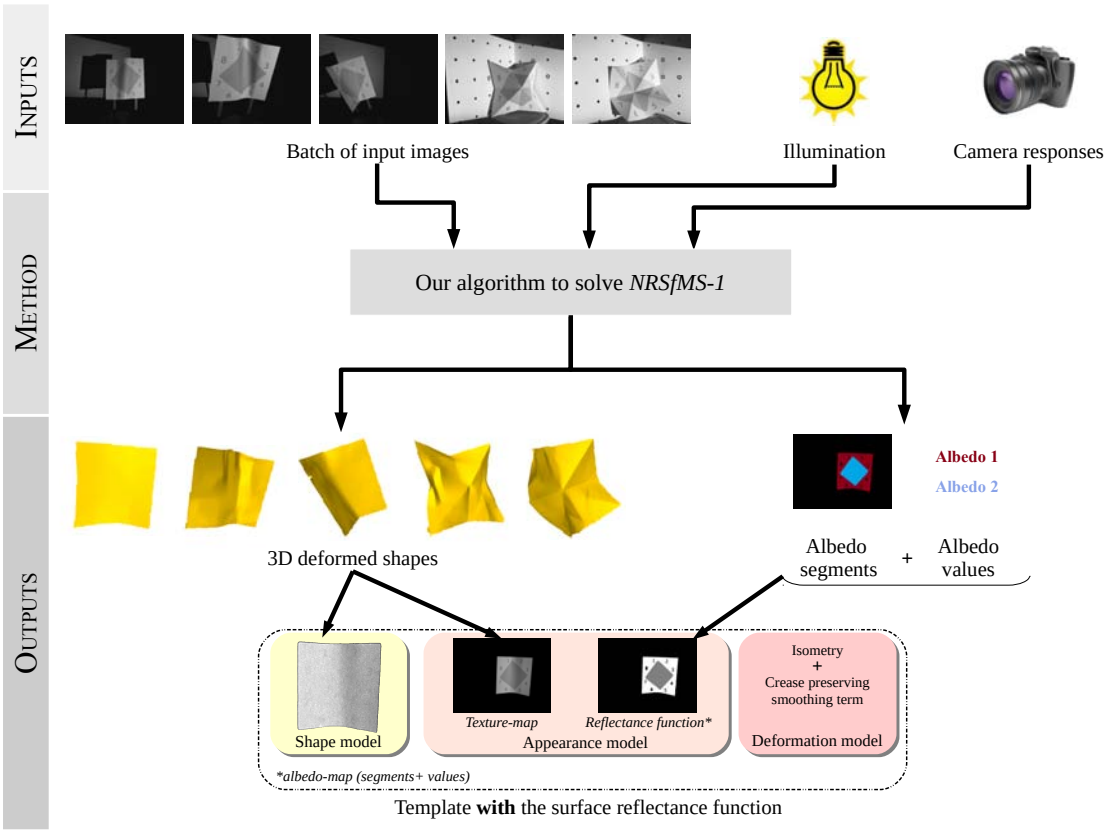


Figure 1.10: Illustration of our solution to the *NRSfMS-1* problem for creasable and poorly-textured surfaces. The inputs of our method are a set of input images, the illumination and the camera responses. The outputs are the 3D shape of the deformed object for each of the input images and the albedo-map. Under the Lambertian assumption, the reflectance function is defined by an albedo-map. Our method allows to build a template with a surface reflectance function, which can be used in other applications [Liu-Yin et al., 2016; Malti and Bartoli, 2014].

We propose in this thesis a proof-of-concept that NRSfM and shading can be combined to reconstruct densely any generic surfaces (poorly and well-textured surfaces) under smooth or non-smooth isometric deformations. Our main contribution with respect to *NRSfMS-1* is an initialization strategy of high non-parametric meshes and albedos, followed by a refinement of multiple image data constraints (shading, feature correspondences and boundary contour)

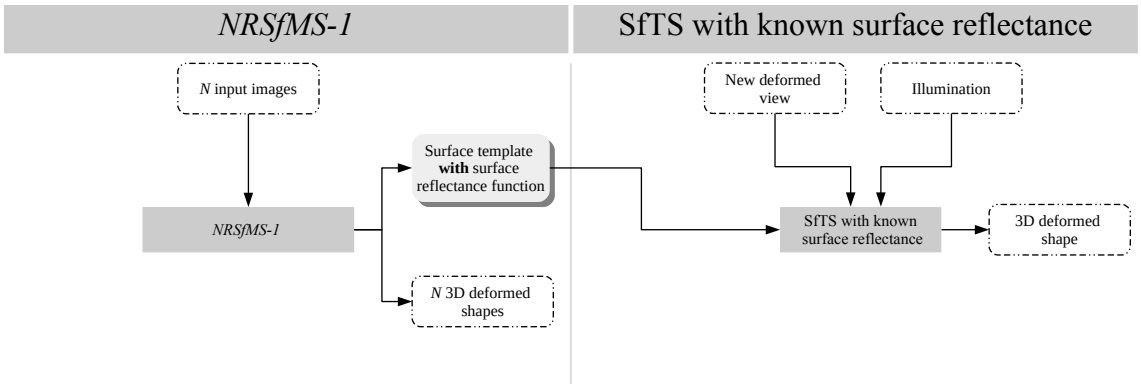


Figure 1.11: Connection between *NRSfMS-1* and SfTS with known surface reflectance.

and two deformation priors (isometry and adaptive smoothing). Because this is the first approach to solve *NRSfMS-1*, we also include an empirical analysis of the problem's stability using perturbation analysis. We provide real experiments with ground-truth data and show that our method can accurately register and reconstruct dense surfaces where state-of-the-art NRSfM methods fail.

1.5 Thesis Outline

We organize the thesis in 7 chapters. Chapter 2 presents a state-of-the-art of SfT and NRSfM. Chapter 3 presents our study of Curve SfT, with geometric problem modeling, numerical solutions, theoretical analysis and experimental validation. Chapters 4, 5 and 6 present our approaches to solve respectively *SfT-1*, *SfTS-1* and *NRSfMS-1*, which can be considered as the extension of *SfTS-1* to the case where the template geometry is unknown *a priori*. We give the geometric problem modeling and the photometric one specifically for chapters 5 and 6. For these three chapters, we also give numerical solutions and experimental validation. Chapter 7 presents our conclusions and perspectives for future work.

1.6 Notation and Nomenclature

1.6.1 General Notation

We recall from §1.2.6 that we use *italic* to denote a particular instance of SfT or NRSfM, such as *Curve SfT-1* and *NRSfMS-1*. We use bold to denote methods to solve a particular instance, such as **Ga16a*** and **Ga17a***.

1.6.2 Mathematical Notation

We use bold fonts for vectors and consider them by default as column vectors. The transpose of the vector $\mathbf{x}$ and the matrix A are respectively denoted $\mathbf{x}^\top$ and $A^\top$. By default, we denote

x_i the i^{th} component of the vector $\mathbf{x}$. We use hats for estimates. Homogeneous coordinates are written with a bar, for instance $\bar{\mathbf{q}} = \begin{pmatrix}\mathbf{q}^\top & 1\end{pmatrix}^\top$.

The first and second derivatives of a scalar function $\varphi : \mathbb{R}^n \to \mathbb{R}$ are written with primes (*e.g.* φ') and double-primes (*e.g.* φ''). For a vector-valued function $\varphi : \mathbb{R}^n \to \mathbb{R}^m$, we use the Jacobian matrix denoted $\mathbf{J}_\varphi$. For the special case where $\varphi : \mathbb{R} \to \mathbb{R}^m$, we use the Hessian matrix denoted $\mathbf{H}_\varphi$.

1.6.3 Nomenclature

1.6.3.1 Shape Nomenclature

Notation	Description
$\mathcal{T}$	An RGB texture-map image of the template, $\mathcal{T}(\mathbf{u}) : \mathbb{R}^2 \to \{0, 255\}^3$
$\Omega_\mathcal{T}, \Omega$	The segmented region of the object of interest in the reference image. For SfT methods, we denote it as $\Omega_\mathcal{T} \subset \mathbb{R}^2$. For NRSfM methods, we denote it as $\Omega \subset \mathbb{R}^2$
$\mathbf{u}$	The 2D position of a point on the segmented region of reference image, $\mathbf{u} \in \Omega_\mathcal{T}$ or $\mathbf{u} \in \Omega$
M	The number of mesh vertices, $M \in \mathbb{N}^+$
F	The number of mesh faces, $F \in \mathbb{N}^+$
N_E	The number of mesh edges, $N_E \in \mathbb{N}^+$
$\mathbf{u}_i$	The 2D position of the i^{th} mesh vertex, $\mathbf{u}_i \in \Omega_\mathcal{T}$ or $\mathbf{u}_i \in \Omega$
$\mathbf{y}_i$	The i^{th} template mesh vertex in 3D camera coordinates, $\mathbf{y}_i \in \mathbb{R}^3$
$\mathcal{Y}$	The vertices of the template, $\mathcal{Y} = \{\mathbf{y}_i\}_{i\in[1,M]} \in \mathbb{R}^{3\times M}$
$\mathbf{v}_t^i$	The i^{th} mesh vertex in 3D camera coordinates, $\mathbf{v}_t^i \in \mathbb{R}^3$
$\mathcal{V}_t$	The vertices in 3D camera coordinates for image t, $\mathcal{V}_t = \{\mathbf{v}_t^i\}_{i\in[1,M]}$
$\mathcal{F}$	The set of mesh faces $\mathcal{F} \triangleq \{f_1, ..., f_F\}$, $f_k \in [1, M]^3$
E	The set of mesh edges, $E \in [1, M]^{2\times N_E}$
φ	The transformation function of any point $\mathbf{u}$ to 3D camera coordinates using $\mathcal{V}_t$, $\varphi(\mathbf{u}; \mathcal{V}_t) : \mathbb{R}^{3\times M} \to \mathbb{R}^3$. This is done using a barycentric interpolation, a linear interpolation of the positions of the three vertices surrounding $\mathbf{u}$
n	The unit normal function which uses $\mathcal{V}_t$ and gives the unit surface normal at any point $\mathbf{u}$, $n(\mathbf{u}; \mathcal{V}_t) : \mathbb{R}^{3\times M} \to \mathbb{S}_3$, with $\mathbb{S}_3$ the unit sphere

1.6.3.2 Geometric Nomenclature

Notation	Description
N	The number of images, $N \in \mathbb{R}^+$
t	The image index, $t \in [1, N]$
Π	The perspective projection function, which can be a 1D or a 2D projection, $\Pi : \mathbb{R}^2 \rightarrow \mathbb{R}$ or $\Pi : \mathbb{R}^3 \rightarrow \mathbb{R}^2$
$\Pi_{\mathcal{T}}$	The perspective projection function of the reference image's camera for SfT methods, $\Pi_{\mathcal{T}} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$
$\mathbf{p}$	The 2D position of a point on image different from the reference image, $\mathbf{p} \in \mathbb{R}^2$
s (SfT)	The number of correspondences between an input image with a deformed object and the texture-map of the object template
$\mathcal{S}_t$ (SfT)	A set of matched putative 2D correspondences from the texture-map of the template to the t^{th} input image, defined by $\mathcal{S}_t = \{(\mathbf{u}_j, \mathbf{p}_t^j)\}$
s_t (NRSfM)	The number of correspondence between the reference image and the t^{th} input image
$\mathcal{S}_t$ (NRSfM)	A set of matched putative 2D correspondences respectively from the reference image to all other images, defined by $\mathcal{S}_t = \{(\mathbf{u}_j, \mathbf{p}_j)\}$ for NRSfM
$N_{\mathcal{B}}$	The number of boundary pixels, $N_{\mathcal{B}} \in \mathbb{R}^+$
$\mathcal{B}$	A set of boundary points defined on the reference image, $\mathcal{B} = \{\mathbf{u}_k\}_{k \in [1, N_{\mathcal{B}}]}$
B_t	A boundariness map of the t^{th} image, $B_t : \mathbb{R}^2 \rightarrow \mathbb{R}^+$

1.6.3.3 Photometric Nomenclature

Notation	Description
I	An RGB image, $I : \mathbb{R}^2 \to \{0, 255\}^3$
L	An intensity image, $L : \mathbb{R}^2 \to \mathbb{R}^+$, which is computed by projecting an RGB image I in the CIE XYZ color space or calibrating radiometrically the camera
R	An irradiance image, $R : \mathbb{R}^2 \to \mathbb{R}^+$
g_t	The camera response function of the t^{th} image, which transforms the irradiance image R_t to an intensity image L_t
$\mathbf{l}$	The illumination vector of the scene, $\mathbf{l} \in \mathbb{R}^{4 \text{ or } 9}$
α	An albedo value, $\alpha \in \mathbb{R}^+$
K	The number of albedos under the assumptions of piecewise-constant surface albedo, $K \in \mathbb{N}^+$
Ω_A	The union of non-textured regions, $\Omega_A \subset \Omega_{\mathcal{T}}$
$\mathcal{A}$	The set of albedos under the assumptions of piecewise-constant surface albedo, $\mathcal{A} = \{\alpha_1, \ldots, \alpha_K\}$
A	The albedo-map that gives the albedo value for a pixel $A(\mathbf{u}) : \Omega_A \to \mathcal{A} = \{\alpha_1, ..., \alpha_K\}$
β_t	The camera response of the t^{th} image under the assumption of a linear general camera response, $\beta_t \in \mathbb{R}^+$

Chapter 2

Background and Previous Work

Summary

This chapter starts by giving the common components which characterize any particular instance of the general 3D reconstruction problem. It continues with the state-of-the-art of SfT and NRSfM by considering five fundamental components: shape modeling, deformation modeling, visual cues, the inference process and global versus sequential methods. We point out the lack of work on SfT for reconstructing curves. We show that current SfT and NRSfM methods provide good performance only for well-textured surfaces that deform smoothly. As nearly all SfT and NRSfM methods use motion as the main visual cue, we give a brief overview of feature-based matching and optical flow techniques, which are used to extract motion information. This chapter ends with a review of 3D reconstruction using shading, including works which integrate shading with SfT. Shading has never been used with NRSfM before.

2.1 Instantiating the General 3D Reconstruction Problem

There exist many possible variations of problems which, at their core, involve reconstructing deformable objects from image data. The previously mentioned problems, SfT, NRSfM and SfS, are actually three families of problems which can vary according to several important ways. However, these problems share in eight common general components. This characterization provides a formal description of any specific 3D reconstruction problem, and helps to link the problems and see how they are closely inter-related. We also make a clear distinction between specifying a particular problem according to these eight components and designing a specific method to solve the given problem. This helps decouple method-specific decisions from general characteristics of the reconstruction problem. We now discuss these eight general components.

(a) Models. A decision must be made about which entities in the scene should be modeled. At a minimum, the target object's surface should be modeled, and, in the case of SfT and NRSfM, its deformation space must be modeled. Because reconstruction is performed with images, a camera model is always required. In general a perspective camera model is used, but affine camera models have also been considered. Depending on the specific nature of the problem, other entities require modeling. For example in SfTS or SfS, surface reflectance and scene illumination models are also required. *(b) Exploited visual cues.* As mentioned in chapter 1, the main visual cue is motion in SfT and NRSfM, and shading in SfS. *(c) Number of required images.* NRSfM always uses a set of two or more images. SfT and SfS usually use a single image, but particular instances which use several images already exist. Some examples are given in §2.5. *(d) Expected types of deformations.* Most of problem instances generally assume no tearing and smooth deformations. Normally, deformation modeling is done to reflect the expected types of deformation. The reason is because these assumptions better constrain the problem and are easily met in practice. *(e) Scene geometry assumptions.* For example, in SfT and NRSfM, usual assumptions are that there are no self or external occlusions and there may be some background clutter. Often in SfT, another common assumption is that there is strong perspective. This is required to strongly constrain the depth of the object, caused by the divergence of optical viewing rays. *(f) Requirement for putative correspondences.* As explained in chapter 1, most SfT and NRSfM methods rely on motion constraints which can be computed *a priori* or simultaneously to the 3D reconstruction. *(g) Surface texture characteristics.* This component defines the distribution of texture over the surface. In SfT and NRSfM, the most common assumption is that the surface is well-textured. On the contrary, in SfS, the most common assumption is that the surface is untextured or, very rarely, poorly-textured. This comes from the fact that SfT and NRSfM generally use motion constraints and SfS uses shading. *(h) Known and unknown model parameters.* The 3D shape is the main unknown of all problem instances of 3D reconstruction. Nearly all problems assume the camera intrinsics to be known, but some particular instances estimate the camera intrinsics jointly with the 3D reconstruction. Some physical properties of the surface or the scene are required to be known in order to

solve particular instances. This is the case of SfS which assumes some physical properties of the surface and the scene illumination to be known. The distinction between which model parameters are known, and which are unknown can have significant impact on the problem's solvability.

All of these components must be specified in order to fully define a problem instance. Of course, many different combinations are possible and then the space of problems is extremely large. Indeed, despite considerable research over the past years, not all problem instances have been so far investigated or solved. For example, while NRSfM has been studied extensively with known correspondences, low-rank shape bases models and the perspective camera with known intrinsics, the problem of solving with unknown intrinsics has not been studied. The vast number of possible problem configurations motivates us to study both the important ones with general and useful applications, and also the ones which, fundamentally, lead to well-posed problems.

2.2 Shape-from-Template

SfT aims to register and reconstruct the 3D shape of a deforming object from a single input image and a template of the object, as figure 2.1 illustrates. Registration and reconstruction are solved by finding the deformation transformation that embeds the template in 3D camera coordinates. Various different cases of SfT exist, but all share the same components. We first describe the three possible SfT cases in terms of template geometry: Curve SfT, Surface SfT and Volume SfT. We then discuss in detail the four main components shared by all SfT methods: *(i)* the template specifications, *(ii)* the data constraints extracted from the input image, *(iii)* the 3D shape inference process and *(iv)* global versus sequential methods.

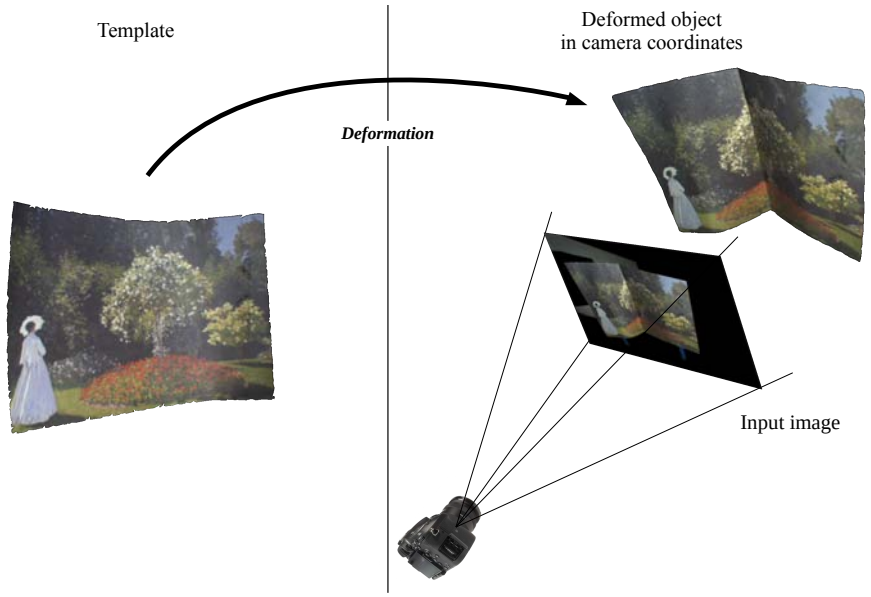


Figure 2.1: Illustration of the SfT problem. It uses the *Monet paper* dataset.

2.2.1 Curve, Surface and Volume SfT

We can categorize SfT methods according to several geometric criteria. The main three properties are as follows: *(a)* the dimension x of the template manifold, *(b)* the dimension y of the Euclidean space $\mathbb{R}^y$ that embeds the template manifold in camera coordinates with $y \geq x$, and *(c)* the dimension z of the Euclidean camera image with $z \leq y$. The primary goal of SfT is then to determine the embedding function $\varphi : \mathbb{R}^x \rightarrow \mathbb{R}^y$ that embeds the shape template in $\mathbb{R}^y$. We denote a particular SfT case with $\text{SfT}^{x \rightarrow y \rightarrow z}$. Table 2.1 summarizes the important cases. Surface and Volume SfT (rows 1 and 2) have been studied significantly in the literature, however Curve SfT (rows 3, 4 and 5) have not.

SfT case	Observational data	Template dimension	Unknown embedding	References
$\text{SfT}^{3\rightarrow3\rightarrow2}$	2D region in 2D input image	3D	3D $\rightarrow$ 3D	[Parashar et al., 2015] [Collins et al., 2016]
$\text{SfT}^{2\rightarrow3\rightarrow2}$	2D region in 2D input image	2D	2D $\rightarrow$ 3D	[Salzmann and Fua, 2009] [Bartoli et al., 2015]
$\text{SfT}^{1\rightarrow3\rightarrow2}$	2D curve in 2D input image	1D	1D $\rightarrow$ 3D	$\times$
$\text{SfT}^{1\rightarrow2\rightarrow2}$	2D curve in 2D input image	1D	1D $\rightarrow$ 2D	$\times$
$\text{SfT}^{1\rightarrow2\rightarrow1}$	1D straight line in 1D input image	1D	1D $\rightarrow$ 2D	$\times$

Table 2.1: Taxonomy of SfT cases in terms of template geometry. $\text{SfT}^{x \rightarrow y \rightarrow z}$ denotes a case with a template manifold of dimension x, a Euclidean embedding space with dimension y and a camera with image dimension z.

The usual case of SfT is Surface SfT, where the shape is a 2D surface embedded in 3D, and the input image is a 2D perspective projection. Volume SfT is where the shape is a 3D object embedded in 3D, and the input image is a 2D projection. This has been recently studied [Collins et al., 2016; Parashar et al., 2015]. We discuss various ways surface and volume templates have been modeled in the next section. Curve SfT comprises three new cases where the template is curvilinear, the shape is a 1D curve. The first and second cases, $\text{SfT}^{1\rightarrow3\rightarrow2}$ and $\text{SfT}^{1\rightarrow2\rightarrow1}$, were introduced in §1.4.1. The second case, $\text{SfT}^{1\rightarrow2\rightarrow2}$, is when the template is embedded in 2D space and observed by a 2D camera. A practical example is a thin rope lying on a ground plane and observed by a 2D camera. $\text{SfT}^{1\rightarrow2\rightarrow2}$ is very different from the other two. It is actually simpler, because shape is deformed on a ground plane. The problem requires determining only the camera's 6 Degree-of-Freedom (DoF) pose relative to the ground plane. By contrast, the other two cases of Curve SfT involve deformable reconstruction. As table 2.1 points out, no solution to Curve SfT has been proposed in the literature. We will expose in chapter 3 that, despite Curve SfT seeming to be a simpler SfT problem, Curve SfT is fundamentally different in terms of theory and requires one to develop new computational solutions. For the sake of clarity and for the following part of chapter 2, we refer to Surface SfT and Volume SfT by SfT.

2.2.2 Template Components

The template is the cornerstone of SfT. It brings strong object-specific prior knowledge to the problem. Several types of template exist, but all of them comprise three components: a *shape model*, an *appearance model* and a *deformation model*. We illustrate these in figure 2.2 with a template.

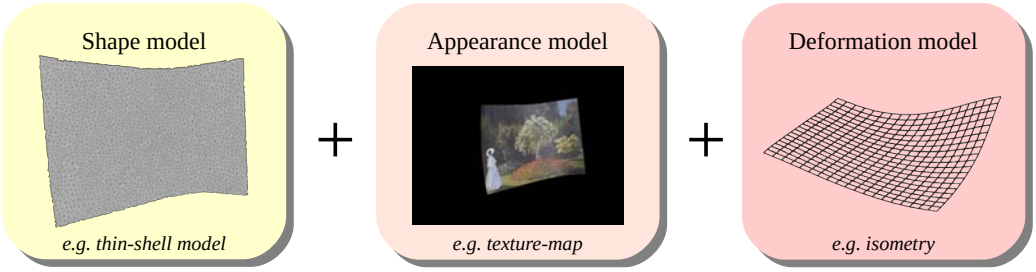


Figure 2.2: The three components of the template. It uses the *Monet paper* template.

2.2.2.1 Shape Model

The template's shape model represents the object's 3D shape in a fixed reference position. The shape model can be acquired with various ways depending on the application, including SfM methods such as [Agisoft, 2014; Wu, 2011], or structured-light methods such as [David 3D Scanner, 2014], or from a 3D CAD model database such as [TurboSquid, 2016; Warehouse, 2016]. There are two main types of shape models. The first use surface templates [Bartoli et al., 2015; Brunet et al., 2014; Collins and Bartoli, 2014; Ngo et al., 2016; Salzmann and Fua, 2011], where only the object's surface is modeled. The second are with volume templates [Parashar et al., 2015], where the object's surface and interior volume are modeled. Surface templates are the most common and give good approximations for thin or hollow surfaces made for example of paper, cloth and plastic. Surface templates have varied in complexity. The earliest model used algebraic models such as smooth B-splines [Brunet et al., 2014] or thin-plate splines [Bartoli et al., 2015; Chhatkuli et al., 2017]. Most recent models include triangulated meshes, which are conceptually simple, can handle general topologies [Collins and Bartoli, 2015; Ngo et al., 2015, 2016; Salzmann and Fua, 2009; Yu et al., 2015], and work for surface and volume templates. An important point of mesh models is the trade-off between the density of the model and the computational time for shape inference.

2.2.2.2 Appearance Model

The appearance model is used to describe the photometric appearance of the object. In nearly all cases of Surface and Volume SfT, this is done using a texture-map [Bartoli et al., 2015; Chhatkuli et al., 2017; Collins and Bartoli, 2015; Ngo et al., 2016; Salzmann and Fua, 2011]. A texture-map models the intensity or color of each surface point up to photometric

transforms caused by illumination, shading variations, and other photometric factors. In most previous works, the texture-map is generated from one or more images of the object in its reference position [Collins et al., 2014], but it can also come from a CAD model [Collins and Bartoli, 2015].

Texture-map models do not physically model surface reflectance. If the texture-map is generated from images, then it is formed from a complex process that mixes the camera response, surface reflectance, illumination, and image blending (to merge texture from different images). If the texture-map comes from a CAD model, then it does not usually consider the physical aspects that alter reflectance such as surface roughness. Texture-map models are however sufficient for SfT methods that use motion as the main visual cue, because motion can be estimated without requiring a reflectance model. For example, feature-based SfT methods do not require a reflectance model because they use illumination-invariant texture matching methods such as SIFT, which can be computed directly from texture-maps and camera images.

2.2.2.3 Deformation Model

The deformation model is used to define the transformation of the template's reference shape and the space of possible deformations. For this, most methods use dimensionality reduction through smooth parameterizations. Several SfT methods use explicit smoothing priors and all methods use physical priors[1].

Smooth parameterizations. These have included thin-plate splines and B-splines, and reduce dimensionality by modeling deformation with a reduced set of control points. Thin-plate splines enforce global smooth deformations, and are not suitable to model surface creases. It is possible to model high-frequency and discontinuous deformations with B-splines by changing the spline's order and introducing repeated control points [Gregorski et al., 2000; He and Qin, 2004]. However, to correctly distribute the control points, one needs to know where the discontinuities are, which in SfT is not known *a priori*. For this reason, discontinuous B-spline models have not been used previously in SfT. For mesh-based shape models, deformation smoothness has been introduced through the mesh laplacian [Sorkine and Alexa, 2007]. In [Ngo et al., 2016], this was used both for smoothing and dimensionality reduction. The idea was to identify the smooth deformation modes by performing a modal analysis on the mesh laplacian. This is done by an eigen-decomposition of the mesh laplacian, where the eigenvectors with lowest eigenvalues are used to build a low-dimensional set of deformation bases. The advantage of doing this is that for smoothly deforming objects only a small number of bases may be required (*e.g.* less than 100), and this can significantly reduce the cost of optimization. However, the problem is that we lose the ability to model high-frequency deformations such as creases.

[1] which is different from statistics-based priors such as the morphable face models of [Blanz and Vetter, 1999]

Explicit smoothing priors. Other methods penalize non-smooth deformations explicitly with a smoothing term based on an ℓ_2 norm [Bartoli and Özgür, 2016; Brunet et al., 2014]. This norm strongly penalizes non-smooth deformations. This provides strong problem regularization, but can prevent the formation of discontinuities such as creases.

Physical priors. *The isometry prior.* Isometry and quasi-isometry are the most commonly used priors [Bartoli et al., 2015; Chhatkuli et al., 2017; Collins and Bartoli, 2014; Liu-Yin et al., 2016; Salzmann and Fua, 2011]. They enforce metric constraints by preventing deformations that locally stretch or shrink the object. Isometry means that the geodesic distance between two points on the surface is preserved by deformation. Isometry is also equivalent to saying that the surface's first fundamental form is preserved by deformation. Figure 2.3 illustrates an isometric deformation. Isometry can be imposed exactly, which means no stretching or shrinking is permitted. Isometry can also be imposed inexactly, meaning that there is non-negligible stretching or shearing, and the model penalizes solutions with increased stretching or shrinking using a penalty function. This is also called *quasi-isometry* in the literature. Isometry and quasi-isometry have been used extensively because they dramatically restrict the solution space, and are applicable for many object classes such as those made of thick rubber, tightly-woven fabrics, paper, cardboard and plastics, as figure 2.4 shows. The isometric prior is very powerful, and has been shown that if imposed exactly, then the problem can be solved uniquely [Bartoli et al., 2015]. The main difficulty with isometry is that it is a non-convex constraint.

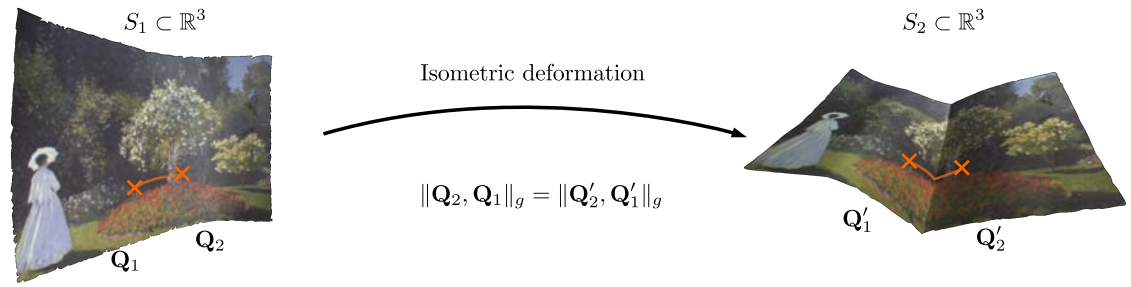


Figure 2.3: Example of an isometric deformation. It uses the *Monet paper* dataset. We denote the geodesic distance by $\|.\|_g$. Surfaces S_1 and S_2 are isometric if and only if $\|\mathbf{Q}_1, \mathbf{Q}_2\|_g = \|\mathbf{Q}'_1, \mathbf{Q}'_2\|_g$.

Figure 2.4: Examples of objects deforming quasi-isometrically. From left to right: newspaper, flag (credits: Zegreg63), cloth, back bag and boat sail (credits: Nocturally).

The inextensibility prior. This is a relaxation of the isometry prior. It prevents the

Euclidean distance between two neighboring surface points from exceeding their geodesic distance, defined on the template. The advantage of the inextensibility constraint is that it is a convex constraint. However, it is too weak to reconstruct geometry accurately and must be combined with additional constraints. This has been done previously using the so-called Maximum Depth Heuristic (MDH) [Perriollat et al., 2011; Salzmann and Fua, 2009], where a depth maximization constraint is imposed to prevent the reconstructed surface from shrinking arbitrarily. The MDH has been shown to produce very good reconstructions when the perspective effects of the camera are strong.

Other physical priors. Weaker physical priors have also been considered to handle objects that can stretch or shrink as they deform. Examples include the conformal prior (angle preservation) [Bartoli et al., 2015; Malti and Bartoli, 2014] or priors based on elasticity [Haouchine et al., 2014; Malti et al., 2013, 2015; Özgür and Bartoli, 2017]. The problem with using these weaker physical priors is that the SfT problem becomes less well-conditioned. For instance, SfT with the conformal prior is solvable up to a global scale factor and convex/concave ambiguities.

2.2.3 Data Constraints in SfT

Data constraints must be extracted from the input image in order to match the template's shape with the object's true shape. By far the most common are motion constraints. Other constraints include contour and shading.

2.2.3.1 Motion Constraints

Motion constraint can be broken down in two types: *correspondences constraints* and *direct constraints*.

Correspondences constraints. Correspondence constraints force 3D points on the template's surface to project at their corresponding 2D points in the input image [Bartoli et al., 2015; Brunet et al., 2014; Ngo et al., 2016; Salzmann and Fua, 2011]. The points used by these constraints are usually obtained by matching features from the template's texture-map and the input image. These constraints have been exploited in various ways: through zeroth-order correspondences [Brunet et al., 2014; Ngo et al., 2016; Salzmann and Fua, 2011], first-order correspondences [Bartoli et al., 2015] or second-order correspondences [Bartoli and Özgür, 2016]. A zeroth-order correspondence is used to constrain the position of a point on the template in camera coordinates. Usually this is implemented using the reprojection error of the correspondence. This however is a non-convex constraint. It is possible to construct a convex zeroth-order constraint by imposing it in camera coordinates. Zeroth-order correspondences are usually computed using feature-based matching, which we describe more in §2.4.1.

First-order correspondences require knowing both the position of the correspondence and the local affine transform about the correspondence. A first-order correspondence is used to constrain both the position of a point on the template in camera coordinates, and also

first-order properties of the deformation at that point. In practically all cases, the first-order properties relate to isometric deformation which states that the first fundamental form is preserved. We discuss this further in §2.2.4. First-order correspondence can be computed in two ways. The first is with a differentiable warp fitted between the template's texture-map and the input image. First-order correspondence can then be computed at any given point by differentiating the warp. The second way is by fitting a local first-order differentiable warp at each correspondence [Collins and Bartoli, 2014]. The main difficulty with first-order correspondences is the need to compute first-order warp derivatives, which is less numerically stable than computing zeroth-order correspondences. Second-order correspondences have also been recently considered [Bartoli and Özgür, 2016] to handle non-isometric deformations.

Correspondence constraints have three main limitations. First, they work well only for densely-textured objects with discriminative texture. This is not common in most real practical applications, particularly with man-made objects and many natural objects that usually have very weak texture. Second, feature-based matching methods may fail to establish correspondences without errors. Third, the computational time to extract features, compute descriptors and perform the matching can be long without high performance GPUs.

Direct constraints. Direct constraints work by maximizing the photometric agreement, *i.e.* brightness constancy, between the deformed template and the input image [Collins and Bartoli, 2015; Malti et al., 2011; Ngo et al., 2015; Yu et al., 2015]. The main advantage of direct constraints is to provide denser motion constraints than feature correspondences [Collins and Bartoli, 2015; Malti et al., 2011; Ngo et al., 2015; Yu et al., 2015].

There are three main limitations of direct constraints. First, they are highly non-convex and they require iterative optimization. Because of non-convexity, they are usually applied in a frame-to-frame tracking setup, as described in §2.2.5, and they require a good initial estimate. Second, direct constraints are sensitive to strong photometric changes which may be induced by complex deformations or complex illuminations. Third, direct constraints may require reasoning about surface visibility (they should be deactivated at surface regions that are occluded). This is non-trivial [Collins and Bartoli, 2015; Malti et al., 2011; Ngo et al., 2015]. [Malti et al., 2011] handles self-occlusions only on textured surfaces by estimating a visibility mask on the reconstructed surface. [Ngo et al., 2015] handles occlusions thanks to an M-estimator and [Collins and Bartoli, 2015] uses dense matches which handles them by construction.

2.2.3.2 Contour Constraints

Contour constraints force the object's occluding contours to align to the corresponding contours in the input image [Salzmann et al., 2007b; Vicente and Agapito, 2013]. The advantages of contour constraint are that they do not depend on the template's texture. They are therefore applicable for poorly-textured surfaces and even surfaces without texture. There exist two types of contour constraints: silhouette contour constraints [Vicente and Agapito, 2013] and boundary contour constraints [Salzmann et al., 2007b]. Silhouette contour constraints

work by forcing the template's silhouette to align with silhouette contours detected in the input image. These constraints can work for all templates. Boundary contour constraints are applicable for open surface templates such as a piece of paper. Boundary contour constraints are actually more like motion constraints since it is possible to compute correspondences along the boundary contour. Typically this is done by enforcing the boundary contour projects to an image edge. Similarly to direct constraints, contour constraints are highly non-convex, usually enforced iteratively and require a good initial estimate. The main challenge with using contour constraints is they are difficult to apply robustly, particularly with strong background clutter.

2.2.3.3 Shading Constraints

The shading constraint is based on the photometric relationship between surface geometry, surface reflectance, illumination, the camera response and pixel intensity. This relationship correspond to the image formation process, as figure 2.5 illustrates. Shading constraints are very attractive: they are dense constraints since they constrain the deformation at all visible regions and they permit to recover complex deformations in poorly-textured surfaces [Pentland, 1988]. However, shading constraints are difficult to use in practice since they require good photometric modeling and calibration. We give in §2.5.3 a more detailed review of SfT methods which use shading.

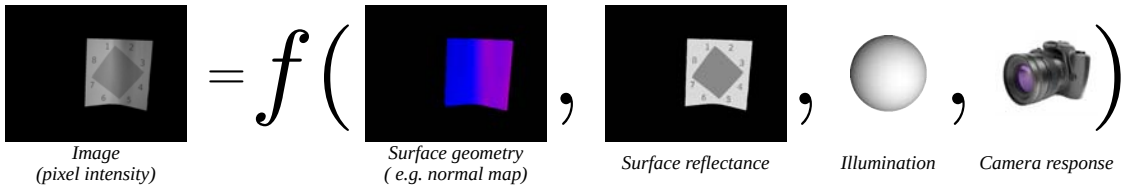


Figure 2.5: Illustration of the photometric relationship used by the shading constraint. It uses the *paper fortune teller* dataset.

2.2.4 Inference in SfT

Inference is performed by determining the deformation parameters that mutually satisfy the data constraints and deformation priors. For this, three inference categories have emerged. Figure 2.6 illustrates these three categories.

Category *(i)*: local analytical solutions using non-holonomic solution to PDE. Category *(i)* methods impose the constraints through a PDE system and solve using non-holonomic solutions. This approach have been used by [Bartoli et al., 2015; Chhatkuli et al., 2017] with the isometric prior and first-order correspondence constraints. Correspondence constraints were computed with a differentiable template-to-image warp which maps the template's texture-map to the input image. With the template-to-image warp, [Bartoli et al., 2015] constructs a first-order PDE system and solves it at each point assuming that the depth

and its gradient are independent. These solutions obtained are called non-holonomic solutions. [Bartoli et al., 2015] only uses the depth solution, however, it suffers from instabilities when the projection geometry tends to affine. Figure 2.6 gives a reconstruction performed by [Bartoli et al., 2015]. [Chhatkuli et al., 2017] solves this issue by improving the stability using the non-holonomic solution of both depth and gradient, which is proven to be stable for both perspective and affine projections. The main advantage of category *(i)* methods is that, as they give an analytical solution at each correspondence, they are fast and they can be parallelized extremely well. The solutions can be used as initial estimate for non-convex refinement which we describe below. The main disadvantage is the need for accurate first-order correspondences.

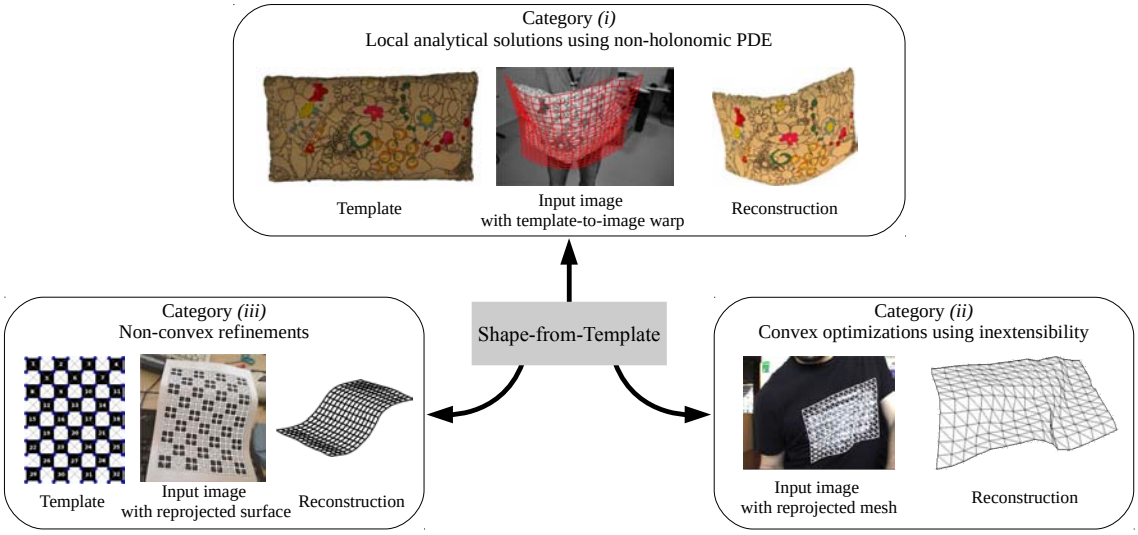


Figure 2.6: Illustration of the different categories of SfT methods. Results are taken from the following works (from category *(i)* to category *(iii)*): [Bartoli et al., 2015], [Salzmann and Fua, 2009] and [Brunet et al., 2014].

Category *(ii)*: convex optimizations using inextensibility. Category *(ii)* methods work by approximating the inference problem with a convex function. The main idea is to deal with a simpler constraint than isometry which is non-convex in order to obtain quasi-isometry. One solution is to relax isometry to inextensibility, which leads to a convex constraint easier to handle. However, inextensibility is insufficient since a trivial solution is given by putting all correspondences at the camera origin. To prevent this solution, the depths of the correspondences are maximized while simultaneously satisfying inextensibility and zeroth-order correspondence constraints. This technique, called MDH [Brunet et al., 2014; Perriollat et al., 2011; Salzmann and Fua, 2009], shows that it can provide solutions that are often quasi-isometric. Two versions of MDH have been proposed and they differ from the way they solve the problem. The first version uses a fast greedy technique [Perriollat et al., 2011]. The second version is based on the remark that the MDH can be formulated as a Second-Order Cone Programming (SOCP) problem [Brunet et al., 2014; Salzmann and

Fua, 2009], and therefore solves globally using efficient methods such as interior point.

MDH has been first proposed by [Perriollat et al., 2011]. It gives an upper bound for the depth on each point of the surface and refine each of them by taking for each point the minimum of neighboring upper bounds. For each point and for each pair of its neighboring points, it computes the upper bound using the inextensibility constraint. Then, for each point, it selects the minimum upper bound through an iterative procedure. A convex formulation of the MDH problem is given in [Salzmann and Fua, 2009]: it maximizes the sum of all the depths such that the inextensibility constraint and the reprojection of the 3D correspondence points are respected. This problem is efficiently and globally solved using SOCP. [Salzmann and Fua, 2009] solves this problem with a linear combination of modes for local patches, which restricts the recoverable deformations to the ones learned. A second shortcoming is that [Salzmann and Fua, 2009] gives unstable reconstruction for weakly perspective geometry. Figure 2.6 gives a reconstruction performed by [Salzmann and Fua, 2009]. Following the same formulation, [Brunet et al., 2014] proposes to reduce the number of unknowns and thus the computational time by representing the surface with B-splines. Despite the improvements of the original formulation [Brunet et al., 2014; Salzmann and Fua, 2009], category *(ii)* methods remain using a relaxation of the isometry, which sacrifices accuracy for obtaining a global solution.

Category *(iii)*: non-convex refinements. The third category works by combining the data and prior constraints into a single optimizable non-convex cost function. In general, because of the physical priors, this cost function is non-convex and thus is solved by gradient-based minimization such as Levenberg-Marquardt [Brunet et al., 2014; Liu-Yin et al., 2016] or Gauss-Newton [Collins and Bartoli, 2015; Ngo et al., 2015].

The solutions are holonomic, *i.e.* deformation and deformation derivatives are dependent, which ensures a better conditioning and stability. The advantages of this category are three-fold: there is no relaxation of the physical priors, complex constraints such as shading can be integrated without difficulty. However, category *(iii)* methods present two main challenges: they are non-convex, they may require more computational time than categories *(i)* and *(ii)* methods. They also require to find good weights of the different constraints. To ensure good convergence, they generally require a reasonably accurate initialization. This can be provided by a category *(i)* or *(ii)* method. Also, they may require some optimization techniques to improve the convergence, such as coarse-to-fine optimization [Collins and Bartoli, 2015]. Examples of category *(iii)* methods are [Brunet et al., 2014; Collins and Bartoli, 2015; Liu-Yin et al., 2016; Malti and Bartoli, 2014; Malti et al., 2011; Ngo et al., 2015; Yu et al., 2015].

A pixel-based cost function robust to self-occluded surfaces is proposed by [Malti et al., 2011]. It uses gradient-descent based optimization to minimize the cost function. This contains a brightness constancy term, an isometry term (which uses the first fundamental form), a first-order smoothing term and two geometric terms to handle self-occlusions. The initial estimate is computed using [Brunet et al., 2014].

[Malti and Bartoli, 2014] uses sequentially zeroth-order correspondence constraints from

SIFT matches and shading constraint through two successive non-convex minimizations. Both uses the same deformation priors, conformity and second-order smoothing constraints, and all constraints are solved using non-convex minimization. The initial solution is obtained by computing a rigid transform using the correspondences between the template and the first video frame.

[Brunet et al., 2014] uses zeroth-order correspondence constraints from SIFT matches, quasi-isometry and the B-spline bending energy to constrain the problem. It optimizes these constraints using Levenberg-Marquardt. The initial estimate is computed by a category *(ii)* method also proposed by [Brunet et al., 2014]. Figure 2.6 gives a reconstruction performed by [Brunet et al., 2014].

[Collins and Bartoli, 2015] uses a direct constraint based specially-designed dense correspondences called Deformable Render-based Block Matching (DRBM), and constrains quasi-isometry and change of surface's curvature. It uses Gauss-Newton to minimize these three constraints and improves the convergence and thus the computation speed using a coarse-to-fine approach with multi-grid optimization. [Collins and Bartoli, 2015] initializes the first input image through a manual registration of the template in the input image and then uses the previous frame to initialize the new frames.

To handle poorly-textured and occlusions, [Ngo et al., 2015] uses a direct constraint, a quasi-isometry constraint and a second-order smoothing constraint. The direct constraint is based on pixel-based image features which are robust to affine changes of lighting. Gauss-Newton optimization is performed to minimize all constraints. To improve convergence, [Ngo et al., 2015] also proposes two more strategies for the direct constraint: it performs the minimization using a coarse-to-fine approach and integrates a relevancy score to weight the influence of each pixel on the direct constraint. The templates used in [Ngo et al., 2015] are pre-registered to the first input image and then the final reconstruction of the previous frame is used to initialize the new frame.

[Yu et al., 2015] uses a direct constraint, a quasi-isometry constraint, a surface smoothing constraint and a temporal smoothing constraint. All constraints are solved using Levenberg-Marquardt and, similarly to [Ngo et al., 2015], it employs a coarse-to-fine approach to improve convergence, which can be difficult with a direct constraint. The initialization is similar to the one of [Ngo et al., 2015]. To reconstruct poorly-textured surfaces and fine details such as wrinkles, [Liu-Yin et al., 2016] replaces the direct constraint of [Yu et al., 2015] by a shading constraint.

2.2.5 Solving SfT with Single Images or Video Sequences

Another important criterion to differentiate SfT methods is whether they *(i)* work on single images without *a priori* initialization [Bartoli et al., 2015; Brunet et al., 2014; Salzmann and Fua, 2011], or *(ii)* work on video sequences, where initialization is provided using temporal continuity [Collins and Bartoli, 2015; Collins et al., 2016; Liu-Yin et al., 2016; Ngo et al., 2016; Yu et al., 2015]. All methods of the category *(i)* can be used to process videos, but not

vice-versa. Advantages of strategy *(i)* are that the performance does not depend on successful solution in the previous frame and it can handle sudden changes and full occlusions trivially without needing to adapt. One important advantage of strategy *(ii)* is that the problem is more constrained than strategy *(i)* because of temporal continuity. The strategy *(ii)* may then provide more accurate reconstructions since it optimizes an initial solution. However, this can turn as a shortcoming. The solution can be stuck in a local minimum if the solution from the previous frame is wrong, because of tracking loss which may happen in case of sudden illumination changes or occlusions. Strategy *(ii)* can also increase the modeling complexity because a temporal model of deformation is required.

2.3 Non-Rigid Structure-from-Motion

The goal of NRSfM is to recover a deformable object's shape from a set of unorganized images or a video. The main difference between SfT and NRSfM is that in NRSfM the object's template is not provided *a priori*, and this makes it a considerably harder problem. In NRSfM, we do not assume the object is rigid in any of the images. There exist a substantial number of approaches to solve NRSfM. The methods can be characterized by five components: *(i)* the deformation priors, *(ii)* the visual cues, *(iii)* the inference method, *(iv)* the camera model *(v)* the input image type, unorganized or sequential.

There is no general consensus on the best way to formulate NRSfM according to the above components. We organize our review by categorizing methods primarily along components *(i)*, *(ii)* and *(iii)*.

2.3.1 Deformation Priors

To overcome measurement noise and ambiguities in NRSfM, three classes of deformation priors have emerged: statistical priors, physics-based priors and temporal smoothness priors.

2.3.1.1 Statistical Priors

Statistical priors have been formulated in two main ways: low-ranks shape bases and low-rank trajectory bases. Figure 2.7 illustrates a work of each of these approaches.

Low-rank shape bases. The idea is to constrain an object's shape to lie in a linear space spanned by a small number of unknown 3D shape bases [Bregler et al., 2000]. The object's shape is defined for each image by a vector of basis weights. The goal is to estimate these jointly with the shape bases and camera poses. The low-rank shape prior operates on a matrix of stacked 2D point correspondences, also named the observation matrix. Nearly all low-rank methods assume an orthographic camera, and this allows the observation matrix to be factorized into orthographic projection, the shape bases, and the shape coefficients (which represent a particular shape for a given image). It is possible to factorize the observation

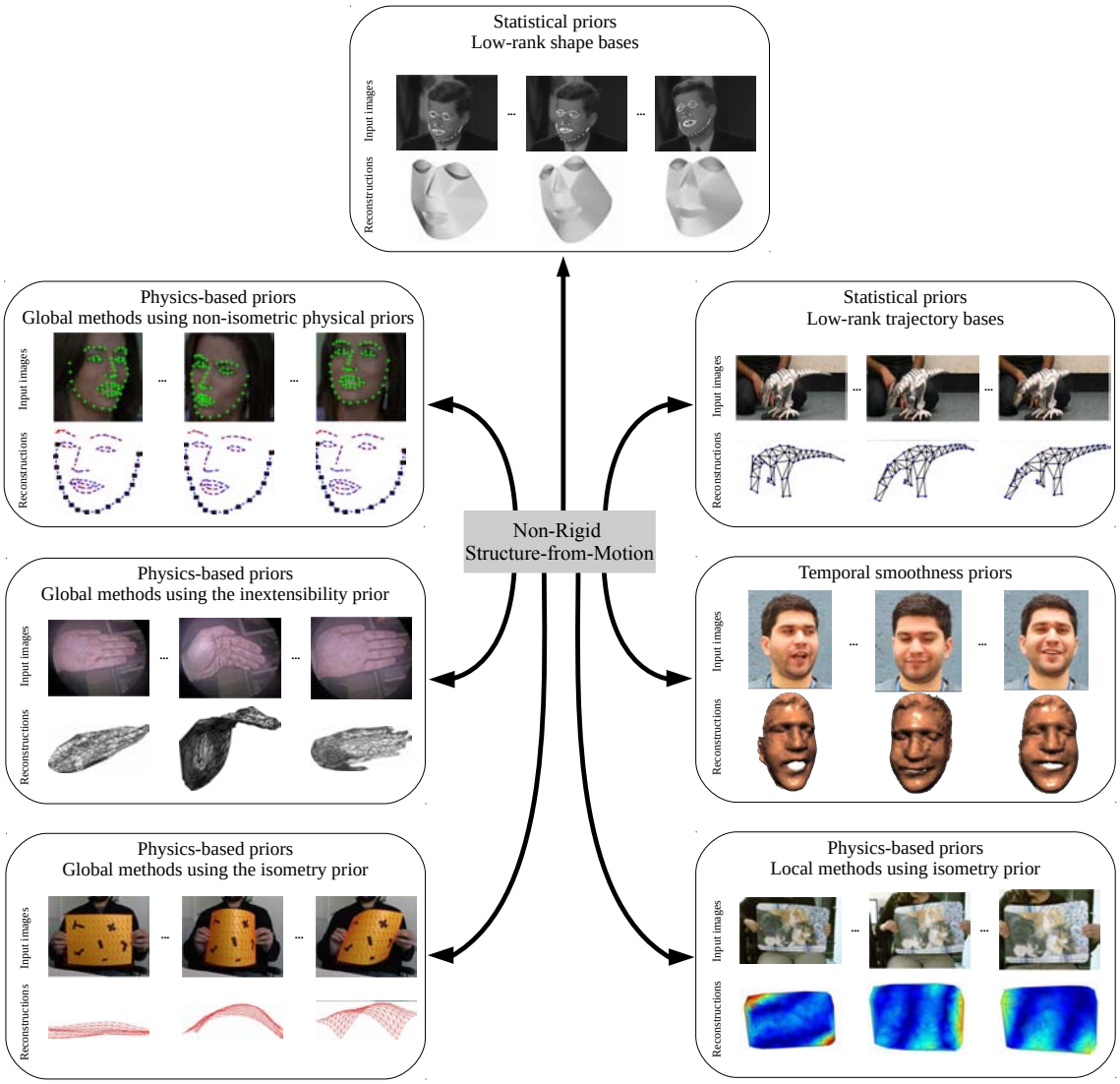


Figure 2.7: Illustration of the different deformation priors used in NRSfM. Results are taken from the following works (clockwise from top): [Bregler et al., 2000], [Akhter et al., 2009], [Garg et al., 2013], [Parashar et al., 2016], [Wang et al., 2016], [Chhatkuli et al., 2016] and [Agudo et al., 2016].

matrix based on the Singular Value Decomposition (SVD). Some recent methods also work with the perspective camera [Hartley and Vidal, 2008].

Despite the apparent simplicity, there are many inherent challenges to this approach. The original formulation of [Bregler et al., 2000] suffers from non-uniqueness of the shape factorization [Xiao et al., 2004]. This motivates the use of additional priors such as spatial and/or temporal smoothness [Akhter et al., 2009; Olsen and Bartoli, 2008; Torresani et al., 2001] or pre-computed shape bases [Del Bue, 2008]. These additional priors are integrated in an energy minimization scheme. Handling missing observations is also a difficult part of NRSfM methods using shape bases, because they cause holes in the observation matrix that prevent the observation matrix being decomposable in closed-form. Two main strategies have

been proposed: filling-in the observation matrix using prediction [Olsen and Bartoli, 2008] or iterative factorizations [Del Bue, 2008]. Another big challenge is the choice of the number of bases, because most of NRSfM methods assume this number to be known. Having a high number of shape bases can cause degeneracy problems and a small number can be insufficient to represent reasonably the shape of an object. For this, [Garg et al., 2013] proposes an automatic selection of the number of shape bases, but this is still an open challenge.

These approaches give good results for objects with a strong rigid component, such as human faces. However, they often require a large number of images and are not suitable for objects with high deformation spaces such as creasing fabric or objects that deform in unexpected ways.

Low-rank trajectory bases. The low-rank prior has also been used in the modeling of point trajectories [Akhter et al., 2009; Dai et al., 2014; Gotardo and Martinez, 2011]. The NRSfM problem is presented as recovering the trajectory of each correspondence in 3D space over time. The trajectory is assumed to lie on a linear subspace of trajectory bases. The shape in each input image is obtained trivially from the correspondence trajectories. Similarly to low-rank shape bases, the SVD can be used to recover the trajectory bases, their coefficients and the pose for each image if correspondences are provided in all images. The trajectory bases have been modeled with Discrete Cosinus Transform (DCT) coefficients in [Akhter et al., 2009; Gotardo and Martinez, 2011]. One important advantage compared to shape bases is that trajectory bases can model large and complex deformations. For instance, [Gotardo and Martinez, 2011] introduces a kernel matrix (Gaussian radial basis function) which extracts strong non-linear deformations from the shape trajectory coefficients. This allows them to capture more complex deformations such as uncorrelated articulations without increasing the number of bases, but requires a more complex optimization strategy. The work of [Dai et al., 2014] has addressed this issue: it proposes a convex solution to the so-called 'prior-free' NRSfM problem. This is done by introducing a corrective matrix which allows them to recover a unique non-rigid shape thanks to a rank-minimization process.

The advantages of trajectory bases priors are several: the number of unknowns are reduced, computation is more stable and large deformations such as human body motion are recoverable. However, these methods still require video sequences or short-baseline data to achieve good results.

2.3.1.2 Physics-Based Priors

Physics-based deformation models operate very differently to statistical models, and restrict the space of possible deformations according to physical properties of the object's material. Similarly to SfT, the most common physics-based priors is isometry or quasi-isometry [Chhatkuli et al., 2014, 2017; Parashar et al., 2016; Varol et al., 2009; Vicente and Agapito, 2012; Wang et al., 2016]. It appears that NRSfM with the isometric prior can be solved up to discrete, spatially localized two-fold ambiguities if motion can be estimated densely across the object's surface [Chhatkuli et al., 2014; Parashar et al., 2016; Varol et al.,

2009]. We illustrate in figure 2.7 some works using physics-based priors and give further details in §2.3.4.

2.3.1.3 Temporal Smoothness Priors

Temporal smoothness priors assume that the object deforms smoothly over time. These priors have been mainly used through two approaches: *(i)* using temporal smoothing constraint [Akhter et al., 2009; Fayad et al., 2010; Garg et al., 2013; Olsen and Bartoli, 2008] and *(ii)* initializing the shape of an input image using the one of the previous input image [Wang et al., 2016]. Temporal smoothness can also be used to assist correspondence estimation generally using optical flow approaches to obtain dense correspondences [Garg et al., 2013; Wang et al., 2016]. Figure 2.7 illustrates the work of [Garg et al., 2013]. Advantages and drawbacks of such approach are similar to the ones of the SfT methods which work on video sequences and use temporal continuity. Details are given in §2.2.5.

2.3.2 Data Constraints in NRSfM

Similarly to SfT, most NRSfM methods rely on motion constraints and can be divided into two types: those which assume the correspondences are computed *a priori* [Chhatkuli et al., 2014, 2017; Garg et al., 2013; Gotardo and Martinez, 2011; Parashar et al., 2016; Varol et al., 2009; Vicente and Agapito, 2012], and those which compute correspondence jointly to deformation inference [Wang et al., 2016].

Only one work has recently tackled the problem of reconstructing poorly-textured surfaces [Wang et al., 2016] directly by computing dense correspondences. Precisely, it uses a brightness constancy constraint which is robust to small illumination variations, mainly caused by shading or illumination changes. [Wang et al., 2016] also uses boundary contour constraints and a temporal smoothing constraint to help the registration. [Wang et al., 2016] requires the image sequence to deform smoothly over time, and has shown to work only with simple smooth surfaces such as bending sheets of paper. Similarly to SfT methods, the reason is such smooth surfaces were considered because motion information is fundamentally insufficient to reconstruct non-smooth deformations.

2.3.3 Unorganized Image Sets Versus Video Inputs

The set of images used as input in NRSfM can be either an unorganized set of images, or frames from a continuous video. A fundamental difference between these two settings is the temporal continuity can be exploited for video inputs, but not for unorganized sets of images. We refer to §2.2.5 for the advantages and the drawbacks of temporal continuity.

To handle these two settings, there are two ways used by NRSfM methods. The first way is as a batch of images. This is typically the case when unorganized images are used [Chhatkuli et al., 2014, 2016; Fayad et al., 2010; Parashar et al., 2016; Taylor et al., 2010; Varol et al., 2009; Vicente and Agapito, 2012; Wang et al., 2016]. It can also be used for video inputs, however usually frame selection is required because batch methods do not generally scale well to

large numbers of images. The second way is to process the image data incrementally [Akhter et al., 2009; Fayad et al., 2010; Garg et al., 2013; Olsen and Bartoli, 2008]. This is usually used for video inputs, where new frames are sequentially added to the optimization problem. Unlike batch methods, incremental methods are designed to scale well for long videos, and are strongly inspired by incremental methods in rigid SfM.

2.3.4 Local and Global Methods to NRSfM

Another important way to characterize NRSfM methods is whether they reconstruct a surface using local surface regions (usually called local methods), or whether they reconstruct the whole surface at once (usually called global methods).

2.3.4.1 Local Methods Using the Isometry Prior

These methods work by dividing the surface into local regions, reconstructing each region individually and then reconstructing the full surface using surface continuity. Most local methods assume isometric deformations. Local approaches mainly differ by the way they locally model the surface. Proposed models include piecewise planes [Collins and Bartoli, 2010; Taylor et al., 2010; Varol et al., 2009], quadrics [Fayad et al., 2010] or PDEs [Chhatkuli et al., 2014; Parashar et al., 2016]. The advantages of local approaches is that they can be fast and they can provide closed-form solutions. However, they also produce sub-optimal results since they do not enforce the physical prior globally over the whole surface. Other drawback are that they may be unstable and present local ambiguities.

[Varol et al., 2009] uses a piecewise planar model that works by estimating motion between pairs of images using local homographies, then reconstructing the corresponding planes' poses using homography decomposition. The decomposition has two solutions, which are resolved using spatial smoothness. Finally, the local planar reconstructions are merged together to form a final surface. This approach assumes that the surface is piecewise planar and has shown to work only on very smooth surfaces such as a bending sheet of paper. This strategy was followed in [Collins and Bartoli, 2010; Taylor et al., 2010] using the orthographic camera model, which sacrifices accuracy for better stability. Instead of piecewise planes, [Fayad et al., 2010] uses quadratic surface patches (known as quadrics). It decomposes each local region as a combination of camera parameters and quadratic deformations parameters. These parameters are locally estimated through a non-linear optimization which minimizes reprojection and temporal constraints. Then, a global optimization is performed to stitch all reconstructed quadrics.

In [Chhatkuli et al., 2014; Parashar et al., 2016], the problem was modeled using continuous differential geometry. [Chhatkuli et al., 2014] improves on [Varol et al., 2009] by assuming infinitesimal planarity. This corresponds to assume that the surface is infinitesimally planar. The main advantage is that there is no explicit segmentation of the surface into planar regions. [Chhatkuli et al., 2014] has shown that, with the infinitesimal planarity assumption, more general shapes and deformations can be modeled and that the solution is unique for

two or more views. [Parashar et al., 2016] also uses infinitesimal planarity, but represents the surfaces as Riemannian manifolds. This allows them to introduce the metric tensor and the Christoffel symbol fields, which are proved to be related across the set of images by simple rules depending only on the inter-image warps. Using these differential quantities and the infinitesimal planarity assumption, [Parashar et al., 2016] obtains a system of two quartics in two variables for each image pair, whose solution directly leads to the surface normal field.

2.3.4.2 Global Methods

Global methods using the isometry prior. Global methods do not divide the surface into local regions that are independently reconstructed. Therefore, they can use constraints acting over the whole surface. These methods produce large, non-convex optimization problems that cannot lead to closed-form solutions. They generally use energy minimization frameworks for optimization. This allows them to handle more complex deformations and to use more complex constraints, leading to potentially more accurate reconstructions. However, they present four drawbacks: they generally require high computation time, they are often difficult to optimize, they require a good initial solution and they are not easily parallelizable.

Global methods [Vicente and Agapito, 2012; Wang et al., 2016] have used quasi-isometric constraints with surface mesh models. In both methods, optimization is done using fusion moves, with proposals generated from locally-optimal solutions found using Levenberg-Marquardt. The difference between [Vicente and Agapito, 2012] and [Wang et al., 2016] is the data constraints used: the first uses feature matches while the second uses dense correspondences assuming brightness constancy.

Global methods using the inextensibility prior. Inextensibility constraint has been also used in [Chhatkuli et al., 2016], a global and easy to implement approach. For this, [Chhatkuli et al., 2016] adapts the MDH approach developed in SfT. It formulates the NRSfM as a convex problem using the inextensibility constraint and maximizing the depth at each point correspondence. The problem is then solved globally and optimally as an SOCP problem. Unlike the SfT case, the template is unknown. In this case, the template geometry is modeled by the 3D Euclidean distances between pairs of correspondences. Because these distances are unknown, extra constraints are required to prevent trivial solutions. For example, zero cost can be achieved by having a template with zero inter-correspondence distances, and zero depth in each frame. To overcome this, a simple and effective strategy was found by forcing the average inter-correspondence Euclidean distances to be 1. Contrary to nearly all global methods, [Chhatkuli et al., 2016] does not require initialization.

Global methods using non-isometric physical priors. There are global approaches which can handle non-isometric deformations, such as elastic [Agudo et al., 2016]. Most of them share some advantages and drawbacks with global methods which use quasi-isometric constraints: they require good initial solution and batch global methods are computationally demanding.

A solution to NRSfM that could handle some stretching deformations was proposed [Agudo et al., 2016]. It models the surface with a thin-shell and deformation with continuum mechanics. The deformation model gives a system of PDEs which is solved using the Finite Element Method (FEM). The solution obtained from FEM is then used to constrain the deformation of the surface in a formulation which combines Simultaneous Localization and Mapping (SLAM) and Extended Kalman Filter (EKF) and jointly estimates the shape of the deforming surface and the camera trajectory. This method has a low computational cost, but it requires to know *a priori* the template's 'rest shape'. This was estimated using an initial video sequence where the object stays rigid. This makes this method not a true NRSfM method, because the template's rest shape is required *a priori*. It is therefore more like an SfT method.

2.4 Further Details on Feature-Based Matching and Optical Flow

The problem of registering 2D image data is a fundamental cornerstone of those 3D reconstruction techniques that rely on motion. For methods which use motion as main visual cue such as SfT (motion between the template's texture-map and an input image) and NRSfM (motion between pairs of input images), obtaining accurate motion constraints is a critical requirement to reach high reconstruction accuracy. We restrict the review here to registration of RGB images (or color images) and grayscale images. Image registration has been mainly addressed through two sub-problems: feature-based matching and optical flow. In general, feature-based matching operates on wide-baseline (*i.e.* unorganized images) and optical flow in narrow-baseline (*i.e.* temporally coherent images with low correspondence displacement).

2.4.1 Feature-Based Matching Methods

Feature-based matching aims to establish correspondences between two sets of features extracted from two images. Features are distinctive local regions of an image, which can be repeatably detected in other images. They are detected by mainly using strong bi-directional image gradient filters and characterized by appearance descriptors. Several types of feature detectors and feature descriptors have been proposed such as SIFT and SURF. Solving the feature matching problem has been done mainly through four approaches: *(i)* local approaches, *(ii)* parametric models, *(iii)* graph-based approaches and *(iv)* neural network approaches. Figure 2.8 illustrates these four categories.

Category *(i)*: local approaches. Category *(i)* methods are the simplest approaches [Lowe, 2004]. For each feature in one image, the distances (usually ℓ_2) between its descriptor and the descriptors of all features in another image are computed. Then the feature in the other image with the lowest distance is outputted as a match. The search is conducted either exhaustively or approximately using efficient data structures such as kd-trees.

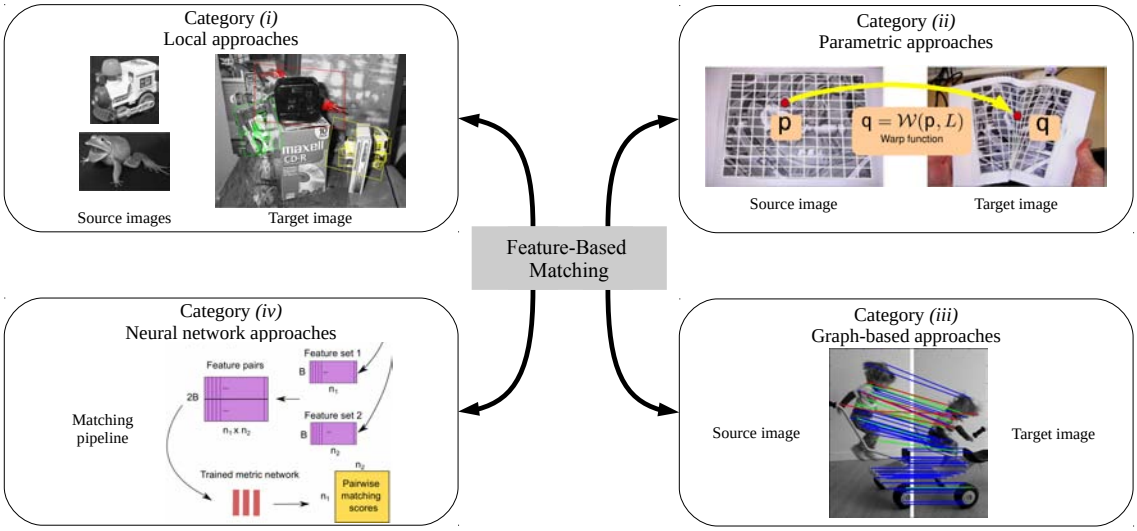


Figure 2.8: Illustration of the different methods of feature-based matching. Results are taken from the following works (from category *(i)* to category *(iv)*): [Lowe, 2004], [Pizarro and Bartoli, 2012], [Torresani et al., 2008b] and [Han et al., 2015].

Spatial constraints are therefore not used for feature matching. This approach usually suffers from a high number of mismatches (features in one image which are incorrectly matched in the other image). To remove these, two strategies are usually used: the nearest neighbor distance ratio test [Lowe, 2004] and the cross check test. This first takes for each feature in one image the two closest distinct features in the other image, and computes the ratio of the distance to these features. Reliable matches are likely to be those with small distance ratios. The second approach keeps matches if they are also found when the roles of the images are swapped. The cross-check is known to remove larger number of mismatches, but often a significant number of correct matches are also lost.

Category *(ii)*: parametric approaches. This category of methods finds correspondences by matching feature descriptors using the support of spatial transformations such as 2D meshes [Pilet et al., 2007] or Thin-Plate Splines (TPS) [Pizarro and Bartoli, 2012; Tran et al., 2012]. Transformation parameters are obtained by robust iterative minimization [Pilet et al., 2007; Pizarro and Bartoli, 2012], or, for low complexity transforms, RANdom SAmple Consensus (RANSAC) [Tran et al., 2012]. The use of spatial support is a strong advantage of this category: it imposes implicitly spatial consistency to the correspondences. Parametric approaches can provide good matching and increase robustness to non-rigid deformations, occlusions and changes of capture conditions (illumination, viewpoint, motion blur). One drawback is that the parametric approaches which use iterative optimization can be quite expensive to perform. However, a real-time implementation is proposed by [Pilet et al., 2007]. Selecting the best model is an important challenge of this category because a model which is too flexible can incorrectly classify mismatches as true matches and a model which is too

stiff can incorrectly classify true matches as mismatches. Two other challenges are surface self-occlusions and high mismatches ratios.

[Pilet et al., 2007] uses a hexagonal mesh and minimizes a robust correspondence constraint and a deformation prior which enforces mesh regularity and surface smoothness. Robustness in the correspondence constraint is obtained through two strategies. The first is the use of an M-estimator which reduces the influence of a match when its associated correspondence residual error is high. The second is to decrease progressively this threshold and eliminate all matches with large residual errors. During the first iterations, this gives more influence to the deformation prior, leading to a very smooth surface.

[Tran et al., 2012] uses RANSAC to find the best affine subspace, *i.e.* which includes the maximum number of matches, and uses the inliers, *i.e.* matches which lies on the best affine subspace, to estimate a global TPS. RANSAC methods require a parametric deformation model, a sampling process to give minimal samples for estimating the parametric defomation, and a consensus process for validating if the deformation can explain well the entire set of matches. In [Tran et al., 2012], the parametric model was similar to an affine transform.

[Pizarro and Bartoli, 2012] finds inliers using a local parametric model. First, it performs a Delaunay triangulation using all matches. Second, for each match, it fits a TPS using its neighboring matches points which satisfy a distance criterion: matches which are retained are inliers. Third, each outlier is tested again by updating the Delaunay triangulation and testing the distance criterion which is obtained by fitting a TPS using the new neighboring matches points. This process repeats until convergence. This method handles the discontinuities occurring from self-occlusions.

Category *(iii)*: graph-based approaches. Category *(iii)* methods are related to the problem of graph matching. This category is arguably the most studied approach to feature matching [Chertok and Keller, 2010; Collins et al., 2014; Duchenne et al., 2011; Gold and Rangarajan, 1996; Lee et al., 2011; Leordeanu and Hebert, 2005; Leordeanu et al., 2009; Torresani et al., 2008b]. Generally, two graphs are defined: graph nodes represent features in each image and graph edges encode affinity between features. Graph matching consists in finding a correspondence between nodes of the graphs. A common way to formulate this problem is with a quadratic binary assignment problem which consists in finding the correspondences which maximize a quadratic non-convex energy function. This energy function generally encodes two types of terms: unary terms which measure how similar two features from the two images are, and pairwise terms which measure how compatible two pairs of features are using geometric and/or spatial coherence constraints. Graph-based approaches allow to model more complex image motion such as discontinuities. However, they may be less constrained than parametric approaches and costly to perform inference on (since it is generally NP-hard). Unlike category *(ii)* methods, there are no real-time category *(iii)* methods. One of the big challenges of this category is to propose efficient, accurate and faster algorithms to solve the problem approximately.

Category *(iv)*: neural network approaches. Category *(iv)* method trains a neural network using very large datasets in order to match features between two images [Han et al., 2015]. The neural network approach has shown to give dense correspondences, under different capture conditions, non-rigid deformations and repetitive textures. Most of the works which use neural networks are focused on predicting discriminative features. From these, category *(i)*, *(ii)* or *(iii)* methods can be built. However, we can note the work of [Han et al., 2015] which learns a similarity function using deep neural networks to match features between image patches. One strong advantage is that these methods can learn more complex similarity functions beyond distance metrics such as Euclidean distance.

2.4.2 Optical Flow Methods

The goal of optical flow is to establish dense correspondence between two images. The basic assumption behind optical flow is brightness constancy, which says that the pixel intensity of a point remains constant during the displacement. The unknown of the problem is the flow field which is the projection of the real world 3D motion onto the 2D image. The problem is challenging because the brightness constancy is a highly non-convex constraint. The strategy used by most of the works is to linearize the brightness constancy equation. However, one big limitation with linearizing brightness constancy is that it is only valid for small displacement (because linearization is only valid locally). Another important point is that the brightness constancy cannot be met in practice and is sensitive to noise and illumination changes. To handle these drawbacks, robust data terms have been proposed: gradient constancy [Brox et al., 2004], higher order derivatives [Papenberg et al., 2006], or using photometric invariant channels [Mileva et al., 2007]. As figure 2.9 shows, optical flow methods can be broken down into four main categories: *(i)* local approaches, *(ii)* local search approaches, *(iii)* variational approaches and *(iv)* neural network approaches. We now discuss these categories.

Category *(i)*: local linearized approaches. These methods use a parametric model to represent local flow field. Spatial consistency is implicitly enforced locally by the parametric model. Several models have been used such as polynomial [Lucas and Kanade, 1981], motion bases [Bergen et al., 1992] or learned bases [Black et al., 1997]. Another important component of this category is the selection of the local domains: these can be square patches with fixed size or adaptive sizes and positions, or segmented regions which are computed *a priori* or jointly to the flow field estimation using for instance SuperPixels [Ren and Malik, 2003]. The advantages of local approaches are they are fast and highly parallelizable (solvable on GPUs). One significant disadvantage is known as the 'aperture problem': this occurs when there is insufficient image structure in the local region to solve the flow (orthogonal motion to the image gradient direction cannot be estimated).

Category *(ii)*: local search approaches and Block Matching. These methods use the fact that good correspondences between a source patch and target patches can be propagated to their adjacent patches (in the target image) [Bao et al., 2014; Hornáček et al., 2014; Hu

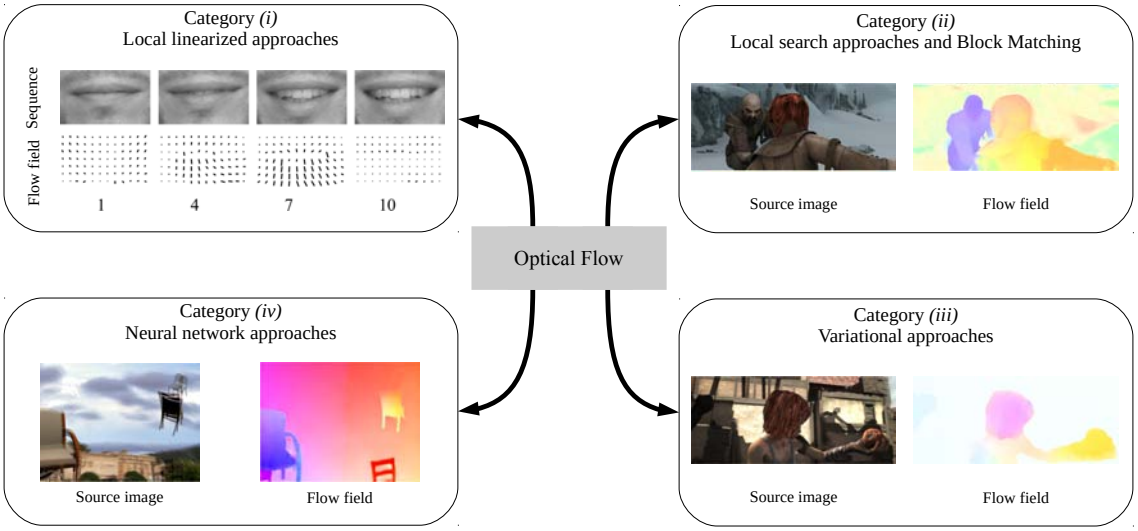


Figure 2.9: Illustration of the different methods of feature-based matching. Results are taken from the following works (from category *(i)* to category *(iv)*): [Black et al., 1997], [Bao et al., 2014], [Revaud et al., 2015] and [Dosovitskiy et al., 2015].

et al., 2016]. The propagation is performed by computing the data term between the source patch and the tested target patch. To move away from local minima, some random patches from the neighborhood are also tested. Category *(ii)* methods handle large displacements and are fast (they can be parallelized), however they may be inaccurate for small displacements and/or low texture scenes.

Category *(iii)*: variational approaches. These methods construct a differentiable cost function composed of a data term and a smoothing term, which acts as spatial consistency, evaluated over all the pixels of the image [Horn and Schunck, 1981]. A robust estimation framework has been proposed by [Black and Anandan, 1993], which promotes piecewise smooth flow field, which commonly occur at occlusion boundaries. [Black and Anandan, 1993] use an M-estimator instead of the ℓ_2 estimator for the smoothing term, which has the effect of modeling piecewise smooth flow fields. Many improvements have been introduced: handling illumination changes [Wedel et al., 2009], occlusions [Revaud et al., 2015], large displacements [Brox and Malik, 2011] and increasing the computational speed [Zach et al., 2007]. The main limitations are that such approaches lead to difficult optimization problems and the global optimum is not guaranteed. The main advantage is that they give a full dense flow field. They generally perform very well for small displacements, and usually provide better results than other category methods on datasets such as Middlebury [Baker et al., 2007].

Category *(iv)*: neural network approaches. These methods train a neural network to predict optical flow and use very large training sets [Dosovitskiy et al., 2015; Ilg et al., 2017].

There are two big difficulties: the first is in acquiring sufficiently large training sets and the second is that acquiring training sets for this problem is particularly hard. To handle this second difficulty, researchers have looked at simulated datasets such as the 'Flying Chairs' dataset [Dosovitskiy et al., 2015]. Despite the lack of realism of such a dataset, [Dosovitskiy et al., 2015] has shown that its convolutional neural network generalizes surprisingly well to realistic datasets. To overcome the difficulty of acquiring ground-truth, there are some recent work on unsupervised training [Meister et al., 2017; Ren et al., 2017]. The main advantage of this category is the computational speed. So far they have however not significantly improved accuracy compared to the best variational methods [Geiger et al., 2012].

2.5 3D Reconstruction Using Shading

Shading links the intensity value with the surface geometry of the object observed, the surface reflectance, the illumination of the scene and the camera response. This relationship is encoded by what we call the shading equation. In general, the shading equation provides one constraint on the surface normal at any given pixel. Shading is a powerful visual cue because, unlike motion, it can constrain 3D shape at weakly textured surface regions. Shading has been first used alone in the paradigm of SfS and then as a complementary visual cue in SfT and other 3D reconstruction problems.

2.5.1 Shape-from-Shading

SfS consists in using shading to recover the 3D shape of an object from a single image. Precisely, it recovers the surface normal at each pixel of the image. As mentioned earlier in §1.2.4, SfS has been intensively studied in the last decades and the SfS literature can be explored through four main components: *(a)* the camera projection model, *(b)* the illumination model, *(c)* the surface reflectance model and *(d)* the 3D shape inference algorithm. For *(a)*, SfS has been first studied with the orthographic camera [Horn, 1970; Pentland, 1984], and then the perspective camera model [Prados and Faugeras, 2005; Tankus et al., 2005]. For *(b)*, most of the existing methods assume a distant light source, but more complex illumination models are also used, such as the near-point lighting with fall-off [Okatani and Deguchi, 1996; Prados and Faugeras, 2005]. Most of SfS methods also assume known illumination. For *(c)*, a very common assumption of reflectance is the Lambertian model [Ecker and Jepson, 2010; Horn, 1970; Ikeuchi and Horn, 1981; Kimmel and Bruckstein, 1994; Lee and Kuo, 1993; Okatani and Deguchi, 1996; Pentland, 1984; Prados and Faugeras, 2005; Richter and Roth, 2015; Rouy and Tourin, 1992; Tsai and Shah, 1994; Xiong et al., 2015]. the Lambertian model assumes diffuse reflection of the surface, *i.e.* the re-emitted light does not depend on the incident direction. Non-Lambertian reflectance models are also studied [Ahmed and Farag, 2007; Lee and Kuo, 1997], such as the Oren-Nayar and Ward models which respectively take into account the micro-facets reflections and specular reflections. Nearly all methods assume either constant and fixed albedo or known albedo. This is because SfS is fundamentally an ill-posed

problem with varying albedo. Some works however propose solutions to handle multi-albedo surfaces. To handle multiple albedos, [Barron and Malik, 2015] forms a complex energy function which simultaneously solves several problems related to the photometric formation of the image, namely SfS, intrinsic images decomposition, color constancy and illumination estimation. For *(d)*, SfS methods can be divided in six subcategories: *(i)* propagation approaches, *(ii)* local approaches, *(iii)* linear approaches, *(iv)* convex minimization approaches, *(v)* non-convex minimization approaches and *(vi)* learning-based approaches. We briefly discuss these approaches. Figure 2.10 illustrates these six categories.

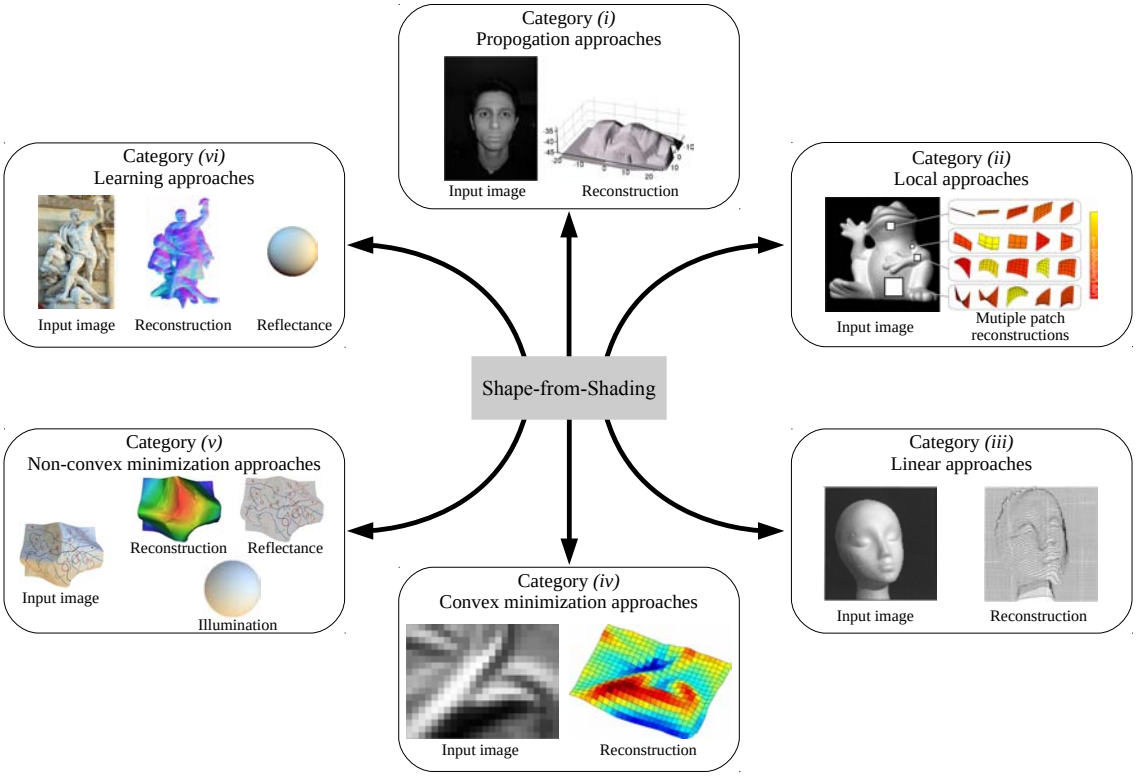


Figure 2.10: Illustration of the different categories of SfS methods. Results are taken from the following works (from category *(i)* to category *(vi)*): [Prados and Faugeras, 2005], [Xiong et al., 2015], [Tsai and Shah, 1994], [Ecker and Jepson, 2010], [Barron and Malik, 2015] and [Richter and Roth, 2015].

Category *(i)*: propagation approaches. These methods propagate known local shape information (usually points or curves) over the whole surface by solving a PDE with a boundary condition corresponding to local shape information. Propagation methods work in three steps. First, they transform the shading equation into a first-order non-linear PDE in terms of surface depth. Second, they formulate the PDE as a Hamilton-Jacobi PDE with boundary conditions. Most of the boundary conditions correspond to points or sets of points on the input image, whose depth is known or can be uniquely determined. These can be singular points (brightest points) [Rouy and Tourin, 1992], curves such as occluding contours [Ahmed

and Farag, 2007; Rouy and Tourin, 1992] or equi-depth contours [Kimmel and Bruckstein, 1994]. [Prados and Faugeras, 2005] uses as boundary conditions the Hamilton-Jacobi PDE, which is positive over the image. This comes from the fact that [Prados and Faugeras, 2005] does not neglect the light fall-off, which makes it impossible to convert the boundary conditions to value conditions for depth. The third step uses an iterative algorithm to solve the Hamilton-Jacobi PDE with the boundary condition. In general, few iterations are required and the computation is fast. Figure 2.10 gives a reconstruction performed by [Prados and Faugeras, 2005].

Category *(ii)*: local approaches. These approaches compute the surface normal at each pixel of the image independently of the other pixels. They then strictly require strong assumption on the surface. For instance, [Pentland, 1984] assumes that the surface is locally spherical. It computes at each pixel the surface normal using the first and second derivatives of the shading equation. [Xiong et al., 2015] exploits local shading context to predict shape from shading. The image is divided into patches at several scales, and for each patch a distribution of quadratic shapes with corresponding likelihoods is computed. Then a minimization process is used to find the most likely combination of local quadratic shapes by exploiting spatial continuity. Figure 2.10 gives a reconstruction performed by [Xiong et al., 2015].

Category *(iii)*: linear approaches. These approaches relax the non-linearity of the shading equation by approximating it as a linear equation. [Pentland, 1988] uses the linear approximation of the shading equation in terms of the surface gradient. It then applies a Fourier transform on the linearized shading equation, which becomes linear in the depth. It obtains an analytical solution of the depth at each point. [Tsai and Shah, 1994] uses a discrete approximation of the surface gradient and then the linear approximation of the shading equation. This leads to a well-determined system of equations expressed in terms of surface depth, which is easily solved by a Jacobi iterative method. Figure 2.10 gives a reconstruction performed by [Tsai and Shah, 1994]. [Tsai and Shah, 1994] inspired [Collins and Bartoli, 2012] which proposes the first real-time implementation of SfS for laparoscopic images.

Category *(iv)*: convex minimization approaches. Category *(iv)* methods relax SfS into a convex problem. [Ecker and Jepson, 2010] formulates the shading equation as a second-order polynomial system in terms of surface depth and transform the SfS problem as an Semi-Definite Programming (SDP) optimization problem. For this, it introduces new variables for the depth products so that the inherent non-linearity of the shading equation becomes linear. Additional constraints are used to reduce the computational burden and to better constrain the problem. The main advantage is that using an SDP formulation guarantees the convergence to a global minimum without any initial estimate. Figure 2.10 gives a reconstruction performed by [Ecker and Jepson, 2010].

Category *(v)*: non-convex minimization approaches. Category *(v)* methods minimize the difference between the input image and the predicted image obtained using the shading equation which encodes the image formation modeling. As the shading equation gives one constraint for two unknowns (the surface gradient), they also integrate additional constraints in the objective function. [Ikeuchi and Horn, 1981] minimizes the shading equation and a smoothing constraint. To ensure good convergence, it initializes the shape with occluding contours. [Lee and Kuo, 1993] follows a similar approach, but models the surface using a linear combination of modes. [Ecker and Jepson, 2010] also proposes a non-convex optimization with a smoothing term which encourages folds at image edges. The interesting work of [Barron and Malik, 2015] solves SfS by solving simultaneously the problem of shape, reflectance and illumination estimations. It forms a complex cost function which optimizes the three unknown types using several constraints, such as piecewise constant reflectances, surface smoothing and occluding contours constraints. This cost function involves several hyperparameters which are computed using a training set created for this purpose. Then, the cost function is minimized by gradient-based optimization. Figure 2.10 gives a reconstruction performed by [Barron and Malik, 2015]. Despite very good reconstruction accuracy, category *(v)* methods require a longer computational time and may fall into local minima.

Category *(vi)*: learning approaches. A category *(vi)* method is proposed in [Richter and Roth, 2015] where it uses a regression forest and a large and high-quality synthetic database of 3D models. It assumes a Lambertian reflectance model and does not know *a priori* the scene illumination. It works in four steps. First, from the input image, it extracts three spatial features: color, texton and silhouette. Second, it uses the silhouette features to estimate the second-order spherical harmonics of the scene illumination. Third, it trains a regression forest model using the three spatial features and the 3D synthetic models of the dataset. Fourth, it uses the trained model to predict the surface normal at each pixel, using the same spatial features, and enforce integrability of the normal field. Figure 2.10 gives a reconstruction performed by [Richter and Roth, 2015]. These category methods give a very good reconstruction accuracy, but requires an appropriate training dataset.

SfS has drawn great interest and has been tackled through many approaches, however, almost all of the existing SfS methods present the same shortcomings. First, they assume the albedo values and the scene illumination to be known, *i.e.* they require a complete photometric calibration, even depth at some points of the image for propagation methods. Second, they suffer from the convex/concave ambiguity. Third, they provide a surface solution up to a global scale, which is an inherent limitation to the SfS problem. As some works [Collins and Bartoli, 2012] underline, shading should be combined with other visual cues to remove for instance the convex/concave ambiguity.

2.5.2 Extending SfS to Multiple Images

Shading has been used previously in several other 3D reconstruction problems. These include photometric stereo [Brostow et al., 2011; Zhou et al., 2013], multi-view SfS [Jin et al., 2008; Wu et al., 2010], multi-view reconstruction [Beeler et al., 2010; Kim et al., 2016; Langguth et al., 2016; Valgaerts et al., 2012; Wu et al., 2011], SfM and SfS [Kim et al., 2016]. Figure 2.11 illustrates these four 3D reconstruction problems. Their main limitations are that they work for rigid objects or/and use unpractical setups.

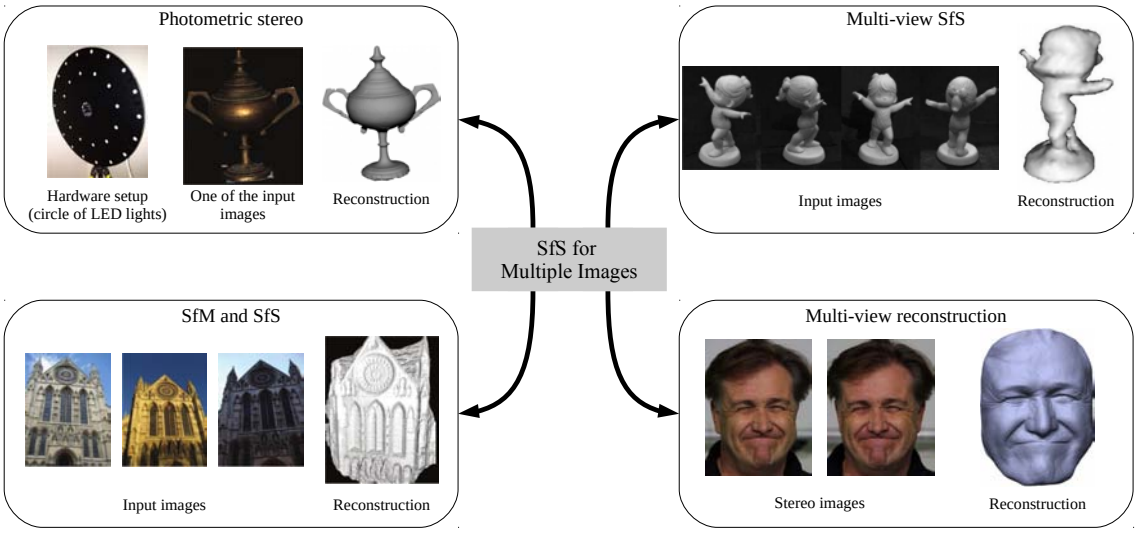


Figure 2.11: Illustration of the different methods which extend SfS to multiple images. Results are taken from the following works (clockwise from top left): [Zhou et al., 2013], [Jin et al., 2008], [Valgaerts et al., 2012] and [Kim et al., 2016].

Photometric stereo is the extension of SfS using multiple light sources. The images taken under different illumination contain no motion. This is one big difference between photometric stereo and the other extensions of SfS. It has shown great success for reconstructing high-accuracy surface details with unknown albedo such as [Brostow et al., 2011; Zhou et al., 2013]. However, it requires a special hardware setup where the scene is illuminated by a sequence of lights placed at different points in the scene, during which time the scene is assumed to be rigid. This setup is not applicable in many situations. Photometric stereo does not require to solve the registration problem since the camera is fixed and the scene is rigid.

Multi-images SfS methods such as [Jin et al., 2008; Wu et al., 2010] have shown that using shading and a collection of images, from monocular [Jin et al., 2008] or several tracked cameras [Wu et al., 2010], provide reasonably good reconstructions of poorly-textured surfaces such as statues or bones.

Multi-view reconstruction methods such as [Beeler et al., 2010; Langguth et al., 2016; Valgaerts et al., 2012; Wu et al., 2011] have shown that shading reveals fine details for *e.g.* clothes or faces. However, these methods assume rigid objects, use two or more cameras and require a special design of the scene, which may not practical.

Shading has also been used in rigid SfM [Kim et al., 2016], which uses multiple images showing a rigid object. This approach initializes the surface using motion through SfM and Multi-View Stereo (MVS) and then refines it by combining motion with shading information. Unlike the other extensions of SfS, this approach requires to solve the registration problem. One limitation may come from the difficulty of establishing correspondences accurately. However, because of the MVS constraints, this approach may achieve higher accuracy than photometric stereo at both textured and textureless regions.

2.5.3 Existing Methods to Solve SfT with Shading

Shading has been already combined with SfT [Liu-Yin et al., 2016; Malti and Bartoli, 2014; Moreno-Noguer et al., 2009; Varol et al., 2012b]. These differ in the way the problem is modeled and optimized. The main difference is that [Moreno-Noguer et al., 2009; Varol et al., 2012b] start by using motion and shading information sequentially, and not in an *integrated* manner. We refer to these as *non-integrated* approaches. By contrast, in [Liu-Yin et al., 2016; Malti and Bartoli, 2014], shading, motion and deformation priors are integrated together into a single non-convex energy function which is minimized through iterative refinement, such as category *(v)* methods of SfS. We refer to these as *integrated* approaches.

Non-integrated methods. [Varol et al., 2012b] works by first segmenting the surface into textured and textureless regions. The textured regions are then independently reconstructed by an existing SfT method that uses only motion information [Salzmann and Fua, 2011], and local textureless patches are independently reconstructed by applying a trained SfS regressor. In a final stage, the patches are stitched together to form the final surface. This method has three drawbacks. First, it assumes the textureless regions have a single albedo. Second, it cannot handle occlusions (both external and self) in the textureless regions. This is because it reconstructs these regions using SfS. Third, it also requires full photometric calibration *a priori*. This is because SfS is solved with a trained regressor, which requires illumination to be known, and this to be the same at both training and test times.

[Moreno-Noguer et al., 2009] proposes a different approach from the previous one: it provides a closed-form solution to the 3D reconstruction problem of stretchable surfaces by using shading information instead of isometry (which does not hold in this problem). It assumes to have a template with a surface reflectance function and works by first transforming the template to camera coordinates using motion constraints (coming from point correspondences). The transform is not unique but up to a low-dimensional set of possible solutions. These correspond to different smooth reconstructions that satisfy the motion constraints. In a final stage, shading information is used to disambiguate the correct solution. The disambiguation problem is convex and cast as an SOCP problem. There are four drawbacks to this method. First, it only works for smooth surfaces. Second, it requires the template's reflectance function to be known *a priori*. Third, it cannot be used to enforce isometry, because it requires convex deformation priors. Fourth, the correct solution must be contained in the set of pos-

sible solutions, which is a significant limitation since it is only possible if the surface and its deformations are extremely smooth.

Integrated methods. [Liu-Yin et al., 2016; Malti and Bartoli, 2014] approach the problem in two stages: first, template construction and reflectance estimation with a rigid video section, and then deformation inference for the subsequent frames. Recall that a rigid observation video is a video of the object taken from different viewpoints before any deformation occurs.

The process to build the template's reference shape is as follows. First, a set of rigid keyframes is extracted from the rigid observation video and used to build the template. The template's geometry is constructed by performing dense rigid SfM using images from the rigid video section using well established methods [Hartley and Zisserman, 2003; Wu, 2011]. Some manual post processing is then required to clean-up the constructed mesh, including hole-filling, background removal and surface smoothing. Most arbitrary templates, from online repositories or 3D acquisition systems, contain a 3D model only with a texture-map, which is different from the surface reflectance function. However, the surface reflectance function is required to use shading with the object template. Estimating the surface reflectance function is a challenging task, particularly in uncontrolled conditions and when the object is deformable. [Liu-Yin et al., 2016; Malti and Bartoli, 2014] simplify the problem using the rigid observation video used for reconstructing the template and assuming the scene illumination to be constant and fix during the rigid observation video. The reason is because one can compute most photometric parameters easily from the rigid views of the object using linear least squares [Luong et al., 2002]. Their approaches are instantiations of a general schema, which we give in figure 2.12. The process to compute the reflectance function is as follows. Because SfM registers the keyframe images with the reference shape mesh, a photometric calibration can be performed by inverting the shading equation. Specifically, the registered shape mesh provides the surface normal estimate for each pixel of the object, in each keyframe image. By making certain assumptions on the reflectance function, scene illumination and camera responses, it is possible to estimate their parameters in closed-form. Table 2.2 gives the specific assumptions made by [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. [Malti and Bartoli, 2014] uses the Cook-Torrance reflectance model, whose parameters are the albedo, the Fresnel factor and the roughness factor. These parameters are estimated by inverting the corresponding shading equation near specular peaks. This is done to significantly simplify the problem, because the illumination vector can be approximated reasonably well at a specular peak. [Liu-Yin et al., 2016] estimates the albedo-map by inverting the shading equation over the whole surface while using a sparse prior to favour piecewise constant albedos.

	[Malti and Bartoli, 2014]	[Liu-Yin et al., 2016]
Reflectance function	Cook-Torrance with one albedo, Fresnel and roughness parameters	Lambertian with spatially varying albedo-map
Illumination model	Distant light model	Second-order spherical harmonics
Camera response model	Fixed and linear	Fixed and linear

Table 2.2: List of model assumptions for state-of-the-art SfT methods which use shading.

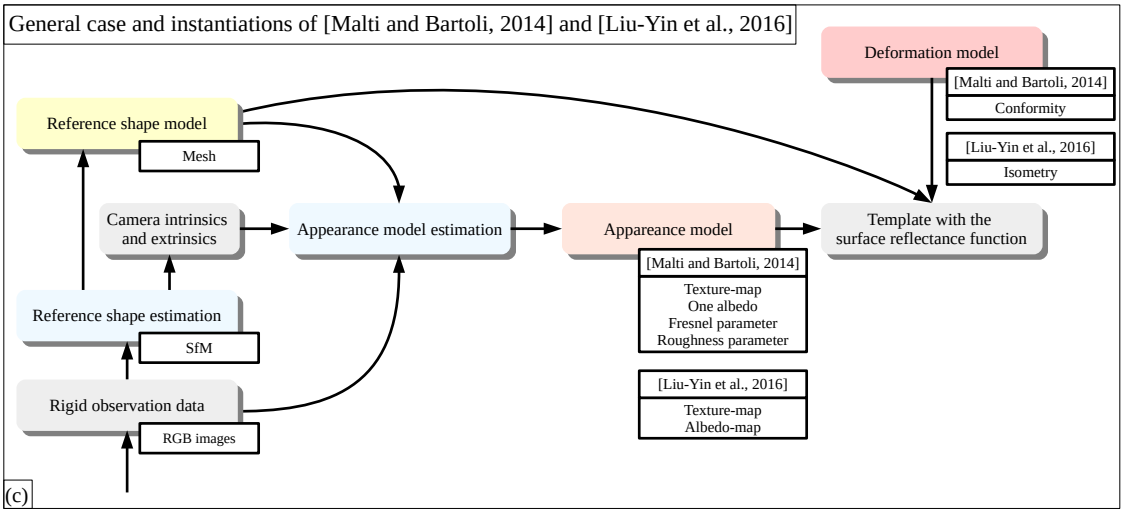


Figure 2.12: Generic case of template construction and surface reflectance estimation from a rigid observation video. This has been instantiated in two recent works [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. Specific modeling decisions of [Liu-Yin et al., 2016; Malti and Bartoli, 2014] are given in white boxes.

Once the template has been constructed and the surface reflectance function estimated, it is used to perform SfTS on the images in the deformable video section. To make the problem well-posed, [Liu-Yin et al., 2016] uses the As-Rigid-As-Possible (ARAP) prior of [Sorkine and Alexa, 2007] and a temporal continuity constraint. The ARAP prior corresponds to a soft isometric prior and a bending prior. [Malti and Bartoli, 2014] uses conformity to permit the surface to stretch while preventing angle changes.

Chapter 3

Shape-from-Template with Curves

Summary

We introduce Curve SfT, comprising two new cases of SfT where the shape is a 1D curve, Curve SfT-1 and Curve SfT-2. We present a thorough theoretical study of these new cases for isometric deformations, which are good approximations for ropes, cables and wires. Unlike Surface SfT, we show that Curve SfT is only ever solvable up to discrete ambiguities. We present the necessary and sufficient conditions for solvability up to these ambiguities with critical point analysis. We further show that unlike Surface SfT, Curve SfT cannot be solved locally using exact non-holonomic solution to a PDE. Our main technical contributions are two-fold. First, we give a stable, global reconstruction method that models the problem as a discrete Hidden Markov Model (HMM). This can generate all candidate solutions. Second, we give a non-convex refinement method using a novel angle-based parameterization. We present quantitative and qualitative results showing that real curved objects such as a necklace or roadway lines can be successfully reconstructed with Curve SfT. This chapter is a considerable extension of our peer reviewed paper [Gallardo et al., 2015]. This chapter is the result of a collaborative work including Daniel Pizarro.

We first review some works of 3D reconstruction of curves from images. We then define the two Curve SfT instances which we propose to solve in this chapter and specify the different models required for solving them. For both problem instances, we give a theoretical analysis and computational solutions.

3.1 Curve Reconstruction from Images

We refer to chapter 2 for the SfT state-of-the-art and we complement it with some background details on the reconstruction problem of 3D curves from images. The problem of 3D curve reconstruction has been addressed in the two last decades through several approaches [Berthilsson et al., 2001; Faugeras and Papadopoulo, 1993; Mai and Hung, 2010; Martinsson et al., 2007]. These works all assume the curves to be rigid, and these use multiple images. The images may come from an unorganized set [Berthilsson et al., 2001; Mai and Hung, 2010; Martinsson et al., 2007] or a monocular video sequence [Faugeras and Papadopoulo, 1993]. [Berthilsson et al., 2001] proposes an affine shape method for 3D curves which optimizes a subspace constraint to align the parameterizations of matched curves over a set of images. It also extends the method of bundle adjustment to 3D curves. [Mai and Hung, 2010] proposes a point-based 3D curve reconstruction method from multiple images with uncalibrated cameras. It selects one image as reference, selects a fixed number of 2D 'representative' points along the curve in the reference image and finds their matches in the other images. Then, it minimizes the reprojection error of the 'representative' points by adjusting alternatively the camera projection and the 'representative' points in 2D and 3D. In practice, it has shown to work only on curves lying on planar surfaces. [Martinsson et al., 2007] reconstructs 3D curves using a set of images and a CAD model. It proposes a two-stage adaptive reconstruction method which uses 3D Non-Uniform Rational Basis Spline (NURBS) to parameterize the curves. First, it optimizes 3D NURBS curves with a fixed number of control points by minimizing image contours and gradient intensity constraints. Second, it inserts new control points using a stop-criterion in order to capture high curvature. [Faugeras and Papadopoulo, 1993] provides a thorough theoretical study of SfM with rigid 3D curves: it gives the assumptions under which reconstruction is solvable with calibrated image sequences. As it relies on high order spatio-temporal derivatives, the method suffers in practice from numerical stability.

Even if these methods reconstruct 3D curves, their assumptions significantly differ from the ones of Curve SfT, which reconstructs a deformable curve from a single image and a 1D template.

Chapter outline. In §3.2, we model Curve SfT and give formally the main theoretical results. We show that *Curve SfT-1* and *Curve SfT-2* are solved with the same IVP, so we first study *Curve SfT-1* and then specialize it to *Curve SfT-2*. In §3.3, we discuss degeneracies and number of solutions for special scenarios of Curve SfT. In §3.4, we give our multi-solution reconstruction method based on an HMM (defined in §1.4.1 as category *(iv)*). We also present

the refinement method (category *(iii)*) and the single-solution methods (categories *(i)* and *(ii)*). In §3.5, we validate our category *(iv)* method with and without refinement on simulated and real datasets, for both instances of Curve SfT.

3.2 Problem Modeling and Theoretical Analysis

We remind the reader that the problem naming convention is given in table 2.1. Figure 3.1 gives a graphical representation of the different SfT cases. We now study the two main sub-cases of the Curve SfT problem, referred as $\text{SfT}^{1\to3\to2}$ and $\text{SfT}^{1\to2\to1}$. $\text{SfT}^{1\to3\to2}$ is when the template is a curve embedded in the 3D space and observed by a regular 2D camera. $\text{SfT}^{1\to2\to1}$ is similar to $\text{SfT}^{1\to3\to2}$, but the camera is 1D.

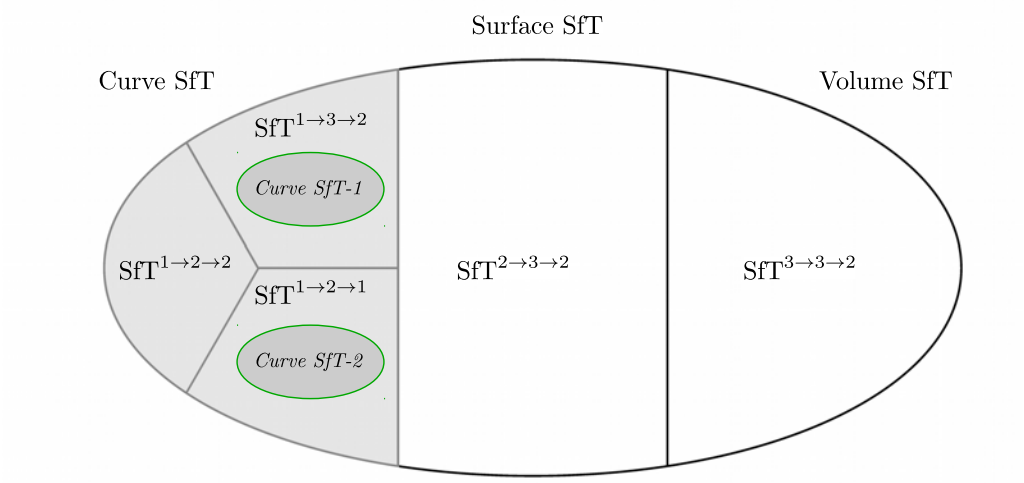


Figure 3.1: Diagram of the different SfT cases. The grey ellipses encircled by the green line denote the Curve SfT instances which we solve in this chapter.

3.2.1 Fundamental Models of Curve SfT

In order to solve Curve SfT, two fundamental models are required: the *template* and the *camera projection* model. The *template* is a fundamental element of Curve SfT since it gives strong physical constraint, as said in §2.2.2. The *camera projection* model determines how to reproject the 3D points used by the motion constraint, as explained in §2.2.3.1.

3.2.2 *Curve SfT-1* and *Curve SfT-2*: Two Instances of Curve SfT

We form the two problem instances of *Curve SfT-1* and *Curve SfT-2* using the eight components given in §2.1. *Curve SfT-1* and *Curve SfT-2* are respectively particular instances of $\text{SfT}^{1\to3\to2}$ and $\text{SfT}^{1\to2\to1}$. Illustrations of *Curve SfT-1* and *Curve SfT-2* are respectively given in figure 3.2 and figure 3.3. We first instantiate the problem components *(a)* to *(h)* for *Curve SfT-1* and give the reasons of each component specification. We then give the differences of instantiations for *Curve SfT-2*.

Fundamental model	Known *a priori*	Fixed or time-varying	Instantiation
Template's shape	✓	Fixed	General function
Template's appearance	✓	Fixed	1D image
Template's deformation	✓	Fixed	Isometric
Camera projection	✓	Fixed	Perspective

Table 3.1: Fundamental model instantiations in *Curve SfT-1*.

The *Curve SfT-1* instance. *(a) Models.* Table 3.1 presents the specifications of each fundamental model given in §3.2.1. We use a general function for the template's shape. We model the template's appearance with a known texture-map which is however unused. This is because we consider motion to be given *a priori*, as we state in the component *(f)*. Deformation is modeled quasi-isometrically. We assume the perspective camera model [Hartley and Zisserman, 2003], which handles well most real-world cameras. *(b) Exploited visual cues.* We use only motion visual cue because it is the main visual cue used in SfT. *(c) Number of required images.* A single image is required because we want to tackle the classical version of SfT. *(d) Expected types of deformations.* We assume quasi-isometric and no tearing. *(e) Scene geometry.* We assume no self or external occlusions, which is a typical assumption in the SfT state-of-the-art. There can be background clutter. *(f) Requirement for putative correspondences.* We assume to know *a priori* a set of putative 1D-2D correspondences from the texture-map of the template to the input image. We assume that these are sufficiently dense along the template. *(g) Surface texture characteristics.* We consider well-textured surfaces since it is an usual assumption in SfT. *(h) Known and unknown model parameters.* A template of the surface, as defined in §2.2.2, and the camera intrinsics are known. The unknowns are the 3D points of the deformed template in 3D camera coordinates.

Specialization to *Curve SfT-2*. *Curve SfT-2* differs from *Curve SfT-1* in three components, *(a)*, *(f)* and *(h)*. For *(a)*, *Curve SfT-2* adapts the perspective camera model to a 2D projection. The input image is 1D. For *(f)*, *Curve SfT-2* assumes 1D-1D correspondences between the texture-map and the input image. For *(h)*, the unknowns of *Curve SfT-2* are the 2D points of the deformed template in 2D camera coordinates.

3.2.3 *Curve SfT-1*: Reconstructing a 3D Curve from a 2D Image and a 1D Template

3.2.3.1 Template and Camera Modeling

The known template is 1D and we write it as $\mathcal{T} \subset \mathbb{R}$. We assume the template is deformed into a smooth unknown curve $\mathcal{S} \subset \mathbb{R}^3$ embedded in 3D. We denote the embedding function that generates $\mathcal{S}$ by $\varphi = (\varphi_x \; \varphi_y \; \varphi_z)^\top \in C^\infty(\mathcal{T}, \mathbb{R}^3)$. The 2D input image $I \subset \mathbb{R}^2$ is a

perspective projection of $\mathcal{S}$. We model projection by the pinhole camera Π:

$$\Pi(\mathbf{Q}) = \left(\frac{x}{z} \; \frac{y}{z}\right)^\top \quad \text{where} \quad \mathbf{Q} = (x \; y \; z)^\top. \tag{3.1}$$

The pinhole camera can be used for general perspective cameras when lens distortion has been corrected and the intrinsic calibration matrix has been standardized to $\mathbf{I}_3$ [Hartley and Zisserman, 2003].

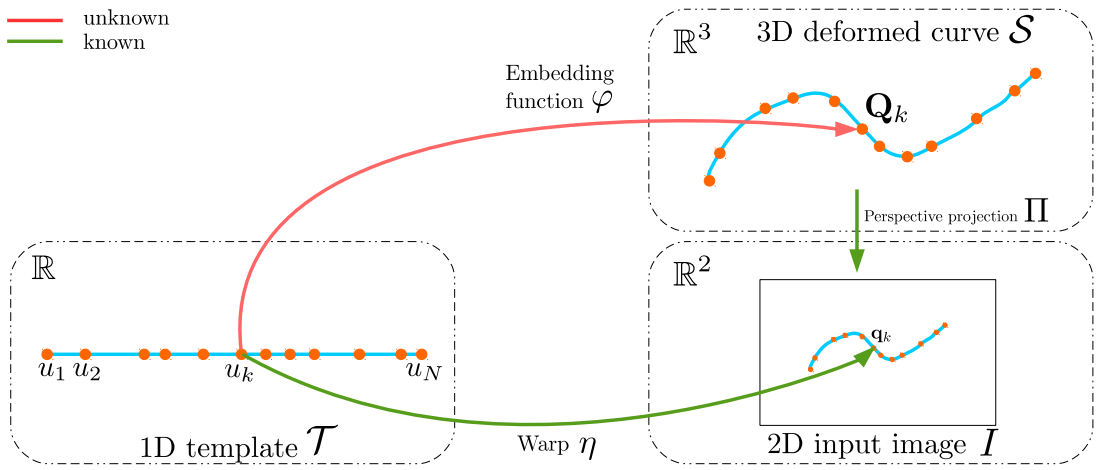


Figure 3.2: General modeling of *Curve SfT-1*.

3.2.3.2 Inputs and Outputs

We now give our inputs. *(i)* one RGB input image $I : \mathbb{R}^2 \to \{0, 255\}^3$ showing a deforming curve. *(ii)* a template of the surface, defined using §3.2.3.1. *(iii)* the camera intrinsics of the perspective 3D projection function Π. *(iv)* a set of N putative 1D-2D correspondences from the texture-map of the template to the input image. We denote the set by $\mathcal{S}_c = \{(u_k, \mathbf{q}_k)\}$ where u_k denotes the correspondence position in $\mathcal{T}$ and $\mathbf{q}_k$ denotes the correspondence position in the input image I. Details for how correspondences are computed for our experimental datasets are given in §3.5.1.2 and §3.5.2.2. We define $\eta \in C^\infty(\mathcal{T}, \mathbb{R}^2)$ as the *template-to-image warp*. As $\mathcal{S}$ has no self-occlusions in I, so η is bijective. As the set of N 1D-2D correspondence points are known between the template and the input image and as these are sufficiently dense, η can be estimated through a smooth interpolation. In practice, we use B-splines to model η.

Our solution to *Curve SfT-1* outputs the 3D points of the deformed template in 3D camera coordinates.

3.2.3.3 Theoretical Analysis

We now express mathematically *Curve SfT-1* and then propose a differential analysis of the problem. Our geometric modeling is shown in figure 3.2. It is inspired from [Bartoli et al.,

2015], for solving Surface SfT with continuous differential geometry. We recall that we study *Curve SfT-1* (and *Curve SfT-2*) under the assumption of known correspondences between the template and the input image. This is a reasonable assumption and a mandatory move to understand the theory behind this problem.

Problem formulation. *Curve SfT-1* involves recovering φ, from the warp η and the projection Π. This is constrained by the *isometry* prior and the *reprojection* constraints implied by η. The warp $\eta \in C^\infty(\mathcal{T}, \mathbb{R}^2)$ maps the template to a 2D input image. The reprojection constraint is therefore:

$$\eta = \Pi \circ \varphi. \tag{3.2}$$

The isometry constraint is a first order differential property in φ:

$$\mathbf{J}_\varphi^\top \mathbf{J}_\varphi = 1. \tag{3.3}$$

From the constraints (3.2) and (3.3), we define *Curve SfT-1* as follows:

$$\text{Find} \quad \varphi \in C^\infty(\mathcal{T}, \mathbb{R}^3) \quad \text{s.t.} \quad \begin{cases} \eta = \Pi \circ \varphi & \text{(reprojection)} \\ \mathbf{J}_\varphi^\top \mathbf{J}_\varphi = 1 & \text{(isometry).} \end{cases} \tag{3.4}$$

ODE formulation. We show that equation (3.4) is equivalent to finding the solution of a first-order non-linear ODE. Using equation (3.1), we first transform the reprojection constraint into:

$$\varphi = \varphi_z \bar{\eta}, \tag{3.5}$$

where φ_z is the depth component of φ and $\bar{\eta}$ is the warp in homogeneous coordinates. We differentiate equation (3.5) once, giving:

$$\mathbf{J}_\varphi = \varphi_z \mathbf{J}_{\bar{\eta}} + \varphi_z' \bar{\eta}. \tag{3.6}$$

We then substitute equation (3.6) into the isometry constraint from equation (3.4) and obtain a first-order non-linear ODE with φ_z as the unknown variable:

$$\varphi_z'^2 \|\bar{\eta}\|^2 + 2\varphi_z \varphi_z' \bar{\eta}^\top \mathbf{J}_{\bar{\eta}} + \varphi_z^2 \mathbf{J}_{\bar{\eta}}^\top \mathbf{J}_{\bar{\eta}} = 1. \tag{3.7}$$

Using the identities $\bar{\eta}^\top \mathbf{J}_{\bar{\eta}} = \eta^\top \mathbf{J}_\eta$ and $\mathbf{J}_{\bar{\eta}}^\top \mathbf{J}_{\bar{\eta}} = \mathbf{J}_\eta^\top \mathbf{J}_\eta$, we arrive at:

$$\varphi_z'^2 \|\bar{\eta}\|^2 + 2\varphi_z \varphi_z' \eta^\top \mathbf{J}_\eta + \varphi_z^2 \mathbf{J}_\eta^\top \mathbf{J}_\eta = 1. \tag{3.8}$$

Equation (3.8) is a first-order ODE. Given a solution to equation (3.8), the problem is solved and φ can be found as $\varphi = \varphi_z \bar{\eta}$. We now propose a change of variable that greatly simplifies equation (3.8) and the study of its solutions.

Change of variable. We define the function $\varepsilon = \|\bar{\eta}\|$, where $\varepsilon' = \frac{1}{\varepsilon}\eta^\top \mathbf{J}_\eta$. We now define a new function θ with:

$$\theta = \varphi_z \varepsilon. \tag{3.9}$$

Equation (3.8) can now be rewritten in terms of θ and θ':

$$\theta'^2 + \xi\theta^2 = 1 \quad \text{with} \quad \xi = \frac{1}{\|\bar{\eta}\|^2}\left(\mathbf{J}_\eta^\top \mathbf{J}_\eta - \frac{1}{\|\bar{\eta}\|^2}\mathbf{J}_\eta^\top \eta\eta^\top \mathbf{J}_\eta\right). \tag{3.10}$$

We use equation (3.10) to study the local solvability and the solution space of *Curve SfT-1*. This leads to two important results: *(i) Curve SfT-1* is not locally solvable exactly and *(ii)* there exist necessary and sufficient conditions for solving *Curve SfT-1* up a discrete number of ambiguities.

Local exact solutions. We explore whether local solutions of equation (3.10) exist using non-holonomic solution analysis. Non-holonomic solutions are based on the creation of new equations by differentiation and a relaxation of the differential dependencies. In our case, this means treating θ and θ' as independent variables. The results are called non-holonomic solutions [Eliashberg and Mishachev, 2002]. The uniqueness of non-holonomic solutions to PDE implies the uniqueness of solutions to the original ODE. Our main motivation is historical as non-holonomic solutions were used successfully in Surface SfT [Bartoli et al., 2015], to prove well-posedness and to give analytic solutions. We prove the following proposition:

Proposition 1 (Impossibility of non-holonomic solutions)**.** *The non-holonomic solution for φ_z in equation* (3.8) *is under-constrained for any order of differentiation.*

Proof. Equation (3.10) gives a single constraint for the two unknowns, θ and θ'. Differentiating equation (3.10) creates extra equations. Differentiating $k-1$ times yields k equations, however, each differentiation introduces one new unknown. For order k, we have a total of $k+1$ unknowns: $\theta, \theta', \ldots, \theta^{(k)}$. As a consequence we have k equations and $k+1$ unknowns, and the problem is under-constrained for any order $k > 0$. □

This proposition is important because it proves that local non-holonomic solutions to the ODE (3.10) do not exist.

Solution space. We now study the *global* solutions to *Curve SfT-1*. In general, a single ODE such as equation (3.10) has an infinite number of solutions. The true curve is one of these solutions. We construct an IVP by adding an initial condition $\theta(u_0) = \theta_0$ to the ODE (3.10). Our IVP writes as:

$$\begin{cases} \theta'(u) = \Psi(u, \theta(u)) = \sqrt{1 - \xi(u)\theta^2(u)} \\ \theta(u_0) = \theta_0 \end{cases} \tag{3.11}$$

The value $\theta_0 \in \mathbb{R}^*$ at u_0 is called the *anchor point*. We now use the following two properties.

Proposition 2 (Number of solutions to IVP (3.11))**.** *For the given IVP* (3.11)*,*

- *when* $\theta'(u_0) \neq 0$*, the IVP* (3.11) *has two solutions in a local interval of* u_0
- *when* $\theta'(u_0) = 0$*, the IVP* (3.11) *has at most two solutions in a local interval of* u_0*.*

Proof. When $\theta'(u_0) \neq 0$, we apply the Picard-Lindelöf theorem and obtain that the IVP (3.11) has two solutions in a local interval of u_0. This theorem is applicable to the IVP (3.11) because this IVP respects the Picard-Lindelöf conditions: the function Ψ is uniformly Lipschitz continuous in θ and continuous in u.

When $\theta'(u_0) = 0$, the number of solution is given by [Casillas-Perez and Pizarro, 2017]: there are at most two solutions in a local interval of u_0. The solution space is thus bounded if the anchor point is available. Here, we use the theorem given and proved in [Casillas-Perez and Pizarro, 2017] instead of the Picard-Lindelöf theorem which cannot be applied in this case. This is because, when $\theta'(u_0) = 0$, the function Ψ is not Lipschitz continuous in θ. To solve this, [Casillas-Perez and Pizarro, 2017] shows that at most two analytical solutions can be constructed in a local interval of u_0 such that their first-order derivative is not null at u_0, which allows one to apply the Picard-Lindelöf theorem. □

However, in practice we do not have an anchor point to form the IVP (3.11), making this approach impractical. We propose a strategy to find solutions of the ODE (3.10), which obtains very good candidates for the true curve without explicitly needing anchor points. Our strategy is based on finding the so-called *critical points*. Critical points have special geometric properties, especially a unique solution to depth and normal direction, that we exploit to find the true curve. Critical points can be computed directly from the coefficients of the ODE (3.10).

Critical points. We now give the formal definition of critical points and their properties.

Definition 4 (Critical point definition in θ)**.** *Given a solution* $\hat{\theta}$ *of equation* (3.10)*,* $u_c \in \mathcal{T}$ *is a critical point of* $\hat{\theta}$ *if and only if* $\hat{\theta}'(u_c) = 0$*. Equivalently, from equation* (3.10)*,* $u_c \in \mathcal{T}$ *is a critical point of* $\hat{\theta}$ *if and only if* $\hat{\theta}^2(u_c)\xi(u_c) = 1$*.*

Proposition 3 (Critical point definition in φ)**.** *Given a solution* $\hat{\varphi}$ *to equation* (3.4)*,* $u_c \in \mathcal{T}$ *is a critical point if and only if* $\hat{\varphi}^\top(u_c)\, \mathbf{J}_{\hat{\varphi}}(u_c) = 0$*. An intuitive interpretation is that a critical point is the point on the curve where the tangent and the optical ray are orthogonal.*

Proof. We start by writing $\mathbf{J}_{\bar{\eta}}$ as a function of $\hat{\varphi}$ from equation (3.4):

$$\mathbf{J}_{\bar{\eta}} = \frac{\hat{\varphi}_z \mathbf{J}_{\hat{\varphi}} - \hat{\varphi}'_z \hat{\varphi}}{\hat{\varphi}_z^2}. \tag{3.12}$$

We substitute equation (3.12) in equation (3.10), then express ξ as a function of $\hat{\varphi}$ and $\mathbf{J}_{\hat{\varphi}}$:

$$\begin{aligned}\xi &= \frac{1}{\|\bar{\eta}\|^2}\left(\frac{1}{\hat{\varphi}_z^4}\left(\hat{\varphi}_z\mathbf{J}_{\hat{\varphi}}-\hat{\varphi}'_z\hat{\varphi}\right)^2-\frac{1}{\|\bar{\eta}\|^2}\frac{1}{\hat{\varphi}_z^6}\left(\hat{\varphi}_z\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}-\hat{\varphi}'_z\hat{\varphi}^\top\hat{\varphi}\right)\left(\hat{\varphi}_z\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}-\hat{\varphi}'_z\hat{\varphi}^\top\hat{\varphi}\right)\right)\\ &= \frac{1}{\|\bar{\eta}\|^2}\frac{1}{\hat{\varphi}_z^4}\left(\left(\hat{\varphi}_z\mathbf{J}_{\hat{\varphi}}-\hat{\varphi}'_z\hat{\varphi}\right)^2-\frac{1}{\|\hat{\varphi}\|^2}\left(\hat{\varphi}_z\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}-\hat{\varphi}'_z\hat{\varphi}^\top\hat{\varphi}\right)\left(\hat{\varphi}_z\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}-\hat{\varphi}'_z\hat{\varphi}^\top\hat{\varphi}\right)\right). \quad (3.13)\end{aligned}$$

We expand equation (3.13) and simplify:

$$\begin{aligned}\xi &= \frac{1}{\|\bar{\eta}\|^2}\frac{1}{\hat{\varphi}_z^4}\left(\hat{\varphi}_z^2\mathbf{J}_{\hat{\varphi}}^\top\mathbf{J}_{\hat{\varphi}}-\frac{\hat{\varphi}_z^2}{\|\hat{\varphi}\|^2}\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}\right)\\ &= \frac{1}{\|\bar{\eta}\|^2}\frac{1}{\hat{\varphi}_z^2}\frac{1}{\|\hat{\varphi}\|^2}\left(\hat{\varphi}^\top\hat{\varphi}\mathbf{J}_{\hat{\varphi}}^\top\mathbf{J}_{\hat{\varphi}}-\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}\right). \quad (3.14)\end{aligned}$$

We use definition 4 which gives $\hat{\theta}^2(u_c)\xi(u_c)=1$ if and only if u_c is a critical point. For this, we express $\hat{\theta}^2\xi$ as a function of $\hat{\varphi}$ and $\mathbf{J}_{\hat{\varphi}}$:

$$\begin{aligned}\hat{\theta}^2\xi &= \hat{\varphi}_z^2\,\|\bar{\eta}\|^2\frac{1}{\|\bar{\eta}\|^2}\frac{1}{\hat{\varphi}_z^2}\frac{1}{\|\hat{\varphi}\|^2}\left(\hat{\varphi}^\top\hat{\varphi}\mathbf{J}_{\hat{\varphi}}^\top\mathbf{J}_{\hat{\varphi}}-\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}\right)\\ &= \frac{1}{\|\hat{\varphi}\|^2}\left(\hat{\varphi}^\top\hat{\varphi}\mathbf{J}_{\hat{\varphi}}^\top\mathbf{J}_{\hat{\varphi}}-\mathbf{J}_{\hat{\varphi}}^\top\hat{\varphi}\hat{\varphi}^\top\mathbf{J}_{\hat{\varphi}}\right). \quad (3.15)\end{aligned}$$

We now replace $\hat{\varphi}$ by its three components $\hat{\varphi}_x$, $\hat{\varphi}_y$ and $\hat{\varphi}_z$:

$$\hat{\theta}^2\xi = \frac{1}{\|\hat{\varphi}\|^2}\left(\left(\hat{\varphi}_x^2+\hat{\varphi}_y^2+\hat{\varphi}_z^2\right)\left(\hat{\varphi}_x'^2+\hat{\varphi}_y'^2+\hat{\varphi}_z'^2\right)-\left(\hat{\varphi}_x\hat{\varphi}_x+\hat{\varphi}_y\hat{\varphi}'_y+\hat{\varphi}_z\hat{\varphi}'_z\right)^2\right). \quad (3.16)$$

By expanding equation (3.16) and simplifying, we obtain:

$$\begin{aligned}\hat{\theta}^2\xi &= \frac{1}{\|\hat{\varphi}\|^2}\left(\left(\hat{\varphi}_x\hat{\varphi}'_y+\hat{\varphi}'_x\hat{\varphi}_y\right)^2+\left(\hat{\varphi}_x\hat{\varphi}'_z+\hat{\varphi}'_x\hat{\varphi}_z\right)^2+\left(\hat{\varphi}_y\hat{\varphi}'_z+\hat{\varphi}'_y\hat{\varphi}_z\right)^2\right)\\ &= \frac{\|\hat{\varphi}\times\mathbf{J}_{\hat{\varphi}}\|^2}{\|\hat{\varphi}\|^2}. \quad (3.17)\end{aligned}$$

We now reintroduce u_c to use definition 4:

$$\begin{aligned}(u_c \text{ is a critical point}) &\Leftrightarrow \left(\hat{\theta}^2(u_c)\xi(u_c)=\frac{\|\hat{\varphi}(u_c)\times\mathbf{J}_{\hat{\varphi}}(u_c)\|^2}{\|\hat{\varphi}(u_c)\|^2}=1\right)\\ &\Leftrightarrow \left(\hat{\varphi}^\top(u_c)\;\mathbf{J}_{\hat{\varphi}}(u_c)=0\right). \quad (3.18)\end{aligned}$$

□

In addition, and directly from definition 4, we have that for a critical point u_c:

$$\hat{\theta}(u_c)=\frac{1}{\sqrt{\xi(u_c)}}. \quad (3.19)$$

Computing super critical points from the ODE. A critical point cannot be determined prior to reconstruction. For instance, from definition 4, one has to know $\hat{\varphi}$, and thus the function $\hat{\theta}$, in order to compute the critical points. For this, we propose to compute a superset of the critical points as this set contains all the critical points of $\hat{\varphi}$. To find this superset, we use equation (3.10) and define the *super curve*, denoted by φ_s, as follows.

Definition 5 (Super curve)**.** *The super curve* $\varphi_s \in C^\infty(\mathcal{T}, \mathbb{R}^3)$ *of the ODE* (3.10) *is defined as* $\varphi_s = \frac{1}{\varepsilon\sqrt{\xi}}\bar{\eta}$.

The super curve is obtained by setting $\theta' = 0$ in equation (3.10) and solving the resulting algebraic equation for θ:

$$\xi\theta_s^2 = 1 \qquad \Rightarrow \qquad \theta_s = \frac{1}{\sqrt{\xi}}. \tag{3.20}$$

The *super curve* is not a solution of the ODE (3.10) except for those points where $\theta_s' = 0$, which form the set of *super critical points*.

Proposition 4 (The set of super critical points)**.** *The set of all critical points contained in any solution of the ODE* (3.10) *belongs to the set of critical points of its super curve* φ_s*. We name this set the super critical point set.*

Proof. We first demonstrate that given a solution $\hat{\varphi}$ and critical point u_c, then u_c is also a critical point of φ_s. From definition 4, we have $\hat{\theta}'(u_c) = 0$, which gives:

$$\hat{\theta}(u_c) = \frac{1}{\sqrt{\xi(u_c)}}. \tag{3.21}$$

From equations (3.21) and (3.20), we have $\hat{\theta}(u_c) = \theta_s(u_c)$. Therefore the two curves meet at u_c. To demonstrate that u_c is also a critical point of φ_s, we first differentiate equation (3.10) to obtain the following second-order ODE:

$$2\theta'\theta'' + \xi'\theta^2 + 2\xi\theta\theta' = 0. \tag{3.22}$$

Because $\hat{\theta}$ is a solution to the ODE (3.10), at u_c we have by substituting equation (3.22):

$$\xi'(u_c)\hat{\theta}(u_c)^2 = 0. \tag{3.23}$$

We then differentiate equation (3.20) to obtain the following constraint on φ_s at u_c:

$$\xi(u_c)\theta_s(u_c)\theta_s'(u_c) + \xi'(u_c)\theta_s(u_c)^2 = 0. \tag{3.24}$$

We substitute equation (3.23) into equation (3.24) and use $\hat{\theta}(u_c) = \theta_s(u_c)$ to obtain:

$$\xi(u_c)\theta_s(u_c)\theta_s'(u_c) = 0. \tag{3.25}$$

Because $\hat{\varphi}$ is a solution of the ODE (3.10) and u_c a critical point, we have $\xi(u_c)\hat{\theta}(u_c)^2 = 1$, so $\xi(u_c)$ and $\theta_s(u_c)$ cannot be null. We then have $\theta_s'(u_c) = 0$ and thus u_c is also a critical point of φ_s. □

We now give a way to characterize the super critical points. Precisely, we give three equivalent characterizations. We use them to investigate different methods of super critical point detection.

Proposition 5 (Super critical point identities)**.** *The following are necessary and sufficient conditions for u_s being a super critical point:*
$(\xi'(u_s) = 0) \Leftrightarrow$
$\Big(\|\bar{\eta}(u_s)\|^4 \mathbf{J}_\eta^\top(u_s)\mathbf{H}_\eta(u_s) - \eta^\top(u_s)\mathbf{J}_\eta(u_s)\big(\|\bar{\eta}(u_s)\|^2\eta^\top(u_s)\mathbf{H}_\eta(u_s) +$
$2\|\bar{\eta}(u_s)\|^2\mathbf{J}_\eta^\top(u_s)\mathbf{J}_\eta(u_s) - 2\mathbf{J}_\eta^\top(u_s)\eta(u_s)\eta^\top(u_s)\mathbf{J}_\eta(u_s)\big) = 0\Big)$
$\Leftrightarrow \Big({\varphi_s}^\top(u_s)\mathbf{J}_{\varphi_s}(u_s) = 0\Big)$,
where φ_s is the super curve constructed from equation (3.8).

Proof. Derivation of the first identity. We derive a necessary and sufficient condition on η that is valid at super critical points. We assume $\hat{\varphi}$ is a solution to equation (3.4) with u_s being a super critical point. We first differentiate equation (3.10) to form the following ODE:

$$2\theta'\theta'' + \xi'\theta^2 + 2\xi\theta\theta' = 0. \tag{3.26}$$

We know that $\hat{\theta} = \varepsilon\hat{\varphi}_y$ is a solution to equation (3.26), and $\hat{\theta}'(u_s) = 0$ from definition 4. We substitute $\hat{\theta}$ in equation (3.26) and evaluate the result at u_s, obtaining the following:

$$\xi'(u_s)\hat{\theta}^2(u_s) = 0. \tag{3.27}$$

Derivation of the second identity. We know $\hat{\theta}^2(u_s) \neq 0$, otherwise $\hat{\varphi}$ would pass through the camera's origin at u_s. We also have that $\xi'(u_s) = 0$ from the first super critical point identity. The second identity is found by differentiating ξ as defined in equation (3.10). To express ξ' as a function of η and its derivatives, we first define two intermediate terms, A_η and B_η, and express ξ' using A_η, B_η and their first derivatives:

$$A_\eta = \mathbf{J}_\eta^\top\mathbf{J}_\eta - \frac{1}{\varepsilon^2}B_\eta \quad \text{and} \quad B_\eta = \mathbf{J}_\eta^\top\eta\eta^\top\mathbf{J}_\eta, \tag{3.28}$$

$$A_\eta' = \mathbf{H}_\eta^\top\mathbf{J}_\eta + \mathbf{J}_\eta^\top\mathbf{H}_\eta - \frac{1}{\varepsilon^4}\left(\varepsilon^2 B_\eta' - 2\varepsilon'\varepsilon B_\eta\right), \tag{3.29}$$

$$B_\eta' = \mathbf{H}_\eta^\top\eta\eta^\top\mathbf{J}_\eta + \mathbf{J}_\eta^\top\mathbf{J}_\eta\eta^\top\mathbf{J}_\eta + \mathbf{J}_\eta^\top\eta\mathbf{J}_\eta^\top\mathbf{J}_\eta + \mathbf{J}_\eta^\top\eta\eta^\top\mathbf{H}_\eta. \tag{3.30}$$

Because η, $\mathbf{J}_\eta$ and $\mathbf{H}_\eta$ are $\mathbb{R}^2$-vector, $\mathbf{H}_\eta^\top\eta\eta^\top\mathbf{J}_\eta = \mathbf{J}_\eta^\top\eta\eta^\top\mathbf{H}_\eta$ and $\mathbf{J}_\eta^\top\mathbf{J}_\eta\eta^\top\mathbf{J}_\eta = \mathbf{J}_\eta^\top\eta\mathbf{J}_\eta^\top\mathbf{J}_\eta$, which simplifies B_η':

$$B_\eta' = 2\mathbf{J}_\eta^\top\eta\eta^\top\mathbf{H}_\eta + 2\mathbf{J}_\eta^\top\eta\mathbf{J}_\eta^\top\mathbf{J}_\eta. \tag{3.31}$$

From equations (3.28), (3.29) and (3.31), we have:

$$\begin{aligned}\xi' &= \frac{1}{\varepsilon^4}\left(\varepsilon^2 A'_\eta - 2\varepsilon'\varepsilon A_\eta\right)\\ &= \frac{1}{\varepsilon^4}\left(2\varepsilon^2\mathbf{J}_\eta^\top\mathbf{H}_\eta - B'_\eta + 2\frac{\varepsilon'}{\varepsilon}B_\eta - 2\varepsilon'\varepsilon\mathbf{J}_\eta^\top\mathbf{J}_\eta + 2\frac{\varepsilon'}{\varepsilon}B_\eta\right)\\ &= \frac{1}{\varepsilon^4}\left(2\varepsilon^2\mathbf{J}_\eta^\top\mathbf{H}_\eta - 2\mathbf{J}_\eta^\top\eta\eta^\top\mathbf{H}_\eta - 2\mathbf{J}_\eta^\top\eta\mathbf{J}_\eta^\top\mathbf{J}_\eta + \frac{4}{\varepsilon^2}\eta^\top\mathbf{J}_\eta\mathbf{J}_\eta^\top\eta\eta^\top\mathbf{J}_\eta - 2\eta^\top\mathbf{J}_\eta\mathbf{J}_\eta^\top\mathbf{J}_\eta\right). \quad (3.32)\end{aligned}$$

Using $\eta^\top\mathbf{J}_\eta = \mathbf{J}_\eta^\top\eta$, we obtain:

$$\xi' = \frac{2}{\varepsilon^6}\left(\varepsilon^4\mathbf{J}_\eta^\top\mathbf{H}_\eta - \eta^\top\mathbf{J}_\eta\left(\varepsilon^2\eta^\top\mathbf{H}_\eta + 2\varepsilon^2\mathbf{J}_\eta^\top\mathbf{J}_\eta - 2\mathbf{J}_\eta^\top\eta\eta^\top\mathbf{J}_\eta\right)\right), \quad (3.33)$$

from which we have that $\xi'(u_s) = 0$ is equivalent to:

$$\begin{aligned}&\varepsilon^4(u_s)\mathbf{J}_\eta^\top(u_s)\mathbf{H}_\eta(u_s) - \varepsilon^2(u_s)\eta^\top(u_s)\mathbf{J}_\eta(u_s)\eta^\top(u_s)\mathbf{H}_\eta(u_s)\\ &- 2\eta^\top(u_s)\mathbf{J}_\eta(u_s)\left(\varepsilon^2(u_s)\mathbf{J}_\eta^\top(u_s)\mathbf{J}_\eta(u_s) - \mathbf{J}_\eta^\top(u_s)\eta(u_s)\eta^\top(u_s)\mathbf{J}_\eta(u_s)\right) = 0. \quad (3.34)\end{aligned}$$

By substituting ε and ε' in terms of η and removing factors in equation (3.34) we obtain the following:

$$\begin{aligned}&\|\bar{\eta}(u_s)\|^4\mathbf{J}_\eta^\top(u_s)\mathbf{H}_\eta(u_s) - \|\bar{\eta}(u_s)\|^2\eta^\top(u_s)\mathbf{J}_\eta(u_s)\eta^\top(u_s)\mathbf{H}_\eta(u_s)\\ &- 2\eta^\top(u_s)\mathbf{J}_\eta(u_s)\left(\|\bar{\eta}(u_s)\|^2\mathbf{J}_\eta^\top(u_s)\mathbf{J}_\eta(u_s) - \mathbf{J}_\eta^\top(u_s)\eta(u_s)\eta^\top(u_s)\mathbf{J}_\eta(u_s)\right) = 0. \quad (3.35)\end{aligned}$$

This only depends on η and its derivatives.

Derivation of the third identity. We use the fact that, at any super critical point u_s, $\hat{\varphi}(u_s) = \varphi_s(u_s)$ (definition 5). We then use proposition 3 which says that the critical points of the super curve φ_s are the points where the tangent of φ_s and the line-of-sight are orthogonal. □

Properties of the solutions of the IVP at a super critical point. Suppose we have one super critical point u_s obtained from the *super curve*. We restate the IVP (3.11) using the critical point as the initial condition:

$$\begin{cases}\theta'^2 + \xi\theta^2 = 1\\ \theta(u_s) = \theta_s(u_s).\end{cases} \quad (3.36)$$

According to [Casillas-Perez and Pizarro, 2017], this problem has two analytical solutions in an open domain $]u_s - \epsilon; u_s + \epsilon[$ sufficiently close to the super critical point with $\epsilon > 0$. One solution is $\theta' = \sqrt{1 - \xi\theta^2}$ corresponding to the case where θ' is positive. The second solution is $\theta' = -\sqrt{1 - \xi\theta^2}$, where θ' is negative. If we combine all pairs of branches on either side of the super critical points we obtain four candidate solutions in the local vicinity of u_s, which are C^1 functions [Casillas-Perez and Pizarro, 2017].

Candidate solutions. Suppose that $u_{s_1}, \cdots u_{s_{N_s}}$ is the set of N_s super critical points. We define a candidate solution as the solution of the following Multiple Initial Value Problem (MIVP):

$$\begin{cases} \theta'^2 + \xi\theta^2 = 1 \\ \theta(u_{s_1}) = \theta_s(u_{s_1}) \\ \vdots \\ \theta(u_{s_{N_s}}) = \theta_s(u_{s_{N_s}}). \end{cases} \tag{3.37}$$

We extend the theorem proposed in [Casillas-Perez and Pizarro, 2017] to the case where there are more than one super critical point. We define the interval $\mathcal{I} = [u_{s_1}; u_{s_2}]$, with u_{s_1} and u_{s_2} two consecutive super critical points.

Proposition 6 (Constant sign between two consecutive super critical points)**.** *Given two consecutive super critical points u_{s_1}, u_{s_2} and given a solution $\hat{\theta}$ of equation* (3.37)*, the sign of $\hat{\theta}'$ remains constant in the interval $\mathcal{I} = [u_{s_1}; u_{s_2}]$.*

Proof. $\hat{\theta}$ has continuous derivatives, $\hat{\theta}'(u_{s_1}) = 0$ and $\hat{\theta}'(u_{s_2}) = 0$ by definition 4. Therefore the function $\hat{\theta}'$ cannot change its sign in the interval $\mathcal{I}$ without passing through a super critical point, which contradicts the fact that there are no super critical points in $\mathcal{I}$. □

Using proposition 6, we obtain the following.

Proposition 7 (Number of candidate solutions between two consecutive super critical points)**.** *Given the interval $\mathcal{I} = [u_{s_1}; u_{s_2}]$ between two consecutive super critical points, there exist two candidate solutions, given by the solutions of equation* (3.37) *in $\mathcal{I}$.*

Proof. We use the super critical point u_{s_1} to construct an IVP and then apply the theorem from [Casillas-Perez and Pizarro, 2017] on $\mathcal{I}$. This shows the existence of two solutions in $\mathcal{I}$. Because u_{s_2} is also a super critical point, both solutions, which intersect at u_{s_1}, also pass through u_{s_2}. This is respected regardless of using u_{s_1} or u_{s_2} as initial condition. Thus, the two solutions of both IVPs are the same. □

From proposition 7 we can derive the following bound on the number of solutions of the MIVP (3.37), given the number of super critical points.

Theorem 1 (Number of candidate solutions)**.** *For N_s super critical points in $\mathcal{T}$, there are 2^{N_s+1} candidate solutions to equation* (3.37)*.*

Proof. Proposition 7 tells us that there are two candidate solutions between two consecutive super critical points. Therefore, the solutions to the MIVP (3.37) are composed of pieces of curves that connect at the super critical points and correspond to all the possible combinations. For N_s super critical points, we thus have 2^{N_s+1} possible combinations. □

Conclusions on the theoretical analysis of *Curve SfT-1*. We have proved for the *Curve SfT-1* problem all properties which we presented in §1.4.1. The two main outcomes of this analysis are the non-solvability of *Curve SfT-1* using local non-holonomic PDE and the finding of the critical points in the data itself, which sufficiently constrain the problem so it can be solved up to a finite set of candidate solutions.

3.2.4 *Curve SfT-2*: Reconstructing a 2D Curve from a 1D Image and a 1D Template

The *Curve SfT-2* problem is a straightforward specialization of the *Curve SfT-1* problem. *This specialization does not imply that any extra constraint or information are imposed.* We illustrate the specialization in figure 3.3.

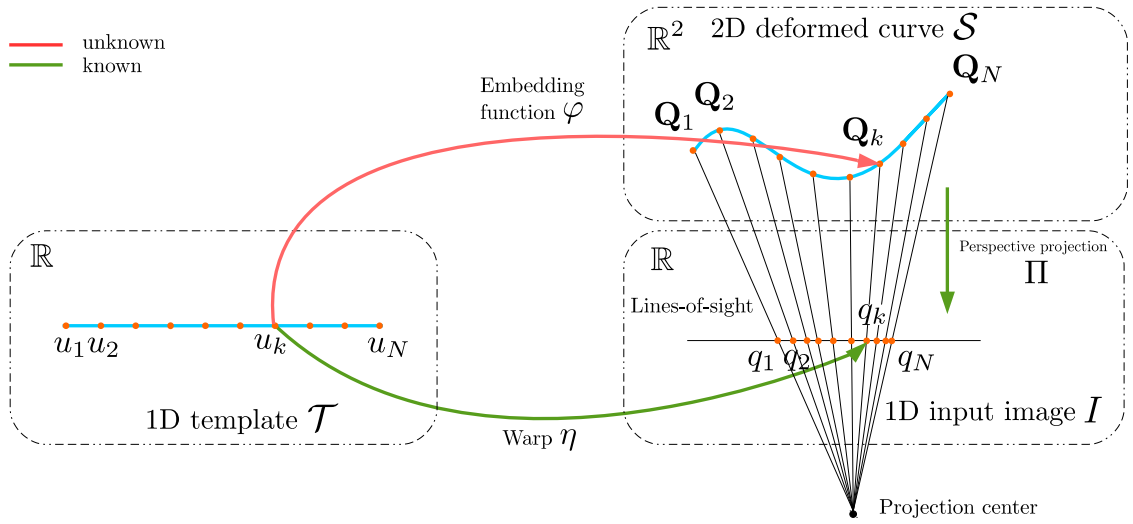


Figure 3.3: General modeling of *Curve SfT-2*.

3.2.4.1 Template and Camera Modeling

Both problems share the same dimension for the known template, $\mathcal{T} \subset \mathbb{R}$. However, the main difference is that the image of the functions, φ, η and Π has a dimension smaller. The template is deformed into an unknown smooth curve $\mathcal{S} \subset \mathbb{R}^2$ embedded in 2D. We parameterize the embedding function by $\varphi = (\varphi_x \; \varphi_y)^\top \in C^\infty(\mathcal{T}, \mathbb{R}^2)$. The 1D input image $I \subset \mathbb{R}$ is modeled as the perspective projection of $\mathcal{S}$ that we denote with a canonical 1D projection function Π:

$$\Pi(\mathbf{Q}) = \frac{x}{y} \quad \text{where} \quad \mathbf{Q} = (x \; y)^\top. \tag{3.38}$$

3.2.4.2 Inputs and Outputs

We now give the inputs and the outputs of *Curve SfT-2*. *(i)* one 1D input image $I : \mathbb{R} \to \{0, 255\}^3$ showing a deforming curve. *(ii)* a template of the surface, defined using §3.2.3.1.

(iii) the camera intrinsics of the perspective 2D projection function Π. *(iv)* a set of N putative 1D-1D correspondences from the texture-map of the template to the input image. We denote the set by $\mathcal{S}_c = \{(u_k, q_k)\}$ where u_k denotes the correspondence position in $\mathcal{T}$ and q_k denotes the correspondence position in the input image I. Details for how correspondences are computed for our experimental datasets are given in §3.5.1.2.

Our solution to *Curve SfT-2* outputs the 2D points of the deformed template in 2D camera coordinates.

3.2.4.3 Theoretical Analysis

We show that *Curve SfT-2* can be formulated as *Curve SfT-1* in problem (3.4). More specifically, it follows the same ODE, which allows us to use all properties given in §3.2.3.3. This leads to the same conclusions as for *Curve SfT-1*: *(i) Curve SfT-2* is not locally solvable exactly and *(ii)*, if $N_s \geq 1$, *Curve SfT-2* has 2^{N_s+1} candidate solutions and an infinite number of solutions otherwise. The differences with *Curve SfT-1* are two-fold: the formula of the ODE coefficients ξ and the critical points definition in φ. We now give these differences.

Problem formulation. Similarly to *Curve SfT-1*, *Curve SfT-2* involves recovering φ, from the warp η and the projection Π using the isometry prior and the reprojection constraints implied by η. The warp $\eta \in C^\infty(\mathcal{T}, \mathbb{R}^1)$ maps the template to a 1D input image. The reprojection constraint is therefore:

$$\eta = \Pi \circ \varphi. \tag{3.39}$$

The *isometry* constraint is a first order differential property in φ:

$$\left\|\varphi'\right\|_2^2 = 1. \tag{3.40}$$

From the constraints (3.39) and (3.40) we define *Curve SfT-2* as follows:

$$\text{Find} \quad \varphi \in C^\infty(\mathcal{T}, \mathbb{R}^2) \quad \text{s.t.} \quad \begin{cases} \eta = \Pi \circ \varphi & \text{(reprojection)} \\ \left\|\varphi'\right\|_2^2 = 1 & \text{(isometry).} \end{cases} \tag{3.41}$$

ODE formulation. We obtain the ODE of *Curve SfT-2* in a similar way as in §3.2.3.3. We first transform the reprojection constraint (3.39) into:

$$\varphi_y \eta = \varphi_x, \tag{3.42}$$

where φ_y is the depth component of φ. We differentiate once, giving:

$$\eta' \varphi_y + \eta \varphi_y' = \varphi_x'. \tag{3.43}$$

By substituting φ_x' from equation (3.43) in the isometry constraint (3.40) and expanding, we arrive at:

$$\varphi_y'^2 \eta^2 + 2\varphi_y \varphi_y' \eta \eta' + \varphi_y^2 \eta'^2 + \varphi_y'^2 = 1. \tag{3.44}$$

This is the same as ODE (3.10). We then perform the change of variable given in §3.2.3.3 in order to study the ODE solutions.

Change of variable. We first define $\varepsilon = \|\bar{\eta}\|$ and thus, $\varepsilon' = \frac{1}{\varepsilon}\eta\eta'$. Introducing ε and ε' in equation (3.44) we have:

$$\left(\varphi'_y\varepsilon + \varphi_y\varepsilon'\right)^2 - \varphi_y^2\varepsilon'^2 + \varphi_y^2\eta'^2 = 1. \tag{3.45}$$

We then define the change of variable:

$$\theta = \varphi_y\varepsilon, \tag{3.46}$$

which allows us to transform equation (3.45) into one depending on θ and θ':

$$\theta'^2 + \xi\theta^2 = 1 \quad \text{with} \quad \xi = \frac{\eta'^2}{\varepsilon^4}. \tag{3.47}$$

Finally, two explicit ODEs can be derived from equation (3.47):

$$\theta' = \pm\sqrt{1 - \xi\theta^2}. \tag{3.48}$$

Given a solution to equation (3.47) we recover a solution of the original ODE (3.44) by simply inverting the change of variable of equation (3.46).

Critical points. We give two propositions regarding the critical points, which differ from *Curve SfT-1*. The reason is that the expression of ξ is different in *Curve SfT-2*.

Proposition 8 (Critical point definition in φ)**.** *Given a solution $\hat{\varphi}$ to equation* (3.41)*, u_c is a critical point if and only if $\hat{\varphi}^\top(u_c)\ \hat{\varphi}'(u_c) = 0$. An intuitive interpretation is that a critical point is the point on the curve where the tangent and the optical ray are orthogonal.*

Proof. We start by writing η' in function of $\hat{\varphi}$ from equation (3.39):

$$\eta' = \frac{\hat{\varphi}'_x\hat{\varphi}_y - \hat{\varphi}_x\hat{\varphi}'_y}{(\hat{\varphi}_y)^2}. \tag{3.49}$$

We substitute equation (3.49) in equation (3.47) and equation (3.46) to express ξ and $\hat{\theta}$:

$$\xi = \frac{\left(\hat{\varphi}'_x\hat{\varphi}_y - \hat{\varphi}_x\hat{\varphi}'_y\right)^2}{\left(\hat{\varphi}_x^2 + \hat{\varphi}_y^2\right)^2} \tag{3.50}$$

$$\hat{\theta} = \sqrt{\hat{\varphi}_x^2 + \hat{\varphi}_y^2}. \tag{3.51}$$

We use definition 4 which gives $\hat{\theta}^2(u_c)\xi(u_c) = 1$ if and only if u_c is a critical point. For this, we express $\hat{\theta}^2\xi$ as a function of $\hat{\varphi}$ and $\hat{\varphi}'$:

$$\begin{aligned}\hat{\theta}^2\xi &= \frac{\left(\hat{\varphi}'_x\hat{\varphi}_y - \hat{\varphi}_x\hat{\varphi}'_y\right)^2}{\hat{\varphi}_x^2 + \hat{\varphi}_y^2} \\ &= \frac{\|\hat{\varphi}\times\hat{\varphi}'\|^2}{\|\hat{\varphi}\|^2}.\end{aligned} \tag{3.52}$$

We now reintroduce u_c to use definition 4:

$$\begin{aligned}(u_c \text{ is a critical point}) &\Leftrightarrow \left(\hat{\theta}^2(u_c)\xi(u_c) = \frac{\|\hat{\varphi}(u_c)\times\hat{\varphi}'(u_c)\|^2}{\|\hat{\varphi}(u_c)\|^2} = 1\right) \\ &\Leftrightarrow \Big(\hat{\varphi}^\top(u_c)\ \hat{\varphi}'(u_c) = 0\Big).\end{aligned} \tag{3.53}$$

□

Similarly to definition 5 and proposition 4, we now introduce the notion of super curve and the set of super critical points. We also give three equivalent ways to characterize the super critical points.

Proposition 9 (Super critical point identities)**.** *A point u_s is a super critical point of equation* (3.41) *if and only if it is a solution of one of the following equations:*
$(\xi'(u_s) = 0) \Leftrightarrow \big(2\eta(u_s)\eta'^2(u_s) - \big(1+\eta^2(u_s)\big)\,\eta''(u_s) = 0\big) \Leftrightarrow \big(\hat{\varphi}^\top(u_s)\ \hat{\varphi}'(u_s) = 0\big)$,
with φ_s the super curve of equation (3.47).

Proof. We follow the same steps as the proof of proposition 4 and obtain that $\xi'(u_s) = 0$, which is the first characterization. A second one can be found by differentiating the analytical expression of ξ given in equation (3.47):

$$\xi' = \frac{2\varepsilon^4\eta'\eta'' - 4\varepsilon^3\varepsilon'\eta'^2}{\varepsilon^8}, \tag{3.54}$$

from which we have that $\xi'(u_s) = 0$ is equivalent to:

$$\frac{\varepsilon^3}{\eta'}\left(\varepsilon(u_s)\eta''(u_s) - 2\varepsilon'(u_s)\eta'(u_s)\right) = 0. \tag{3.55}$$

By substitution of ε and ε' in terms of η and removing factors in equation (3.55) we have the second identity:

$$2\eta(u_s)\eta'^2(u_s) - (1+\eta^2(u_s))\eta''(u_s) = 0, \tag{3.56}$$

which only depends on η and its derivatives.

For the third characterization, we use the fact that, at any super critical point u_s, $\hat{\varphi}(u_s) = \varphi_s(u_s)$ (definition 5). We then use proposition 8 and obtain that the critical points of the super curve φ_s are the points where the tangent of φ_s and the line-of-sight are orthogonal. □

3.3 The Number of Solutions

We have shown that these are very under-constrained problems, described with a single ODE which has an infinite number of solutions. Adding anchor points from the true curve makes the corresponding IVP well-constrained, but it remains impractical. The main contribution of this paper is a method, the HMM solution, for finding good candidates of the true curve using only the ODE coefficients. Our method implies finding the so-called critical points and solving efficiently an MIVP which limits the true curve to belong to a discrete number of candidate solutions. In our experiments, we show that this strategy almost always finds the true curve, or a curve very close to it. Intuitively our strategy forces the candidate solutions to be smooth as it forces the first derivatives of the depth variable to be small at the super critical points. This also pushes the candidate solutions away from the camera center as our ODE are sum-of-squares of the depth and its first derivatives. In addition we empirically observed that candidate solutions that do not pass through many of the super critical points tend to be very non-smooth and tend to come very close to the camera center.

In Curve SfT, we face two fundamental cases. In the first case, the true curve does not have any critical point. It is then not recoverable. Mathematically, the ODE (3.10) has an infinite number of solutions and cannot be upgraded into an IVP. In the second case, the true curve has at least one critical point. It is then recoverable up to a finite number of ambiguities. Mathematically, the ODE (3.10) has an infinite number of solutions, but can be upgraded into the MIVP (3.37). In order to guarantee that the true curve belongs to the recovered set of curves, we thus have to enumerate all possible subsets of super critical points. These subsets give the *extended set of candidate solutions*, which we formally define as follows:

Definition 6 (Extended set of candidate solutions)**.** *Given N_s super critical points in $\mathcal{T}$, the extended set of candidate solutions is the set of all candidate solutions of the ODE* (3.10) *which pass through a combination of the N_s super critical points.*

From theorem (1), we compute the size of the extended set of candidate solutions as follows.

Proposition 10 (Size of the extended set of candidate solutions)**.** *Given N_s super critical points in $\mathcal{T}$, the size of the extended set of candidate solutions is $M_{ext} = 2\left(3^{N_s}\right) - 2$.*

Proof. For each subset of i super critical points selected from the set of super critical points, the number of combinations is given by the binomial coefficient $\binom{N_s}{i}$. From theorem (1), we know that, for each subset of i super critical points, we have 2^{i+1} candidate solutions of the MIVP (3.37). Then, the number of candidate solutions considering all possible subsets is $M_{ext} = \sum_{i=1}^{N_s}\binom{N_s}{i}2^{i+1} = N_s 2^2 + ... + N_s 2^{N_s} + 2^{N_s+1}$. This can be simplified using the combinatorial binomial theorem which gives that for $n \geq 0$ and $(x, y) \in \mathbb{R}^2$, $(x+y)^n = \sum_{i=0}^{n}\binom{n}{i}x^i y^{n-i}$. We then obtain $M_{ext} = 2\Big(\sum_{i=0}^{N_s}\binom{N_s}{i}2^i - \binom{N_s}{0}2^0\Big) = 2\Big((2+1)^{N_s} - 1\Big) = 2\left(3^{N_s}\right) - 2$. □

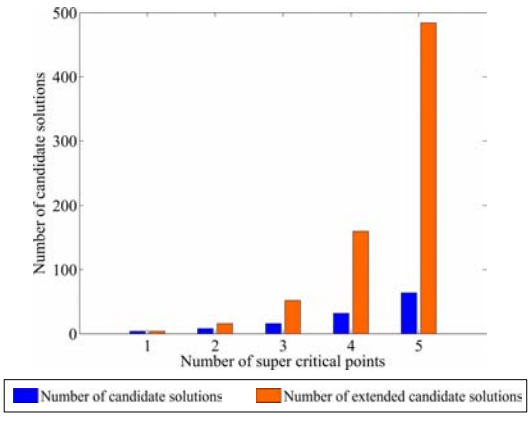
Considering all possible subsets of super critical points, we now give a condition of the recoverability of the true curve and the set of candidate solutions in which the true curve is contained.

Theorem 2 (Recoverability of the true curve)**.** *Given the ODE* (3.10) *and* $N_s \in \mathbb{N}$ *super critical points in* $\mathcal{T}$*,*

- *if the true curve does not pass through any critical point, it is not recoverable*
- *if the true curve passes through* $1 \leq n_s \leq N_s$ *critical points, the true curve is recoverable up to a finite number of ambiguities and is contained in the extended set of candidate solutions.*

Proof. We start with the case where the true curve does not pass through any critical point. As setting an *anchor point* as the initial condition of the IVP (3.11) requires additional information, we do not have any way to set an initial condition. Then we cannot recover the true curve from the ODE (3.10). We now consider the case where the ODE (3.10) has $N_s \geq 1$ super critical points and the true curve passes through $1 \leq n_s \leq N_s$ critical points. We can form the MIVP (3.37) with n_s initial conditions. The true curve is then recoverable and, from definition (6), the true curve is contained in the extended set of candidate solutions. □

From proposition 10, we have that the extended set of candidate solutions is much larger than the set of candidate solutions given by theorem 1. Figure 3.4 compares how grows the number of both sets as function of the number of super critical points. We note that the extended set of candidate solutions grows importantly with the number of super critical points, compared to the set of candidate solutions. Therefore, considering the extended set of candidate solutions for implementation may massively increase the computational burden. This is why we consider the set of candidate solutions in §3.4.



(a)

Number of super critical points	1	2	3	4	5
Number of candidate solutions	4	8	16	32	64
Number of extended candidate solutions	4	16	52	160	484

(b)

Figure 3.4: Size of the extended and normal set of candidate solutions as function of the number of super critical points. (a) bar-plots of the set sizes as function of the number of super critical points, (b) associated values of the set sizes.

3.4 Computational Solutions

As mentioned in §2.2.4, three categories of method have been proposed to solve SfT. We propose here a method for the three categories and a four method from a new category which uses a graphical model. For each category of method, we first describe in detail the method for *Curve SfT-1* and then give the specialization to *Curve SfT-2*.

In the theoretical analysis, we use a general function for the template's shape model. However, such model cannot be used for these four categories. We propose then to adapt the template's shape model for each category. We follow here the same notation of SfT inference categories given in §2.2.4. We now give the specific shape model of the template for each category: a differentiable function for category *(i)*, a point set for category *(ii)* and an angle-based parameterization (presented in §3.4.3.1) for category *(iii)*. For the new category *(iv)* based on graphical model, the shape is modeled by a chain.

3.4.1 Single-Solution Methods (Categories *(i)* and *(ii)*)

The following methods are fast and simple solutions to Curve SfT, however they have limited practical use because they compute only a single candidate solution, which may be the wrong one. We describe them for completeness as they follow directly from existing category *(i)* and *(ii)* methods for Surface SfT.

3.4.1.1 A Category *(i)* Method

Case Curve SfT-1. We consider non-holonomic solutions and assume that the deformed curve $\mathcal{S}$ is infinitesimally linear. This is equivalent to $\mathcal{S}$ being a succession of infinitesimal lines, *i.e.* to consider $\mathbf{J}_\varphi(u) = \mathbf{0}_{3\times 1}, \forall u \in \mathcal{T}$. By differentiating equation (3.6) and substituting, we obtain:

$$\varphi_z \mathbf{J}_{\bar{\eta}} + \varphi'_z \bar{\eta} = \mathbf{0}_{3\times 1} \quad \Rightarrow \quad \varphi'_z = -\varphi_z \mathbf{J}_{\bar{\eta}}^\top \bar{\eta}^{-1}. \tag{3.57}$$

By substituting φ'_z from equation (3.8) in equation (3.57), we obtain two solutions for φ_z:

$$\varphi_z = \pm \frac{1}{\sqrt{\mathbf{J}_{\bar{\eta}}^\top \mathbf{J}_{\bar{\eta}} - \bar{\eta}^\top \mathbf{H}_{\bar{\eta}} + \frac{1}{4}\bar{\eta}^\top \mathbf{H}_{\bar{\eta}} \mathbf{J}_{\bar{\eta}}^{-1^\top} \mathbf{J}_{\bar{\eta}}^{-1} \mathbf{H}_{\bar{\eta}}^\top \bar{\eta}}}. \tag{3.58}$$

The infinitesimal linearity assumption can be generalized to higher orders assuming $\varphi^{(k)}(u) = \mathbf{0}_{3\times 1}, \forall u \in \mathcal{T}$. This makes φ locally polynomial of finite order. However, unlike for infinitesimal linearity, for $k > 2$, analytical solutions are difficult, but not impossible, to find.

Specialization to Curve SfT-2. Similarly to *Curve SfT-1*, we obtain the analytical solution by differentiating equation (3.43) and assuming infinitesimal planarity, *i.e.* $\varphi''(u) = \mathbf{0}_{2\times 1}$. This produces the solutions:

$$\varphi_y = \pm \frac{2\eta'}{\sqrt{(-\eta''\eta + 2\eta'^2)^2 + (-\eta'')^2}}. \tag{3.59}$$

For both cases, we discard the negative solution because it is behind the camera.

3.4.1.2 A Category *(ii)* Method

Case Curve SfT-1. We model the problem as an SOCP optimization. This is a direct adaptation of the so-called MDH [Perriollat et al., 2011; Salzmann and Fua, 2009], which we presented in chapter 2. We discretize the template into a chain of N nodes and place the nodes at the correspondence points. We set the positions of each node in the input image at the 2D image correspondences, denoted by $\mathbf{q}_i$, $i \in [1, N]$. The unknowns are then the set of depths $d_i \in \mathbb{R}^+$ at each node. Any pair of nodes is constrained by the following *inextensibility constraint*:

$$\forall (i,j) \in [1,N]^2 \text{ with } i \neq j, \quad \left\| d_i(\mathbf{q}_i^\top \; 1)^\top - d_j(\mathbf{q}_j^\top \; 1)^\top \right\| \leq l_{ij}. \tag{3.60}$$

where $l_{ij} \in \mathbb{R}^+$ denotes the geodesic distance between nodes i and j, known from the template. The problem is then cast as an SOCP by searching for the maximal depth of each node such that equation (3.60) is satisfied.

Specialization to Curve SfT-2. The extension of the MDH to *Curve SfT-2* is trivial and follows the formulation (3.60) for $\text{SfT}^{1\to2\to1}$ where, for each $i \in [1, N]$, q_i is now a 1D point.

3.4.2 A Multi-Solution Method with HMM (Category *(iv)*)

3.4.2.1 Overview

We model the problem as a discrete HMM. This overcomes the limitations of category *(i)* and *(ii)* methods because *all* candidate solutions are reconstructed. Note that from now we restrict our search to the set of candidate solutions given in theorem 1 and *not* the extended set of candidate solutions, presented in §3.3. This is motivated by two reasons: computing the extended set of candidate solutions increases massively the computational time of the category *(iv)* method and, in practice, we found most of the time that the set of candidate solutions contains a solution very close to the ground-truth.

Our multi-solution method is based on two main remarks:

- Super critical points separate the curve $\mathcal{S}$ (and then the 1D template) into contiguous pieces
- Proposition 7 tells us that each piece is recoverable up to a two-fold ambiguity.

Our method is illustrated by figure 3.5. From the estimated warp η, we detect the N_s super critical points using a method described in §3.4.2.2. We use $u_j^s \in \mathbb{R}$ to denote the position of the j^{th} super critical point in the template. We use d_j^s to denote its corresponding depth. From proposition 6 we know that between two consecutive super critical points the sign of θ' is constant. We then reconstruct the full template by specifying a sign combination vector $\mathbf{s} = \{s_i\}_{i\in[1,N_s+1]} \in [-1,+1]^{N_s+1}$, where s_i denotes a selection for either the positive solution $\theta' = \sqrt{1-\xi\theta^2}$ or the negative solution $\theta' = -\sqrt{1-\xi\theta^2}$. When N_s is small it is feasible to generate all candidate solutions using every possible $\mathbf{s}$. This is the case with our experimental

data where N_s is typically lower than 12. When this is not the case, the template can be generated on demand with a specific $\mathbf{s}$.

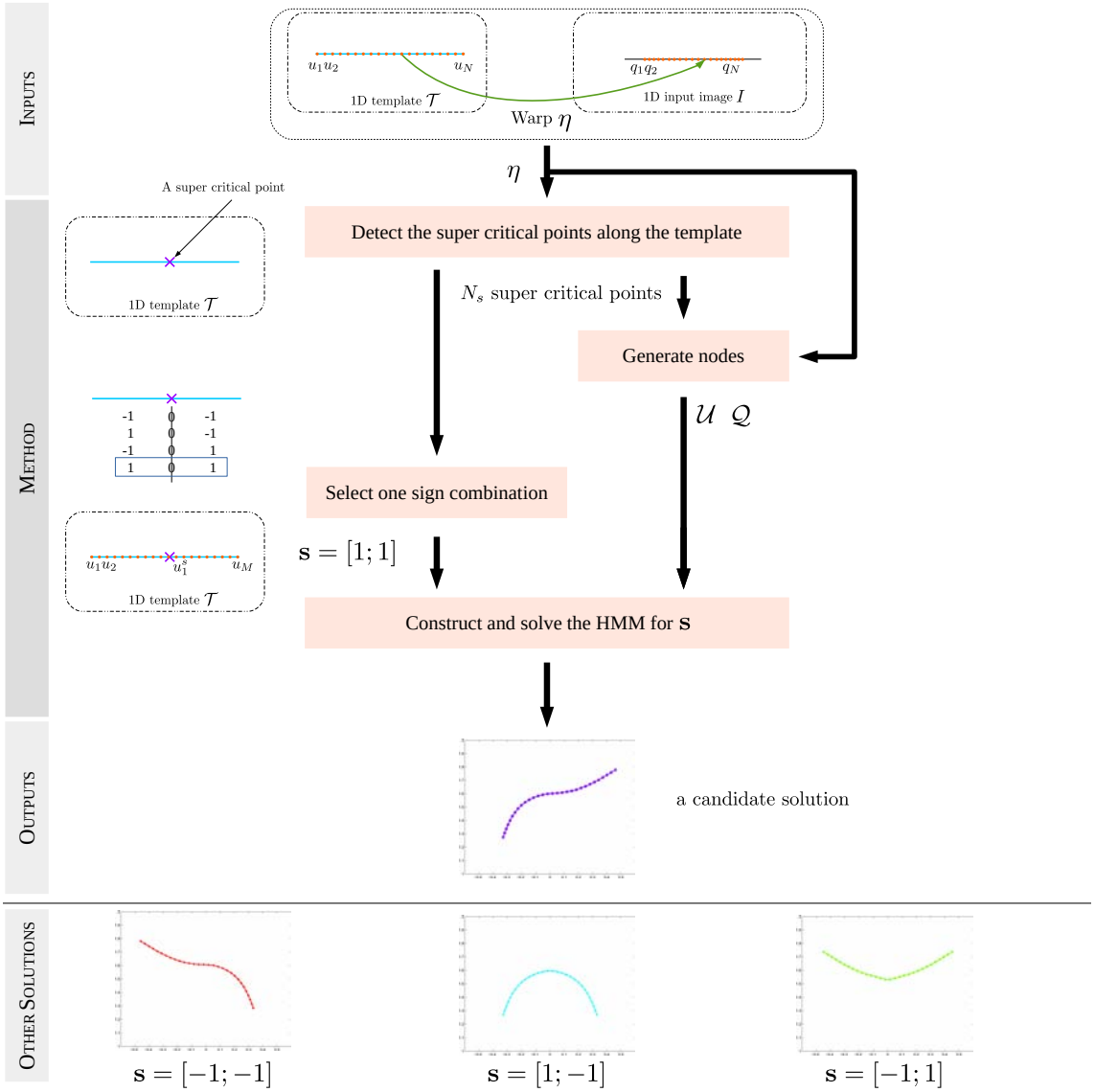


Figure 3.5: Reconstruction pipeline of our proposed category *(iv)* method. **Top:** proposed category *(iv)* method for solving *Curve SfT-2* and so 4 candidate solutions. We illustrate the pipeline with an example where there is $N_s = 1$ super critical point. **Bottom:** we give the other candidate solutions, which are obtained by selecting different sign combinations $\mathbf{s}$.

3.4.2.2 Super Critical Point Detection

All equivalent characterizations of a super critical point given by proposition 4 can be used for detection. They only require one to know the warp η. We have found the method using the roots of ξ' to be the most accurate and the most stable.

3.4.2.3 Graphical Modeling

We setup the HMM as follows. We generate its nodes by discretizing the template into $M + N_s$ 1D points: $\mathcal{U} = \{u_1, u_2, \dots u_{M+N_s}\}$. These are made by combining the N_s super critical point positions u_j^c with M uniformly sampled template points spanning the whole template. We order these such that $u_{i+1} \geq u_i$. The position of each node in the input image is denoted by $\mathcal{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_{M+N_s}\} \in \mathbb{R}^{M+N_s}$. These are computed using the warp η.

The state of each node holds its unknown depth $d_i \in \mathbb{R}$. We draw d_i from a discrete set of D depth samples, denoted by $\mathcal{D} \in \mathbb{R}^+$. We discuss how $\mathcal{D}$ is created in §3.4.2.5.

The graph's edges are constructed between consecutive nodes, producing $M + N_s - 1$ edges. We use the set $\mathcal{E}_i$ to denote the neighbors of node i. We define the graph's energy using first and second-order potentials. This has the following form:

$$E(\{d_i\}; \mathbf{s}, \mathcal{U}, \mathcal{Q}) = E_{iso}(\{d_i\}; \mathcal{U}, \mathcal{Q}) + \lambda_{cp} E_{cp}(\{d_i\}; \mathcal{Q}) + E_{sign}(\{d_i\}; \mathbf{s}, \mathcal{U}, \mathcal{Q}), \tag{3.61}$$

The energy term E_{iso} denotes the *isometric energy*, and is a second-order energy between consecutive nodes that enforces the isometry prior. The energy term E_{cp} denotes the *critical point energy*, and is a second-order energy between super critical point nodes and their neighboring nodes. This energy forces the tangent at each super critical point node to be orthogonal its line-of-sight (proposition 3). The last energy term E_{sign} is a first-order energy and forces the gradient signs of the super critical point nodes to agree with a particular sign combination vector $\mathbf{s}$.

3.4.2.4 Energy Definitions

The energy term E_{iso} is defined as follows:

$$E_{iso}(\{d_i\}; \mathcal{U}, \mathcal{Q}) = \sum_{i=1}^{M+N_s} \sum_{j \in \mathcal{E}_i} \left(\left\| d_i \left(\mathbf{q}_i^\top \; 1 \right)^\top - d_j \left(\mathbf{q}_j^\top \; 1 \right)^\top \right\| - l_{i.j} \right)^2, \tag{3.62}$$

where $l_{i,j} = |u_i - u_j|$ denotes the distance in the template domain between two neighboring nodes. This prevents the curve from stretching or compressing.

The energy term E_{cp} acts only at super critical point nodes. Suppose node i is a super critical point node. Its critical point energy is as follows:

$$E_{cp}^i = \sum_{j \in \mathcal{E}_i} \left| \left(\frac{d_i \left(\mathbf{q}_i^\top \; 1 \right)^\top - d_j \left(\mathbf{q}_j^\top \; 1 \right)^\top}{\left\| d_i \left(\mathbf{q}_i^\top \; 1 \right)^\top - d_j \left(\mathbf{q}_j^\top \; 1 \right)^\top \right\|} \right)^\top \frac{d_i \left(\mathbf{q}_i^\top \; 1 \right)^\top}{\left\| d_i \left(\mathbf{q}_i^\top \; 1 \right)^\top \right\|} \right|. \tag{3.63}$$

This term models the curve gradient using a finite difference and computes the dot product between the super critical point node gradient and the line-of-sight at the super critical point. We compute E_{cp} as the sum of equation (3.63) over all super critical point nodes.

The energy term E_{sign} also only acts at super critical point nodes. Suppose node i is

a super critical point node with sign selection s_i. The sign energy forces the sign of the gradient of θ'_i for node i to agree with the sign selection s_i. For this, it uses equation (3.9) and the sign selection s_i. For instance, we know that $\theta'_i < 0$, *i.e.* $s_i = -1$, implies that $d_i \left\| \left(\mathbf{q}_i^\top \; 1\right)^\top \right\| > d_{i+1} \left\| \left(\mathbf{q}_{i+1}^\top \; 1\right)^\top \right\|$. Then the sign energy is as follows:

$$E^i_{sign} = \sum_{j \in \mathcal{E}_i} \infty \Big| \text{sign}\,(u_i - u_j)\, \text{sign}\left(\hat{d}_i \left\| \left(\mathbf{q}_i^\top \; 1\right) \right\| - \hat{d}_j \left\| \left(\mathbf{q}_j^\top \; 1\right) \right\| \right) - s_i \Big|. \tag{3.64}$$

3.4.2.5 Depth Discretization

We define the lower and upper bounds on depth and uniformly sample the bounds using $D = 500$ intervals. The upper bound d_{max} is generated using the category *(ii)* method. We define $d_{max} = \max(\{d_i\})$, with $\{d_i\}$ the depth of all correspondences estimated by the category *(ii)* method. We set the lower bound to $d_{min} = f$, with f the camera focal length. For this, we assume that the curve is beyond the focal length, which is a reasonable assumption in practice.

3.4.2.6 Inference

Because our graph is a chain, global inference can be performed with the Viterbi algorithm [Rabiner, 1989]. Thus, we obtain a candidate solution whose θ' function follows the sign combination $\mathbf{s}$, as figure 3.5 illustrates.

3.4.2.7 Specialization to *Curve SfT-2*

The adaption of the full reconstruction pipeline based on HMM is straightforward. As we formulate the reconstruction problem using only depths as unknowns, the HMM energies can be trivially extended to the reconstruction of 2D curves. All other components of the method are kept similar. In figure 3.5, we present reconstructions in the case of *Curve SfT-2* for a simpler understanding of the output from the category *(iv)* method. Figure 3.5 (bottom) also shows the other candidate solutions with respect to each possible sign combination.

3.4.3 Solution Refinement (Category *(iii)*)

We propose a non-convex cost function that models Curve SfT, by balancing the reprojection error with a smoothing prior. Isometry is enforced implicitly using a novel angle-based parameterization. This tends to generate the most accurate results but requires suitable initialization. We provide this with the HMM solution. Figure 3.6 gives an overview of the method.

3.4.3.1 Angle-Based Parameterization

The exact enforcement of isometry has two advantages. First, we do not need to balance it with reprojection and smoothing terms in the cost function, because isometry is always en-

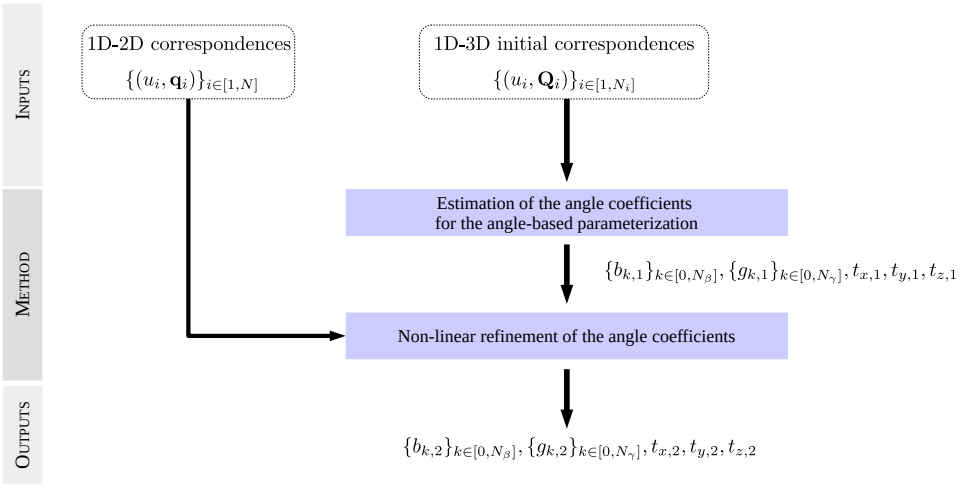


Figure 3.6: Proposed refinement method for solving *Curve SfT-1*.

forced. Second, it reduces the number of optimization variables. However, this parameterization presents one limitation: it cannot model quasi-isometry. We define the parameterization using a spherical parameterization with two angle functions $\beta : \mathcal{T} \to \mathbb{R}$ and $\gamma : \mathcal{T} \to \mathbb{R}$:

$$\forall u_i \in \mathcal{T}, \quad \varphi(u_i; \mathbf{b}, \mathbf{g}, t_x, t_y, t_z) = \sum_{j=2}^{i} l_{j-1,j} \begin{pmatrix} \sin\beta(u_{j-1}; \mathbf{b}) \ \cos\gamma(u_{j-1}; \mathbf{g}) \\ \sin\beta(u_{j-1}; \mathbf{b}) \ \sin\gamma(u_{j-1}; \mathbf{g}) \\ \cos\beta(u_{j-1}; \mathbf{b}) \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}, \quad (3.65)$$

with $l_{j-1,j}$ the length in the template between the $(j-1)^{th}$ and j^{th} correspondences. Figure 3.7 illustrates the angle-based parameterization. We construct the angle functions β and γ using respectively a degree N_β polynomial and a degree N_γ polynomial:

$$\forall u_j \in \mathcal{T}, \quad \beta(u_j; \mathbf{b}) = \sum_{k=0}^{N_\beta} u_j^k b_k \quad \text{and} \quad \gamma(u_j; \mathbf{g}) = \sum_{k=0}^{N_\gamma} u_j^k g_k, \quad (3.66)$$

with $\mathbf{b} = \{b_k\}_{k\in[0,N_\beta]}$ the set of polar angle coefficients and $\mathbf{g} = \{g_k\}_{k\in[0,N_\gamma]}$ the set of azimuthal angle coefficients. φ is defined by $N_\beta + N_\gamma + 5$ parameters: the polar angle coefficients $\mathbf{b} = \{b_k\}_{k\in[0,N_\beta]}$, the azimuthal angle coefficients $\mathbf{g} = \{g_k\}_{k\in[0,N_\gamma]}$ and the 3D translation parameters t_x, t_y and t_z.

We now explain how to compute $\mathbf{b}$, $\mathbf{g}$, t_x, t_y and t_z from the N_i initial 1D-3D correspondences which can be obtained from categories *(i)*, *(ii)* and *(iv)* methods. The value of N_i may differ with respect to the method used to initialize: for instance, for our methods from categories *(i)* and *(ii)* $N_i = N$, and for our method from category *(iv)* $N_i = M + N_s$ as explained in §3.4.2.3. We first set the translation vector to the first 3D point of the curve. For the angle coefficients, we compute analytically the polar angle and the azimuthal angle between two 3D points using the trigonometric functions. Then, we fit the sought degree N_β and degree N_γ polynomials to the estimated polar and azimuthal angles respectively.

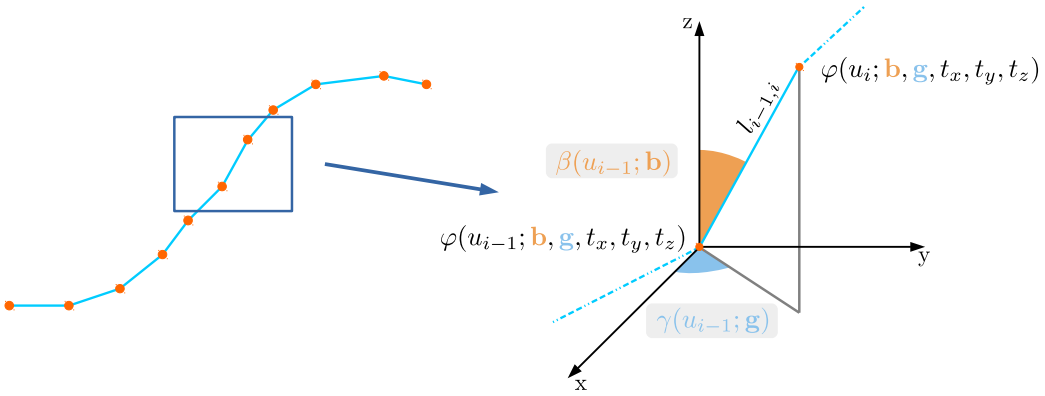


Figure 3.7: The angle-based parameterization for 3D curve. **Left**: a 3D curve parameterized by the angle-based parameterization (3.65). **Right**: a zoom on the angle-based parameterization. β refers to the polar angle and γ to the azimuthal angle.

3.4.3.2 Refinement

Case Curve SfT-1. The refinement is performed by non-linear least-squares optimization of a cost function containing a *reprojection* and a *smoothing* constraints:

$$C(\mathbf{b}, \mathbf{g}, t_x, t_y, t_z) = C_{reproj}(\mathbf{b}, \mathbf{g}, t_x, t_y, t_z) + \lambda_{smooth} C_{smooth}(\mathbf{b}, \mathbf{g}), \tag{3.67}$$

$$\text{with} \quad C_{reproj}(\mathbf{b}, \mathbf{g}, t_x, t_y, t_z) = \frac{1}{N} \sum_{i=1}^{N} \left\| \Pi \circ \varphi(u_i; \mathbf{b}, \mathbf{g}, t_x, t_y, t_z) - \mathbf{q}_i \right\|^2,$$

$$\text{and} \quad C_{smooth}(\mathbf{b}, \mathbf{g}) = \frac{1}{N-1} \left(\sum_{i=2}^{N} \Big(\beta(u_i; \mathbf{b}) - \beta(u_{i-1}; \mathbf{b}) \Big)^2 + \Big(\gamma(u_i; \mathbf{g}) - \gamma(u_{i-1}; \mathbf{g}) \Big)^2 \right),$$

where $\lambda_{smooth} \geq 0$ is the smoothing weight. This is solved using Levenberg-Marquardt.

Specialization to Curve SfT-2. The specialization simplifies the angle-based parameterization and the cost function. For the angle-based parameterization, we define φ using only one angle function $\alpha : \mathcal{T} \to \mathbb{R}$:

$$\forall u_i \in \mathcal{T}, \quad \varphi(u_i; \mathbf{a}, t_x, t_y) = \sum_{j=2}^{i} l_{j-1,j} \begin{pmatrix} \cos \alpha(u_{j-1}; \mathbf{a}) \\ \sin \alpha(u_{j-1}; \mathbf{a}) \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{3.68}$$

where $l_{j-1,j}$ is the length in the template between the $(j-1)^{th}$ and j^{th} correspondences, $\mathbf{a} = \{a_k\}_{k \in [0, N_\alpha]}$ the set of angle coefficients and $\alpha(u_{j-1}; \mathbf{a}) = \sum_{k=0}^{N_\alpha} u_{j-1}^k a_k$, a degree N_α polynomial. φ is then defined by $N_\alpha + 3$ parameters: the angle coefficients $\mathbf{a} = \{a_k\}_{k \in [0, N_\alpha]}$ and the 2D translation parameters t_x and t_y. The cost function includes a smoothing which penalizes non-smooth variations:

$$C(\mathbf{a}, t_x, t_y) = C_{reproj}(\mathbf{a}, t_x, t_y) + \lambda_{smooth} C_{smooth}(\mathbf{a}), \tag{3.69}$$

$$\text{with} \quad C_{reproj}(\mathbf{a}, t_x, t_y) = \frac{1}{N} \sum_{i=1}^{N} \left\| \Pi \circ \varphi(u_i; \mathbf{a}, t_x, t_y) - q_i \right\|^2,$$

$$\text{and} \quad C_{smooth}(\mathbf{a}) = \frac{1}{N-1}\sum_{i=2}^{N}\Big(\alpha(u_i;\mathbf{a}) - \alpha(u_{i-1};\mathbf{a})\Big)^2.$$

3.5 Experimental Validation

3.5.1 *Curve SfT-2* Experiments

3.5.1.1 Methods

We evaluate thoroughly the category *(iv)* method with and without refinement. As categories *(i)* and *(ii)* methods are single-solution methods, we give only some results for the category *(ii)* method, showing that we can rule it out. The category *(iv)* method is denoted **2DHMM** and the category *(iv)* method followed by iterative refinement is denoted **2DHMM+REF**. The category *(ii)* method, without and with refinement, is denoted respectively **2DMDH** and **2DMDH+REF**.

All methods are implemented in Matlab. We use YALMIP [Löfberg, 2004] and SeDuMi [Sturm, 1999] to implement the category *(ii)* method. We use [Schmidt, 2007] to construct and solve the HMMs (category *(iv)*). We perform the non-linear refinement (category *(iii)*) using the Matlab function `lsqnonlin`. We use the curve fitting toolbox of Matlab to detect the critical points as explained in §3.4.2.2.

We compute the template-to-image warp η using an interpolation function and a set of N 1D correspondence points between the template and the input image. We construct η using a spline which is obtained by the Matlab function `spaps` and the curve fitting toolbox. Table A.1 in appendix A gives the hyperparameters for each method for the sake of reproducibility.

3.5.1.2 Datasets

Simulated datasets. We evaluated performance with two simulated datasets: the *convex-to-concave* dataset and the *free-form* dataset. The *convex-to-concave* dataset consists of 14 input images that were generated by fixing the middle point of the template at a same depth and in front of the camera center and by decreasing the curvature of the embedded curve in 14 increments, going from convex to concave. Input images 1 to 7 show convex examples and 8 to 14 show concave examples. The *free-form* dataset consists of 20 input images generated by isometrically deforming the template using a degree 5 polynomial. For both datasets, we used a 1D template with unit length, simulated 30 correspondences for each input image and added gaussian noise to theses correspondences with a standard deviation of 1.0 px. We show some curves of both datasets in figure 3.10. Critical points are computed by finding where the first derivative of the ground-truth function θ is equal to zero (definition 4). We show in figure 3.8 the simulated 2D curves for both datasets.

Real datasets. We tested our methods on two real datasets: the *paper* and the *cable* datasets. The idea behind these real datasets is to test our methods on 2D curves which have

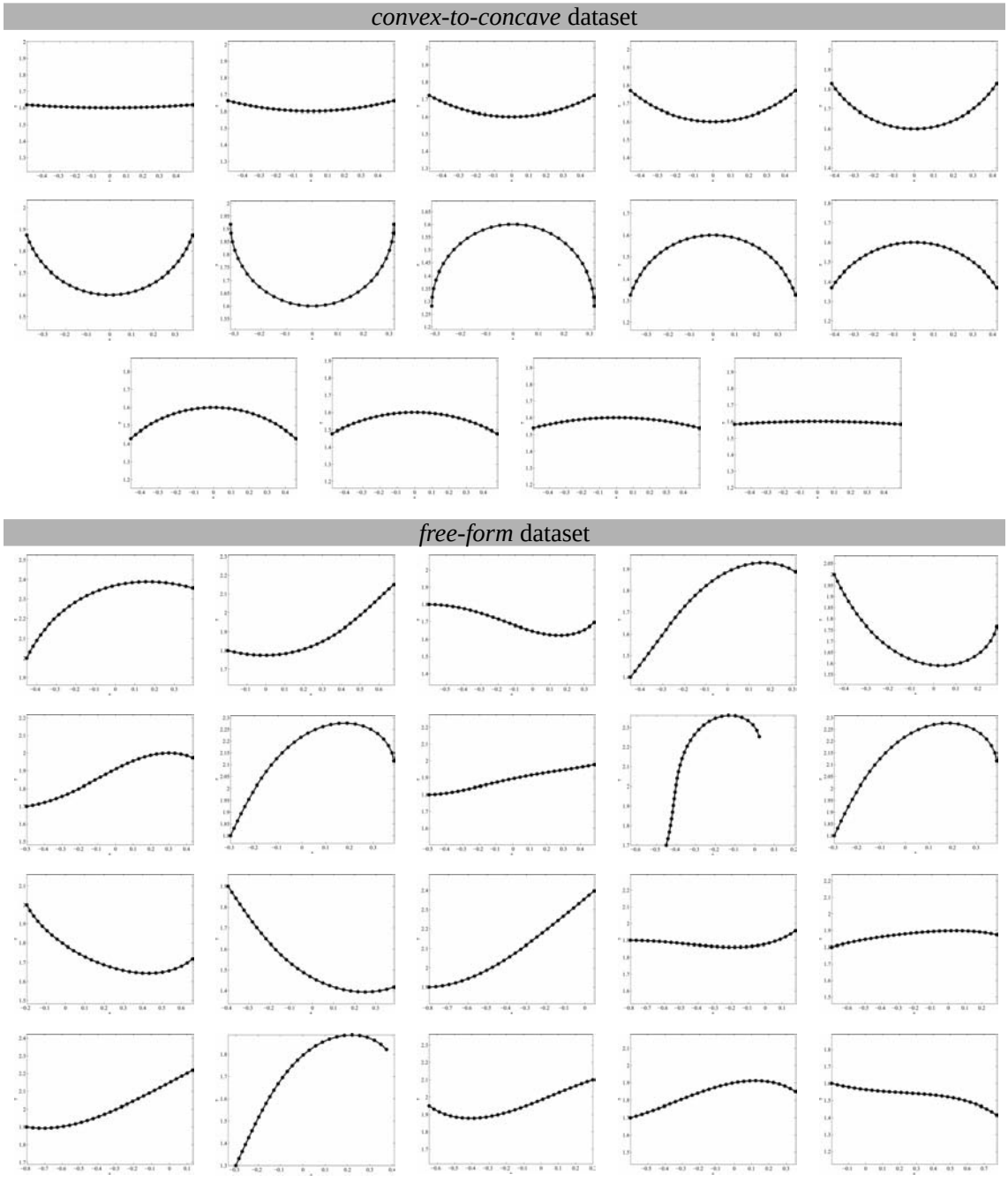


Figure 3.8: Visualization of the ground-truth 2D curves for the two simulated datasets.

a physical meaning. For these two real datasets, the 2D curves which we want to reconstruct are images along a plane of 3D deformations of a paper and a cable. This is illustrated by the figure 3.16, where we see that the reconstructed 2D curves are images of the 3D deformations visible in the 2D images.

The *paper* dataset was built from a 3D reconstruction of a bent paper (figure 3.16) generated by Agisoft Photoscan [Agisoft, 2014]. From this reconstruction, 1D input images could be drawn by sampling lines on the paper. We used the central line illustrated in yellow. To

generate the 1D input image, the camera's y-axis was aligned to the central line. The middle row of the image was thus used as the 1D input image. The *cable* dataset was built using two cameras looking at a cable over a table. One camera took images by aligning its y-axis with the table. The second camera had an over-head view. Its relative pose to the first camera was computed through stereo calibration, and was used to compute the cable's ground-truth shape. We used 5 deformations for the *paper* dataset and 10 for the *cable* dataset. For both datasets, the images had a width of 4800 px. We computed manually 30 correspondences for the *paper* dataset and 40 for the *cable* dataset.

3.5.1.3 Evaluation Metrics

We measure accuracy through four metrics: 2D mean point error, normal error, super critical point precision and super critical point accuracy. We emphasize that, as Curve SfT cannot be solved uniquely, we only evaluate the best candidate solution (with lowest 2D mean point error). We measure the super critical point precision and accuracy only for the simulated datasets. This is because the computation of the ground-truth super critical points is more reliable on simulated datasets, where the first derivative of the ground-truth function θ is less noisy.

2D mean point error (MPE). We construct an evaluation grid by uniformly sampling the template in $G = 30$ points and denote it by $\mathcal{G} = \{u_j\}$. We compute the 2D mean point error (in %) between the reconstructed curve $\hat{\varphi}$ and the ground-truth shape φ^* on $\mathcal{G}$ as:

$$MPE(\hat{\varphi}, \varphi^*, \mathcal{G}) = \frac{1}{G} \sum_{j=1}^{G} \frac{\|\hat{\varphi}(u_j) - \varphi^*(u_j)\|}{\|\varphi^*(u_j)\|}. \tag{3.70}$$

2D normal error (NE). We denote the 2D normal of $\hat{\varphi}$ at a template point u by $\hat{n}(u)$ and the 2D normal of φ^* by $n^*(u)$. In practice, we fit a spline to compute the normals and select the normals with negative y-component. We compute then the 2D normal error (in degrees) between the reconstructed curve $\hat{\varphi}$ and the ground-truth shape φ^* at $\mathcal{G}$ by:

$$NE(\hat{\varphi}, \varphi^*, \mathcal{G}) = \frac{1}{G} \sum_{j=1}^{G} \cos^{-1}\left(\hat{n}^\top(u_j)\, n^*(u_j)\right) \tag{3.71}$$

We now give details on the normal computation. We first fit a spline over the correspondence points between the template and the input image. For this, we use the Matlab function `spaps` with smoothing parameter of 1e−5 for the *convex-to-concave*, *free-form* and *paper* datasets, and 1e−7 for the *cable* dataset. Second, we use the curve fitting toolbox of Matlab to differentiate φ and to obtain their 2D tangents.

Super critical point precision. We compute the super critical point precision as the fraction of the number of true super critical points over the number of detected super critical

points. We define a true super critical point as a super critical point which is close to a ground-truth super critical point up to T % of the template length. We set $T = 5\%$.

Super critical point accuracy (SCPA). We denote by $\{u_j^{s*}\} \in \mathbb{R}^{N_s^*}$ the set of ground-truth super critical points with $u_j^{s*} \in \mathcal{T}$. We denote the closest detected super critical point to each ground-truth super critical point by $\{\hat{u}_j^s\}$. The super critical point accuracy is given by:

$$SCPA(\{\hat{u}_j^s\}, \{u_j^{s*}\}) = \frac{1}{N_s^*} \sum_{j=1}^{N_s^*} \frac{|u_j^s - u_j^{s*}|}{L}, \tag{3.72}$$

where L is the template length.

3.5.1.4 Results on Simulated Datasets

Reconstruction accuracy. Figure 3.9 shows the reconstruction accuracy on the two simulated datasets. Both **2DHMM** and **2DHMM+REF** provide convincing reconstructions, in terms of depth and normals, which we can observe visually in figure 3.10. **2DHMM+REF** is clearly more accurate. This is explained by two reasons. The main one is that the accuracy of **2DHMM** depends on the accuracy of the super critical point locations. If they are badly localized, **2DHMM** cannot be expected to provide a very accurate solution. By contrast, **2DHMM+REF** is free to optimize the curve without being limited to the accuracy of the super critical point locations. Secondly, there is a smoothing term in **2DHMM+REF** that is not present in **2DHMM**, which penalizes non-smooth curve solutions. The benefit is indicated in the empirical results, which show that the normal error is generally strongly reduced with **2DHMM+REF**. We also note that, even if the curve location is not accurate in all input images, the global curvature is well recovered in general. This can be explained by the hard constraint on the tangent at super critical points imposed through E_{sign}. In figure 3.11, we illustrate the inherent limitation of the category *(ii)* method, **2DMDH** and **2DMDH+REF**, which can only generate a single solution. We see that the initial solution from **2DMDH** is wrong, and that the refinement from **2DMDH+REF** is trapped in an incorrect minimum.

We then tested the influence of *(a)* correspondence density and *(b)* correspondence noise on reconstruction accuracy. For *(a)* we added gaussian noise with a standard deviation of 1.0 px to between 10 and 100 random correspondences. To do this, for each input image of the two simulated datasets, we generated the correspondences by uniformly sampling N points along the template with $10 \leq N \leq 100$. For *(b)*, we set the number of correspondences at 30 and run our algorithm with 9 different noise levels with a standard deviation between 0 px and 4.0 px. Figure 3.12 shows the results for both experiments. For both datasets, we note that **2DHMM** is not sensitive to the number of correspondences. This can be explained by the use of a fixed number of nodes in the HMM construction given in §3.4.2.3. The reconstructions with **2DHMM+REF** are slightly better and this is because the refinement uses the real 1D correspondences, as discussed in §3.4.3. The variation of the **2DHMM**

reconstruction accuracy for the *free-form* dataset may be explained by the shape of the 2D curves which are more complex than the ones of the *convex-to-concave* dataset. However, for both datasets, we note that the refined solution **2DHMM+REF** is improved with higher numbers of correspondences, which makes sense as we have more data constraints. Regarding the noise level, we also note the high robustness of **2DHMM** for the *convex-to-concave* dataset, which is due to the fact that its dataset has very simple curves. The *free-form* dataset is more informative: we observe that the increase in noise level degrades the performance of **2DHMM** and **2DHMM+REF**. Globally, we note that in all experiments the refinement improves the reconstruction significantly.

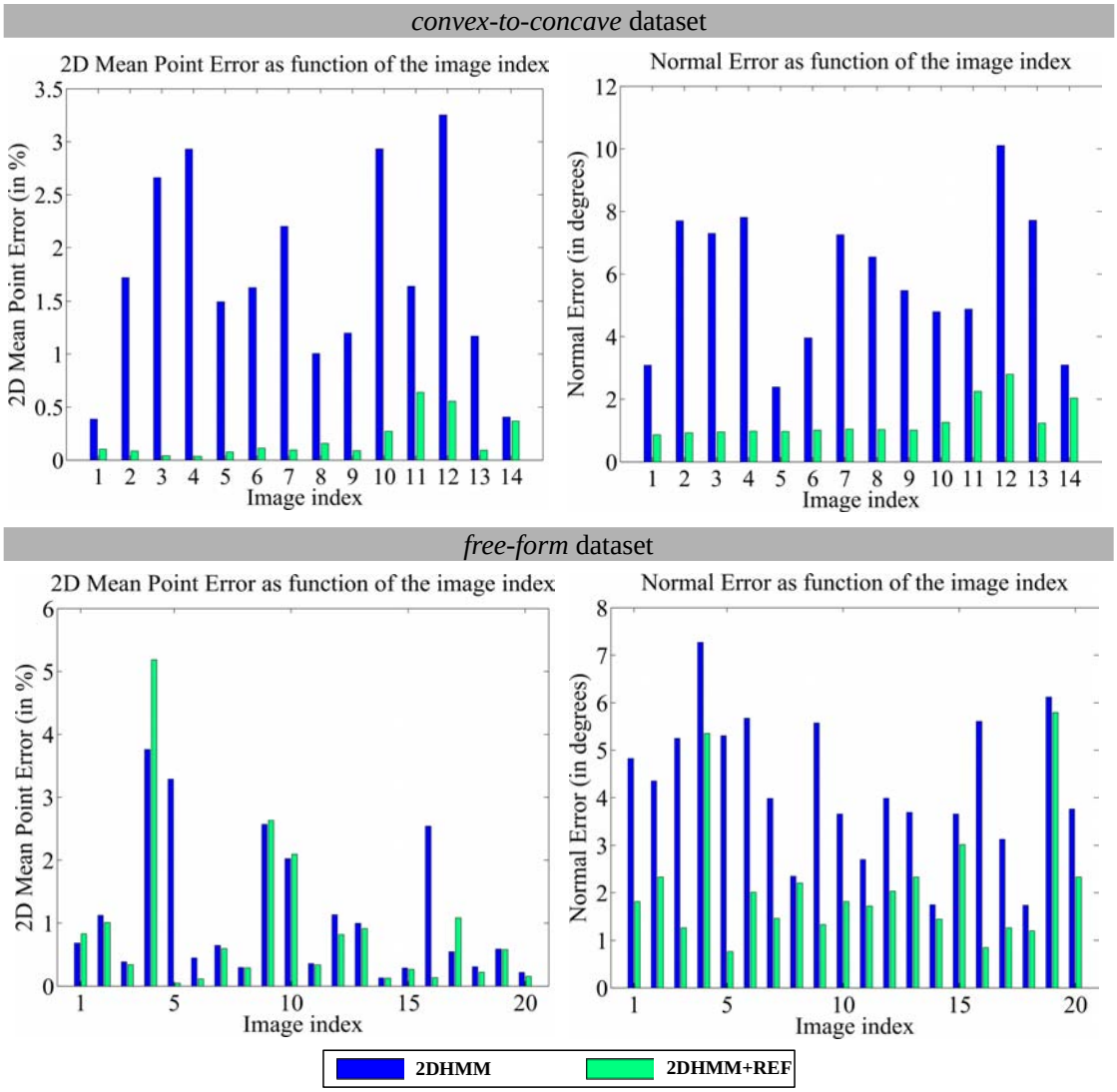


Figure 3.9: Reconstruction accuracy of **2DHMM** and **2DHMM+REF**, for the two simulated datasets.

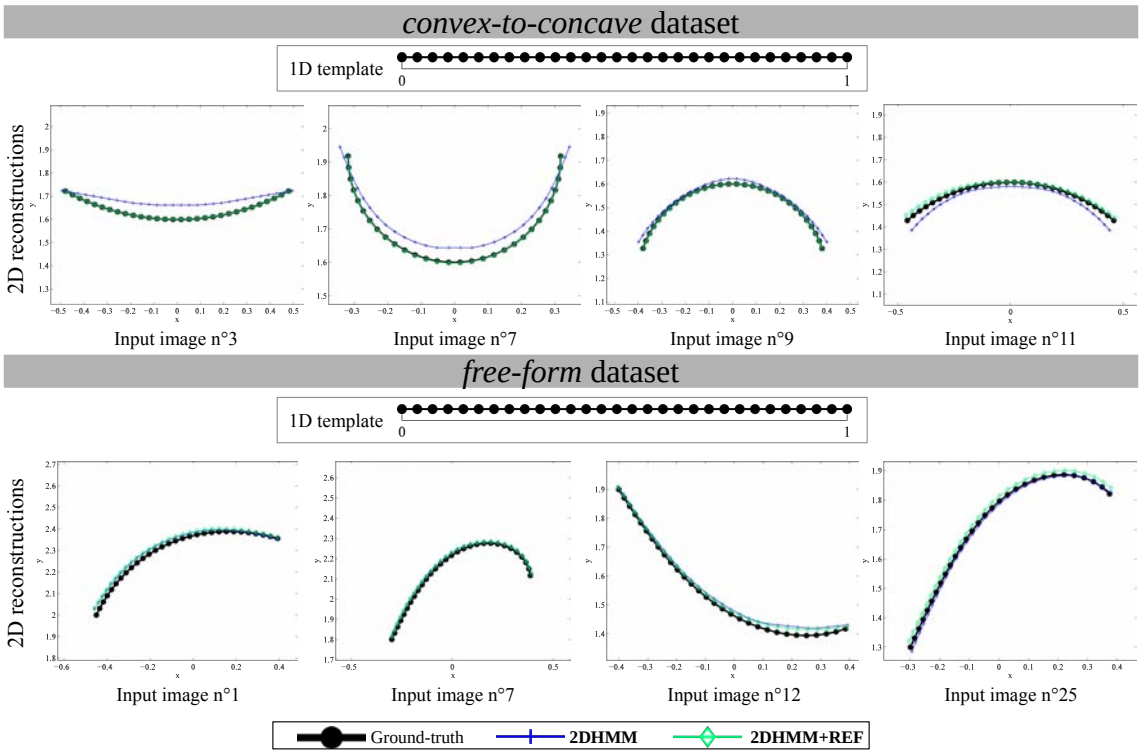


Figure 3.10: Visual results of **2DHMM** and **2DHMM+REF**, for the two simulated datasets. We show the 2D reconstructed curves and their ground-truth solutions. As the 2D reconstructed curves are very close to the ground-truth solution, see the digital version of the document for better visualization.

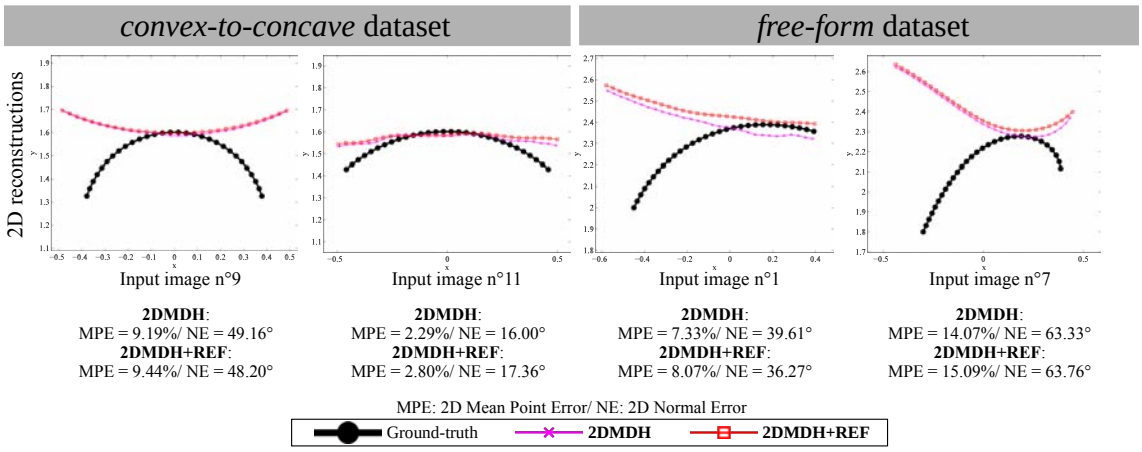


Figure 3.11: Visual results and reconstruction accuracy of **2DMDH** and **2DMDH+REF** on a subset of input images used in figure 3.10. We verify that the category *(ii)* method does not always give the correct solution because it is a single-solution method.

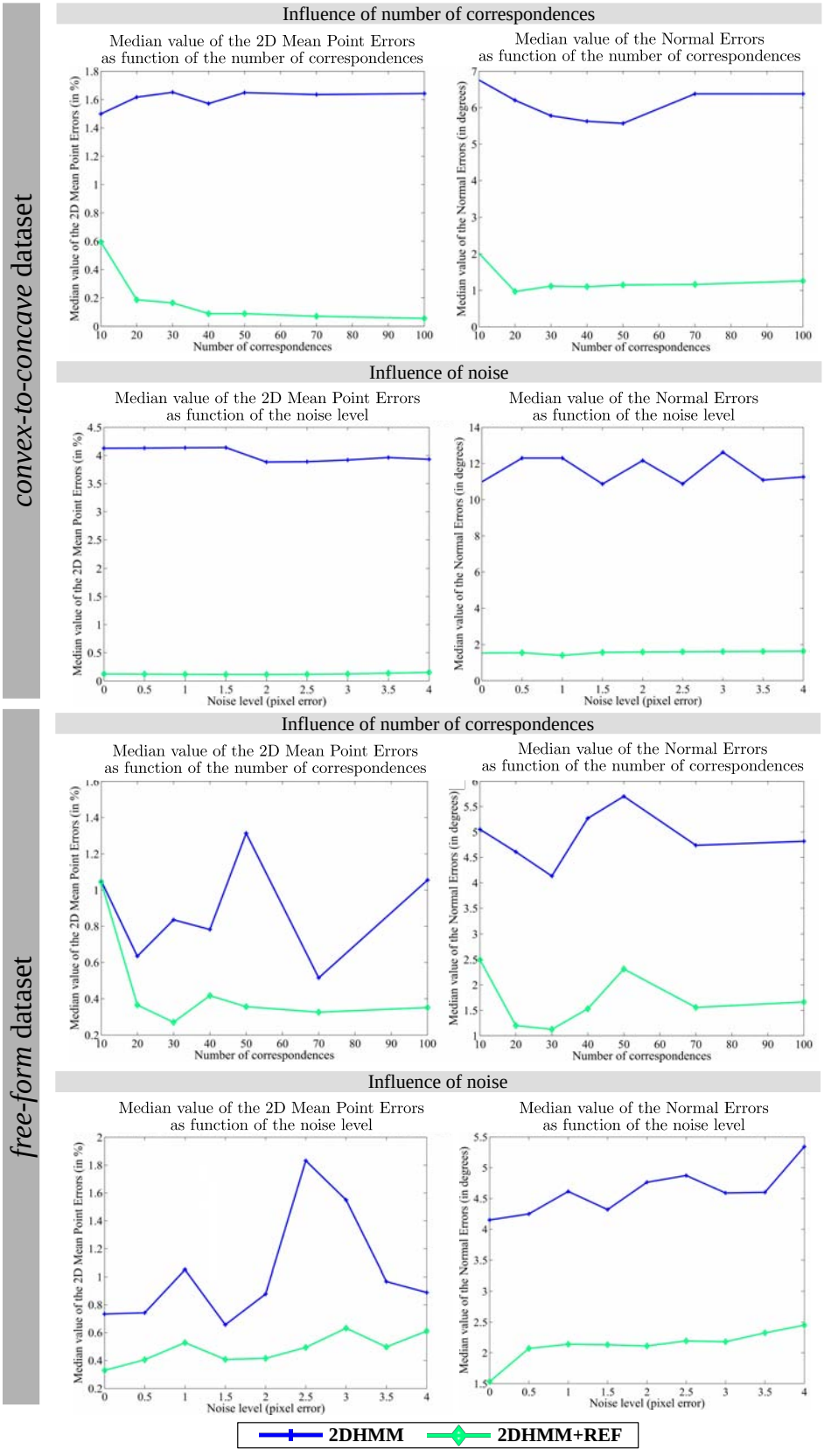


Figure 3.12: Experimental analysis of **2DHMM** and **2DHMM+REF**, using the two simulated datasets. We ran both methods with varying numbers of correspondences (with fixed noise level) and for several noise levels (with fixed number of correspondences). We recall that, to compute these errors, we only use the best solution among the multiple ones given by **2DHMM**.

Super critical point detection. Figure 3.13 shows the super critical point precision for the simulated datasets. Our detection method has perfect precision for the *convex-to-concave* dataset and slightly over-detects the super critical points for the *free-form* dataset. We explain the very good results on the *convex-to-concave* dataset by the fact that the curves of this dataset are relatively smooth, so the interpolation of the warp with the correspondences is very accurate everywhere.

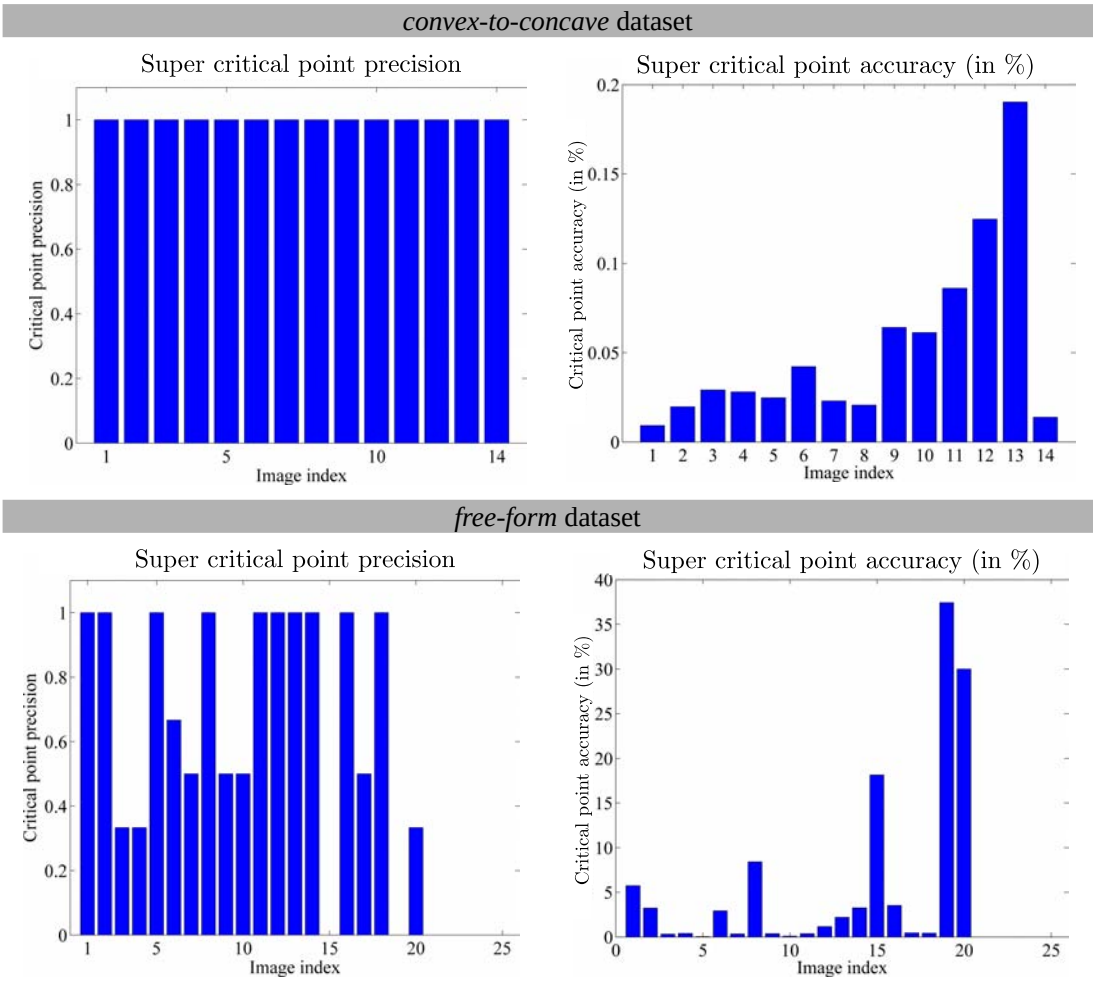


Figure 3.13: Super critical point precision and accuracy. We use the two simulated datasets.

Influence of super critical point uncertainty on reconstruction accuracy. Our final test consisted in evaluating how sensitive the reconstructions are to incorrectly-located super critical points. Figure 3.14 shows the reconstruction accuracy for different perturbation levels of the ground-truth super critical point locations. A perturbation level of $X\%$ means that all super critical points were translated by $X\%$ of the template length along the template. We use the median value of the 2D mean point error and of the 2D normal error to measure the reconstruction accuracy, as we show in the second row of figure 3.14. We note first that the global minimum is very close to the perturbation level of 0%. We can explain this

Figure 3.14: Reconstruction accuracy as function of the perturbation on the super critical point location along the template. We use the two simulated datasets.

discrepancy by numerical issues since the ground-truth super critical points are not directly observed, but detected by a process which involves fitting and differentiation of ground-truth data. We then observe that the local minima are strongly symmetric for the *convex-to-concave* dataset. This is because the curves are symmetric around the super critical point, which is located near the midpoint of the curve, *i.e.* the point around which we change the shape of the curve. Super critical point uncertainty does not have a significant impact on the reconstruction accuracy: they degrade the reconstructions by less then 3% 2D mean point error and by less than 3.5 degrees normal error for the *convex-to-concave* dataset, and by less than 1% 2D mean point error and by less than 1.5 degrees normal error for the *free-form* dataset. **2DHMM** and **2DHMM+REF** are more robust on the *free-form* dataset, even if it is a more challenging dataset than the *convex-to-concave* dataset. There is a relationship between the curve's shape and the sensitivity to super critical point localization and this is demonstrated by the difference in performance between the two datasets. It is out of scope

to analyze this relationship, but it may be possible with perturbation theory. Our method **2DHMM** and **2DHMM+REF** are able to generate reasonable candidate solutions despite a relatively large error in super critical point locations (until ±10% of the template length).

3.5.1.5 Results on Real Datasets

In figure 3.16, we observe that our methods produce convincing reconstructions. This is coherent with the reconstruction accuracy presented in figure 3.15. We see that the refinement method **2DHMM+REF** produces the most accurate results: the refinement globally improves the normals orientation. There is no significant difference in 2D mean point error for the *paper* dataset between **2DHMM** and **2DHMM+REF**. In figure 3.17, we show the reconstruction results with the category *(ii)* method with and without refinement, **2DMDH** and **2DMDH+REF**. Note again that it is not capable to provide the correct solution in all cases.

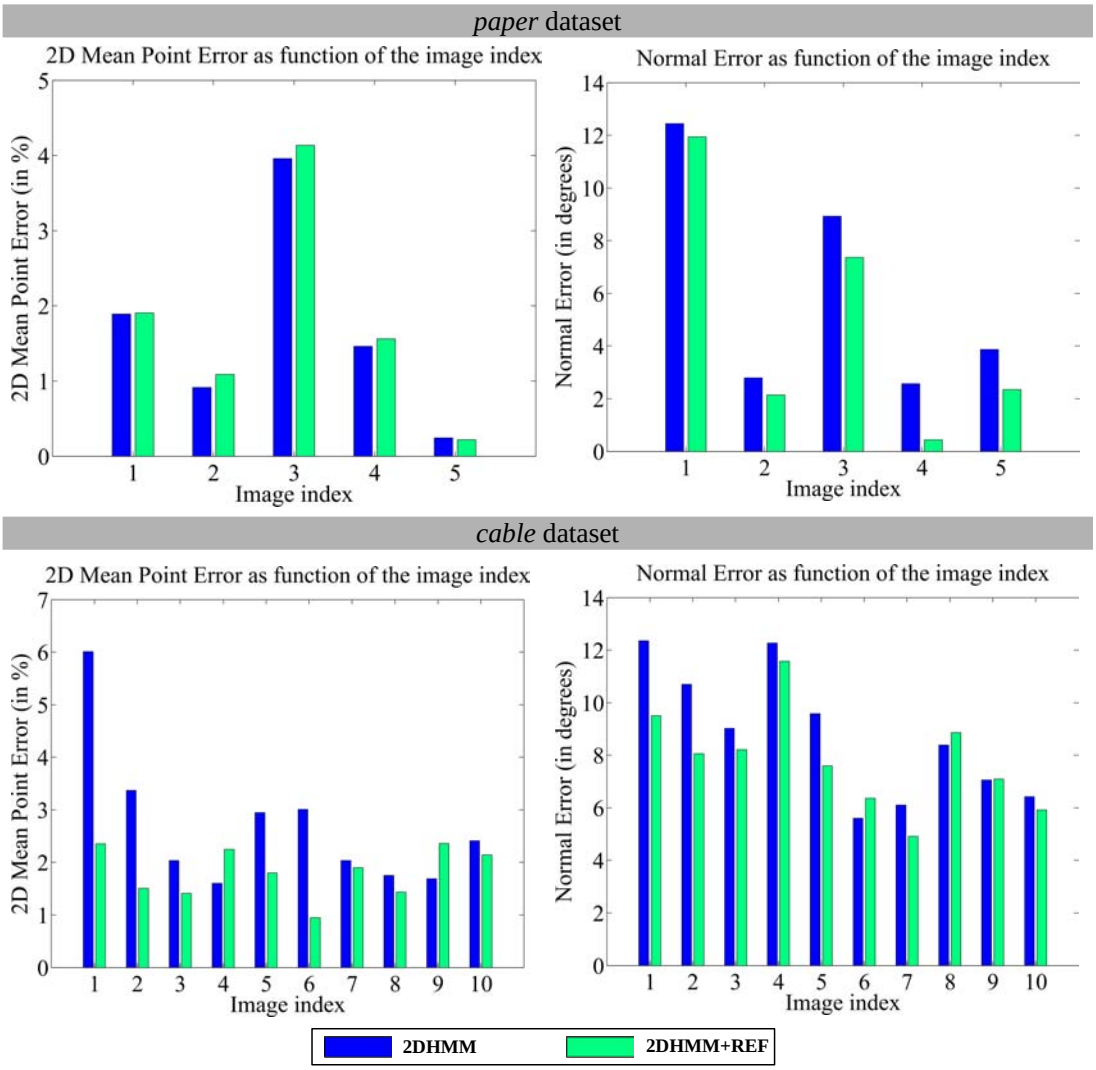


Figure 3.15: Reconstruction accuracy of **2DHMM** and **2DHMM+REF**, for the two real datasets.

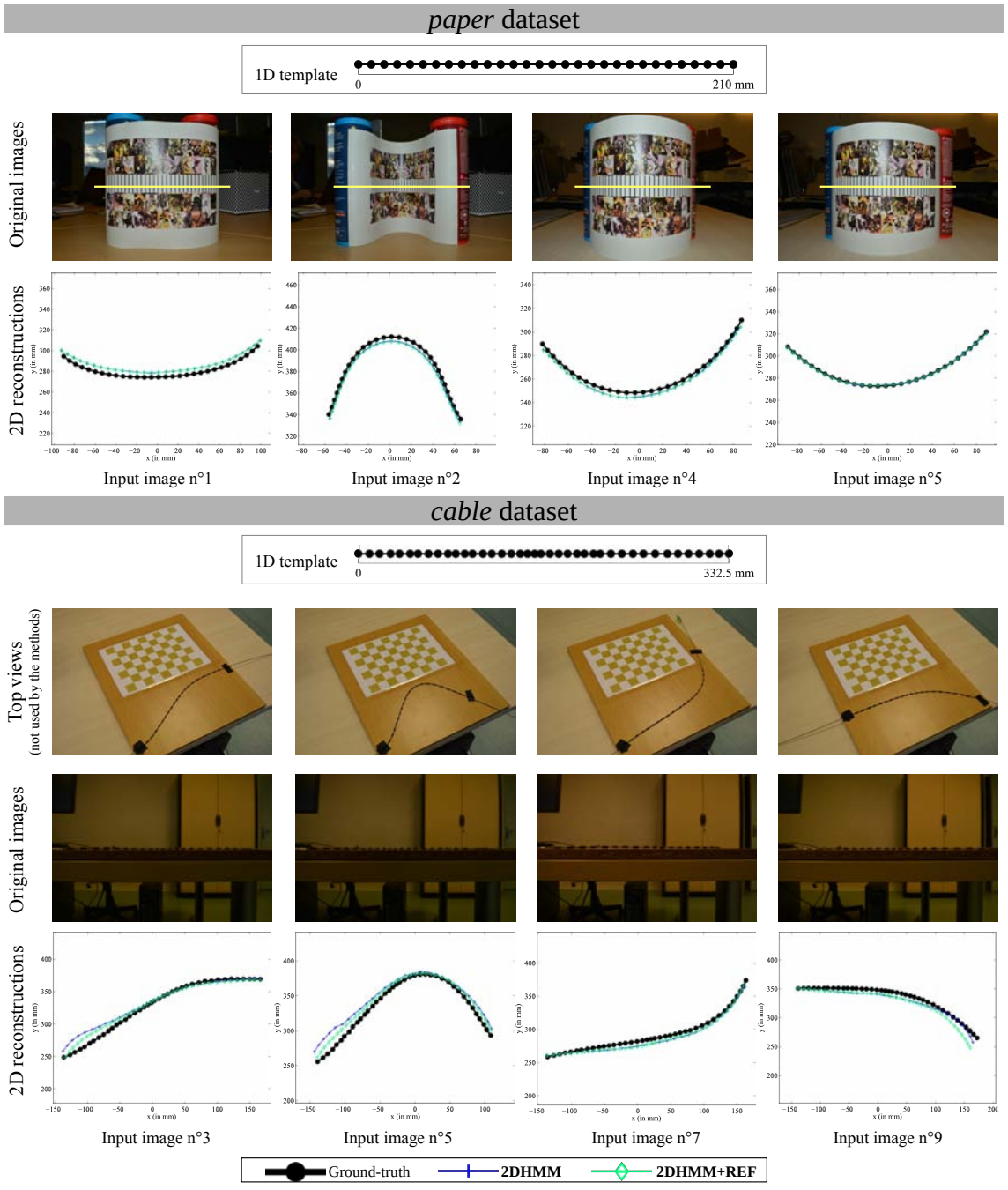


Figure 3.16: Visual results of **2DHMM** and **2DHMM+REF**, for the two real datasets. We show 2D reconstructed curves and their ground-truth solutions. For each dataset, the original images correspond to the images from which we get the 1D input images. For the *cable* dataset, we show a top view of the deformed cable for each input image. As the 2D reconstructed curves are very close to the ground-truth solution, we refer readers to the digital version of the document for better visualization.

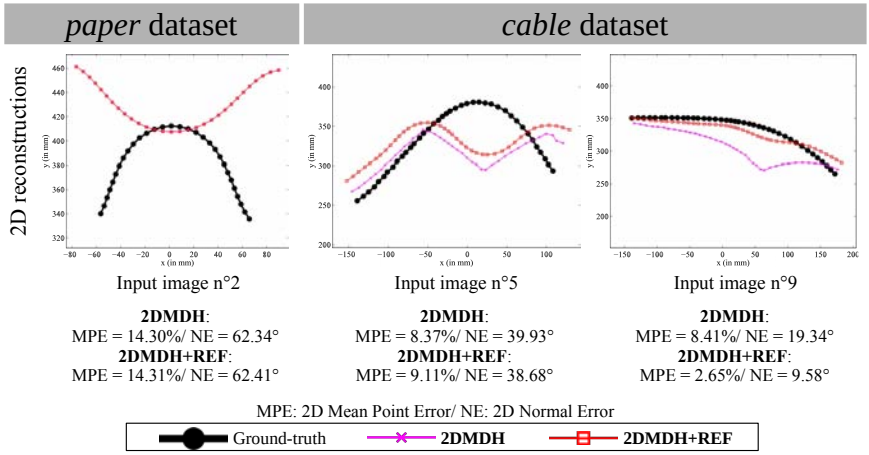


Figure 3.17: Visual results and reconstruction accuracy of the category *(ii)* methods, **2DMDH** and **2DMDH+REF**, on a subset of input images used in figure 3.16.

3.5.2 *Curve SfT-1* Experiments

3.5.2.1 Methods

Similarly to §3.5.1, we only evaluate the HMM method, with and without refinement. Because the outputs of our methods are 3D curves, we name the methods respectively as **3DHMM** and **3DHMM+REF**. We show results of the category *(ii)* method, without and with refinement, denoted respectively as **3DMDH** and **3DMDH+REF**. We refer to §3.5.1.1 for the implementations details. Table A.2 in appendix A gives the hyperparameters for each method for the sake of reproducibility.

3.5.2.2 Datasets

Simulated dataset. We evaluated with a simulated dataset: the *3D cord* dataset. We built it using the software Blender [Blender, 2017]. We simulated a set of 40 spheres linked together by a Bézier curve. Moving the Bézier curve allows us to move the set of spheres to behave as points on a near-isometric curve. We simulated 15 curve deformations, rendered them on images of 960×540 px. The curves are placed on average at 200 mm of the camera center and the focal length is set to 35 mm. We created the 1D template by using the distance between the sphere centers along the Bézier curve. In each input image, we use the projection of the sphere centers as data points and match them with the 1D template. We added to the 2D image correspondences a gaussian noise of $\sigma = 2.0$ px. We show in figure 3.18 the simulated 2D curves for both datasets. Some examples of input images are shown in figure 3.18.

Real datasets. We tested our methods on two real scenes: the *road* and the *necklace* datasets. The first dataset is composed of one input image of a road with a varying curvature, as figure 3.22 shows. Its 1D template is defined by the distance between each transition of

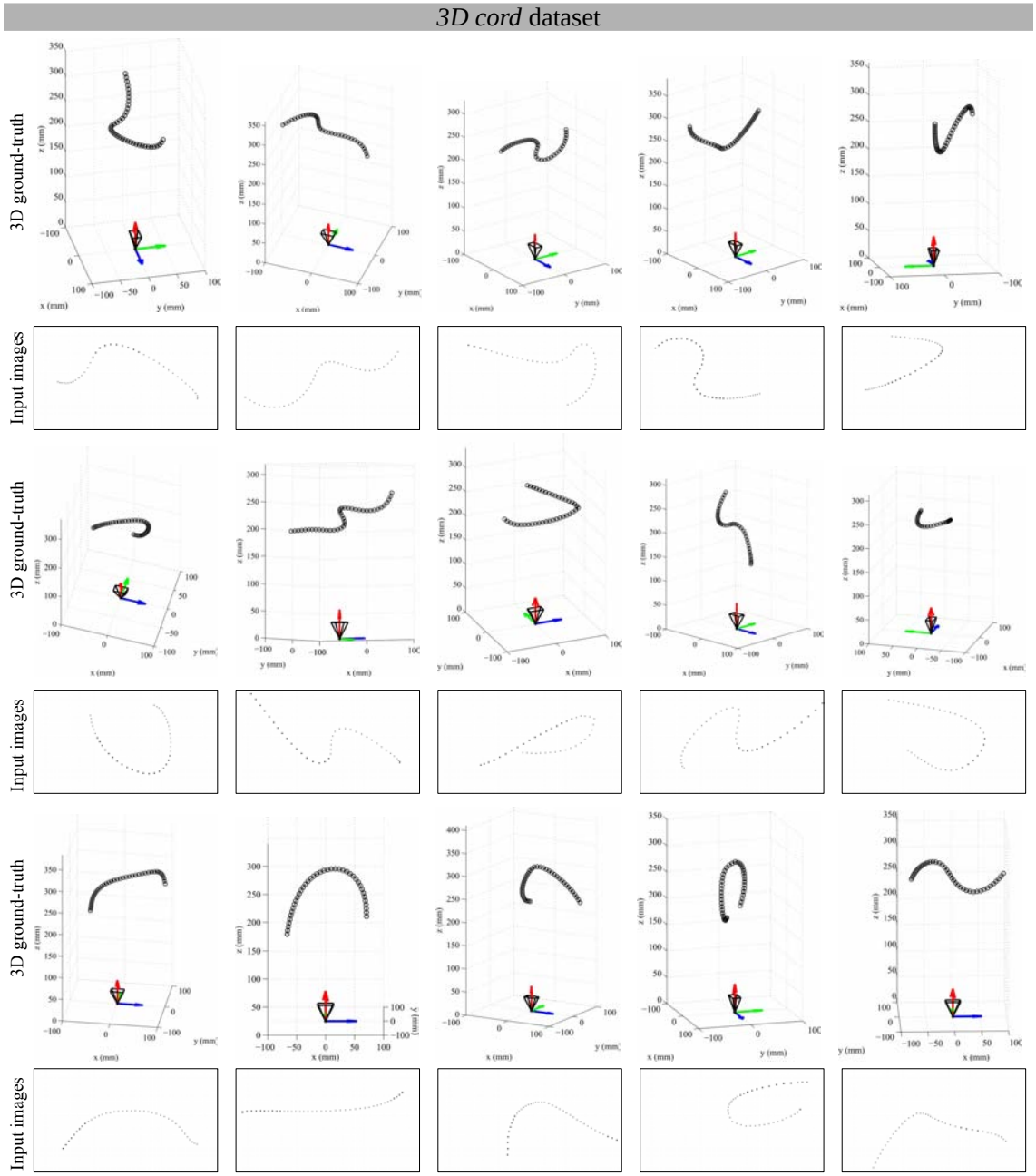


Figure 3.18: Visualization of the ground-truth 3D curves for the simulated dataset, *3D cord*. We give the associated 2D input images.

road signs, which is standard (0.5 m). We computed manually 63 correspondences between the 1D template and the 2D input image by selecting the left corners of the road signs shown in the 2D input image, shown in figure 3.22. The 2D input image is of size 4608×3072 px. The second dataset is composed of one input image of a necklace laid over a pillow. Its 1D template is defined by the distance between the mass centers of the pearls, as shows figure 3.23. We computed manually 28 correspondences between the 1D template and the 2D input image by selecting then the mass centers of the pearls. The 2D input image is of size

3600 × 2800 px. The software Agisoft Lens [Agisoft, 2013] is used to calibrate the cameras and Agisoft Photoscan [Agisoft, 2014] is used to reconstruct the 3D scene.

3.5.2.3 Evaluation Metrics

3D mean point error (MPE). This error is the trivial extension of the 2D mean point relative error.

3D tangent error (TE). We use 3D tangent error to evaluate the shape accuracy on the 3D curves because of the ambiguity on the normals of 3D curves (the normal of $\varphi \in C^\infty(\mathcal{T}, \mathbb{R}^3)$ is defined up to a rotation about the curve's tangent vector). We denote the 3D tangent of $\hat{\varphi}$ at a template point u by $\hat{t}(u)$ and the 3D tangent of φ^* by $t^*(u)$. Similarly to §3.5.1.3, we fit a spline to compute the tangents with a smoothing parameter of 1e1 for all datasets. We then compute the 3D tangent error (in degrees) between the reconstructed curve $\hat{\varphi}$ and the ground-truth shape φ^* at $\mathcal{G}$:

$$TE(\hat{\varphi}, \varphi^*, \mathcal{G}) = \frac{1}{G} \sum_{j=1}^{G} \cos^{-1}\left(\hat{t}^\top(u_j)\, t^*(u_j)\right). \tag{3.73}$$

3.5.2.4 Results on Simulated Datasets

In figure 3.19, we show the reconstruction errors for our methods, **3DHMM** and **3DHMM+REF**. They perform globally well, but we can note that **3DHMM+REF** performs less well than **2DHMM+REF**, which may be explained by the complexity of the angle-based parameterization of 3D curves compared than the one of 2D curves. In figure 3.20, we display some 3D reconstructions computed by our methods. We can see the results from different viewpoints and observe that **3DHMM** and **3DHMM+REF** provide shapes which are quite close to the ground-truth shapes. As for *Curve SfT-2*, we note in figure 3.21 the limitation of the category *(ii)* method regarding the non-uniqueness of the problem. We show in figure 3.21 some failure cases of **3DMDH** and **3DMDH+REF**.

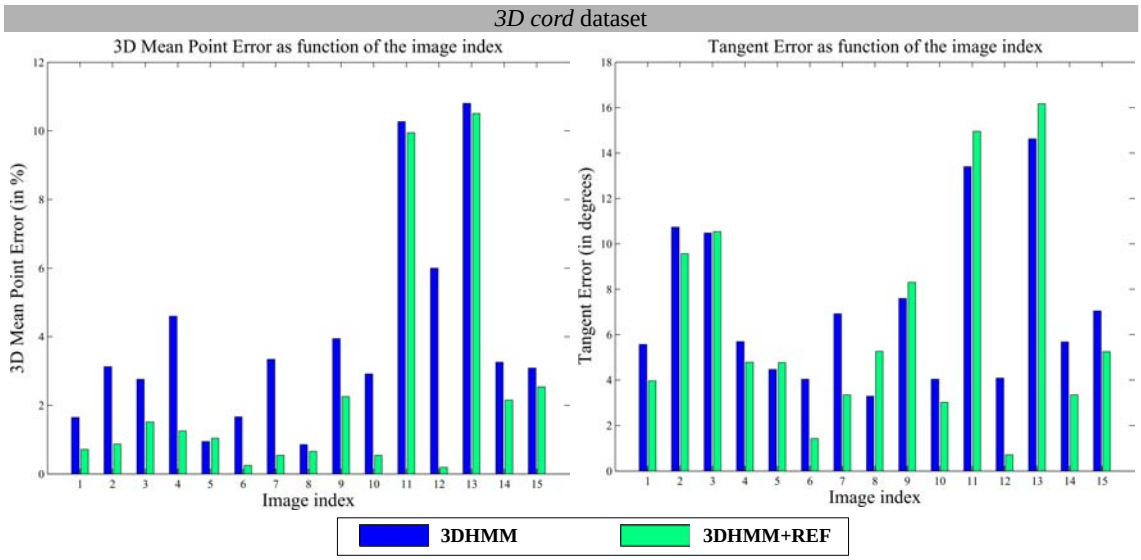


Figure 3.19: Reconstruction accuracy of **3DHMM** and **3DHMM+REF**, for the simulated dataset, *3D cord*.

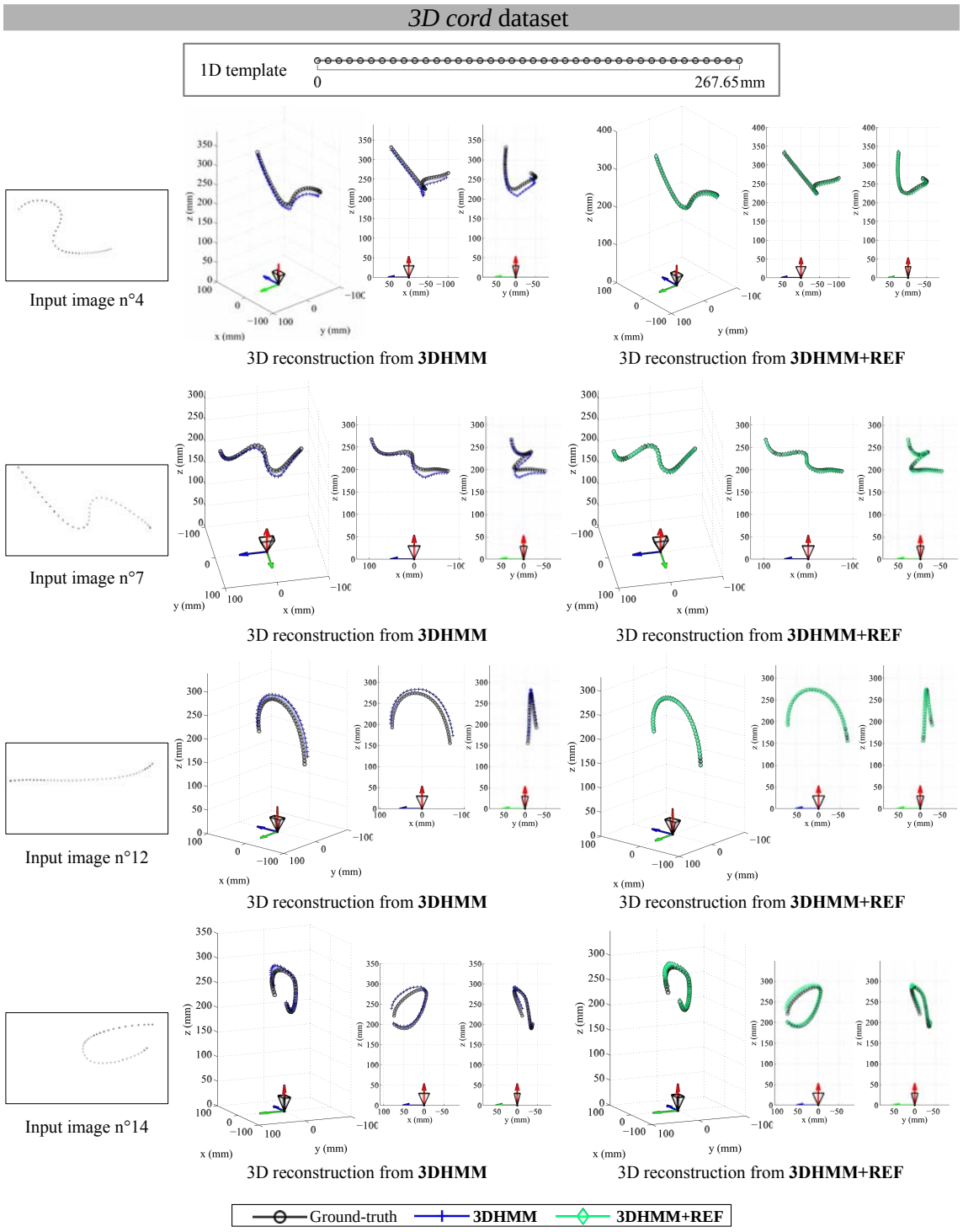


Figure 3.20: Visual results of **3DHMM** and **3DHMM+REF**, for the simulated dataset, *3D cord*. We show the 3D reconstructed curves and the ground-truth solutions. Each row corresponds to one input image with the reconstructions given **3DHMM** and **3DHMM+REF**. For each reconstruction, we give three different viewpoints. As the 3D reconstructed curves are very close to the ground-truth solution, we refer the readers to the digital version of the document for better visualization.

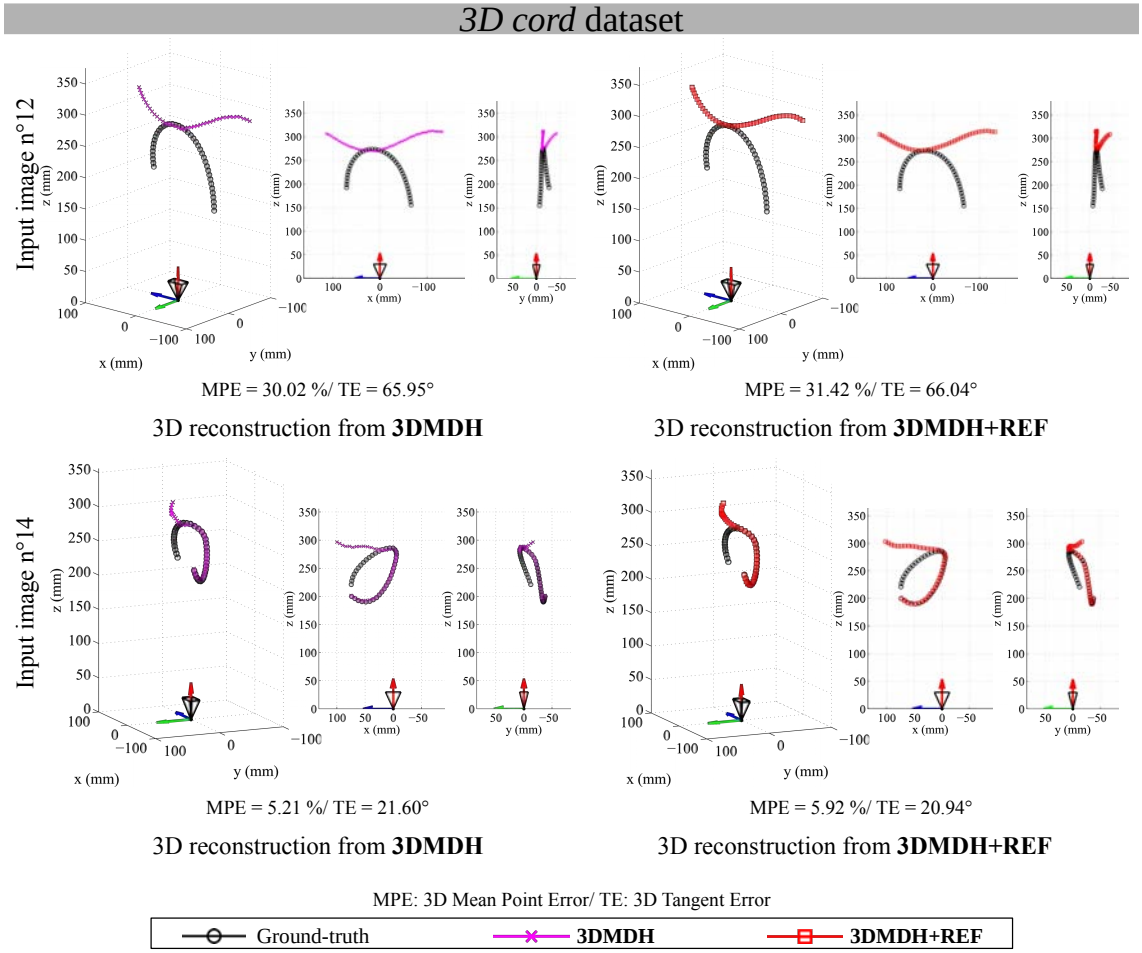


Figure 3.21: Visual results and reconstruction accuracy of **3DMDH** and **3DMDH+REF** on the simulated dataset, *3D cord*. Each row corresponds to one input image with the reconstructions given by **3DMDH** and **3DMDH+REF**. The input images are shown in figure 3.20. For each reconstruction, we give three different viewpoints.

3.5.2.5 Results on Real Datasets

***Road* dataset.** We give the reconstructions of the *road* dataset in figure 3.22. The curvature of both results, **3DHMM** and **3DHMM+REF**, is consistent with the curvature of the road visible in the two images. We can note that the curve seems to bend in accordance with the two viewpoints given. Precisely, we can also observe that the furthest segments of curve seem to lie almost on a plane.

***Necklace* dataset.** We give the reconstructions of the *necklace* dataset in figure 3.23, for the categories *(ii)* and *(iv)* methods with and without refinement. We see that **3DMDH** does not give the correct solution: the curvature at the left end of the curve is wrong. The refinement **3DMDH+REF** cannot change the curvature, which may indicate it is a local minimum. However, we observe that **3DHMM** provides the correct solution, *i.e.* the correct curvature along the curve, which is supported by very good reconstruction accuracy.

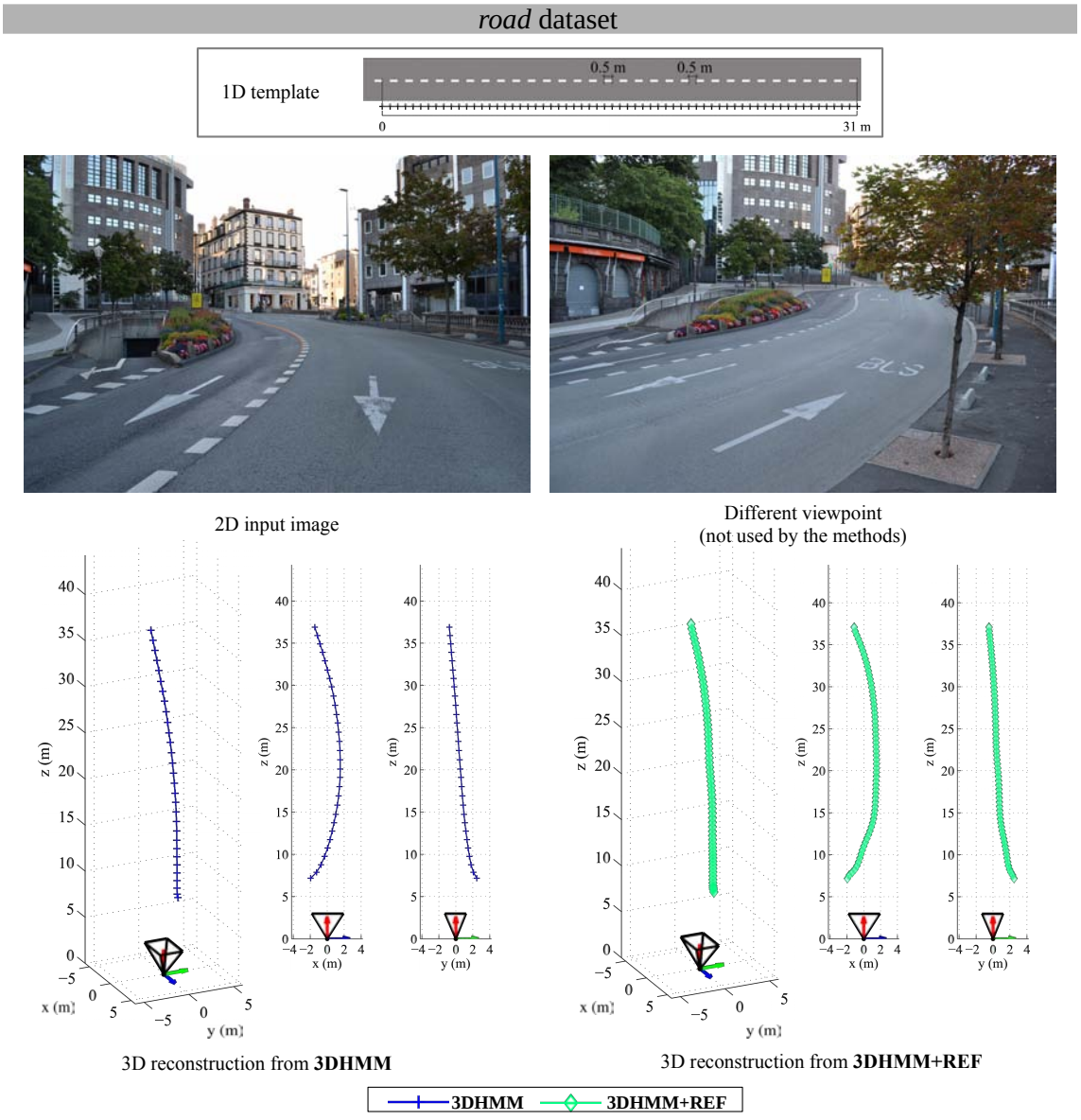


Figure 3.22: Visual results of **3DHMM** and **3DHMM+REF**, for the real dataset, *road*. In the first row, we give the 1D template and an illustration of the line of road signs. In the second row, we show the 2D input image and a different viewpoint of the same scene. The 3D curve to reconstruct corresponds to left corners of the road signs shown in the input image with orange crosses. In the third row, we give, for each reconstruction method, three different viewpoints.

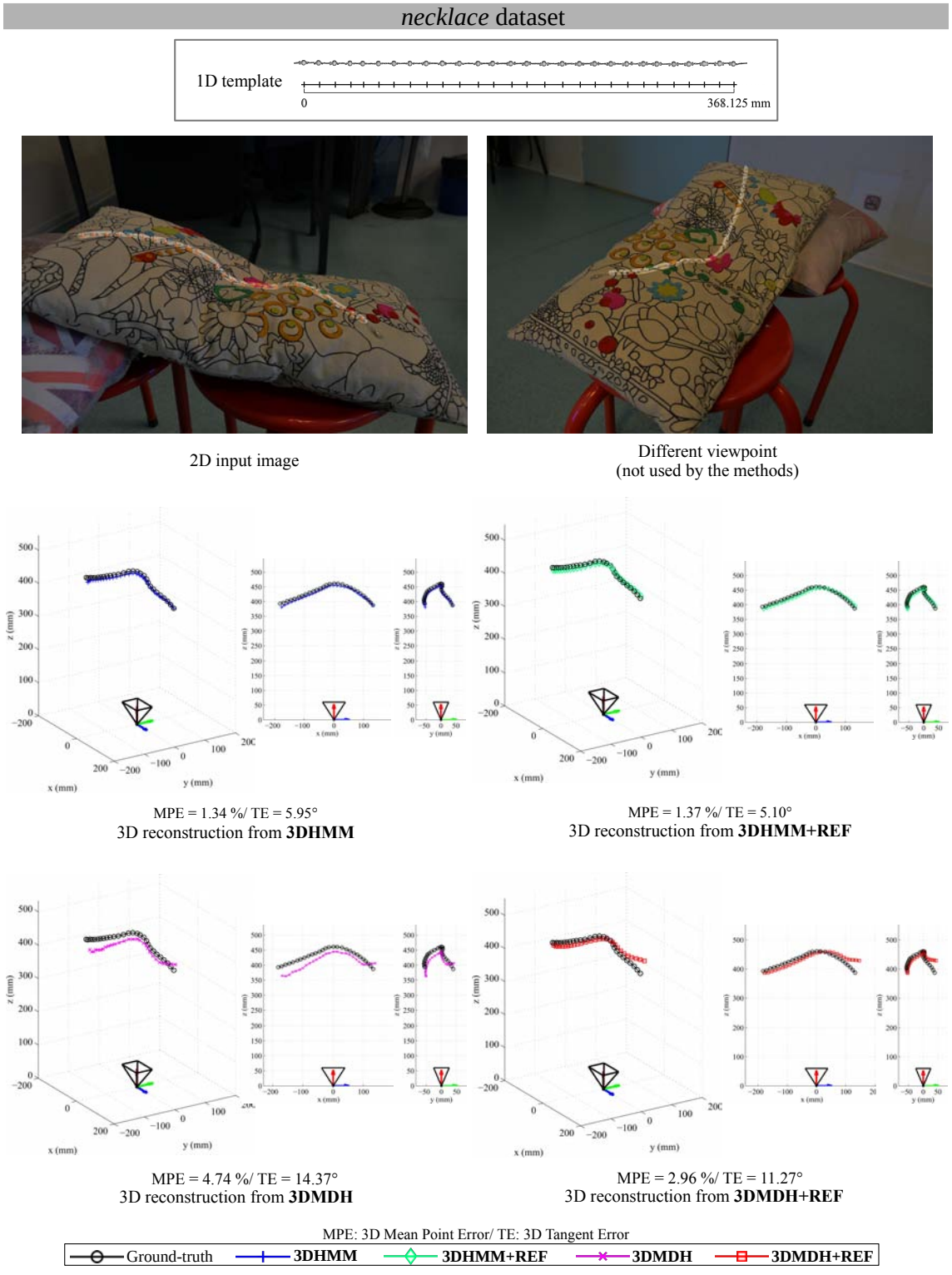


Figure 3.23: Visual results and reconstruction accuracy of all methods, **3DHMM**, **3DHMM+REF**, **3DMDH** and **3DMDH+REF**, for the real dataset, *necklace*. In the first row, we show its 1D template and a picture of the necklace used to construct the dataset. In the second row, we show the 2D input image and a different viewpoint of the same scene. The 3D curve to reconstruct corresponds to mass centers of the pearls visible in the 2D input image with orange crosses. In the third row, we give the visual and reconstruction accuracy of **3DHMM** and **3DHMM+REF**. In the four row, we give the visual and reconstruction accuracy of **3DMDH** and **3DMDH+REF**. For each reconstruction, we give three different viewpoints.

3.5.3 Limitations and Failure Modes

We discuss here the main limitations and the failure modes of our solutions to Curve SfT. One limitation is that the parameters of our methods are manually set and they may vary for some datasets. However, as the number of tuned parameters is relatively small, this is not a critical issue. Another limitation is that our methods work only for isometric deformations. This assumption is essential since it allows us to prove our theoretical results and construct our computational solutions (category *(i)* to category *(iv)*). An important limitation is our assumption on the correspondences between the template and the input image, which we give in §3.2.2. The correspondences should be sufficiently dense so that the warp can be estimated through a smooth interpolation. When this assumption is not met in practice, we face the main failure modes of our computational solutions. For methods of all categories, the failure mode is that there is not enough motion information to infer the whole curvature. For the HMM solution, the failure mode is that the detection of critical points and thus the reconstruction accuracy can be significantly impacted.

3.6 Conclusion

We have presented a theoretical study of isometric Curve SfT and its implementation to recover respectively 2D and 3D curves using a 1D template. We have revealed the complexity of both problems, *Curve SfT-2* and *Curve SfT-1*, thanks to a differential analysis. We have arrived at a deep understanding of Curve SfT using the very informative super critical points which can be detected directly from the input data. The main theoretical outcome is that, when Curve SfT has N_s super critical points, there exist 2^{N_s+1} candidate solutions. Among several proposed computational solutions, we have proposed a distinctive method based on a discrete HMM which generates all ambiguous solutions using our theory and the super critical points. This method (category *(iv)*), without and with the refinement (category *(iii)*), presents satisfying reconstruction accuracy on simulated and real datasets, while an usual approach of convex optimization appears to be limited (category *(ii)*) showing inaccurate reconstruction results. These results encourage us to see if such super critical points can be found in other 3D reconstruction problems and how graph-based approaches, such as HMM, can be employed to solve other 3D reconstruction problems. Future works are discussed in chapter 7.

Chapter 4

Shape-from-Template for Creasable Surfaces

Summary

We address the first limitation of SfT: handling creasable surfaces. Most of current SfT methods usually fail to reconstruct non-smooth deformations such as surface creases. This is due to the sparsity of the motion constraint and strong ℓ_2 regularization. Our main idea is to implicitly model creases with a dense mesh-based surface representation and an associated robust bending energy constraint, which deactivates curvature smoothing automatically as and when needed. This robust regularizer is based on an M-estimator and, crucially, the crease locations are not required a priori since they emerge as the lowest-energy points at convergence. Therefore, as the crease modeling is driven by 2D data, the registration of the surface has to be accurate. To do so, we complement the motion constraint with a robust boundary contour constraint. We refer to this special instance of SfT as SfT-1 and we propose a cascaded optimization framework for solving it. We evaluate our solution with quantitative analysis and show that it is possible to accurately register and reconstruct creasable surfaces without knowing a priori the crease locations, from a single image. This chapter is based on our peer reviewed paper [Gallardo et al., 2016a].

We first review some works of 3D reconstruction of creasable surfaces. We then give a clear definition of the SfT instance we propose to solve in this chapter. From this definition, we present our choice for the different models required and our formulation to solve the problem instance.

4.1 Reconstruction of Creasable Surfaces

We divide this section into two parts. In the first part, we discuss a previous attempt to handle creases in [Salzmann and Fua, 2009]. In the second part, we discuss how creases and surface discontinuities are modeled in other 3D reconstruction problems.

4.1.1 Modeling Creases in SfT

As mentioned in chapter 2, practically all existing SfT methods use an ℓ_2 norm to regularize surface bending, or use deformation models with some dimensionality reduction that eliminates high-frequencies deformation components. However, the former cannot model creases because the ℓ_2 norm incorrectly penalizes non-smooth solutions and the latter only models smooth deformations. The problem of creases or 'sharp folds' has been looked at before in [Salzmann and Fua, 2009], via a convex formulation which maximizes the depth of each vertex and relaxes the isometry constraint. The isometry constraint preserves the geodesic distance between two vertices. Following [Perriollat et al., 2008], [Salzmann and Fua, 2009] relaxed this constraint into the inextensibility constraint: the geodesic distance between two surface points is replaced by the Euclidean distance between the two surface points and then this Euclidean distance is upper-bounded by the geodesic distance between the two points in the template. As the Euclidean distance may decrease when creases appear, vertices may come closer to each other. This occurs without making the surface extend, thanks to the distance upper-bound, or shrink, thanks to the depth maximization. This allows creases to appear, however they are not explicitly modeled since they are a by-product of the inextensibility constraint. We observed experimentally that [Salzmann and Fua, 2009] is not capable to register and reconstruct accurately creases.

4.1.2 Modeling Creases in Other Problem Domains

The problem of fitting non-smooth surfaces, including creases, has been extensively addressed in the curve [Kaess and Dellaert, 2003] and surface [Fleishman et al., 2005; Gal et al., 2007; Hoppe et al., 1994] fitting literature. These generally address the problem of fitting curves or surfaces to 2D or 3D point sets respectively. Two approaches exist: one can densify a mesh [Fleishman et al., 2005; Hoppe et al., 1994; Kaess and Dellaert, 2003] or register non-rigidly a model to the data [Gal et al., 2007; Pauly et al., 2005]. [Hoppe et al., 1994] proposes the idea of tagging control points of a mesh and using subdivision surfaces to model discontinuities like creases and corners. This 3D concept is adapted to the 2D case by [Kaess and Dellaert, 2003]. Another use of adaptive meshes is proposed by [Fleishman et al., 2005]. It

starts by fitting a smooth model to a downsampled set of points that excludes wrongly scanned points. Then, to provide a better fitting, it selects iteratively new data points which have the smallest prediction residuals. The second category reconstructs discontinuous surfaces from 3D point cloud by having the user select a set of global forms [Pauly et al., 2005] or local shape examples [Gal et al., 2007].

The problem of reconstructing discontinuous surfaces from 3D data is strongly data-driven, which is different to SfT. Specifically, in SfT, we do not have 3D data. Instead we only have 2D projection data present in the input image, which is much weaker information. Indeed the whole reason why we require a template in SfT is to form a well-posed problem by using the template's physical deformation constraints. To do this, we must simultaneously register the template and reconstruct deformation, including creases. This has not been achieved previously.

Discontinuity in data has been also studied in other computer vision problems such as optical flow [Black and Anandan, 1993; Zach et al., 2007]. One common strategy is to use M-estimators for handling non-smooth solutions, and this gave us much inspiration. However, it was unclear whether M-estimators offered a good solution to handle SfT, where 3D shape has to be reconstructed from 2D data.

Chapter outline. In §4.2, we present our implicit crease energy model and its associated cost function. In §4.3, we present the full optimization framework. In §4.4, we validate our method with real data using ground-truth generated by a high-accuracy structured-light scanner. In §4.5, we provide our conclusions.

4.2 Problem Modeling

This section first gives the fundamental models used in SfT. We then specialize SfT with a concrete problem instance which handles creases and is applicable in most real-world situations.

4.2.1 Fundamental Models of SfT

In order to solve SfT, two fundamental models are required: the *template* and the *camera projection* model. The *template* is a fundamental element of SfT since it gives strong physical constraint, as described in §2.2.2. The *camera projection* model determines how to reproject the 3D points used by the motion constraint, as explained in §2.2.3.1.

4.2.2 *SfT-1*: Instantiating SfT for Creasable Surfaces

We form the problem instance *SfT-1* using the eight components given in §2.1. An illustration of *SfT-1* is given in figure 4.1. We instantiate the problem components *(a)* to *(h)* and give the reasons of each component specification. Table 4.1 presents the instantiations of the fundamental models, given in §4.2.1, for *SfT-1*.

Fundamental model	Known *a priori*	Fixed or time-varying	Instantiation
Template's shape	✓	Fixed	High-resolution thin-shell 3D mesh
Template's appearance	✓	Fixed	2D image
Template's deformation	✓	Fixed	Isometric, crease-preserving parameterized by barycentric interpolation
Camera projection	✓	Fixed	Perspective

Table 4.1: Fundamental model instantiations in *SfT-1*.

(a) Models. We use a high-resolution thin-shell 3D mesh for the template's shape and a barycentric interpolation for the template's deformation (preventing any dimensionality reduction) in order to be capable of modeling complex and unpredictable deformations. Deformation is modeled quasi-isometrically and creases are modeled as described in §4.2.5.4. We model the template's appearance with a known texture-map. We assume the perspective camera model [Hartley and Zisserman, 2003], which handles well most real-world cameras. *(b) Exploited visual cues.* The visual cues we use are motion and boundary contour constraints. We use motion because it is the main visual cue used in SfT. We use boundary contours for two reasons: to complement motion and to improve registration and the reconstruction of creases for weakly-textured objects. *(c) Number of required images.* A single image is required because we want to tackle the classical version of SfT. *(d) Expected types of deformations.* We assume quasi-isometric and piecewise-smooth deformations. We assume that there is no tearing. *(e) Scene geometry.* We assume no self or external occlusions, which is a typical assumption in the SfT state-of-the-art. There can be background clutter. *(f) Requirement for putative correspondences.* We assume to know *a priori* a set of putative 2D correspondences from the texture-map of the template to the input image. We assume there may be a small proportion of mismatches *e.g.* $< 20\%$. *(g) Surface texture characteristics.* We consider well-textured surfaces since it is an usual assumption in SfT. *(h) Known and unknown model parameters.* A template of the surface, as defined in §2.2.2, and the camera intrinsics are known. The unknowns are the vertices of the deformed template in 3D camera coordinates.

4.2.3 Template and Camera Modeling

We now define the specific models which we use for the template (*shape model*, *deformation model* and *appearance model*) and the camera.

The *shape model* is a thin-shell 3D mesh model in a known reference position, consisting of a set of M 3D vertices $\mathcal{Y} \triangleq \{\mathbf{y}_1, ..., \mathbf{y}_M\} \in \mathbb{R}^{3\times M}$ and F faces $\mathcal{F} \triangleq \{f_1, ..., f_F\}, f_k \in [1, M]^3$. Because we assume a high-resolution shape model, M is on the order of 10^4. We denote the set of mesh edges as $E \in [1, M]^{2\times N_E}$, where N_E is the number of edges. Because we assume the surface does not tear, the mesh edges and faces are fixed over time.

For the *appearance model*, we use a texture-map $\mathcal{T}(\mathbf{u}) : \mathbb{R}^2 \to \{0, 255\}^3$. It models the color at each point on the template's surface. In SfT, $\mathcal{T}$ is typically generated from photographs of the template in a static position [Agudo et al., 2016; Collins et al., 2014], but it can also come from a CAD model [Collins and Bartoli, 2015; TurboSquid, 2016; Warehouse, 2016]. In the simple case when the template's surface can be seen entirely in a single calibrated image, texture-mapping is particularly simple, and can be done by inverting the image projection, as shown in figure 4.1 (bottom left). We refer to the calibrated image as the reference image, and we assume the template is registered to the reference image. We denote by $\Pi_{\mathcal{T}}$ the projection function of the reference image's camera. We define $\Omega_{\mathcal{B}} \subset \Omega_{\mathcal{T}}$ as the boundary points of the texture-map. This modeling generalizes straightforwardly to multiple reference images, which may be required for templates with non-disc topology.

For the *deformation model*, we model the position of each vertex $i \in \{1, \dots, M\}$ in camera coordinates by an unknown vector $\mathbf{v}^i \in \mathbb{R}^3$. We transform a point $\mathbf{u} \in \Omega_{\mathcal{T}}$ on the texture-map to camera coordinates according to $\mathcal{V}$ with a barycentric interpolation φ, which is a linear interpolation of the positions of the three vertices surrounding $\mathbf{u}$, denoted by $\mathbf{v}^i$, $\mathbf{v}^m$ and $\mathbf{v}^n$:

$$\varphi(\mathbf{u}; \mathcal{V}) = b_1\mathbf{v}^i + b_2\mathbf{v}^m + b_3\mathbf{v}^n \in \mathbb{R}^3, \tag{4.1}$$

where b_1, b_2 and $b_3 = 1 - b_1 - b_2$ are the barycentric coordinates of the point $\mathbf{u}$ with respect to $\mathbf{v}^i$, $\mathbf{v}^m$ and $\mathbf{v}^n$. The barycentric interpolation therefore defines a piecewise-linear embedding function from $\Omega_{\mathcal{T}}$ to 3D, parameterized by the vertex positions. We denote the embedding function of the template in its reference position by $\varphi_{\mathcal{T}}$. We also assume isometry and crease-preserving smoothness and impose them through the cost function which we define in §6.2.5.

For the *camera model*, we use $\Pi : \mathbb{R}^3 \to \mathbb{R}^2$ to denote the perspective projection from camera coordinates to normalized pixel coordinates. All pixel positions are given in normalized pixel coordinates since we know the intrinsics camera parameters.

4.2.4 Inputs and Outputs

We now give our inputs. *(i)* one RGB input image $I : \mathbb{R}^2 \to \{0, 255\}^3$ showing a deforming object. *(ii)* a template of the surface, defined using §4.2.3. *(iii)* the camera intrinsics of the perspective projection function Π. *(iv)* a set of s putative 2D correspondences from the texture-map of the template to the input image. Correspondences can be computed using existing methods such as SURF or SIFT. We denote them by $\mathcal{S}_c = \{(\mathbf{u}_j, \mathbf{p}_j)\}$ where $\mathbf{u}_j$ denotes the correspondence position in $\Omega_{\mathcal{T}}$ and $\mathbf{p}_j$ denotes the correspondence position in I. These are assumed to be mostly correct with some mismatches due to ambiguous textures and other factors. Details for how this is done for our experimental datasets are given in §4.4.3.

Our solution to *SfT-1* outputs $\mathcal{V}$ the vertices of the deformed template in 3D camera coordinates.

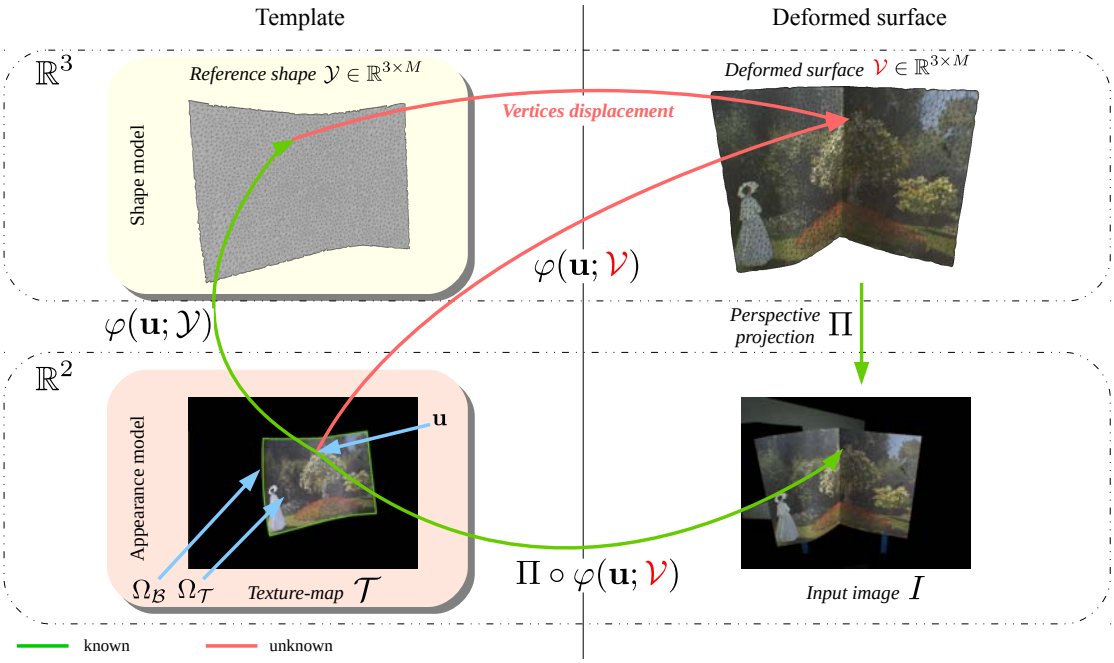


Figure 4.1: Geometric setup of SfT with the mesh parameterization and a texture-mapped template. We use red for the unknowns of our problem.

4.2.5 Problem Modeling with an Integrated Cost Function

We first write the cost function and then describe each term in detail. We solve the problem by combining *image data constraints* (motion and boundary contour constraints) and *physical deformation priors* (quasi-isometry and smoothing constraints). The cost function is as follows:

$$C_{total}(\mathcal{V}) \triangleq C_{motion}(\mathcal{V}) + \lambda_{contour} C_{contour}(\mathcal{V}) + \lambda_{iso} C_{iso}(\mathcal{V}) + \lambda_{smooth} C_{smooth}(\mathcal{V}). \tag{4.2}$$

The terms C_{motion} and $C_{contour}$ are motion and boundary contour data constraints respectively. The terms C_{smooth} and C_{iso} are physical deformation prior constraints, which respectively encourage the deformation to be generally smooth and quasi-isometric. The terms λ_{motion}, $\lambda_{contour}$, λ_{iso} and λ_{smooth} are positive weights and are the method's tuning parameters. Note that our weights are normalized: λ_{motion} by the number of correspondences, $\lambda_{contour}$ by the number of boundary points, λ_{iso} by the number of edges and λ_{smooth} by the area of the surface. To solve *SfT-1*, we solve the following minimization problem:

$$\min_{\mathcal{V}} C_{total}(\mathcal{V}). \tag{4.3}$$

4.2.5.1 The Motion Constraint

We recall from §4.2.4 that the set $\mathcal{S}_c = \{(\mathbf{u}_j, \mathbf{p}_j)\}$ holds s putative correspondences between the template surface and the input image. We compute these correspondences with an existing method. In experiments presented in §4.4, we used SURF features that were matched between

the texture-map and the input image with the graph-based method from [Collins et al., 2014][1]. The correspondences on the template's surface were then computed by back-projecting from the texture-map to the template's reference mesh. Note that virtually all feature-matching methods are never guaranteed to be mismatches free. We deal with this with a robust cost function defined as follows:

$$C_{motion}(\mathcal{V}) \triangleq \sum_{(\mathbf{u},\mathbf{p})\in\mathcal{S}_c} \rho\Big(\|\Pi\circ\varphi(\mathbf{u};\mathcal{V})-\mathbf{p}\|\Big), \tag{4.4}$$

where ρ is an M-estimator. This encourages the function φ to project each point $\mathbf{u}_j$ onto the input image at the correspondence position $\mathbf{p}_j$, but in a way that can tolerate mismatches through an M-estimator ρ.

4.2.5.2 The Boundary Contour Constraint

Principle. The aim of this constraint is to align $\Omega_{\mathcal{B}}$ to where it is visible in the input image. More precisely, it encourages the boundary of the surface to project to strong boundary-like edges in the input image. This constraint works for surfaces with disc topology. We discretize the boundary of $\Omega_{\mathcal{T}}$ to obtain a set of boundary pixels $\mathcal{B} \triangleq \{\mathbf{u}_{k\in[1,N_{\mathcal{B}}]}\}$, with $N_{\mathcal{B}}$ the number of boundary pixels. We then compute a 'boundariness map' for the input image $B : \mathbb{R}^2 \rightarrow \mathbb{R}^+$, where surface boundary locations behave like potential wells: high values of $B(\mathbf{p})$ correspond to a high likelihood of pixel $\mathbf{p}$ being on the boundary contour. The constraint is evaluated as:

$$C_{contour}(\mathcal{V}) \triangleq \frac{1}{|N_{\mathcal{B}}|}\sum_{\mathbf{u}_k\in\mathcal{B}} \rho\Big(B\big(\Pi\circ\varphi(\mathbf{u}_k;\mathcal{V})\big)\Big). \tag{4.5}$$

where $\rho(\mathbf{x}) = 2(\sqrt{1+\|\mathbf{x}\|_2^2/2}-1)$ is the (ℓ_1-ℓ_2) M-estimator. We use it to reduce the influence of false boundary points on the cost function.

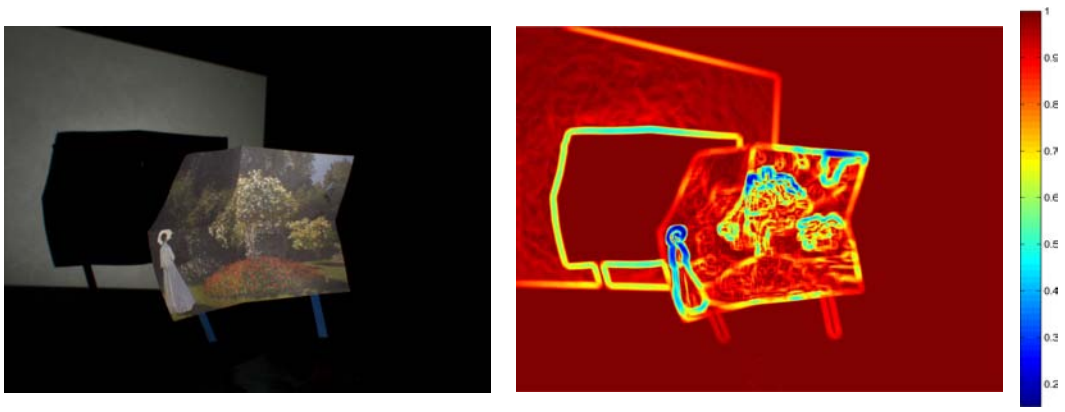
Baseline implementation. We first define a baseline boundariness map B by considering only image gradient magnitude. We construct an edge response filter which acts as a potential well. We compute it in two steps. First, we compute a blurred grayscale version of the input image I using a Gaussian filter with the parameters (h,σ), where h is the kernel size and σ the standard deviation. We denote it by G. Second, we use the following formula:

$$B = \exp\left(-\frac{|\nabla G|}{s}\right), \tag{4.6}$$

where ∇G is the gradient of the blurred image G and s is the bandwidth of the potential well. In figure 4.2, we show an input image (left) and its corresponding boundariness map (right) according to equation (4.6). The true boundaries are represented with low potentials, but so are many false boundaries corresponding to background clutter and texture edges. Figure 4.3 also illustrates this presence of ambiguous boundaries. This is a serious problem

[1]The code is available at igt.ip.uca.fr/~ab/Research/GAIM_v1p2.zip

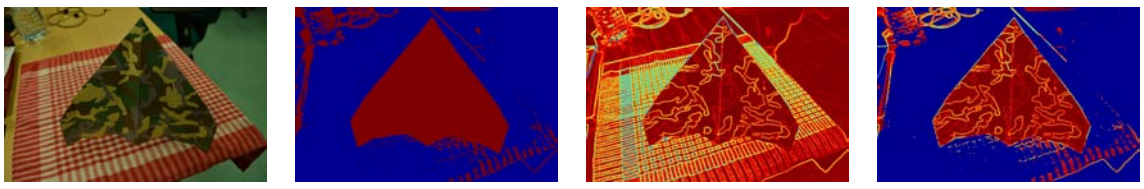
because they may attract the solution to a wrong local minimum.



(a) Input image (b) Baseline boundariness map

Figure 4.2: Baseline boundariness map. (a) input image, (b) baseline boundariness map computed with $(h, \sigma, s) = (10, 5, 0.1)$.

Enhanced implementation. To reduce false positives, we build an edge response filter which is modulated to suppress false positives according to one or more segmentation cues. The right cue depends on the particular dataset, for example color distribution if the background is constant over the image set or if the object has a distinct color distribution to the background. For the datasets used in this chapter, we propose to exploit color information to significantly reduce false boundary edges. This works by applying a color-based foreground detector, trained on the target surface, to each input image pixel, and setting $B(\mathbf{p}) = 1$ for any pixel at position $\mathbf{p}$, which has a detection score below a threshold T_d. We define the target surface as an estimation of the foreground in the input image and explain in §4.3.3.2 how we obtain the target surface. We train the detector using the target surface and use a default threshold of $T_d = 50$. In our experiments, we use an RGB Gaussian Mixture Model of 4 components. In figure 4.3, we show the differences between the baseline boundariness map and the enhanced boundariness map using the color-based statistical filter. Here we see that many false boundary edges in the background have been removed.



Input image Color model segmentation Baseline boundariness map Enhanced boundariness map

Figure 4.3: Comparison of baseline and enhanced boundariness maps.

4.2.5.3 A Robust Smoothing Constraint that Supports Crease Formation

The need for regularization and limitations of ℓ_2 regularization. The quasi-isometry constraint penalizes within-plane stretching or shearing, however it does not penalize curvature change. Therefore, it is insufficient to use as a regularizer to penalize curvature change.

In SfT, the ℓ_2 norm of the curvature is usually used as a regularizer in order to penalize non-smooth deformations. However, this does not allow to handle creases [Brunet et al., 2014]. The main reason is that large residuals, caused by high changes of curvature (from creases) have a very strong impact on the cost function. In our case, the data constraints (the boundary contour constraint) may support the formation of creased regions, however the ℓ_2 norm of the curvature can nullify their influence and all creases will be smoothed.

How to model creases with an M-estimator? Various M-estimators have been proposed in the literature [Zhang, 1997]. Each M-estimator has slightly different qualities and this is why it is very hard to know *a priori* which one works best for a given problem. M-estimators can either be parameterless, such as (ℓ_1-ℓ_2), or have at least one free parameter (usually it is only one), such as Huber and Tukey. They may also be redescending (Tukey) or non-redescending ((ℓ_1-ℓ_2) and Huber). Table 4.2 gives the definitions of the following M-estimators: (ℓ_1-ℓ_2), Huber and Tukey. Figure 4.4 compares them with ℓ_1 and ℓ_2. We display these functions at two scales, (a) and (b). It is difficult to know *a priori* which M-estimators work for our problem. Therefore, before investigating this with experimental validations, we propose to show why using an M-estimator on the regularizer can enable crease formation. We then discuss the implications for using redescending versus non-redescending M-estimators and for using sparsity-preserving M-estimators.

(ℓ_1-ℓ_2)	Huber	Tukey
$\rho(x) = 2\left(\sqrt{1+\frac{x^2}{2}} - 1\right)$	$\begin{cases} \text{if } \lvert x\rvert \leq k, & \rho(x) = \frac{x^2}{2} \\ \text{if } \lvert x\rvert \geq k, & \rho(x) = k\left(\lvert x\rvert - \frac{k}{2}\right) \end{cases}$	$\begin{cases} \text{if } \lvert x\rvert \leq k, & \rho(x) = \frac{k^2}{6}\left(1 - \left[1 - (\frac{x}{k})^2\right]^3\right) \\ \text{if } \lvert x\rvert > k, & \rho(x) = \frac{k^2}{6} \end{cases}$

Table 4.2: Definition of three main M-estimators.

Why penalizing curvature changes with M-estimators can enable crease formation? Compared to the ℓ_2 norm, M-estimators reduce the impact on the cost function of high residuals. In figure 4.4 (a), we observe that M-estimator functions grow sub-quadratically at high residuals. Regarding our problem, high residuals in the regularizer correspond to high changes of curvature, which occur at creased regions. Therefore, the impact of high residuals on the optimization of the regularizer will be much smaller when using an M-estimator rather than the ℓ_2 norm. This means that M-estimators do not discourage so much the emergence of creases.

One important point to consider is that the data terms and the regularizer usually compete with each other. Data terms encourage the solution to fit to the data and the regularizer encourages the solution to have generally low curvature changes. The exact

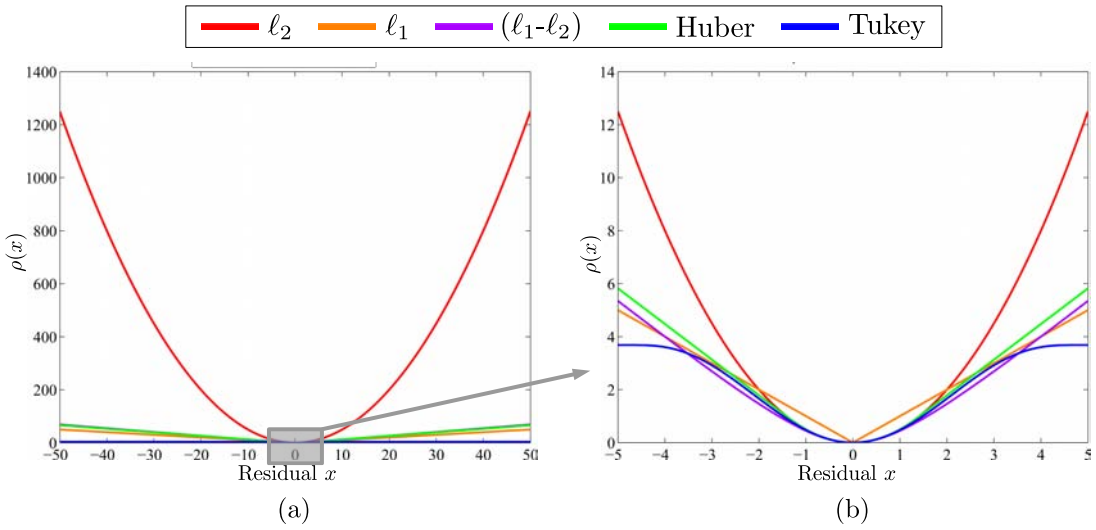


Figure 4.4: Graphic representations of the ℓ_1 and ℓ_2 norms and the three studied M-estimators. The functions representation in (b) is a zoom-in of the functions representation in (a) between $x \in [-5, 5]$. The Huber and Tukey M-estimators are computed using respectively $k = 1.345$ and $k = 4.6851$. These values are proposed in [Zhang, 1997] to obtain an asymptotic efficiency of 95% for residuals distributed with the standard normal distribution.

behavior of the regularizer is likely to have a strong influence on the ability to form creases at regions for which there are weak or no data constraints, such as at poorly-textured regions. The behavior of the regularizer will be determined mainly by whether it is a redescending or non-redescending M-estimator and whether or not it promotes sparsity.

What are the implications for using redescending versus non-redescending M-estimators? We consider specifically gradient-based optimization where redescending M-estimators have vanishing gradients after a certain threshold. Table 4.2 and figure 4.4 illustrate this point with the Tukey M-estimator. Therefore, regarding our problem, the presence of vanishing gradients implies that, if the initial solution is overly creased, incorrect creases would not be smoothed by the regularizer. Another potential problem with redescending M-estimators is that, if creases were to form in the wrong region during gradient-based optimization *e.g.* because of initial misregistration, then a redescending M-estimator will likely not encourage the region to undo the creases during optimization.

By contrast, non-redescending M-estimators have always non-zero gradients, except usually at the origin. Table 4.2 and figure 4.4 illustrate this point with the (ℓ_1-ℓ_2) and Huber M-estimators. This implies that non-redescending M-estimators applied on the regularizer act to smooth every region of the surface, whether it is smooth or creased. This implication may be a limitation regarding the problem that we want to solve. However, an advantage is that it may smooth creases which are not provided by the data terms.

What are the implications for using sparsity-preserving M-estimators (e.g. ℓ_1)? The ℓ_1

norm is known to promote sparsity. The ℓ_1-sparsity means that the obtained solution is sparse, *i.e.* presents many zero-residuals. This has been first shown empirically with results on seismologic applications [Claerbout and Muir, 1973; Taylor et al., 1979], and then rigorous results have been proposed to prove the ability of ℓ_1 to form sparse vectors [Donoho and Stark, 1989]. Studying the implications for using sparsity-preserving M-estimators is important for our problem because, for our smoothing constraint, creases can appear thanks to the formation of non-zero-residual regions. Indeed, in our problem, sparsity-preserving M-estimators could act to nullify residuals at some regions of the surface, which corresponds to having no curvature, *i.e.* a flat surface, and let some other regions with very high residuals, *i.e.* high changes of curvature, leading to creases. In the case of a flat template, this may lead to a piecewise-planar surface, unless making some residuals non-zero reduces notably some other constraints in the global cost function, such as motion and boundary contour constraints.

Another implication is that such sparsity-preserving M-estimators would exaggerate the strongest creases and make planar smaller creases and smooth regions. These implications make sparsity-preserving M-estimators, *e.g.* ℓ_1, not suitable for our problem.

Following the discussion above, we propose to use on the smoothing constraint a robust estimator based on M-estimators [Zhang, 1997]. This will lead to a discontinuity-preserving smoother which automatically deactivates smoothing where needed at creased regions. Precisely, our smoothing constraint penalizes the surface curvature change using a robust second-order total variation of φ as follows:

$$C_{smooth}(\mathcal{V}) \triangleq \frac{1}{|\Omega_{\mathcal{T}}|} \sum_{\mathbf{u}_j \in \Omega_{\mathcal{T}}} \rho\left(\frac{\partial^2 \varphi}{\partial \mathbf{u}^2}(\mathbf{u}_j; \mathcal{V})\right), \tag{4.7}$$

where ρ is a robust penalization based on an M-estimator, which allows crease formation. In §4.4.5.1, we investigate the performance of three M-estimators: (ℓ_1-ℓ_2), Huber and Tukey. We found that (ℓ_1-ℓ_2) works best.

4.2.5.4 The Quasi-Isometry Constraint

The quasi-isometry constraint is used to penalize surface extension and compression, and is required in general to make the SfT problem well posed. We use a standard definition which penalizes deviation of the template mesh's edges between the rest and deformed positions [Salzmann et al., 2007a]. We define the quasi-isometry constraint as:

$$C_{iso}(\mathcal{V}) \triangleq \frac{1}{|E|} \sum_{(i,j)\in E} \left(1 - \|\mathbf{y}^i - \mathbf{y}^j\|_2^{-2} \, \|\mathbf{v}^i - \mathbf{v}^j\|_2^2\right)^2. \tag{4.8}$$

4.3 Optimization Strategy

4.3.1 Overview

Equation (4.2) leads to a large-scale, non-convex optimization problem, which cannot be solved globally. Because we use a triangulated mesh parameterization, all constraints are sparse with respect to the unknowns, the vertex set $\mathcal{V}$. The system is sparse because each constraint only depends on a small number of vertices. However, the difficulty is that we have to handle a sparse system which is large-scale (typically $\mathcal{O}(10^4)$ unknowns). For instance, the shape to reconstruct in figure 1.6 involves $11,557$ unknowns and approximately $300,000$ constraints. We propose to solve these challenges in two steps: we compute an initial estimate $\mathcal{V}_0$ using an existing SfT method which only uses motion, then we refine the solution with a numerical quasi-newton minimization which uses the boundary contour constraint and the crease-preserving smoothing constraint. Figure 4.5 illustrates the whole process.

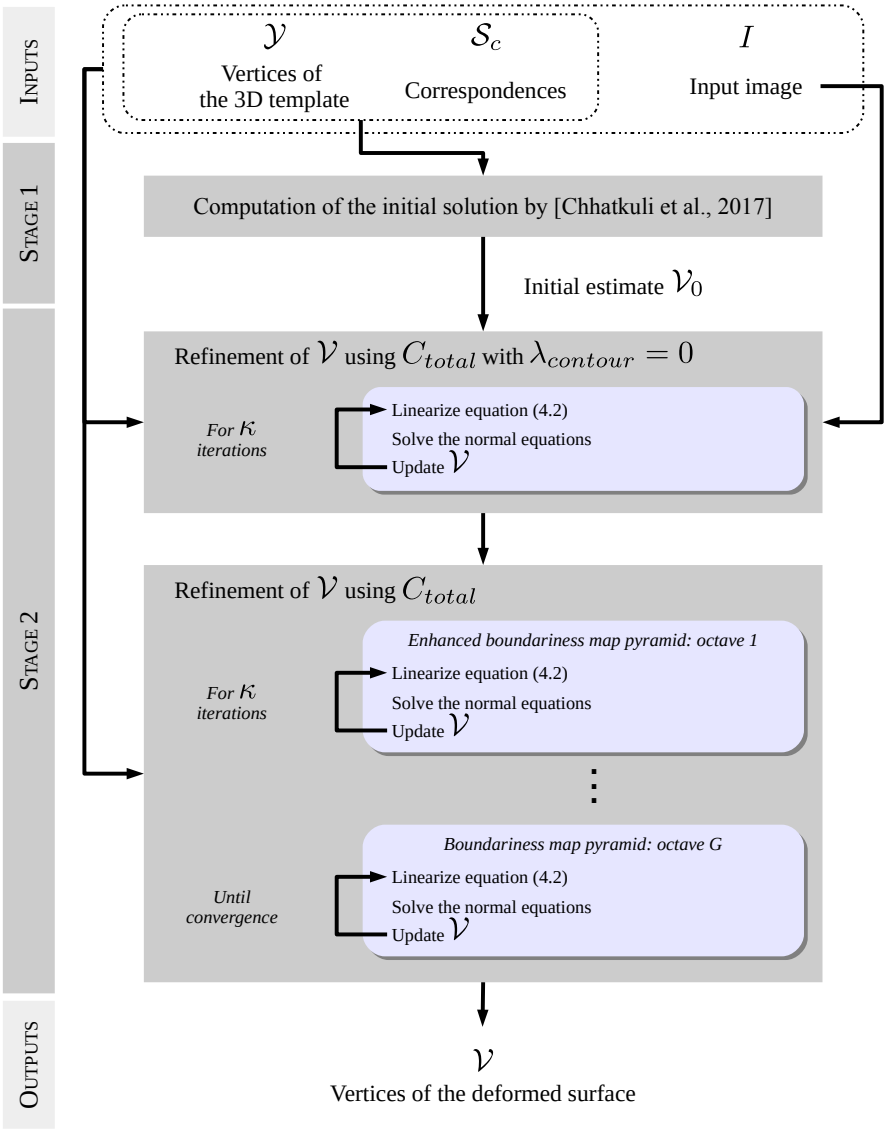


Figure 4.5: Schematic of our proposed solution to solve *SfT-1*.

4.3.2 Stage 1: Motion-Based Initialization

We determine $\mathcal{V}_0$ using an existing SfT method [Chhatkuli et al., 2017][2] which does not need any initial estimate. This method relaxes the isometry constraint by computing a non-holonomic solution to a PDE built from correspondence points. It does not use boundary contour constraints and also assumes that the surface is smooth.

4.3.3 Stage 2: Crease-Preserving SfT Refinement

Having initialized, we refine C using Gauss-Newton (GN) iterations, implemented with Iteratively Reweighted Least Squares (IRLS). To ensure convergence, we use backtracking line-search [Armijo, 1966]. We also propose two additional strategies to improve the convergence of the boundary contour constraints.

4.3.3.1 Optimization with IRLS

We now show how our SfT formulation leads us to solving a weighted linear least-squares system. At each iteration, we use GN to update an estimate $\mathbf{x}_0$ of $\mathbf{x}$ with $\mathbf{x} \leftarrow \mathbf{x}_0 + \Delta\mathbf{x}$. To find $\Delta\mathbf{x}$, we build and solve the associated normal equations. Let N_c be the total number of constraints in the system. Each constraint has an associated M-estimator $\{\rho_i\}_{i\in[1,N_c]}$. We can write the cost function as the sum:

$$C_{total}(\mathbf{x}) = \sum_{i=1}^{N_c} C_i(\mathbf{x}), \quad \text{and} \quad C_i(\mathbf{x}) \stackrel{\triangle}{=} \rho_i(r_i(\mathbf{x})). \tag{4.9}$$

where r_i denotes the residual associated to the i^{th} constraint. Then, we linearize C_i using the chain rule:

$$C_i(\mathbf{x}) \quad \approx \quad C_i(\mathbf{x}_0) + \frac{\partial C_i}{\partial \mathbf{x}}(\mathbf{x})\Delta\mathbf{x} \quad \approx \quad C_i(\mathbf{x}_0) + w_i(r_i(\mathbf{x}))r_i(\mathbf{x})\frac{\partial r_i}{\partial \mathbf{x}}(\mathbf{x})\Delta\mathbf{x}, \tag{4.10}$$

with the weight function $w_i(r_i(\mathbf{x})) \stackrel{\triangle}{=} \frac{\partial \rho_i}{\partial r_i}(r_i(\mathbf{x}))/r_i(\mathbf{x})$. Note that, when ρ_i is the ℓ_2 norm, $\omega_i = 1$. Then, using equation (4.9) with the approximation from equation (4.10), we obtain:

$$\left(\min_{\mathbf{x}} C_{total}(\mathbf{x})\right) \Rightarrow \left(\min_{\mathbf{x}} \sum_{i=1}^{N_c} w_i(r_i(\mathbf{x}_0))r_i^2(\mathbf{x})\right). \tag{4.11}$$

Equation (4.11) gives a weighted linear least-squares system which we solve thanks to sparse Cholesky decomposition. To ensure $C_{total}(\mathbf{x}_0 + \Delta\mathbf{x}) \leq C_{total}(\mathbf{x}_0)$, the magnitude of $\Delta\mathbf{x}$ is adapted using backtracking line-search. Then, we update the unknowns $\mathbf{x} \leftarrow \mathbf{x}_0 + \Delta\mathbf{x}$ and repeat this process.

[2]The code is available at http://igt.ip.uca.fr/~ab/Research/SfT_v0p2.zip

4.3.3.2 Improving convergence

One caveat is that the boundary contour constraints are highly non-convex and can cause convergence to the wrong local minimum. The color-based filtering described in §4.2.5.2 partially deals with this, however we also introduce two more strategies.

The first strategy is to cascade the constraints, by first optimizing the solution without using boundary contour constraint ($\lambda_{contour} = 0$) for the first few iterations (we use $\kappa = 10$ iterations). Once done, the detector presented in §4.2.5.2 is trained using the region of the input image that overlaps with the deformed template. Then, boundary contour constraints are introduced and the solution is refined. Figure 4.5 illustrates this cascade strategy. The target surface used in the boundary contour constraint in §4.2.5.2 is obtained by projecting in the input image the surface which is estimated by the refinement without using boundary contour constraint. The second strategy is to use an image pyramid, which gives coarse-to-fine versions of the boundariness map and increases the convergence basin. This is a standard practice used in related problems such as optical flow. We currently use a three-level pyramid ($G = 3$): the kernel sizes and standard-deviations are respectively $h_1 = (10, 10)$ and $\sigma_1 = 5$ and $h_2 = (5, 5)$ and $\sigma_2 = 2.5$ for a default image size of 1288×964 pixels. At the finest level, we do not apply the color-based filtering to the boundariness map. This is because assuming correct convergence, at the start of the finest level the boundaries should align reasonably closely to their true locations, and we therefore have less risk of false boundary edges steering the solution away to a wrong local minimum. The benefit is to use all edge information at the finest level, including edges where there is little color separation between the surface and its background. For coarse levels of image pyramid, we optimize C_{total} for $\kappa = 10$ iterations, and, for the finest level, we optimize C_{total} until convergence.

4.4 Experimental Validation

4.4.1 Methods Compared

We compared the accuracy of our method with four competitive SfT methods with publicly available code [Bartoli et al., 2015; Chhatkuli et al., 2017; Ngo et al., 2016; Salzmann and Fua, 2009], which we denote respectively **Ba15a**, **Ch17a**, **Sa09a** and **Ng16a**. **Sa09a** refers to the convex formulation of [Salzmann and Fua, 2009]. Table 4.3 gives the main differences between these four methods. We use the star * to denote a proposed method. We denote our method of §4.3 by **Ga16a***.

Acronym	Source	Shape model	Constraints	Principle
Ga16a*	Proposed	Mesh	M+C+I+Ss	Initialization (**Ch17a**) + iterative non-linear refinement
Ba15a	[Bartoli et al., 2015]	B-splines	M+I+Ss	Computation of non-holonomic solutions for **depth** from a system of partial differential equations + non-linear refinement [Brunet et al., 2014]
Ch17a	[Chhatkuli et al., 2017]	B-splines	M+I+Ss	Computation of non-holonomic solutions for **depth** and **gradient** from a system of partial differential equations + non-linear refinement [Brunet et al., 2014]
Sa09a	[Salzmann and Fua 2009]	Mesh	M+Inex	Convex formulation (**Maximum Depth Heuristic**), where the surface with the maximum depth is favored, then SOCP
Ng16a	[Ngo et al., 2016]	Laplacian mesh	M+Inex+Ss	Analytical resolution of a linear system built from correspondences and using a **Laplacian mesh** (which favours smooth solution) + non-linear refinement

M: Motion, C: boundary Contour, I: Isometry, Inex: Inextensibility, Ss: Surface smoothing

Table 4.3: List of SfT methods used for the comparison. We give their specific components of modeling and optimization.

4.4.2 Ground-Truth Acquisition Setup

Some previous datasets with ground-truth 3D exist [Salzmann et al., 2007a], however these are low resolution, noisy and do not contain creased surfaces. To accurately evaluate our method, new datasets with ground truth were required. We constructed three new datasets of three different objects with a highly-accurate commercial structured light system [David 3D Scanner, 2014]. This consists of a HD data projector and an industrial machine vision camera [Point Grey] and the latest SDK FlyCapture2. This system captures depth-maps to sub-millimeter accuracy. Another strong advantage of this setup is that the depth-maps are constructed in the camera's coordinate frame, so there is no need to register them to the camera's image. Figure 4.6 shows this setup. RGB images were captured from the camera [Point Grey] at a resolution of 1288×964 pixels. It takes approximately 10 seconds to capture an image and its associated depth-map.

4.4.3 Datasets

Our datasets consist of three creased objects scanned at approximately 20 cm using the structured light system described in §4.4.2: a *Monet paper*, a *folded aeroplane* and a *cardboard box*, shown in figure 4.11. Correspondences were computed using the public code from [Collins et al., 2014]. We also evaluated the accuracy of our method on an existing smooth dataset [Varol et al., 2012a], called *Kinect paper*, to assess how our approach coped when creased reconstruction was not required. The details of the datasets are given in table 4.4. To show the amount and distribution of the correspondences computed for each dataset, we display in figure 4.7 the correspondences for one input image for each dataset.

For the three creased datasets, we counted on average the mismatches by counting for each image of each dataset the mismatches for a set of 100 correspondences randomly. For the

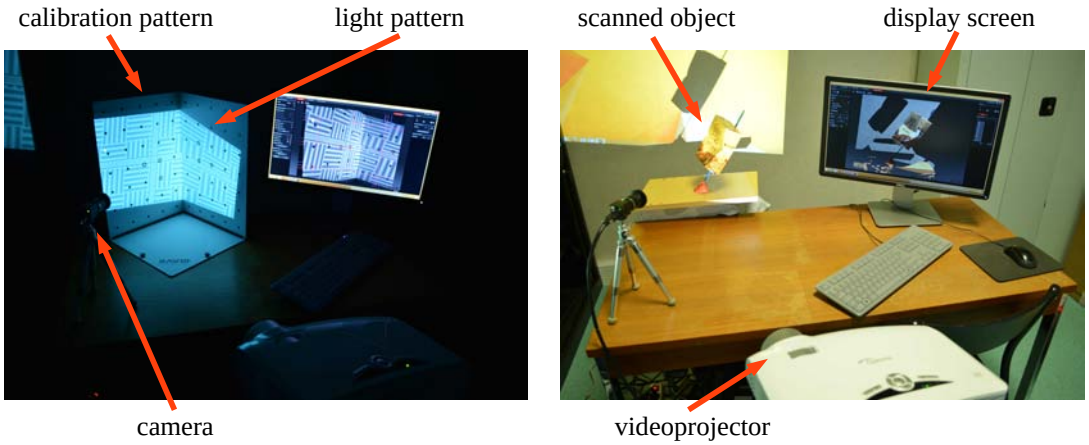


Figure 4.6: Acquisition system of high-accuracy ground-truth 3D surfaces. We use a structured light system [David 3D Scanner, 2014] to create our datasets with a sub-millimeter accuracy. **Left:** picture of the setup during calibration. The videoprojector projects a sequence of calibration pattern, which is recorded by the camera (whose live video is shown in the display screen). **Right:** picture of the setup after the acquisition of one deformation.

Monet paper, the *folded aeroplane* and the *cardboard box* datasets, we respectively counted 14.5%, 9.3% and 8.8% of mismatches. For the *Kinect paper* dataset, there is no mismatch.

Name	Nb of images	Nb of corresp.	Matching methods	GT available	Template construction
Monet paper	6	≈ 470	[Collins et al., 2014]	✓	§4.4.2
folded aeroplane	9	≈ 320	[Collins et al., 2014]	✓	§4.4.2
cardboard box	8	≈ 450	[Collins et al., 2014]	✓	§4.4.2
Kinect paper	40	≈ 988	[Lowe, 2004]	✓	[Salzmann and Fua, 2009]

Table 4.4: List of well-textured surfaces datasets for SfT comparison.

Figure 4.7: Visualization of the correspondences on one input image for each dataset for the *SfT-1* problem. We show the correspondences between the texture-map and one input image. **Row n°1**: input image n°1 of the *Monet paper* dataset. **Row n°2**: input image n°1 of the *folded aeroplane* dataset. **Row n°3**: input image n°6 of the *cardboard box* dataset. **Row n°4**: frame n°101 of the *Kinect paper* dataset.

4.4.4 Implementation Details and Evaluation Metrics

For all experiments, we constructed the templates by laying a triangulated 100×100 vertex regular grid on the texture-map, defined in §2.2.2.2, which was then cropped to $\Omega_{\mathcal{T}}$. We found that this resolution was sufficient to accurately reconstruct creases. We discretized the boundary points of the texture-map to $N_{\mathcal{B}} = 1000$ uniformly spaced points. For the state-of-the-art methods, there is no way to automatically tune their free parameters. Therefore we tried our best to do this by hand, to obtain the best 3D mean point error on all datasets. This was done by a search starting from the default values, and modifying each free parameter in turn to improve the 3D mean point error. For our method, all experiments were ran using the same parameters, which were manually set. We give these parameters for each dataset in appendix B.

We measured reconstruction accuracy by comparing 3D distances and normals with respect to ground-truth. On the datasets with creased surfaces, we used two evaluation metrics. As we investigated the improvement at creased regions, we averaged results with two evaluation grids: *(i)* densely across the template, and *(ii)* densely at only creased regions. The first grid was built by uniformly sampling the texture-map at 5 px intervals. The second grid used local regions around each crease, with a neighborhood distance of approximately 5 mm.

3D mean point error (MPE): This measures the average relative depth error over the evaluation grid $\mathcal{G}$. We denote as $Q^* : \mathbb{R}^3 \to \mathbb{R}^3$ the function which gives the 3D point of the ground-truth surface closest to the input 3D point. We computed this (in %) between the reconstructed surface $\mathcal{V}$ and the ground-truth surface on $\mathcal{G}$:

$$MPE(\mathcal{V}, Q^*, \mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\mathbf{u} \in \mathcal{G}} \frac{\|\varphi(\mathbf{u}; \mathcal{V}) - Q^*(\varphi(\mathbf{u}; \mathcal{V}))\|}{\|\varphi(\mathbf{u}; \mathcal{V}^*)\|}. \tag{4.12}$$

Mean normal error (MNE): This measures the average error in surface normal over the grid $\mathcal{G}$. We denote as $n^* : \mathbb{R}^3 \to \mathbb{S}_3$ the function which gives the 3D normal of the 3D ground-truth point closest to the input 3D point. We computed this (in degrees) between the reconstructed surface $\mathcal{V}$ and the ground-truth surface on $\mathcal{G}$:

$$MNE(\mathcal{V}, n^*, \mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\mathbf{u} \in \mathcal{G}} \cos^{-1}\left(n^\top(\mathbf{u}; \mathcal{V})\, n^*(\varphi(\mathbf{u}; \mathcal{V}))\right), \tag{4.13}$$

with $n(\mathbf{u}; \mathcal{V}) : \mathbb{R}^{3\times M} \to \mathbb{S}_3$, the unit normal at $\mathbf{u}$.

4.4.5 Results

The first part covers our investigation to see whether the choice of M-estimator in the smoothing constraint has a significant impact on reconstruction accuracy. The second part compares our results with the state-of-the-art methods using the new high-quality evaluation datasets. The third part compares our results with state-of-the-art methods using the dataset from [Salzmann et al., 2007a].

4.4.5.1 Comparing Different Smoothing Energy M-estimators

We systematically compared the performance of two non-redescending M-estimators ((ℓ_1-ℓ_2) and Huber) and one redescending M-estimator (Tukey). We do this in three steps. First, to see if the free parameter was sensitive and required careful tuning. Second, to see whether there was a significant performance difference between these M-estimators. Third, to see if redescending M-estimators can be used for our problem, we evaluated our method with the redescending M-estimator Tukey.

We evaluated our method systematically with the two non-redescending M-estimators, (ℓ_1-ℓ_2) and Huber. For this, we use 17 different M-estimator settings on the smoothing constraint and corresponding smoothing weight (λ_{smooth}). The first 16 were with Huber using 16 different k-values in the range $k \in [1e{-}6, 1e2]$. The 17^{th} was with (ℓ_1-ℓ_2). We evaluated performance using all input images in all three new datasets. For each M-estimator setting, we ran our method with 9 different smoothing weights from λ_{smooth}, from $[8e{-}14, 8e{-}10]$ (which was a sufficient range), and then took the smoothing weight which produced the lowest 3D mean point error. The corresponding results for all 17 M-estimator settings are shown in figure 4.8. We observe that with (ℓ_1-ℓ_2) we obtain very similar results to the best result obtained with Huber. The best Huber k-value changes according to the error metric, but we can say reasonably that the best k varies between $k_7 = 0.05$ and $k_9 = 0.005$. This suggests that for our problem, given a well-chosen smoothing weight there is no clear difference between using (ℓ_1-ℓ_2) or Huber, and the choice for Huber's free parameter is important but not extremely sensitive in the range $k_7 = 0.05$ to $k_9 = 0.005$.

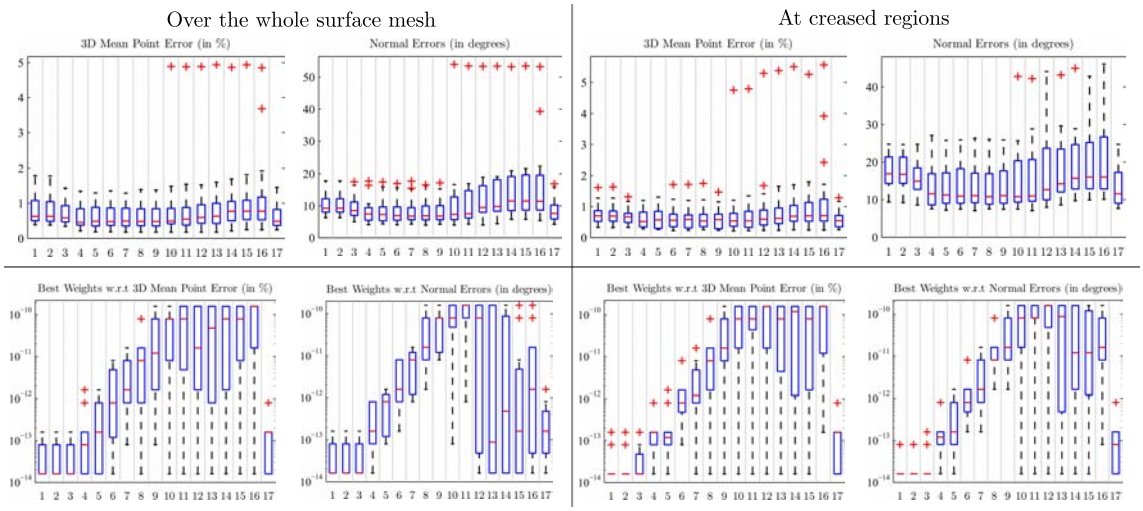


Figure 4.8: Results of smoothing energy estimator analysis. We consider all input images of all the objects of our three datasets, *Monet paper*, *folded aeroplane* and *cardboard box*. **Row n°1**: Errors obtained by running our method with 17 different estimators settings. In the first 16 settings we use Huber with 16 different hyper-parameter values. In the setting n°17, we use (ℓ_1-ℓ_2). **Row n°2**: the distribution of optimal weights λ_{smooth} for each estimator setting.

We then investigated the range of smoothing weights for a given M-estimator that produces good reconstructions. The purpose was to see how easy it is in practice to tune the

smoothing weight for a given M-estimator. This was done by evaluating the best smoothing weight for each test image independently, then quantifying the corresponding spread of reconstruction accuracy. The results are shown in figure 4.8. Figure 4.8 shows that in general the spread of the best smoothing weight is similar for (ℓ_1-ℓ_2) and Huber with its parameter in the range $k \in [k_7, k_9]$. This implies that the difficulty of choosing a good weight for the smoothing constraint is the same for the two robust estimators. From these experiments, we can conclude that there is very little difference in practice between using (ℓ_1-ℓ_2) or Huber with its parameter in the range $k \in [k_7, k_9]$.

To study the third point, we manually tuned the free parameter of Tukey and the weight of the smoothing constraint λ_{smooth} to obtain the best possible reconstructions. We could not find it for the redescending M-estimator Tukey. The best reconstruction obtained with the Tukey M-estimator is far from the reconstructions obtained with the two non-redescending M-estimators. More precisely, strong changes of curvature cannot be reconstructed with the Tukey M-estimator. For the input image of the *Monet paper* dataset shown in figure 4.11, the best reconstruction with the Tukey M-estimator is smooth at the crease in the middle: the normal error at the crease is 17.91 degrees for the Tukey M-estimator, compared to 8.46 degrees for (ℓ_1-ℓ_2) and 9.54 degrees for the Huber M-estimator. We note that it is hard to use the Tukey M-estimator due to its non-convexity. However, we cannot conclude that the Tukey M-estimator do not allow to model creases.

4.4.5.2 Results on Creased Datasets

To compare our method (**Ga16a***), we used the (ℓ_1-ℓ_2) M-estimator for the smoothing constraint with a corresponding weight of $\lambda_{smooth} = 8\text{e}{-13}$. In appendix B, we give the weights of the different constraints and the hyperparameters for our method and the compared methods.

Figure 4.11 shows the results of the compared methods using a representative input image from each of the three datasets. In figure 4.9, we give summary statistics for each method across all images. These visual observations support the statistical results in figure 4.9. We notice that our method provides the best accuracy at the neighbors of the creases and a best global reconstruction. We remark that the large smooth regions of the surfaces are also reconstructed well in general. In figure 4.11, 5^{th} row, 3^{rd} column, we show an example of a failure mode, where the reconstruction at the bar code does not appear correct. The reason for this is that the boundary points were incorrectly fitted to the bottom of the bar code, which was compensated by the surface bending away from the camera in order to respect the isometry constraint. The reconstruction accuracy of our method given in figure 4.9 and the reconstructions presented in figure 4.11 also suggest that mismatches were mostly rejected by the M-estimator applied on the motion constraint.

Figure 4.10 illustrates the interest of the enhanced boundariness map to avoid false boundary edges. We observe that the region shown in the green rectangle in figure 4.10 (middle bottom) is better reconstructed than the one in figure 4.10 (right bottom) since many false boundary edges in this region were removed in the enhanced boundariness map.

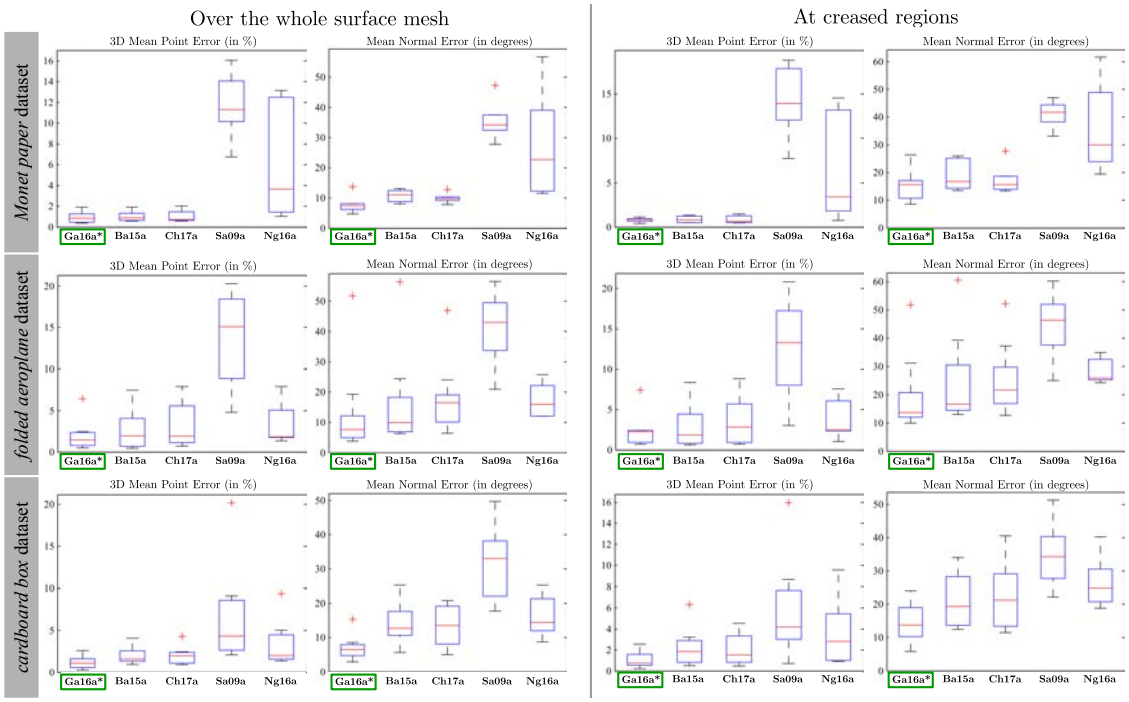
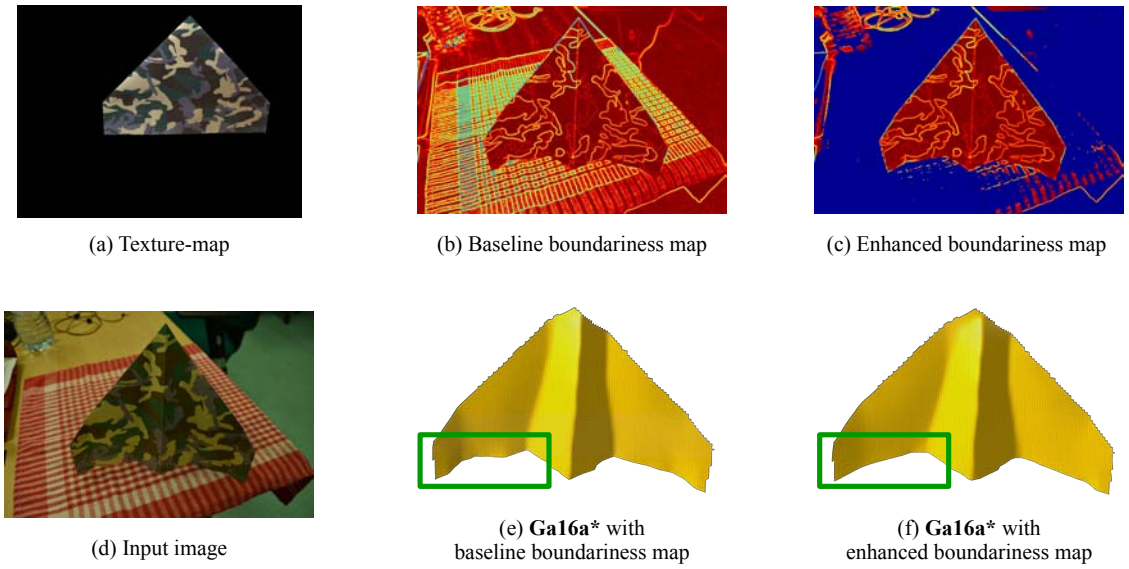


Figure 4.9: Reconstruction accuracy on the three new datasets. **Row n°1**: *Monet paper* dataset. **Row n°2**: *folded aeroplane* dataset. **Row n°3**: *cardboard box* dataset.



(a) Texture-map

(b) Baseline boundariness map

(c) Enhanced boundariness map

(d) Input image

(e) **Ga16a*** with baseline boundariness map

(f) **Ga16a*** with enhanced boundariness map

Figure 4.10: Comparison of 3D reconstructions using the baseline and enhanced boundariness maps.

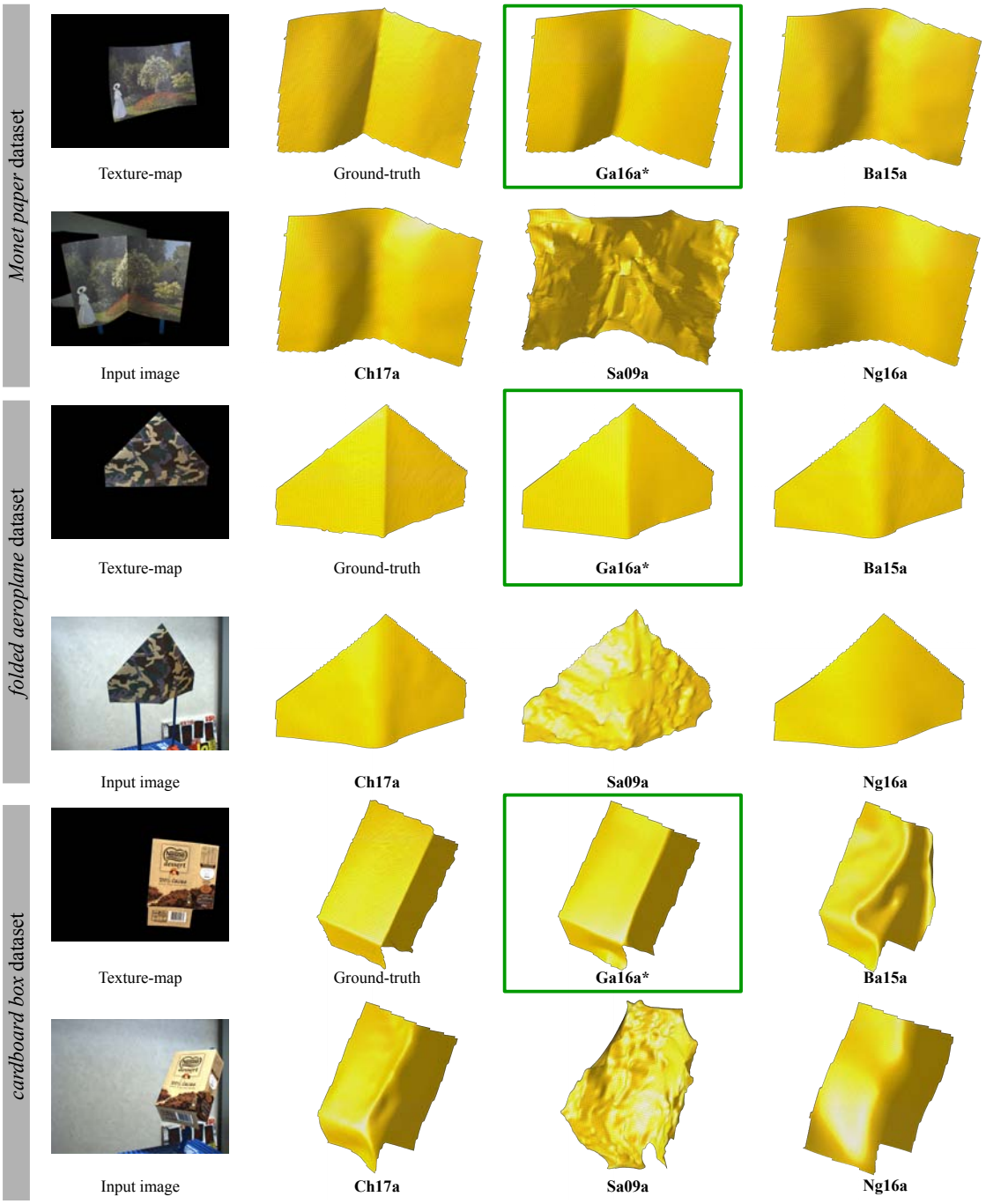


Figure 4.11: The three creased datasets and results visualizations for the *SfT-1* problem. **Rows n°1** and **n°2**: input image n°1 of the *Monet paper* dataset. **Rows n°3** and **n°4**: input image n°1 of the *folded aeroplane* dataset. **Rows n°5** and **n°6**: input image n°6 of the *cardboard box* dataset.

4.4.5.3 Results on an Existing Smooth Dataset

We also tested whether our method also works well for simpler problems with smooth surfaces where ℓ_2 regularization is sufficient. We have found this to be the case with existing benchmark datasets. On the commonly-used public EPFL Kinect paper dataset[3] (193 frames) and using the same parameters, we evaluated accuracy using 40 images uniformly sampled, which produced a 3D mean point error of 5.63 mm. This puts it close to the best performing method, presented in [Chhatkuli et al., 2017], which uses an ℓ_2 regularization and which gives a mean 3D mean point error of 5.74 mm. Figure 4.12 shows the results of our method and [Chhatkuli et al., 2017].

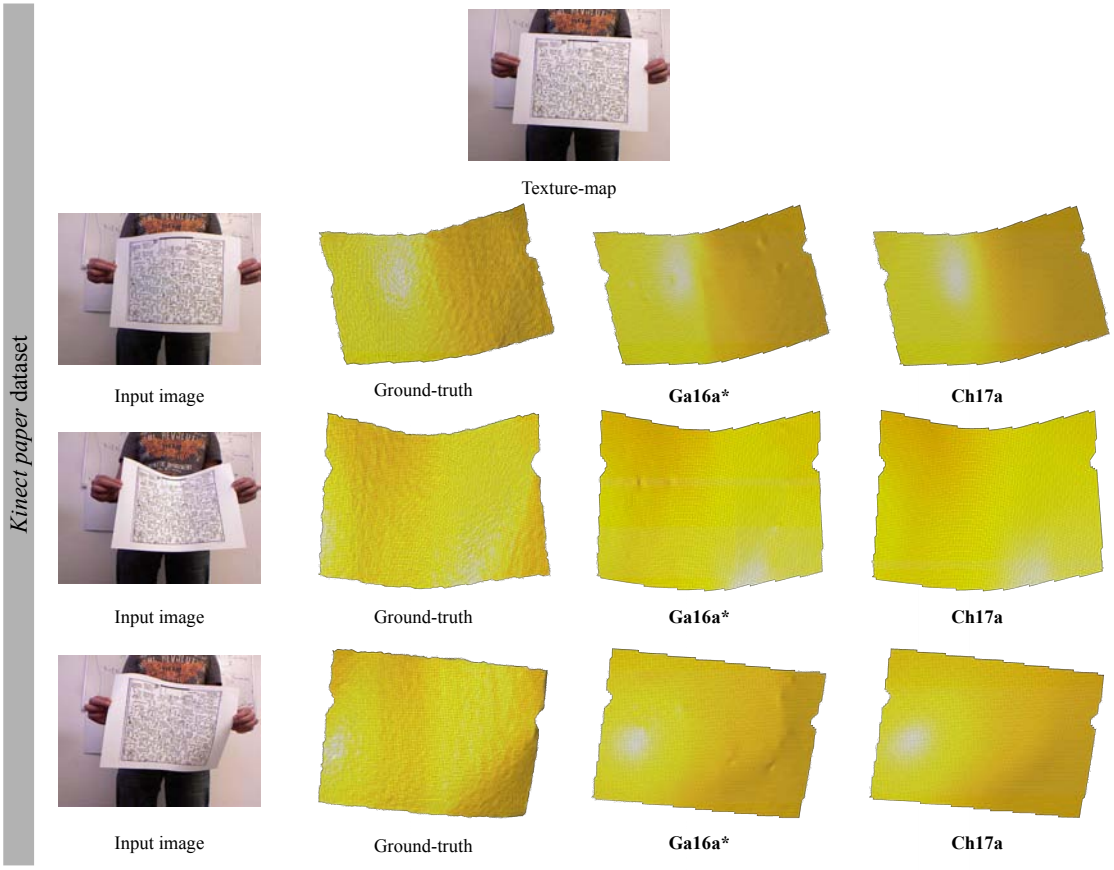


Figure 4.12: The smooth dataset *Kinect paper* and results visualizations for the *SfT-1* problem. We show the renderings for our method **Ga16a*** and [Chhatkuli et al., 2017] on three test input images. **Row n°1**: texture-map. **Row n°2**: frame n°36. **Row n°3**: frame n°101. **Row n°4**: frame n°115.

4.4.6 Limitations and Failure Modes

We discuss here the main limitations and the failure modes of our solution to *SfT-1*. One limitation is that this solution works only for well-textured surfaces. However, this allows us to have enough motion constraints at creased regions to infer the creases. Another limitation

[3]The dataset is available at https://cvlab.epfl.ch/data/dsr

is that the boundary contour constraint works only for surfaces with disc topology. Our method is also limited by the assumptions made to define the *SfT-1* instance, which we present in §4.2.2. An important assumption is that deformations are isometric. There are two main failure modes. The first is when the initial solution given by stage 1 of our refinement is not reliable. Figure 4.13 illustrates this failure mode. The initial solution suffers from a concave/convex ambiguity which our refinement does not succeed to avoid. The second failure mode is related to one limitation of our solution and is when the data constraints are not informative enough at creased regions to infer creases. This may occur in two cases: when the correspondences are not sufficiently dense at creased regions and when creases do not appear at surface boundaries.

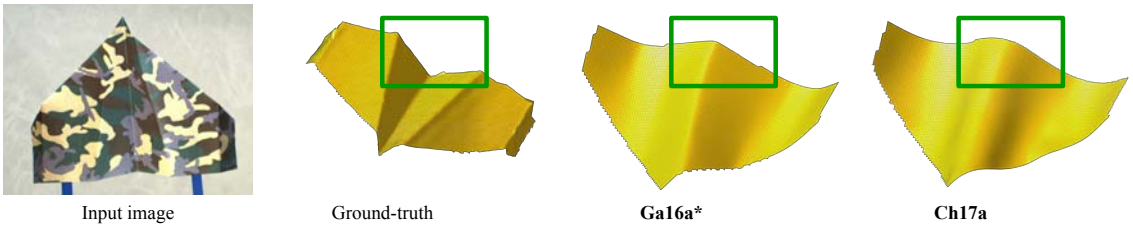


Figure 4.13: Illustration of one failure mode of our solution to *SfT-1*. We use the object of the *aeroplane paper* dataset. We show the renderings of the ground-truth shape and for our method **Ga16a*** and [Chhatkuli et al., 2017]. We use a green rectangle to indicate a clear evidence of ambiguous reconstruction.

4.5 Conclusion

We have developed a modeling and optimization framework for the *SfT-1* problem: it reconstructs smooth and creased 3D surfaces from a single image and a deformable 3D template. We implicitly model creases using a dense mesh-based surface representation with an associated robust bending energy constraint whose influence is governed by an M-estimator. Starting from 2D images, this M-estimator allows us to reconstruct piecewise-smooth 3D surfaces and so creases without any *a priori* about the crease location. We have seen that the redescending Tukey M-estimators cannot be used effectively with gradient-based optimization. It would be interesting to use such an estimator on a pre-refined solution with a ℓ_2 norm on the smoothing constraint (which is then assumed to be close to the global minimum) and see if this permit creases modeling. We have also observed that there is little difference in practice between the two common non-redescending M-estimators ($(\ell_1$-$\ell_2)$ and Huber with a correctly set hyper-parameter), and our results indicate significantly better performance compared to previous state-of-the-art methods. Using jointly motion and boundary contour constraints provides more accurate surface registration and 3D reconstructions. Disambiguating non-boundary image edges with statistical color models appears to be effective when the surface's color is significantly different to the background. We give some research axes for future work in chapter 7.

Chapter 5

Joint Photometric Calibration and 3D Reconstruction of Creasable Surfaces Using a Template and Shading Constraints

Summary

We address a limitation of prior SfT methods: handling poorly-textured surfaces under complex deformations. For this, we propose to combine shading and SfT. We refer to this as Shape-from-Template-and-Shading (SfTS). There exist a few previous works which also exploit shading. However, they either require all shading parameters (surface reflectance, illumination and camera responses) to be known a priori, or require a set of rigid views of the object before any deformation occurs. We propose an integrated solution without these requirements. It is the first approach for simultaneously computing an object's deformation, its reflectance properties, the scene illumination and camera response terms from deformed monocular images, which can be either videos or unorganized images. We evaluate with qualitative and quantitative results and show that it is also possible to accurately register and reconstruct poorly-textured surfaces with creases without any photometric calibration known a priori. This chapter is based on our peer reviewed paper [Gallardo et al., 2016b].

We start by briefly reminding the reader about the use of template with shading. We then define the SfTS instance we propose to solve in this chapter, following the same characterization of chapter 4. We instantiate the different models required to solve the problem instance and present our formulation.

5.1 Combining a Template with Shading

As shown in chapter 2, SfS and SfT are two approaches which have been intensively developed and which require very different modelings. However, some recent works propose to combine both approaches such as [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. The idea is to take advantage of the template which provides strong physical priors and the shading which densely constrains the reconstruction. An important challenge is the estimation of the surface reflectance function since this surface characteristic is rarely known *a priori*. We show in §2.5.3 that [Liu-Yin et al., 2016; Malti and Bartoli, 2014] handle this by assuming a rigid observation video, where the surface stays rigid. With this, they estimate the surface reflectance function while constructing the template, which limits significantly the applicability of these methods. In contrast, we propose to estimate the surface reflectance function from a template and a set of images where the surface deforms.

Chapter outline. In §5.2, we present our modeling of the problem and our shading-based energy cost function. In §5.3, we present our optimization framework. In §5.4, we validate our method with both high-accuracy ground-truth datasets and qualitative results. In §5.5, we provide our conclusions.

5.2 Problem Modeling

This section first gives the fundamental models used in the SfTS problem which we define in §1.2.6. We then instantiate SfTS with a concrete problem which is important to solve and remains general.

5.2.1 Fundamental Models of SfTS

In order to solve SfTS, four fundamental models are required: the *shadable template*, the *illumination* model, the *camera response* model and the *camera projection* model. The *camera projection* model has been defined in §4.2.2. The *illumination* model gives the amount and the spatial distribution of light. We denote the unknown illumination coefficients by $\mathbf{l}$. SfTS uses shading as a complementary visual cue, which requires surface reflectance to be modeled. We now define a surface reflectance model: it tells how the surface reflects the incident light, for a given scene illumination and a surface. SfTS contrasts classical SfT where surface reflectance is not required to be modeled. This motivates us to distinguish two types of template, which we define as follows.

Definition 7 (Non-shadable template)**.** *We define a non-shadable template as a textured 3D deformable model (figure 5.1(a)). Precisely, it models the 3D shape of the object in reference position and a deformation law such as isometry. Texture is modeled with a texture-map, which models the appearance of the object up to photometric variations caused by shading, scene illumination, inter-reflections and other phenomena.*

Definition 8 (Shadable template)**.** *We define a shadable template as a non-shadable template augmented with a known surface reflectance model (figure 5.1(b)).*

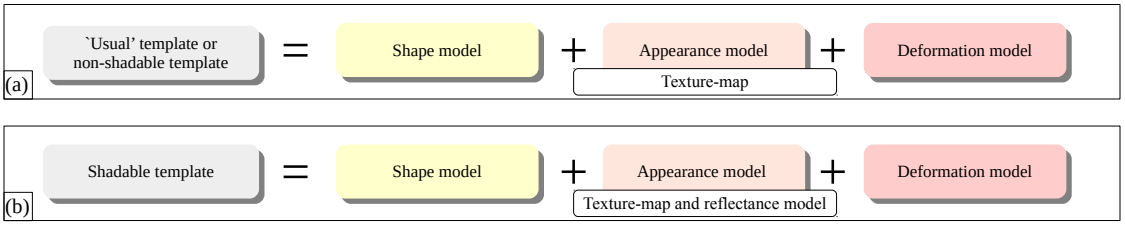


Figure 5.1: Templates definition. (a) Non-shadable template definition. (b) Shadable template definition. The non-shadable template corresponds to the usual template used in the SfT literature.

The *camera response* model maps the *irradiance image*, *i.e.* the image which stores the light striking the image plane at each pixel coordinate of the camera, to the *intensity image*, which is the output of the camera.

As SfTS relies on shading information, we need to model the shading equation. This equation predicts the intensity of a pixel given the models of illumination, surface shape, surface reflectance, camera projection and camera response. This starts with the *surface irradiance* which is the amount of light received by the surface. We use the function r to denote the surface irradiance for a normal vector $\mathbf{n}$ according to $\mathbf{l}$. This image contains the photometric variations caused by shading in particular. At any time t, we denote the irradiance image by R_t and the intensity image by L_t . We denote the camera response function by $g_t : \mathbb{R} \to \mathbb{R}$ which transforms the irradiance image R_t into the intensity image L_t.

5.2.2 *SfTS-1*: Instantiating SfTS for Unknown Photometric Parameters

We form the problem instance of *SfTS-1* using the eight components given in §2.1. An illustration of *SfTS-1* and its geometric and photometric setup are given respectively in figure 5.2 and figure 5.3. We now instantiate the problem components *(a)* to *(h)* and give the reasons of each component specification. Table 5.1 presents the instantiations of the fundamental models, given in §5.2.1, for *SfTS-1*.

Fundamental model	Known *a priori*	Fixed or time-varying	Instantiation
Shadable template's shape	✓	Fixed	High-resolution thin-shell 3D mesh
Shadable template's texture-map	✓	Fixed	2D image
Shadable template's reflectance	×	Fixed	Lambertian with piecewise-constant albedo
Shadable template's deformation	✓	Fixed	Isometric, crease-preserving parameterized by barycentric interpolation
Illumination	×	Fixed	Spherical harmonics (first and second-order) and attached to the camera
Camera projection	✓	Fixed	Perspective
Camera response	×	Time-varying	Linear general response

Table 5.1: Fundamental model instantiations in *SfTS-1*.

(a) Models. We use the same instantiations as the ones of the chapter 4 for the shape model, the texture-map and the deformation model of the shadable template, and the camera projection model. For the surface reflectance, we use the Lambertian model with piecewise-constant albedo. This is a reasonable assumption because this model gives a good approximation of many surfaces such as clothes, fabrics and cardboards. We do not assume that albedo is pre-segmented. For the illumination, we assume that it is unknown and constant over time, and fixed in camera coordinates. In practice, this can be assumed if we have a camera-light rig setup such as an endoscope or camera with flash, or a non-rig where the light and a camera are not physically connected but do not move relative to each other during image acquisition. We investigate two illumination models: the first-order and second-order spherical harmonic model, which are very common in SfS with respectively 4 and 9 parameters. For the camera response, we assume that it is linear, which is a valid assumption for many CCD cameras. We assume that it can change over time in order to handle changes due to camera shutter speed and/or exposure. *(b) Exploited visual cues.* The visual cues we use are motion, boundary contour and shading. The motivations of using motion and boundary contour cues are given in §4.2.2. *(c) Number of required images.* A set of at least 4 images is required. This is required by our initialization algorithm of the photometric parameters (illumination, camera response and surface reflectance). Details of this algorithm are given §5.3.2. *(d) Expected types of deformations.* As in §4.2.2, we assume quasi-isometric and smooth or piecewise-smooth deformations, and no tearing. *(e) Scene geometry.* As in §4.2.2, we assume no self or external occlusions, but there can be background clutter. *(f) Requirement for putative correspondences.* We assume to know *a priori* a set of putative 2D correspondences from the texture-map of the template to each input image. We assume there may be a small proportion of mismatches *e.g.* $< 20\%$. *(g) Surface texture characteristics.* We consider generic surfaces which present textured and poorly-textured regions. *(h) Known and unknown model parameters.* The unknowns are the vertices of the deformed template in the camera coordinates of each input image, the camera response terms for each input image, the illumination vector and the albedo-map (segments and values). All other model

parameters are assumed known.

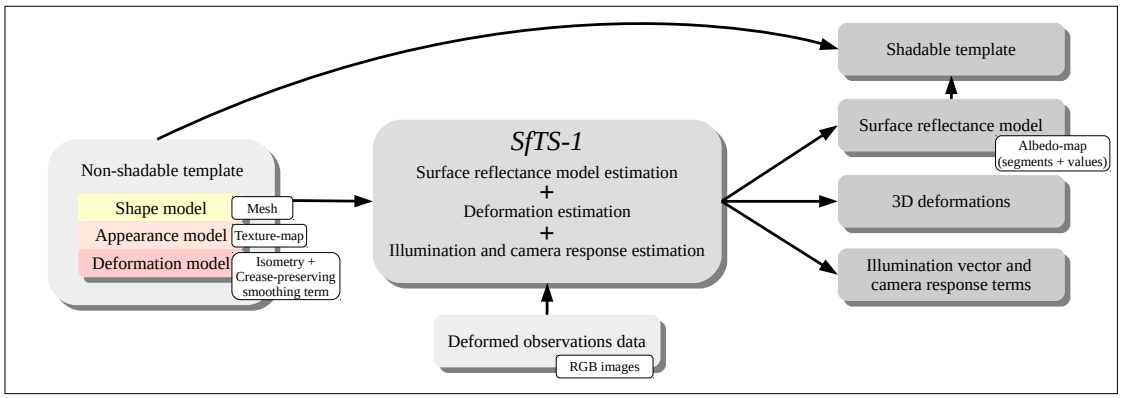


Figure 5.2: Illustration of the *SfTS-1* problem. Unlike [Liu-Yin et al., 2016; Malti and Bartoli, 2014], this problem does not require a set of rigid observations. This makes *SfTS-1* more difficult to solve, yet more applicable in unconstrained settings.

5.2.3 Shadable Template Modeling

We now give the specific models used to instantiate the shadable template (*shape model*, *deformation model* and *appearance model*). We use the same *shape* and *deformation* models as in §4.2.3, which we now recall.

The *shape model* is a high-resolution thin-shell 3D mesh model in a known reference position consisting of a set of M 3D vertices $\mathcal{Y} \triangleq \{\mathbf{y}_1, ..., \mathbf{y}_M\} \in \mathbb{R}^{3\times M}$ and F faces $\mathcal{F} \triangleq \{f_1, ..., f_F\}, f_k \in [1, M]^3$. Similarly to *SfT-1*, M is on the order of 10^4 in our experiments. We denote the set of mesh edges as $E \in [1, M]^{2\times N_E}$, where N_E is the number of edges and where the edges are fixed over time since we assume no tearing.

The *appearance model* includes a texture-map $\mathcal{T} : \mathbb{R}^2 \rightarrow \mathbb{R}^+$, defined as in §4.2.3, and a surface reflectance model. We use a Lambertian model and we handle model deviations due to *e.g.* specular reflections with a robust data constraint (see §5.2.6). We define an *albedo-map* $A(\mathbf{u}) : \Omega_\mathcal{T} \rightarrow \mathbb{R}^+$ as the mapping from a surface point $\mathbf{u}$ to albedo. Estimating A is part of the problem. We assume that A is constant over time and piecewise-constant over the surface. The piecewise-constant assumption is used to reduce ambiguity between smooth intensity variation caused by albedo variation versus surface gradient variation, and reduce the number of unknown variables. A is therefore given by $A(\mathbf{u}) : \Omega_\mathcal{T} \rightarrow \mathcal{A} = \{\alpha_1, ..., \alpha_K\}$ where α_k denotes the k^{th} unknown albedo value and where K is also unknown.

The *deformation model* transforms each vertex to 3D camera coordinates: we model the position of each vertex $i \in \{1 \dots M\}$ in camera coordinates by $\mathbf{v}_t^i \in \mathbb{R}^3$, where t denotes time. We transform a point $\mathbf{u} \in \Omega_\mathcal{T}$ to camera coordinates according to $\mathcal{V}_t$ with the same barycentric interpolation φ detailed in §4.2.3 and $n(\mathbf{u}; \mathcal{V}_t) : \mathbb{R}^{3\times M} \rightarrow \mathbb{S}_3$ to represent its unit surface normal. We also assume isometry and crease-preserving smoothness and impose them through the cost function which we define in §5.2.6.

5.2.4 Illumination and Camera Modeling

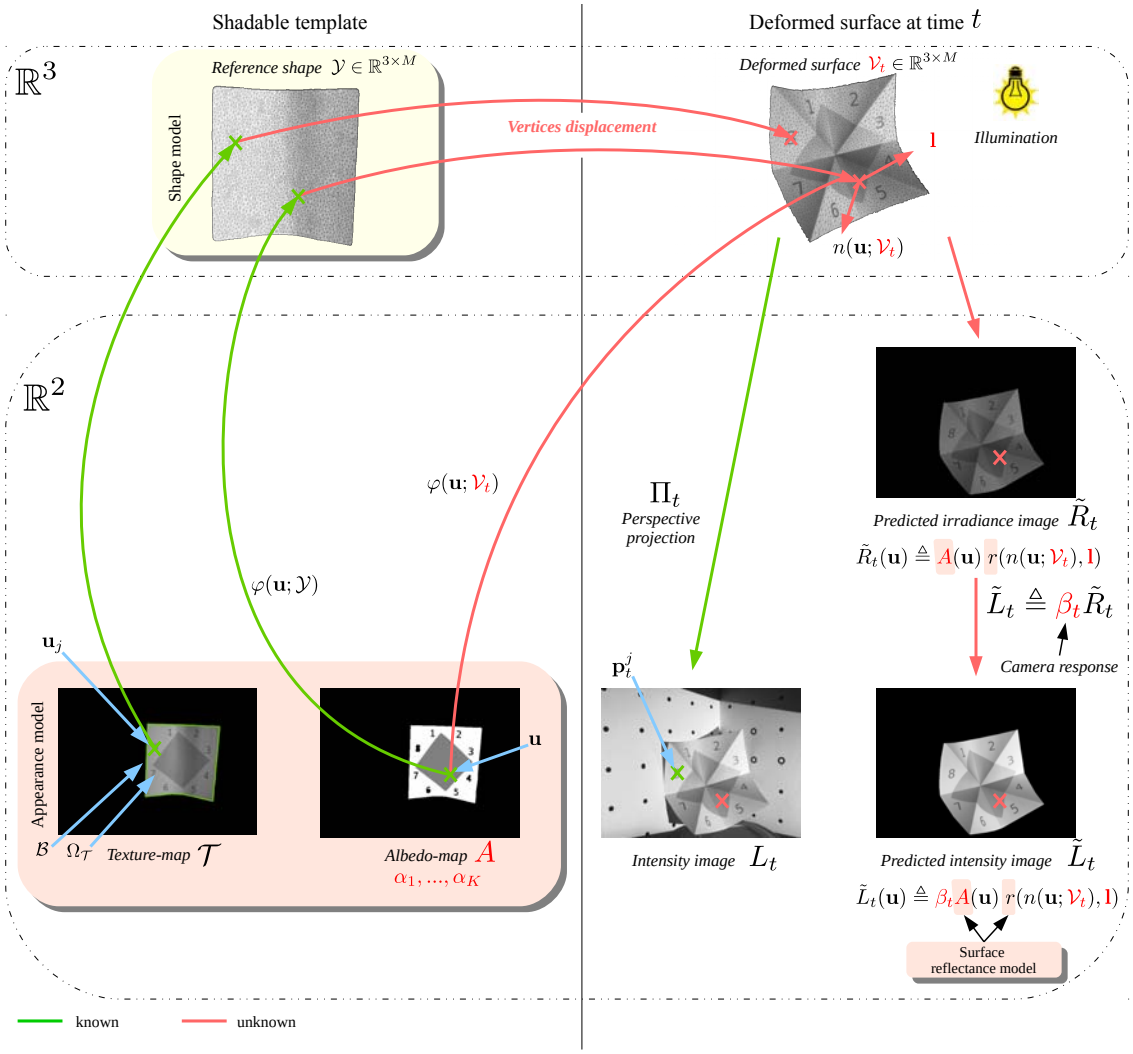


Figure 5.3: Geometric and photometric setup of the *SfTS-1* problem. This diagram illustrates the motion constraint explained in §5.2.6.2 with the matched points $(\mathbf{u}_j, \mathbf{p}_t^j)$), and the shading constraint explained in §5.2.6.1 with the point $\mathbf{u} \in \Omega_\mathcal{T}$. We use red for the unknowns of our problem. We highlight in light red the components of the appearance model (texture-map, albedo-map and surface reflectance model).

We refer to figure 5.3 for the modeling of the image formation process, which is required to use shading. When the first-order spherical harmonic model is used, the illumination model is a combination of a light source at the infinity and an ambient term. Note that, as $\mathbf{l}$ is represented by spherical harmonics, the surface irradiance r is linear in $\mathbf{l}$. As we assume the Lambertian reflectance model, we have $r(\mathbf{n},\mathbf{l}) = \mathbf{n}^\top\mathbf{l}$. As we assume g_t is linear, we have $L_t = \beta_t R_t$ with $\beta_t \in \mathbb{R}^+$.

5.2.5 Inputs and Outputs

We now give our inputs. *(i)* a batch of N input RGB images $\{I_t\}_{t\in[1,N]}$, $I_t : \mathbb{R}^2 \rightarrow \{0, 255\}^3$ of a deforming object and the corresponding intensity images denoted $\{L_t\}_{t\in[1,N]}$, $L_t : \mathbb{R}^2 \rightarrow \mathbb{R}^+$. In practice, the intensity image L_t is obtained by selecting the second component of the projection of the input RGB image I_t in the CIE XYZ color space. *(ii)* a non-shadable template of the object, defined using §5.2.3. *(iii)* the camera intrinsics of all perspective projection functions Π_t. *(iv)* N sets $\mathcal{S}_t$ of matched putative 2D correspondences from the texture-map $\Omega_\mathcal{T}$ to each input image I_t. We denote it by $\mathcal{S}_t = \{(\mathbf{u}_j, \mathbf{p}_t^j)\}$ where $\mathbf{u}_j$ denotes the correspondence position in $\Omega_\mathcal{T}$ and $\mathbf{p}_t^j$ denotes the correspondence position in the t^{th} input image I_t. The number of correspondences for each image t is denoted by s_t. In our experiments, these are correct. Details for how this is done for our experimental datasets are given in §5.4.3. We did not evaluate quantitatively how robust to mismatches is our method, but it has the potential to handle them as we explain in §5.2.6.

The outputs of our solution to *SfTS-1* are: *(i)* the 3D shape of the deformed template, $\mathcal{V}_t \triangleq \left\{\mathbf{v}_t^1, ..., \mathbf{v}_t^M\right\} \in \mathbb{R}^{3\times M}$, corresponding to image I_t, *(ii)* the segmented albedo-map A with its K segments and values $\{\alpha_1, ..., \alpha_K\}$, *(iii)* the illumination vector $\mathbf{l}$ and *(iv)* the camera response β_t corresponding to image I_t.

5.2.6 Problem Modeling with an Integrated Cost Function

The cost function combines the shading constraint with the motion and boundary contour constraints and *physical deformation priors* (quasi-isometry and smoothing constraints). The cost function C for a single input image I_t is denoted by:

$$\begin{aligned} &C(\mathcal{V}_t, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \beta_t) \triangleq C_{shade}(\mathcal{V}_t, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \beta_t) + \\ &\lambda_{motion} C_{motion}(\mathcal{V}_t) + \lambda_{contour} C_{contour}(\mathcal{V}_t) + \lambda_{iso} C_{iso}(\mathcal{V}_t) + \lambda_{smooth} C_{smooth}(\mathcal{V}_t). \end{aligned} \tag{5.1}$$

The terms C_{shade}, C_{motion} and $C_{contour}$ are shading, motion and boundary contour data constraints respectively. The terms C_{smooth} and C_{iso} are physical deformation prior constraints. The terms λ_{motion}, $\lambda_{contour}$, λ_{iso} and λ_{smooth} are positive weights and are the method's tuning parameters. The constraint weights λ_{motion}, $\lambda_{contour}$, λ_{iso} and λ_{smooth} are normalized as in chapter 4. The shading weight λ_{shade} is normalized with the number of shading points. We do not assume temporal dependency in β_t and $\mathcal{V}_t$. This allows us to handle both unorganized image sets and images from video streams. As we consider a batch set of N input images we define the complete cost function C_{total} as the sum of costs from each image:

$$C_{total}(\{\mathcal{V}_t\}, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \{\beta_t\}) \triangleq \sum_{t=1}^{N} C(\mathcal{V}_t, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \beta_t). \tag{5.2}$$

To solve *SfTS-1*, we solve the following minimization problem:

$$\min_{\{\mathcal{V}_t\}, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \{\beta_t\}} C_{total}(\{\mathcal{V}_t\}, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \{\beta_t\}). \tag{5.3}$$

5.2.6.1 The Shading Constraint

The shading constraint robustly encodes the Lambertian relationship between surface, albedo, surface irradiance, pixel intensity and camera response. We use the piecewise-constant albedo model given in §5.2.3, and we decide to not optimize all albedo segments. There are two reasons. First, there is a potential difficulty with using shading at textured regions. This comes from the fact that the mis-registration errors at textured regions may imply mis-registration of the albedo-map over the surface. This then may lead to large errors in albedo estimation and surface reconstruction because of the linear dependency of the shading constraint in albedo values. Second, textured regions are very informative for motion constraints. The shading constraint is then less useful or even not useful at textured regions. Therefore, we propose to not use shading in textured regions. For this, we use the fact that textured regions can be detected as small albedo segments. We propose to exclude from the optimization albedo segments which are smaller in area than the threshold T_A (in % of the number of pixels contained in the image). In practice, we found that using $T_A = 0.022\%$ allows to reduce reconstruction errors at textured regions. We evaluate the shading constraint at each pixel of albedo segments larger than T_A, which gives:

$$C_{shade}\left(\mathcal{V}_t, \alpha_1, \ldots, \alpha_K, \mathbf{l}, \beta_t\right) \triangleq \sum_{\mathbf{u}\in\Omega_{\mathcal{T}}} \rho\Big(\beta_t\, A(\mathbf{u})\, r\left(n(\mathbf{u};\mathcal{V}_t), \mathbf{l}\right) - L_t\big(\Pi_t \circ \varphi(\mathbf{u};\mathcal{V}_t)\big) \Big). \quad (5.4)$$

The function $\rho : \mathbb{R} \to \mathbb{R}$ is an M-estimator which is used to enforce similarity between the modeled and measured intensity, while also allowing for some points to violate the model (caused by specular reflection, small shadows and other unmodeled factors). When the residual of such points is not too high, we find that a robust estimator based on an M-estimator is very effective to handle them. We also use M-estimators in some of the other cost function terms, and defer the precise choice to the implementation section §5.4.4.

In order to reduce computation time, texture-map pixels from Ω_A are downsampled by a factor of $X\%$. In practice, we found that using $X = 50\%$ gives good reconstructions.

5.2.6.2 Other Constraints

The motion constraint. We specify the motion constraint of §4.2.5.1 for each input image $t \in [1, N]$. We recall that the set $\mathcal{S}_t = \{(\mathbf{u}_j, \mathbf{p}_t^j)\}$ holds the putative correspondences between the texture-map of the template and the t^{th} input image. $\mathbf{u}_j$ denotes the position of the j^{th} correspondence in the texture-map and $\mathbf{p}_t^j$ denotes the position of the j^{th} correspondence in the t^{th} input image. We also assume there may be a small fraction of mismatches due to ambiguous texture regions or other factors, which we handle with an M-estimator. The motion constraint robustly encourages the texture-map points to project to their matches and is given by:

$$C_{motion}(\mathcal{V}_t) \triangleq \sum_{(\mathbf{u}_j, \mathbf{p}_t^j)\in\mathcal{S}_t} \rho\left(\left\|\Pi_t\left(\varphi(\mathbf{u}_j;\mathcal{V}_t)\right) - \mathbf{p}_t^j\right\|\right). \quad (5.5)$$

The boundary contour constraint. This constraint is based on the boundary contour constraint in §4.2.5.2. This constraint works for surfaces with disc topology. It encourages the surface's boundary contour to lie close to image edges. We discretize the boundary of $\Omega_{\mathcal{T}}$ to obtain a set of boundary pixels $\mathcal{B} \triangleq \{\mathbf{u}_{k\in[1,N_{\mathcal{B}}]}\}$, with $N_{\mathcal{B}}$ the number of boundary pixels. We use the enhanced implementation given in §4.2.5.2 to construct the enhanced boundariness map for each input image I_t: we denote it as B_t. The boundary contour constraint is defined as follows:

$$C_{contour}(\mathcal{V}_t) \triangleq \frac{1}{|N_{\mathcal{B}}|} \sum_{\mathbf{u}_k \in \mathcal{B}} \rho\Big(B_t\big(\Pi_t \circ \varphi(\mathbf{u}_k; \mathcal{V}_t)\big) \Big), \tag{5.6}$$

where a robust estimator ρ is used to handle the fact that sometimes there may be little contrast difference between the surface and background structures.

The deformation priors. For the quasi-isometric constraint (C_{iso}) and the crease-preserving smoothing constraint (C_{smooth}), we respectively use the constraints of §4.2.5.4 and §4.2.5.3:

$$C_{iso}(\mathcal{V}_t) \triangleq \frac{1}{|E|} \sum_{(i,j)\in E} \Big(1 - \|\mathbf{y}^i - \mathbf{y}^j\|_2^{-2} \, \|\mathbf{v}_t^i - \mathbf{v}_t^j\|_2^2\Big)^2, \tag{5.7}$$

$$C_{smooth}(\mathcal{V}_t) \triangleq \frac{1}{|\Omega_{\mathcal{T}}|} \sum_{\mathbf{u}_j \in \Omega_{\mathcal{T}}} \rho\left(\frac{\partial^2 \varphi}{\partial \mathbf{u}^2}(\mathbf{u}_j; \mathcal{V}_t)\right). \tag{5.8}$$

5.3 Optimization Strategy

5.3.1 Overview

The cost function (5.1) is more challenging to optimize than the one in chapter 4 because the unknowns (deformation, albedos, illumination and camera responses) are all linked through the shading constraint (5.4) and because shading constraints are generally much more non-convex than motion and boundary contour constraints. This means that we cannot solve the problem by considering each image solution in isolation. We present a solution using a fast cascaded initialization strategy followed by iterative gradient-based numerical optimization. A schematic of the whole process is illustrated in figure 5.4. Note that the problem always has a global photometric scale ambiguity between albedos, camera response and illumination strength because of their trilinear product in C_{shade}. This can be fixed by arbitrarily setting $\beta_1 = \alpha_1 = 1$. We can do this because our aim is to recover the geometry of the surfaces and not to recover the absolute scale of the illumination, which is, in this problem, equivalent to ignoring the absolute scale of camera response and albedo.

5.3.2 Cascaded Initialization

We propose a cascaded initialization strategy shown in figure 5.4. This works by successively introducing the motion then boundary contour constraints to obtain an initial estimate for the deformation parameters for each input image. Next the illumination and camera response

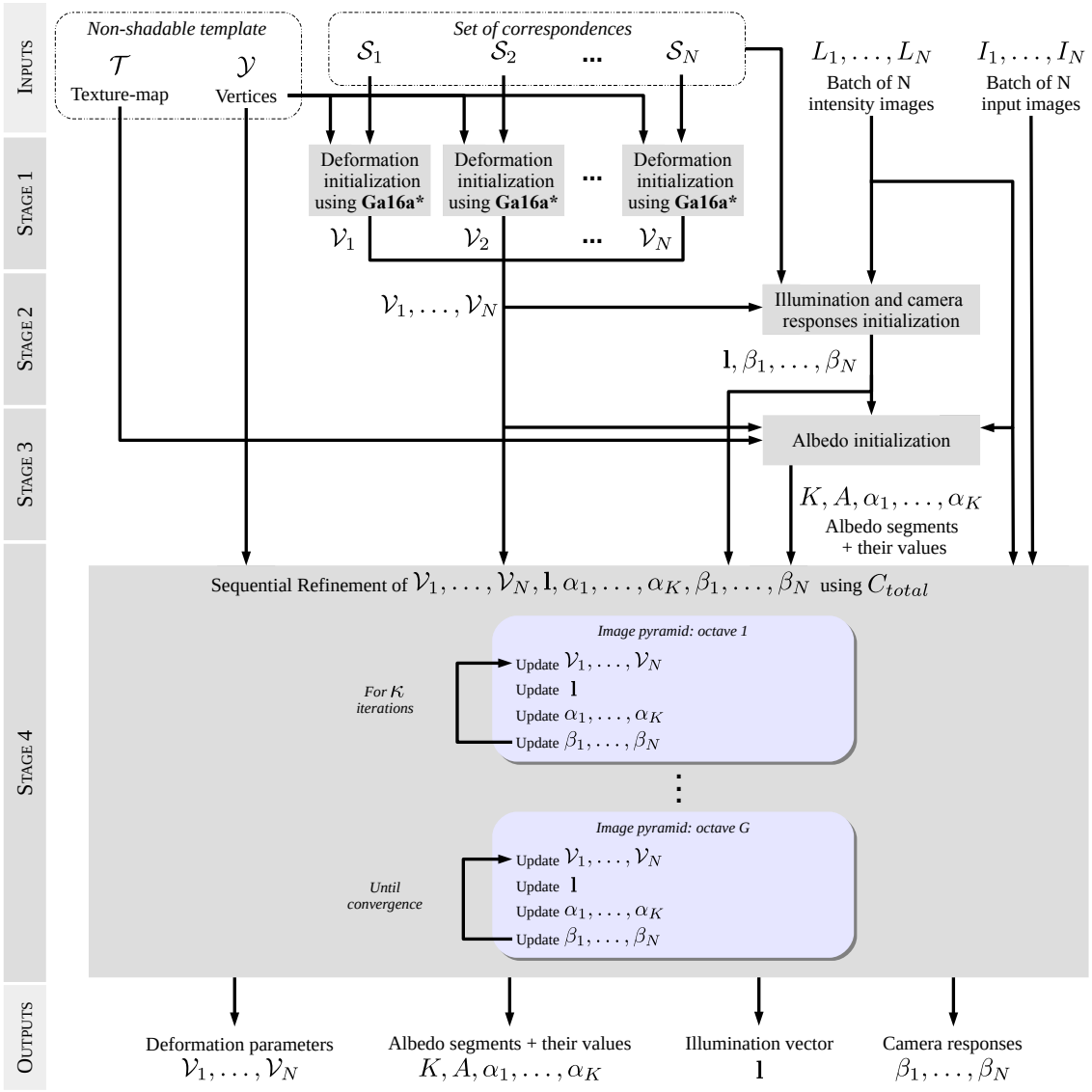


Figure 5.4: Schematic of our proposed solution to solve *SfTS-1*.

parameters are estimated through a robust model-sampling based approach. This leverages the fact that at smooth surface regions, deformation can usually be estimated well at point correspondences without needing any boundary contour or shading constraints. Given deformations at the correspondences, we initialize the photometric parameters by inverting the shading equation using pixel intensities and the estimated surface normals *around each correspondence*. At the final stage the albedo-map is initialized by first segmenting the intensity texture-map through an intrinsic image decomposition, and then estimating the albedo of each segment by inverting the shading equation using the deformation, illumination and camera response estimates.

5.3.2.1 Stage 1 - Deformation Initialization

This stage gives an initial solution for each input image by solving the *SfT-1* problem independently for each input image. For this, it uses our solution presented in chapter 4) We use the two strategies given in §4.3.3.2 to improve the convergence of the refinement of the chapter 4. For this, first we refine only with the motion constraint (5.5) as image data constraint, then we add the boundary contour constraint (5.6). Second, we construct the boundariness map B_t (for the boundary contour constraint (5.6)) using an image pyramid (we found that three octaves provide good convergence), and sequentially optimize with each pyramid level until convergence.

5.3.2.2 Stage 2 - Illumination and Camera Responses Initialization

Algorithm 1 gives the procedure to initialize the illumination vector and the camera responses. Using the initial deformation estimates, we first estimate the surface normal for each correspondence in $\mathcal{S}_t$. These normals are given in 3D camera coordinates and denoted by $\mathbf{n}_t^j$, with t being the image index and j being the point index. For each point $\mathbf{u}_j$ in the set of 2D correspondences $\mathcal{S}_t$ we also model the average albedo within a small square window (in our experiments we use 11×11 windows), which we denote by $a_j \in \mathbb{R}$. Recall that because we do not yet know the albedo-map, we do not yet know a_j. We estimate the average albedo a_j on a small square window in order to be robust to image noise. We use $L_t^j \in \mathbb{R}$ to denote the average pixel intensity within the local window at the j^{th} correspondence in the t^{th} image. Our goal is to estimate the illumination vector $\mathbf{l}$, the camera response terms $\{\beta_t\}$ and the albedo values $\{a_j\}$ by inverting the Lambertian shading equation $\beta_t\, a_j\, r(\mathbf{n}_t^j; \mathbf{l}) = L_t^j$. This is a hard non-convex inverse problem. It can be greatly simplified if $\{\beta_t\}$ is known, because with spherical harmonic models r is linear in $\mathbf{l}$, so it becomes linear by dividing through by a_j. In some cases $\{\beta_t\}$ can be estimated from Exchangeable Image File Format (EXIF) tags. In other cases, if the image background does not change between images we can approximate $\{\beta_t\}$ by taking the ratio of pixel intensities in the background between different images. When neither is possible, we can make a different assumption: that the camera response terms are roughly similar in a few of the images, but we do not know which ones *a priori*. We then can tackle the problem with random sample consensus. To do this we require a low dimensional illumination model, so we use first-order spherical harmonics (with 4 dimensions). We will first describe how the unknown parameters are computed from a minimal sample of a point with 4 correspondences. We will then describe how this fits into a RANSAC framework, with pseudo-code given in algorithm 1.

We consider a single surface point j matched in 4 images where β_t is assumed constant for the 4 images. We can solve $\mathbf{l}$ up to scale with an exact linear system. Recall that this scale is never recoverable but it is not actually important to us. Given $\mathbf{l}$, each correspondence is then used to estimate β_t by taking intensity ratios L_t^j / L_1^j for the image t and the correspondence j. Recall that $\beta_1 = 1$ (to fix the photometric scale ambiguity), so after rearranging we have $\beta_t = L_t^j r(\mathbf{n}_1^j; \mathbf{l}) / L_1^j r(\mathbf{n}_t^j; \mathbf{l})$. Therefore each correspondence produces a value for β_t. We

Inputs:
the surface normals $\{\mathbf{n}_t^j\}$
the pixel intensity $\{L_t^j\}$
Outputs:
illumination vector $\mathbf{l} \in \mathbb{R}^{4\times 1}$ (first-order spherical harmonics)
camera responses $\{\beta_t\}_{t\in[1,N]}$
Index convention:
$j \in [1, S]$, where S is the number of correspondence in the texture-map
$t \in [1, N]$, where N is the number of input images
Notation convention:
$L \in \mathbb{R}^{N\times S}$ is the matrix of average pixel intensity within a local window at each correspondence and for all input images
$\mathcal{N} \in \mathbb{R}^{N\times S\times 3}$ is the matrix of normals at each correspondence for all input images
$\backslash$ is matrix division
./ is element-wise matrix division

$iteration \leftarrow 0$
$nbPoints \leftarrow S\,N$
$maxNbInliers \leftarrow 0$
for $(t, j) \in [1, N] \times [1, S]$, $\bar{\mathbf{n}}_t^j \leftarrow \left[\mathbf{n}_t^{j\top},\ 1\right]^\top$
for $(t, j) \in [1, N] \times [1, S]$, $\mathcal{N}(t, j, :) \leftarrow \bar{\mathbf{n}}_t^j$
for $(t, j) \in [1, N] \times [1, S]$, $L(t, j) \leftarrow L_t^j$
while $(iteration < k \quad \text{or} \quad maxNbInliers \geq \mathcal{C}o \times nbPoints)$
  Select randomly one point $j \in [1, S]$
  Select randomly 4 input images $(t_1, t_2, t_3, t_4) \in [1, N]^4$ where point j has a correspondence
  $\mathbf{A} \leftarrow \left[\bar{\mathbf{n}}_{t_1}^j, \bar{\mathbf{n}}_{t_2}^j, \bar{\mathbf{n}}_{t_3}^j, \bar{\mathbf{n}}_{t_4}^j\right]^\top$ % $\mathbf{A} \in \mathbb{R}^{4\times 4}$
  $\mathbf{b} \leftarrow \left[L_{t_1}^j, L_{t_2}^j, L_{t_3}^j, L_{t_4}^j\right]^\top$ % $\mathbf{b} \in \mathbb{R}^{1\times 4}$
  $\tilde{\mathbf{l}} \leftarrow \mathbf{A}\backslash\mathbf{b}$
  $\gamma \leftarrow L./\left(\mathcal{N}\,\tilde{\mathbf{l}}^\top\right)$ % $\gamma \in \mathbb{R}^{N\times S}$
  for $t \in [1, N]$, $\tilde{\beta}_t \leftarrow median\left(\gamma(t, :)./\gamma(1, :)\right)$
  for $j \in [1, S]$, $\tilde{a}_j \leftarrow median\left(\gamma(:, j)./\left[\tilde{\beta}_1, ..., \tilde{\beta}_N\right]^\top\right)$
  for $(t, j) \in [1, N] \times [1, S]$, $error(t, j) \leftarrow \left|\beta_t \tilde{a}_j \bar{\mathbf{n}}_t^j\,\tilde{\mathbf{l}}^\top - L(t, j)\right|$
  $nbInliers \leftarrow sum(error \leq \tau\, 1_{N\times S})$
  if $nbInliers > maxNbInliers$
    $maxNbInliers \leftarrow nbInliers$
    $\mathbf{l} \leftarrow \tilde{\mathbf{l}}$
    $\{\beta_t\}_{t\in[1,N]} \leftarrow \{\tilde{\beta}_t\}_{t\in[1,N]}$
  end
  $iteration \leftarrow iteration + 1$
end

Algorithm 1: RANSAC-based robust estimation of illumination and camera response using 2D correspondences on a deforming surface. The default value of hyperparameters k, $\mathcal{C}o$ and τ are given in table C.2 of appendix C.

compute β_t robustly by taking the median across all correspondences. We then compute a_j as follows. Each image provides an estimate with $a_j = L_t^j/(\beta_t\, r(\mathbf{n}_t^j; \mathbf{l}))$, so a robust estimate of a_j is computed by taking the median across the images.

The second component of RANSAC is to validate the parameters through consensus. We do this by computing the number of point correspondences where the shading equation is satisfied up to noise: $|\beta_t\, a_j\, r(\mathbf{n}_t^j; \mathbf{l}) - L_t^j| \leq \tau$. This requires an acceptance tolerance τ and in all experiments we use $\tau = 0.04$.

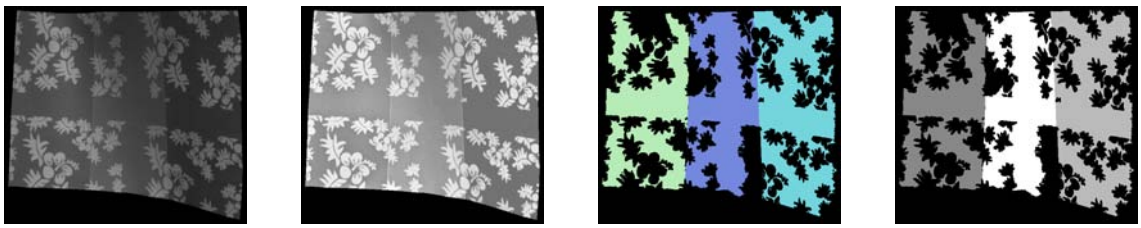
The third component of RANSAC is random selection. We do this by first randomly selecting a point from $\mathcal{S}_t$ (with uniform probability), then selecting 4 images where it has a correspondence (with uniform probability). We terminate RANSAC if either $\mathcal{C}o = 50\%$ consensus is reached or an iteration limit $k = 20,000$ iterations had passed. On a standard Intel i7 desktop workstation $k = 20,000$ takes a few seconds to reach using an unoptimized Matlab code. The choice of k depends on the likelihood of drawing 4 images with approximately the same camera response, which is difficult to know *a priori*.

5.3.2.3 Stage 3 - Albedos Initialization

Recall that we assume the surface albedos are piecewise-constant, as written in table 5.1. We illustrate the albedo-map initialization process in figure 5.5. We first perform an intensity-based segmentation of the template's texture-map using an output image from the intrinsic image decomposition method [Bell et al., 2014]. Precisely, we cluster the output image which [Bell et al., 2014] names the 'reflectance image' and which we show in figure 5.5 (b). We do not use the reflectance image as the albedo-map for the following reason. The reflectance image from [Bell et al., 2014] is intended to be similar to the albedo-map, but in practice it does not work well consistently, as figure 5.5 (b) shows. The reason is that estimating an albedo-map from a single image is severely ill-posed. We cluster the reflectance image into piecewise-constant albedos by using the Mean Shift algorithm [Fukunaga and Hostetler, 1975] with a low cluster tolerance (we use a default of 10). An illustration of a clustered image is given in figure 5.5 (c). The albedo-map initialization process is designed to be an oversegmentation, and within each segment we assume the albedo is constant. This oversegmentation allows neighboring segments to share the same albedo. We aim for an oversegmentation because our method is not designed to recover from an undersegmentation. Even if oversegmentation requires more unknowns, undersegmentation is a more difficult problem since it may strongly impact the estimation of surface orientation and illumination and requires then an automatic process to re-segment the albedo-map when needed. The last step of the clustering is the thresholding of the pixels number of each region to remove the textured regions. In our experiments, we found that a default value of $T_A = 0.022\%$ (of the number of pixels contained in the image) allows us to remove most of textured regions. The black holes visible in the surface in figure 5.5 (c) correspond to these textured regions. If there are K segments, then the albedo set $\{\alpha_1, \ldots \alpha_K\}$ has size K.

In the next step, we estimate for each albedo segment its corresponding albedo. For this,

we invert the shading equation using the initial estimates of $\mathcal{V}_t$, $\mathbf{l}$ and β_t. This can be done in parallel for each segment. Figure 5.5 (d) shows an illustration of an initial albedo-map obtained at the end of the albedo initialization.



(a) (b) (c) (d)

Figure 5.5: Running example showing albedo initialization. (a) texture-map, (b) reflectance image obtained from [Bell et al., 2014], (c) clustered image, (d) initial albedo-map (albedo segments and their initial values).

5.3.3 Refinement

Having initialized, we use the refinement approach extending chapter 4: we refine the cost function (5.1) using GN iterations with backtracking line-search. Before this refinement, we can keep using the first-order spherical harmonic model (required in the stage 2) or switch to the second-order model which allows to model more accurately more complex illumination. We present results for both models. The deformation parameters $\mathcal{V}_t$ are all linked in the optimization through the shading constraint. This is a difference from chapter 4. Thus the optimization problem size grows with the number of views. For a small number of views, *e.g.* up to ten views, it is possible to solve the normal equations directly with sparse Cholesky decomposition (typically done in under a few minutes in Matlab on a desktop PC). However, this does not scale well to more images. Iterative methods such as the conjugate gradient are therefore needed. To improve convergence, we also use an image pyramid (three octaves as for the boundary contour constraint) of blurred intensity images.

5.4 Experimental Validation

5.4.1 Methods Compared

We compared the accuracy of our method with the four competitive SfT methods used in chapter 4 [Bartoli et al., 2015; Chhatkuli et al., 2017; Ngo et al., 2016; Salzmann and Fua, 2009], which we denote respectively by **Ba15a**, **Ch17a**, **Sa09a** and **Ng16a**. We briefly describe each method in table 5.2. We evaluated our method in five cases to fairly assess the benefits of using shading. We recall that the star ***** stands for the proposed methods. The first two use a first-order spherical harmonic illumination model which is either calibrated *a priori* or uncalibrated, denoted by **Ga16b_S4K*** and **Ga16b_S4U*** respectively. The second two use a second-order spherical harmonic illumination model that is

either calibrated *a priori* or uncalibrated, denoted by **Ga16b_S9K*** and **Ga16b_S9U***. For simplicity, we refer by **Ga16b*** to all four methods given in this chapter (**Ga16b_S4K*** **Ga16b_S4U***, **Ga16b_S9K*** and **Ga16b_S9U***). The fifth is when shading is omitted by assigning $\lambda_{shade} = 0$: it corresponds to the method described in chapter 4 and is denoted by **Ga16a***.

Acronym	Source	Shape model	Constraints	Principle
Ga16a*	Proposed	Mesh	M+C+I+Ss	Initialization (**Ch16a**) + Constraints minimization
Ga16b_S4K*	Proposed	Mesh	M+C+Sh+I+Ss	Initialization (**Ga16a***) + Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **known** and modeled by **first-order** spherical harmonics
Ga16b_S4U*	Proposed	Mesh	M+C+Sh+I+Ss	Initialization (**Ga16a***) + **Illumination**/Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **unknown** and modeled by **first-order** spherical harmonics
Ga16b_S9K*	Proposed	Mesh	M+C+Sh+I+Ss	Initialization (**Ga16a***) + Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **known** and modeled by **second-order** spherical harmonics
Ga16b_S9U*	Proposed	Mesh	M+C+Sh+I+Ss	Initialization (**Ga16a***) + **Illumination**/Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **unknown** and modeled by **second-order** spherical harmonics

M: Motion, C: boundary Contour, Sh: Shading, I: Isometry, Ss: Surface smoothing

Table 5.2: List of the different versions of our method. We give their specific components of modeling and solving.

5.4.2 Ground-Truth Acquisition

Few datasets of deforming and poorly-textured surfaces exist [Ngo et al., 2015; Salzmann et al., 2008]. However, [Salzmann et al., 2008] has no ground-truth and [Ngo et al., 2015] does not contain creased surfaces. Furthermore, since it uses Kinect to acquire the ground-truth, accuracy can only be measured to within 1-2 cm. Thus we propose here two new datasets to evaluate our method and future methods, with highly accurate ground-truth. We constructed the datasets using the structured light system [David 3D Scanner, 2014], which we present in §4.4.2, with the same PointGrey RGB camera [Point Grey]. We set the camera's response to be linear using the software of [David 3D Scanner, 2014]. As specified in table 5.1, the illumination and the camera do not move relative to each other whilst the images are being acquired. The illumination vector is then the same in the 3D camera coordinates of each input image. The illumination is calibrated using multiple images of a MacBeth chart, using surface normals computed by fitting corresponding planes to the depth-maps.

5.4.3 Datasets

We tested our method with five real-world datasets which mostly respect the Lambertian assumption. These exhibit complex non-smooth deformations without any *a priori* photometric calibration. Therefore, they cannot be handled by SfS methods nor previous methods combining shading and SfT [Malti and Bartoli, 2014; Moreno-Noguer et al., 2009; Varol et al., 2012b].

For quantitative evaluations, we tested with two datasets: *floral paper* and *paper fortune teller*, shown in figure 5.8. Both datasets consist of two creased objects scanned at approximately 20 cm using the structured light system described in §4.4.2. Both datasets have the following conditions: *(i)* the object has a poorly-textured surface, *(ii)* several images show the surface creased, *(iii)* a highly-accurate depth-map associated with each image, *(iv)* the illumination vector is in 3D camera coordinates.

For qualitative evaluations, we used three datasets: *floral sequence*, *t-shirt sequence* and *pillow sequence*, shown respectively in figures 5.12, 5.13 and 5.14. These three datasets have the two following conditions: *(i)* the object has a poorly-textured surface and *(ii)* several images show the surface creased. For *floral sequence*, 8 frames were extracted from a one minute video (uniformly sampled over time), and correspondences were generated with a dense point tracking covering the textured regions. For the *t-shirt sequence* and *pillow sequence* datasets, we used a different camera (Nikon D800 with two different lens) and acquired images with 1920×1080 pixels size. For the *floral sequence* and *pillow sequence* datasets, we constructed the non-shadable template by taking multiple pictures of the object in a rest pose and used Agisoft Photoscan [Agisoft, 2014] to reconstruct the 3D object. Note that the non-shadable template of the *pillow sequence* dataset, contains the pillow and the hand which deforms the pillow as the input images of the figure 5.7 shows.

Table 5.3 summarizes the details of the five poorly-textured datasets. To show the amount and distribution of the correspondences computed for each dataset, we display in figures 5.6 and 5.7 the correspondences for one input image for each dataset.

Name	Nb of images	Nb of corresp.	Matching methods	GT available	Non-shadable template construction
floral paper	8	20	Manual	✓	§5.4.2
paper fortune teller	4	24	Manual	✓	§5.4.2
floral sequence	8	1000	Dense point tracking	×	SfM
t-shirt sequence	5	199	Manual	×	§5.4.2
pillow sequence	4	156	Manual	×	SfM

Table 5.3: List of poorly-textured surfaces datasets for SfT comparison.

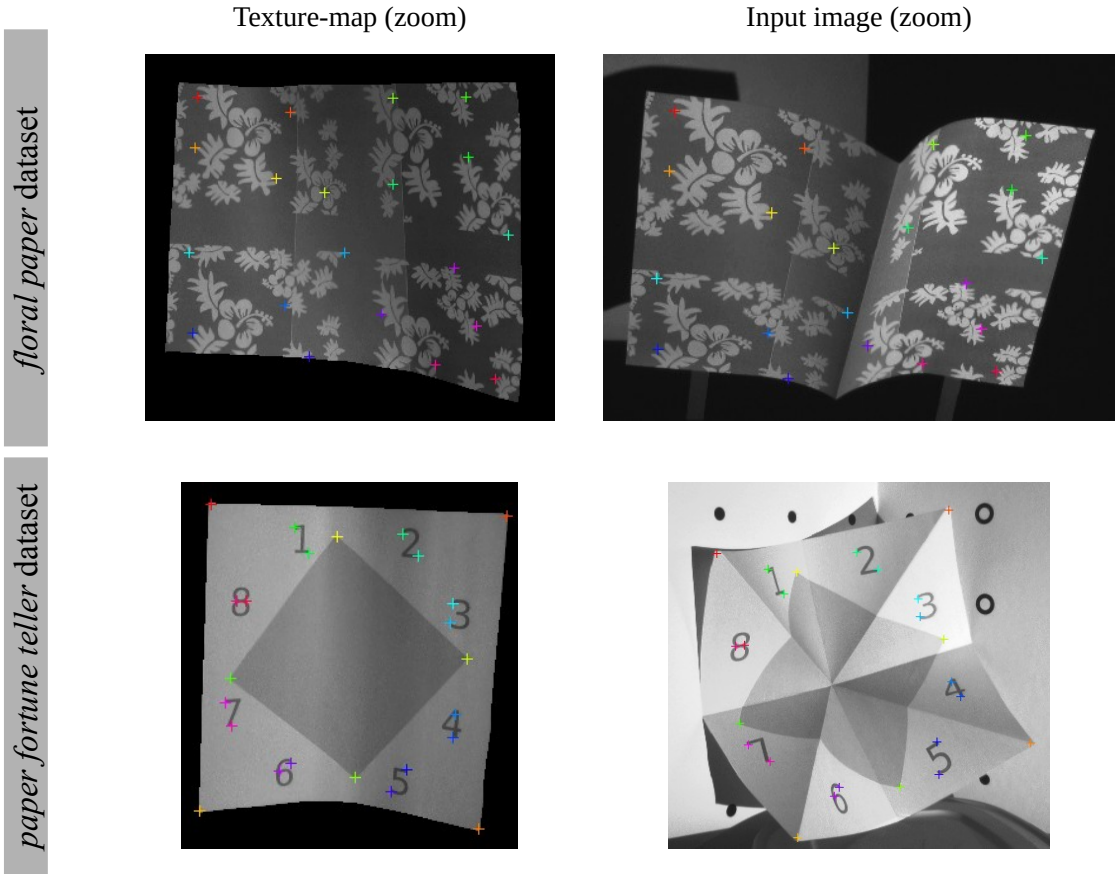


Figure 5.6: Visualization of the correspondences on one input image for the two datasets with ground-truth for the *SfTS-1* problem. We show the correspondences between the texture-map and one input image. **Row n°1**: input image n°5 of the *floral paper* dataset. **Row n°2**: input image n°4 of the *paper fortune teller* dataset.

5.4.4 Implementation Details and Evaluation Metrics

For each dataset, a non-shadable template was constructed by laying a triangulated 100×100 vertex regular grid on the texture-map. It has been shown in chapter 4 that this density allows reconstructing creases. We discretized the boundary points of the texture-map to $N_{\mathcal{B}} = 1000$ uniformly spaced points. Similarly to chapter 4, we used the (ℓ_1-ℓ_2) M-estimator for motion, boundary and smoothing constraints. We used the Huber M-estimator for the shading constraint. In appendix B, we give the hyperparameters for the compared methods. In appendix C, we give the hyperparameters for our four methods. For all methods, we manually set their free parameters to achieve the best performance on all datasets (for our methods these are the weight constraints in C and the Huber parameter).

For the evaluation, we used the relative 3D mean point error (in %) and the mean normal error (in degrees), defined in §4.4.4.

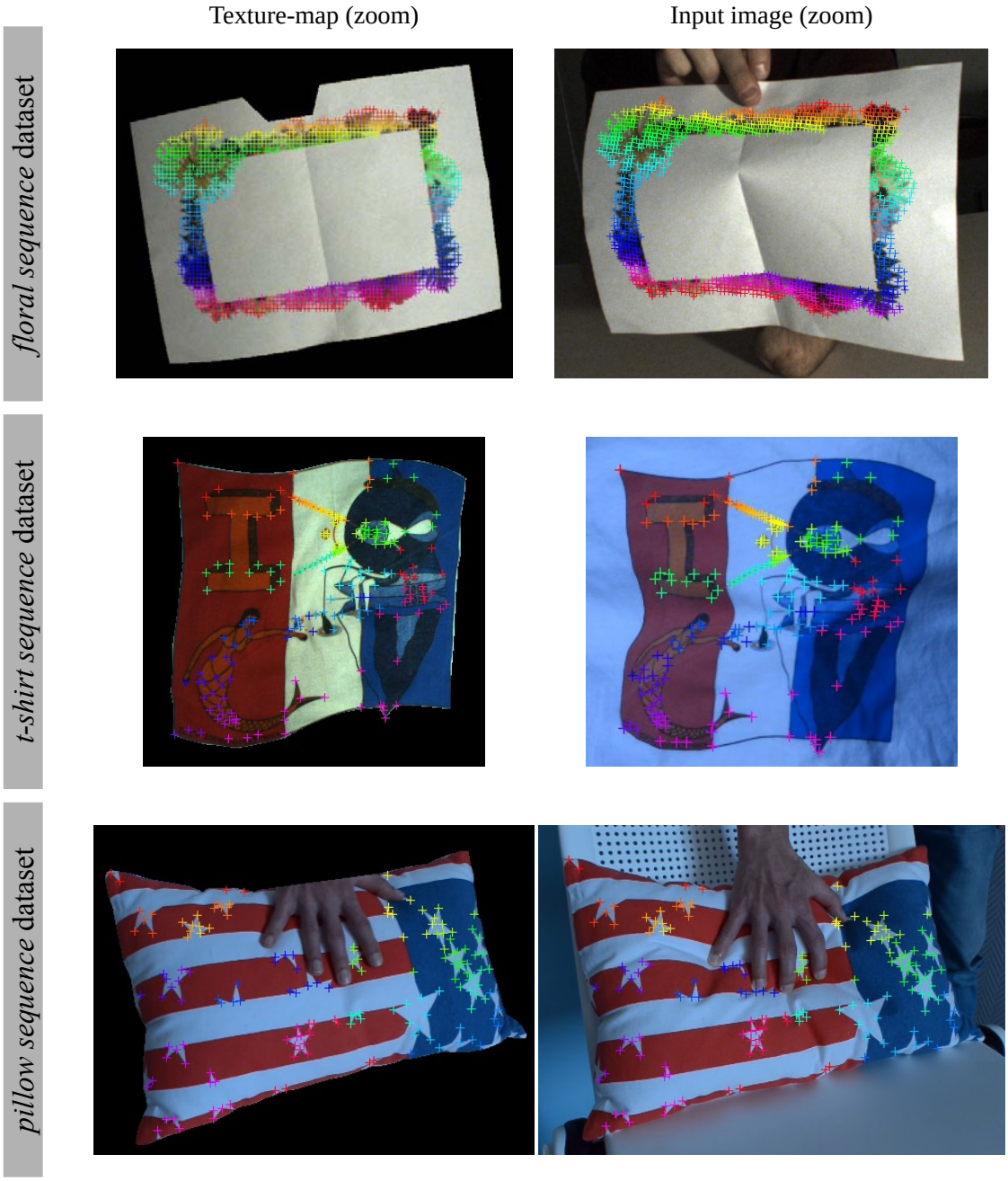


Figure 5.7: Visualization of the correspondences one one input image the three datasets without ground-truth for the *SfTS-1* problem. We show the correspondences between the texture-map and one input image. **Row n°1**: input image n°8 of the *floral sequence* dataset. **Row n°2**: input image n°3 of the *t-shirt sequence* dataset. **Row n°3**: input image n°4 of the *pillow sequence* dataset.

5.4.5 Experiments on Creasable and Poorly-Textured Surfaces

5.4.5.1 Quantitative Results

For the two datasets with ground-truth *floral paper* and *paper fortune teller*, we present in figure 5.8 the 3D reconstructions produced by all the methods. **Ba15a**, **Ch17a** and **Ng16a**

produce smooth surfaces and do not reconstruct the creases. This comes from the fact that these methods interpolate the surface between correspondences using an ℓ_2 regularization. **Sa09a** forms non-smooth deformations but not in the appropriate regions. The reason is that the reconstructed creases are a by-product of the inextensibility constraint which is a relaxation of the isometry constraint. We observe that **Ga16a*** succeeds to create creases when they appear at the surface boundaries. However, these reconstructed creases are not sharp enough and the creases which do not touch the surface boundary are not reconstructed, as the *paper fortune teller* dataset in figure 5.8 illustrates. Our method, **Ga16b_S9K*** and **Ga16b_S9U***, produces significantly better results compared to the state-of-the-art methods: the creases are well registered and reconstructed. We note that the rendered solutions of **Ga16b_S4K*** and **Ga16b_S4U*** are similar to the ones of **Ga16b_S9K*** and **Ga16b_S9U***.

Figure 5.9 shows the reconstruction accuracy of the compared methods for a set of input images in the *floral paper* and *paper fortune teller* datasets. Our methods, **Ga16b_S4K***, **Ga16b_S4U***, **Ga16b_S9K*** and **Ga16b_S9U***, produce the best accuracy for 3D mean point errors and normal errors. In particular, we note that the shading improves notably the normal error. The normal errors are coherent with the renderings in figure 5.8. We note that using second-order spherical harmonics, **Ga16b_S9K*** and **Ga16b_S9U***, improves slightly the reconstructions since this model is a good trade-off between complexity and constraints on the data.

In figure 5.10, we show that we recover the camera responses $\{\beta_t\}$. As ground-truth camera responses are unknown, we can only evaluate $\{\beta_t\}$ qualitatively and this is done by inspecting the estimated values. These are globally similar for our four methods. The higher value of camera response in the last image of the *paper fortune teller* dataset can be explained by comparing the four input images of this dataset. In figure 5.8, we can see that the last image of the *paper fortune teller* dataset looks brighter compared to the three others. This implies some automatic changes of the camera response which our methods succeed to capture.

Figure 5.11 gives the initial, refined and ground-truth illumination vectors for the two datasets with ground-truth and when the first-order of spherical harmonics is used to model the illumination. Only the directional component of the illumination is displayed. We can observe that the initial illumination computed with §5.3.2.2 gives a reasonable good estimate. We also see that the refined illumination vector gives a good estimate for both datasets. We computed the angle error of the illumination vectors by using their directional component, which is given by the first three coefficients of the first-order spherical harmonics model. For the *floral paper* dataset, we obtained an angle error of 23.07 degrees for the initial illumination vector and 3.66 degrees for the refined illumination vector. For the *paper fortune teller* dataset, we obtained an angle error of 24.33 degrees for the initial illumination vector and 16.60 degrees for the refined illumination vector. Recall that this has been achieved without any *a priori* estimate of the illumination vector, which has never been achieved before with images of a deforming surface.

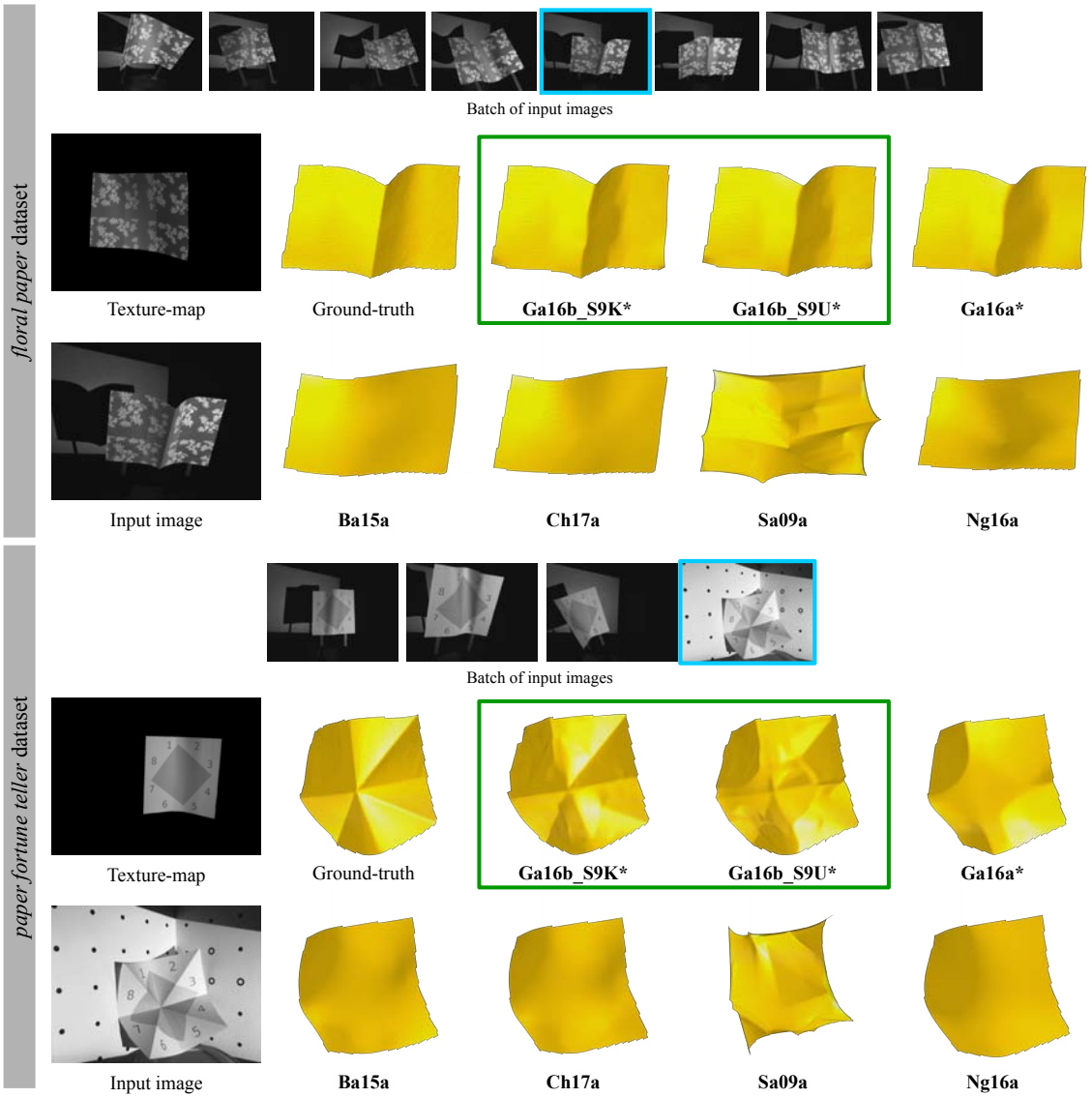


Figure 5.8: Renderings for the two datasets with ground-truth for the *SfTS-1* problem. **Rows n°1** and **n°2**: input image n°5 of the *floral paper* dataset. **Rows n°3** and **n°4**: input image n°4 of the *paper fortune teller* dataset. We indicate by a green rectangle the method which produces the lowest 3D mean point error.

5.4.5.2 Qualitative Results

For the three datasets without ground-truth, we present qualitative results in figures 5.12, 5.13 and 5.14 using our method without and with shading. We note the shading contribution to reveal creases and deformations over textureless regions. There is also another contribution of shading. It allows to reconstruct small changes of curvature, as the *t-shirt sequence* dataset shows with the arrows c and e. For the *t-shirt sequence* dataset, our method reveals at the beret a small deformation which is not present, but this is caused by a mistake on the albedo segments: during the estimation of the albedo segments, the region of the beret is associated to the white part of the t-shirt.

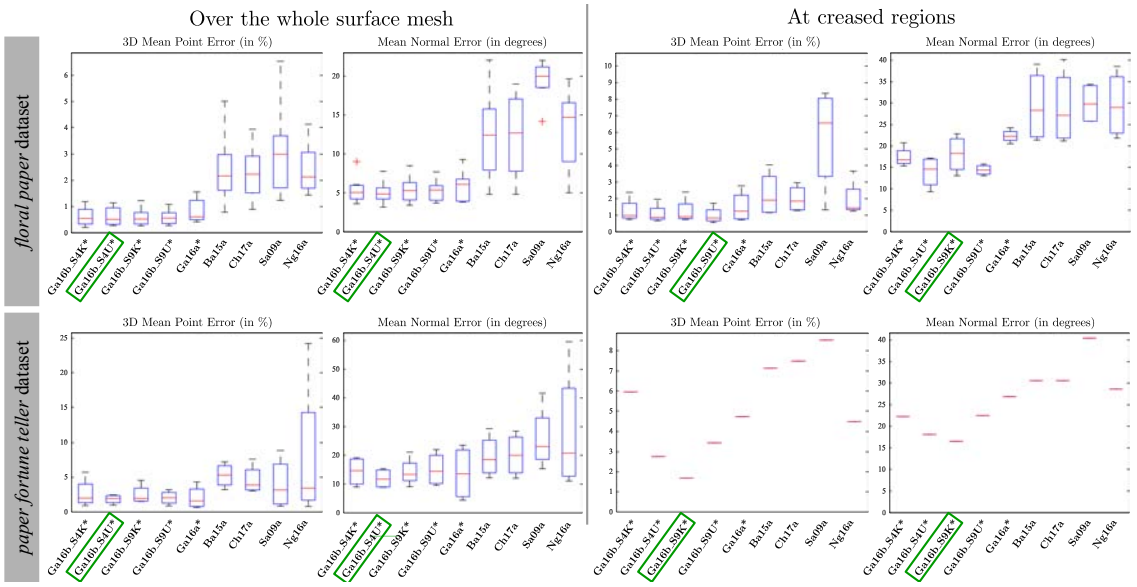


Figure 5.9: Reconstruction accuracy for the two datasets with ground-truth. We indicate by a green rectangle the method which produces the lowest 3D mean point error.

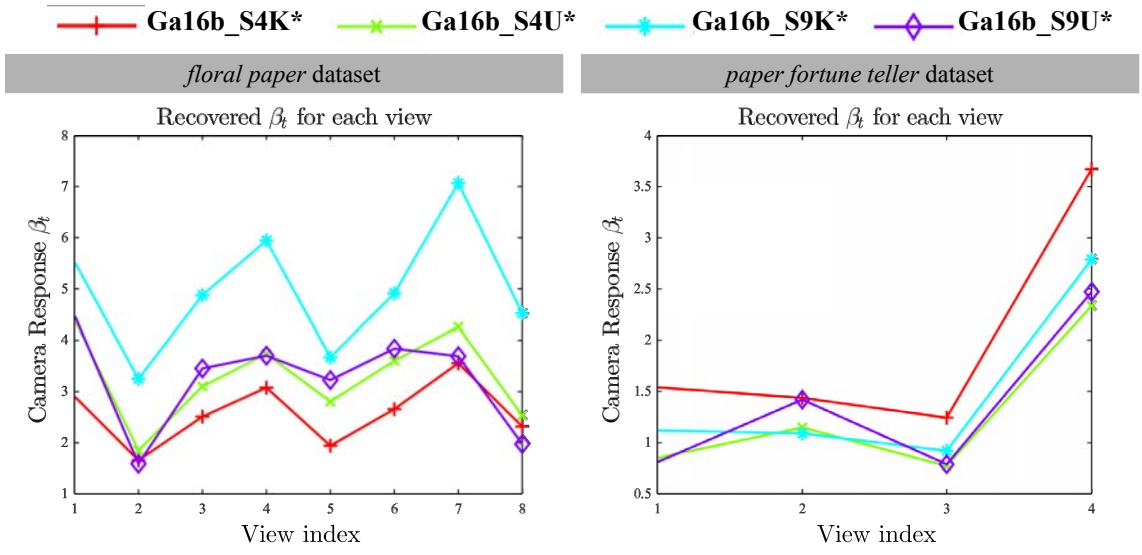


Figure 5.10: Camera responses $\{\beta_t\}$ for the two datasets with ground-truth. Best viewed in color.

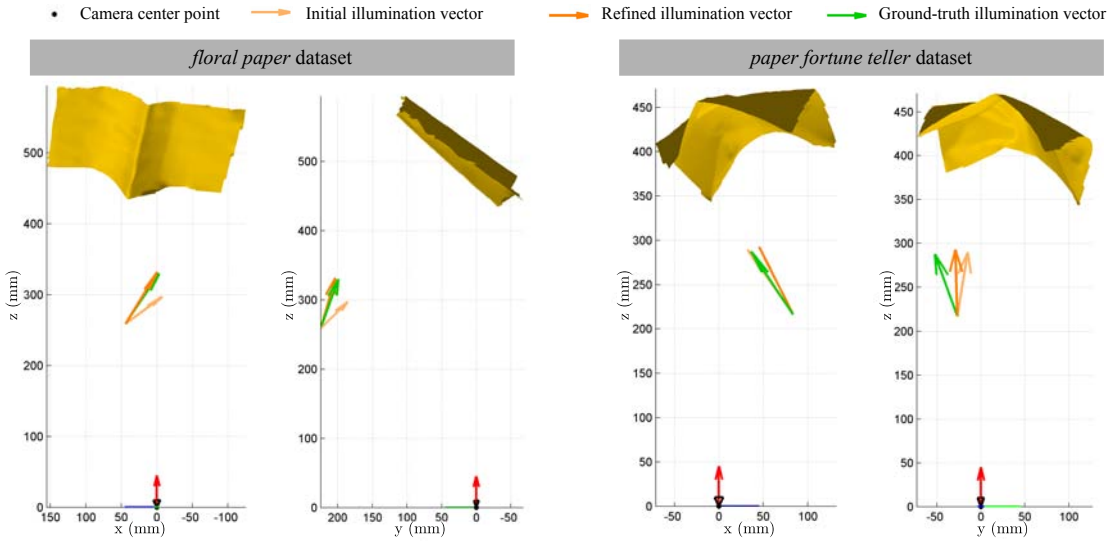


Figure 5.11: 3D visualization of the illumination vector for the two datasets with ground-truth with a first-order of spherical harmonics model. This is the result obtained by **Ga16b_S4U***. **Columns n°1** and **n°2**: views from xz-plane and from yz-plane for the input image n°5 of the *floral paper* dataset. **Columns n°3** and **n°4**: views from xz-plane and from yz-plane for the input image n°4 of the *paper fortune teller* dataset.

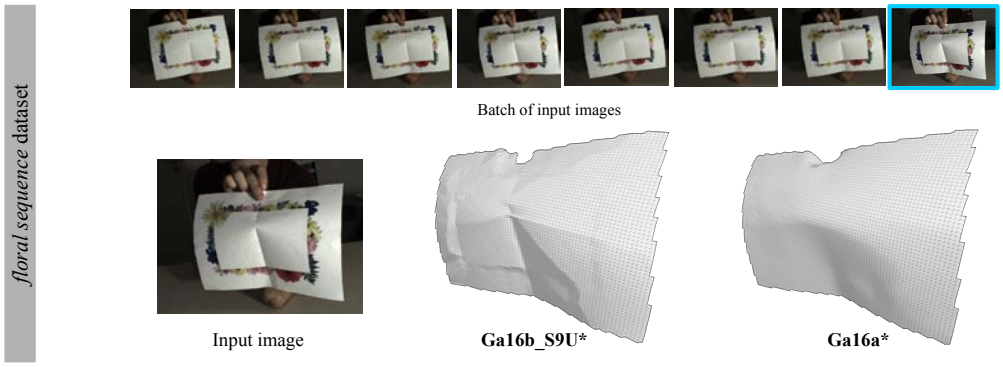


Figure 5.12: Renderings for the *floral sequence* dataset for the *SfTS-1* problem. Note that to improve the visualization of shading contributions, we used a different illumination from the real one.

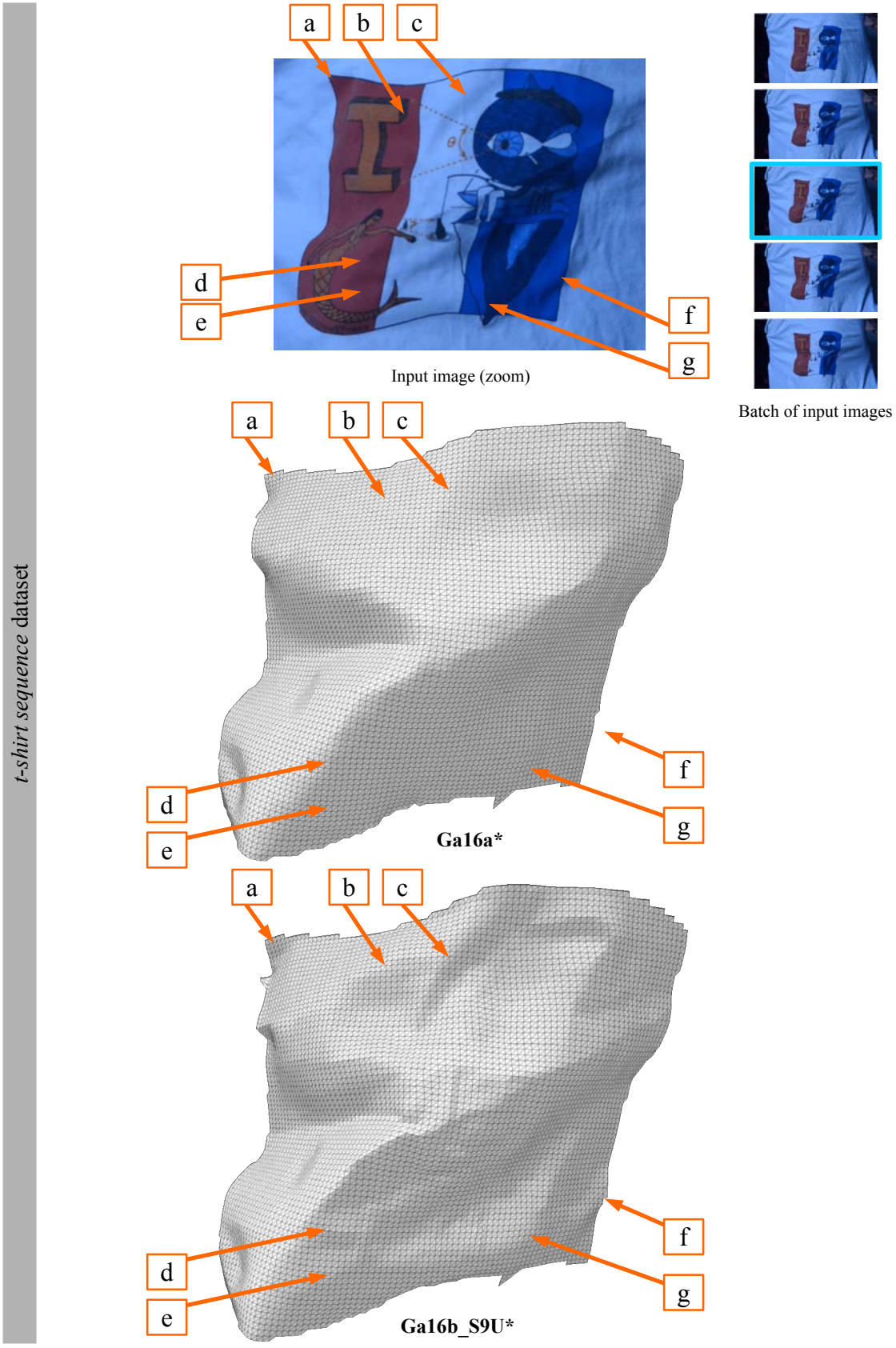


Figure 5.13: Renderings for the *t-shirt sequence* dataset for the *SfTS-1* problem. Note that to improve the visualization of shading contributions, we used a different illumination from the real one. We zoom in the input image to see easily the little 'bumps' over the poorly-textured regions. Some contributions of shading in the reconstruction are indicated by orange arrows.

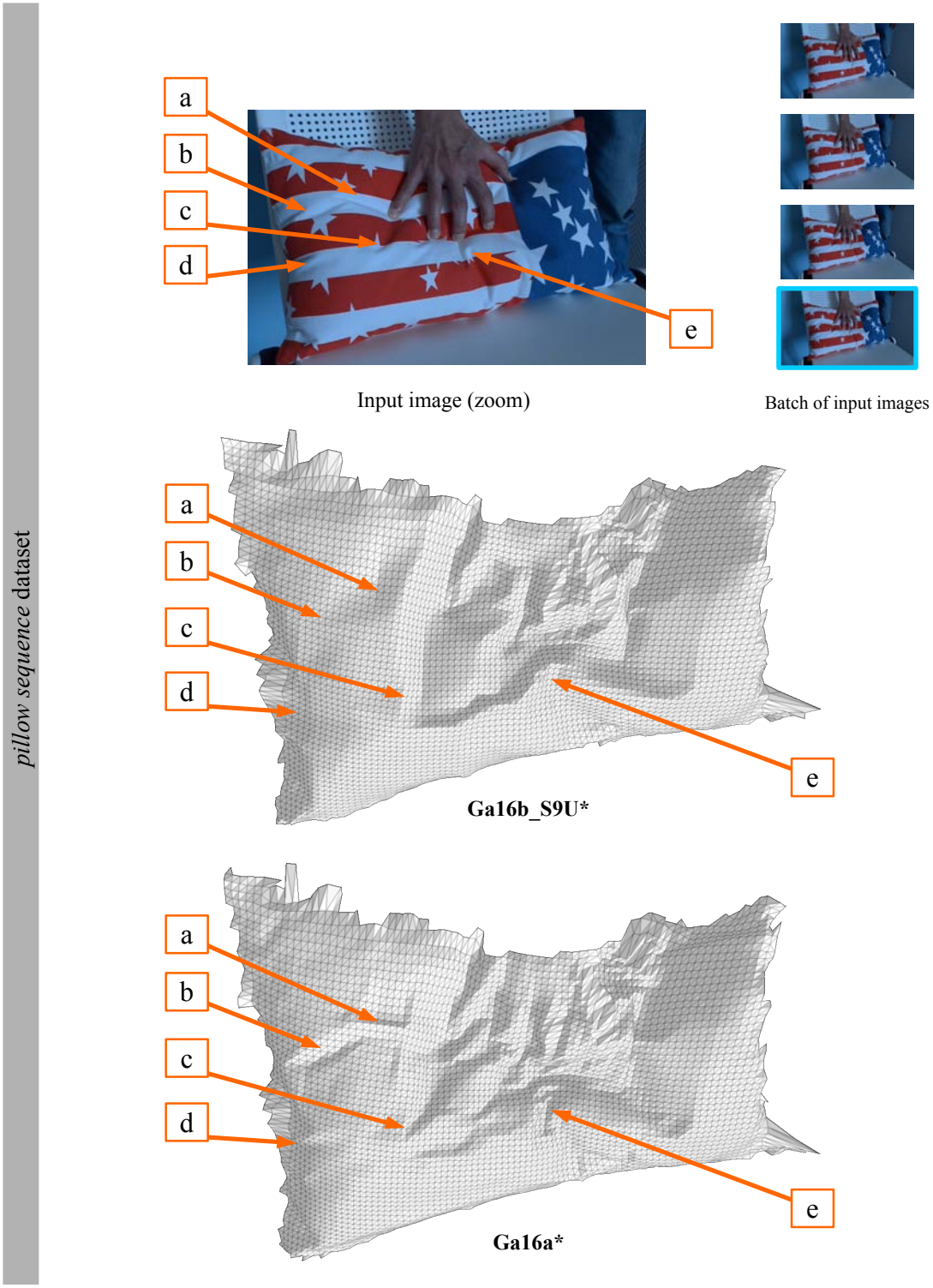


Figure 5.14: Renderings for the *pillow sequence* dataset for the *SfTS-1* problem. Note that to improve the visualization of shading contributions, we used a different illumination from the real one. We zoom in the input image to see easily the little 'bumps' over the poorly-textured regions. Some contributions of shading in the reconstruction are indicated by orange arrows.

5.4.6 Limitations and Failure Modes

We discuss here the main limitations and the failure modes of our solution to *SfTS-1*. The main limitations come from the assumptions of *SfTS-1* we made in §5.2.2: isometric, piecewise-constant albedo and fixed illumination assumptions. There are four main failure modes. The first is when the initial solutions given by the stage 1 are not reliable. Typically this occurs if there are very few, poorly-distributed point correspondences. In these cases, it is difficult to initialize dense shape with any current SfT method. For unorganized image sets, this is a difficult problem to overcome. For video sequences, dense point correspondences can usually be obtained by exploiting temporal continuity and dense frame-to-frame tracking [Collins and Bartoli, 2015]. The second failure mode comes from the use of shading. As in SfS, our method may then suffer from the convex/concave ambiguity, which tends to worsen flawed initial solutions. The third failure mode is the under-segmentation of the albedo-map, which we illustrate with the beret of the *t-shirt sequence* dataset. This may lead to false curvature reconstruction. The fourth failure mode is the presence of some false positive creases, as figure 5.8 shows. This failure mode is linked to the third one, but is more general and can integrate other sources of errors such as mis-registration or the robust estimator applied in the shading constraint.

5.5 Conclusion

We have presented an integrated approach for the reconstruction of complex surface deformations from images using a 3D non-shadable template with shading information. Importantly, we make very light prior assumptions that are common in practical settings. We do not assume the object's reflectance function is known *a priori*, nor do we assume the camera responses, scene illumination and deformations are known *a priori*. The complete set of assumptions are systematically detailed in the *SfTS-1* problem definition §5.2.2. This is the first time this kind of problem has been solved, and is an important step forward in SfT. We have developed a modeling and optimization framework which uses a dense mesh-based surface representation with an associated robust smoothing constraint led by an M-estimator. Combining also a 3D non-shadable template, motion, boundary contour and shading constraints shows that it is possible to reconstruct poorly-textured surfaces under complex deformations and to estimate simultaneously the scene illumination, the camera responses and the surface reflectance parameters such as the albedos, which was not possible with previous methods in SfT or SfS. Our method is the first one to estimate the reflectance parameters using deformed observations and thus does not require a rigid video of the surface, contrary to the previous methods. Our method also allows one to perform SfTS with tracking methods that require a shadable template such as [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. In response to the limitations and the failure modes of our solution to *SfTS-1*, we propose some improvements in chapter 7.

Chapter 6

Using Shading for Joint Template-Free Reconstruction of Creasable, Generic Surfaces and Albedos Estimation

Summary

We address the problem of NRSfMS for creasable and poorly-textured surfaces. The challenge we face is to simultaneously and densely estimate non-smooth, non-rigid shape from each image together with spatially-varying surface albedo. We solve this with a cascaded initialization and a non-convex refinement that combines a physical, discontinuity-preserving deformation prior with motion, shading and boundary contour information. Our approach works on both unorganized and organized small-sized image sets, and has been empirically validated on six real-world datasets for which all state-of-the-art approaches fail. This chapter is based on our peer reviewed paper [Gallardo et al., 2017].

After briefly motivating the general problem NRSfMS, we define the NRSfMS instance we solve in this chapter, following the same characterization of chapter 4. We then instantiate the different models required to solve the problem instance and present our formulation.

6.1 Multi-Images Surface Reconstruction with Shading

In parallel to the use of motion information in NRSfM, shading information has also been used in the case of multi-images surface reconstruction. Several approaches have been studied such as photometric stereo, multi-view SfS, multi-view reconstruction or SfM with SfS. As we show in §2.5.2, these methods assume the objects to be rigid, require complex setups (with two or more cameras) and/or tracked cameras, which restricts significantly their practicability. In response, we propose to combine NRSfM with shading.

Chapter outline. In §6.2, we present our modeling of the problem and our template-free shading-based cost function. In §6.3, we present our optimization framework. In §6.4, we study the basin convergence of our method and validate it with high-accuracy ground-truth datasets and qualitative results. In §6.5, we provide our conclusions.

6.2 Problem Modeling

This section first gives the fundamental models used in the NRSfMS problem which we define in §1.2.6. We then instantiate NRSfMS with a concrete problem which is important to solve and remains general.

6.2.1 Fundamental Models of NRSfMS

In order to solve NRSfMS, six fundamental models are required: the *shape* model, the *surface reflectance* model, the *deformation* model, the *illumination* model, the *camera response* model and the *camera projection* model. The use of these six fundamental models is motivated by the same reasons as in SfTS.

6.2.2 *NRSfMS-1*: Instantiating NRSfMS for Unknown Surface Reflectance Function

An NRSfMS problem specifies the eight components given in §2.1. We refer to this problem instance as *NRSfMS-1*. Table 6.1 presents the instantiations of the fundamental models, given in §6.2.1, for *NRSfMS-1*.

(a) Models. The instantiations are the same as the ones given for *SfTS-1* and their use is motivated by the same reasons. *(b) Exploited visual cues.* The visual cues we use are motion, boundary contour and shading. Similarly to *SfTS-1*, motion is used to constrain textured regions of the surface and boundary contour to constrain surfaces edges. Motion and boundary contour allow to obtain a good registration of the surface, which is really important

Fundamental model	Known *a priori*	Fixed or time-varying	Instantiation
Shape	×	Fixed	High-resolution thin-shell 3D mesh
Surface reflectance	×	Fixed	Lambertian with piecewise-constant albedo
Deformation	✓	Fixed	Isometric, crease-preserving parameterized by barycentric interpolation
Illumination	✓	Fixed	Spherical harmonics (first and second-order) and attached to the camera, as in §5.2.2
Camera projection	✓	Fixed	Perspective
Camera response	✓	Time-varying	Linear general response, as in §5.2.2

Table 6.1: Fundamental model instantiations in *NRSfMS-1*.

to use shading. We use shading constraint to densely reconstruct surfaces and reveal creases in poorly-textured regions. *(c) Number of required images.* In our experiments, we use batch sizes of 5. We discuss the implications of using smaller batch sizes in the conclusion §6.4.7. *(d) Expected types of deformations.* As in §4.2.2, we assume quasi-isometric and piecewise-smooth deformations, and no tearing. *(e) Scene geometry.* As in §4.2.2, we assume no self or external occlusions, but there can be background clutter. Another aspect is related to what we call the *reference image*. The reference image is one of the input images which indicates the surface to reconstruct (with a segmentation mask of the surface). Our modeling and algorithm may in principle use any image as the reference image. In practice however, we have obtained better reconstruction accuracy for a reference image where the surface is smooth. *(f) Requirement for putative correspondences.* We assume to know *a priori* a set of putative 2D correspondences. We assume there may be a small proportion of mismatches *e.g.* $< 20\%$. *(g) Surface texture characteristics.* We consider generic surfaces which present both textured and poorly-textured regions. *(h) Known and unknown model parameters.* The unknowns are the albedo-map (segments and values) and the vertices of the shape model in the camera coordinates of each input image. The illumination, the camera responses and the camera intrinsics are known. These assumptions are reasonable for two reasons. First, the illumination and the camera can be calibrated and the camera responses can be obtained from the camera or computed using *e.g.* the background. Second, it is unrealistic to know *a priori* the reflectance model of a surface as the object is *a priori* unknown, contrary to SfT.

6.2.3 Shape, Deformation and Reflectance Modeling

We now define the specific fundamental models which we use: the *shape model*, *deformation model* and *surface reflectance model*. These are the same as the ones in §5.2.3. We recall the shape model of §4.2.3 and adapt it to model the shape of the first image. Then, we recall the deformation model of §4.2.3 and the surface reflectance of §5.2.3.

For the *shape model*, we use a high-resolution thin-shell 3D mesh to model the object's 3D

surface, which we build by meshing Ω using a regular 2D triangular mesh, with M vertices. Similarly to *SfT-1* and *SfTS-1*, M is on the order of 10^4 in our experiments. We denote the mesh's edges as E, where N_E is the number of edges. Our task is to determine, for each mesh vertex i, its position $\mathbf{v}_t^i \in \mathbb{R}^3$ in 3D camera coordinates for each image $t \in [1, N]$. We use $\mathcal{V}_t = \{\mathbf{v}_t^i\}_{i\in[1,M]}$ to denote the vertices in 3D camera coordinates for image t. We parameterize $\mathcal{V}_1$ along lines-of-sight. Specifically, let $\mathbf{u}_i \in \mathbb{R}^2$ denote the 2D position of the i^{th} vertex in the first image, defined in normalized pixel coordinates. Its corresponding position in 3D camera coordinates at $t = 1$ is $\mathbf{v}_i^1 = d_i[\mathbf{u}_i^\top, 1]^\top$, where d_i is its unknown depth. We collect these unknown depths into the set $\mathcal{D} = \{d_1, \ldots, d_N\}$. The full set of unknowns that specify the object's shape in all images is therefore $\{\mathcal{D}, \mathcal{V}_2, \ldots, \mathcal{V}_N\}$, which corresponds to $3M(N-1) + M$ real-valued unknowns.

The *deformation model* transforms each vertex to 3D camera coordinates: we model the position of each vertex $i \in \{1 \ldots M\}$ in camera coordinates by $\mathbf{v}_t^i \in \mathbb{R}^3$, where t denotes time. We transform a point $\mathbf{u} \in \Omega_\mathcal{T}$ to camera coordinates according to $\mathcal{V}_t$ with the same barycentric interpolation φ detailed in §4.2.3 and $n(\mathbf{u}; \mathcal{V}_t) : \mathbb{R}^{3\times M} \to \mathbb{S}_3$ to represent its unit surface normal. We also assume isometry and crease-preserving smoothness and impose them through the cost function which we define in §6.2.5.

For the *surface reflectance model*, we define an *albedo-map* $A(\mathbf{u}) : \Omega_A \to \mathbb{R}^+$ as the function that gives the unknown albedo for a pixel $\mathbf{u} \in \Omega_\mathcal{T}$. From the piecewise-constant assumption we can write this as $A(\mathbf{u}) : \Omega_\mathcal{T} \to \mathcal{A}$ where $\mathcal{A} = \{\alpha_1, \ldots \alpha_K\}$ denotes a discrete set of K unknown albedos with $\alpha_k \in \mathbb{R}^+$. We discuss how A is built in §6.3.4.

6.2.4 Inputs and Outputs

Our inputs are as follows. *(i)* a set of N input RGB images $\{I_t\}_{t\in[1,N]}$, $I_t : \mathbb{R}^2 \to 0, 255^3$ with a deforming object and the corresponding intensity images $\{L_t\}_{t\in[1,N]}$, $L_t : \mathbb{R}^2 \to \mathbb{R}^+$. *(ii)* the camera intrinsics of all perspective projection functions Π_t. *(iii)* a segmentation of the object of interest in the reference image, denoted by the region $\Omega \subset \mathbb{R}^2$. *(iv)* the scene illumination coefficients which we denote by $\mathbf{l} \in \mathbb{R}^{4 \text{ or } 9}$. *(v)* N sets $\mathcal{S}_t$ of matched putative 2D correspondences from Ω to each input image I_t. We denote it by $\mathcal{S}_t = \{(\mathbf{u}_j, \mathbf{p}_t^j)\}$ where $\mathbf{u}_j$ denotes the j^{th} 2D point in Ω and $\mathbf{p}_t^j$ denotes its corresponding position in the t^{th} input image I_t. The number of correspondences for each image t is denoted by s_t. In our experiments, these are correct. Details for how this is done for our experimental datasets are given in §6.4.3. We did not evaluate quantitatively how robust to mismatches is our method, but it has the potential to handle them as we explain in §6.2.5.

The outputs of our solution to *NRSfMS-1* are: *(i)* the vertices $\mathcal{V}_t$ of the shape model in the camera coordinates for all input images and *(ii)* the segmented albedo-map A with its K segments and values $\{\alpha_1, ..., \alpha_K\}$

6.2.5 Problem Modeling with an Integrated Cost Function

The cost function combines *physical deformation priors* (quasi-isometry and smoothing constraints) with shading, motion and boundary constraints extracted from all images without knowing a template (contrary to §4 and §5). The objective function C_{total} has the following form:

$$C_{total}(\mathcal{V}_1,\dots,\mathcal{V}_N,\alpha_1,...,\alpha_K) \triangleq \sum_{t=1}^{N} C_{shade}(\mathcal{V}_t,\alpha_1,...,\alpha_K) + \tag{6.1}$$
$$\lambda_{motion}C_{motion}(\mathcal{V}_t) + \lambda_{contour}C_{contour}(\mathcal{V}_t) + \lambda_{iso}C_{iso}(\mathcal{V}_1,\mathcal{V}_t) + \lambda_{smooth}C_{smooth}(\mathcal{V}_t).$$

The terms C_{shade}, C_{motion} and $C_{contour}$ are shading, motion and boundary contour data constraints respectively. The terms C_{smooth} and C_{iso} are physical deformation prior constraints. The terms λ_{motion}, $\lambda_{contour}$, λ_{iso} and λ_{smooth} are positive weights and are the method's tuning parameters. The image data constraints and the smoothing constraint are similar to the ones of **Ga16b***. The isometry constraint is different since in our problem (and more globally in NRSfM) we do not have the template of the surface *a priori*. To solve *NRSfMS-1*, we solve the following minimization problem:

$$\min_{\mathcal{V}_1,...,\mathcal{V}_N,\alpha_1,...,\alpha_K} C_{total}(\mathcal{V}_1,\dots,\mathcal{V}_N,\alpha_1,\dots,\alpha_K). \tag{6.2}$$

The shading constraint. The shading constraint robustly encodes the Lambertian relationship between albedo, surface irradiance and pixel intensity. Similarly to chapter 5, we evaluate the shading constraint at each pixel of albedo segments wider than T_A, which gives:

$$C_{shade}\left(\mathcal{V}_t,\alpha_1,...,\alpha_K\right) \triangleq \frac{1}{|\Omega_A|}\sum_{\mathbf{u}\in\Omega_\mathcal{T}} \rho\Big(A(\mathbf{u})\, r\left(n(\mathbf{u};\mathcal{V}_t);\mathbf{l}\right) - L_t\big(\Pi_t\circ\varphi(\mathbf{u};\mathcal{V}_t)\big)\Big). \tag{6.3}$$

We use the Huber M-estimator with free parameter set to 0.005. We set T_A to the same default parameter as in chapter 5 ($T_A = 0.022\%$ of the number of pixels contained in the image).

The motion constraint. We recall that the set $\mathcal{S}_t$ holds s_t putative correspondences between Ω and image $t \in [1, N]$. The constraint robustly encourages each point $\mathbf{u}_j$ to transform to its corresponding point $\mathbf{p}_t^j$, and is given by:

$$C_{motion}(\mathcal{V}_t) \triangleq \sum_{(\mathbf{u}_j,\mathbf{p}_t^j)\in\mathcal{S}_t} \rho\Big(\Big\|\Pi_t\circ\varphi(\mathbf{u}_j;\mathcal{V}_t) - \mathbf{p}_t^j\Big\|\Big). \tag{6.4}$$

Similarly to chapters 4 and 5, we handle mismatches with an M-estimator.

The boundary contour constraint. We discretize the boundary of Ω to obtain a set of boundary pixels $\mathcal{B} \triangleq \{\mathbf{u}_{k\in[1,N_\mathcal{B}]}\}$, with $N_\mathcal{B}$ the number of boundary pixels. We then compute

a boundariness map for each image $B_t : \mathbb{R}^2 \to \mathbb{R}^+$ where high values of $B_t(\mathbf{p})$ correspond to a high likelihood of pixel $\mathbf{p}$ being on the boundary contour. The constraint is evaluated as:

$$C_{contour}(\mathcal{V}_t) \triangleq \frac{1}{N_{\mathcal{B}}} \sum_{\mathbf{u}_k \in \mathcal{B}} \rho\Big(B_t\big(\Pi_t \circ \varphi(\mathbf{u}_k; \mathcal{V}_t)\big) \Big). \tag{6.5}$$

We build it using an edge response filter that is modulated to suppress false positives according to one or more segmentation cues. We use two different segmentation cues: the projection-based and the color-distribution segmentation cues. We give the exact choice for computing B_t for each tested dataset in appendix E.

The crease-preserving smoothing constraint. This is based on the smoothing constraint used in chapters 4 and 5:

$$C_{smooth}(\mathcal{V}_t) \triangleq \frac{1}{|\Omega|} \sum_{\mathbf{u}_j \in \Omega} \rho\left(\frac{\partial^2 \varphi}{\partial \mathbf{u}^2}(\mathbf{u}_j; \mathcal{V}_t)\right). \tag{6.6}$$

The quasi-isometry constraint. We enforce quasi-isometry using mesh edge-length constancy. Specifically, we measure the constancy with respect to the mesh edges in the reference image. This is defined as follows:

$$C_{iso}(\mathcal{V}_1, \mathcal{V}_t) \triangleq \frac{1}{|E|} \sum_{(i,j) \in E} \left(1 - \|\mathbf{v}_1^i - \mathbf{v}_1^j\|_2^{-2} \, \|\mathbf{v}_t^i - \mathbf{v}_t^j\|_2^2\right)^2. \tag{6.7}$$

Handling the scale. In our cost function (6.1), the shading, the motion, the boundary contour and the quasi-isometry constraints are invariant to the scale of the reconstruction, however the smoothing constraint is not invariant. This is because a trivial solution for the smoothing constraint is to put all vertices at the origin. Therefore, to rule out the dependency on scale, we constrain the mean depth of the reconstruction to a fixed positive value. Details are given in §6.3.5.

6.3 Optimization Strategy

6.3.1 Overview

Optimizing equation (6.1) is a non-trivial task because it is large-scale (typically $O(10^5)$ unknowns), is highly non-convex, and the shading constraint requires dense, pixel-level registration. Recall that we do not assume the images come from an uninterrupted video sequences, which makes dense registration much harder to achieve. Our strategy is to first achieve a rough initial estimate for the shape parameters $(\mathcal{D}, \mathcal{V}_2, \dots \mathcal{V}_N)$ (and hence an initial estimate for registration) using only motion constraints from the point correspondences. We then introduce the boundary contour constraints and refine these estimates by optimizing equation (6.1) using iterative numerical minimization. Next we estimate albedos by fixing the

shape parameters, and finally optimize equation (6.1) over all unknowns using all information (point correspondences, boundary and shading) using iterative numerical minimization. We propose this strategy because point correspondences can be used to provide a rough, smooth solution to non-rigid shape without requiring an initial estimate. By contrast we find that boundary contour and shading constraints require a good initialization to prevent incorrect convergence in a local minimum. Concretely, our optimization strategy is divided into four stages which we now describe in detail. A schematic of the whole process is illustrated in figure 6.1.

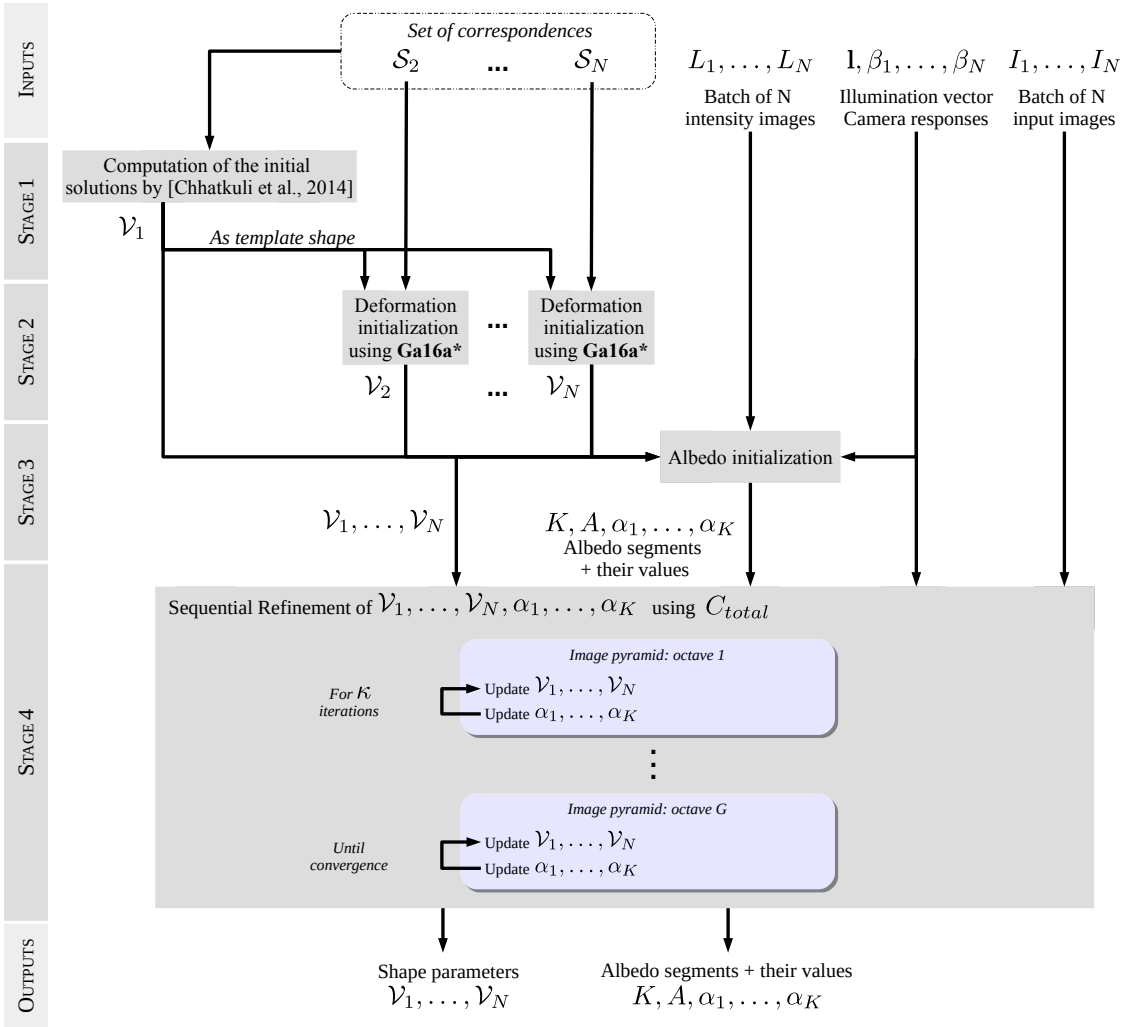


Figure 6.1: Schematic of our proposed solution to solve *NRSfMS-1*.

6.3.2 Stage 1: Correspondence-Based Template Initialization

We take the point correspondences $\{\mathcal{S}_t\}$ and input them to an existing surface-based, initialization-free NRSfM method. The method we currently used is [Chhatkuli et al., 2014] which has publicly available code[1]. This provides us with a rough estimate of the reference

[1]The code is available at igt.ip.uca.fr/~ab/Research/LIIP-NRSfM_v1p0.zip

image's vertex depths $\mathcal{D}$. Note that all existing initialization-free surface-based methods assume the object's surface is smooth in all views, thus the initial estimate will not normally be highly accurate.

6.3.3 Stage 2: Motion and Boundary-Based Shape-from-Template

We back-project the mesh vertices in the reference image using their initial depth estimates $\mathcal{D}$. This gives a rough estimate of the object's 3D shape in a reference position (corresponding to the reference image). We then use this mesh as a template, and call an existing SfT method to initialize, for each image, the vertex positions $\mathcal{V}_t$ using the correspondence set $\mathcal{S}_t$. The current method we use is **Ga16a***. We then optimize equation (6.1) without shading by setting $\lambda_{shade} = 0$, over the shape unknowns $\{\mathcal{V}_2, \dots \mathcal{V}_N\}$ with $\mathcal{D}$ kept fixed. This can be done efficiently because the unknowns are now decoupled between images, so each $\mathcal{V}_t$ can be optimized independently. Finally we optimize equation (6.1) over the shape unknowns $\{\mathcal{V}_2, \dots \mathcal{V}_N\}$ with $\lambda_{shade} = 0$ and by fixing the first shape $\mathcal{D}_1$. To achieve good convergence we compute the boundariness map (equation (6.5)) with an image pyramid, using $G = 3$ octave.

6.3.4 Stage 3: Albedo Initialization

We now use our current shape estimates to infer albedos using the shading constraint. For this we segment the reference image into local superpixel-like clusters, where within each cluster we assume the albedo is constant. For the same reasons as in §5.3.2.3, we aim for an oversegmentation: neighboring segments can share the same albedo but within each segment we assume the albedo is constant. We achieve this by performing an intrinsic image decomposition [Bell et al., 2014] on the reference image's intensity image and cluster the resulting 'reflectance image' using [Fukunaga and Hostetler, 1975] with a low cluster tolerance (we use a default of 10). For each cluster k, we assign a corresponding albedo α_k. This is done by taking each pixel $\mathbf{u}_j$ in the cluster, estimating its albedo by inverting the shading equation: $\alpha \approx L_t\left(\Pi_t \circ \varphi(\mathbf{u}; \mathcal{V}_t)\right) r\left(n(\mathbf{u}; \mathcal{V}_t); \mathbf{l}\right)^{-1}$. We then initialize α_k as the median over all estimates within the cluster.

6.3.5 Stage 4: Full Refinement

We refine our estimates by minimizing equation (6.1) using all constraints and over all unknowns, which is achieved with Gauss-Newton iterative optimization and backtracking line-search. Because of the very large number of unknowns, at each iteration we solve the normal equations using an iterative solver (diagonally-preconditioned conjugate gradient), with a default iteration limit of 200. Recall that there is a scale ambiguity (as in all NRSfM problems), because we cannot differentiate a smaller surface viewed close to the camera from a large surface viewed far away. We fix the scale ambiguity by scaling all vertices to have a mean depth of 1 after each iteration. To achieve good convergence, we blur each L_t with a Gaussian blur pyramid, with a default of three octaves. For the first two pyramid levels, we

run Gauss-Newton until either convergence is reached or a fixed number of iterations have passed (we use $\kappa = 20$ iterations). For the final pyramid level, we run it until convergence. Processing time is typically several minutes for small-sized image sets (< 10 images), with an unoptimized Matlab implementation on the CPU.

6.4 Experimental Validation

6.4.1 Overview

We divide the experimental validation into two parts. In the first part, we analyze the convergence basin of our energy function through perturbation analysis. This is to understand both how sensitive our formulation is to the initial solution, and fundamentally, whether the *NRSfMS-1* problem can be cast as an energy-based minimization with a strong local minimum near the true solution. In the second part, we compare performance to state-of-the-art NRSfM methods. Our evaluation has been performed using public datasets and three new datasets, all with ground-truth.

6.4.2 Methods Compared

We compare with the following competitive NRSfM methods [Chhatkuli et al., 2014, 2016; Parashar et al., 2016; Taylor et al., 2010; Torresani et al., 2008a; Varol et al., 2009; Vicente and Agapito, 2012], denoted respectively with **To08a**, **Va09a**, **Ta10a**, **Vi12a**, **Ch14a**, **Ch16a** and **Pa16a**. **To08a**, **Ta10a**, **Vi12a** and **Ch16a** are methods which reconstruct only point correspondences, whereas **Va09a**, **Ch14a** and **Pa16a** are methods which reconstruct dense surfaces. We recall that the star ***** stands for the proposed methods. To see the contribution of some constraints of equation (6.1), we compare with two versions of our method, **Ga17a_NoS***, where shading is not used, and **Ga17a_NoB***, where the boundary constraint is not used in stages 2 and 4. We briefly describe each method in table 6.2.

6.4.3 Datasets

We evaluated on six datasets which mostly respect the Lambertian assumption. Each dataset consists of a disc-topology surface in 5 different deformed states, with one state per image. We show these in figures 6.5, 6.6 and 6.7. From top down we have *floral paper* from chapter 5, the *paper fortune teller* from chapter 5, *creased paper*, *pillow cover*, *hand bag* and *Kinect paper* from [Varol et al., 2012a]

The *Kinect paper* dataset is a video dataset and has no accompanying illumination parameters and no camera response function. We approximated camera response with a constant linear model, and estimated the illumination parameters using the image data and the accompanying depth-maps. This was performed by selecting in a small rectangular region on the surface with both constant albedo and non-saturated pixels, then measuring the average pixel intensity within the region and fitting a local plane to the region using the depth map. This was repeated using 30 images in the sequence, and we then estimated the spherical

Acronym	Source	Shape model	Constraints	Principle
Ga17a*	Proposed	Mesh	M+C+Sh+I+Ss	Initialization (**Ch16a**) + Constraints minimization
Ga17a_NoS*	Proposed	Mesh	M+C+I+Ss	Initialization (**Ga16a***) + Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **known** and modeled by **first-order** spherical harmonics
Ga17a_NoB*	Proposed	Mesh	M+Sh+I+Ss	Initialization (**Ga16a***) + **Illumination**/Camera responses initialization + Albedo initialization + Shading-based minimization with illumination **unknown** and modeled by **first-order** spherical harmonics
To08a	[Torresani et al., 2008]	Point set	M+Ss+Ts	Modeling the 3D shape using **shape basis**, a spatial prior (Gaussian prior on shape coefficients) and a temporal smoothing (linear dynamical model of the shape)
Va09a	[Varol et al., 2009]	Point set	M+I+Ss	Estimation of homographies for **local** patches + Decomposition of each homography to obtain normal up to two-fold ambiguity + Disambiguation with **spatial smoothness** + Normal integration
Ta10a	[Taylor et al., 2010]	Mesh	M+I	Solving SfM for **locally-rigid triangles** + Grouping the reconstructed triangles using global consistency
Vi12a	[Vicente and Agapito, 2012]	Mesh	M+I+Ss	Generation of proposals using a greedy region growing with isometry constraint + Selection of the proposal with minimum energy + **Fusion moves** optimization with gradient descent strategy to generate new proposals
Ch14a	[Chhatkuli et al., 2014]	Point set	M+I+Ss	Estimation of homographies for **infinitesimal** planes + Decomposition of each homography to obtain normal up to two-fold ambiguity + Disambiguation using **additional views** + Normal integration
Ch16b	[Chhatkuli et al., 2016]	Point set	M+Inex	Convex formulation (**Maximum Depth Heuristic**), where the surface with the maximum depth is favored and where the point inter-distances sum to 1, then optimization over a second-order cone
Pa16a	[Parashar et al., 2016]	Point set	M+I+Ss	Solving a sum-of-squares of systems of two quartics in two variables for each image pair + Normal integration

M: Motion, C: boundary Contour, Sh: Shading, I: Isometry, Inex: Inextensibility, Ss: Surface smoothing, Ts: Temporal smoothing

Table 6.2: List of NRSfM methods used for the comparison. We give their specific components of modeling and resolution.

harmonics illumination vector by inverting the Lambertian shading model using linear least squares. The 5 images we used for evaluation were uniformly sampled from the video.

We followed the same procedure as described in §5.4.2 to make the *creased paper*, *pillow cover* and *hand bag* datasets. Each dataset has a set of point correspondences between the first and all other images. As all datasets, except the *Kinect paper* dataset, are poorly-textured, the correspondences are sparse. We note that manual correspondences are commonly used to evaluate NRSfM methods and this is why the correspondences of our datasets were computed manually. To show the amount and distribution of the correspondences computed for each

dataset, we display in figure 6.2 the correspondences for one input image for each dataset.

Name	Nb of images	Nb of corresp.	Matching methods	GT available
floral paper	5	20	Manual	✓
paper fortune teller	5	24	Manual	✓
creased paper	5	20	Manual	✓
Kinect paper	5	1503	[Garg et al., 2013]	✓
pillow cover	5	69	Manual	✓
hand bag	5	155	Manual	✓

Table 6.3: List of datasets for NRSfM methods comparison.

6.4.4 Implementation Details and Evaluation Metrics

Similarly to the problems of *SfT-1* and *SfTS-1*, we constructed, for all experiments, the embedding meshes by laying a triangulated 100×100 vertex regular grid on the reference image which was then cropped to Ω. We also discretized the boundary points of the texture-map to $N_{\mathcal{B}} = 1000$ uniformly spaced points. For the state-of-the-art methods, there is no way to automatically optimize their free parameters. Therefore we tried our best to do this by hand, to obtain the best reconstruction accuracy on all datasets. This was done by a search starting from the default values, and modifying each free parameter in turn to improve the reconstruction accuracy. For our method, all experiments were ran using the same parameters, which were manually set. In appendix D, we give the weights of the different constraints and the hyperparameters for our method and the compared methods.

We measured reconstruction accuracy by comparing 3D distances and normals with respect to ground-truth. Because reconstruction is up to scale, we computed for each method the best-fitting scale factor that aligns the predicted point correspondences with their true locations in the ℓ_2 sense, then measured accuracy with the scale-corrected reconstruction. This was done at three locations: *(i)* at point correspondences, *(ii)* densely across the ground truth surface, and *(iii)* densely at creased regions, which are any points on the ground-truth surfaces that are within 5 mm of a surface crease. *(ii)* and *(iii)* were used to investigate the contribution of the shading constraint over the whole surface and at creased regions. The equivalent grids for *(ii)* and *(iii)* were constructed by sampling uniformly the respective locations.

3D point root mean square error (PRMSE): We computed the PRMSE (in mm) between the reconstructed surface $\mathcal{V}$ and the ground-truth surface on the grid $\mathcal{G}$:

$$PRMSE(\mathcal{V}, Q^*, \mathcal{G}) = \sqrt{\frac{\sum_{\mathbf{u}\in\mathcal{G}} \left(\varphi(\mathbf{u};\mathcal{V}) - Q^*(\varphi(\mathbf{u};\mathcal{V}))\right)^2}{|\mathcal{G}|}}, \tag{6.8}$$

with $Q^* : \mathbb{R}^3 \to \mathbb{R}^3$ the function which gives the 3D point of the ground-truth surface closest to the input 3D point.

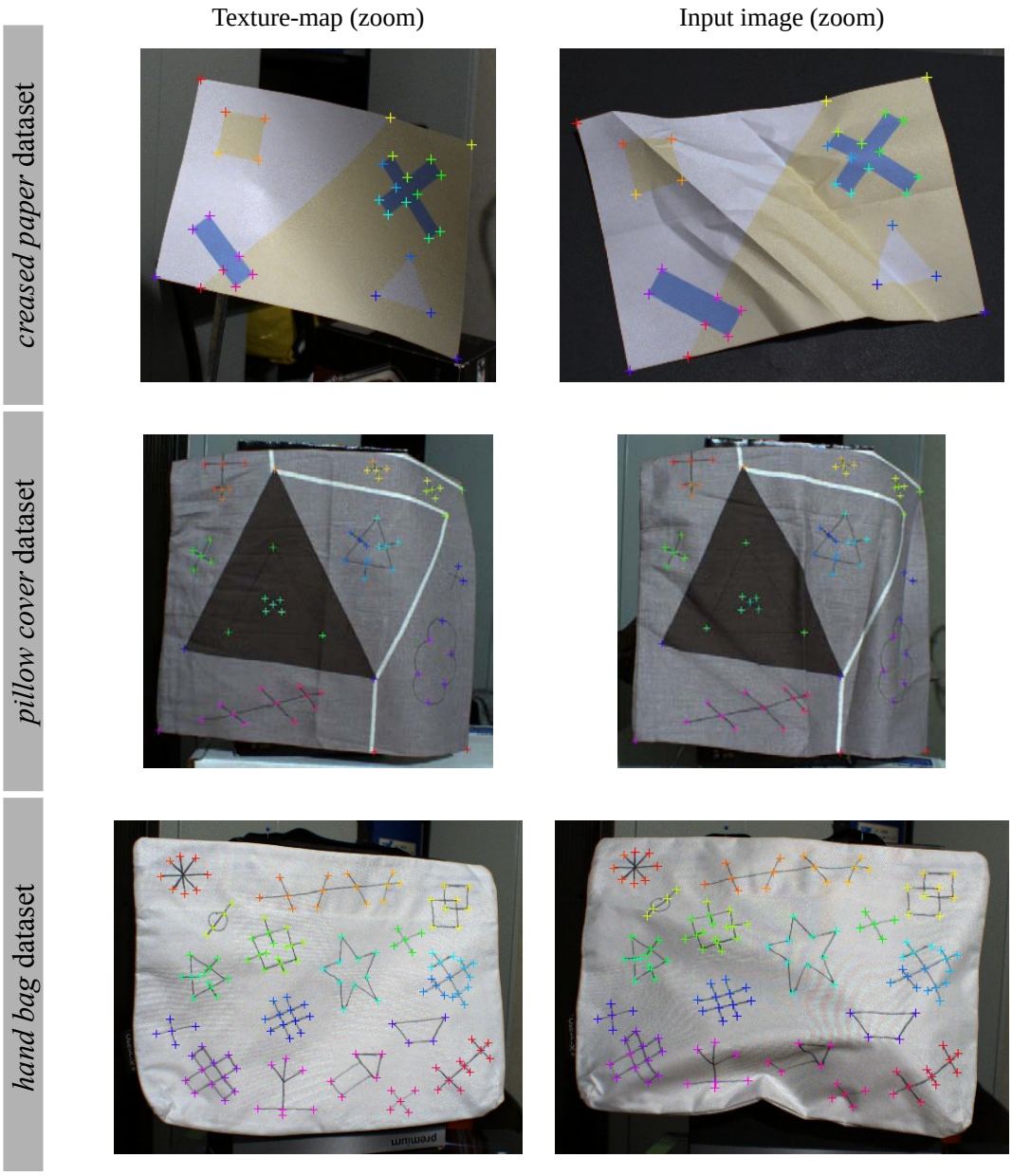


Figure 6.2: Visualization of the correspondences on one input image for the three datasets with ground-truth, *creased paper*, *pillow cover* and *hand bag*, for the *NRSfMS-1* problem. We show the correspondences between the reference image and one input image. **Row n°1**: input image n°5 of the *creased paper* dataset. **Row n°2**: input image n°5 of the *pillow cover* dataset. **Row n°3**: input image n°5 of the *hand bag* dataset.

Mean normal error (MNE): This is the average error in surface normal over the region $\mathcal{G}$ in the input image belonging to the surface. We computed the normal error (in degrees) between the reconstructed surface $\mathcal{V}$ and the ground-truth surface on the grid $\mathcal{G}$:

$$MNE(\mathcal{V}, n^*, \mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\mathbf{u} \in \mathcal{G}} \cos^{-1} \left(n^\top(\mathbf{u}; \mathcal{V})\, n^*(\varphi(\mathbf{u}; \mathcal{V})) \right), \tag{6.9}$$

with $n(\mathbf{u};\mathcal{V}) : \mathbb{R}^{3\times M} \to \mathbb{S}_3$, the unit normal and $n^* : \mathbb{R}^3 \to \mathbb{S}_3$ the function which gives the 3D normal of the 3D ground-truth point closest to the input 3D point.

6.4.5 Quantitative and Qualitative Results

We show in figures 6.5, 6.6 and 6.7 the test datasets and the reconstructions from our method and the best performing previous method (the one with lowest PRMSE with respect to *(ii)* above). Visually we can see that considerable surface detail is accurately reconstructed by our method as well as the global shape.

In figure 6.4, we give the reconstruction accuracy statistics across all test datasets and all compared methods. The first row gives from left to right the distance PRMSE at point correspondences *(i)*, over the whole ground-truth surfaces *(ii)* and over creased regions in the ground-truth surfaces *(ii)*. The second row gives the respective surface mean normal error. The *Kinect paper* dataset has no creases and the deformation is very smooth in all images. We observe that, for all datasets other than *Kinect paper*, there is a good improvement with respect to all error metrics compared to the other methods. This is strongest in the second and third columns, which show our method successfully exploits shading information in textureless and creased regions. For the *Kinect paper* dataset, we see that our method does not obtain the highest accuracy across all error metrics. The reason is that it is a very smooth, densely textured surface, and shading is not needed to achieve an accurate reconstruction. However, our method still obtains competitive results on this dataset. We observe that the use of shading improves globally the shape of the reconstructions and that the boundary contour constraint allows using shading better.

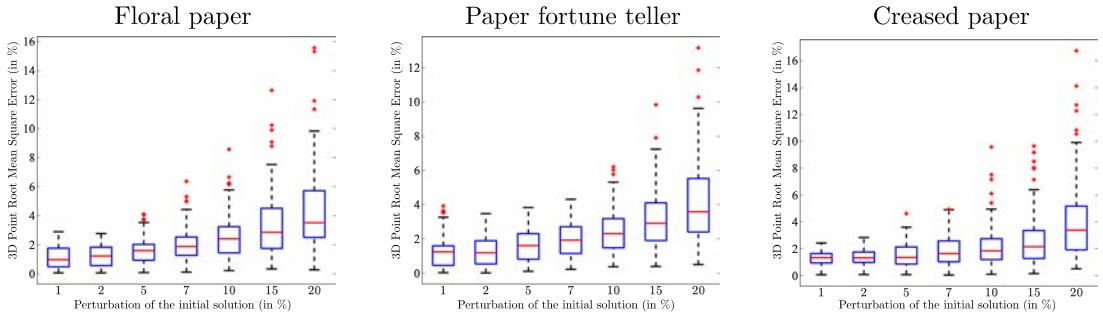


Figure 6.3: Numerical results of the convergence basin analysis.

6.4.6 Convergence Basin Analysis

We performed perturbation analysis as follows. We started with an initial reconstruction close to the ground-truth, then applied a low-pass filter (to smooth out creases, because we do not expect these to be present in the initial solution), then randomly perturbed the vertex positions using smooth deformation functions. For each perturbation, we optimized equation (6.1) by performing stages 3 and 4 in §6.3. The initial solution was carefully done by hand, using the ground-truth surfaces, point correspondences, and a quasi-isometric non-

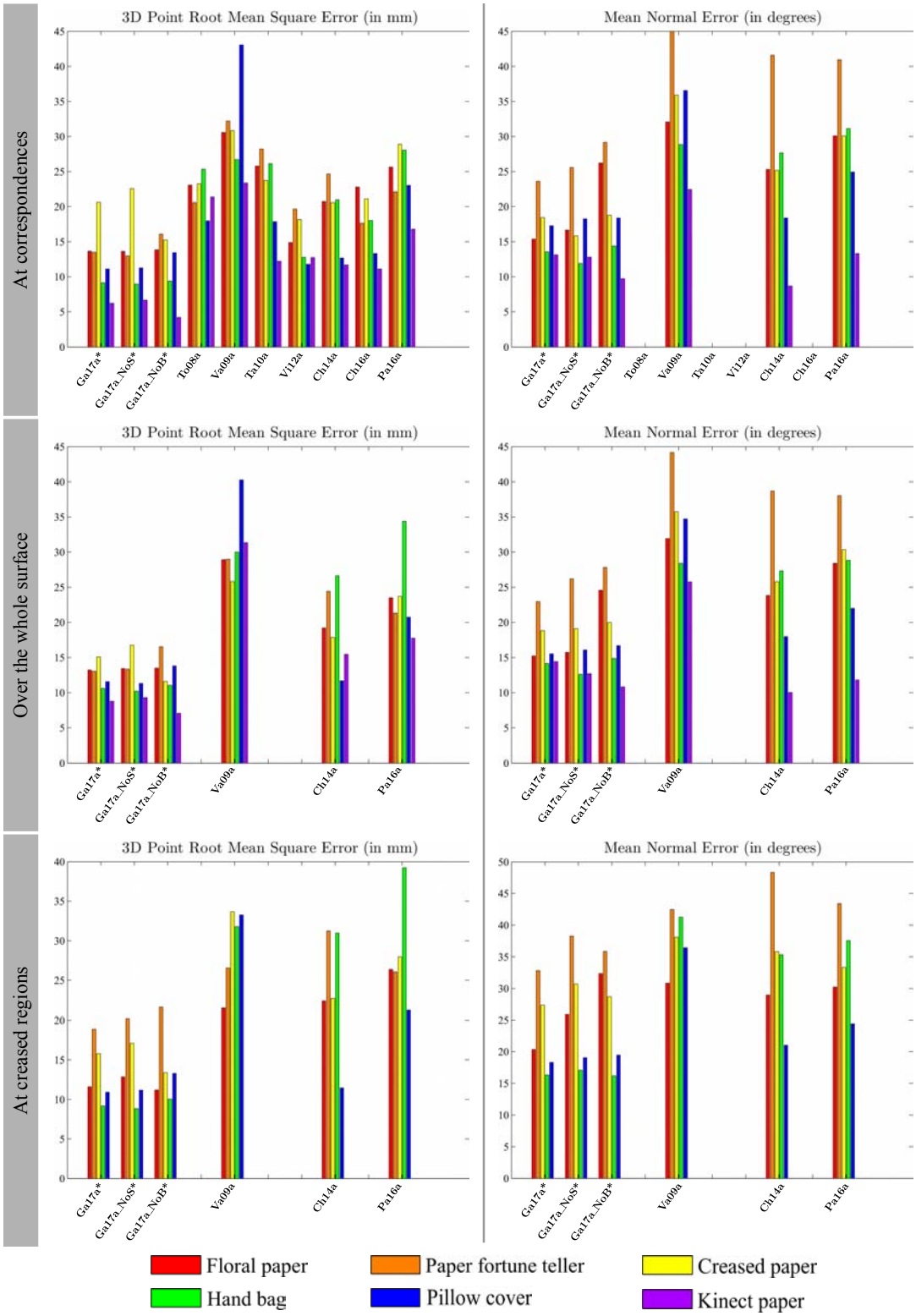


Figure 6.4: Reconstruction accuracy statistics across all test datasets and all compared methods. We recall that **To08a**, **Ta10a**, **Vi12a** and **Ch16a** reconstruct only point correspondences, whereas **Va09a**, **Ch14a** and **Pa16a** reconstruct dense surfaces. Also, the *Kinect paper* dataset does not present any crease.

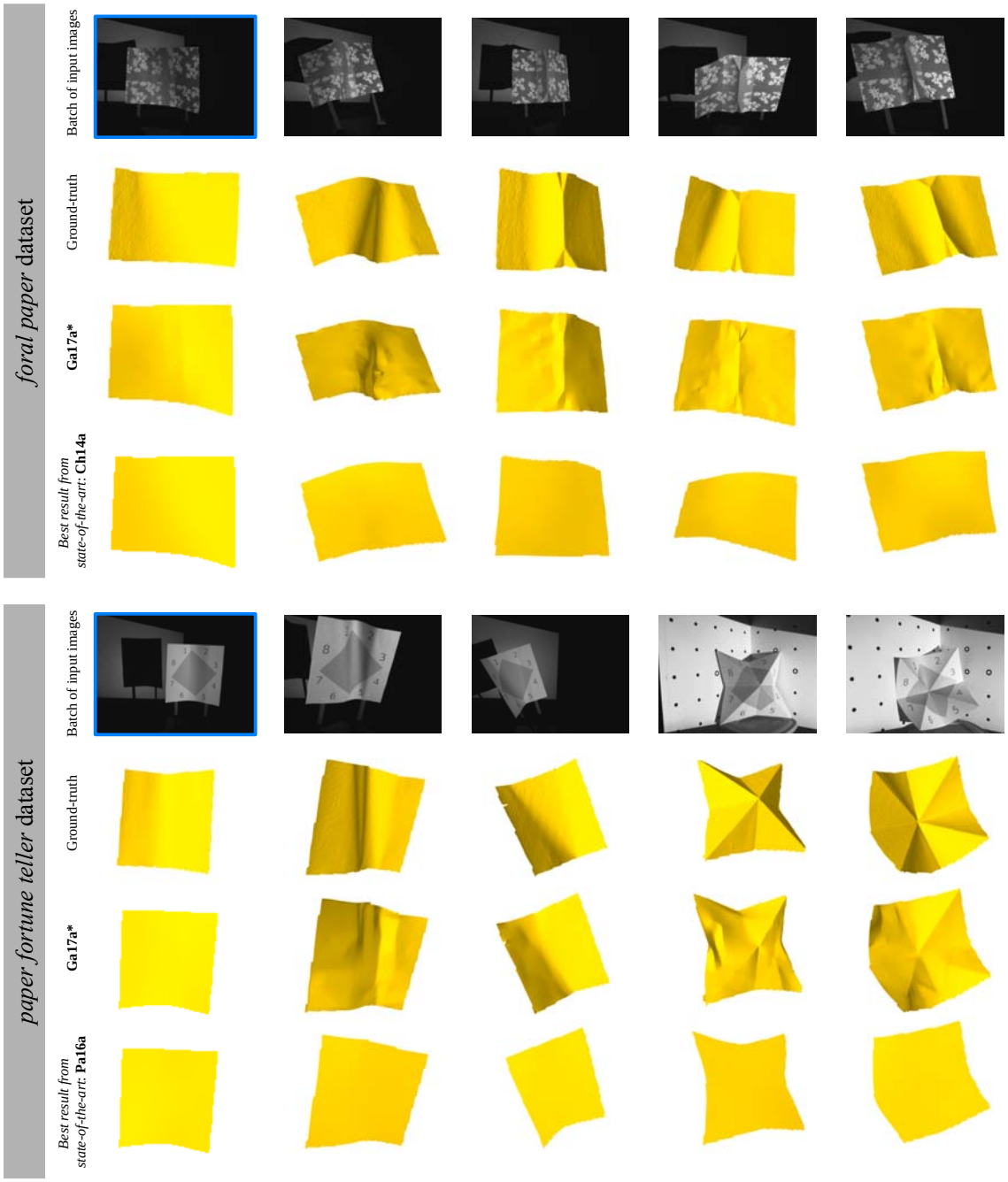


Figure 6.5: Renderings for the *floral paper* and *paper fortune teller* datasets with ground-truth. Here we show the images from each dataset, and sample reconstructions from one of the images using our method and the best performing NRSfM method. We frame the reference image using a blue line.

rigid Iterative Closest Point (ICP) registration. The perturbations were designed to globally deform the initial solutions, which is more realistic than a local perturbation of each vertex. This was implemented using a $4 \times 4 \times 4$ B-spline enclosing the reconstructed surfaces and randomly perturbing the spline's control points at 7 different noise levels, with 30 random perturbations per noise level. We report results as box-plots for the *floral paper*, *paper fortune teller* and *creased paper* datasets in figure 6.3. The x-axis gives the average perturbation in

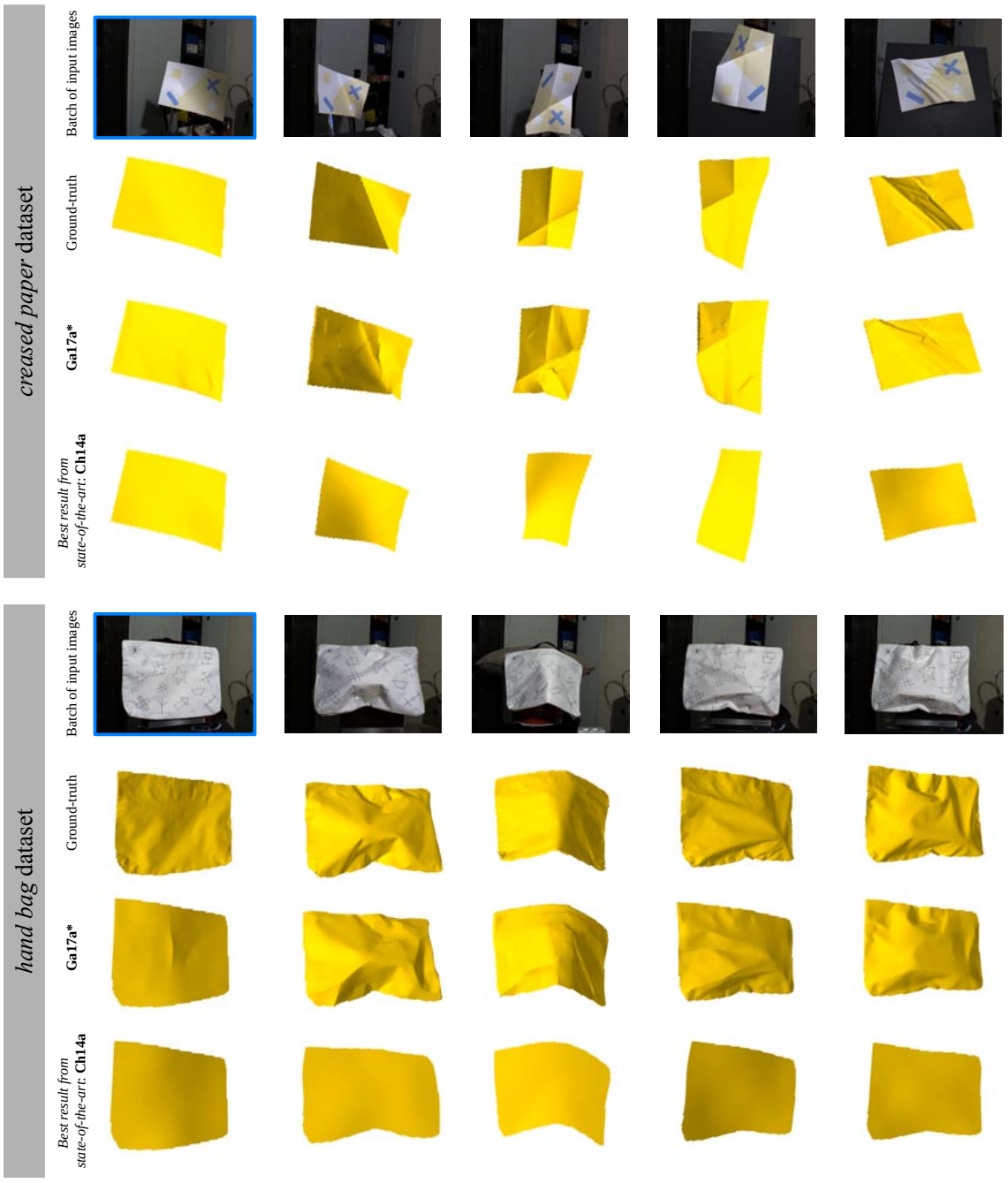


Figure 6.6: Renderings for the *creased paper* and *hand bag* datasets with ground-truth. Here we show the images from each dataset, and sample reconstructions from one of the images using our method and the best performing NRSfM method. We frame the reference image using a blue line.

mm for each noise level from the initial solution. The y-axis gives the dense surface PRMSE as defined in §6.4.4 for each random sample. For small noise levels ($< 5\%$), the box-plots are very similar, which tells us our energy landscape has a strong local minimum close to the ground-truth, which supports our claim that the *NRSfMS-1* problem can be cast as an energy-based minimization (via equation (6.1)). For larger noise levels ($> 5\%$), we can see a significant increase in error, indicating that the optimization now becomes trapped more

Figure 6.7: Renderings for the *pillow cover* and *Kinect paper* datasets with ground-truth. Here we show the images from each dataset, and sample reconstructions from one of the images using our method and the best performing NRSfM method. We frame the reference image using a blue line.

frequently in local minima.

6.4.7 Limitations and Failure Modes

We discuss here the main limitations and the failure modes of our solution to *NRSfMS-1*. One limitation of our solution is that the parameters of our method are set manually and may vary with the datasets. This is because we observe that we did not find default parameters for all

datasets yielding to the best reconstruction accuracy. As the number of of tuned parameters is relatively small, this is not a critical issue. It would be interesting to investigate whether there exist fixed tuning parameters which work well on all datasets, using *e.g.* grid search. Another limitation is that we perform our experiments with batches of 5 images and we have not performed a theoretical analysis to establish the minimal number of images to solve *NRSfMS-1*. On the one hand, NRSfM can be solved up to ambiguities with two images. On the another hand, SfS can be solved with one image when the illumination and the surface reflectance are known. At first sight, two images seem to be sufficient to solve *NRSfMS-1*, however a thorough theoretical study would be required. Our method is also limited by the assumptions made in §6.2.2. These assumptions are that we have isometric deformations, piecewise-constant albedo, fixed and known illumination vector and known camera responses. Regarding failure modes, our solution to *NRSfMS-1* presents the same ones as in chapter 5, which we give in §5.4.6. These are when a good initial solution cannot be obtained after stages 1 and 2, when there is a convex/concave ambiguity, when the albedo-map is undersegmented and when there are some false positive creases. As figures 6.5, 6.6 and 6.7 show, the fourth failure mode is more significant in the context of NRSfM rather than in the context of SfT. The difference can be explained by the fact that the registration problem is more difficult to solve for *NRSfMS-1* than for *SfTS-1*.

6.5 Conclusion

We have studied the *NRSfMS-1* problem by combining NRSfM and shading with unknown, spatially varying albedos. This is a hard and important vision problem, needed for high-accuracy dense reconstruction of poorly-textured surfaces undergoing non-smooth deformation from 2D images. We have proposed an energy-based solution and a cascaded numerical optimization strategy, and have demonstrated encouraging results on six real-world datasets, for which all competitive NRSfM methods fail. This marks the first time that strongly creased, deformable, poorly-textured surfaces with unknown albedos have been densely reconstructed and registered from 2D image sets without a 3D template. In the next chapter, we give a broad perspective of all the works presented in the thesis as well as directions for possible future works.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis describes our contributions to monocular deformable 3D reconstruction through the study of curvilinear objects and the use of multiple visual cues for surface models. On the one hand, we have studied the case of SfT with 1D curves which deform isometrically or quasi-isometrically. On the other hand, we have proposed new methods of SfT and NRSfM which combine several visual cues in order to handle more complex deformations, specifically creasing, and weakly-textured surfaces, which could not be handled by previous methods. We gather here our conclusions regarding our different contributions and then propose directions for future work.

7.1.1 Shape-from-Template for Curvilinear Models

We have presented a theoretical study of Curve SfT and its implementation to reconstruct respectively 2D and 3D curves from 1D template. Unlike Surface SfT, we have proved that Curve SfT has ambiguous solutions in general. We have given the necessary and sufficient conditions to solve the problem using the super critical points, which are computed directly from the data. We have shown that, unlike Surface SfT, Curve SfT cannot be locally solved using non-holonomic solutions. Regarding implementation, we have given four methods of different categories. Two methods, from categories *(i)* and *(ii)*, provide only one solution. The method from category *(iii)* refines a given solution using a new angle-based parameterization of the 2D and 3D curves. Only the method from category *(iv)*, which is based on a discrete HMM, estimates all candidate solutions by taking advantage of our theory and more precisely by using the super critical points. We have also proposed several methods of super critical point detection. We have quantitatively and qualitatively evaluated the method based on the discrete HMM and its refined version on simulated and real curves, such as a necklace. This has shown that such curves can be reconstructed by Curve SfT.

7.1.2 Use of Multiple Visual Cues for SfT and NRSfM

The majority of existing SfT and NRSfM methods present two main limitations: they cannot reconstruct poorly-textured surfaces and surfaces under complex deformations such as creases. This can be explained by two characteristics of current methods. First, the methods which use only motion constraints (which constitute the majority of methods) are fundamentally insufficient because motion information is not available at poorly-textured regions. Secondly, the methods which use shading constraints use ℓ_2 curvature-based regularization, which prevents the formation and reconstruction of creases. Third, the methods which use shading constraints also require the surface reflectance to be known prior to any deformation. This assumption significantly simplifies the problem and implies that the scene is controlled, which limits their practical use. To overcome both limitations, we have proposed two ideas which are integrated to a modeling and optimization framework based on a non-convex cost function. These ideas have been applied in the template-based setting (SfTS) and the template-free setting (NRSfMS). Fundamentally, we have presented the first approach to solve SfTS when surface reflectance is unknown a priori and the surface can crease. We have also presented the first approach to solve NRSfMS, a problem not previously investigated in the literature.

The first idea is to use an adaptive smoothing constraint which allows us to model creases without knowing *a priori* their locations. We have built this constraint using a robust penalization based on an M-estimator. Thanks to an analysis of different M-estimators, we have verified that the non-redescending M-estimators (ℓ_1-ℓ_2) or Huber allow the formation of creases and show very similar reconstructions. Experiments of real object reconstruction by SfT have underlined the capability of our method to reconstruct creases. The second idea is to combine the visual cues of motion and boundary contour with shading. This allows us to constrain densely poorly-textured surfaces, which cannot be done with motion and boundary contour constraints. The use of the previously mentioned adaptive smoothing constraint is an essential prerequisite to the integration of shading: this makes shading capable of revealing creases. However, the use of shading requires one to know the photometric parameters, which are the surface reflectance, the illumination and the camera response. Our assumptions are that the surface reflectance is unknown and Lambertian with piecewise-constant albedo regions, which is a good approximation of a large number of common objects. For SfTS, we have proposed a method which simultaneously estimates the surface deformation and all photometric parameters using at least four images where the surface deforms. Because the use of shading leads to a highly non-convex problem, a key element of this method is the initialization of the photometric parameters from this set of images. For NRSfMS, as the problem is less constrained than SfTS (due to the lack of a template), we have assumed the illumination and the camera response to be known. We have presented the first solution of NRSfM with shading when the reflectance model is unknown, which is a more difficult problem, but more useful. The contribution of shading has been demonstrated quantitatively and qualitatively thanks to an evaluation of our SfT and NRSfM methods on several real objects such as paper and fabric. This thesis has shown that the use of multiple visual cues and an

adaptive smoothing constraint enlarges the spectrum of surfaces and isometric deformations which SfT and NRSfM can reconstruct.

7.2 Future Work

Some aspects of our contributions need to be developed and other research directions to be explored.

7.2.1 Shape-from-Template for Curvilinear Models

Two open problems for Curve SfT are the improvement of detection of super critical points and the study of closed curves.

Improving the detection of super critical points. Two key elements of our HMM solution are the super critical points and their detection. However, this detection requires in practice the interpolation of several functions involved in the ODE, such as the image warp. The detection involves the second derivatives of these interpolations, which may be unstable, leading then to inaccurate estimation of the super critical points' location. Improving this estimation will improve the reconstruction accuracy. The study of different interpolation functions may lead to a more robust and accurate detection of super critical points, yielding to better reconstruction accuracy.

Extension to closed curves. The second direction is the adaptation of the HMM-based method to handle templates with loops such as a closed necklace. This makes the problem more complicated because solving the graphical problem becomes NP-hard. However, we expect that good results can be achieved using approximate inference methods such as loopy belief propagation.

7.2.2 Use of Multiple Visual Cues for SfT and NRSfM

We give here three concrete research directions, but there exist many other open problems such as improving the use of shading, extending NRSfM to volumetric models and using more complex photometric models.

Learning-based approaches for SfT and the estimation of surface reflectance and illumination. Recently, learning-based approaches such as Convolutional Neural Networks (CNN) have shown very good performance to solve problems where input data present the same object contained in the learning data. Examples of these problems includes object detection and recognition or pose estimation. As SfT assumes a prior knowledge of the object to reconstruct (the template), learning-based approaches seem to be an interesting direction to solve SfT. In this sense, some works such as [Golyanik et al., 2018; Pumarola et al., 2018] have recently proposed to use CNN to solve SfT. However, [Golyanik et al., 2018]

does not show results for real images and [Pumarola et al., 2018] trains the proposed CNN from the depth-maps and their associated 2D images which are present in the test data. One limitation of both methods is that they seem to reconstruct only smooth deformations.

An important challenge which this thesis tried to take up is the estimation of the reflectance and the illumination. Some works using CNN [Kim et al., 2017; Mandl et al., 2017] have also tried to solve separately these problems. [Kim et al., 2017] proposes to estimate the reflectance (diffuse and specular components) of a rigid object from an image sequence and their associated depth-maps (given by the Kinect for instance). [Mandl et al., 2017] proposes to estimate the scene illumination from a single image where a known rigid object is visible. It would be interesting to see if such approaches can be generalized to deformable objects.

Studying how to combine mathematical models used in this thesis and presented in the state-of-the-art with learning-based approaches is an important direction, since it can allow one to overcome some limitations of mathematical models and to use the data more efficiently with these mathematical models.

Automating the construction of the boundariness map. The boundary contour constraint proposed in this thesis requires a boundariness map. We have proposed an enhanced version where the false edges are suppressed thanks to statistical models. However, in our experiments on *NRSfMS-1*, we have set manually the parameters to compute the enhanced boundariness map for each dataset. It would be interesting to automate the parameter tuning.

Suppressing strong gradients from the surface texture is also an important problem to solve in order to improve the convergence of our boundary contour constraint. As the construction of the enhanced boundariness maps requires the segmentation of the surface on the input images, other segmentation methods can be explored. In SfT, as the object to segment in the input image is known *a priori* thanks to the template, the segmentation methods which can integrate this knowledge can be very appropriate solutions. Solving jointly the problem of recognition and segmentation has already shown improvements of results for both problems [He et al., 2017]. In NRSfM, methods of multi-view segmentation [Wang and Collomosse, 2012] or co-segmentation [Vicente et al., 2011] should be tested.

Extension of *NRSfMS-1* to the incremental approach. In our experiments on *NRSfMS-1*, we used sets of 5 images. The use of larger sets of images would improve the reconstruction accuracy of the 3D shapes in the different images and the surface albedos. This may come from having more image information (or even using temporal continuity for video inputs), which would improve the geometry of the surface and then its registration over the images, which would help for a better estimation of the surface albedos. Therefore, it would give a more accurate template of the surface. An incremental approach which integrates new images by mini-batches of images would allow one to use more image information and obtain better estimates without increasing the computational cost of our method. This could also lead to real-time versions.

Appendices

Appendix A

Hyperparameters for *Curve SfT-2* and *Curve SfT-1* Experiments

		Category *(iv)* method §3.4.2			Category *(iii)* method §3.4.3	
	Datasets	Smoothing parameter for η	Smoothing parameter for ξ	Number of nodes M	Polynomial order N_α	Smoothing weight λ_{smooth}
Curve SfT-2	*convex-to-concave*	9e−6	1e−7	30	12	3e−4
	free-form	9e−8	1e−7	30	12	3e−4
	paper	3e−5	6e−13	30	12	3e−4
	cable	9e−4	6e−13	30	12	3e−4

Table A.1: Hyperparameter values for different category methods to solve *Curve SfT-2* for all datasets.

		Category *(iv)* method §3.4.2			Category *(iii)* method §3.4.3	
	Datasets	Smoothing parameter for η	Smoothing parameter for ξ	Number of nodes M	Polynomial order N_β and N_γ	Smoothing weight λ_{smooth}
Curve SfT-1	*3D cord*	9e−6	1e−7	30	$12, 12$	3e−4
	necklace	9e−4	3e−13	30	$12, 12$	0.03
	road	3e−5	6e−13	30	$12, 12$	3e−4

Table A.2: Hyperparameter values for different category methods to solve *Curve SfT-1* for all datasets.

Appendix B

Hyperparameters for SfT Experiments for Creasable Surfaces

Ga16a*	**Ba15a**	**Ch17a**	**Sa09a**	**Ng16a**
M = 1e4 $N_{\mathcal{B}}$ = 1e3 $\lambda_{contour}$ = 4e−4 λ_{iso} = 0.16 λ_{smooth} = 8e−13	phi.nC = 100 phi.er = 0.001 eta.nC = 50 eta.er = 0.01 delta.nC = 20 delta.er = 0.01		No parameter	cst_weight = 200.00 mu = 0.016

Table B.1: Hyperparameter values used to evaluate all SfT methods compared.

Appendix C

Hyperparameters for SfT Experiments for Poorly-Textured Surfaces

Ga16b*
$M = 1e4$
$N_{\mathcal{B}} = 1e3$
$k_{shade} = 0.005$
$\lambda_{motion} = 2.5e6$
$\lambda_{contour} = 1e3$
$\lambda_{iso} = 4e5$
$\lambda_{smooth} = 1e{-}8$

Table C.1: Hyperparameter values used to evaluate our four methods, **Ga16b_S4K***, **Ga16b_S4U***, **Ga16b_S9K*** and **Ga16b_S9U***.

Ga16b_S4U* and **Ga16b_S9U***
$k = 20,000$
$\mathcal{C}o = 0.5$
$\tau = 0.04$

Table C.2: Hyperparameter values used for the illumination initialization of **Ga16b_S4U*** and **Ga16b_S9U***.

Appendix D

Hyperparameters for NRSfM Experiments

		floral paper	*paper fortune teller*	*creased paper*	*pillow cover*	*hand bag*	*Kinect paper*
Ga16a*	M	1e4					
	$N_{\mathcal{B}}$	1e3					
	$\lambda_{contour}$	1e−5	4e−4	4e−4	4e−4	4e−4	0.04
	λ_{iso}	4e−4	0.16	4e−3	4e−3	0.04	0.04
	λ_{smooth}	6e−15	2.4e−13	1.6e−14	1.6e−14	1.6e−14	4e−13
Ga17a*	M	1e4					
	$N_{\mathcal{B}}$	1e3					
	k_{shade}	5e−3	5e−3	5e−3	5e−3	5e−3	5e−3
	λ_{motion}	0.088	0.154	1	10	10	10
	$\lambda_{contour}$	1.25e−4	0.011	0.01	1.67e−4	1.67e−4	1.67e−4
	λ_{iso}	3.8e−3	0.025	0.167	0.167	0.167	0.5
	λ_{smooth}	2.5e−12	9.2e−12	3.33e−11	3.33e−11	3.33e−11	8.33e−10

Table D.1: Hyperparameter values used to evaluate our *NRSfMS-1* method.

		floral paper	*paper fortune teller*	*creased paper*	*pillow cover*	*hand bag*	*Kinect paper*
To08a	use_lds	1					
	max_en_iter	60					
	tol	1e−5					
	K	4					
Va09a	depth.nC	30	30	30	28	30	30
	depth.er	6	0.06	0.2	8	0.2	6
	embedding.nC	30					
	embedding.er	0.01	1e−6	1e−6	1e−4	1e−6	0.01
	homographies.neigh	100					
Ta10a	No parameter						
Vi12a	No parameter						
Ch14a	depth.nC	28	30	28	16	28	30
	depth.er	5	1	0.7	1	8	0.9
	warps.nC	28	20	28	16	28	30
	warps.er	0.01	1e−3	9e−4	1e−4	1e−3	0.01
	homographies.neigh	40	40	40	80	40	40
Ch16a	neighborhood_size	20					
Pa16a	schwarzianParam	2e−5					1e−3
	warps.nC	60					
	warps.er	1e−4					
	depth.nC	100					
	depth.er	1					10

Table D.2: Hyperparameter values used to evaluate all compared NRSfM methods.

Appendix E

Computing Enhanced Boundariness Maps for NRSfM Experiments

In this appendix, we provide supplementary information about the enhanced boundariness map computation. It is used in the boundary contour term which encourages the boundary of the surface to project to strong intensity edges in the image. This is a very useful constraint for poorly-textured surfaces. However, boundary may be attracted by false surface edges, which correspond to background clutter or surface texture. The segmentation cue to remove such false positives from the boundariness map depends on the particular dataset. Depending on the dataset, we use two segmentation cues: the image projection of the estimated surface and the color information of the image.

Projection-based segmentation. This segmentation cue handles datasets where the object surface has the same color distribution as the background. This is the case of the *floral paper*, *paper fortune teller* and *Kinect paper* datasets. For this case, we use simple morphological operations to create a rough boundary mask and propose a three-levels image pyramid to improve the convergence of the boundary contour constraint.

Firstly, we use the projection of the current estimated surface on the image plane. We erode the region defined by the convex hull of this projection using a square of 40×40 pixels, which gives a rough mask of the foreground R_t^{fg}. We also dilate the region defined by the convex hull of this projection using a square of 90×90 pixels and compute its complementary, which gives a rough mask of the background R_t^{bg}. Then, we define as R_t the complementary of the union of R_t^{fg} and R_t^{bg}. R_t corresponds to the rough boundary mask, where each pixel belonging to a neighbor of 100 pixels of the true surface edges is set to 1.

Secondly, we use a three-levels image pyramid which gives coarse-to-fine versions of the boundariness map and which increases the convergence basin. We define them as following:

- for the first level, we apply a Gaussian filter with the parameters $(H_1 = 120, \Sigma_1 = 20)$ on the boundary mask R_t to create $G(R_t; H_1, \Sigma_1)$ and compute the enhanced boundariness

map as:

$$B_t = \exp\left(-\frac{G(R_t; H_1, \Sigma_1)}{s}\right), \tag{E.1}$$

with the bandwidth of the potential well $s = 0.5$. This map allows one to attract strongly edges which may be quite far from their true locations.

- for the second level, we apply a Gaussian filter with the parameters ($H_2 = 40, \Sigma_2 = 10$) on the boundary mask R_t to create $G(R_t; H_2, \Sigma_2)$ and compute the enhanced boundariness map by filtering the baseline boundariness map (left term) given in §4.2.5.2 as:

$$B_t = \exp\left(-\frac{G(R_t; H_2, \Sigma_2)}{s}\right) \exp\left(-\frac{|\nabla G(I_t; h_2, \sigma_2)|}{s}\right), \tag{E.2}$$

with $h_2 = 5$, $\sigma_2 = 2.5$ and $s = 0.5$. This map creates a potential well using the image edges weighted by the boundary mask.

- for the last level, we do not use any segmentation cue and thus use the whole gradient image:

$$B_t = \exp\left(-\frac{|\nabla G(I_t; h_3, \sigma_3)|}{s}\right), \tag{E.3}$$

with $h_3 = 1$, $\sigma_3 = 1$ and $s = 0.5$. This is a reasonable approach since we can assume that the boundaries should be close to their true locations after the two first levels. Another advantage is that small color changes between the background and the surface can used.

Color-based segmentation. Color information can be used for datasets where the color distribution of the object surface is different from the one of the background. We use this segmentation cue for the *creased paper*, the *pillow cover* and the *hand bag* datasets. This cue allows us to create a fine boundary mask, F_t, where pixels close to the true locations of surface boundaries are set to 1. In our experiments, these pixels belong to a neighbor of 30 pixels.

For this, we create for each image a color-based foreground and background detectors which are trained respectively on the estimated foreground and background regions. We use an RGB Gaussian Mixture Model of 4 components to train both detectors. The estimated foreground and background regions are computed in the same way as for the projection-based segmentation and thus correspond to R_t^{fg} and R_t^{bg}. We define two masks M_t^{fg} and M_t^{bg} which respectively tell which pixels belong to the foreground and the background. We create both masks in three steps. Firstly, the foreground (respectively the background) detector set $M_t^{fg} = 1$ (respectively $M_t^{bg} = 1$) for any pixel which has a detector score below a threshold T_t^{fg} (respectively T_t^{bg}). We found that setting T_t^{fg} (respectively T_t^{bg}) as the 95^{th} percentile of the detector scores in the region defined by R_t^{fg} (respectively R_t^{bg}) achieves a good segmentation of the foreground (respectively the background). Secondly, we apply on M_t^{fg} and M_t^{bg} a closure with a square 5×5 pixels to remove false negative pixels from the

detectors. Thirdly, we apply on M_t^{fg} and M_t^{bg} an erosion with a square 30×30 pixels and then compute the intersection of their union, which achieves to a fine boundary mask, F_t.

Similarly to the first segmentation cue, we use a three-levels image pyramid:

- for the first level, we apply a Gaussian filter with the parameters $(h_1 = 10, \sigma_1 = 5)$ on the image I_t and we compute the enhanced boundariness map as:

$$B_t = \exp\left(-\frac{|\nabla G(I_t; h_1, \sigma_1)|}{s}\right) F_t, \tag{E.4}$$

  with $h_1 = 10$, $\sigma_1 = 5$ and $s = 0.5$.

- for the second level, we compute the enhanced boundariness map as:

$$B_t = \exp\left(-\frac{|\nabla G(I_t; h_2, \sigma_2)|}{s}\right) F_t, \tag{E.5}$$

  with $h = 5$, $\sigma = 2.5$ and $s = 0.5$.

- for the last level, we use the baseline boundariness map for similar reason as in the projection-based segmentation:

$$B_t = \exp\left(-\frac{|\nabla G(I_t; h_3, \sigma_3)|}{s}\right), \tag{E.6}$$

  with $h_3 = 1$, $\sigma_3 = 1$ and $s = 0.5$.

Appendix F

Résumé des travaux

Ce chapitre présente un résumé de mes travaux de thèse, qui portent sur la reconstruction 3D déformable monoculaire. Nous commençons par une description du contexte scientifique et une vue globale des principales approches de reconstruction 3D déformable monoculaire et de recalage. Nous en présentons ensuite quelques applications pour en souligner l'intérêt. Nous expliquons ensuite les principales limitations de deux des principales approches et présentons nos contributions pour ces deux approches : l'étude des modèles curvilinéaires et l'utilisation d'indices visuels multiples pour des modèles surfaçiques. Ce chapitre s'appuie sur l'introduction du manuscrit et quelques éléments de l'état de l'art, des méthodes proposées, des résultats expérimentaux obtenus et des conclusions des différents chapitres du manuscrit.

F.1 Vision par ordinateur, reconstruction 3D et recalage

La discipline de la vision par ordinateur s'intéresse aux théories et systèmes pour extraire de l'information à partir d'images et de vidéos, pour en déduire une compréhension de haut niveau. Les problèmes fondamentaux de la vision par ordinateur peuvent être regroupés par les "3R" : Reconnaissance, Recalage et Reconstruction. Le domaine de la reconnaissance cherche à déterminer quels objets, activités ou événements sont présents dans l'image. Le domaine du recalage cherche à déterminer quels points de plusieurs images correspondent à un même point physique, que ce soit des images de même modalité ou non. Le domaine de la reconstruction 3D cherche à retrouver la forme 3D d'un objet à partir d'une ou plusieurs images. Cette thèse se concentre sur la résolution des problèmes de recalage et de reconstruction 3D pour des objets déformables.

Les méthodes de reconstruction 3D se distinguent généralement par six composantes : le système d'acquisition (capteur actif ou passif), le fait que l'objet ou la scène soit rigide ou déformable, le type d'information image utilisé (aussi appelé indice visuel), le type d'*a priori* utilisé, la modélisation et la solution. La reconstruction 3D pour des objets rigides a été étudiée en profondeur. La géométrie multi-vues avec des objets rigides est bien comprise [Hartley and Zisserman, 2003] et a permis l'émergence et la maturation de techniques telles que le Structure-from-Motion (SfM) [Hartley and Zisserman, 2003] et la Stéréo Multi-Vues (SMV) [Furukawa and Ponce, 2010]. Plusieurs solutions ont même été commercialisées [3Dflow, 2017; Agisoft, 2014].

Cependant, le recalage et la reconstruction 3D d'objets déformables restent des problèmes ouverts. Fondamentalement, ces problèmes sont bien moins contraints que le cas rigide. Relever ces défis est important, étant donné que de nombreux objets d'intérêt sont déformables, tels que les visages, les corps, les organes et les tissus. De bonnes solutions auraient leur place dans de nombreuses applications dans le divertissement, l'imagerie médicale et la mécanique, notamment. Un autre point important est que la reconstruction 3D déformable manque actuellement d'une compréhension théorique profonde. L'extension de la géométrie multi-vues pour les objets rigides aux objets déformables n'est pas triviale et les avancées sur ce sujet sont encore limitées.

Le récent développement de capteurs temps-réel de profondeur à faible coût, comme la Kinect, a facilité de nombreuses applications importantes pour les objets déformables. Toutefois, résoudre le problème de la reconstruction 3D déformable à l'aide de méthodes monoculaires passives reste un problème important et pertinent. Cela vient des limitations inhérentes des capteurs de profondeur : ils ont une plage d'acquisition limitée (ils ne peuvent pas obtenir la profondeur lorsque l'objet est trop loin ou trop près du capteur), une plus forte consommation électrique que les caméras RVB, et sont souvent sensibles aux conditions d'éclairage extérieur. Il peut y avoir également des exigences physiques qui empêchent leur utilisation pour des applications spécifiques, telles que l'imagerie endoscopique. Enfin, des millions de caméras RVB sont quotidiennement utilisés sur des appareils mobiles, ce qui laisse entrevoir un énorme potentiel d'utilisation et de commercialisation et souligne le besoin de

résoudre le problème du recalage et de la reconstruction 3D déformable monoculaire.

F.2 Les principaux paradigmes de la reconstruction 3D déformable

Quatre paradigmes approchent le problème de la reconstruction 3D déformable à partir d'images monoculaires : le Shape-from-Template (SfT), le Non-Rigid Structure-from-Motion (NRSfM), le Shape-from-Shading (SfS) et les méthodes de reconstruction 3D monoculaire basées apprentissage. La figure F.1 représente les principales différences entre ces paradigmes, en spécifiant leurs entrées et sorties. Nous donnons maintenant une vue globale de ces quatre paradigmes.

F.2.1 Définitions préalables

Comme la notion de pli est fondamentale dans notre travail, nous donnons ici sa définition précise et deux termes associés.

Définition 1 (*Pli*)**.** *Nous définissons un pli par une discontinuité de la première dérivée de la surface.*

Définition 2 (*Surface pliable*)**.** *Nous définissons une surface pliable par une surface qui est susceptible de faire des plis.*

Définition 3 (*Surface pliée*)**.** *Nous définissons une surface pliée par une surface qui présente au moins un pli.*

F.2.2 Shape-from-Template

L'objectif du SfT est de reconstruire la forme 3D d'un objet déformé en utilisant une seule image et un modèle 3D texturé de l'objet dans une position de référence [Bartoli et al., 2015; Salzmann and Fua, 2011]. Le SfT fonctionne en alignant l'objet sur l'image d'entrée et en déformant le patron de l'objet : le SfT réalise simultanément un recalage et une reconstruction 3D dense. Des premiers travaux de [Gumerov et al., 2004; Perriollat et al., 2008; Salzmann et al., 2007a] aux méthodes plus récentes fonctionnant en temps-réel [Collins and Bartoli, 2015; Ngo et al., 2016], le SfT est un des paradigmes qui s'est développé le plus rapidement. Les méthodes de SfT sont aussi appelées méthodes de reconstruction 3D basées patron ou basées modèle. La figure F.1 (ligne n°1) fournit une description générale du SfT. Un patron peut être décomposé en un modèle de forme, un modèle d'apparence avec une carte de texture et un modèle de déformation. Le modèle de forme est un modèle 3D d'un objet dans une position de référence connue, qui peut varier selon l'application. Par exemple, le patron peut être généré à partir d'un modèle CAO (Conception Assistée par Ordinateur) de l'objet, ou reconstruit à partir de données grâce à des méthodes comme le dense SfM avec un ensemble de vues rigides de l'objet. Une majorité des méthodes de SfT sont nommées méthodes de

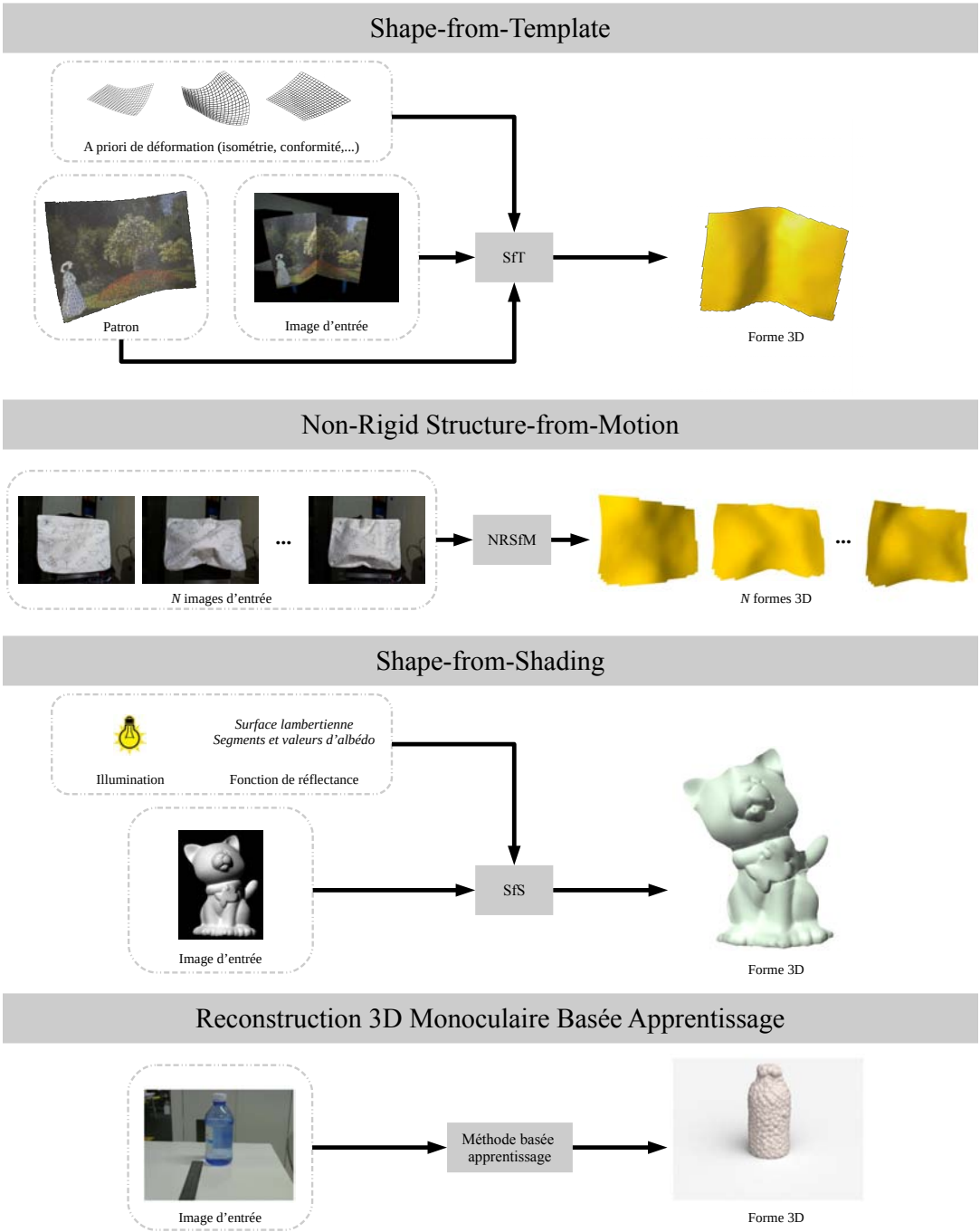


Figure F.1: Quatre techniques de reconstruction 3D déformable : SfT (ligne n°1), NRSfM (ligne n°2), SfS (ligne n°3) et reconstruction 3D monoculaire basée apprentissage (ligne n°4). **Ligne n°1 :** la méthode de SfT utilisée ici est [Chhatkuli et al., 2017], donnant le meilleur résultat pour le jeu de donnée montré. **Ligne n°2 :** la méthode de NRSfM utilisée ici est [Chhatkuli et al., 2014], donnant le meilleur résultat pour le jeu de donnée montré. **Ligne n°3 :** la méthode de SfS utilisée ici est [Xiong et al., 2015]. Il s'agit d'une méthode de l'état de l'art. Résultat fourni par [Xiong et al., 2015]. **Ligne n°4 :** la méthode de reconstruction 3D monoculaire basée apprentissage utilisée ici est [Fan et al., 2017]. Il s'agit d'une méthode de l'état de l'art. Résultat fourni par [Fan et al., 2017].

SfT surfaçique puisque le patron est un modèle à coque mince, sans volume. Il existe aussi des méthodes de SfT volumique [Collins and Bartoli, 2015; Parashar et al., 2015]. Celles-ci modélisent l'objet par un modèle de déformation volumique, en utilisant soit des modèles continus, tels que les splines 3D [Parashar et al., 2015], soit des modèles discrets, tels que les maillages tétrahédraux [Collins and Bartoli, 2015]. Cependant, le développement du SfT pour des patrons curvilinéaires n'a pas encore été rapporté dans la littérature. Nous introduisons le cas spécial du SfT curvilinéaire et révélons son utilité pratique et sa complexité cachée.

La plupart des méthodes de SfT utilisent le mouvement apparent de l'objet (aussi appelé indices de mouvement). En pratique, ce mouvement s'observe à l'aide de correspondances de primitives image, telles que Scale-Invariant Feature Transform (SIFT) [Lowe, 2004] ou Speeded Up Robust Features (SURF) [Bay et al., 2008], pour construire des correspondances parcimonieuses entre le patron et l'image d'entrée. Des correspondances denses telles que celles données par [Collins and Bartoli, 2015; Malti et al., 2011; Yu et al., 2015] peuvent aussi être utilisées. Cependant, les indices de mouvement sont souvent insuffisants pour déduire la forme 3D de l'objet déformé, puisqu'une image peut être générée par un ensemble de déformations potentiellement infini, ce qui forme l'ambiguïté de profondeur. Pour rendre le SfT, et plus globalement, tous les problèmes de reconstruction 3D déformable bien contraint, des *a priori* de déformation sont nécessaires. Cette idée se retrouve en reconstruction 3D rigide, où l'*a priori* de rigidité est utilisé. L'*a priori* de déformation le plus étudié est l'isométrie : elle signifie que les distances sur la surface de l'objet sont préservées durant la déformation. Le succès de l'isométrie vient du fait qu'elle est simple à modéliser mathématiquement et qu'elle approxime bien le comportement de nombreux objets soumis à des déformations quasi-isométriques, tels que le tissu ou le carton. De plus, il a été montré que, pour de telles déformations, le SfT surfaçique est bien posé en ce sens qu'il a une solution unique [Bartoli et al., 2015; Chhatkuli et al., 2017] pour des correspondances denses. D'autres *a priori* de déformation ont été utilisés tels que la conformité (conservation des angles) [Bartoli et al., 2015] et l'élasticité [Haouchine et al., 2014; Malti et al., 2015].

Plusieurs méthodes de SfT existantes fournissent des solutions stables et précises, mais uniquement pour des surfaces texturées soumises à des déformations lisses. Ces méthodes peuvent échouer dans deux cas : lorsque l'objet est peu texturé ou lorsqu'il se déforme de manière non lisse. Au niveau des régions peu texturées, les primitives image sont très éparses et peu fiables, les correspondances denses ne peuvent pas être calculées correctement. Fondamentalement, le mouvement est insuffisant dans ces régions pour reconstruire la surface déformée ou recaler précisément le patron. L'incapacité de ces méthodes pour reconstruire les déformations non lisses s'explique par deux raisons. La première raison est que l'information de mouvement n'est généralement pas suffisante pour déduire avec précision les plis dans les régions peu texturées. La deuxième raison est que la plupart des méthodes modélisent des déformations de faible dimensionnalité en utilisant des modes de déformations [Ngo et al., 2016; Salzmann and Fua, 2011] ou réduisent sa dimensionnalité en utilisant des *a priori* de lissage basés sur la norme ℓ_2, ce qui est fait dans presque toutes les méthodes existantes pour réduire la dimensionnalité du problème et fournir une régularisation suffisante. Cependant,

le problème est que cet *a priori* produit des solutions lisses et non discontinues. La figure F.1 (ligne n°1) illustre l'incapacité de la méthode de l'état de l'art [Chhatkuli et al., 2017] à reconstruire le pli au milieu de la feuille de papier. Des méthodes ont abordé le problème de reconstruction 3D de plis [Salzmann and Fua, 2009] ou de surfaces peu texturées [Liu-Yin et al., 2016; Malti and Bartoli, 2014; Ngo et al., 2015; Varol et al., 2012b]. La méthode de [Salzmann and Fua, 2009] propose une formulation convexe du SfT, qui maximise la profondeur de chaque sommet d'un maillage et relâche la contrainte d'isométrie.

Toutefois, les plis ne sont pas modélisés explicitement dans [Salzmann and Fua, 2009], une calibration photométrique complète pour utiliser l'ombrage est requise dans [Varol et al., 2012b], la méthode de [Ngo et al., 2015] nécessite un prérecalage du patron sur la première image, et [Liu-Yin et al., 2016; Malti and Bartoli, 2014] ne fonctionnent que pour des vidéos débutant par une séquence d'images où l'objet est supposé rigide et qui est utilisée pour créer le patron.

F.2.3 Non-Rigid Structure-from-Motion

L'objectif du NRSfM est de reconstruire les formes 3D d'un objet déformable en utilisant un ensemble d'images monoculaires. Contrairement au SfM, le NRSfM considère un objet qui peut se déformer entre chaque image [Bregler et al., 2000]. Les méthodes de NRSfM sont aussi appelées méthodes de reconstruction 3D sans patron ou sans modèle. La figure F.1 (ligne n°2) illustre les différents composants du NRSfM. Ce problème est bien plus difficile que le SfT puisqu'aucun patron n'est disponible et que par conséquent la structure physique de l'objet est inconnue. Le NRSfM considère un ensemble d'images monoculaires au lieu d'une seule image, comme dans le SfT. La difficulté accrue du NRSfM explique pourquoi les méthodes de SfT ont évolué plus rapidement que les méthodes de NRSfM et pourquoi aucune implémentation temps-réel de NRSfM n'a été précédemment proposée.

Comme pour le SfT, la plupart des méthodes existantes de NRSfM utilisent le mouvement apparent, calculé à partir de correspondances de primitives images ou du flux optique [Garg et al., 2013]. Ils intègrent nécessairement des *a priori* de déformation pour rendre le NRSfM soluble et peuvent être alors séparés en deux catégories : les méthodes statistiques [Akhter et al., 2009; Bregler et al., 2000; Dai et al., 2014; Garg et al., 2013; Gotardo and Martinez, 2011; Torresani et al., 2008a] et les méthodes physiques [Agudo et al., 2016; Chhatkuli et al., 2014, 2016; Varol et al., 2009; Vicente and Agapito, 2012; Wang et al., 2016]. Les méthodes statistiques utilisent une réduction de dimensionnalité et ne modélisent donc pas explicitement le comportement physique de l'objet. Des *a priori* physiques sont l'isométrie [Chhatkuli et al., 2014; Collins and Bartoli, 2010; Taylor et al., 2010; Varol et al., 2009; Vicente and Agapito, 2012], l'inextensibilité [Chhatkuli et al., 2016] et l'élasticité linéaire [Agudo et al., 2016]. C'est parce que ces méthodes modélisent le comportement réel de l'objet qu'elles aboutissent généralement à un problème mieux contraint et à des résultats plus précis que ceux obtenus par des méthodes statistiques. Cependant, une majorité des méthodes existantes de NRSfM souffrent des mêmes limitations que les méthodes de SfT : elles ne parviennent

pas à reconstruire des objets dont la surface est peu texturée ou qui se déforment de manière non lisse. La figure F.1 (ligne n°2) présente une reconstruction 3D obtenue par une méthode de l'état de l'art [Chhatkuli et al., 2014], qui ne peut pas reconstruire des déformations complexes comme les plis. En lien avec une de nos contributions, [Wang et al., 2016] a proposé une méthode qui gèrent les surfaces peu texturées, mais celle-ci ne reconstruit que les surfaces très lisses.

F.2.4 Shape-from-Shading

L'objectif du SfS est d'estimer la profondeur de la surface en chaque pixel d'une seule image en utilisant l'information d'ombrage. Deux points importants sont que le SfS ne possède pas de patron de la surface *a priori* et ne réalise aucun recalage. Le SfS utilise exclusivement l'information d'ombrage à travers la relation photométrique entre la surface de l'objet, la réflectance du matériau, l'illumination de la scène et l'intensité du pixel [Horn, 1970; Zhang, 1997]. La réflectance de l'objet explique comment la lumière est reflétée par sa surface. La figure F.1 (ligne n°3) montre les différentes composantes du SfS. À la différence du mouvement utilisé dans le SfT et le NRSfM, l'ombrage est l'indice visuel le plus important pour déduire des détails de haute-fréquence d'une forme 3D au niveau des régions peu texturées [Pentland, 1988], comme le montre la figure F.1 (ligne n°3).

Cependant, le SfS est un problème fondamentalement mal posé [Belhumeur et al., 1997; Pentland, 1984] et requiert ainsi une calibration photométrique de la scène *a priori*. Une calibration photométrique implique de connaître la réflectance de la surface (principalement diffuse et spéculaire) et l'illumination. La non-unicité de la solution au SfS est souvent présentée à travers l'ambiguïté concave/convexe [Belhumeur et al., 1997]. Tout cela fait du SfS une méthode peu réaliste. Cela explique pourquoi la plupart des méthodes de SfS travaillent avec des environnements très contrôlés (illumination et réflectance connues) et/ou données simulées.

La majorité des méthodes de SfS utilisent le modèle de caméra orthographique, mais certains travaux ont aussi étudié le modèle perspectif [Prados and Faugeras, 2005]. Plusieurs modèles d'illumination et de réflectance existent. Le modèle d'illumination distant sans et avec un terme ambiant et le modèle de réflectance lambertienne (diffus) sont les modèles les plus utilisés. Toutefois, des travaux se sont déjà intéressés à des modèles d'illumination ponctuelle [Wu et al., 2007] ou des modèles de réflectance plus complexes comme Oren-Nayar [Ahmed and Farag, 2007].

Les méthodes de SfS différent principalement par leur approche de résolution du problème. Cinq catégories de méthode existent : *(i)* les approches par propagation [Ahmed and Farag, 2007; Kimmel and Bruckstein, 1994; Prados and Faugeras, 2005; Rouy and Tourin, 1992], *(ii)* les approches locales [Pentland, 1984; Xiong et al., 2015], *(iii)* les approches linéaires [Pentland, 1988; Tsai and Shah, 1994], *(iv)* les approches par minimisation [Barron and Malik, 2015; Ikeuchi and Horn, 1981; Lee and Kuo, 1993] et *(v)* les approches basées apprentissage [Richter and Roth, 2015]. En général, les méthodes des catégories *(iv)* et *(v)* sont les

plus robustes, alors que les autres sont plus rapides.

F.2.5 Reconstruction 3D monoculaire basée apprentissage

L'objectif des méthodes basées apprentissage est de prédire la forme 3D d'un objet ou une carte de profondeur de la scène à partir d'une seule image à l'aide d'une base de donnée d'apprentissage. De même que les méthodes de SfS, ces méthodes ne réalisent aucun recalage. Ce paradigme définit le problème de reconstruction 3D monoculaire déformable comme un problème d'apprentissage supervisé. Dans cette catégorie, un premier travail [Saxena et al., 2009] a proposé d'apprendre à prédire des cartes de profondeur en utilisant des modèles graphiques. Cependant, le récent développement des réseaux de neurones profonds a fait apparaître un grand nombre de méthodes utilisant des réseaux de neurones profonds [Choy et al., 2016; Fan et al., 2017; Godard et al., 2017; Kar et al., 2015; Laina et al., 2016; Richardson et al., 2016]. Ces méthodes ont montré qu'elles pouvaient être utilisées pour des classes d'objets usuels pour lesquelles de très grandes bases de données sont disponibles. Des exemples sont les voitures ou les meubles [Choy et al., 2016; Fan et al., 2017; Kar et al., 2015], des scènes d'intérieures particulières [Godard et al., 2017; Laina et al., 2016] telles que des chambres, des bureaux et des routes, ou des visages [Richardson et al., 2016] qui utilise des modèles de déformation à faible dimensionnalité. Cette catégorie n'a pas montré de reconstruction 3D d'objets soumis à des déformations de haute dimensionnalité, ce qui limite considérablement son applicabilité. Deux autres inconvénients sont à noter : les bases de données d'apprentissage et de test doivent être relativement semblables et l'analyse du caractère bien posé et des ambiguïtés pour les réseaux de neurones profonds est très difficile. En effet, pour l'instant, aucune réponse à ce sujet n'a été proposée, alors qu'une telle analyse est importante pour diagnostiquer lorsqu'un problème ne peut pas être résolu. La figure F.1 (ligne n°4) donne un exemple de reconstruction 3D fournie par [Fan et al., 2017].

F.2.6 Reconstruction 3D et recalage à l'aide de plusieurs indices visuels, et convention de nommage

Une majorité des méthodes de SfT et de NRSfM utilisent le mouvement comme principal indice visuel. Cependant, certaines méthodes fonctionnent en combinant le mouvement avec d'autres indices visuels pour gérer les scénarios difficiles comme les surfaces peu texturées. D'autres méthodes diffèrent aussi par les paramètres additionnels qu'elles estiment, tels que la réflectance de surface. Par souci de clarté, nous complétons les deux principaux paradigmes, SfT et NRSfM, par le suffixe "S" lorsque l'ombrage est utilisé. En utilisant cette notation, nous proposons dans le tableau F.1 une définition systématique des différents problèmes de SfT et de NRSfM Pour chaque abréviation, nous donnons quelques travaux existants.

Nous appelons bords la silhouette d'un objet 3D de topologie plate. Dans la littérature du SfT, les principaux indices visuels complémentaires sont les bords [Salzmann et al., 2007b; Vicente and Agapito, 2013], et l'ombrage [Liu-Yin et al., 2016; Malti and Bartoli, 2014], qui sont désignés comme des méthodes de Shape-from-Template-and-Shading (SfTS) qui est le

Abréviation	Description	Références
SfT	SfT à l'aide du mouvement	[Salzmann and Fua, 2009] [Bartoli et al., 2015]
SfTS	Shape-from-Template-and-Shading à l'aide du mouvement et de l'ombrage	[Malti and Bartoli, 2014] [Liu-Yin et al., 2016]
NRSfM	NRSfM à l'aide du mouvement	[Bregler et al., 2000]
NRSfMS	Non-Rigid Structure-from-Motion-and-Shading à l'aide du mouvement et de l'ombrage	×

Table F.1: Définition des différents problèmes de SfT et de NRSfM.

sujet d'une de nos contributions.

Dans la littérature du NRSfM, un seul travail combine le NRSfM avec des contraintes de bord [Wang et al., 2016]. Il utilise une contrainte de bord qui réduit la distance sur l'image d'entrée entre les bords de la surface projetés et les bords de l'image, qui dont détectés par un filtre de Canny. [Wang et al., 2016] utilise une contrainte d'intensité constante qui fournit des correspondances denses. Il n'existe pas de travaux précédents pour le NRSfMS.

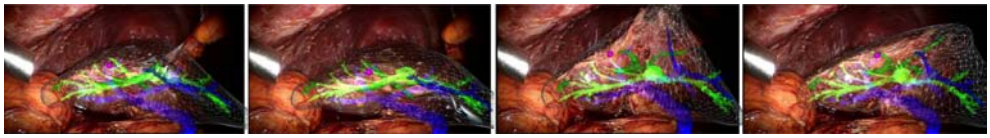
[Liu-Yin et al., 2016; Malti and Bartoli, 2014; Salzmann et al., 2007b; Vicente and Agapito, 2013] et [Wang et al., 2016] sont respectivement des instances spécifiques des problèmes généraux de SfT et de NRSfM. De nombreuses composantes du problème doivent être spécifiées lors de la définition d'un problème (SfT, SfTS, NRSfM ou NRSfMS) et ensuite lors de celle d'une instance particulière. Certains papiers ne proposent pas de descriptions précises et systématiques des composantes du problème. Nous présentons une caractérisation complète d'une instance de problème en terme de modèles, d'hypothèses sur la scène et de paramètres connus ou inconnus. Pour désigner une instance particulière du problème général SfT ou NRSfM, nous complétons SfT ou NRSfM par le suffixe '-Y', avec Y un entier positif et nous l'écrivons en *italique*. Nous résolvons des instances importantes du SfT, SfTS et NRSfMS. Nous les désignons à l'aide de la nomenclature *SfT-1*, *SfTS-1* et *NRSfMS-1*. Le *SfT-1* se caractérise principalement par la résolution du problème à l'aide de contraintes de mouvement et d'un patron qui peut se déformer d'une manière complexe et non-lisse à l'image de plis. Les solutions actuelles ne fournissent pas de résultats satisfaisants. Le *SfTS-1* se caractérise principalement par la résolution du problème où la réflectance de la surface et les réponses de caméra sont inconnues *a priori*. Cela n'a pas été réalisé auparavant et, contrairement au SfT, il nécessite de réaliser la reconstruction sur plusieurs images (quatre ou plus). Le *NRSfMS-1* se caractérise principalement par la résolution du problème où la réflectance de la surface est inconnue *a priori* et où l'objet subit une déformation différente sur chaque image. Cela n'a jamais été réalisé auparavant.

F.3 Motivation et applications du recalage et de la reconstruction 3D d'objets déformables

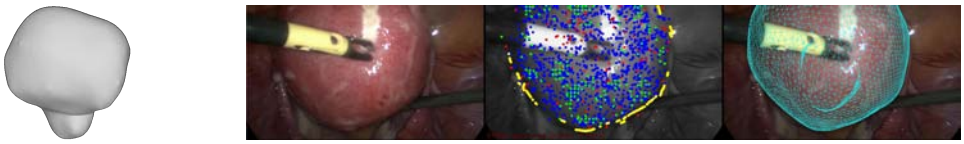
La recherche en recalage et reconstruction 3D déformable a suscité un intérêt notable pour de nombreuses applications, telles que l'imagerie médicale, la capture et l'édition d'expressions faciales, le transfert interactif de déformations, les jeux de Réalité Augmentée (RA) et l'analyse mécanique. Quelques exemples sont fournis en figure F.2.

Une application importante est la RA médicale avec la Chirurgie Minimalement Invasive (CMI) et plus précisément la chirurgie cœlioscopique. Il s'agit d'une technique avancée de chirurgie qui est réalisée en insérant par de petites incisions de fins instruments chirurgicaux et un cœlioscope (fin télescope comprenant à son extrémité un système d'éclairage et une lentille optique pour visualiser la cavité abdomino-pelvienne). Le chirurgien utilise le flux vidéo fourni par le cœlioscope pour réaliser l'opération. Les principaux avantages de la CMI sont que le traumatisme est fortement réduit et le temps de rétablissement du patient est raccourci. Cependant, lors d'une CMI, les chirurgiens sont confrontés à trois problèmes : les points de vues sont limités, la localisation en 3D et la perception en profondeur sont plus difficiles. La RA apparaît donc être une solution appropriée pour fournir un retour en temps-réel lors de la CMI. Cela est réalisé en augmentant en temps-réel le flux vidéo avec la projection des formes 3D des organes et de ses structures internes, obtenues à partir d'images pré-opératoires telles que des images IRM (Imagerie à Résonance Magnétique). Par exemple, une méthode de SfT a été proposée pour la chirurgie du foie [Haouchine et al., 2013]. Comme le montre la figure F.2, cette méthode aligne les images d'un cœlioscope stéréoscopique avec les tumeurs (en violet) et les structures internes du foie, telles que les veines hépatiques (en bleu) et les veines portes (en vert), en utilisant un patron 3D construit à l'aide d'examens tomodensitométriques (TDM). En utilisant un cœlioscope monoculaire, un recalage déformable d'un patron pré-opératoire d'un foie (obtenu à partir d'images TDM) a été présenté par [Koo et al., 2017]. Cela permet de recaler simultanément la tumeur (en vert) et les structures internes du fois telles que les veines (en bleu). Cela est réalisé en utilisant les indices visuels de silhouette et d'ombrage. La méthode temps-réel de SfT volumique de [Collins and Bartoli, 2015] a été appliquée dans [Collins et al., 2016] pour du suivi déformable d'organes, tels que le rein ou l'utérus, sur des vidéos de cœlioscopie monoculaire. Cela permet également d'actualiser à la demande la carte de texture de l'organe, ce qui est utile étant donné que la texture de l'organe peut changer durant l'opération.

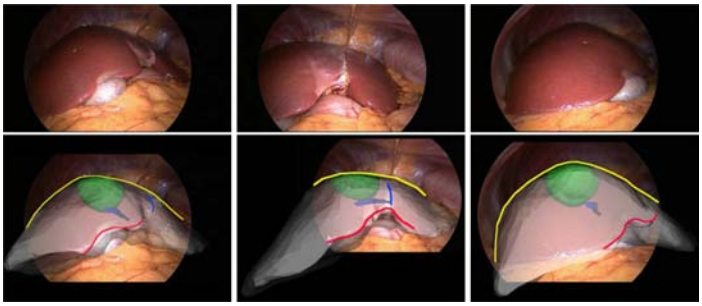
Le recalage et la reconstruction 3D d'objets déformables a également de multiples applications dans la post-production cinématographique. Les monteurs de film ont souvent besoin d'éditer les films après leur enregistrement, en supprimant, en ajoutant ou en modifiant du contenu. Lorsque le contenu est déformable, cela peut demander une très grande quantité de travail. Cependant, la plupart des films ne sont pas enregistrés à l'aide de capteurs de profondeur, ce qui rend alors les méthodes monoculaires très intéressantes. Un système temps-réel d'acquisition et d'édition d'expressions faciales a été proposé par [Thies et al., 2016]. Cela fonctionne en reconstruisant en 3D les visages d'un acteur source et d'un acteur



Applications à la RA médicale : l'augmentation temps-réel d'un réseau vasculaire et de tumeurs pour la CMI [Haouchine et al., 2013]



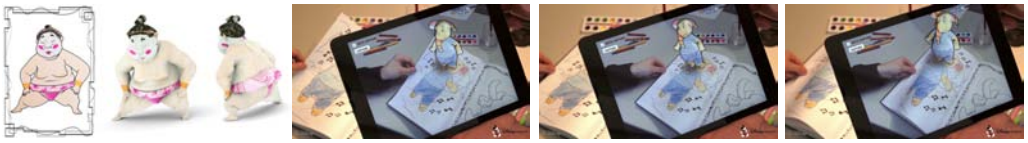
Applications à la RA médicale : suivi déformable temps-réel d'organes pour la CMI [Collins et al., 2016]



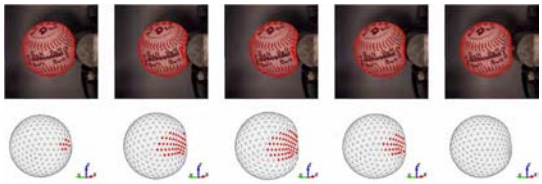
Applications à la RA médicale : recalage 3D déformable du foie et quelques structures internes pour la CMI [Koo et al., 2017]

Applications pour le transfert temps-réel d'expressions faciales : [Thies et al., 2016]

Applications pour le transfert temps-réel de déformations génériques : [Collins and Bartoli, 2015]



Applications pour les jeux RA : une application RA de livre de coloriage de [Magnenat et al., 2015]



Applications l'analyse de modèle mécanique : l'analyse d'impact d'une balle molle [Smith et al., 2016]

Figure F.2: Applications du recalage et de la reconstruction 3D déformable.

cible, et en transférant l'expression faciale l'acteur source vers l'acteur cible. Tout type de système d'acquisition peut être utilisé, y compris une simple caméra RVB. Plus généralement, une méthode temps-réel et interactive de transfert a été présentée par [Collins and Bartoli, 2015]. Celle-ci utilise une méthode temps-réel de SfT et fonctionne pour tout type d'objet avec un patron connu.

Le jeu en RA offre un large domaine d'application. L'idée est d'offrir aux joueurs de nouvelles expériences de jeu et un environnement de jeu différent étant donné que les jeux en RA utilisent l'environnement réel du joueur. Par exemple, une application RA de livre de coloriage a été présentée dans [Magnenat et al., 2015] : les enfants peuvent voir en 3D les personnages qu'ils ont coloriés sur un livre de coloriage, qui est déformable. L'application s'exécute à partir de tout type d'appareil possédant une caméra, telle qu'une tablette. Un algorithme de reconstruction 3D déformable est utilisé pour reconstruire en temps-réel la page du livre afin d'y afficher le personnage colorié en 3D en surimpression.

Un autre domaine d'application est l'analyse mécanique. Des données expérimentales spécifiques sont nécessaires pour analyser le comportement de corps mous et réduire l'écart entre les modèles de simulation et le comportement réel des matériaux. Pour certains produits, obtenir de telles données nécessite de procéder à un recalage et une reconstruction 3D déformable. C'est le cas d'une balle molle, comme le présente [Smith et al., 2016]. Ce travail utilise la méthode de SfT de [Ngo et al., 2016] pour reconstruire en 3D et aligner la surface d'une balle molle lors d'un impact. Celle permet d'évaluer la précision des algorithmes de simulation.

F.4 Déroulé de la thèse et contribution

Nous illustrons le déroulé de cette thèse en figure F.3. Nous avons avancé l'état de l'art dans quatre directions principales en considérant quatre de ses limitations fondamentales. Nous énumérons notre contributions et les détaillons ensuite.

1) Le SfT curvilinéaire. La littérature du SfT manque de solutions et de compréhensions théoriques du cas spécial d'un patron 1D, *c-à-d.* le cas où la forme est une courbe 1D plongée dans un espace 2D ou 3D.

2) Le SfT surfaçique pour des surfaces pliables. La plupart des méthodes existantes de SfT sont construites de telle manière qu'elles sont capables de reconstruire uniquement des surfaces lisses. Cependant, les surfaces non lisses, telles que des feuilles de papier pliées, sont très communes en pratique. Les limitations de ces méthodes sont d'une part l'insuffisance de la contrainte de mouvement pour déduire les zones d'apparition des plis et d'autre part l'utilisation de réduction de dimensionnalité et d'*a priori* de lissage qui limitent leur application à des objets se déformant de manière lisse.

3) Le SfT surfaçique pour des surfaces pliables et peu texturées. La plupart des méthodes existantes de SfT donnent de bons résultats pour des surfaces texturées et presque toutes utilisent des contraintes de correspondance basées sur des primitives images. Cependant, en pratique, de nombreux objets ont des surfaces texturées présentant également des régions

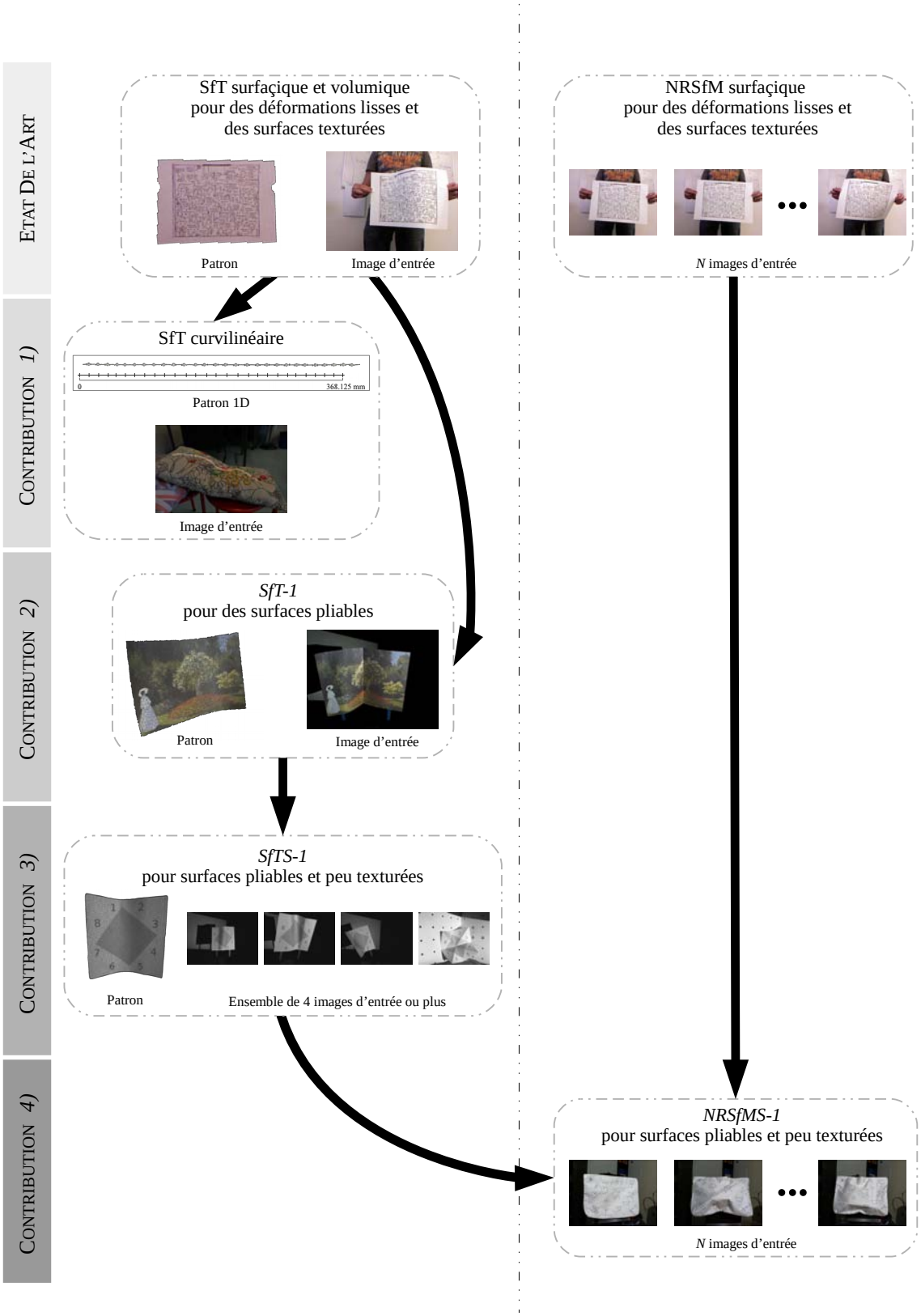


Figure F.3: Vue d'ensemble des principales contributions de cette thèse.

peu texturées qui sont non négligeables. Comme les correspondances (éparses ou denses) ne peuvent pas être facilement obtenues sur des surfaces peu texturées, d'autres indices visuels doivent être utilisés.

4) Le NRSfM surfaçique pour des surfaces pliables et peu texturées. Pour les mêmes raisons que le SfT, la plupart des méthodes existantes de NRSfM ne peuvent pas reconstruire de manière précise les surfaces peu texturées soumises à des déformations complexes. Résoudre ce problème change fondamentalement le paradigme de NRSfM puisque nous ne pouvons pas séparer le recalage de surface de sa reconstruction 3D et puisque nous devons introduire dans le problème des contraintes photométriques qui sont non-convexes et qui complexifient grandement la procédure d'optimisation.

Nous fournissons ci-dessous le détail de nos contributions à ses quatre limitations.

F.4.1 Contribution au SfT curvilinéaire

Nous proposons une étude théorique approfondie et des algorithmes de résolution du problème du SfT curvilinéaire. Dans cette thèse, nous considérons deux sous-cas du SfT curvilinéaire. Le premier cas correspond à un patron qui est une courbe plongée dans un espace 3D et observée par une caméra 2D. Un exemple pratique consiste en la reconstruction 3D d'un collier mince autour du cou d'une personne, lorsque le patron de ce collier est donné. Cet exemple est montré en figure F.4. Le second cas correspond au premier cas avec une caméra 1D. Cela peut être créé par exemple par une vue orthogonale à une surface plane. Comme tous les cas de SfT, le SfT curvilinéaire requiert l'utilisation d'*a priori* de déformation à cause de la perte d'information générée par la projection de la caméra. Nous utilisons l'*a priori* d'isométrie.

À première vue, l'utilisation d'un patron 1D semble rendre le SfT plus simple qu'avec des patrons 2D habituels. Toutefois, nous avons montré que le *SfT curvilinéaire présente des différences fondamentales à propos des dégénérescences du problème, de son caractère bien-posé et de l'unicité de sa solution.* Ces différences nous motivent à proposer de nouvelles solutions théoriques et algorithmiques.

Contributions théoriques au SfT curvilinéaire. À l'aide de la géométrie continue différentielle, nous étudions les solutions locales, le caractère bien-posé du problème et ses ambiguïtés. Nous montrons que les deux sous-cas du SfT curvilinéaire, qui sont deux problèmes de dimension différente, peuvent être formulés avec la même Équation Différentielle Ordinaire (EDO) et résolus à l'aide d'un Problème de Valeur Initiale (PVI). Cependant, la condition initiale nécessaire à la résolution du PVI est la connaissance de la profondeur d'un point. À première vue, cette information supplémentaire n'est pas en général disponible. Nous proposons de résoudre le PVI en donnant une condition initiale qui est directement obtenue à partir de l'EDO. Cette condition initiale utilise des points spéciaux de la courbe, appelés les points critiques. À travers le PVI utilisant les points critiques, la formulation mathématique du SfT curvilinéaire fournit plusieurs solutions que nous appelons *solutions candidates.* Nous prouvons les résultats suivants :

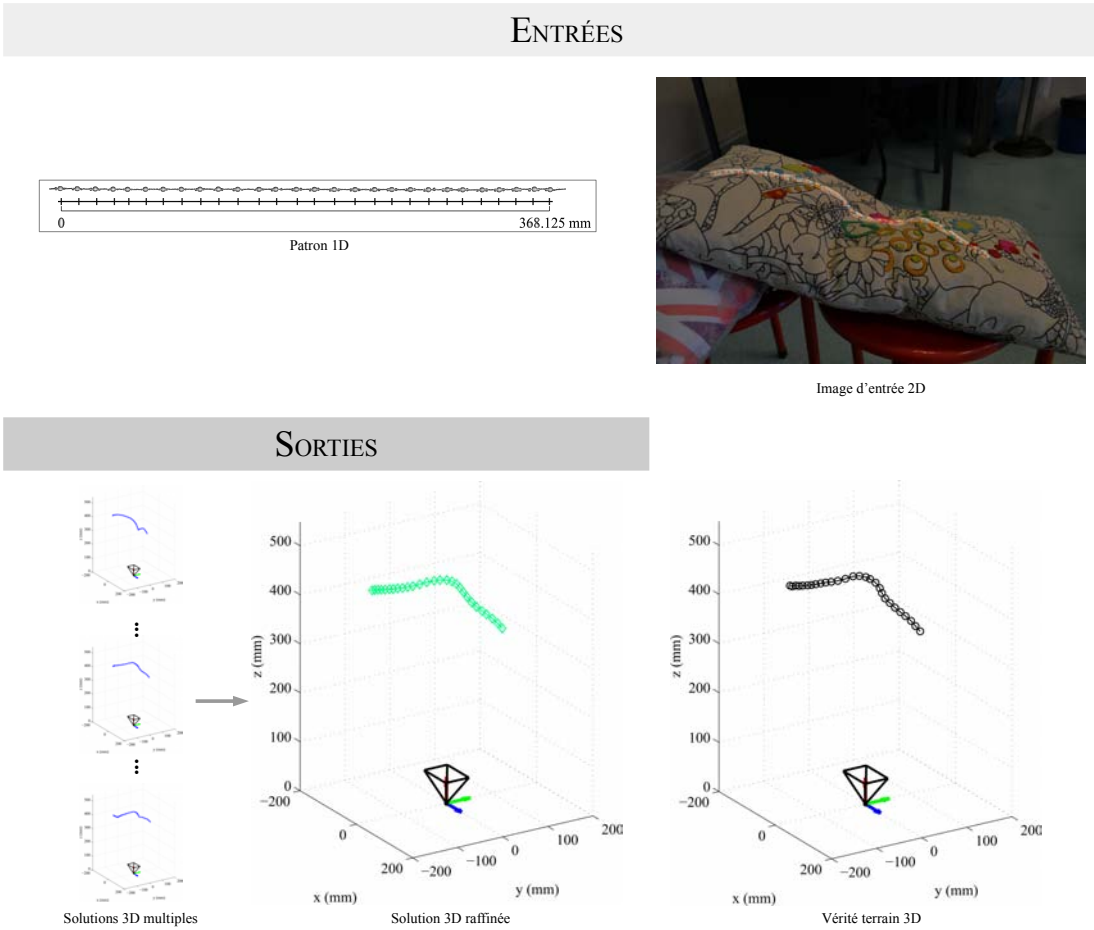


Figure F.4: Un exemple de reconstruction **3D** de courbe à partir d'une image d'entrée **2D** et d'un patron **1D** en utilisant notre solution de raffinement. Cet exemple utilise le jeu de donnée *collier*. Afin de fournir une meilleure visualisation du collier sur le coussin, nous avons mis en avant la région près du collier en assombrissant le reste de l'image. Pour notre méthode, les entrées sont les correspondances entre le patron 1D et l'image d'entrée, qui sont les centres de gravité des perles. Nous montrons plusieurs solutions candidates obtenues à partir de notre méthode basée sur un modèle graphique et donnons une version raffinée du meilleur candidat.

1. Pour le SfT curvilinéaire, la profondeur à un point peut être estimée de manière unique si et seulement si c'est un point critique.

2. Le SfT curvilinéaire peut être résolu avec un nombre fini de solutions si et seulement s'il existe au moins un point critique.

3. Un segment du patron délimité par deux points critiques est reconstructible à une double ambiguïté près.

4. Un problème de SfT curvilinéaire avec N_c points critiques possède 2^{N_c+1} solutions candidates.

Nous étudions également la solubilité du SfT curvilinéaire à travers les solutions locales non-holonomes de l'EDO. Nous prouvons les résultats suivants :

1. Contrairement au SfT surfaçique, le SfT curvilinéaire ne peut pas être résolu exactement à l'aide de solutions locales non-holonomes de l'EDO.
2. Sous l'hypothèse de linéarité infinitésimale, il est possible de résoudre le SfT curvilinéaire à l'aide de solutions locales non-holonomes de l'EDO. La linéarité infinitésimale pour une courbe suppose une courbure nulle sur un segment infini de courbe, mais une courbure globale.

Contributions techniques au SfT curvilinéaire. Dans la littérature, il existe trois catégories de méthodes pour résoudre le SfT surfaçique et volumique : *(i)* solutions locales analytiques, *(ii)* optimisation convexe et *(iii)* optimisation itérative non-convexe. Nous donnons une solution algorithmique pour chaque catégorie dans le cas du SfT curvilinéaire. Pour la catégorie *(i)*, nous réalisons d'abord une étude différentielle du problème comme [Bartoli et al., 2015] et ensuite nous étudions les solutions non-holonomes sous l'hypothèse de linéarité infinitésimale. Pour la catégorie *(ii)*, nous adaptons une formulation convexe du problème de SfT surfaçique, appelé le Maximum Depth Heuristic (MDH) [Perriollat et al., 2011; Salzmann and Fua, 2011]. Cette formulation utilise la contrainte d'inextensibilité, une relaxation de la contrainte d'isométrie qui permet de rendre le problème convexe. Pour la catégorie *(iii)*, nous proposons une formulation non-convexe qui peut être optimisée efficacement à l'aide d'une minimisation basée gradient. Nous réalisons cela avec une nouvelle paramétrisation angulaire qui modélise implicitement les déformations isométriques. Nous introduisons aussi une nouvelle catégorie, *(iv)*, de méthodes de SfT, qui utilise des modèles graphiques discrets. Notre méthode modélise le SfT à l'aide d'un Modèle de Markov Caché (MMC) discret *sans relâcher l'isométrie* et *sans nécessiter une solution initiale*. Cette catégorie est très différente des trois autres catégories de méthodes précédemment utilisées pour résoudre le SfT surfaçique et volumique. Un point important est que notre méthode de catégorie *(iv)* génère toutes les solutions candidates du SfT curvilinéaire, comme le montre la figure F.4. La méthode de catégorie *(iv)* s'appuie sur deux remarques de la théorie : les points critiques séparent la courbe déformée (et donc le patron 1D) en un ensemble de segments et un segment délimité par deux points critiques est reconstructible à une double ambiguïté près. La méthode de catégorie *(iv)* nous permet de sélectionner une de ces deux ambiguïtés sur chacun des segments et de produire un candidat. Ainsi, lorsque la courbe déformée possède N_c points critiques, les 2^{N_c+1} solutions candidates sont estimés en utilisant le MMC pour toutes les combinaisons possibles d'ambiguïtés. Nous fournissons aussi plusieurs méthodes pour détecter ces points critiques à partir de l'EDO.

Résultats et conclusions. Nous avons évalué notre méthode de catégorie *(iv)* pour les deux sous-cas du SfT à l'aide de jeux de données simulés et réels. Cette évaluation est quantitative et qualitative et seul le candidat le plus proche de la vérité terrain est utilisé. Comme les méthodes des catégories *(i)* et *(ii)* n'estiment qu'une seule solution, nous n'avons comparé que la méthode de catégorie *(ii)* avec la méthode de catégorie *(iv)*, suivie ou non du raffinement réalisé par la méthode de catégorie *(iii)*.

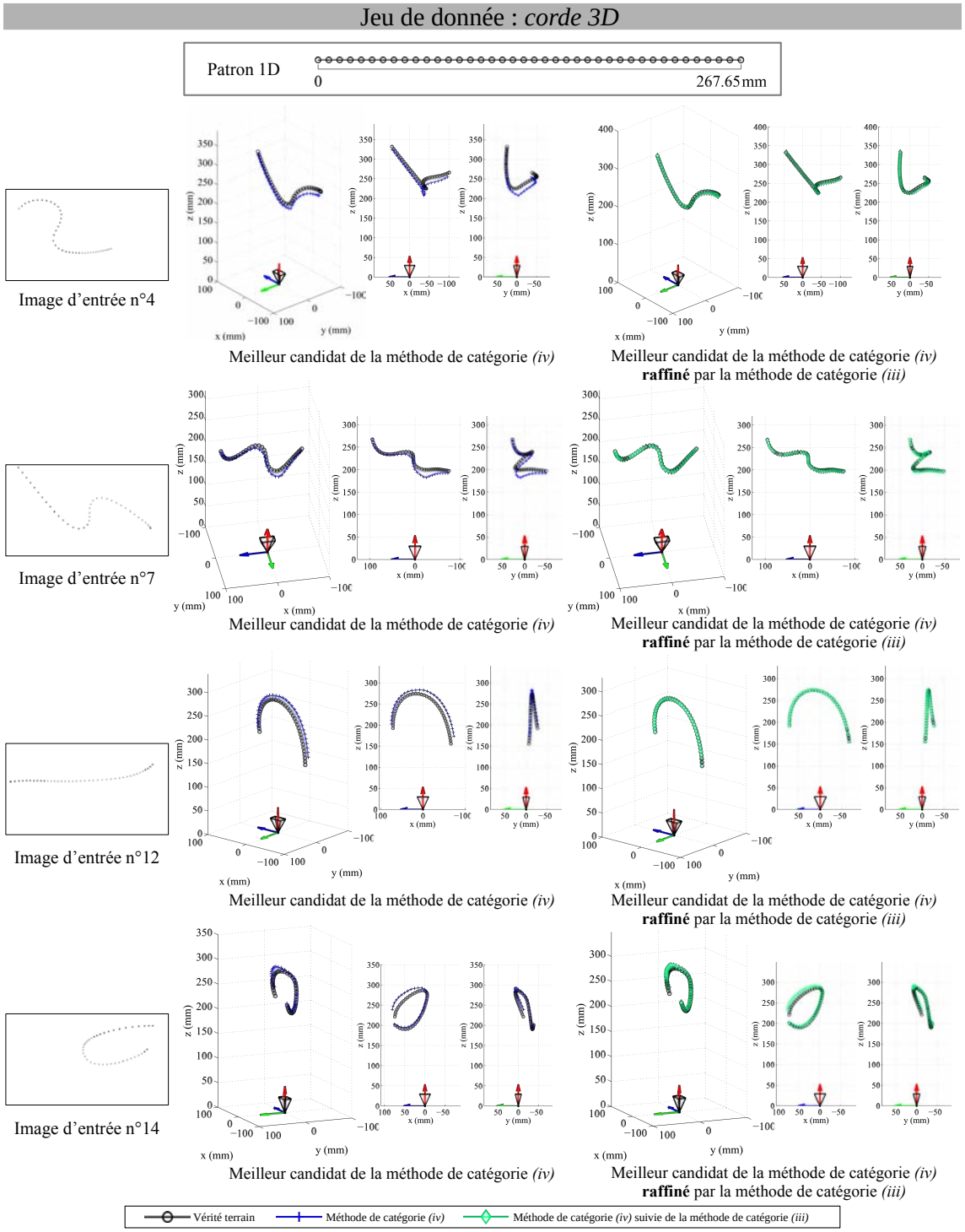


Figure F.5: Résultats visuels de notre méthode de catégorie *(iv)*, suivie ou non de la méthode de catégorie *(iii)*, pour résoudre le SfT curvilinéaire. Le jeu de donnée utilisé ici est *corde 3D* et correspond à une corde simulée sur Blender [Blender, 2017] comprenant 40 correspondances. Pour chaque image d'entrée, nous donnons le meilleur candidat fourni par la méthode de catégorie *(iv)* et sa version raffinée par la méthode de catégorie *(iii)*. Pour chaque reconstruction 3D, nous donnons trois points de vue dont un est le plan xz et un autre le plan yz. Comme les reconstructions 3D sont très proches des vérités terrains, la version digitale fournit une meilleure visualisation.

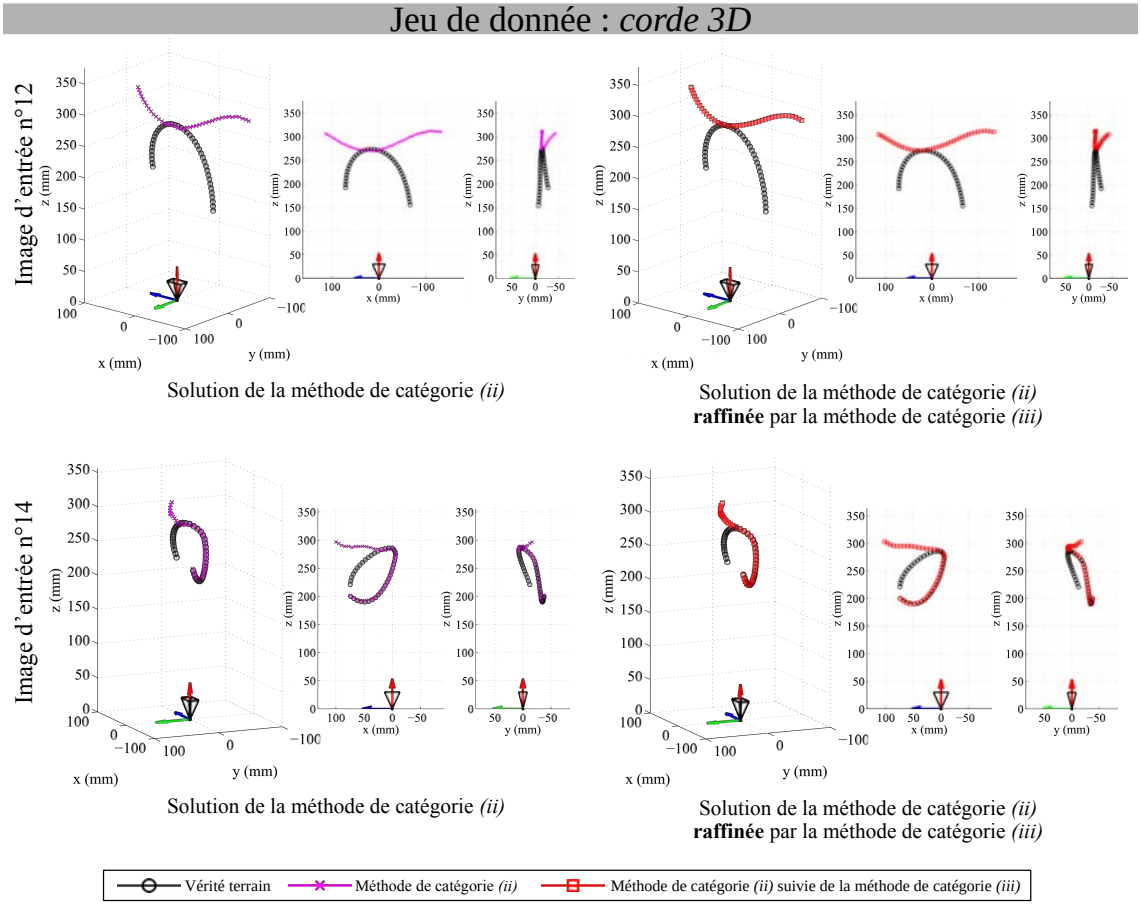


Figure F.6: Résultats visuels de notre méthode de catégorie *(ii)* pour résoudre le SfT curvilinéaire. Le jeu de donnée utilisé ici est *corde 3D* et les images d'entrée sont présents en figure F.5 Pour chaque image d'entrée, nous donnons la solution fournie par la méthode de catégorie *(ii)* et sa version raffinée par la méthode de catégorie *(iii)*. Pour chaque reconstruction 3D, nous donnons trois points de vue, dont un est le plan xz et un autre le plan yz.

Dans les deux cas de SfT, nous avons observé numériquement que la méthode de catégorie *(iv)* fonctionne bien dans la mesure où le meilleur candidat fournit une reconstruction de bonne précision. Nous avons aussi noté que la méthode de catégorie *(iii)* permet d'améliorer la reconstruction du meilleur candidat fourni par la méthode de catégorie *(iv)*. La figure F.5 illustre les bons résultats obtenus par ces deux méthodes et la figure F.6 donne quelques exemples des ambiguïtés inhérents à la méthode de catégorie *(ii)* qui n'estime qu'une seule solution. Un aspect important de l'évaluation de la méthode de catégorie *(iv)* est celle de la détection des points critiques à partir de l'EDO, nécessaire à l'obtention de toutes les solutions candidates possibles. Nous avons restreint l'étude à la méthode de détection de points critiques qui était la plus précise et la plus stable. Une analyse de perturbation sur la position des points critiques détectés a souligné que de grandes erreurs de détection ($\pm$ 10 % de la longueur du patron) aboutissent tout de même à des reconstructions 3D d'une précision raisonnable.

L'étude du problème de SfT curvilinéaire isométrique a mis en évidence sa complexité

non-apparente avec l'existence de solutions multiples en présence de points spéciaux de la courbe que sont les points critiques. Ces mêmes points critiques nous ont permis de proposer un algorithme basé sur des MMC discrets, qui fournit toutes les solutions candidates possibles du problème de SfT curvilinéaire.

F.4.2 Contribution au SfT surfaçique pour les surfaces pliables

Comme mentionné précédemment, la plupart des méthodes existantes de SfT ne sont pas capables de reconstruire et de recaler précisément des déformations complexes comme des plis. [Salzmann and Fua, 2009] a déjà étudié le problème de reconstruction 3D des plis. [Salzmann and Fua, 2009] modélise la surface par un maillage et propose de relâcher l'isométrie en la contrainte d'inextensibilité : la distance géodésique est remplacée par la distance euclidienne. Les sommets peuvent ainsi se rapprocher les uns des autres sans compresser ou étirer la surface, permettant l'apparition de plis. Les plis reconstruits sont alors un sous-produit de la façon dont l'isométrie est relâchée. De plus, les maillages utilisés expérimentalement sont de faible résolution, de l'ordre de $\mathcal{O}(10^2)$, ce qui ne permet pas de vérifier que la méthode recale et reconstruise correctement les plis. La figure F.7 montre qu'en pratique il n'y a pas de méthode de SfT de l'état-de-l'art qui reconstruise des plis, tels que ceux observés sur l'avion en papier.

Reconstruire et recaler des déformations complexes est un problème difficile pour deux raisons. Tout d'abord, nous devons utiliser un espace de déformation de haute dimensionnalité, qui est nécessaire pour modéliser des déformations non lisses telles que des plis qui se formeraient à des positions arbitraires. Nous ne pouvons donc pas approximer le problème en utilisant une représentation de surface lisse globalement (ce qui est fréquemment utilisé dans les méthodes existantes de SfT). Cela augmente alors grandement l'espace de recherche et aboutit à un problème fortement non-convexe. Ensuite, nous ne connaissons pas *a priori* la position des plis. Cela rend très difficile l'utilisation de paramétrisations existantes de plis dans des modèles comme les B-splines, étant donné que nous ne savons pas *a priori* où modifier la spline afin de permettre de grands changements de courbure.

Nous cherchons à résoudre le problème de *SfT-1* pour des surfaces pliables. Pour cela, nous proposons un raffinement par minimisation itérative de contraintes images (correspondances et bords), et des *a priori* de déformation (isométrie et lissage). Nous résumons ici nos deux principales contributions.

Modélisation implicite de plis par un terme adaptatif de lissage agissant sur un maillage surfaçique non-paramétrique de haute résolution. Nous proposons de ne pas imposer globalement un lissage des déformations et de ne pas appliquer de réduction de dimensionnalité. En revanche, nous utilisons une approche non-paramétrique où la surface est modélisée par un maillage triangulaire dense. Nous avons observé que les surfaces pliées peuvent être reconstruites à l'aide de maillage à haute résolution, de l'ordre de $\mathcal{O}(10^4)$ sommets. Nous pouvons travailler à haute résolution puisque les contraintes que nous appliquons sur le maillage sont très éparses (chaque contrainte s'applique uniquement à un

petit nombre de sommets) et que cela nous permet ainsi de résoudre le système résultant de manière itérative à l'aide de solveurs itératifs parcimonieux. Notre solution ne nécessite pas de posséder *a priori* une quelconque information concernant la position des plis. Les plis apparaissent à l'état d'énergie minimale durant l'optimisation. Notre *a priori* utilise une pénalisation robuste basée sur un M-estimateur. Sa construction est inspirée des techniques de flux optique préservant les discontinuités [Black and Anandan, 1993; Zach et al., 2007]. L'idée est de désactiver le lissage de manière adaptative à la géométrie locale pour empêcher ce dernier de "gommer" les plis. Les M-estimateurs sont principalement de types, redescendants et non-redescendants. Nous avons comparé de manière systématique deux des plus communs des M-estimateurs non-redescendants ($(\ell_1$-$\ell_2)$ et Huber) et le plus commun M-estimateur redescendant (Tukey).

Une contrainte robuste de bord. Nous complétons la contrainte de correspondance par une contrainte de bord et ce pour deux raisons. Tout d'abord, les contraintes de correspondance sont souvent des contraintes éparses. Ensuite, comme la position des plis n'est pas connue *a priori*, la surface doit être bien alignée afin que les plis apparaissent à leur position exacte. La contrainte de bord encourage le bord de la surface à se projeter sur les forts gradients de l'image. En pratique, nous projetons le bord de la surface sur une carte de bords, que nous illustrons en figure F.8 (b). Il s'agit d'une contrainte forte et elle doit être utilisée dès que cela est possible. Notre principal défi est de s'assurer que le bord est attiré par les bons gradients, ce qui n'est pas trivial. Pour cela, nous utilisons un modèle statistique de couleur pour éliminer les faux gradients (provenant par exemple des textures de la surface ou du fond). À ce défi s'ajoute la non-convexité de la contrainte de bord, qui peut aussi pousser la solution dans un minimum local. Pour réduire cet effet, deux stratégies sont mises en place : utiliser les contraintes image en cascade (mouvement, puis mouvement et bord) et utiliser une pyramide d'images pour la projection des bords. Ces deux stratégies permettent d'agrandir le bassin de convergence.

Résultats et conclusions. Dans un premier temps, nous avons comparé différentes pénalisations basées sur des M-estimateurs pour déterminer quel type de pénalisation doit être choisi pour notre optimisation. Dans un deuxième temps, nous avons évalué la capacité de notre méthode à recaler et reconstruire des plis à l'aide d'une analyse quantitative et qualitative. Pour cela, nous avons constitué trois jeux de données réelles avec des vérités terrains de haute résolution (23 images au total), construites à l'aide d'un système commercial de lumière structurée de grande précision [David 3D Scanner, 2014]. Nous avons aussi vérifié le bon fonctionnement de notre contrainte robuste de bord.

Nous avons observé qu'en fixant correctement l'hyperparamètre du M-estimateur de Huber, les résultats obtenus avec les M-estimateurs non-redescendants sont similaires et que les deux M-estimateurs non-redescendants aboutissent à la reconstruction 3D de plis. Nous avons toutefois noté que le M-estimateur redescendant ne permet pas de telle reconstruction 3D. Pour la suite de nos expériences, nous avons décidé d'utiliser une pénalisation $(\ell_1$-$\ell_2)$

pour la contrainte de lissage.

L'analyse quantitative a été réalisée en calculant l'erreur de position 3D de la surface et l'erreur sur l'orientation de ses normales à deux niveaux : sur toute la surface et au voisinage des plis. Les résultats numériques ont souligné que notre méthode de SfT fournit des reconstructions 3D plus précises que celles de l'état de l'art, en particulier au niveau des plis, ce que les méthodes existantes ne parviennent pas à faire. La figure F.7 illustre cette amélioration en présentant des rendus visuels pour une image de chaque jeu de données utilisé. La figure F.8 présente une comparaison des reconstructions 3D obtenus selon le type de carte de bords utilisé. Nous avons pu ainsi noter que lever l'ambiguïté sur les gradients de l'image avec le modèle statistique de couleur semble être efficace lorsque la texture de la surface est différente de celle du fond et que cela permet d'éviter certains minima locaux lors de la reconstruction 3D.

C'est ainsi que nous montrons qu'à l'aide d'un maillage dense, d'un terme robuste de lissage basé sur un M-estimateur et d'une utilisation conjointe des contraintes de mouvement et de bord il est possible de recaler et reconstruire avec précision des surfaces pliées en 3D.

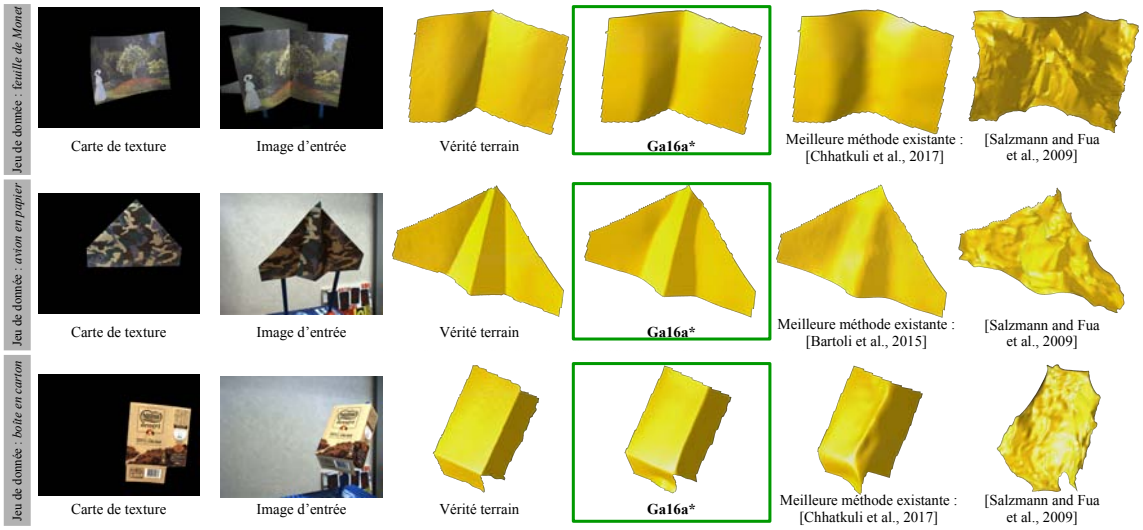
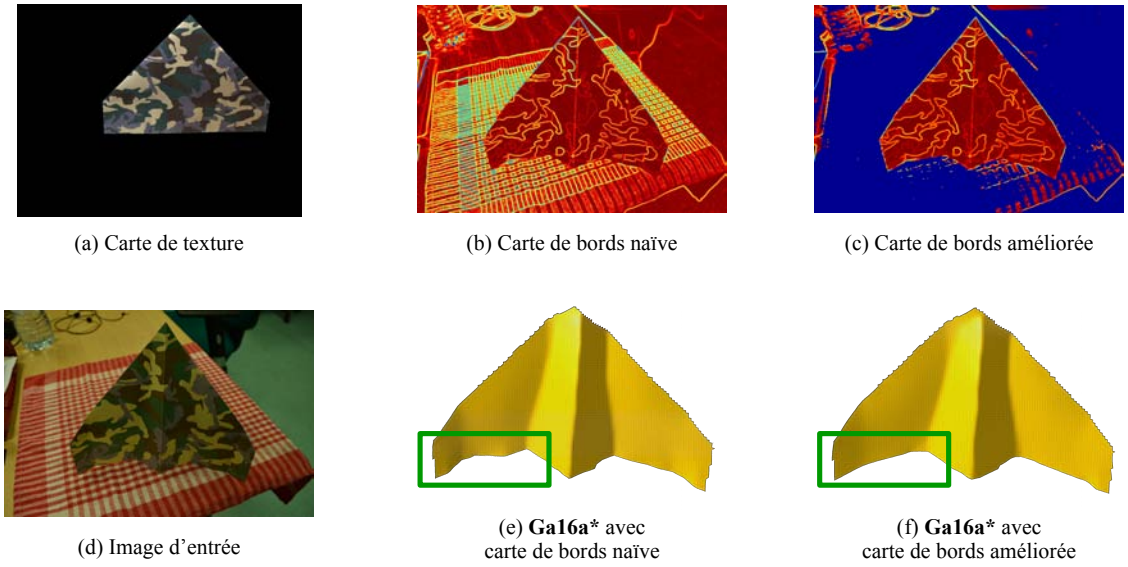


Figure F.7: Résultats visuels de notre solution pour le problème de SfT pour les surfaces pliables et de méthodes existantes. Les trois jeux de données sont créés à l'aide de trois objets réels : *feuille de Monet*, *avion en papier* et *boîte en carton*. Notre méthode, dénotée **Ga16a***, est comparée à la méthode de l'état de l'art fournissant la meilleure reconstruction 3D, [Bartoli et al., 2015] ou [Chhatkuli et al., 2017]. Contrairement à notre méthode, [Bartoli et al., 2015], [Chhatkuli et al., 2017] et [Salzmann and Fua, 2009] ne parviennent pas à reconstruire les plis.



(a) Carte de texture (b) Carte de bords naïve (c) Carte de bords améliorée

(d) Image d'entrée (e) **Ga16a*** avec carte de bords naïve (f) **Ga16a*** avec carte de bords améliorée

Figure F.8: Comparaison des reconstructions 3D en utilisant la carte de bord naïve et améliorée.

F.4.3 Contribution au SfTS surfaçique pour les surfaces pliables et peu texturées

Comme la figure F.11 le souligne, les surfaces peu texturées constituent la deuxième limitation d'une majorité des méthodes SfT de l'état de l'art, en particulier en présence de plis. Nous souhaitons utiliser l'ombrage pour contraindre de manière dense la surface au niveau des régions peu texturées. Comme indiqué en §F.2.4, l'ombrage est utilisable sur des zones peu texturées afin de reconstruire les détails de la surface. Nous souhaitons donc combiner les contraintes de mouvement (applicables aux zones texturées) et d'ombrage (applicables aux zones peu texturées) avec des contraintes physiques venant du patron. Ainsi, nous proposons d'y parvenir en recalant et reconstruisant simultanément le patron et en appliquant les contraintes d'ombrage de manière dense sur la surface du patron. Il s'agit d'un problème nouveau et difficile. Pour appliquer les contraintes d'ombrage, nous devons modéliser la réflectance de la surface (correspondant à des albédos pour des surfaces lambertiennes), l'illumination de la scène et la réponse radiométrique de la caméra. Dans des conditions particulières, la réflectance de la surface peut être connue *a priori*, mais cela est rare. Par exemple, les patrons construits à partir de modèles CAO tels que [TurboSquid, 2016; Warehouse, 2016] ou à partir de systèmes d'acquisition 3D tels que [David 3D Scanner, 2014] n'en contiennent pas en général. De même, l'illumination de la scène et la réponse de la caméra ne sont pas connues *a priori*. Nous proposons d'estimer conjointement toutes ces inconnues (formes 3D, réflectance de la surface, illumination de la scène et réponse de la caméra) à l'aide d'un seul système. Nous y parvenons en utilisant au moins quatre images. Nous précisons qu'il y a une réponse de caméra par image. Notre solution permet également de faire évoluer le patron surfaçique en un patron surfaçique possédant une fonction de réflectance de surface. En figure F.9, nous présentons notre procédure générale.

Les sorties de notre algorithme comportent la forme du patron 3D sur chaque image et la fonction de réflectance de surface. Étant donné la fonction de réflectance de surface, il est donc possible d'utiliser des méthodes existantes de SfTS qui nécessitent de connaître la fonction de réflectance de surface [Liu-Yin et al., 2016; Malti and Bartoli, 2014]. Nous illustrons les relations entre ces algorithmes en figure F.9.

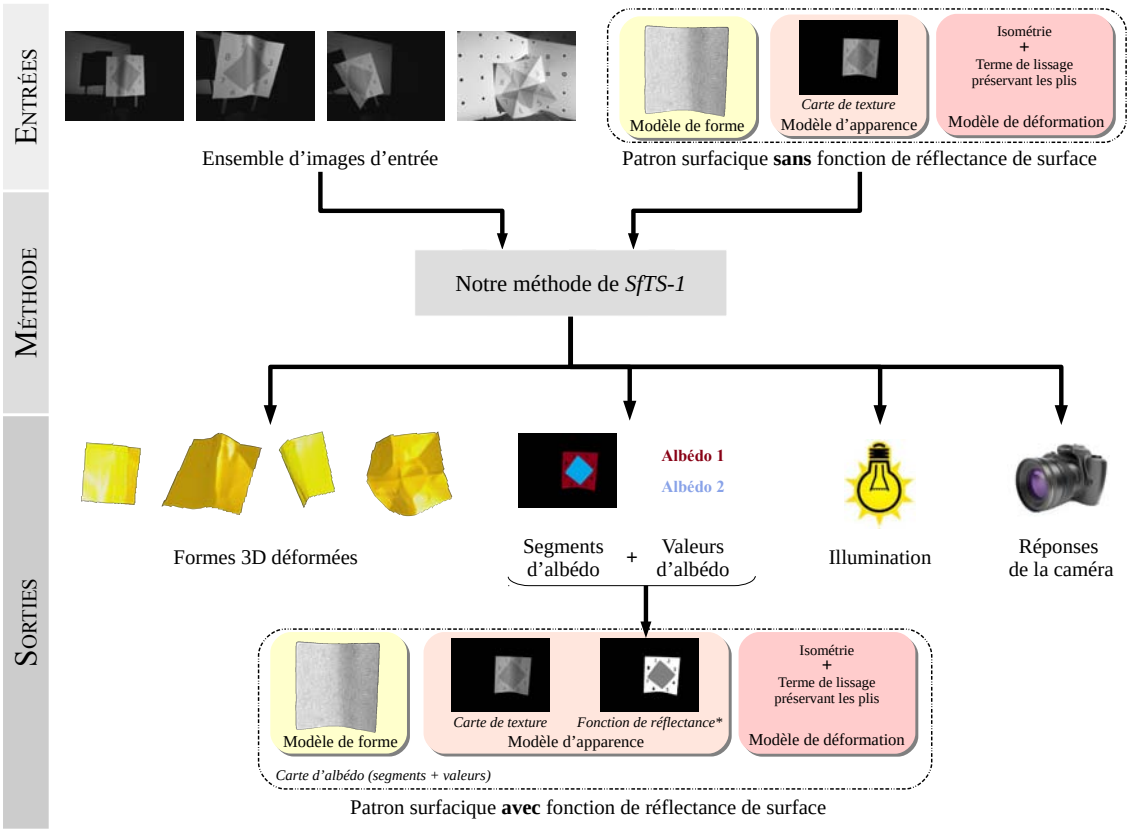


Figure F.9: Illustration de notre solution pour *SfTS-1*. Les entrées de notre méthode sont un ensemble d'au moins quatre images et un patron surfaçique sans fonction de réflectance de surface. Les sorties sont la forme 3D de l'objet déformé et la réponse de la caméra pour chaque image d'entrée, la carte d'albédo et l'illumination. Sous hypothèse de surfaces lambertiennes, notre méthode fournit une estimation de la fonction de réflectance de surface. Notre méthode permet d'intégrer la fonction de réflectance de surface dans le patron surfaçique.

Modélisation du problème en une estimation conjointe de la déformation et des paramètres photométriques à l'aide de trois indices visuels complémentaires. Nous proposons une nouvelle approche intégrée pour résoudre le *SfTS-1* pour des surfaces pliables et peu texturées. Cette approche utilise une contrainte de lissage adaptatif présentée dans notre solution pour le SfT pour les surfaces pliables et elle est construite pour combiner les avantages du SfS et du SfT. Comme pour le SfT, nous utilisons un patron pour apporter au problème des contraintes physiques fortes. Comme pour le SfS, nous utilisons les contraintes d'ombrage pour révéler les déformations complexes. L'utilisation du lissage adaptatif est très importante : il permet d'utiliser la pleine puissance de l'ombrage.

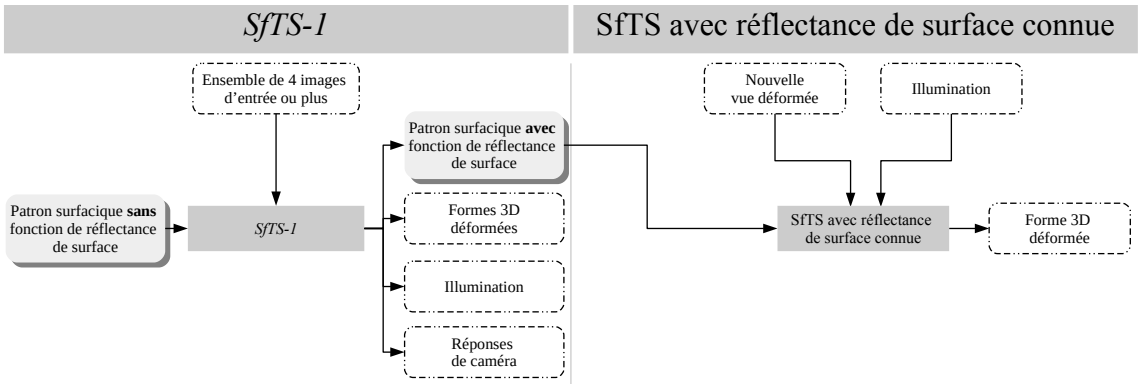


Figure F.10: Lien entre *SfTS-1* et SfTS.

Le problème est résolu par une optimisation des déformations en utilisant les trois indices visuels et les contraintes physiques de déformation, tout en réalisant une auto-calibration des paramètres photométriques nécessaires à l'utilisation de l'ombrage. Contrairement aux méthodes précédentes, notre méthode fonctionne pour tout type de patrons (provenant d'une base de données CAO ou d'un modèle scanné) et ne nécessite pas que la vidéo comporte une séquence d'images où la surface ne se déforme pas.

Résolution par optimisation non-convexe avec initialisation en cascade. Le problème est de grande échelle ($\mathcal{O}(10^4)$ inconnues) et non-convexe. Nous présentons une méthode composée d'une initialisation en cascade suivie d'un raffinement.

L'initialisation estime de manière séquentielle la déformation, l'illumination, la réponse de la caméra et ensuite la carte d'albédo (segments et valeurs) en utilisant au moins quatre images. Un exemple de carte d'albédo est fourni en figure F.9. Tout d'abord, nous utilisons notre méthode de SfT pour les surfaces pliables pour obtenir une solution initiale pour les paramètres de déformation. Nous rappelons que seuls les contraintes de mouvement et de bord sont imposées. Ensuite, pour initialiser l'illumination et la réponse de la caméra, nous utilisons le fait que les déformations sont bien estimées aux points de correspondance dans les régions lisses sans avoir recours au terme d'ombrage. Ainsi, nous initialisons les paramètres photométriques en inversant l'équation d'ombrage et en utilisant l'intensité des pixels et les normales estimées *autour de chaque point de correspondance*. Enfin, la carte d'albédo est initialisée en segmentant d'abord la carte de texture grâce à une décomposition en images intrinsèques [Bell et al., 2014]. Les valeurs d'albédo sont alors estimées pour chaque région segmentée en inversant l'équation d'ombrage et en utilisant les solutions initiales de déformation, d'illumination et de réponse de la caméra.

Nous réalisons le raffinement par une minimisation itérative, où la fonction d'énergie regroupe trois termes de données images (correspondances, bords et ombrage) et deux termes d'*a priori* physiques (isométrie et lissage adaptatif). Nous utilisons le même cadre de raffinement que pour notre méthode de SfT pour les surfaces pliables et intégrons également une pyramide d'images pour améliorer la convergence du terme d'ombrage.

Résultats et conclusions. Pour évaluer la capacité de notre méthode à recaler et reconstruire des surfaces pliables et peu texturées, nous procédons de la même manière que pour évaluer notre contribution *2)*, mais ici nous utilisons des surfaces peu texturées. Nous avons créé deux jeux de données réelles (12 images au total) avec vérité terrain de haute résolution à l'aide du système à lumière structurée précédemment utilisé. Nous avons aussi créé deux jeux de données réelles (9 images au total) sans vérité terrain.

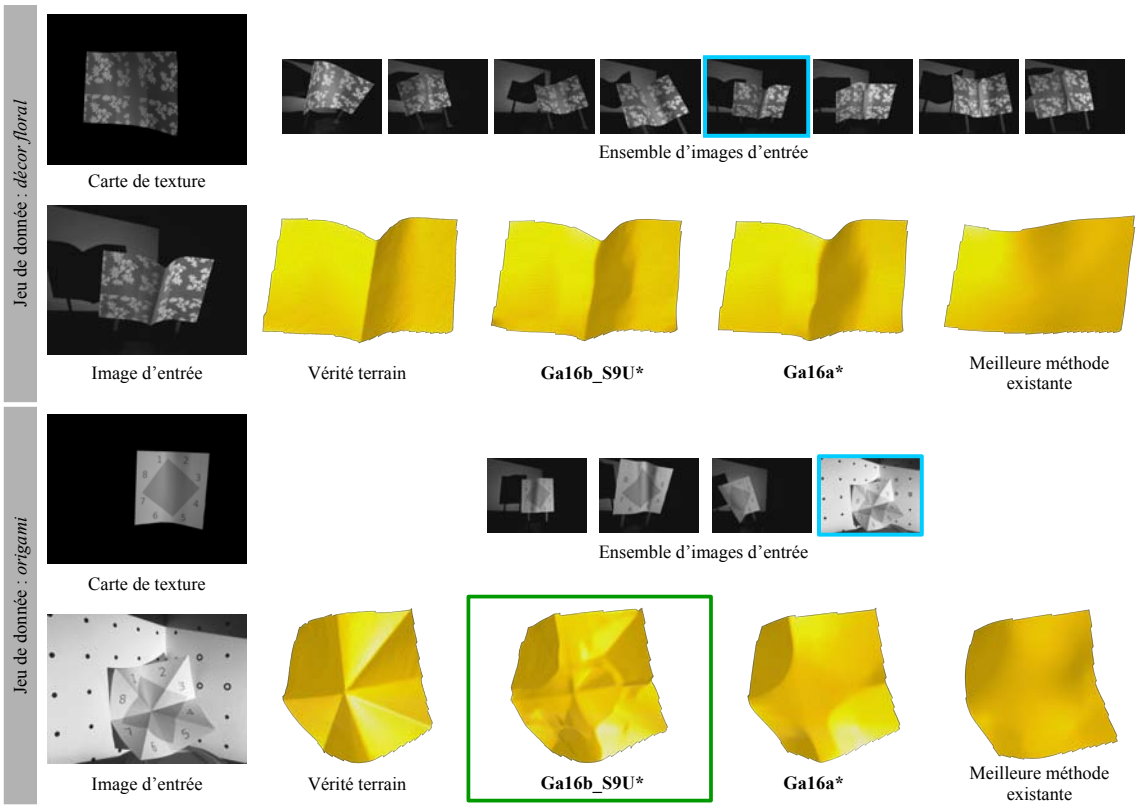


Figure F.11: Résultats visuels, avec vérité terrain, de notre solution pour le problème de *SfTS-1*. Les deux jeux de données sont créés à l'aide de deux objets réels : *décor floral* et *origami*. Notre méthode, notée **Ga16b_S9U***, est comparée à notre solution **Ga16a*** pour le SfT pour les surfaces pliables (qui n'utilise pas l'ombrage) et à la méthode de l'état de l'art fournissant la meilleure reconstruction 3D [Chhatkuli et al., 2017]. La comparaison des reconstructions 3D obtenues avec **Ga16b_S9U*** et **Ga16a*** souligne la contribution de l'ombrage dans la reconstruction 3D de plis sur des régions peu texturées. Contrairement à notre méthode **Ga16b_S9U***, [Chhatkuli et al., 2017] ne parvient pas à reconstruire en 3D les plis sur les régions peu texturées.

Nous avons observé que la précision des reconstructions 3D de notre méthode, notée **Ga16b_S9U***, est meilleure que celle des méthodes de l'état de l'art et celle de notre méthode qui n'utilise pas l'ombrage, **Ga16a***. Les figures F.11, F.13 et F.14 proposent des comparaisons des rendus visuels de reconstruction 3D entre les différentes méthodes. Ces rendus visuels confirment les résultats numériques obtenus et mettent en évidence la contribution de l'ombrage dans le recalage et la reconstruction 3D de plis sur des surfaces peu texturées. Un point important de notre méthode est que celle-ci ne requiert pas de calibration photométrique *a priori*, mais estime conjointement à la reconstruction 3D les paramètres photométriques,

tels que la carte d'albédo (segments et valeurs), la réponse de la caméra et l'illumination de la scène. En figure F.12, nous pouvons par exemple noter que notre initialisation en cascade fournit une estimation raisonnable de l'illumination de la scène.

Notre méthode montre ainsi qu'il est possible de reconstruire une surface soumise à des déformations complexes et non lisses sur tout type de régions visibles (texturées et peu texturées) et d'estimer simultanément la fonction de réflectance de surface (les albédos) sans avoir recours à aucune calibration photométrique *a priori*.

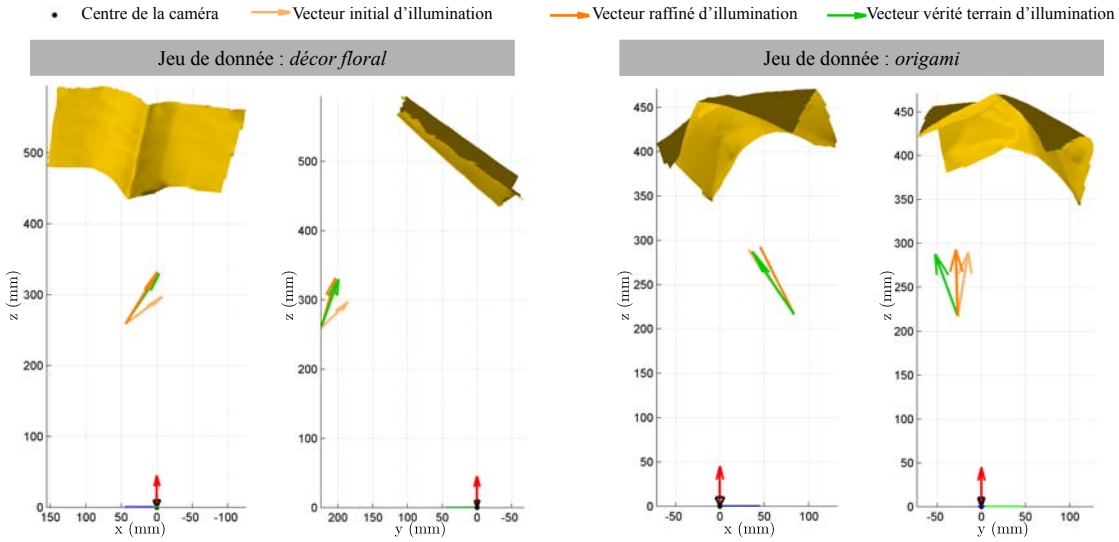


Figure F.12: Visualisation 3D du vecteur d'illumination pour les deux jeux de données réelles avec vérité terrain. L'illumination est modélisée par des harmoniques sphériques du premier ordre, ce qui correspond à un vecteur 3D et un terme ambiant. Nous comparons ici les vecteurs d'illumination obtenus en de l'initialisation en cascade et après le raffinement avec celui de la vérité terrain. **Colonne n°1 et n°2** : vues à partir du plan xz et du plan yz pour l'image d'entrée du jeu de donnée *décor floral* utilisée en figure F.11. **Colonne n°3 et n°4** : vues à partir du plan xz et du plan yz pour l'image d'entrée du jeu de donnée *origami* utilisée en figure F.11.

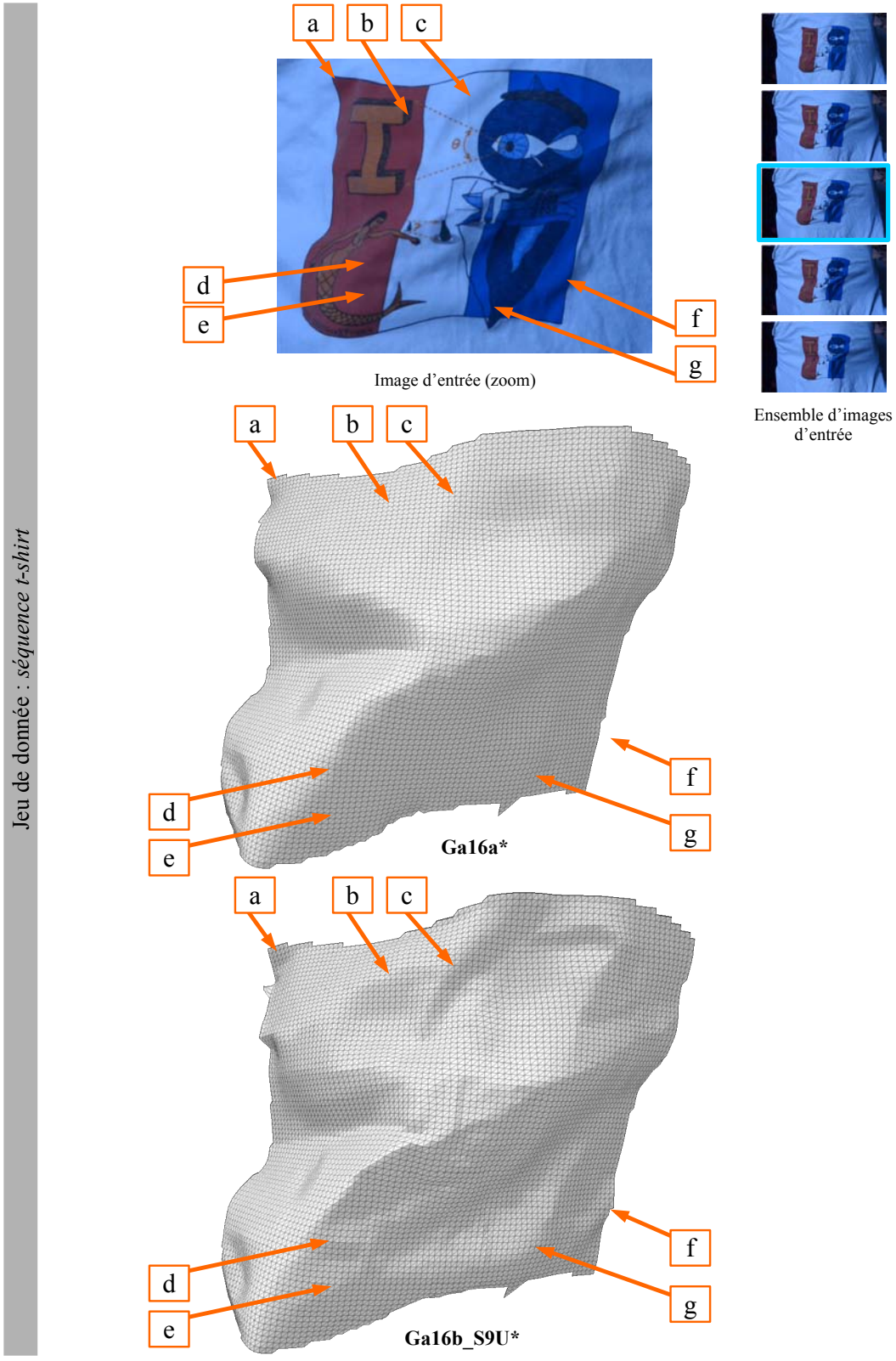


Figure F.13: Résultats visuels, sans vérité terrain, de notre solution pour le problème de *SfTS-1* pour le jeu de donnée réel *séquence t-shirt*. Nous donnons un agrandissement de l'image d'entrée afin de voir plus facilement les plis sur les surfaces peu texturées. Afin d'observer les contributions de l'ombrage, nous comparons notre méthode **Ga16b_S9U*** avec notre méthode **Ga16a*** qui n'utilise pas l'ombrage. Nous précisons que nous avons changé l'illumination pour faciliter la visualisation des contributions de l'ombrage. Certaines contributions de l'ombrage sont indiquées à l'aide de flèches oranges.

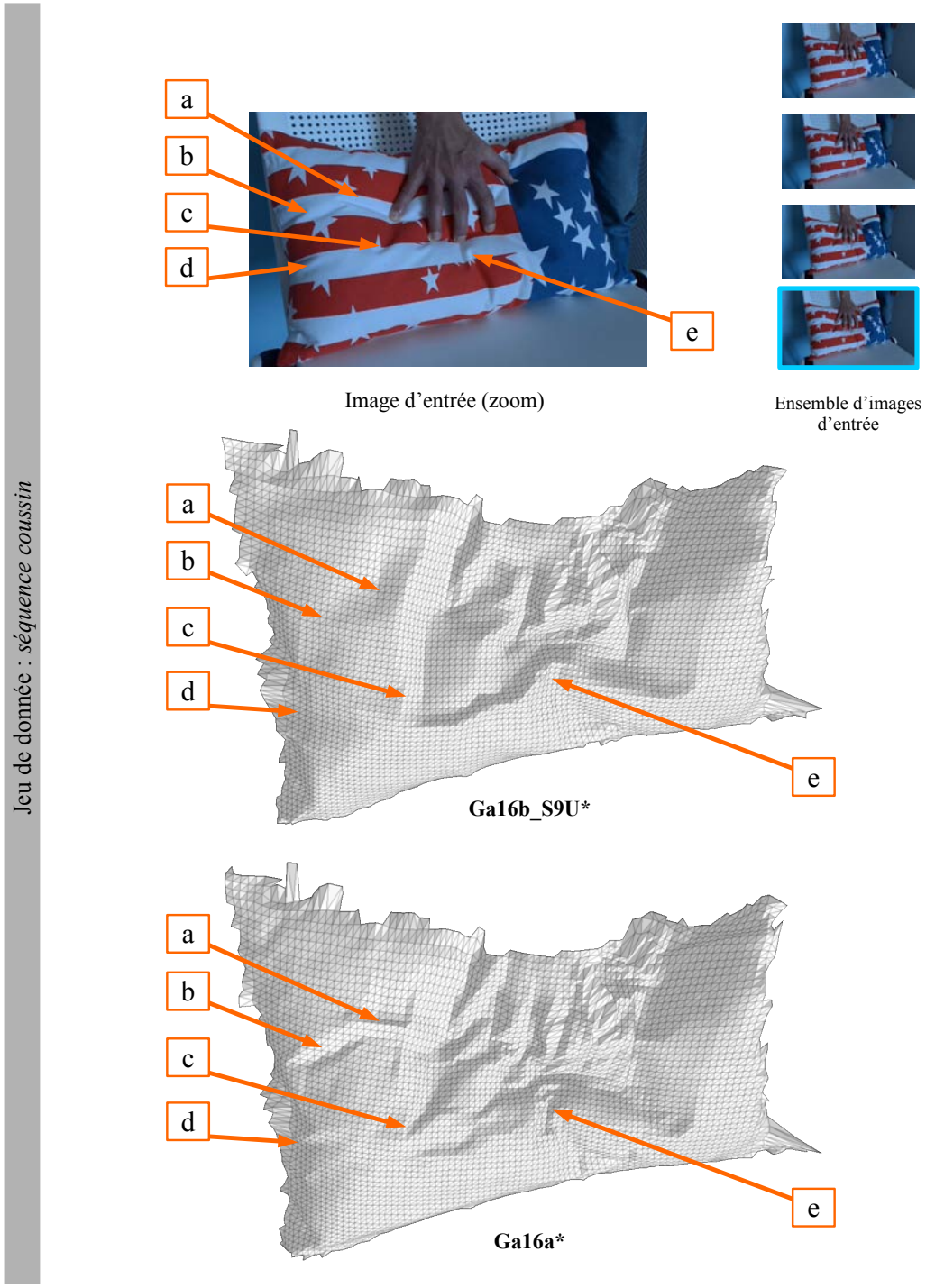


Figure F.14: Résultats visuels, sans vérité terrain, de notre solution pour le problème de *SfTS-1* pour le jeu de donnée réel *séquence coussin*. Nous donnons un agrandissement de l'image d'entrée afin de voir plus facilement les plis sur les surfaces peu texturées. Afin d'observer les contributions de l'ombrage, nous comparons notre méthode **Ga16b_S9U*** avec notre méthode **Ga16a*** qui n'utilise pas l'ombrage. Nous précisons que nous avons changé l'illumination pour faciliter la visualisation des contributions de l'ombrage. Certaines contributions de l'ombrage sont indiquées à l'aide de flèches oranges.

F.4.4 Contribution au NRSfM surfaçique pour les surfaces pliables et peu texturées

Le NRSfM souffre des mêmes limitations que le SfT : la plupart des méthodes existantes de NRSfM ne peuvent pas reconstruire les surfaces peu texturées soumises à des déformations complexes, comme le montre la figure F.17. Les résultats de notre méthode pour le problème de *SfTS-1* nous ont encouragé à repousser les limitations du NRSfM en combinant les indices visuels de mouvement, de bord et d'ombrage avec un modèle générique de déformation physique capable de représenter des déformations non lisses avec précision. Pour les mêmes raisons que pour le SfT, nous proposons de reconstruire simultanément la forme 3D de la surface dans chaque image d'entrée et sa fonction de réflectance. Cela constitue le problème de *NRSfMS-1*, qui est plus difficile que le problème de *SfTS-1* puisqu'aucun patron de la surface n'est disponible. C'est pour cela que nous restreignons notre étude au cas où l'illumination de la scène et la réponse de la caméra sont connues. En figure F.15, nous montrons la procédure générale de notre méthode. En figure F.17, nous montrons des reconstructions 3D obtenues par notre méthode et les meilleures reconstructions 3D obtenues parmi les méthodes de l'état de l'art. Le problème de *NRSfMS-1* n'a jamais été résolu auparavant et est donc un élément manquant pour la reconstruction dense dans un environnement non contrôlé.

De même que pour *SfTS-1*, la figure F.16 montre comment le problème de *NRSfMS-1* peut être relié au problème de SfTS. À partir d'un ensemble d'images d'entrée, nous pouvons construire un patron de la surface et l'actualiser avec la fonction de réflectance de surface en résolvant le *NRSfMS-1*. Ensuite, le patron peut être utilisé pour reconstruire la forme 3D de la même surface visible sur une nouvelle image ou une vidéo prise dans des conditions différentes.

Ce que nous proposons dans cette thèse est une preuve de concept que le NRSfM et l'ombrage peuvent être combinés pour reconstruire de manière dense tout type de surface (texturée ou peu texturée) soumises à des déformations isométriques lisses ou non lisses. Notre principale contribution vis-à-vis du problème de *NRSfMS-1* est une initialisation en cascade de maillages de haute résolution et d'albédos, suivie d'un raffinement de plusieurs contraintes image (correspondances, bords et ombrage) et de deux *a priori* de déformations (isométrie et lissage adaptatif). Comme il s'agit de la première approche qui résout *NRSfMS-1*, nous joignons une étude empirique de la stabilité du problème à l'aide d'une analyse de perturbation. Nous fournissons également des résultats expérimentaux sur des données réelles avec vérité terrain.

Initialisation et raffinement. Notre formulation du problème de NRSfM aboutit à trois grands challenges : la haute dimensionnalité de l'espace de déformation (accrue par l'absence de patron, par rapport aux deux précédentes méthodes), la non-convexité du problème et la nécessité d'un recalage pixellique. Encouragés par les résultats de notre solution au problème de *SfTS-1*, nous proposons une stratégie d'initialisation suivie d'un raffinement pour répondre à ces trois défis.

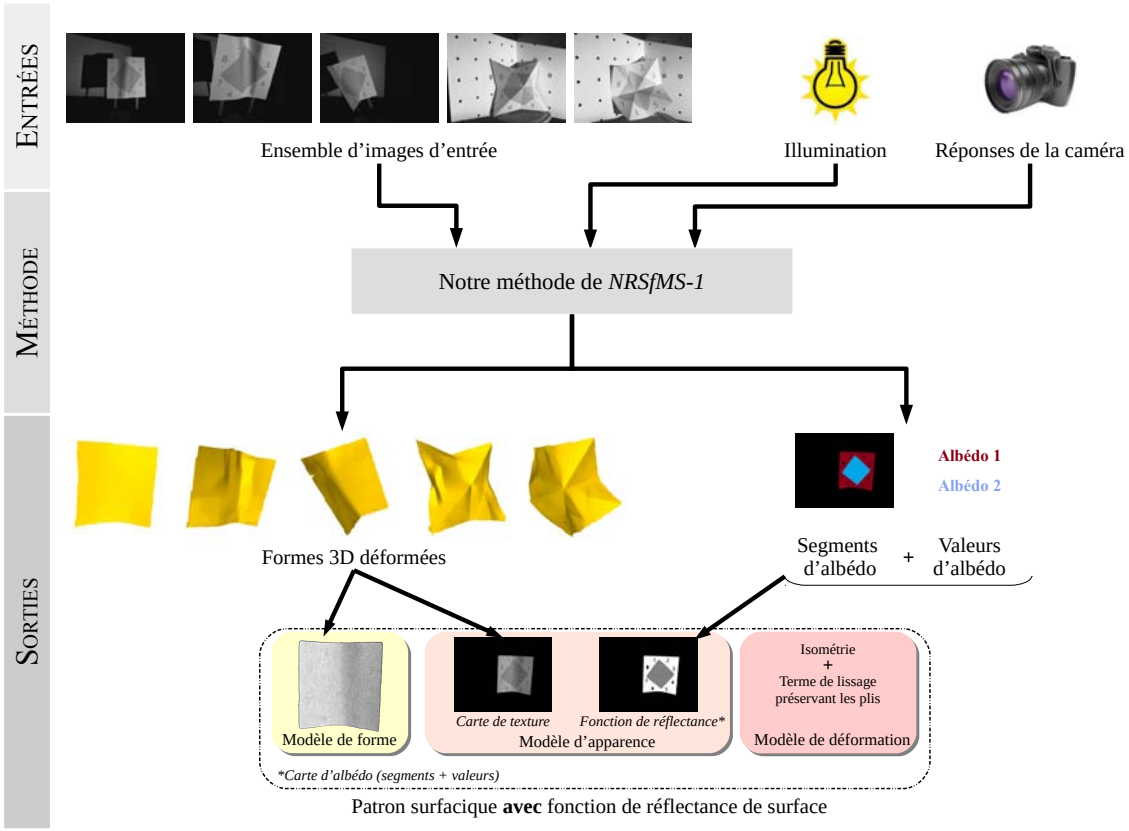


Figure F.15: Illustration de notre solution pour *NRSfMS-1*. Les entrées de notre méthodes sont un ensemble d'images d'entrée, l'illumination et la réponse de la caméra. Les sorties sont les formes 3D de l'objet déformé pour chaque image d'entrée et la carte d'albédo. Sous hypothèse de surfaces lambertiennes, notre méthode fournit une estimation de la fonction de réflectance de surface en suivant la même approche que notre solution au *SfTS-1* : une segmentation de la carte de texture en un ensemble de régions d'albédo constante par morceaux et d'autre part une estimation de la valeur des albédos par inversion de l'équation d'ombrage. Notre méthode permet de construire un patron surfaçique avec une fonction de réflectance de surface.

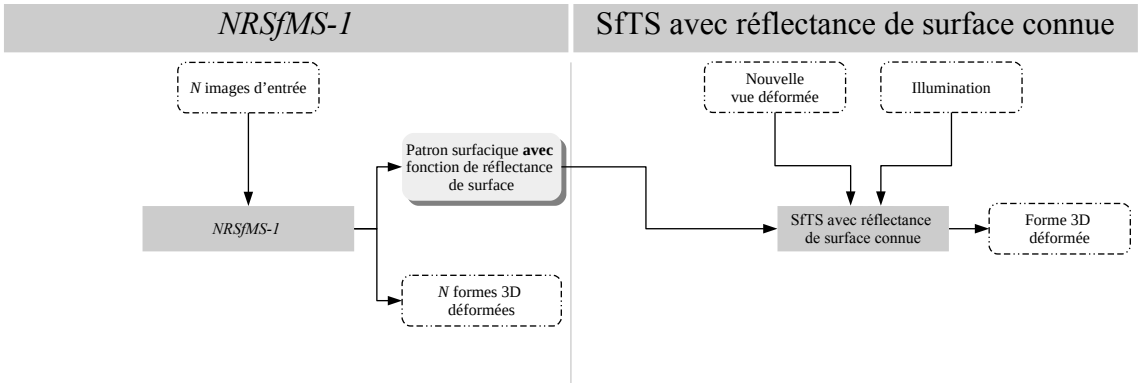


Figure F.16: Lien entre *NRSfMS-1* et SfTS.

L'initialisation comporte trois étapes. La première étape consiste en l'initialisation d'une forme de référence à partir des points de correspondance entre l'image de référence et les

autres images. Nous supposons que la forme sur l'image de référence est lisse. En principe, notre modélisation et notre algorithme fonctionnent pour des images de référence lisses ou non lisses, mais c'est pour des images de référence lisses que nous avons obtenu les meilleurs reconstructions 3D. La forme de référence est estimée à l'aide d'une méthode existante de NRSfM [Chhatkuli et al., 2014]. Deux points importants sont que cette méthode ne nécessite pas d'initialisation et fournit des reconstructions 3D lisses. La deuxième étape considère la forme de référence comme un patron et optimise la forme 3D de l'objet indépendamment sur chacune des images d'entrée différentes de l'image de référence. Pour cela, nous utilisons notre méthode **Ga16a***, qui nous permet d'obtenir des reconstructions 3D plus précises. La troisième étape consiste en la segmentation de la carte de texture en un ensemble de régions d'albédo constante et à l'estimation de la valeur des albédos par inversion de l'équation d'ombrage. Cela est possible puisqu'à cette étape nous connaissons l'illumination de la scène et la réponse de la caméra pour chaque image d'entrée et possédons une estimation de la forme 3D de l'objet sur chacune des images d'entrée.

Le raffinement optimise pour toutes les images (y compris l'image de référence) une fonction d'énergie rassemblant trois termes de données image (correspondances, bords et ombrage) et deux termes d'*a priori* physiques (isométrie et lissage adaptatif). Pour améliorer la convergence, nous utilisons également une pyramide d'images pour la contrainte de bord et d'ombrage.

Résultats et conclusions. Pour réaliser l'analyse de perturbation, nous avons utilisé trois jeux de données présentant des surfaces peu texturées avec des déformations non lisses et dont la vérité terrain de haute résolution a été obtenue à l'aide du système à lumière structurée précédemment utilisé. L'analyse de perturbation a montré que la précision des reconstructions 3D était comparable pour de petites perturbations, ce qui nous informe sur la présence d'un fort minimum local près de la vraie solution et ce qui nous conforte sur notre formulation du problème de *NRSfMS-1* comme un problème de minimisation d'énergie.

Concernant l'évaluation quantitative de notre méthode, notée **Ga17a***, nous avons utilisé au total cinq jeux de données réels (5 images par jeu) de haute résolution présentant des surfaces peu texturées avec des déformations non lisses et un jeu de donnée réel (5 images) présentant une surface texturée soumise à des déformations lisses. Ce dernier jeu de donnée nous permet de montrer que notre méthode fonctionne également pour des surfaces (texturées sous déformations lisses) pour lesquelles les méthodes de l'état de l'art proposent de bonnes solutions. Numériquement, pour les six jeux de données, notre méthode **Ga17a*** fournit globalement des reconstructions 3D plus précises que les méthodes existantes de NRSfM, en particulier au niveau des plis. La figure F.17 confirme visuellement ces résultats et montre que les méthodes de NRSfM de l'état de l'art ne parviennent pas à reconstruire précisément des surfaces pliées peu texturées.

Cette première étude du problème de *NRSfMS-1* montre qu'il est possible en combinant le NRSfM et l'ombrage de reconstruire en 3D et de recaler des surfaces génériques (texturées et peu texturées) qui sont soumises à des déformations isométriques lisses et non lisses et

dont l'albédo est inconnue et varie spatialement par morceaux. Un résultat intéressant de notre méthode est la création d'un patron possédant une fonction de réflectance de surface, qui peut être utilisé pour reconstruire en 3D la même surface dans des conditions différentes à l'aide d'une méthode de SfTS.

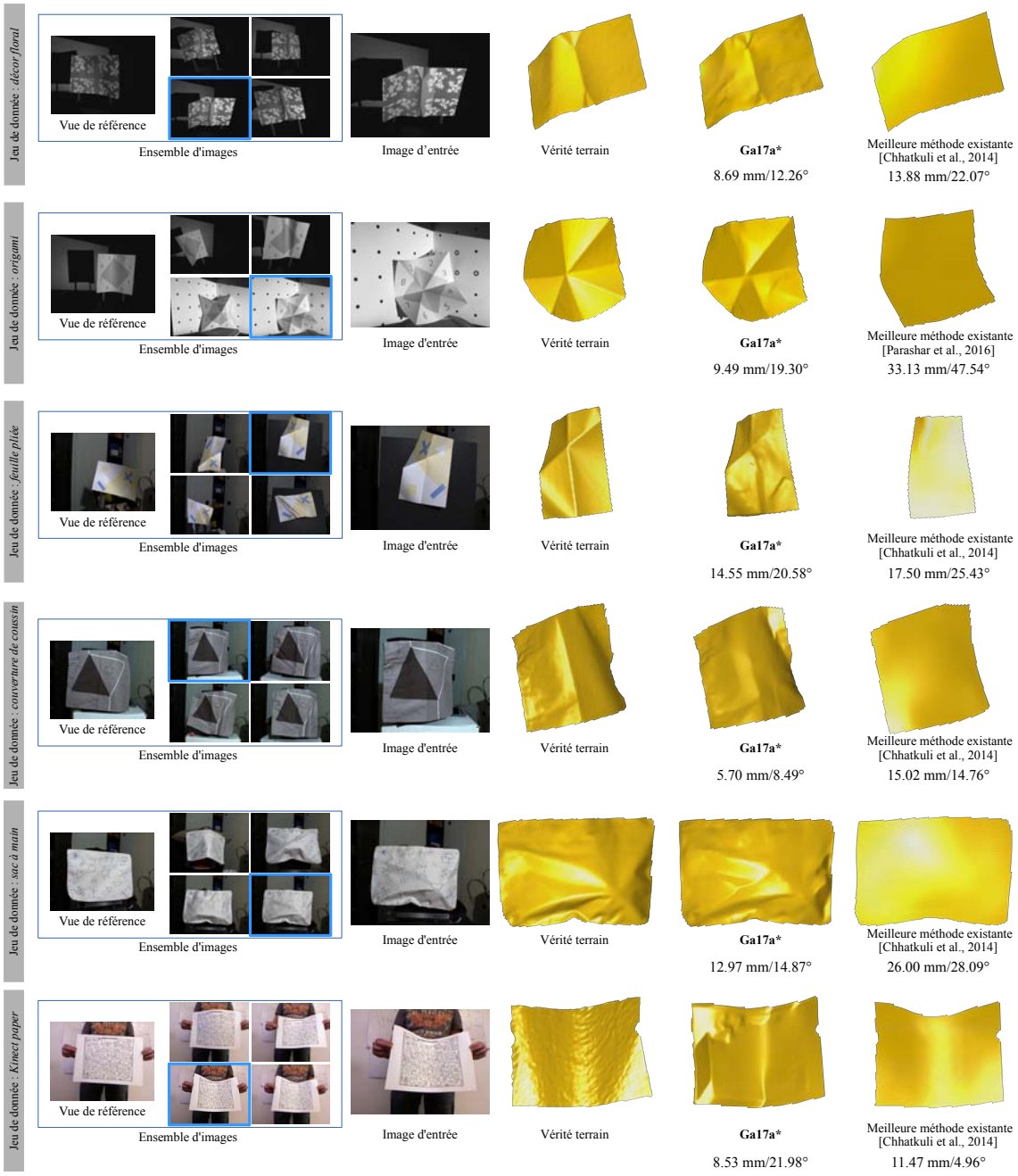


Figure F.17: Résultats visuels, avec vérité terrain, de notre solution pour le problème de NRSfM pour les surfaces pliables et peu texturées et d'une méthode existante. Les cinq premiers jeux de données sont créés à l'aide de cinq objets réels : *décor floral*, *origami*, *feuille pliée*, *housse de coussin* et *sac à main*. Le sixième jeu de donnée est *Kinect paper*, fourni par [Varol et al., 2009]. Notre méthode, dénotée **Ga17a***, est comparée à la méthode de l'état de l'art fournissant la meilleure reconstruction 3D. La comparaison des reconstructions 3D souligne la contribution de l'ombrage dans la reconstruction 3D de plis sur des régions peu texturées. Contrairement à notre méthode **Ga17a***, les méthodes de l'état de l'art ne parviennent pas à reconstruire les plis sur les régions peu texturées.

F.5 Conclusions et perspectives

F.5.1 Conclusions sur nos travaux

Cette thèse décrit nos contributions au problème de reconstruction 3D déformable monoculaire via l'étude de modèles curvilinéaires et l'utilisation d'indices visuels multiples pour des modèles surfaçiques. D'une part, nous avons étudié le cas du SfT avec des courbes 1D se déformant de manière isométrique. D'autre part, nous avons proposé de nouvelles méthodes de SfT et de NRSfM qui combinent plusieurs indices visuels afin de faire avancer l'état de l'art concernant les types gérés de surfaces et de déformations isométriques. Nous rassemblons ci-dessous nos conclusions quant à nos différentes contributions et proposons quelques pistes pour de futurs travaux.

F.5.1.1 Shape-from-Template pour des modèles curvilinéaires

Nous avons présenté une étude théorique du problème du SfT curvilinéaire et son implémentation pour reconstruire respectivement des courbes 2D et 3D à partir d'un patron 1D. À la différence du SfT surfaçique, nous avons montré que le SfT curvilinéaire possède des solutions ambiguës. Nous avons donné les conditions nécessaires et suffisantes pour la solubilité du problème à l'aide des points critiques, qui sont calculés directement à partir des données. Nous avons aussi montré que, contrairement au SfT surfaçique, le SfT curvilinéaire ne peut pas être résolu localement à l'aide de solutions non-holonomes. En terme d'implémentation, nous avons fourni quatre méthodes de catégorie différente. Deux méthodes, de catégorie *(i)* et *(ii)*, ne fournissent qu'une seule solution. La méthode de catégorie *(iii)* raffine une seule solution à l'aide d'une nouvelle paramétrisation angulaire des courbes 2D et 3D. Seule la méthode de catégorie *(iv)*, basée sur un MMC discret, estime toutes les solutions candidates du problème en utilisant les éléments théoriques développés et, en particulier, les points critiques. Nous avons également proposé plusieurs méthodes de détection de points critiques. L'évaluation quantitative et qualitative de la méthode basée sur un MMC discret et de sa version raffinée sur des courbes simulées et réelles, comme un collier, a montré que de telles courbes peuvent être reconstruites par du SfT curvilinéaire.

F.5.1.2 Utilisation de plusieurs indices visuels pour le SfT et le NRSfM

Une très grande majorité des méthodes actuelles de SfT et de NRSfM présentent deux grandes limitations : elles ne parviennent pas à reconstruire en 3D des surfaces peu texturées et soumises à des déformations complexes telles que des plis. Cela s'explique par deux caractéristiques des méthodes actuelles. Tout d'abord, elles utilisent le mouvement de points de correspondance, qui s'avère insuffisant pour reconstruire les régions peu texturées. Ensuite, elles utilisent des réductions de dimensionnalité ou de forts lissages sur ces mêmes régions, ce qui empêche la reconstruction 3D de fortes courbures et donc de détails. Pour surmonter ces deux limitations, nous avons proposé deux idées qui s'intègrent à un cadre de modélisation et d'optimisation non-convexe d'une fonction de coût.

La première idée est d'utiliser un terme de lissage adaptatif qui permet de modéliser des plis, sans nécessiter de connaître leur position *a priori*. Nous avons construit ce terme de lissage à l'aide d'une pénalisation robuste basée sur un M-estimateur. À l'aide d'une étude de différents M-estimateurs, nous avons pu vérifier que les pénalisations robustes basées sur les M-estimateurs non-redescendants de (ℓ_1-ℓ_2) ou Huber aboutissent à la formation de plis et présentent des reconstructions 3D très similaires. Des expériences de reconstruction 3D d'objets réels par SfT ont souligné la capacité de notre méthode à modéliser des plis.

La seconde idée est de combiner les indices visuels du mouvement et de bord avec celui de l'ombrage qui permet de contraindre de manière dense les régions peu texturées, ce que ne peuvent pas faire les contraintes de mouvement et de bord. L'utilisation du lissage adaptatif précédemment présenté est un pré-requis essentiel à l'intégration de l'ombrage pour permettre à cet indice visuel de révéler des plis. Toutefois, l'utilisation de l'ombrage requiert de connaître les paramètres photométriques, qui sont la réflectance de la surface, l'illumination et la réponse de la caméra. Nous supposons alors que la surface est lambertienne et que la réflectance de la surface consiste à un ensemble de régions d'albédo constante par morceaux, ce qui est une bonne approximation d'un grand nombre de surfaces de la vie quotidienne. Pour le problème du SfT, nous avons proposé une méthode qui estime simultanément la déformation de la surface et les paramètres photométriques en utilisant un ensemble d'au moins quatre images où la surface se déforme. Puisque l'utilisation de l'ombrage conduit à un problème fortement non-convexe, un élément essentiel de cette méthode est l'initialisation des paramètres photométriques à partir de cet ensemble d'images. Ensuite, comme le problème de NRSfM est bien moins contraint que celui du SfT (par l'absence du patron), nous avons considéré connues l'illumination et la réponse de la caméra et proposé une preuve de concept qu'en combinant l'ombrage et le NRSfM il est possible de reconstruire la réflectance d'une surface soumise à des déformations isométriques complexes, sans en connaître un patron *a priori*. La contribution de l'ombrage a pu être observé quantitativement et qualitativement grâce à une évaluation des méthodes de SfT et de NRSfM sur plusieurs objets réels, comme du papier ou du tissu.

Cette thèse a pu montrer ainsi que l'utilisation d'indices visuels multiples et d'un lissage robuste permet d'agrandir l'éventail des types de surfaces et de déformations isométriques que les méthodes de SfT et de NRSfM peuvent reconstruire.

F.5.2 Perspectives sur nos travaux

Certains aspects de nos contributions nécessitent d'être développés et d'autres axes de recherche peuvent être explorés.

F.5.2.1 Shape-from-Template pour des modèles curvilinéaires

Deux problèmes ouverts du SfT curvilinéaire sont l'amélioration de la détection des points critiques et l'étude des courbes fermées.

Amélioration de la détection des points critiques. Un élément central de la théorie et des solutions pratiques du problème de SfT curvilinéaire est les points critiques et leur détection. Cependant, cette détection nécessite en pratique le calcul de dérivées secondes de fonctions construites par interpolation de données d'entrée. L'étude de différentes fonctions d'interpolation pourrait alors aboutir à une détection plus robuste et précise des points critiques et à de meilleures reconstructions 3D.

Étude des courbes fermées. Un seconde axe de recherche est d'adapter notre méthode basée MMC pour l'utilisation de patron curvilinéaire avec une boucle comme un collier fermé. Cela complique le problème puisque la résolution du MMC devient un problème NP-difficile. Cependant, nous pensons que de bons résultats peuvent être obtenus en utilisant des méthodes d'inférences approximatives telles que la propagation des convictions à boucle.

F.5.2.2 Utilisation de plusieurs indices visuels pour le SfT et le NRSfM

Nous donnons ici trois axes concrets de recherche, mais d'autres problèmes ouverts existent, tels que l'amélioration de l'utilisation de l'ombrage, l'extension du NRSfM à des modèles volumétriques et l'utilisation de modèles photométriques plus complexes.

Approches basées apprentissage pour le SfT et l'estimation de la réflectance et de l'illumination. Récemment, les techniques d'apprentissage basées sur des Réseaux de Neurones à Convolution (RNC) ont montré des performances élevées pour résoudre des problèmes où les données d'entrée présentent le même objet contenu dans les jeux de données d'apprentissage. Des exemples de ces problèmes sont la détection et la reconnaissance d'objets ou calcul de pose. Comme le SfT suppose une connaissance *a priori* de l'objet à reconstruire (le patron), de telles techniques d'apprentissage s'avèrent donc être une piste intéressante à étudier pour résoudre le SfT. C'est ainsi que des premiers travaux tels que [Golyanik et al., 2018; Pumarola et al., 2018] ont proposé d'utiliser les RNC pour résoudre le SfT. Cependant, [Golyanik et al., 2018] ne montre pas de résultats pour des images réelles et [Pumarola et al., 2018] utilise comme base d'apprentissage des cartes de profondeur et les images 2D associées qui sont présentes dans la base de tests. Une limitation de ces deux méthodes est qu'elles ne semblent reconstruire que des déformations lisses.

Un défi important qu'a cherché à relever cette thèse est l'estimation de la réflectance et de l'illumination. Des travaux utilisant des RNC [Kim et al., 2017; Mandl et al., 2017] ont également essayé de résoudre ces problèmes séparément. [Kim et al., 2017] propose d'estimer la réflectance (avec des composantes diffuse et spéculaire) d'un objet rigide à partir d'une séquence d'images et de ses cartes de profondeur associées (données par la Kinect par exemple). [Mandl et al., 2017] propose d'estimer l'illumination d'une scène à partir d'une seule image où un objet rigide connu est visible. Il serait donc intéressant de voir si de telles approches peuvent être généralisées à des objets déformables.

Étudier comment combiner les modèles mathématiques utilisés dans cette thèse et présentés dans l'état de l'art avec de telles approches basées apprentissage est une tâche

très importante, car elle peut permettre à la fois de contourner certaines limites des modèles mathématiques et d'utiliser plus efficacement les données à l'aide de ces mêmes modèles mathématiques.

Automatisation de la construction des cartes de bord. La contrainte de bord proposée dans cette thèse nécessite une carte de bords pour laquelle nous avons proposé un modèle statistique de couleur afin d'éliminer les faux gradients. Toutefois, lors de nos expériences avec le NRSfM, nous avons fixé manuellement les paramètres de construction des cartes de bords améliorées pour chaque jeu de donnée. Il serait alors intéressant d'automatiser le réglage de ces paramètres en fonction des images d'entrée.

Comme la construction de ces cartes améliorées nécessite la segmentation de la surface sur les images d'entrée, d'autres méthodes de segmentation pourraient être envisagées. Dans le cas du SfT, comme l'objet à segmenter dans l'image d'entrée est connu *a priori* grâce au patron, les méthodes de segmentation pouvant intégrer cette connaissance peuvent être des solutions particulièrement appropriées. De plus, la résolution conjointe des problèmes de reconnaissance et de segmentation a déjà montré une amélioration des résultats pour les deux problèmes [He et al., 2017]. Dans le cas du NRSfM, des méthodes de segmentation multi-vues [Wang and Collomosse, 2012] ou co-segmentation [Vicente et al., 2011] sont des approches à explorer.

La suppression des forts gradients provenant de la surface elle-même est également un problème important à résoudre pour améliorer la convergence de la contrainte de bord.

Extension à une approche incrémentale de notre solution au *NRSfMS-1*. Dans nos expériences de *NRSfMS-1*, nous avons utilisé des ensembles de 5 images. L'utilisation d'ensembles d'images plus larges permettraient une meilleure estimation des formes 3D de la surface dans les différentes images, des albédos de la surface et d'un patron de cette même surface. Cela viendrait du fait que posséder plus d'informations image (ou utiliser la continuité temporelle pour des vidéos) améliorerait la géométrie de la surface et donc son recalage sur les images, aboutissant ainsi à une meilleure estimation des albédos de la surface. Cela nous fournirait un meilleur patron de la surface. Une approche incrémentale intégrant de nouvelles images par groupe d'images nous permettrait alors d'utiliser plus d'informations image et d'obtenir de meilleures estimations tout en évitant d'accroître grandement le coût algorithmique de notre méthode. Cela peut conduire des versions temps-réel.

Bibliography

3Dflow. 3DF Zephyr. https://www.3dflow.net, 2017.

Agisoft. Agisoft Lens Version 0.4.1 beta 64 bit (build 1718). http://www.agisoft.com, 2013.

Agisoft. Agisoft PhotoScan version 1.2.3 build 2331 (64 bit). http://www.agisoft.com, 2014.

A. Agudo, F. Moreno-Noguer, B. Calvo, and J. M. M. Montiel. Sequential Non-Rigid Structure from Motion Using Physical Priors. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 38(5):979–994, May 2016.

A. H. Ahmed and A. A. Farag. Shape from Shading Under Various Imaging Conditions. In *CVPR*, 2007.

I. Akhter, Y. Sheikh, S. Khan, and T. Kanade. Nonrigid Structure from Motion in Trajectory Space. In *NIPS*, 2009.

L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

S. Baker, S. Roth, D. Scharstein, M. J. Black, J. P. Lewis, and R. Szeliski. A Database and Evaluation Methodology for Optical Flow. In *ICCV*, 2007.

L. Bao, Q. Yang, and H. Jin. Fast Edge-Preserving PatchMatch for Large Displacement Optical Flow. IEEE *Transactions on Image Processing*, 23(12):4996–5006, December 2014.

J. T. Barron and J. Malik. Shape, Illumination, and Reflectance from Shading. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, August 2015.

A. Bartoli and E. Özgür. A Perspective on Non-Isometric Shape-from-Template. In *ISMAR*, 2016.

A. Bartoli, Y. Gérard, F. Chadebecq, T. Collins, and D. Pizarro. Shape-from-Template. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 37(10):2099–2118, October 2015.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross. High-Quality Single-Shot Capture of Facial Geometry. In SIGGRAPH, 2010.

P. N. Belhumeur, D. J. Kriegman, and A. L. Yuille. The Bas-Relief Ambiguity. In *CVPR*, 1997.

S. Bell, K. Bala, and N. Snavely. Intrinsic Images in the Wild. *ACM Trans. on Graphics (SIGGRAPH)*, 33(4), 2014.

J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical Model-Based Motion Estimation. In *ECCV*, 1992.

R. Berthilsson, K. Åström, and A. Heyden. Reconstruction of General Curves, Using Factorization and Bundle Adjustment. *International Journal of Computer Vision*, 41(3):171–182, February 2001.

M. J. Black and P. Anandan. A Framework for the Robust Estimation of Optical Flow. In *ICCV*, 1993.

M. J. Black, Y. Yacoob, A. D. Jepson, and D. J. Fleet. Learning Parameterized Models of Image Motion. In *CVPR*, 1997.

V. Blanz and T. Vetter. Morphable Model for the Synthesis of 3D Faces. In SIGGRAPH, 1999.

Blender. Blender 2.78a. https://www.blender.org, 2017.

C. Bregler, A. Hertzmann, and H. Biermann. Recovering Non-Rigid 3D Shape from Image Streams. In *CVPR*, 2000.

G. J. Brostow, C. Hernández, G. Vogiatzis, B. Stenger, and R. Cipolla. Video Normals from Colored Lights. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33(10): 2104–2114, 2011.

T. Brox and J. Malik. Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33 (3):500–513, March 2011.

T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High Accuracy Optical Flow Estimation Based on a Theory for Warping. In *ECCV*, 2004.

F. Brunet, R. Hartley, and A. Bartoli. Monocular Template-Based 3D Surface Reconstruction: Convex Inextensible and Nonconvex Isometric Methods. *Computer Vision and Image Understanding*, 125:138–154, August 2014.

D. Casillas-Perez and D. Pizarro. Solutions of Quadratic First-Order ODEs applied to Computer Vision Problems. *ArXiv*, 1710.04265, 2017.

M. Chertok and Y. Keller. Efficient High Order Matching. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2205–2215, December 2010.

A. Chhatkuli, D. Pizarro, and A. Bartoli. Non-Rigid Shape-from-Motion for Isometric Surfaces using Infinitesimal Planarity. In *BMVC*, 2014.

A. Chhatkuli, D. Pizarro, T. Collins, and A. Bartoli. Inextensible Non-Rigid Shape-from-Motion by Second-Order Cone Programming. In *CVPR*, 2016.

A. Chhatkuli, D. Pizarro, A. Bartoli, and T. Collins. A Stable Analytical Framework for Isometric Shape-from-Template by Surface Integration. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 39(5):833–850, May 2017.

C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. In *ECCV*, 2016.

J. F. Claerbout and F. Muir. Robust Modeling With Erratic Data. *Geophysics*, 38(5): 826–844, 1973.

T. Collins and A. Bartoli. Locally Affine and Planar Deformable Surface Reconstruction from Video. In *VMV*, 2010.

T. Collins and A. Bartoli. Towards Live Monocular 3D Laparoscopy using Shading and Specularity Information. In *International Conference on Information Processing in Computer-Assisted Interventions*, 2012.

T. Collins and A. Bartoli. Using Isometry to Classify Correct/Incorrect 3D-2D Correspondences. In *ECCV*, 2014.

T. Collins and A. Bartoli. Realtime Shape-from-Template: System and Applications. In *ISMAR*, 2015.

T. Collins, P. Mesejo, and A. Bartoli. An Analysis of Errors in Graph-Based Keypoint Matching and Proposed Solutions. In *ECCV*, 2014.

T. Collins, A. Bartoli, N. Bourdel, and M. Canis. Dense, Robust and Real-time 3D Tracking of Deformable Organs in Monocular Laparoscopy. In *MICCAI*, 2016.

Y. Dai, H. Li, and M. He. A Simple Prior-Free Method for Non-Rigid Structure-from-Motion Factorization. *International Journal of Computer Vision*, 107(2):101–122, 2014.

David 3D Scanner. http://www.david-3d.com/en/products/david4, 2014.

A. Del Bue. A Factorization Approach to Structure from Motion with Shape Priors. In *CVPR*, 2008.

D. L. Donoho and P. B. Stark. Uncertainty Principles and Signal Recovery. *SIAM J. Appl. Math.*, 49(3):906–931, June 1989.

A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *CVPR*, 2015.

O. Duchenne, F. Bach, I. S. Kweon, and J. Ponce. A Tensor-Based Algorithm for High-Order Graph Matching. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33(12): 2383–2395, December 2011.

A. Ecker and A. D. Jepson. Polynomial shape from shading. In *CVPR*, 2010.

Y. Eliashberg and N. M. Mishachev. *Introduction to the h-Principle*. Number Grad. Stud. Math. 48. American Mathematical Society, 2002.

H. Fan, H. Su, and L. J. Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *CVPR*, 2017.

O. Faugeras and T. Papadopoulo. A Theory of the Motion Fields of Curves. *International Journal of Computer Vision*, 10(2):125–156, April 1993.

J. Fayad, L. Agapito, and A. Del Bue. Piecewise Quadratic Reconstruction of Non-Rigid Surfaces from Monocular Sequences. In *ECCV*, 2010.

S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust Moving Least-Squares Fitting with Sharp Features. *ACM Trans. Graph.*, 24(3):544–552, July 2005.

K. Fukunaga and L. Hostetler. The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. IEEE *Transactions on Information Theory*, 21(1): 32–40, January 1975.

Y. Furukawa and J. Ponce. Accurate, Dense, and Robust Multiview Stereopsis. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, Aug 2010.

R. Gal, A. Shamir, T. Hassner, M. Pauly, and D. Cohen-Or. Surface Reconstruction using Local Shape Priors. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, 2007.

M. Gallardo, D. Pizarro, A. Bartoli, and T. Collins. Shape-from-Template in Flatland. In *CVPR*, 2015.

M. Gallardo, T. Collins, and A. Bartoli. Can we Jointly Register and Reconstruct Creased Surfaces by Shape-from-Template Accurately? In *ECCV*, 2016a.

M. Gallardo, T. Collins, and A. Bartoli. Using Shading and a 3D Template to Reconstruct Complex Surface Deformations. In *BMVC*, 2016b.

M. Gallardo, T. Collins, and A. Bartoli. Dense Non-Rigid Structure-from-Motion and Shading with Unknown Albedos. In *ICCV*, 2017.

R. Garg, A. Roussos, and L. Agapito. Dense Variational Reconstruction of Non-rigid Surfaces from Monocular Video. In *CVPR*, 2013.

A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR*, 2012.

C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised Monocular Depth Estimation with Left-Right Consistency. In *CVPR*, 2017.

S. Gold and A. Rangarajan. A Graduated Assignment Algorithm for Graph Matching. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, April 1996.

V. Golyanik, S. Shimada, K. Varanasi, and D. Stricker. HDM-Net: Monocular Non-Rigid 3D Reconstruction with Learned Deformation Model. *ArXiv*, 1803.10193, 2018.

P. F. U. Gotardo and A. M. Martinez. Kernel Non-Rigid Structure from Motion. In *ICCV*, 2011.

B. F. Gregorski, B. Hamann, and K. I. Joy. Reconstruction of B-spline Surfaces from Scattered Data Points. In SIGGRAPH, 2000.

N. Gumerov, A. Zandifar, R. Duraiswami, and L. S. Davis. Structure of Applicable Surfaces from Single Views. In *ECCV*, 2004.

X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching. In *CVPR*, 2015.

N. Haouchine, J. Dequidt, I. Peterlik, E. Kerrien, M. O. Berger, and S. Cotin. Image-Guided Simulation of Heterogeneous Tissue Deformation for Augmented Reality During Hepatic Surgery. In *ISMAR*, 2013.

N. Haouchine, J. Dequidt, M. O. Berger, and S. Cotin. Single View Augmentation of 3D Elastic Objects. In *ISMAR*, 2014.

R. Hartley and R. Vidal. Perspective Nonrigid Shape and Motion Recovery. In *ECCV*, 2008.

R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. Second Edition.

K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. In *ICCV*, 2017.

Y. He and H. Qin. Surface Reconstruction with Triangular B-splines. In *GMP*, 2004.

H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 295–302, 1994.

B. K. P. Horn. Shape from Shading: A Method for Obtaining the Shape of a Smooth Shape of a Smooth Opaque Object from One View. Technical report, Cambridge, MA, USA, 1970.

B. K. P. Horn and B. G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1): 185–203, 1981.

M. Hornáček, F. Besse, J. Kautz, A. Fitzgibbon, and C. Rother. Highly Overparameterized Optical Flow Using PatchMatch Belief Propagation. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *ECCV*, 2014.

Y. Hu, R. Song, and Y. Li. Efficient Coarse-to-Fine Patch Match for Large Displacement Optical Flow. In *CVPR*, 2016.

K. Ikeuchi and B. K. P. Horn. Numerical shape from shading and occluding boundaries. *Artificial Intelligence*, 17(1):141–184, 1981.

E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *CVPR*, 2017.

H. Jin, D. Cremers, D. Wang, E. Prados, A. Yezzi, and S. Soatto. 3-D reconstruction of shaded objects from multiple images under unknown illumination. *International Journal of Computer Vision*, 76(3):245–256, 2008.

M. Kaess and F. Dellaert. Reconstruction of Objects with Jagged Edges through Rao-Blackwellized Fitting of Piecewise Smooth Subdivision Curves. In *Proceedings of the First* IEEE *International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, 2003.

A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-Specific Object Reconstruction from a Single Image. In *CVPR*, 2015.

K. Kim, A. Torii, and M. Okutomi. Multi-view Inverse Rendering Under Arbitrary Illumination and Albedo. In *ECCV*, 2016.

K. Kim, J. Gu, S. Tyree, P. Molchanov, M. Nießner, and J. Kautz. A Lightweight Approach for On-the-Fly Reflectance Estimation. In *ICCV*, 2017.

R. Kimmel and A. M. Bruckstein. Global Shape-from-Shading. In *ICPR*, 1994.

B. Koo, E. Özgür, B. Le Roy, E. Buc, and A. Bartoli. Deformable Registration of a Preoperative 3D Liver Volume to a Laparoscopy Image Using Contour and Shading Cues. In *MICCAI*, 2017.

I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper Depth Prediction with Fully Convolutional Residual Networks. In *3DV*, 2016.

F. Langguth, K. Sunkavalli, S. Hadap, and M. Goesele. Shading-aware Multi-view Stereo. In *ECCV*, 2016.

J. Lee, M. Cho, and K. M. Lee. Hyper-Graph Matching via Reweighted Random Walks. In *CVPR*, 2011.

K. M. Lee and C.-C. Kuo. Shape from Shading with a Generalized Reflectance Map Model. *Computer Vision and Image Understanding*, 67(2):143–160, 1997.

K. M. Lee and C. C. J. Kuo. Shape from Shading with a Linear Triangular Element Surface Model. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(8):815–822, August 1993.

M. Leordeanu and M. Hebert. A Spectral Technique for Correspondence Problems Using Pairwise Constraints. In *ICCV*, 2005.

M. Leordeanu, M. Hebert, and R. Sukthankar. An Integer Projected Fixed Point Method for Graph Matching and MAP Inference. In *NIPS*, 2009.

Q. Liu-Yin, R. Yu, L. Agapito, A. Fitzgibbon, and C. Russell. Better Together: Joint Reasoning for Non-rigid 3D Reconstruction with Specularities and Shading. In *BMVC*, 2016.

J. Löfberg. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *International Symposium on Computer-Aided Control System Design*, 2004.

D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, 1981.

Q.-T. Luong, P. Fua, and Y. Leclerc. The Radiometry of Multiple Images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 24(1):19–33, 2002.

S. Magnenat, D. T. Ngo, F. Zund, M. Ryffel, G. Noris, G. Rothlin, A. Marra, M. Nitti, P. Fua, M. Gross, and R. Sumner. Live Texturing of Augmented Reality Characters from Colored Drawings. IEEE *Transactions on Visualization and Computer Graphics*, 21(11): 1201–1210, 2015.

F. Mai and Y. S. Hung. 3D Curves Reconstruction from Multiple Images. In *Digital Image Computing: Techniques and Applications*, 2010.

A. Malti and A. Bartoli. Combining Conformal Deformation and Cook-Torrance Shading for 3D Reconstruction in Laparoscopy. IEEE *Transactions on Biological Engineering*, 61(6): 1684–1692, June 2014.

A. Malti, A. Bartoli, and T. Collins. A Pixel-Based Approach to Template-Based Monocular 3D Reconstruction of Deformable Surfaces. In *Proceedings of the* IEEE *International Workshop on Dynamic Shape Capture and Analysis at ICCV*, 2011.

A. Malti, R. Hartley, A. Bartoli, and J. Kim. Monocular Template-Based 3D Reconstruction of Extensible Surfaces with Local Linear Elasticity. In *CVPR*, 2013.

A. Malti, A. Bartoli, and R. I. Hartley. A Linear Least-Squares Solution to Elastic Shape-from-Template. In *CVPR*, 2015.

D. Mandl, K. M. Yi, P. Mohr, P. M. Roth, P. Fua, V. Lepetit, D. Schmalstieg, and D. Kalkofen. Learning Lightprobes for Mixed Reality Illumination. In *ISMAR*, 2017.

H. Martinsson, F. Gaspard, A. Bartoli, and J. Lavest. Energy-Based Reconstruction of 3D Curves for Quality Control. In *EMMCVPR*, 2007.

S. Meister, J. Hur, and S. Roth. UnFlow: Unsupervised Learning of Optical Flow with a Bidirectional Census Loss. *ArXiv*, 1711.07837, 2017.

Y. Mileva, A. Bruhn, and J. Weickert. Illumination-Robust Variational Optical Flow with Photometric Invariants. In *DAGM Conference on Pattern Recognition*, 2007.

F. Moreno-Noguer, M. Salzmann, V. Lepetit, and P. Fua. Capturing 3D Stretchable Surfaces from Single Images in Closed Form. In *CVPR*, 2009.

T. D. Ngo, S. Park, A. A. Jorstad, A. Crivellaro, C. Yoo, and P. Fua. Dense image registration and deformable surface reconstruction in presence of occlusions and minimal texture. In *ICCV*, 2015.

T. D. Ngo, J. Östlund, and P. Fua. Template-based Monocular 3D Shape Recovery using Laplacian Meshes. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 38 (1):172–187, 2016.

T. Okatani and K. Deguchi. Shape Reconstruction from an Endoscope Image by SfS Technique for a Point Light Source at the Projection Center. *Computer Vision and Image Understanding*, 66(2):119–131, July 1996.

S. I. Olsen and A. Bartoli. Implicit Non-Rigid Structure-from-Motion with Priors. *Journal of Mathematical Imaging and Vision*, 31(2):233–244, Jul 2008.

E. Özgür and A. Bartoli. Particle-SfT: A Provably-Convergent, Fast Shape-from-Template Algorithm. *International Journal of Computer Vision*, 123(2):184–205, June 2017.

N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly Accurate Optic Flow Computation with Theoretically Justified Warping. *International Journal of Computer Vision*, 67(2):141–158, Apr 2006.

S. Parashar, D. Pizarro, A. Bartoli, and T. Collins. As-Rigid-As-Possible Volumetric Shape-from-Template. In *ICCV*, 2015.

S. Parashar, D. Pizarro, and A. Bartoli. Isometric Non-rigid Shape-from-Motion in Linear Time. In *CVPR*, 2016.

M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas. Example-Based 3D Scan Completion. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, 2005.

A. P. Pentland. Local Shading Analysis. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 6(2):170–187, February 1984.

A. P. Pentland. Shape Information From Shading: A Theory About Human Perception. In *ICCV*, 1988.

M. Perriollat, R. Hartley, and A. Bartoli. Monocular Template-Based Reconstruction of Inextensible Surfaces. In *BMVC*, 2008.

M. Perriollat, R. Hartley, and A. Bartoli. Monocular Template-Based Reconstruction of Inextensible Surfaces. *International Journal of Computer Vision*, 95(2):124–137, November 2011.

J. Pilet, V. Lepetit, and P. Fua. Fast Non-Rigid Surface Detection, Registration and Realistic Augmentation. *International Journal of Computer Vision*, 76(2):109–122, 2007.

D. Pizarro and A. Bartoli. Feature-Based Deformable Surface Detection with Self-Occlusion Reasoning. *International Journal of Computer Vision*, 97(1):54–70, March 2012.

Point Grey. Flea2G 1.3 MP Color Firewire 1394b (Sony ICX445). https://www.ptgrey.com.

E. Prados and O. Faugeras. Shape From Shading: a well-posed problem? In *CVPR*, 2005.

A. Pumarola, A. Agudo, L. Porzi, A. Sanfeliu, V. Lepetit, and F. Moreno-Noguer. Geometry-Aware Network for Non-Rigid Shape Prediction from a Single View. In *CVPR*, 2018.

L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. IEEE*, 77(2):257–286, February 1989.

X. Ren and J. Malik. Learning a Classification Model for Segmentation. In *ICCV*, 2003.

Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha. Unsupervised Deep Learning for Optical Flow Estimation. In *AAAI Conference on Artificial Intelligence*, 2017.

J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *Computer Vision and Pattern Recognition*, 2015.

E. Richardson, M. Sela, and R. Kimmel. 3D Face Reconstruction by Learning from Synthetic Data. In *3DV*, 2016.

S. R. Richter and S. Roth. Discriminative Shape from Shading in Uncalibrated Illumination. In *CVPR*, 2015.

E. Rouy and A. Tourin. A Viscosity Solutions Approach to Shape-from-Shading. *SIAM J. Numer. Anal.*, 29(3):867–884, June 1992.

M. Salzmann and P. Fua. Reconstructing Sharply Folding Surfaces: A Convex Formulation. In *CVPR*, 2009.

M. Salzmann and P. Fua. Linear Local Models for Monocular Reconstruction of Deformable Surfaces. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33(5):931–944, May 2011.

M. Salzmann, R. Hartley, and P. Fua. Convex Optimization for Deformable Surface 3D Tracking. In *ICCV*, 2007a.

M. Salzmann, J. Pilet, S. Ilic, and P. Fua. Surface Deformation Models for Nonrigid 3D Shape Recovery. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 29(8): 1481–1487, August 2007b.

M. Salzmann, M. Urtasun, and P. Fua. Local Deformation Models for Monocular 3D Shape Recovery. In *CVPR*, 2008.

A. Saxena, M. Sun, and A. Y. Ng. Make3D: Learning 3D Scene Structure from a Single Still Image. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 31(5):824–840, May 2009.

M. Schmidt. UGM: A Matlab toolbox for probabilistic undirected graphical models. http://www.cs.ubc.ca/~schmidtm/Software, 2007.

L. Smith, D. Nevins, N. T. Dat, and P. Fua. Measuring the accuracy of softball impact simulations. *Sports Engineering*, 19(4):265–272, December 2016.

O. Sorkine and M. Alexa. As-Rigid-As-Possible Surface Modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, pages 109–116, 2007.

J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Version 1.05 available from http://fewcal.kub.nl/sturm.

A. Tankus, N. Sochen, and Y. Yeshurun. Shape-from-Shading Under Perspective Projection. *International Journal of Computer Vision*, 63(1):21–43, June 2005.

H. L. Taylor, S. C. Banks, and J. F. McCoy. Deconvolution with the l-1 Norm. *Geophysics*, 44(1):39–52, 1979.

J. Taylor, A. D. Jepson, and K. Kutulakos. Non-Rigid Structure from Locally-Rigid Motion. In *CVPR*, 2010.

J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In *CVPR*, 2016.

L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler. Tracking and Modeling Non-Rigid Objects with Rank Constraints. In *CVPR*, 2001.

L. Torresani, A. Hertzmann, and C. Bregler. Nonrigid Structure-from-Motion: Estimating Shape and Motion with Hierarchical Priors. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 30(5):878–892, 2008a.

L. Torresani, V. Kolmogorov, and C. Rother. Feature Correspondence Via Graph Matching: Models and Global Optimization. In *ECCV*, 2008b.

Q.-H. Tran, T.-J. Chin, G. Carneiro, M. S. Brown, and D. Suter. In Defence of RANSAC for Outlier Rejection in Deformable Registration. In *ECCV*, 2012.

P.-S. Tsai and M. Shah. Shape from Shading Using Linear Approximation. *Image and Vision Computing*, 12(8):487–498, 1994.

TurboSquid. http://www.turbosquid.com, 2016.

L. Valgaerts, C. Wu, A. Bruhn, H.-P. Seidel, and C. Theobalt. Lightweight Binocular Facial Performance Capture under Uncontrolled Lighting. In SIGGRAPH, 2012.

A. Varol, M. Salzmann, E. Tola, and P. Fua. Template-Free Monocular Rconstruction of Deformable Surfaces. In *ICCV*, 2009.

A. Varol, M. Salzmann, P. Fua, and R. Urtasun. A constrained latent variable model. In *CVPR*, 2012a.

A. Varol, A. Shaji, M. Salzmann, and P. Fua. Monocular 3D Reconstruction of Locally Textured Surfaces. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 34 (6), 2012b.

S. Vicente and L. Agapito. Soft Inextensibility Constraints for Template-Free Non-rigid Reconstruction. In *ECCV*, 2012.

S. Vicente and L. Agapito. Balloon Shapes: Reconstructing and Deforming Objects with Volume from Images. In *3DV*, 2013.

S. Vicente, C. Rother, and V. Kolmogorov. Object Cosegmentation. In *CVPR*, 2011.

T. Wang and J. Collomosse. Probabilistic Motion Diffusion of Labeling Priors for Coherent Video Segmentation. *IEEE Transactions on Multimedia*, 14(2):389–400, April 2012.

X. Wang, M. Salzmann, F. Wang, and J. Zhao. Template-Free 3D Reconstruction of Poorly-Textured Nonrigid Surfaces. In *ECCV*, 2016.

D. Warehouse. https://3dwarehouse.sketchup.com, 2016.

A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers. *An Improved Algorithm for TV-L1 Optical Flow*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

C. Wu. VisualSFM: A visual structure from motion system. http://ccwu.me/vsfm, 2011.

C. Wu, S. G. Narasimhan, and B. Jaramaz. Shape-from-Shading under Near Point Lighting and Partial views for Orthopedic Endoscopy. In *Workshop on Photometric Analysis For Computer Vision*, 2007.

C. Wu, S. G. Narasimhan, and B. Jaramaz. A Multi-Image Shape-from-Shading Framework for Near-Lighting Perspective Endoscopes. *International Journal of Computer Vision*, 86 (2):211–228, 2010.

C. Wu, K. Varanasi, Y. Liu, H. P. Seidel, and C. Theobalt. Shading-based Dynamic Shape Refinement from Multi-view Video under General Illumination. In *ICCV*, 2011.

J. Xiao, J. Chai, and T. Kanade. A Closed-Form Solution to Non-Rigid Shape and Motion Recovery. In *ECCV*, 2004.

Y. Xiong, A. Chakrabarti, R. Basri, S. J. Gortler, D. W. Jacobs, and T. Zickler. From Shading to Local Shape. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 37(1), 2015.

R. Yu, C. Russell, N. D. F. Campbell, and L. Agapito. Direct, Dense, and Deformable: Template-Based Non-rigid 3D Reconstruction from RGB Video. In *ICCV*, 2015.

C. Zach, T. Pock, and H. Bischof. A Duality Based Approach for Realtime TV-L1 Optical Flow. In *DAGM Conference on Pattern Recognition*, 2007.

Z. Zhang. Parameter Estimation Techniques: a Tutorial with Application to Conic Fitting. *Image and Vision Computing*, 15(1):59–76, 1997.

Z. Zhou, Z. Wu, and P. Tan. Multi-view Photometric Stereo with Spatially Varying Isotropic Materials. In *CVPR*, 2013.