Motion Capture of Deformable Surfaces in Multi-View Studios

Cédric Cagniart

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques et Informatique**

Arrêté ministérial : 7 août 2006

Présentée par

Cédric Cagniart

Thèse dirigée par **Edmond Boyer**
et codirigée par **Slobodan Ilic**

préparée au sein de l' **Inria Grenoble, Laboratoire Jean Kuntzmann**
et de l'école doctorale **Mathématiques, Sciences et Technologies de l'Information, Informatique**

Motion Capture of Deformable Surfaces in Multi-View Studios.

Thèse soutenue publiquement le **16 juillet 2012**,
devant le jury composé de :

Mr Radu Horaud
Directeur de Recherche, INRIA, Président
Mr Adrian Hilton
Professor, University of Surrey, Rapporteur
Mr Christian Theobalt
Professor, Max Planck Institüt Informatik, Saarbrücken, Rapporteur
Mr Edmond Boyer
Directeur de Recherche, INRIA, Directeur de thèse
Mr Slobodan Ilic
Doctor, Technische Universität München, Co-Directeur de thèse
Mr Nassir Navab
Professor, Technische Universität München, Examinateur

Résumé

Cette thèse traite du suivi temporel de surfaces déformables. Ces surfaces sont observées depuis plusieurs points de vue par des caméras qui capturent l'évolution de la scène et l'enregistrent sous la forme de vidéos. Du fait des progrès récents en reconstruction multi-vue, cet ensemble de vidéos peut être converti en une série de clichés tridimensionnels qui capturent l'apparence et la forme des objets dans la scène.

Le problème au coeur des travaux rapportés par cette thèse est de complémenter les informations d'apparence et de forme avec des informations sur les mouvements et les déformations des objets. En d'autres mots, il s'agit de mesurer la trajectoire de chacun des points sur les surfaces observées. Ceci est un problème difficile car les vidéos capturées ne sont que des séquences d'images, et car les formes reconstruites à chaque instant le sont indépendemment les unes des autres. Si le cerveau humain excelle à recréer l'illusion de mouvement à partir de ces clichés, leur utilisation pour la mesure automatisée du mouvement reste une question largement ouverte. La majorité des précédents travaux sur le sujet se sont focalisés sur la capture du mouvement humain et ont bénéficié de la nature articulée de ce mouvement qui pouvait être utilisé comme a-priori dans les calculs. La spécificité des développements présentés ici réside dans la généricité des méthodes qui permettent de capturer des scènes dynamiques plus complexes contenant plusieurs acteurs et différents objets déformables de nature inconnue a priori.

Pour suivre les surfaces de la façon la plus générique possible, nous formulons le problème comme celui de l'alignement géométrique de surfaces, et déformons un maillage de référence pour l'aligner avec les maillages indépendemment reconstruits de la séquence. Nous présentons un ensemble d'algorithmes et d'outils numériques intégrés dans une chaîne de traitements dont le résultat est un maillage animé. Notre première contribution est une méthode de déformation de maillage qui divise la surface en une collection de morceaux élémentaires de surfaces que nous nommons patches. Ces patches sont organisés dans un graphe de déformation, et une force est appliquée sur cette structure pour émuler une déformation élastique par rapport à la pose de référence. Comme seconde contribution, nous présentons une formulation probabiliste de l'alignement de surfaces déformables qui modélise explicitement le bruit dans le processus d'acquisition. Pour finir, nous étudions dans quelle mesure les a-prioris sur la nature articulée du mouvement peuvent aider, et comparons différents modèles de déformation à une méthode de suivi de squelette.

Les développements rapportés par cette thèse sont validés par de nombreuses expériences sur une variété de séquences. Ces résultats montrent qu'en dépit d'a-prioris moins forts sur les surfaces suivies, les idées présentées permettent de traiter des scènes complexes contenant de multiples objets tout en se comportant de façon robuste vis-a-vis de données fragmentaires et d'erreurs de reconstruction.

Mots-clefs : suivi de surfaces déformables, multi-vues, scène dynamique, alignement de surfaces, Espérance-Maximisation, EM.

Abstract

In this thesis we address the problem of digitizing the motion of three-dimensional shapes that move and deform in time. These shapes are observed from several points of view with cameras that record the scene's evolution as videos. Using available reconstruction methods, these videos can be converted into a sequence of three-dimensional snapshots that capture the appearance and shape of the objects in the scene.

The focus of this thesis is to complement appearance and shape with information on the motion and deformation of objects. In other words, we want to measure the trajectory of every point on the observed surfaces. This is a challenging problem because the captured videos are only sequences of images, and the reconstructed shapes are built independently from each other. While the human brain excels at recreating the illusion of motion from these snapshots, using them to automatically measure motion is still largely an open problem. The majority of prior works on the subject has focused on tracking the performance of one human actor, and used the strong prior knowledge on the articulated nature of human motion to handle the ambiguity and noise inherent to visual data. In contrast, the presented developments consist of generic methods that allow to digitize scenes involving several humans and deformable objects of arbitrary nature.

To perform surface tracking as generically as possible, we formulate the problem as the geometric registration of surfaces and deform a reference mesh to fit a sequence of independently reconstructed meshes. We introduce a set of algorithms and numerical tools that integrate into a pipeline whose output is an animated mesh. Our first contribution consists of a generic mesh deformation model and numerical optimization framework that divides the tracked surface into a collection of patches, organizes these patches in a deformation graph and emulates elastic behavior with respect to the reference pose. As a second contribution, we present a probabilistic formulation of deformable surface registration that embeds the inference in an Expectation-Maximization framework that explicitly accounts for the noise and in the acquisition. As a third contribution, we look at how prior knowledge can be used when tracking articulated objects, and compare different deformation model with skeletal-based tracking.

The studies reported by this thesis are supported by extensive experiments on various 4D datasets. They show that in spite of weaker assumption on the nature of the tracked objects, the presented ideas allow to process complex scenes involving several arbitrary objects, while robustly handling missing data and relatively large reconstruction artifacts.

Keywords: deformable surface tracking, multi-view, dynamic scene, deformable registration, Expectation-Maximization, EM.

Remerciements

J'adresse tout d'abord mes remerciements aux membres du jury pour les points de vue pertinents qu'ils ont apportés sur le travail présenté ici. Merci à Adrian Hilton et Christian Theobalt d'avoir accepté la charge de rapporteur. Merci à mes directeur de thèse Edmond Boyer et Slobodan Ilic, ainsi qu'à Nassir Navab d'avoir été examinateurs. Merci à Radu Horaud d'avoir présidé le jury.

Merci tout particulièrement à mes deux directeurs de thèse de m'avoir donné la chance de travailler sur un sujet passionnant, pour leurs conseils, leurs encouragements et pour tout ce que j'ai appris à leur contact ces dernières années. Merci ensuite à Sebastian Möller et Nassir Navab de m'avoir permis de mener mes travaux de recherche dans leurs groupes respectifs au sein des universités techniques de Berlin et Munich. Merci aussi aux groupes Perception et Morpheo a l'INRIA Rhône-Alpes. Travailler dans ces différents laboratoires a fait de ma thèse une expérience très riche, d'un point scientifique mais aussi personnel. Merci aussi à Deutsche Telekom Laboratories pour le soutien financier apporté à mes travaux.

Un grand merci également a tous mes collègues de bureau à Berlin, Munich, Grenoble et Rennes pour les enrichissantes discussions et les bons moments passés avec eux.

Merci finalement à mes parents et mes frères d'avoir accepté la charge de second comité de thèse et d'avoir fait le déplacement pour ma défense.

Contents

CHAPTER 1

Introduction

Over the last few decades, the cost of image sensors and computing capabilities has been significantly cut down. As a consequence, digital cameras have become ubiquitous and the ability to effortlessly record, store and transmit snapshots of the world has become a familiar part of our day-to-day lives. However, such uses constitute only a fraction of the possibilities opened by the available technologies. Beyond the acquisition, storage and rendition of appearance enabled by photography, there is a need for tools that automatically measure and interpret the world underlying the pictures. Computer vision concerns itself with these challenges. As such, one might say that while photography emulates the perceptual process of vision, computer vision strives to replicate our cognitive evaluation of reality.

A significant part of computer vision is dedicated to measuring shape from visual data. In particular, considerable efforts have been made to contrive algorithms that automatically build three dimensional models of objects that were observed from multiple views. The problem has been approached from many directions and multi-view 3D reconstruction has grown into a mature topic in the computer vision community. Yet when applied to multiple videos of moving and deforming objects, most of the available methods will treat each frame independently, ignoring the dynamic nature of the observed event and thus the temporal redundancy in the data.

This way of digitizing dynamic scenes is in fact the 3D extension of the cinematographic process, that simply records and renders a rapid succession of snapshots of the scene but does not measure motion itself. The dynamic information is only implicitly represented and human brain still has to interpret the series of snapshot to recreate the illusion of continuous motion.

Automatically measuring motion from visual data remains a challenging and fundamental task of computer vision. Typically, the movement between two images of a video sequence is described as a two-dimensional vector field called *optical flow*. However, because optical flow is usually computed from appearance exclusively, it only captures the

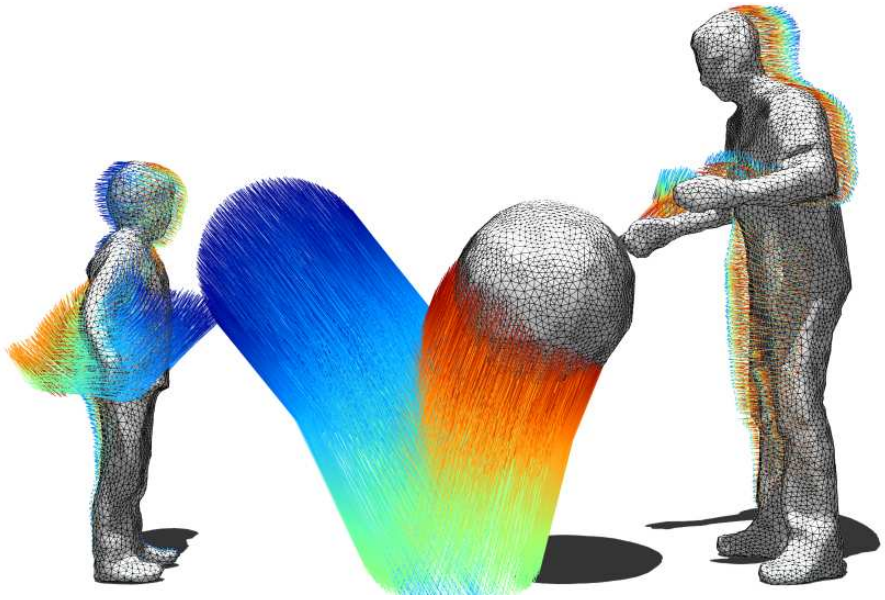


Figure 1.1: In this thesis, we build 4D models from visual data by augmenting the results of 3D reconstruction methods with temporal trajectories for every point of the reconstructed objects.

displacements of brightness patterns in image space. In their seminal work of 1999, Vedula et al. [11] introduced the term *scene flow* to describe the three-dimensional vector field encoding the motion of every point on the observed surfaces. After exploring the connections between optical flow and scene flow, they conclude by hinting that computing scene flow without first resorting to optical flow should a promising path for further research. Indeed, if more information is available on the scene than its appearance, computing optical flow from the appearance alone makes little sense.

The core interest of this thesis is the inference for motion in 3D space. We build on the recent significant progress made on passive 3D reconstruction from multiple views that provide us with *snapshots* of the scene structure, and explore the possibilities opened when such information is available. Our interest is to simultaneously digitize shape and motion, that is to advance to a more complete representation of dynamic scenes that complements the recovered deformable 3D shapes with their temporal evolution. The goal is to automatically reconstruct animated shapes rather than perform series of static shape reconstructions. We refer to such digital representations of the captured scenes as space-time models, or *4D models*.

1 Applications of 4D Capture

The automatic computation of four-dimensional descriptions of scenes has a wide range of applications. In this section we list a number of topics which we feel are already impacted or soon to be changed by the recent progress on the subject, among which the contributions of this thesis.

- **Content production** The recent progress made on 3D reconstruction suggests that footage need not be processed as series of 2D images. Manipulating directly 3D shapes instead of their 2D projections indeed allows much more efficiency and freedom in the artistic process. For example, this allows to composite an object into a new scene with different illumination conditions, then to render the whole from a novel point of view, with a camera that has a different focal length and depth of field.

  Furthermore, CGI artists have needs that exceed the simple acquisition and rendition of 3D data. They need to edit this acquired content. Consider for example the task of realistically adding a virtual logo to the shirt of an actor. Editing the 2D video frame-by-frame to draw the logo would be extremely tedious, as one needs to account for occlusions, folds of the shirt, self-cast shadows and illumination conditions. Drawing on a series of 3D meshes frame-by-frame would solve part of these issues but remain very impractical. If a *4D model* of the same performance is available, completing the same task becomes be much less tedious. Because *4D models* contain the trajectories of each point of the shirt across time, the logo can be added to the first frame, and its deformation across time can be automatically computed. In other words, this information allows to *automatically propagate edits through time*. Moreover, the possibilities opened by 4D models for edition are not limited to the appearance of the objects. The geometry itself can be modified in a temporally consistent fashion and the actor can be made taller or skinnier [3]. The motion from the captured performance can also be transfered to a new character [2, 9].

  Finally, it also becomes possible to insert the 4D models into dynamic simulations. This allows for example to capture dynamic objects and make them influence simulated particle systems, or to give an actor virtual long hair or clothing that will interact more realistically with his captured body.

- **Compression, transmission, and real-time rendering** 4D models contain explicit information on the motion of objects, and therefore on the temporal redundancy in the data. This exposed redundancy can be compressed, yielding much more compact descriptions of dynamic scenes. Consider the limit case of the rigid motion of an object. With a series of 3D models, the scene is spatially discretized at each frame. If a 4D model is available, one can encode the shape information once, and only transmit one rigid transformation per frame. Such concise representations are of evident interest suited for storage, transmission over networks and real-time rendering.

- **Human-Machine interaction** Capturing movement at interactive frame rates opens possibilities for human machine interaction. Measured body poses can be used as user input to provide intuitive manipulation of virtual objects or natural interaction with virtual agents. Beyond simple measurement, computed motion cues allow to recognize body

gestures or complex actions and in some sense to use temporal information to evaluate the context and semantical content of 4D data.

- **Medical applications** Recovering shape and motion has also many potential medical applications. For example, measuring the 3D motion of athletes allows for the biomechanical analysis of skilled movement such as a golf swing. It is also useful to measure the evolution of a gait pattern during physiotherapy or to automatically detect, record and document the evolution of patients for pathologies such as epileptic seizures. Motion capture also has applications for the observation of medical personnel. It allows to automatically document procedures and compare them to established work flow. This can save precious time when writing reports and help with the training of new staff. Furthermore, knowing the current stage of a procedure can help predict the end of a surgical intervention and get the next patient ready so that the usage of the operating room can be maximized. Other applications of interest include collision avoidance [6] or the estimation of the cumulated exposure to radiation of personnel in contact with x-ray sources [7] (our work on the subject is presented in chapter 6).

2 4D Capture: Beyond Marker-based Skeletal Animation

The research presented in this dissertation was for the most part supported by *Deutsche Telekom Laboratories* and linked to its *Free View-Point TV* activities. Because of this link to industrial requirements, most of our studies were steered towards the problem of animated 3D content production.

Limitations of marker-based motion capture The established pipelines for the production of animated 3D content are currently marker-based. This means that they compute the trajectories of a restricted set of optical markers that are relatively easy to detect and track along sequences. In figure 1.2 we show a typical motion capture environment used in the production of movies and games. The tight black suits worn by the actors show that marker-based techniques come with a number of drawbacks:

- Attaching markers is tedious and requires to be done before each acquisition.
- Markers can interfere with the movement.
- Markers prevent the *simultaneous acquisition of appearance, shape, and motion.*

As such, marker-less motion capture has been the subject of a large body of work, mostly focused on human motion [8, 10].

Limitations of skeletal models The vast majority of human motion capture system evaluate body movement in terms of joint angles on a kinematic tree representing the human

Figure 1.2: Optical Motion Capture on the set of LA Noire (©Rockstar Games).

skeleton. The limited expressive ranges of these articulated structures confers a lot of robustness to the inference of body poses by constraining the solution space. But it also severely narrows the nature of motions that can be recovered. In particular, capturing the deformation of loose clothing such as skirts is beyond reach. Fine deformations of the human body are also lost, which can significantly hinder the realism of the recovered animation. The limitations of thinking of the human body as an articulated structures are for instance exposed in *The Illusion of Life: Disney Animation* [4]. In this book, two animators recall the introduction of live footage in the animation process at the Disney Studios. Each frame of the film was printed to a sheet and these sheets were pinned together to the pegs of an animation desk so that the animator could flip through them and study the movement.

> We were amazed at what we saw. The human form in movement displayed far more overall activity than anyone had supposed. It was not just the chest working against hips, or the backbone bending around, it was the very bulk of the body pulling in, pushing out, stretching, protruding.
>
> Frank Thomas and Ollie Johnston

They explain how this confirmed what they had previously empirically discovered as one of the keys to drawing life-like animations.

> The loose flesh on a figure, [...] will move at a slower speed than the skeletal parts. This trailing behind in an action is sometimes called “drag,” and it gives a looseness and a solidity to that figure that is vital to the feeling of life.
>
> Frank Thomas and Ollie Johnston

Because of the limitations of skeletal-based motion capture, such small yet valuable details are lost during the performance capture, and usually have to be re-synthesized in later stages

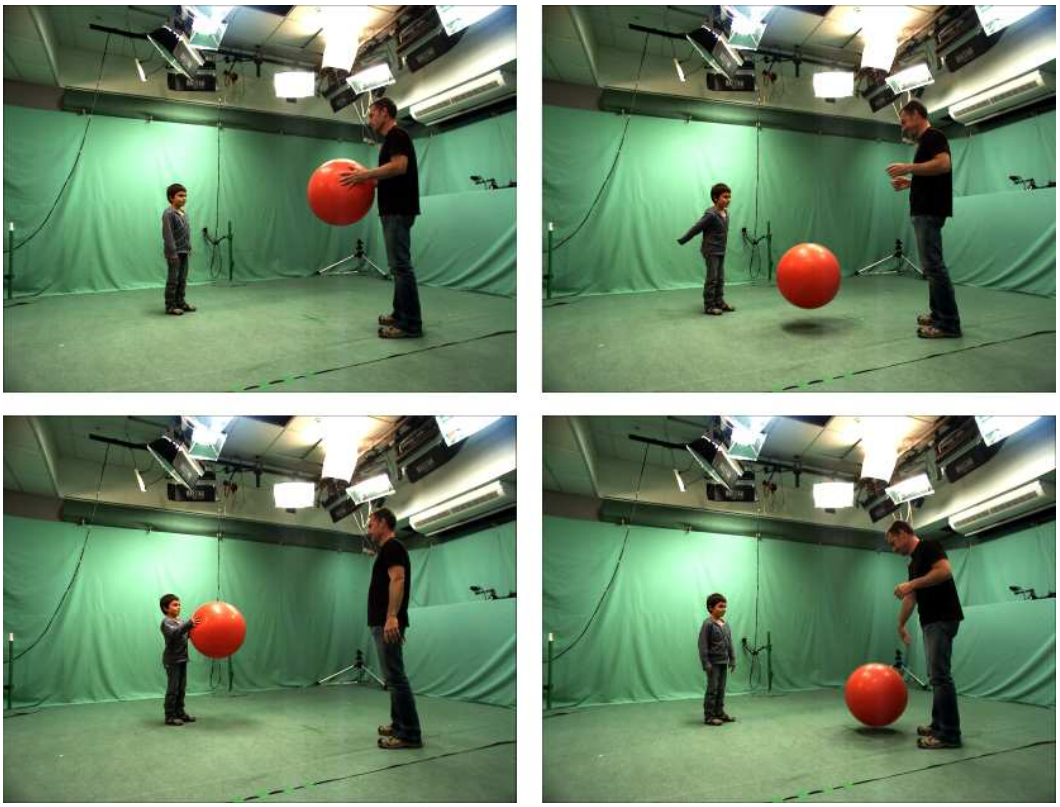


Figure 1.3: The Ball sequence from INRIA illustrates the need for non skeletal-based data-driven mesh animation methods, as the movement and deformation of the ball are of interest but can not be modeled by an articulated structure.

of production from databases of previous observations or by physical simulation techniques

More fundamentally, skeletal methods are limited to capturing humans or other rare articulated structures. They were not designed to handle more complex scenes involving several objects of arbitrary nature. A complete spatio-temporal description of the scene presented in figure 1.3 for example should include the motion and deformation of the two persons, but also of the ball. Skeletal-based methods are insufficient in these more elaborate settings, which motivates our research for new, more general solutions to marker-less motion-capture.

Objectives and technical context The work presented in this thesis neither uses markers nor articulated models. The goals are to avoid the tedious fastening of markers, to allow as much freedom as possible in terms of what kind of objects and movements can be captured, and to minimize the overall need for user intervention . In the long run, digitizing the 3D structure and temporal evolution of an event should be as easy as it is today to capture its appearance in a video. Our part in the ongoing research on the subject is the contribution of a set of developments aimed at:

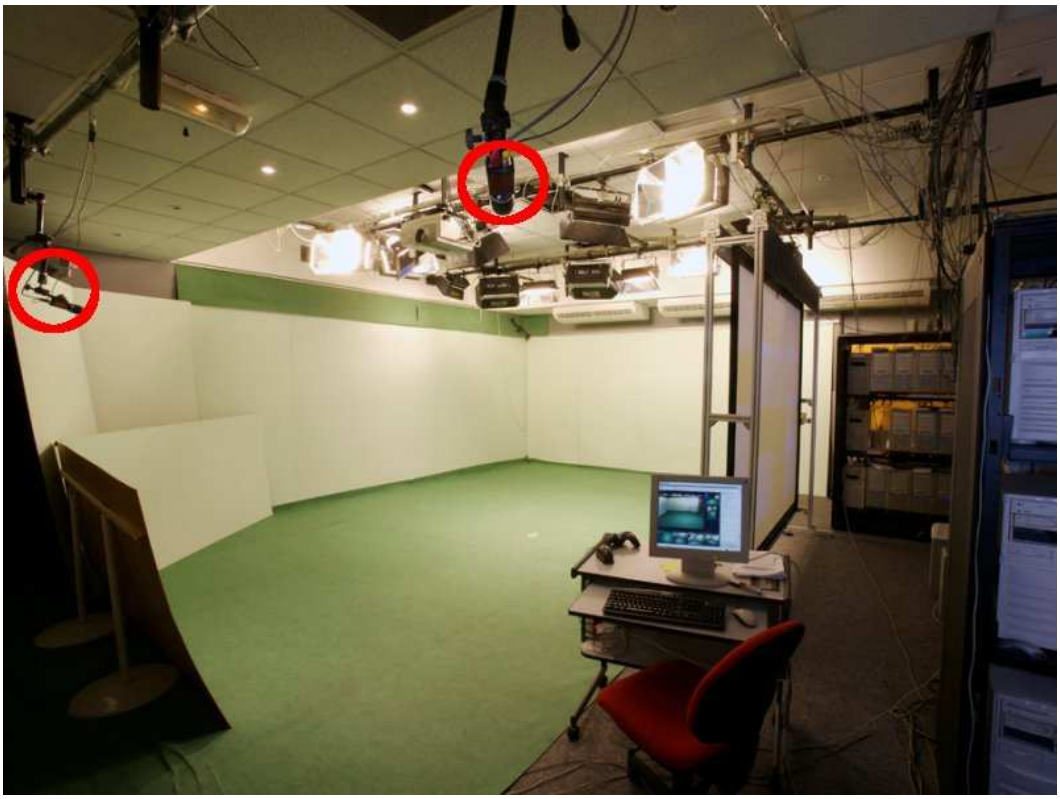


Figure 1.4: The Grimage plateform.

- the concurrent capture of the appearance, shape and motion in the scene.
- the capture of scenes involving several objects of different nature.

Our developments contribute to a body of research on the uses of multi-camera setups. These setups combine several cameras that are arranged around a volume of interest and that synchronously acquire videos of the scene from multiple points of view. The perspectives opened by such systems were identified more than 15 years ago, and one notable pioneering experiment was the CMU *3D dome* [5] that recorded data on analog VCRs. Since then, the improvements in camera technologies and storage capacity have been integrated in the development of several digital multi-view studios. We had access to three of them: the Grimage Platform in INRIA Rhône-Alpes, which is displayed in figure 1.4, a similar multi-camera system deployed at Deutsche Telekom Laboratories in Berlin, and a smaller scale system deployed a the Technical University of Munich. Such systems are relatively inexpensive to deploy and allow an unintrusive capture of performances from multiple views. Today, the progress on passive 3D reconstruction makes them emerge as a competitive solution to 3D content production [1]. Our goal in this thesis is to advance the state of the art to a point where these systems can be used for the full 4D digitization of performances.

3 Challenges and Contributions

The objectives that were established in the previous section delineate the desired outcome of our research, as well as its technical context. In this section, we identify the main scientific challenges to be met, and introduce the corresponding contributions of our work.

Visual data As previously stated, our data originally consists of images acquired in a multi-camera setup, and we rely on a number of existing reconstruction algorithms to obtain an estimation of the scene 3D structure. These reconstructions are not only degraded by noise and occlusion, they are also heavily biased by the simplified models used in low-level treatment of the image information and by the shortcomings of 3D reconstruction algorithms themselves. As such, our developments should robustly handle the artifacts that commonly arise in these pre-processing stages.

- Visual data is *inherently fractional* because of occlusion. Occlusion not only degrades the signal: it actually results in the absence of information on some parts of the shape. This raises considerable challenges when trying to track points of a deformable surface, as some parts of the surface might be observable at a certain time frames and not at others.
- Visual data is *often ambiguous*. By ambiguity we mean that points on the surface rarely have discriminative appearances. In figure 1.3, pretty much all the points on the father's t-shirt have the same uniform black appearance. In the same figure, it can also be seen that a point on the ball will greatly vary in appearance over time because of the strong specularity of the material. It clearly is a hard problem to follow points in space and time from images if their appearance varies so sharply and if large parts of the surface tend to have a uniform color.
- The information can be degraded in processing stages that precede the motion estimation. A typical example in the multi-view studio setting is the *segmentation* stage that separates the objects of interest from the background and sometime mislabels parts of the images. *3D Reconstruction algorithms* also have shortcomings when the appearance of objects is too ambiguous and can output shapes with significant artifacts.

Deformable models This thesis uses a *model-based* approach to recover motion. The role of a model in a surface tracking context is not only to parametrize the deformation of surfaces, but more importantly to define what is a plausible configuration of a deformable object. In other words, the deformable model encodes prior knowledge that allows the tracking algorithm to confine the search for surface evolution to the space of plausible configurations, and to steer its output towards more likely configurations. This effectively allows to increase robustness with respect to the challenges that come with visual data. One of the key issues explored by this thesis is the trade-off that deformable models offer between expressiveness and robustness. If they are too constraining or object-specific like articulated structures, they limit the nature of objects that can be tracked and the range of deformations that can be recovered. If they are too permissive or too general, they stop

playing their regularizing role: they no longer compensate effectively for ambiguous data and also can fail to prevent the over-fitting of erroneous data.

Contributions This dissertation is organized around two main complemental contributions aimed at recovering 4D models from multiple videos. The first contribution consists of a generic mesh deformation and numerical optimization framework that builds on ideas from the field of interactive deformation in computer graphics. This framework makes no assumption on the nature of the object and simply represents a reference surface as a collection of small surface elements, or patches to which the vertices of the original mesh are attached. These patches are each associated to a rigid transformation and organized in a *deformation graph* that controls the deformation with respect to the reference mesh. We show that while allowing large and complex deformations, this representation of the geometry and parametrization of the deformation are robust tools to tackle the recovery surface deformation from visual data. The inference for deformation benefits from the averaging effects of visual cues over each patch's surface. Furthermore, elastic forces are emulated on the deformation graph to constrain and regularize the recovered deformation when faced with fractional, ambiguous or erroneous data.

However, because this mesh deformation framework is generic and makes only weak assumptions on how the object deforms, we show as a second contribution that the deformation model can be complemented with a Probabilistic model of the data acquisition process that accounts for its uncertainty and errors. The surface tracking problem is then cast as the 3D registration of surfaces that we embed in an Expectation- Maximization framework. We show that this probabilistic formulation effectively manages artifacts in 3D reconstructions and has improved convergence properties.

Extensive experiments on various 4D datasets show that these two ideas allow to robustly handle missing data and relatively large reconstruction artifacts. More importantly, the novelty of our work appears through our results on complex scenes involving several objects of arbitrary nature, where previous art in the multi-camera setting had mostly aimed at tracking a single human actor. We additionally evaluate our approach on these simpler scenes, compare our results to these of methods that make much stronger assumptions on the nature of the object, and show that we perform comparatively well.

4 Thesis Outline

The remainder of this dissertation is structured as follows:

Chapter 2 recalls the technical and scientific background our work builds upon. Then an overview of the related works that address deformable surface tracking in multi-view

setups is presented. In other words, this section presents the prior art on our problem and puts our contributions in perspective.

Chapter 3 provides an overview of state-of-the-art methods on mesh manipulation and deformation. Then, our numerically robust mesh deformation framework is presented, as well as applicative cases that demonstrate its usefulness for the purpose data-driven mesh deformation and tracking.

Chapter 4 presents our developments on the animation of 3D surfaces by probabilistic registration. These developments build on the generic deformation model of the previous chapter to allow the tracking of complex scenes involving several objects of arbitrary, and a priori unknown nature. Furthermore, this chapter formalizes the problem in a Bayesian framework that increases the robustness to reconstruction artifacts.

Chapter 5 explores the applicability of the developments from the previous chapter to the tracking of articulated objects, and compares the results of our generic patch-based method with more constraining tracking algorithms. More specifically, this chapter looks at skeletal-based tracking and at an extension of mesh deformation model of chapter 3 that accounts for rigid clusters in the object.

Chapter 6 presents our conclusions, as well as a number of works that are already impacted by the presented developments, and discusses the perspectives for further research.

References

[1] 4D Views
. http://www.4dviews.com/.

[2] Baran, I., D. Vlasic, E. Grinspun, and J. Popović
2009. Semantic deformation transfer. *SIGGRAPH*.

[3] Jain, A., T. Thormählen, H.-P. Seidel, and C. Theobalt
2010. Moviereshape: Tracking and reshaping of humans in videos. *SIGGRAPH Asia*.

[4] Johnston, O. and F. Thomas
1981. *The Illusion of Life: Disney Animation*. Hyperion.

[5] Kanade, T., P. J. Narayanan, and P. W. Rander
1995. Virtualized reality: concepts and early results. In *Proceedings of the IEEE Workshop on Representation of Visual Scenes*, VSR '95.

[6] Ladikos, A., S. Benhimane, and N. Navab
2008. Real-time 3d reconstruction for collision avoidance in interventional environ-

ments. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*.

[7] Ladikos, A., C. Cagniart, R. Gothbi, M. Reiser, and N. Navab
2010. Estimating radiation exposure in interventional environments. In *MICCAI*.

[8] Moeslund, T. B., A. Hilton, and V. Krüger
2006. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*.

[9] Noh, J.-y. and U. Neumann
2006. Expression cloning. In *SIGGRAPH*.

[10] Poppe, R.
2007. Vision-based human motion analysis: An overview. *CVIU*.

[11] Vedula, S., S. Baker, R. Collins, T. Kanade, and P. Rander
1999. Three-dimensional scene flow. *Computer Vision, IEEE International Conference on*.

CHAPTER 2

Related Works

This chapter presents the technical and scientific context around the research reported by this dissertation. We articulate our discussion around three main topics. The first part is dedicated to shape, and presents the data acquisition studio and 3D reconstruction pipeline. In a second part, we address motion and present existing methods for establishing temporal correspondence between images or between 3D shapes. These methods can be integrated as building blocks for our purpose, and have been used by previous works that directly address surface tracking in multi-view setups. In the third part of this chapter, we discuss how these works have performed this integration and try to outline their limitations as well as the ideas that were the key to their successes. This allows motivate our choices of research direction and to put in perspective the contributions presented in the remainder of the thesis.

1 Inferring Shape

As explained in the introduction, the developments of this thesis are mostly targeted at multi-camera environments such as the Grimage Platform developed in INRIA Rhône-Alpes. In its latest version, Grimage can count up to 32 cameras that are distributed around a volume of interest. However most of our experiments were performed on smaller scale systems with 8 to 16 cameras. These cameras are calibrated, which means that their intrinsic characteristics and relative poses in spaces are known. A whole segment of the computer vision litterature is dedicated to calibration and several software packages [20, 39] are open-source. These cameras are also synchronized, which means that the snapshot provided by one frame acquired by the system consists of a set of images all taken at the same time.

In the next paragraphs, we give a brief overview of possible 3D reconstruction pipelines. Although the contributions of this thesis are not directly on this topic, our work heavily relies on 3D data as input and is therefore directly impacted by the performance, precision and limitations of static 3D reconstruction methods.

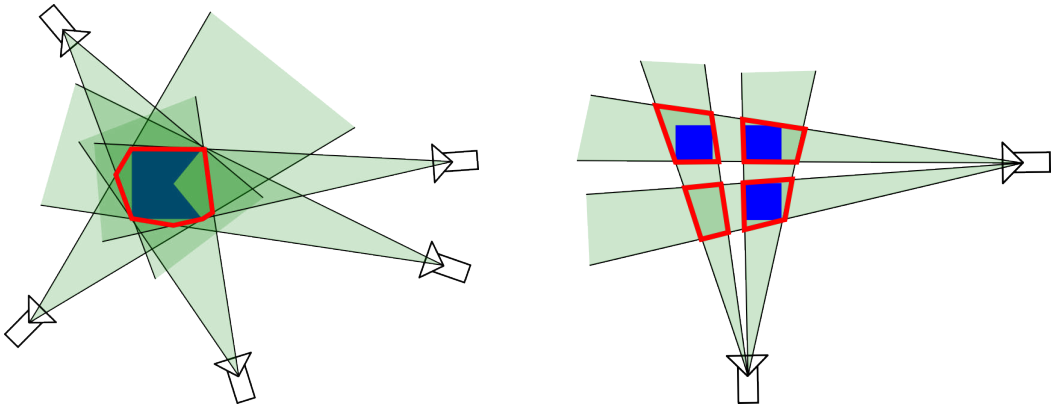
Background subtraction is in the majority of cases the first step of the reconstruction pipeline when dealing with data captured in a multi-camera studio. As shown in figure 1.3, the scene is often surrounded by painted walls and floor. Using blue or green as color allows to easily differentiate the background from skin-color for example and makes chroma-keying possible. It is also possible to use more elaborate background subtraction methods such as [18] and to learn from a sample of background frames a statistical model for the distribution of brightness and chromaticity components at each pixel. This can particularly help to handle the case of shadows which can be incorrectly segmented as foreground.

Shape-From-Silhouette is arguably the simplest paradigm for reconstruction from multi-view data. The idea is to look for the largest 3D volume consistent with the silhouettes of the object in the different views. This volume is named the *visual hull* [24] and although it only approximates the shape of interest, the estimate it provides is good enough for a number of purposes. Once textured for example, the visual hull is often a good geometry proxy that allows for convincing renderings. It is also a very good initialization point for more precise reconstruction algorithms as it provides a coarse estimate of the geometry and of the self-occlusions.

The main strengths of shape-from-silhouette are linked to its conceptual simplicity. First, basic algorithms are very straightforward to implement. Second, these simple approaches make no assumption on the nature or the smoothness of the observed object. This means that there is very little need for parameter tweaking beyond the foreground segmentation stage. The third and most important attractive characteristic of shape-from-silhouette methods is that several implementations [26, 12, 46, 23] were shown to be real-time capable.

Shape-from-silhouette algorithms however come with a number of restrictions. The main limitation appears when dealing with concavities in objects. Concavities such as the inside of a cup for example are not observable in a silhouette, regardless of the point-of-view on the object. Figure 2.1 shows a simple 2D example in which the shown concavity can not be recovered, no matter how many cameras are added to observe the object. A second limitation is the robustness of the visual hull computation to erroneous input. Not making assumptions on the smoothness of the observed shape confers a lot of genericity and limits the addition of artifacts by regularization terms but it also means that errors that arose in the pre-processing are not compensated for. Although some work has been done to increase the robustness of shape-from-silhouette [11, 34], more elaborate methods lose the simplicity and real-time capability that make these approaches attractive.

Some of the results presented in chapters 4 and 5 have been obtained using visual hulls as shape input data. We implemented a basic volumetric algorithm to perform recon-



(a) Shape-from-silhouette techniques can not recover concavities that do not appear in the silhouettes of the object.

(b) Too few cameras around the scene can result in reconstruction artifacts. In this situation the visuall hull contains an additional connected component that does not correspond to any object in the scene.

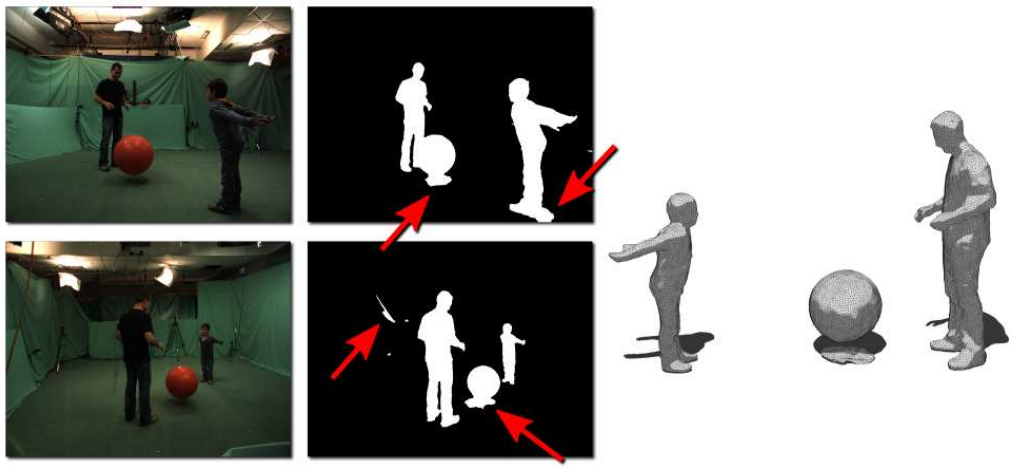
Figure 2.1: Limitations of shape-from-silhouette reconstruction methods. In these examples, the real 3D object is in blue, the cones corresponding to is silhouettes in green, and the visual hull in red.

struction from silhouettes. This algorithm simply processes every voxel of a 3D grid in a bounding box and checks whether its projection falls in or out of every silhouette. Because of errors in the background subtraction like those of figure 2.2 however, some of the reconstructions obtained by this simple approach exhibit consequent artifacts. The basketball sequence presented in figure 4.8 of the results section in chapter 4 is a particularly good illustration of the range of artifacts that can occur: it contains toplogical collapses, missing geometry due to erroneous segmentation and bad camera calibration, as well as outlying geometry due to non-recoverable concavities.

Stereo reconstruction methods do not limit themselves to the purely *geometric* silhouette information, but consider *photometric* information, which consists of the observed intensity values in the images. Stereo builds on the idea of *photoconsistency*. For example, if reflections are ignored, and a Lambertian model is assumed for the reflectance of the observed object, a 3D point of its surface should appear with the same color in every view and a matching score can be defined as the sum of squared (or absolute) differences (SSD, SAD) on a small image patches. More robust photoconsistency scores such as normalized cross-correlation (NCC) or mutual information (MI) can also be used [16].

Under the hypothesis of a calibrated binocular setup, the search for correspondence is traditionally constrained using a result from epipolar geometry: for a given point in the first image, a corresponding point in the second image necessarily lies on a well defined line called the epipolar-line [see 14, for an extensive study of epipolar geometry]. Thus, efficient algorithms reduce their search for correspondences to 1D scans along scanlines that can be aligned in memory in a rectification stage for even greater speedups. The computation of disparity is usually *regularized*, which means that the algorithm tries to

(a) The silhouette of the little boy gets incorrectly segmented as foreground. However, because of the redundancy brought by 16 cameras, these errors do not significantly impact the visual hull reconstruction for this particular frame, as shown on the right.



(b) The variation of illumination under the ball is large enough to create a background subtraction error in most of the cameras. In that case, parasite geometry appears in the reconstruction.

Figure 2.2: Examples of typical foreground segmentation artifacts in multi-view studios. The silhouettes are shown for two cameras and illustrate the limitations of background subtraction in areas that are weakly illuminated.

favor some consistence in the depth estimations of neighbouring pixels (see the survey of Scharstein and Szeliski [32] for more details). Challenges then arise to handle textureless objects, object boundaries, and occlusion. Several regularization schemes have been proposed, and the most successful approaches balance data fidelity and regularization in a global optimization on the whole image. In particular, a number of works have formulated the depth estimation problem as a labeling task, and given it a Bayesian modelisation by considering the depth images as Markov Random Fields (MRF). The labeling can then be run in approximate inference [40] algorithms based on graph-cuts or belief-propagation.

When many views of the object are available, it becomes possible to reconstruct a complete 3D model instead of a single depth-map. The problem is known as *Multi-View Stereo*, involves much longer baselines than binocular configurations, and therefore requires careful handling of occlusion. Seitz et al. [33] provide a recent and rather complete survey and taxonomy of the field. More recent and notable results for multi-camera studio usage were mostly linked to volumetric approaches using global optimizations with graph cuts [38] or gradient descent of convex energies [22, 47].

Some of the results presented in in chapters 4 and 5 have been obtained using the output of the multi-view stereo algorithm of Starck and Hilton [37]. This algorithm first computes the visual hull. It then performs a global optimization based on volumetric graph-cuts inside the visual hull to maximize photoconsistency and surface smoothness while respecting a number of edge features detected in the images.

2 Inferring Motion

The previous section has presented how 3D shape could be recovered from the multi-view studio. However, the focus of this thesis is motion: we are interested in establishing dense temporal trajectories for every point at the surface of deforming objects, as shown in figure 1.1. As such, the key issue can be thought of as finding reliable *correspondence* between two observations of a deformable object at different times.

Finding correspondence in images, 3D shapes or any other type of visual data is a fundamental task of computer vision. The computation of correspondences is actually found at the heart of most computer vision problems. As we have seen in the previous section, stereo reconstruction for example requires correspondences between multiple views of a scene. Similarly, motion estimation requires correspondences between two frames of a temporal sequence.

The rest of this section is organized around a brief overview of the existing paradigms for *establishing* such correspondences. We begin with methods that try to establish dense correspondence from the appearance of the object directly. Then we look at methods that

match geometry, and do so globally. They offer a top-down point of view on the problem by matching 3D shape as a whole to later obtain dense correspondences. We finally adopt a bottom-up point-of-view by considering methods that match sparse points between two images or two shapes, relying on following regularization mechanisms whose role is to prune erroneous matches and diffuse the sparse correspondence information to obtain a dense correspondence between the two surfaces.

2.1 Dense appearance matching

Let us consider the sequences of images provided by each camera. The dense correspondence between two consecutive images from each of these videos is known in the litterature as *optical flow*. As we already recalled in the introduction, the optical flow is a 2D vector field in the image plane that describes the movement of brightness patterns between two frames. If we make the assumption that a surface point will project with the same brightness in both images the optical flow can be identified with the 2D motion flow that is the projection of a 3D vector field describing the motion of surfaces in 3D space [43]. This vector field is known as *scene flow* and is the information we are interested in.

The computation of optical flow has been the subject of significant research effort over the last decades and the resulting advances are well summarized in the recent survey of Baker et al. [2]. Despite the impressive progress that was made on optical flow computation however, computing a reliable scene flow from multiple videos is not as straightforward as backprojecting the optical flow of every camera using the equations of [43]. This is explained by two reasons: first, optical flow only measures variations of appearance, and not the projection of scene flow directly. These variations of appearance can be caused by a multitude of factors such as illumination, surface orientation and reflectance properties and radiometric properties of the camera. Second, optical flow algorithms constrain their output by enforcing regularity properties on the flow field. However this regularization happens in 2D and does not account for the structure of the 3D scene, where the physical events underlying the observations take place. As reported by [2], a number of algorithms use robust norms in their penalization of irregularities to account for sharp variations of appearance in the image and prevent oversmoothing at occlusion boundaries. To some extent, this process amounts to obtain more precise regularity constraints by trying to guess information on the scene structure from its appearance.

In our case, we have access to the scene structure. It is therefore tempting to avoid the effects of potentially error-prone 2D regularizations and regularize the scene flow directly in 3D. For example Pons et al. [30] compute the scene flow from the appearance and 3D structure of the scene and propose in a continuous formulation to minimize the harmonic energy of the flow over the known surfaces. We'll discuss in chapter 3 other regularization energies that have been proposed to control the deformation of discrete meshes. Here, we note that regularization models can only help whith ambiguous appearance information up to a point. As already mentioned in the introduction, objects are rarely textured, and they tend to have large portions of their surfaces with uniform colors. Furthermore, the appearance of each point also varies from one frame to the next. Modeling for some lighting

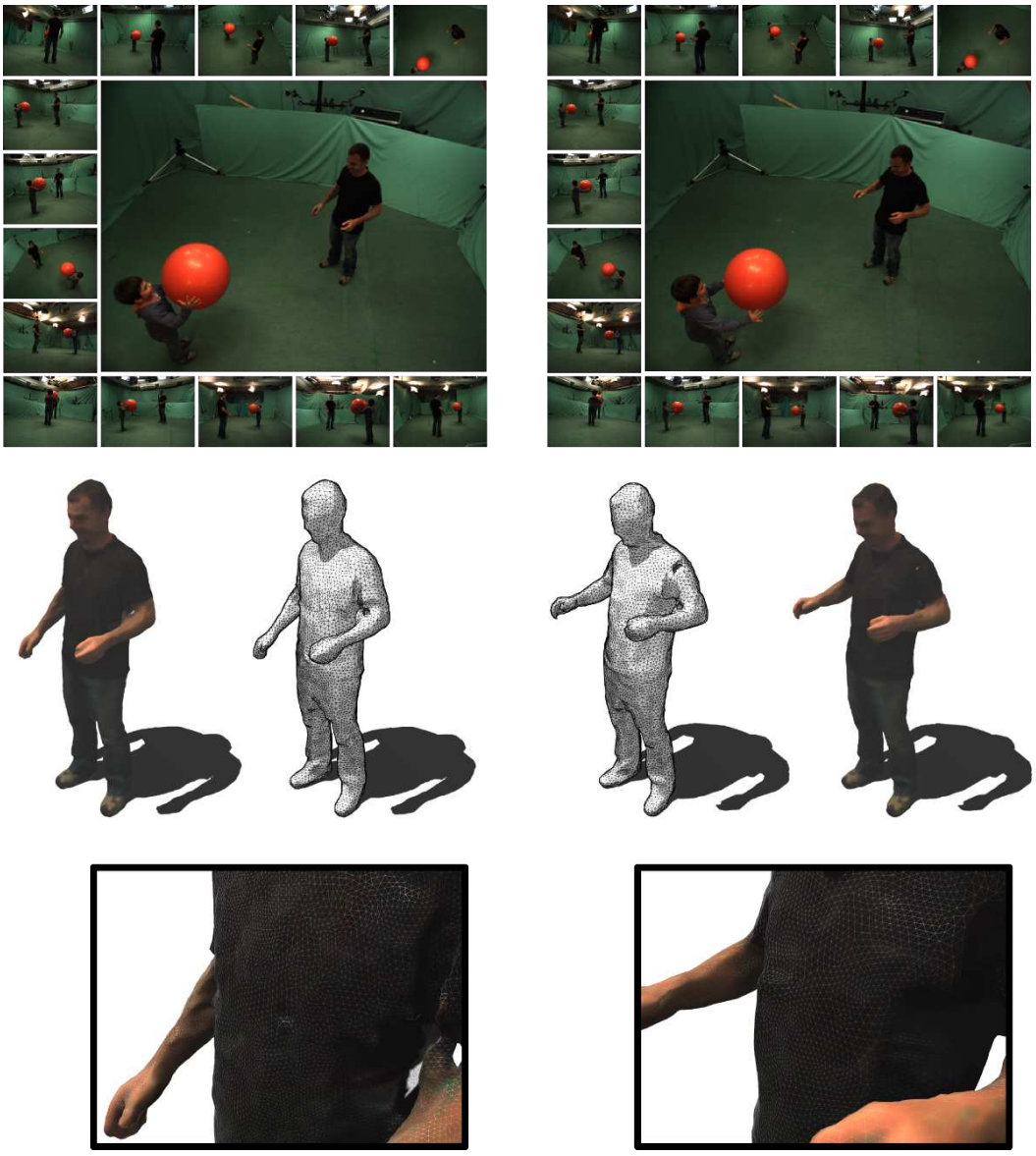


Figure 2.3: The correspondence problem in the deformable case, between two successive temporal frames. This figure contain the information that is available as input in our context: input images, indendent 3D reconstructions, and the reconstructions painted with colors sampled in the input images. The correspondences can be searched for in these images directly, or on the reconstructed surface. Note that the black t-shirt offers little distinctive photometric feature, and that there is also no geometrically characteristic feature in the middle of the chest. It is thus difficult to establish precise correspondence in this region.

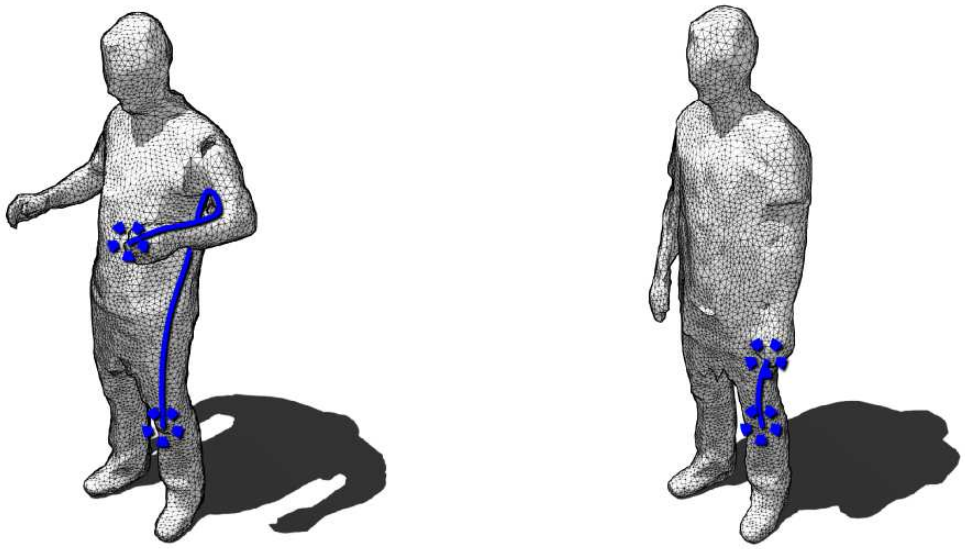
effects by taking the orientation of the surface with respect to a light source is possible for reasonnably simple environments [43, 7]. Handling the case of specularities or (self-)cast shadows for examples is much more complex. Because *photometric information* is so ambiguous, we turn in the next paragraph to methods which are mostly concerned with matching some *geometric* properties of surfaces. The other photometry-matching approaches that will be reviewed do not concern themselves with dense appearance, illumination or precise image-formation model but limit themselves to the sparse matching of areas where sharp variation of color occur.

2.2 Establishing correspondence in embedding spaces

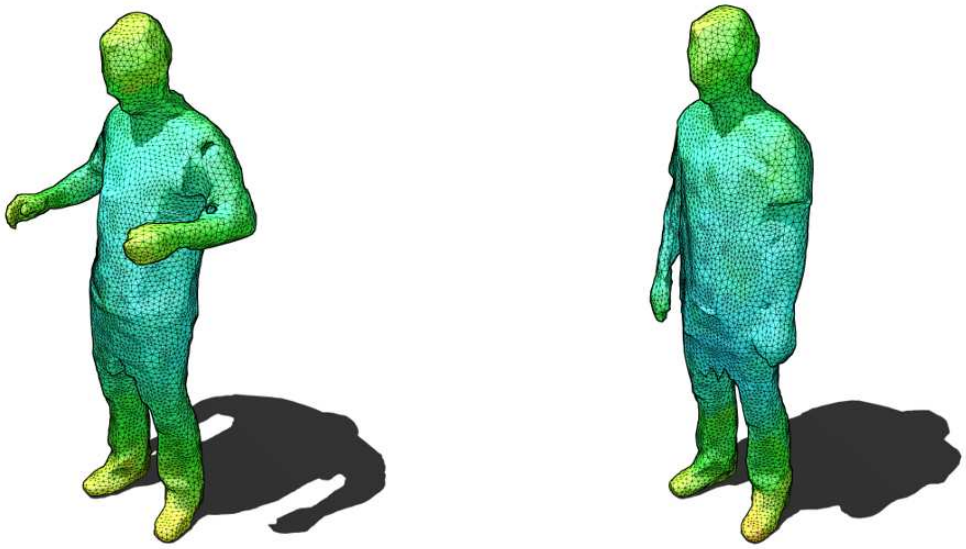
Global shape matching algorithms have mostly been developed for shape retrieval, where an input object is used to query a database for similar objects. This usually happens by mapping the shape space to a feature space where a similarity measure is defined and where the actual matching happens. Popular geometric descriptors include the spin images of Johnson and Hebert [21] or the shape context descriptor of Belongie et al. [4]. This type of descriptors allows to match shapes and have invariance properties with respect to rigid transformations (with an extension in the case of shape context descriptors). However, once two shapes have been matched, these methods do not allow to establish dense correspondence between the matched surfaces because the shapes were treated as a whole and and mapped it to a single feature descriptor. They are not structure-preserving in that there is no way to map part of this shape descriptor back to the constitutive elements of the original shape that were voxels or vertices.

In this subsection, our interest goes to structure-preserving embedding techniques. Here again, the shape is mapped to an intermediate representation that provides some invariance with respect to a class of transformation, like rigid motion in some cases or non-rigid deformations in other. Contrary to global shape descriptors, these embeddings preserve the structure of the shape, in that the vertices of the mesh and their connectivity are also mapped to the embedding space and can be matched there. The work of Starck and Hilton [35] for example establishes dense correspondence between two temporally adjacent reconstructions by mapping them to a spherical domain where the correspondence is computed, using both shape and appearance cues. One limitation of this embedding however is that the surfaces have to be of genus zero for the spherical mapping to be performed.

Other approaches try to respect the topology of the matched shapes more closely and actually use this topological information for the matching. These methods are interested in intrinsic characteristic of the shape, and notably in geodesic distances on its surface or in its volume. The geodesic distance between two points is indeed invariant to a large class of non-rigid deformations, while their distance in Euclidean space can vary greatly. Hilaga et al. [15] propose to compute for each point of the surface the geodesic integral, which is the average geodesic distance to all other points. Then they build the Reeb Graph [5] of this scalar function on the manifold described by the mesh. These graphs are much more compact representations of the topological characteristics of the shape and the matching is much more efficiently performed on them. Other works [28, 31] search for an alignment



(a) Variation of the geodesic distance between two points



(b) Variation of the geodesic integral, which is computed at each point as the average geodesic distance to all the other points. It can be seen that the left hand is a detectable extremum in the left mesh but not in the right one.

Figure 2.4: Changes of topology are common in sequences of meshes reconstructed from visual data. They result in large variations of the shape's intrinsic characteristics. This limits the reliability of these characteristic for matching. This example shows a frequent toplogical collapse that happens when the left arm occludes the side of the body.

in the spectral domain. In the work by Mateus et al. [28] for example, the topological and metric characteristics of the shape are studied through the eigenfunctions of the of the graph Laplacian of the original geometry. The shapes are aligned in the space defined by the first eigenfunctions, where the protrusions are very distinct. However, and even though some works have explored more robust characteristics such as the heat diffusion distance [6], these methods are inherently very sensitive to changes of topology. The output of 3D reconstruction algorithms exhibits such topology changes, that are mostly due to occlusions. In figure 2.4 for example, a human actor goes from having his arm up to having it along his body, and this changes intrinsic properties of the shape dramatically. More importantly, this topological change happens locally but impacts some intrinsic properties *globally*.

2.3 Sparse feature matching on the surfaces

To deal with geometry acquired from real data, and to increase the robustness of the matching to occlusion and partial reconstructions, it is reasonable to try to find matches between points of the two surfaces instead of matching the surfaces as a whole directly.

2.3.1 Finding and describing features

To match points between two surfaces, a similarity measure must be defined. This is commonly performed by first building *feature vectors* that describe geometric or photometric characteristics on a spatial neighborhood of each point. Measuring similarity between these feature vectors, or descriptors allows to evaluate the quality of point-wise matches.

Point-wise geometry descriptors typically describe the geometry of the surface around a point. The idea is well illustrated by the spin images of Johnson and Hebert [21]. These spin images are location histograms of the surface created around a point by spinning a half-plane about the surface normal and accumulating in 2D bins on that half-plane when the surface intersects it. These histograms can be matched using cross correlation or other more robust statistical measures. A similar descriptor called shape context was proposed by Belongie et al. [4]. It consists of a histogram of the distribution of the relative log-polar coordinates of the other points. This logarithmic sampling in distance decreases the influence of distortions that happen far away from the point where the descriptor is computed. In [13], an extension of the descriptor to 3D is presented, as well as harmonic shape context which keep the lowest frequency component of the harmonic representation of the shape context and grants it some rotational invariance properties. These methods work rather well to describe and match rigid shapes but when applied to non-rigid matching, the radius of the neighborhood on which the descriptors are computed offers a trade-off: if the radius is too small, the features become very local and less discriminant because of self similarities. If the radius is too big, the features become global and non-rigid deformations become a problem.

Point-wise intrinsic geometry descriptors are built from intrinsic characteristics of the shapes. For instance, the extrema of the geodesic integral, or protrusions, are distinctive features that are rare enough to be matched [42, 41]. In the case of humans typically, these would be the hand, feet and the head. The precision obtained on the localization of these extrema can however suffer from small topological or reconstruction artifacts. For example, if part of the foot disappears in one frame, the geodesic extrema might switch from the toe to the heel and cause an erroneous match between these two points. While this would have little incidence on a very coarse skeletal tracking method, it can in the case of dense surface tracking cause a local flip of the surface in the area of the leg.

For other points that are less distinctive than geodesic extrema, Starck and Hilton [36] propose to match geodesic histograms that bin for a point of interest the geodesic distance to all other points on the surface. These point-wise descriptors however suffer from the limitations that were already mentioned for their global topology matching counterparts. For example in the right mesh of figure 2.4, the extremum of the geodesic integral corresponding to the left hand does not disappear completely but its magnitude becomes too small for the protrusion to be detected. It is also easy to see that this local change of topology affects the geodesic histogram at every other point of the surface.

Point-wise photometry descriptors describe the color, or reflectance properties of the object. They are a very well studied topic in computer vision, because of the widespread need for image correspondences. The survey by Mikolajczyk and Schmid [29] offers a good overview and comparison of several region detectors and descriptors in images. In the context of 3D deformable surface tracking, the most used approaches have been the Scale Invariant Feature Transform (SIFT) by Lowe [27] and the more recent and speed-focused Speeded-Up Robust Features (SURF) by Bay et al. [3]. SIFT and SURF are quite similar and detect interesting regions in both space and scale. Because they aim at scale and rotation invariance, these approaches define a sampling grid that is aligned with the dominant gradient orientation and scaled to match the scale at which the region of interest was detected. The descriptors are then built as histograms of gradient orientation and location. In the case of SIFT, the resulting descriptor has 128 dimensions, because the sampling grid is of size 4x4 and because the orientations are split in 8 bins. The 3D locations of these features on the surfaces are simply obtained by back-projecting the 2D key-point on the reconstructed surface.

Scale invariance is a very important trait for purely image-based algorithms that need to find key-points from objects in images regardless of the distance of the object to the camera. When 3D information is available however, resorting to such algorithms amounts to ignore the fact that the 3D size of a photometric feature is known. In this case, descriptors should be build to encode how the color varies *on the surface* rather than how it varies *in an image*. This idea has been developed by recent work such as [45, 48]. Wu et al. [45] build a SIFT descriptor on oriented 3D patches, and therefore remove its dependence on the point-of-view. Zaharescu et al. [48] go further and directly consider the manifold sampled by the mesh as an image domain, and treat color intensities as scalar functions defined on this domain. Similar to the original 2D photometric detectors/descriptors, their approach

detects key-points in scale space and builds the feature from histograms of gradient. The key difference is that the whole process happens on the surface itself instead of the image domain.

2.3.2 Feature-space coherence

As it was mentioned for geometric descriptors, the precision-recall performance of point-wise features offers a trade-off linked to their locality. If the features are computed on a wide neighborhood, the matching has a bad recall performance when confronted to occlusions or partial reconstructions. The matching can also suffers from non-rigid deformations if the features do not describe intrinsic characteristics. If the neighborhood is too small, the matching looses its specificity and becomes sensitive to noise and self-similarities in the object.

Specificity is a crucial issue as we want to avoid false positives, or outliers, as much as possible. The first solution to this problem is to exclusively build descriptors in regions that are expected to be discriminant. This is what the detector part of SIFT does for example. However, this decision is often made locally and does not account for symmetries or repeated structures. Therefore, other heuristics are used to prune correspondences from as many outliers as possible. When matching a feature against a database, one can check that closest match is isolated enough in feature space, that is that there a no other very close candidates. This ensures that the correspondence is distinctive. When matching points between two surfaces, one can also check the consistency of the forward-backward correspondence and make sure two corresponding features are each other's best matches.

The problem is that even with aggressive thresholding, none of these methods can guarantee an outlier free output. Figure 2.5 shows the result of the surface-based photometric matching of [48] where there are remaining outliers in the sparse matches, even after applying the presented pruning heuristics. One of the main limitations of these methods is that the coherence in 3D space is lost when the points are mapped to the feature space: even though two neighboring points on the surface will most likely have correlated movement in 3D space, they can be mapped to remote locations of a high dimensional feature space (recall the 128 dimensions of SIFT). Moreover, distant points in 3D space like the left and right foot of figure 2.5 can get mapped to the same positions in feature space where temporal matching will be ambiguous, even though each foot hardly moved at all in 3D space.

2.3.3 3D-space coherence

Assuming that the surface points will have spatially coherent displacements can help reject outliers and disambiguate point-wise correspondences by considering the shape matching from a global perspective in 3D space. For a rigid object typically, this is done by considering that there can not be more than 6 degrees of freedom to the set of 3D point displacements. Given three or more matches from the set of candidate correspondences, it

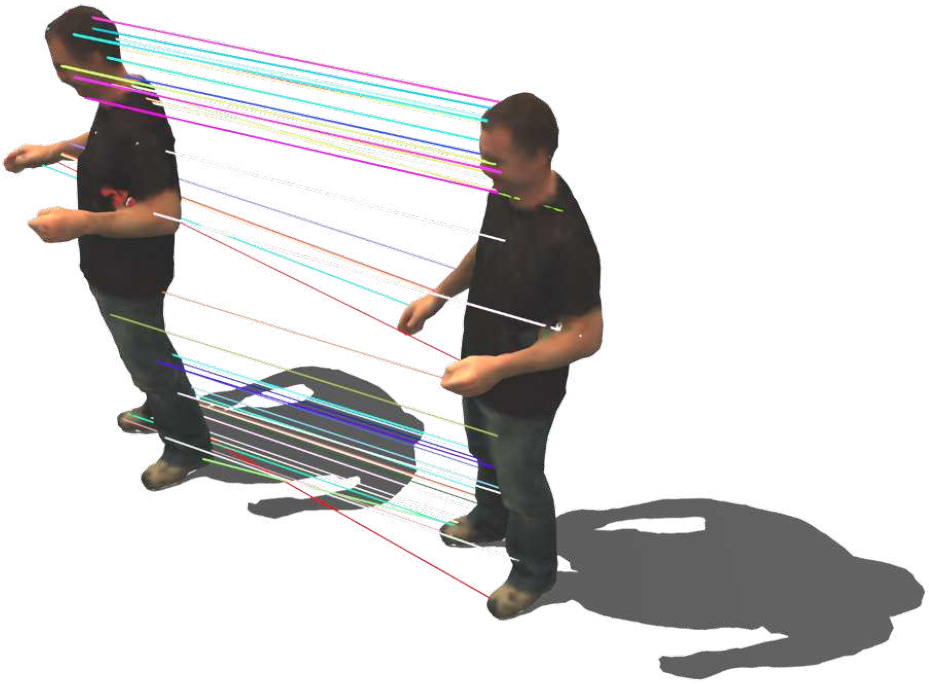


Figure 2.5: Result of the photometric feature matching of Zaharescu et al. [48]. In spite of outlier rejection, there are remaining matches between the left foot and arm and the right foot and arm.

is possible to compute a rigid motion [17] and to evaluate how well the rest of the candidate correspondences agree with it. This is the idea of the RANdom SAmple Consensus (RANSAC) algorithm by Fischler and Bolles [10]. Candidate rigid motions can be built from randomly picked subsets of matches until a significant part of the candidate matches agrees with the candidate rigid motion. RANSAC is a very effective method provided a candidate model can be built from a small subset of matches, and provided that the proportion of outliers is sufficiently low. It is extensively used for example in Structure-From-Motion problems [14] where it is combined with epipolar geometry to bring two images of a rigid scene into correspondence.

For the non-rigid deformations that we consider however, RANSAC is difficult to apply. First, non-rigid deformations have too many degrees of freedom to be simply constrained by a limited subset of sparse matches, and it would require too large a subset to build a candidate deformation to test the rest of the matches against. Furthermore chances of picking an outlier-free subset would decrease in a combinatorial way as the size of the required subset would increase.

Instead of trying to build one model that the majority of the candidate matches should agree with, other approaches consider more local pairwise spatial relationship between

correspondences. The method by Leordeanu and Hebert [25] builds an sparse symmetric affinity matrix A that has as many dimensions as there are candidate correspondences. $A[i, j]$ measures if feature number i and feature number j respect a given geometric constraint when considered together. Then the method computes the eigenvector of A corresponding to the largest eigenvalue. This vector measures the strength of the association between each correspondence and the strongest connected cluster in the graph of all correspondences. The final subset of accepted correspondences is essentially this cluster, with additional constraints preventing one-to-many or many-to-one correspondences. The key question to be answered when using this spectral method resides in the definition of the affinity measure between two correspondences, that is of the geometric constraint the pair of matches must respect. This affinity measure can be the preservation of the geodesic distance [19] when the deformation is large. If the correspondence is established between subsequent frames of a temporal sequence, the simpler variation of Euclidean distance can also be used [8].

The method by Starck and Hilton [36] casts the task of finding a coherent subset of matches as a discrete labeling problem. In their work, they build the graph formed by the points of interest on the first shape, considering that two points are neighbors in the graph if they are within a given geodesic distance of each other on the mesh. For every point of interest, the goal is then to find an optimal label in a set of labels formed by possible endpoints on the second shape. The pairwise consistency between the labeling of two neighboring interest points is defined as the preservation of the geodesic distance, and the graph is labeled by processing it as a Markov Random Field in a Loopy Belief Propagation algorithm. This method allows to put into correspondence two observations of a deformable object in very different configurations. Furthermore it is robust to some variations of topology because it only considers the geodesic distance locally. However, this method is computationally quite involved.

3 Dense surface tracking

Inferring dense correspondence between two shapes from the motion cues that were reviewed until now is a regularization problem. There are two aspects to this regularization: first, the motion needs to be interpolated to the whole surface, particularly when using sparse motion cues. Second, the regularization must prevent the over-fitting of erroneous motion cues. As such, the task resembles a smoothed interpolation problem. We have seen in the case of sparse matching that outlier rejection heuristics and algorithms enforce geometric constraints based on prior knowledge on the way points on the surface should move with respect to each other. Similarly here, the regularization should account for some prior knowledge on how the surface deforms. Furthermore, and as it was motivated the discussion of optical flow and scene flow, the regularization should take place in 3D, where

the physical processes that underlie the observations happen, rather than in image space.

Deformable model The prevalent way of describing the motion field between two shapes is to deform a mesh of one of them with the motion cues obtained from the data. This allows to have Lagrangian specification of the flow field, where the discrete structure used for computations follows the deformations of the object through time (as opposed to the Eulerian specification that describes the flow of matter through a fixed grid). More importantly, this mesh structure is useful to define the regularization of the motion field. Indeed, in absence of prior knowledge on the nature of the discretized shape, the connectivity of the mesh offers a good indication on the connectivity of the object. This was already mentioned in the outlier rejection algorithms for sparse matches: if two points are close to each other geodesically, they will most likely have coherent motion. As such, the motion field can be regularized by making sure that these local neighborhoods are preserved. Many methods exist to deform meshes while preserving the local geometry as much as possible. The first part of chapter 3 will discuss these works in details. What is of importance in this chapter is how these deformable models are of use when tracking the temporal evolution of surfaces through long sequences.

Some works do not explicitly deform the mesh but simply use it as support to compute and regularize the flow field. We already mentioned the work by Pons et al. [30] who use the Laplace-Beltrami operator defined by the mesh reconstructed at frame t to define and penalize the harmonic energy of the flow field. Similarly, Ahmed et al. [1] establish dense correspondence between two surfaces without resorting to a mesh deformation framework. Their idea is to create a local parametrization of the surface by localizing a point with his distance to the sparse features on the surface. In place of expensive to compute geodesic distances, they pre-compute on the mesh the values of one harmonic function per sparse feature. Then any given point on the first surface can be defined by the values of the harmonic functions associated to the K closest sparse feature. The corresponding point on the second surface is obtained by intersecting the isolines of the K harmonic functions of the matched sparse features. Because it only considers the closest sparse matches, this method offers some robustness to small and localized topology changes. It should also be expected that the redundancy of using K closest features instead of only three would help in the case of remaining outliers in the sparse correspondences. However, this method's performance is very dependent on homogeneously sampled sparse feature matches and the effects of outliers are not really adressed in the paper. More importantly, the presented results show the first mesh of the sequence deformed across the whole sequence. This first mesh is used as model to regularize once more the computed dense frame-to-frame correspondence during the propagation of deformation. This seems to be the real key to the robustness of the method, which would otherwise yield drifting in the trajectories if the frame-to-frame matches were simply accumulated.

Using a unique reference mesh to regularize the motion during the whole sequence is not only a convenient way to obtain a temporally consistent sampling of the shape. It also prevents the accumulation of tracking error because the geometric variations of the local neighborhoods on the surface are always compared to the same reference template. How-

ever, this robustness comes with a drawback: the topology of the mesh can not evolve. The work of Varanasi et al. [42] explores this compromise. In this work, the first mesh of the sequence is iteratively deformed through the sequence, but allowed to change topology. In practice, the mesh that was obtained as solution at time t is roughly aligned to the independently reconstructed surface from $t + 1$ using sparse photometric and geometric features and a Laplacian mesh deformation method (1.4.3). Then a *mesh morphing* step involving smoothing, local edge flips, vertex insertion and deletion as well as handling of self-intersections is run to guarantee a clean sampling of the second surface. This method provides a good estimate of the movement over short periods, and effectively allows to process sequences that involve changes of topology. However, the precision of the recovered point trajectories over more than one or two seconds is subject to drifting, which makes this approach unsuited for the purpose of building consistent 4D representations of longer sequences.

If one is willing to accept the limitation of a fixed topology, deforming a unique reference mesh for the whole sequence is a powerful way to constrain the trajectories of surface points. The works by Vlasic et al. [44] or de Aguiar et al. [8] provide good illustrations of this idea, as well as impressive results. In [44], the critical part of the registration is accomplished using an articulated tracking algorithm that aims at fitting the visual hulls with a skeleton. The dense surface correspondence is then obtained by first deforming the template mesh with this articulated pose, then deforming it to fit the silhouettes while preserving local geometric properties with respect to the reference mesh . As such, in this work, the general body motion is regularized by the skeletal model while the mesh deformation is more of an interpolation step. A parallel can be established with [8] where a coarse volumetric mesh is first deformed to obtain a rough registration of the template mesh with the observed data. Then the reference template itself is deformed with an "as-rigid-as-possible" mesh deformation method to fit finer shape cues such as stereo data. The common point between these methods is that the registration stage first happens at a coarse level where the deformation is decorrelated from the complexity of the surface geometry. This means that the deformation is spatially sampled at a coarser level of detail than the geometry, which effectively reduces the number of degrees of freedom in the optimization and improves the robustness of these methods.

Importance of silhouettes Another common point found in many of the works that tackle surface deformation in multi-camera studios is the importance of silhouettes in the inference for motion. At first sight, this can seem counter-intuitive as silhouettes are purely geometric information and do not explicitly contain any motion data, as opposed to photometric feature matches for example. However, they are very constraining in terms of where the surface can or can not be. Furthermore, they are the most reliable information available in studio environments because the lighting and background can be controlled.

For example, the work by de Aguiar et al. [9] deforms a reference template using optical flow as motion cue, but relies heavily on silhouettes to filter out erroneous optical flow vectors by checking that their endpoints lie inside the segmented silhouette of the object. In following work [8], SIFT features matches between frames are used as motion cues to

drive the deformation of the template mesh. However the process described in the paper is more complicated than simply feeding positional constraints into a mesh deformation framework. The mesh is not constrained by the endpoint positions of the feature matches directly but these constraints are instead progressively displaced towards the endpoints, making sure that the silhouette reprojection error decreases at each step. So in effect, although the SIFT matches are used to drive the deformation, it appears that the silhouette reprojection error is what is being minimized. This makes sense if we consider that there potentially still are outliers or noisy correspondence in the sparse correspondence set, even after pruning.

4 Conclusions

In the light of these previous works, it appears that sparse feature matches can be used to build hypothesis on local surface motion, but that their reliability makes them difficult to work with. In particular, they are difficult to use when building continuous energy functions that should have their minimum at the solution we are looking for. One has to account for the imprecision and the possible outliers in sparse correspondences. One of the questions addressed by this dissertation is whether sparse feature matches are absolutely required or if geometric information such as silhouettes, combined with a deformation prior, are sufficiently constraining to recover meaningful surface deformations in multi-camera systems. We therefore explore two aspects of the problem: in the next chapter we look into the deformation prior and into generic, efficient and robust numerical tools to deform meshes from visual data. In chapter 4, we propose a method to register a template mesh to a sequence of independently reconstructed meshes, and show on numerous experiments that pure geometric registration can very consistently yield very convincing results.

References

[1] Ahmed, N., C. Theobalt, C. Rössl, S. Thrun, and H.-P. Seidel
2008. Dense correspondence finding for parametrization-free animation reconstruction from video. In *IEEE CVPR*.

[2] Baker, S., D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski
2011. A database and evaluation methodology for optical flow. *IJCV*.

[3] Bay, H., A. Ess, T. Tuytelaars, and L. V. Gool
2008. Surf: Speeded up robust features. *CVIU*.

[4] Belongie, S., G. Mori, and J. Malik
2000. Matching with shape contexts. In *IEEE Workshop on Content-based access of Image and Video-Libraries*.

[5] Biasotti, S., L. De Floriani, B. Falcidieno, P. Frosini, D. Giorgi, C. Landi, L. Papaleo, and M. Spagnuolo
2008. Describing shapes by geometrical-topological properties of real functions. *ACM Comput. Surv.*

[6] Bronstein, A. M., M. M. Bronstein, R. Kimmel, M. Mahmoudi, and G. Sapiro
2010. A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. *IJCV*.

[7] Carceroni, R. L. and K. N. Kutulakos
2002. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3d motion, shape and reflectance. *IJCV*.

[8] de Aguiar, E., C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun
2008. Performance capture from sparse multi-view video. In *SIGGRAPH*.

[9] de Aguiar, E., C. Theobalt, C. Stoll, and H.-P. Seidel
2007. Marker-less deformable mesh tracking for human shape and motion capture. In *CVPR*.

[10] Fischler, M. A. and R. C. Bolles
1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*.

[11] Franco, J.-S. and E. Boyer
2005. Fusion of multi-view silhouette cues using a space occupancy grid. In *ICCV*.

[12] Franco, J.-S., C. Menier, E. Boyer, and B. Raffin
2004. A distributed approach for real time 3d modeling. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*.

[13] Frome, A., D. Huber, R. Kolluri, T. Bulow, and J. Malik
2004. Recognizing objects in range data using regional point descriptors. In *ECCV*.

[14] Hartley, R. I. and A. Zisserman
2000. *Multiple View Geometry in Computer Vision*.

[15] Hilaga, M., Y. Shinagawa, T. Kohmura, and T. L. Kunii
2001. Topology matching for fully automatic similarity estimation of 3d shapes. In *SIGGRAPH*.

[16] Hirschmuller, H. and D. Scharstein
2009. Evaluation of stereo matching costs on images with radiometric differences. *PAMI*.

[17] Horn, B. K. P.
1987. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*.

[18] Horprasert, T., D. Harwood, and L. S. Davis
1999. A statistical approach for real-time robust background subtraction and shadow detection. In *ICCV*.

[19] Huang, Q., B. Adams, M. Wicke, , and L. J. Guibas
2008. Non-rigid registration under isometric deformations. In *EUROGRAPHICS*.

[20] Jean-Yves Bouguet
. Matlab Camera Calibration Toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[21] Johnson, A. E. and M. Hebert
1999. Using spin images for efficient object recognition in cluttered 3d scenes. *PAMI*.

[22] Kolev, K., M. Klodt, T. Brox, and D. Cremers
2009. Continuous global optimization in multiview 3d reconstruction. *IJCV*.

[23] Ladikos, A., S. Benhimane, and N. Navab
2008. Efficient visual hull computation for real-time 3d reconstruction using cuda. In *Proceedings of the 2008 Conference on Computer Vision and Pattern Recognition Workshops*.

[24] Laurentini, A.
1994. The visual hull concept for silhouette-based image understanding. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*.

[25] Leordeanu, M. and M. Hebert
2005. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*.

[26] Li, M., M. Magnor, and H. peter Seidel
2003. Hardware-accelerated visual hull reconstruction and rendering. In *In Graphics Interface 2003*.

[27] Lowe, D. G.
2004. Distinctive image features from scale-invariant keypoints. *IJCV*.

[28] Mateus, D., R. P. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer
2008. Articulated shape matching using laplacian eigenfunctions and unsupervised point registration. In *CVPR*.

[29] Mikolajczyk, K. and C. Schmid
2005. A performance evaluation of local descriptors. *PAMI*.

[30] Pons, J.-P., R. Keriven, and O. Faugeras
2007. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *IJCV*.

[31] Reuter, M.
2010. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *IJCV*.

[32] Scharstein, D. and R. Szeliski
2001. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*.

[33] Seitz, S. M., B. Curless, J. Diebel, D. Scharstein, and R. Szeliski
2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*.

[34] Snow, D., P. Viola, and R. Zabih
2000. Exact voxel occupancy with graph cuts. In *CVPR*.

[35] Starck, J. and A. Hilton
2005. Spherical matching for temporal correspondence of non-rigid surfaces. *ICCV*.

[36] Starck, J. and A. Hilton
2007a. Correspondence labelling for wide-timeframe free-form surface matching. In *ICCV 2007*.

[37] Starck, J. and A. Hilton
2007b. Surface capture for performance based animation. *CGA*.

[38] Starck, J., A. Maki, S. Nobuhara, A. Hilton, and T. Matsuyama
2009. The multiple-camera 3-d production studio. *IEEE Trans. Cir. and Sys. for Video Technol.*

[39] Svoboda, T., D. Martinec, and T. Pajdla
2005. A convenient multicamera self-calibration for virtual environments. *Presence: Teleoper. Virtual Environ.*

[40] Tappen, M. F. and W. T. Freeman
2003. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *ICCV*.

[41] Tung, T. and T. Matsuyama
2010. Dynamic surface matching by geodesic mapping for 3d animation transfer. In *CVPR*.

[42] Varanasi, K., A. Zaharescu, E. Boyer, and R. P. Horaud
2008. Temporal surface tracking using mesh evolution. In *ECCV*.

[43] Vedula, S., S. Baker, R. Collins, T. Kanade, and P. Rander
1999. Three-dimensional scene flow. *Computer Vision, IEEE International Conference on*.

[44] Vlasic, D., I. Baran, W. Matusik, and J. Popović
2008. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH*.

[45] Wu, C., B. Clipp, X. Li, J.-M. Frahm, and M. Pollefeys
2008. 3d model matching with viewpoint-invariant patches (vip). In *CVPR*.

[46] Wu, X., O. Takizawa, and T. Matsuyama
2006. Parallel pipeline volume intersection for real-time 3d shape reconstruction on a pc cluster. In *Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*.

[47] Zach, C., T. Pock, and H. Bischof
2007. A globally optimal algorithm for robust TV-L1 range image integration. In *ICCV*.

[48] Zaharescu, A., E. Boyer, K. Varanasi, and R. P. Horaud
2009. Surface feature detection and description with applications to mesh matching. In *CVPR*.

CHAPTER 3

A Robust Mesh Deformation Model

In this chapter we address the problem of defining a numerically robust mesh deformation framework for the purpose of data-driven mesh deformation, which is an essential component of this dissertation. The chapter begins with a discussion of the existing representations of shape and presents a survey of the state-of-the art on mesh manipulation and deformation that spawns over the fields of computer vision and computer graphics. Although these two fields have different objectives and applications, parallels can be drawn between their respective developments and we show that recently proposed deformation models for interactive mesh editing are of interest in the context of data-driven tracking of deformable surfaces. We introduce a patch-based mesh deformation framework that builds on these recent developments, and detail its integration in contexts that require data-driven mesh deformation. We present an evaluation of its behavior in various applications such as interactive deformation, 3D tracking of a deformable cloth from monocular image data, and silhouette fitting.

1 Representing and Deforming 3D Shapes

In this section, we present an overview of deformable models in computer vision and computer graphics. There is considerable overlap between the problems addressed by these two fields and their developments on surface deformation have mirrored each other for the last 25 years. Within computer graphics, our focus goes mostly to interactive editing techniques, which have in most cases no access to the precise mechanical properties of the

objects they operate on, but still aim at emulating real-world physical behavior. The recent attention that was given to this topic and the resulting advances are of particular importance in the scope of this dissertation, as they provide crucial insight into the challenges posed by deformable surfaces, as well as theoretical developments and practical experience on computationally efficient machinery to handle them. Our review of deformable models in computer vision is conducted in parallel to emphasize the shared ideas between the two fields. While computer graphics concentrate on synthesizing plausible and aesthetically pleasing images of deformed objects, computer vision algorithms try to solve the inverse problem and recover the deformations of surfaces from visual data. Much like in interactive editing, they rarely have access to physically correct models and resort to simplified models of deformation. However, the goal in this case is to infer the most probable deformation that explains the observed data. The deformable model then not only constrains the result to a space of plausible deformation: it acts as *regularizer* and allows to solve ill-posed problems in the case of missing data or to fight the over-fitting of noisy information.

In the following pages, we first justify our choice of the 3D mesh as discrete representation of temporally evolving 3D surfaces, then discuss the available computational tools to deform and manipulate these structures. This helps put our work into perspective and motivates our choices for the deformation model presented at the end of the chapter. Our review is organized around a classification similar to that of Salzmann and Fua [54], and identifies three main paradigms for the definition of *plausible* deformation.

- The first class of approaches is referred to as *data-based*, and contains the methods that learn the space of acceptable configurations of a deformable object from a corpus of previous observations.
- The second class of approaches consists of *intrinsically regularized models*. These methods design specific parametrizations of the deformation to guarantee that the output shape will be a valid deformed instance of the object.
- The third class of approaches uses prior knowledge to favor some deformed configurations over others. This prior knowledge is usually associated with less constrained, more expressive parameter spaces. We refer to this third class as *extrinsically regularized models*.

1.1 Representing 3D shapes

The representation of static 3D surfaces is a well studied subject and the available options can roughly be categorized as *implicit representations* or *explicit representations*. We briefly recall here key concepts on these two options, and discuss how these representations have advantages and drawbacks in the context of surfaces that deform over time.

Implicit representations define the surface as the zero set of a scalar function that operates on a higher dimensional space. For example many 3D surfaces can be represented as:

$$S = \{x \in \mathbb{R}^3 | f(x) = 0\}, \text{ where } f : \mathbb{R}^3 \mapsto \mathbb{R}.$$

.

In some case cases such scalar functions, and thus the corresponding shapes, can be manipulated through a mathematical definition of f. For example the 3D sphere of radius r centered at the origin can be described as the zero set of the polynomial function $f : (x, y, z) \mapsto x^2 + y^2 + z^2 - r^2$. Varying the value of r allows to express a sub-set of the possible deformations of the sphere, namely to obtain spheres of different radii. The benefit of such models resides in the relatively small number of parameters needed to define the shape and the possible smoothness properties inherited from f. Their significant drawback however is their limited expressive range, even when more elaborate models such as superquadrics [3] are used.

An alternative and more flexible option is to directly manipulate the scalar values of f on volumetric grids. The evolution of the function values is then commonly governed by a Eulerian partial differential equation (PDE) that assumes and enforces a number of differentiability and smoothness properties on f and thus on the surface. This idea constitutes the foundation of the *level-set method*, introduced by Osher and Sethian [47]. This method allows to represent a vast variety of smooth surfaces, and has the advantage of naturally handling topology changes during the evolution of the surface. This is an essential feature if the topology of the shape is not known a priori, as it occurs in 3D reconstruction tasks for example. However, it can also be argued that in some cases where a strong prior model is available, the topology should not be free to evolve. Furthermore, if we look at surface evolution in the level-set method using terminology from fluid mechanics, it appears that this representation and the PDEs that control its evolution are closely tied to an Eulerian specification of the flow field. This means that the point of view is attached to the grid and not to the flowing matter, as it is in a Lagrangian specification. As such, it is difficult to maintain point correspondence as the shape evolves. Even though some works [50] have attempted to solve this issue, explicit methods are much better suited to describe the flow field from a Lagrangian point of view, that is to describe the trajectories of surface points, which is precisely the focus of this dissertation.

Explicit Representations directly map a set of variables to a 3D surface. We first present parametric surfaces, then motivate the use of the 3D mesh as discrete representation of temporally evolving 3D surfaces.

Parametric surfaces are explicit surface representations that map sets of $\mathbf{R}^2$ to 2-manifolds in $\mathbf{R}^3$. As such, the point correspondence is directly maintained via the parametrization.

$$S : \begin{bmatrix} u \\ v \end{bmatrix} \mapsto \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix}$$

For example, the paraboloid $S(u, v) = (u, v, au^2 + v^2)$ is a *parametric* surface whose deformation is *parametrized* by the scalar a. The parametric function can be algebraically defined in the case of quadrics. It is again also possible to use superquadrics to design more complex surfaces. A third option is to use polynomials functions with finite support. This idea is at the core of B-Spline surfaces and is more convenient to use at it resorts

to 3D control-points that provide very intuitive control of the surface's deformation. We return later to these models in our discussion of interactive modeling tools. The most interesting trait of these representations appears when the mapping has good differentiability properties. It is then possible to use the tools of differential geometry [18]. This means that concepts such as *curvature, surface normals, orientability*, and the notion of differentiability for functions defined on the surface are well defined. This allows for principled mathematical modeling when evolving the surface in variational energy-minimization frameworks. However, these models have limited expressive range and the description of complex shapes often requires a delicate smooth stitching of several elementary pieces of surface.

Polygonal meshes are arguably a more versatile and popular discrete representation of 3D surfaces. Although it is possible to use different types of polygons, we favor here *triangular meshes* that exclusively use triangles and yield a simple and uniform representation of the surface's connectivity M:

$$M = (\nu, \tau),$$

where ν is a set of vertices and τ a set of triangles. The configuration of the surface in space is then encoded by a position function $\mathbf{x} : \nu \mapsto \mathbb{R}^3$ mapping each vertex to a 3D coordinate.

$$\mathbf{x} : v \mapsto \begin{bmatrix} x(v) \\ y(v) \\ z(v) \end{bmatrix}$$

This function is commonly linearly interpolated in the triangles using barycentric coordinates. The deformation of the surface is completely controlled by a vector that contains the value of position $\mathbf{x}$ for each vertex and is therefore very intuitive to manipulate.

Meshes suffer from two major limitations: first, they are discrete structures that do not allow for a straightforward transposition of concepts of differential geometry. Computing curvature. surface normals, gradients and Laplacian of scalar functions defined on the surface requires care when points are not inside of the triangles. This is because the position function is piecewise affine and thus not differentiable on edges and vertices. The second limitations is linked to topological changes. As the vertex coordinates evolve, self-intersections can appear. Figure 2.4 displays such a situation, where two meshes of the same actor in different poses have different topologies. Evolving one of the mesh to another by simply moving vertices is thus impossible. If the algorithm that drives the mesh evolution needs to garantee that the mesh samples a closed 2-manifold, some processing would be required to maintain a well conditionned sampling of the surface. This is currently an very active field of research [78, 49, 14, 76]. However, and as motivated in chapter 2, we leave in this thesis the problem of topology changes aside. We assume that we have access to a mesh that can be used as reference for tracking all along the sequence.

In this dissertation, we use the triangular mesh as discrete representation of the deforming surfaces. Our choice is mainly motivated by our need to maintain point-correspondence when the shape deforms, as the temporal trajectory of points is what we are interested in. Furthermore, meshes handle open surfaces and boundaries naturally, which allows to describe shapes such as simple pieces of cloth. Finally, and although meshes do not provide

a continuous representation of the surface, there are methods to approximate a number of differential properties on them, provided that the sampling is reasonably uniform spatially. As we will see in our discussion of mesh deformation, this is of importance because these differential properties appear in the equations of elasticity.

1.2 Data-based deformable models

Our review of deformable models starts with data-based approaches which often are a natural choice in computer vision. Indeed, data can be easily acquired, and provided some work force to manually register and annotate it, large databases of deformed shapes can be built. Data-based approaches deal with these samples of the deformed object in different configurations (keyframes) and try to infer the structure of the space of deformed shapes from the training set.

1.2.1 Blend-shapes

Given a rest mesh and a reasonnably small database of the same mesh in deformed configurations, one of the simplest things that can be done is to synthesize new poses by linear blending of the vertices position in the keyframes. This is the idea of blend shapes [48] that directly interpolate the vertex displacements of several sample poses and allow to create new poses by manipulating the blending weights.

An interesting extension called Pose Space Deformation (PSD) can be found in the work of Lewis et al. [41] that uses blend shapes to correct the mesh obtained by skeletal-based deformation. They build a database of manual corrections to the output of the articulated mesh deformation algorithm and automatically blend these corrections during synthesis. For instance, this means that artists can manually input a keyframe shape that indicated that the biceps should bulge when the elbow is bent. Then for a given pose of the descriptive model, a distance to that sample can be computed -i.e. how much is the elbow bent- yielding a blending weight for the correction of vertices positions. This idea is extended and successfully applied to data-driven facial deformation in the work of Bickel et al. [6] where the fine scale wrinkles of the face are blended from a database, using the infered local surface streching of a coarse deformation graph as a feature vector.

The work Sumner et al. [66] called Mesh IK represents every keyframe as a high dimensional feature vector. This vector that encodes not the vertices positions, but the spatial variation of these positions on the mesh for every given keyframe. The blending operates on these features rather than the vertex coordinates directly. In their work, the blending weights are computed by measuring how well each feature vector (and thus sample shape in the database) fits a set of positional constraints. In the work by White et al. [75], MeshIK is used to recover the shape of moving pieces of cloth from a set of color markers printed on it. The benefit of the approach is in this context is that Mesh IK can convincingly interpolate occluded parts of the surface by using the database of previous observations.

1.2.2 Dimension reduction

When the database of samples is large enough, it becomes impractical to associate a weight to every single keyframe. It is however possible to try to extract a low dimensional parameter space automatically. The goal is then to feed the training data into a dimension-reduction algorithm, hoping that it will infer some meaningful structure of the deformation space. Principal Component Analysis (PCA) has been used extensively to this end but recent works [64] have tried to capture the non linearities of the deformation space by using non-linear dimension reduction methods. Dimension reduction techniques have limited interest for human guided synthesis because the inferred parameter space may not have semantical value and therefore can end up being difficult to interpret and manipulate. However, when no semantical value is sought after, dimension reduction methods are a powerful tool to constrain the deformation space and limit the complexity of the search. For example they have recently been used to learn from physically simulated data a low-dimensional model of how clothing moves with respect to body pose and dynamics for fast synthesis in interactive environments [25]. In computer vision, the constrained and low-dimensional deformation space that they yield makes them a key component of many data-driven motion/deformation recovery algorithms.

If there is little variation of the viewpoint on the deformable object, the learning of deformation models can be cast as the learning of point distribution models in 2D. A notable example of such an approach is the Active Shape Model (ASM) of Cootes et al. [23]. In ASM, a PCA is run on a database of 2D points sets that usually describe the contour of an object. Given an image where an instance of the shape is to be localized, the shape is expressed as a linear combination of the principal directions. The weights are optimized for iteratively so that the points of the current approximation will better fit the observation. This work was later extended as Active Appearance Model (AAM) [22] where the texture information is warped to a common shape and is run through PCA to find principal modes of appearance. This approach was in particular shown to be quite effective at capturing the variability of human faces.

Similar dimension-reduction techniques also have been successfully applied to full 3D representations. The work by Blanz and Vetter [8] tackles the variability of human faces in neutral expression. The database built for this purpose contains 200 complete 3D models of the head acquired by laser scanning. These models are all registered together, so that a unique mesh can be used to sample all of them, leaving only the vertices positions as variable. As with AAMs, the principal modes of shape (3D coordinates) and texture (RGB coordinates) are extracted by PCA. Given an illumination and reflectance model, the parameters of this morphable model can automatically be ajusted to fit a single image and recover the full 3D shape of the face. In following work [7] this model was extended to handle principal modes of facial expressions and allowed to reanimate single pictures after the fitting process. In other work on capturing the variability of human bodies, Anguelov et al. [1] define one transformation matrix per triangle of their template mesh and run PCA on the vectors created by concatenating the 9 parameters of every triangle of a shape.

Principal modes of deformation are not necessarily learned from a database of real data

brought into correspondence. The work by [56] sample the possible deformations on a synthesized database. It samples the space of inextensible triangular meshes by varying the parameters of angles between adjacent triangles of mesh representing a piece of cloth. PCA is then applied to this sampled data and modes are extracted. An interesting result brought to light in this work is that even though the training data contains no instances of stretched mesh, the resulting low dimensional PCA space does. This is an important reminder that in spite of the success of PCA-based methods, the space of deformed configuration of a mesh should in general be expected to be non-linear. This suggests that non-linear dimension reduction techniques should be applied to build more faithful parametrizations of the deformation space.

1.2.3 Other machine learning techniques

Learning the behavior of parts The deformation space of an object needs not be learned globally. If all parts of the object are assumed to obey similar elementary deformation rules, one can attempt to learn these rules. The work by Salzmann et al. [57] illustrates this idea, and assumes that every small surface patch on the surface obeys the same deformation model. The probability density function of shape configurations for a patch from a sheet of paper is learned using Gaussian Process Latent Variable Model. The deformation is guided by edge and texture information from a single image, and the global deformation of the whole sheet of paper is constrained by a product of experts that each assess the probability of the deformations of local patches.

Discriminative learning The approaches presented up to now can be labeled as *generative*. This means that the distribution of deformed shapes is learned from the training set, and that new samples can be resynthesised for comparison with an actual observation. Another branch of *data-based* methods does nott concern itself with modeling the distribution of deformed configurations, but instead learns to differentiate observations directly. Such approaches are called *discriminative*. Contrary to *generative* approaches, they have the significant advantage of not requiring a good initialization.

For example Huang et al. [33] use shape retrieval techniques and global shape descriptors to discriminate between different poses of the same object, rather than the original purpose that was to discriminate between multiple objects. In their work, the database of deformed instances is made of independently reconstructed meshes that contain no explicit dense correspondence information. However, given an extensive database of deformed instances of a shape put in dense correspondence, this type discriminative method could be used as a deformation model.

Another good illustration of discriminative learning for pose estimation is the recent work by Shotton et al. [60]. However, it does not deal with mesh data directly, but tackles instead human pose estimation. In this work, the range of human poses in depth images is captured with a forest of decision trees. The task is cast as pixel classification in range data, where each pixel gets associated to a class that represents a body part. The approach

represents each pixel with a feature vector encoding the relative depth of two pixels in its neighborhood. For each of the node, the dimension of the feature vector along which the split is to be made is chosen as the one that discriminates the most classes in the training data. This algorithm manages to classify most pixels correctly. The joint locations, and thus the pose of the object can be recovered in a second step by looking for centers of clusters of pixels labeled as a body part. In following work, Girshick et al. [30] replaced classification forest with regression forest. This means that for a given pixel run through a tree, we do not find a label when reaching a leaf, but instead find a set of votes for the relative 3D positions of joints.

1.2.4 Automatic skinning

For articulated structures, the dimensionality of the deformation space can be significantly reduced by automatically computing an underlying skeletal model, performing what is called mesh skinning. Baran et al. [2] align a defined articulated skeleton with a static model using only its shape in the inference. More generic approaches by James and Twigg [35], Ladislav Kavan [40] take an animated mesh as input and automatically infer a set of bones (each of which moves rigidly), as well as blending weights linking each vertex to a small subset of these bones. Not only do these methods work particularly well for articulated structure, they also display results on more intricate deformations such as these of cloth. This shows that smoothly blending several local rigid transformations with respect to a rest pose can be applied rather generally to represent deformations, provided a sufficiently dense set of bones. This idea is at the core of the patch-based deformation model that we present in this chapter.

1.2.5 Conclusions on data-based deformation models

Learning from data is a powerful way to generate a parameter space of limited dimension and to constrain the space of outputs. However, for databases involving significant deformations, simple blending or PCA do not capture the non-Euclidean structure of the space of deformed configurations. In this thesis, we wish to process deforming objects of unknown nature and need to handle large deformations. This a priori unknown nature of the objects precludes largely the use of data-based methods, except for machine-learning techniques that would capture elementary deformation rules for small localized surface elements as in [57]. We will see in our discussion of extrinsically regularized deformation models that emulating such rules for these small elements can also be done efficiently and convincingly using the equations of elasticity.

1.3 Intrinsically regularized deformable models

We discuss here methods that use carefully designed parameter space to ensure that any point of this space maps to an acceptable instance of the deformed model. We differentiate

in this class of approaches two subgroups that mostly differ on their definition of what an acceptable deformation is. On one hand, we label as *descriptive* the approaches that deal with a specific real object for which a human has designed a specific set of deformation parameters, using his perception, experience and understanding of how the object deforms. On the other hand, we consider more general approaches for which the validity of a deformation is linked to geometric constraints which are enforced through the parametrization.

1.3.1 Descriptive deformable models

In the case of *descriptive* deformable models, a human manually defines the parameter space. Because this process typically involves a semantical component, the resulting set of parameters is often more intuitive to interpret and manipulate than those produced by *data-based* approaches. As a large part of the developments on such deformation models was driven by the need to synthesise, capture and edit compelling animated performances of humans, we illustrate *descriptive* models on the problems of body and face animation.

Articulated structures are a straightforward example of simplified deformation model, routinely used to render convincing animations of the human body without resorting to a complex anatomical dynamic simulation of bones, joints and muscles. The process usually requires to perform *character skinning*, that is to attach a mesh of the human body to an underlying skeletal structure. The deformed body can then be animated by manipulating a small number of joint angles directly. A possible extension consists in defining degrees of freedom for the skeleton from which the joint angles can be computed, usually as simple linear mappings. Articulated models are extensively used in computer vision, mostly for marker-based and marker-less motion capture of human performances. This parametrization alone already greatly constrains the deformation, and allows to express reasonably realistic poses for the body, provided some additional constraints such as limited angular range of articulations or minimization of self intersections of the body. We return in chapter 5 in details on articulated 3D models.

Facial animation is another field where descriptive approaches have been successfully employed. Among the parametrization of human expressions, one of the most influential is the Facial Action Coding System of Ekman and Friesen [27] that was originally designed to allow human annotators to label sequences. It consists of a vocabulary made of action units that describe elementary movements on the face, such as a wrinkled nose for example. Although this parametrization emerged in research on psychology, the work of Rydfalk [52] linked the concept of action units with a deformable polygonal model of the human face that allowed to synthesize expressive faces from this restricted vocabulary. This polygonal model was quickly used for tracking purposes in Li et al. [42].

1.3.2 Geometric design tools

Early mathematical methods for the manipulation of deforming shapes were linked to *Computer Aided Design* (CAD) tasks. Even if some of these methods are not strictly speaking model-based deformation methods, they have been used in computer vision and their limitations will motivate the discussion of extrinsic regularizers.

Parametrized shape primitives allow to quickly generate a variety of shapes from a restricted set of parameters, and to use these shape as basis before further refinements. We cite here the family of superquadrics as an interesting example because of its wide expressive range[3], and the possibilities opened by applying further deformations to them such as tapering, bending and twisting [4]. Superquadrics were also used for automatic 3D reconstruction (e.g. [61]) but only provide rather crude approximations when the shapes are complex. It was however shown that this limitation can to some extent be alleviated by combining them with more expressive deformation models [69] that further refine the parametric shape in a post-processing step.

Smooth interpolation Another group of CAD techniques deals with interpolating smooth curves and surfaces respecting a given set of positional constraints. This problem needed to be solved for the drawing of models in the automobile, aircraft and shipbuilding industries. It is in that context that Bezier curves -and later surface patches [51]- were developed in the early 1960s, soon to be followed by their B-spline and NURBS extensions. The original *Free-Form Deformation* (FFD) approach by Sederberg and Parry [58] essentially transposes these ideas to 3D by embedding the deformed object into a parallelepipedical solid whose deformation is driven by control points placed on a lattice. This provides designers with an intuitive shape manipulation tool: the control points are moved by the user and the resulting smoothly interpolated deformation field is transfered to the embedded object.

In computer vision, B-Splines, FFD and other smooth interpolation methods have also been used extensively for motion estimation and 2D and 3D image registration (e.g. [67]). Here also, an embedding structure is overlaid on the object or the space on which the deformation field is to be computed. However an inverse problem is solved, and the positions of the control points that parametrize the motion field are automatically computed from data instead of being manually defined by a user. In effect, the motion cues in the data are integrated in the neighborhood of each control point and their contributions weighted with the smooth interpolation functions. These methods originally designed for interpolation thus act as regularizer in two ways: first, they intrinsically guarantees the smoothness of the deformation field, which is inherited from the basis spline functions it builds on. Second. for sufficiently coarse control grid, there are large averaging effects in the image domain which can prevent the over-fitting of noisy motion cues.

One significant limitation of these methods is linked to the topology of the control grid: enforcing smoothness on a parallelepipedical grid can often produce strange looking results if different parts of the object are close in the reference pose and get deformed

together when they should move independently. Coquillart [24] showed that manually defined control lattice topologies, tailored for the embedded object, could alleviate this problem and provide more natural looking deformations.

These methods were initially aimed at modeling from a basis shape rather than deforming a reference shape. As such, it was assumed that positional constraints could be defined densely by the modeler trying to input detailed deformations. The only responsibility of the interpolating algorithm is to output a smooth surface, no matter how far the control points are pulled apart. This however does not reflect at all how real-life objects deform. If we pull a point of a piece of cloth, it is expected that the rest of that piece of cloth will move with it, including other potential control points. This marks the difference between interactive modeling tools and the interactive deformation tools that we will discuss later.

Smooth interpolation methods are used for computer vision applications but only play part of the *regularizer role* that is awaited from a deformation model. For example, [67] discusses how this appears clearly when trying to compute optical flow. When there are untextured areas in the image, the overall computed deformation might not move control points that lie in these areas, while the surrounding control points move because they have motion cues in their region of influence. This can lead to unlikely deformations and therefore requires the introduction of an additional regularizer force that maintains an overall coherence between the motion of the control points.

As such, to deform a reference model in a realistic manner, we need more than simple interpolation. We need smoothing splines rather than interpolating splines. These objects introduce a regularization that balances finite external external forces and the internal strain of the deformed object. This brings to the next category of deformation models which use extrinsic regularizers.

1.4 Extrinsically regularized deformable models

Extrinsically regularized approaches constitute the third branch of our taxonomy of deformation models. Here, parameter spaces with a broader expressive range can be used to describe the object's deformations, and prior knowledge on the plausibility of each point of this space is encoded either as a probability density function or an energy function. Extrinsic regularizations are often combined with data-based or intrinsically regularized parameter spaces. The role of this additional, external regularizing term is to favor some configurations of the shape over others.

For computer vision, this translates in practice into a regularization energy term in an energy minimization problem:

$$\operatorname{argmin} E_{data}(shape) + E_{reg}(shape) \tag{3.1}$$

Such equations can often be looked at from a probabilistic point of view by considering that they represent a generative model that can draw samples of observations from the distribution characterized by the energies. E_{data} then encodes for the log-likelihood of the observed visual data conditioned on a realization of the shape's configurations, while

E_{reg} encodes for a prior on the likelihood of that configuration. This Bayesian interpretation then makes the energy minimization equivalent to finding the Maximum A Posteriori (MAP) shape deformation parameters. Therefore, extrinsic regularization can be looked at a way of accounting for prior knowledge on the distribution of the deformation of the shape of interest. While *data-based* deformable models had the option of estimating such a distribution from a corpus of previous observations, the challenge here is to encode for the deformation priors analytically. This model should of course reflect some meaningful distribution of the possible configurations, but must also be convenient to minimize in equation (3.1).

1.4.1 Physics-based Models

How do real-world objects deform ? Because of the need for plausible and compelling animations that resemble real-world physical behavior, computer graphics and computer vision scientists started looking towards physical models in the 1980s. Among the notable early works, the research of Terzopoulos and Witkin [71] introduced simplifications of the PDEs of elasticity theory and structural mechanics into the field of computer graphics. Since then, physically based deformable models have emerged as a major subject of research and drawn a lot of attention, as shown by the recent survey of Nealen et al. [46]. Physical simulation mostly deals with the simplification, discretization and solving of the PDEs of continuum mechanics. Currently, the most widespread tool for discretizing and solving these PDEs is the Finite Element Method (FEM), combined with temporal integration schemes in the case of dynamic simulations. Such simulations constitute a vast field of computer science, that extends well beyond computer graphics and deals with problems such as elasticity, plasticity, fracture or fluid dynamics.

In this thesis, we restrict our discussion to elasticity, which means that the surface is assumed to always comes back to its rest configuration once its not under load. Furthermore, we notice that even though we deal with dynamic scenes, it is difficult to link our observations to underlying physical meaning because we assume no prior on the nature or mass of the observed object, and have no information on the external and internal mechanical forces applied to it. Therefore, we can not really infer anything about quantities such as inertia or damping and full blown dynamic models are out of scope. That being said, and even if we ignore the temporal evolution, the PDEs of elasticity and their FEM discretization still constitute valuable insight on how real-world materials deform.

If we assume small displacements with respect to the rest pose, the finite element method allows to discretize the object of interest and to compute a constant *stiffness matrix* $\mathbf{K}$. Then, given a set of external forces f applied to the object, the problem becomes that of a finding a small displacement u with respect to the rest-pose that corresponds to an equilibrium, or steady-state. The discretization allows to solve this problem through a linear system:

$$\mathbf{K}u = f.$$

The potential energy stored by the object in this deformed configuration is then:

$$\frac{1}{2}u^T\mathbf{K}u$$

It should be noted that for large deformations that lie outside of the realm of infinitesimal strain theory, $\mathbf{K}$ becomes a non-linear function of the configuration.

Physics-inspired tracking When tracking elastic objects, E_{reg} is generally chosen as the potential energy associated to a configuration of the object. This strain energy can be used to prevent minimizations of the type of equation (3.1) from reaching incorrect deformation states. In essence, this amounts to favoring configurations with low potential energy, which in the case of elasticity are the configurations close to the rest state of the object, or rigidly transformed versions of it.

If a specific object is to be tracked, and if there is a priori knowledge on its properties, it is possible to precisely compute the strain energy by using a mechanical model of the object. However, such models are complex and except for some rare attempts [34], very few works go to the trouble of using a Finite Element discretization for disambiguating visual data. This can be in part explained by the computational load of the FEM once large deformations bring non-linearity into play. Another explanation is that building an elasticity prior to favor the rest state over every other -and possibly valid- configuration is from the start a somewhat flawed approximation, and that there is as such no incentive to refine it with a precise energy computation. From a probabilistic point of view, the elasticity prior can be considered a flawed approximation because there is no reason for us to expect the object to be in its rest state in the images we get. In fact we should expect it not to be in this rest configuration since the images are supposed to be interesting. Nonetheless the elasticity assumption is very effective at penalizing grossly erroneous configurations, and we will see in the next paragraphs how simpler approximations of the strain energy are actually sufficient for this purpose.

As a side note, we remark that accurate modeling of physics still is of interest for vision-based force measurement, and that this seems to have driven most of the recent developments on combining precise physical models with visual data. Vision-based force measurement is particularly sought after for its unintrusiveness, and has found applications in the analysis of micro-electromechanical systems [73, 31].

1.4.2 Smoothing splines

The work of Terzopoulos et al. [70] was seminal in the fields of computer graphics. In this work, it is argued that the mathematical properties of objects as computed by differential geometry can be used to define "a reasonable strain energy for elastic bodies". For example, the resistance of a parametrized curve $\mathbf{x}(s) : \mathbb{R} \mapsto \mathbb{R}^3$ to stretching and bending

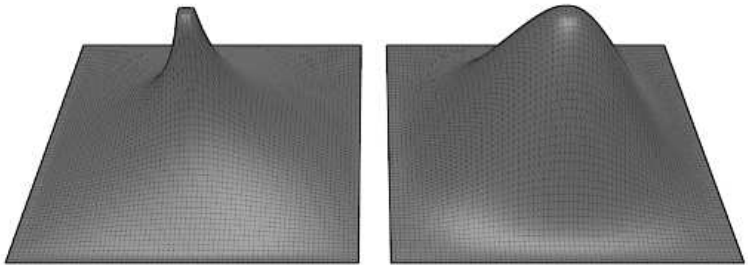


Figure 3.1: Comparison of membrane (left) and thin-plate (right) energy minimization under positional constraints (set on the borders and a 4x4 block in the middle).

(we ignore twisting here) can be emulated by minimizing the following energies:

$$E_{rod} = \int \alpha(s) \| \frac{\partial \mathbf{x}}{\partial s} \|^2 ds \tag{3.2}$$

$$E_{beam} = \int \beta(s) \| \frac{\partial^2 \mathbf{x}}{\partial s^2} \|^2 ds. \tag{3.3}$$

The first order term penalizes stretching, while the second order penalizes bending. For example setting $\beta(s)$ to zero at one point allows the snake to develop a corner.

This energy has counterparts for shells (2D) and volumes. In the case of a parametrized shell, similar simplified energies can be defined to penalize stretching and bending. In the survey of Botsch and Sorkine [13], these energies are defined as follows:

$$E_{membrane} = \iint \| \frac{\partial \mathbf{x}}{\partial u} \|^2 + \| \frac{\partial \mathbf{x}}{\partial v} \|^2 du dv \tag{3.4}$$

$$E_{plate} = \iint \| \frac{\partial^2 \mathbf{x}}{\partial u^2} \|^2 + 2 \| \frac{\partial^2 \mathbf{x}}{\partial u \partial v} \|^2 + \| \frac{\partial^2 \mathbf{x}}{\partial v^2} \|^2 du dv \tag{3.5}$$

The first order term favors membrane (rubber sheet) behavior while the second-order term makes the surface act like a thin plate (metal sheet) (see figure 3.1).

Cubic and Thin-Plate Splines are related to this physical modeling are mostly found in a whole different branch of the literature concerned with function regression and regularization. In the 1D setting typically, and given a set of samples $\{(x_i, y_i)\}$, these approaches seek a function $\hat{f}$ that minimizes the following functional:

$$E[f] = \sum (y_i - f(x_i))^2 + \lambda \int (\frac{\partial^2 f}{\partial x^2})^2 dx \tag{3.6}$$

This equation is very similar to equation (3.1) where the regularization term is the beam (or plate) energy from equation (3.3). With variational calculus, it appears that $\hat{f}$ is a linear combination of basis functions $\{f_i(x)\}$ that satisfy:

$$\frac{\partial^4 f_i}{\partial x^4}(x) = \delta(x - x_i) \tag{3.7}$$

These are Green's functions of the 4^{th} order linear differential operator $\frac{\partial^4}{\partial x^4}$. In 1D these basis functions are cubic splines $|x - x_i|^3$. In 2D they are thin-plate splines $\|\mathbf{x} - \mathbf{x_i}\|^2 \log \|\mathbf{x} - \mathbf{x_i}\|$. In 3D they are simply the absolute distance to the sample point $\|x - x_i\|$.

Bookstein [9] introduced the thin-plate-spline as a useful regularization tool for image processing and computer vision tasks in general. He insists on the physical interpretation of this model, which solves for the deflection on the z axis of a xy sheet of metal submitted to punctual external loads. The regularization term minimizes then the total bending energy of the metal sheet. He replaces the deflection function with two 1D displacement functions $\mathrm{x}(x, y)$ and $\mathrm{y}(x, y)$ that are smoothed with the framework, and combines them with a global affine function. The parameters of this deformation model are thus the weights of the thin-plate basis functions $\{\|\mathbf{x} - \mathbf{x_i}\|^2 \log \|\mathbf{x} - \mathbf{x_i}\|\}$ and of the affine warp. The whole is referred to in the literature as TPS-warp and has been used to regularize image warpings or the deformations of sheets of paper from monocular data [5] for example. The resulting deformation is intrinsically smooth because of the basis functions, and allows for a convenient expression of the extrinsic regularizer that has a closed-form solution for a given set of data constraints and a given lambda. TPS-warps as a deformation model is of interest because they have been used to parametrize and regularize the non-rigid registration of 2D and 3D point clouds [21], which will be addressed in chapter 4.

However, these methods have two main issues: first, it must be noted that in the 3D extension of TPS, the basis functions are not differentiable at the sample points. Second, these basis functions do not have compact support like the piecewise polynomial splines for example, and they actually grow as $\|\mathbf{x} - \mathbf{x_i}\|$ increases. Intuitively, this means that there is no notion of locality in the deformable model and that a data point $\mathbf{x_j}$, far from $\mathbf{x_i}$ will influence the weight of f_i in the solving process. In practice this means that the linear systems that appear in the solving process are not sparse. Moreover, the assumption of smoothness of the deformation over the whole 3D domain makes TPS-warps unsuited for our setting: a hand moving away from the thigh should not result in a movement of the thigh, even if they were close in the reference shape. This is a common problem with space deformation methods, already mentioned for FFD. However, we saw that FFD had been extended to use lattice topologies adapted to the object. Defining arbitrary domains and boundary conditions is not as easy with the TPS formalism. Finally, the non-locality of the parametrization is an even bigger issue as a movement of the hand would induce significant changes for the weights of the basis functions associated to all the sample points in space.

Discrete Splines We have seen that the TPS formulation originated from the minimization of a functional that was translated to a PDE. Then fundamental solutions for the differential operator (or Green's functions) were superposed to satisfy the whole PDE. Other

spline formulations adopt a different method to solve PDEs, that resemble more the FEM or Finite Differences.

Extrinsically regularized discrete splines were quickly identified as interesting for computer vision purposes, and used early on as shape regularizers for segmentation tasks. The seminal work of Kass et al. [37] introduced a deformation model called snakes, that uses the simplified elastic deformation model of equation (3.3) to penalize the stretching and bending of a 2D curve, and uses a finite difference approximation of the energy. The image data is then used to define a potential energy, and the combination of these two energies is minimized in a variational framework to a state that should delineate the structures of interest in the image. Also named Active Contour Model, this idea has also been extended to 3D segmentation and reconstruction tasks. Other spline parametrizations are closer to the FEM such as the already mentioned [67] that use basis functions that are compactly supported polynomials on the discretized domain

1.4.3 Differential mesh Representations

Since deformation energies are closely tied to differential properties, differential mesh manipulation methods handle differential representations of the surfaces directly. This means that the geometry is encoded with differential coordinates that allow for an easy expression of the deformation energies. The surface can be reconstructed from this representation by integration, given sufficient boundary constraints. The following paragraphs present recent developments on these topics. The energies that are presented resemble these of equation (3.5) but do not penalize absolute stretching and curvature: they penalize variations with respect to a rest pose of the surface.

Gradient-based representations of meshes such as the work of Yu et al. [77] were inspired by gradient-domain image editing. The idea is to preserve local details of the surface $\mathcal{S}$ by preserving the gradients of the coordinate function $x : \mathcal{S} \mapsto \mathbf{R}$ (resp. y, z). This is done by using as reference the scalar fields x^0, y^0, z^0. With at least a vertex position fixed as constraint, the new position can be recovered as finding the functions x (resp. y, z) that minimize the energy functional:

$$E_G(x) = \int_{\mathcal{S}} \|\nabla x - \nabla x^0\|^2 dS \tag{3.8}$$

Laplacian-based approaches such as that of Sorkine et al. [63] were developed concurrently. These methods try to preserve the mean curvature at each point of the manifold rather than the gradient. Instead of preserving this scalar field directly however, they preserve the Laplacian of the position function whose norm is precisely the mean curvature, which results in minimizing the following functional:

$$E_L(x) = \int_{\mathcal{S}} \|\triangle_{\mathcal{S}}\mathbf{x} - \triangle_{\mathcal{S}}\mathbf{x}^0\|^2 dS, \tag{3.9}$$

where $\triangle_{\mathcal{S}}$ is the Laplace-Beltrami operator.

Discretization Translating these equations from their continuous formulation to the discrete case must be handled with care. The differentiable manifold $\mathcal{S}$ is now approximated by a mesh (ν, τ), that is by a discrete graph represented by a set of vertices ν and a set of triangles τ that encode for its topology. The main problem which such a discretization is that the mesh is a piecewise linear approximation of the surface, therefore not differentiable at vertices and edges. Meyer et al. [45] and Wardetzky et al. [74] provide discussion on the choice of discretization for the continuous Laplace-Beltrami operator. Following their conclusions, the Laplacian matrix $\mathbf{L}$ on the graph of vertices is usually built using the cotangent weights. The mesh in its reference pose can be encoded by three $|\nu| \times 1$ vectors $\boldsymbol{\delta}_{\boldsymbol{x}}^{\mathbf{0}}$, $\boldsymbol{\delta}_{\boldsymbol{y}}^{\mathbf{0}}$ and $\boldsymbol{\delta}_{\boldsymbol{z}}^{\mathbf{0}}$ obtained by applying the operator to the coordinates vectors $\mathbf{x^0}$, $\mathbf{y^0}$, $\mathbf{z^0}$ of the reference mesh.

$$\boldsymbol{\delta}_{\boldsymbol{x}}^{\mathbf{0}} = \mathbf{L}\mathbf{x}^{\mathbf{0}}. \tag{3.10}$$

This leads to a discrete formulation of equation (3.9):

$$E_L(\mathbf{x}) = \|\mathbf{L}\mathbf{x} - \boldsymbol{\delta}_{\boldsymbol{x}}^{\mathbf{0}}\|^2. \tag{3.11}$$

Since the Laplacian operator is invariant to translations, $\mathbf{L}$ is rank-deficient and a positional constraint on one vertex per connected component needs to be added to the minimization so that the mesh can be rebuild from the differential coordinates.

Local rotations However, differential coordinates, whether Laplacian or gradient, are encoded in the global coordinate system and therefore not invariant to rotations. This means that if 3 vertices of a mesh are rotated around the the origin and then used as positional constraints, the reconstruction from the Laplacian coordinates brought by the minimization of the energy of equation (3.11) will be not be the whole shape rotated. This problem has been addressed by a number of works [59, 44, 39] that have proposed rotation-invariant mesh encodings. However, these encodings lose the simplicity of the linear system in equation (3.11) and have to perform large-scale non-linear optimizations to reconstruct the mesh from the rotation-invariant coordinates. To the best of our knowledge these representations have not been used in computer vision.

Differential coordinates on the other hand have been very popular, but the algorithms built around them had to be modified to account for the local rotations of the surface: in the case of gradient representations, Yu et al. [77] simply propose to have the user define local changes of frame and scale on the constrained vertices, then to explicitly propagate these frame changes to the unconstrained vertices using blending weighted by functions of the geodesic distance. In the same direction, Zayer et al. [79] replace these weights with an interpolation scheme based on harmonic fields. In the case of Laplacian representations, Sorkine et al. [63] propose to first solve equation (3.10), then to solve for a linearized version the local transformations, and finally to solve a modified version of equation (3.10) again, that accounts for the implicitly computed local frames. The linearization however restricts the method to small rotations. In a following work, Sorkine and Alexa [62] implicitly evaluate the local rotations on 1-rings around every vertex. For each each vertex v, the values $[\delta_x(v), \delta_y(v), \delta_z(v)]$ are obtained by applying this rotation to $[\delta_x^0(v), \delta_y^0(v), \delta_z^0(v)]$. The modified equation (3.11) then becomes:

$$E_L(\mathbf{x}) = \|\mathbf{L}\mathbf{x} - \boldsymbol{\delta}_{\boldsymbol{x}}\|^2. \tag{3.12}$$

This formulation allows to handle large transformations by alternatively solving for the local rotations (and thus the rotated original Laplacian vectors) and the vertices positions. This boils down to performing a non-linear optimization by iteratively minimizing equation (3.12) which is a local quadratic approximation of the true bending energy that depends non-linearly on the current vertices positions.

Usage in computer vision The Laplacian-based energy of equation (3.9) has been extensively used as regularizing term for computer vision purposes used over the last few years. This success is largely explained by the ease of implementation of [62], and the fact that this method requires only one costly factorization of $\mathbf{L}^T\mathbf{L}$. Moreover, the method behaves very well for its original purpose, that is for interactive mesh deformation.

However, the use of this energy as regularizer for computer vision is questionable. Firstly, and as noticed by others [75], Laplacian-based methods do little to penalize strain. If we consider a flat mesh, and scale it isotropically, we introduce no bending and the energy from equation (3.11) remains null. Secondly, equation (3.12) is only a quadratic approximation of the true bending energy to which we have no access. This approximation depends on the evaluation of the local rotations of the surface, which can be erroneous if the vertices positions were brought to an aberrant state. These two issues have little impact in an interactive mesh deformation setting. However, when the effects of potentially noisy data terms that need to be balanced, it is more reassuring to be able to check that we are effectively decreasing a well defined energy, and that the local quadratic approximations of it that we build are precise.

If we look at works directly related to ours, such as the methods of Vlasic et al. [72], Gall et al. [29] or de Aguiar et al. [26], we notice that they do not use the Laplacian deformation framework on the surface as their main regularization tools. For [72, 29] the strong regularizer of the template deformation is an articulated model. The bending energy of the surface is only use in a later stage to regularize adjustments destined at fitting the silhouettes. In the case of [26] the preservation of Laplacian coordinates is used on a coarse volumetric tetrahedral mesh. The surface bending energy is there again only used during refinements to match silhouettes and stereo information. Our own experience with the framework [15, 17] is that this smoothing energy effectively had to be helped by a coarser elastic model that penalized strain.

1.4.4 Embedded deformation

The methods presented until here manipulated the mesh directly by considering the original geometry as variables: vertices, faces, or local frames. In contrast, *space deformation methods* deform the ambient space in which the original geometry is embedded. This is particularly advantageous in terms of computational cost and scalability, as it allows to decouple the complexity of the deformation from that of the shape. In other words, the basis shape and its deformation field can be sampled with different spatial resolutions. A second advantage of these methods is that they can handle low-quality non-manifold meshes

and even allow deformations of point clouds, polygon soups or volumetric representations. However, and as it was already discussed for FFD and TPS warps, one of the limitations of embedded deformations methods is that the control structures must respect the topology of the deformed shape.

Cage methods embed the original geometry in a closed control mesh of much lower complexity. The vertices of the deformed mesh are interpolated from those of the deformed control structure. This allows to place constraints on the original mesh itself, but to solve for the deformation of the cage. For example Huang et al. [32] propose a formulation that allows to solve for equation (3.11) using the cage vertices as variables. They also demonstrate the potential of cage based approaches by integrating other constraints to the minimization such as skeletal constraints or the preservation of the total volume of the mesh. However, subsequent works such as [20] remark that interpolating vertices positions from the deformed control structure is a complex problem. Actually minimizing the distortion of the interpolated field can be brought back to solving PDEs inside the cage while using the control mesh to enforce boundary conditions, which is difficult for arbitrary control meshes.

Other space deformation methods use different control structures. Botsch et al. [11] embed the surface in a layer of volumetric prisms that are obtained by extruding the faces of the original mesh. Then elastic forces are emulated between adjacent prisms and rigid motions for each of the prisms are used as variables. The optimal rigid motions are obtained with a non-linear minimization performed in a Gauss-Newton framework. The process is even sped up by clustering together neighboring prisms in a coarse-to-fine hierarchy. In following work, Botsch et al. [12] embed the mesh in an array of volumetric cells and minimize a similar physics-inspired rigidity. The cubic cells are arranged in an octree that allows to refine locally the embedding structure where the deformations need to be sampled more finely. Sumner et al. [65] propose to embed the deforming geometry in a control graph consisting of points seeded in space. There, the rigidity energy is expressed between the nodes and the variables consist of one affine transformation per node. The corresponding minimization is performed similarly to [11, 12].

One of the key characteristics of the methods cited in the previous paragraph is that they optimize explicitly on one *rigid transformation* per control point, while nearly all the deformation models reviewed until now limited themselves to using the *positions* of these control points as variables. As such, these methods explicitly solve non-linear minimization problems, as opposed to the methods of 1.4.3 that implicitly evaluate the local rotations of the surface. This non-linear minimization might at first seem much more complex, for reasons as simple as the 6 parameters required per control point instead of 3. Yet, [11, 65, 12] are used as interactive deformation tools, which shows that these non-linear approaches remain computationally tractable. Two mechanisms make this possible: first of all, decoupling the control geometry from the complexity of the original geometry obviously allows to decrease the number of unknowns. Secondly, efficient numerical tools are available to solve the large sparse linear systems that appear in these computations (see the survey by [10] for a good overview).

1.5 Motivation for the presented research

In this section we saw that the deformation of possibly very dense meshes can often be described by a low dimensional set of parameters, instead of vertex positions. Articulated structures are a common example, used for both interactive animation and data-driven recovery of deformation. However, articulated structures are rare, and articulated models fail to capture fine deformations such as the cloth on a human. Recent results in computer graphics on the compression of arbitrary mesh animations [35, 25] have shown that a small set of rigid transformations and weighting functions can encode for visually complex deformations such as these of cloth. In other words, mesh skinning methods need not be restricted to articulated objects. The limited expressive range of articulated models can be overcome by extending the notions of bone and joints, and by sampling uniformly and sufficiently densely the local transformations of the surface.

This idea matches some of the embedded deformation models that were presented in the previous paragraph. More precisely, it corresponds to the embedded deformation methods that explicitly on local transformations of the surface, similarly to [11, 65, 12]. These methods were developed for interactive deformation. One of the propositions in this dissertation and in one of our papers [16], also supported by recent results on geometric registration [43, 19], is that these tools are useful for data-driven animation. We motivate our research in that direction with three main ideas:

- Embedded deformation effectively decouples the parametrization of the deformation from the complexity of the original geometry. For computer vision applications, this decouples the parametrization of the deformation from the *sampling domain* that is the mesh. The mesh can be used as sampling domain for data terms while the deformation is computed on a coarser structure. This allows to integrate noisy data terms on the area of influence of control element, and to benefit from averaging effects in the inference for the local transformations of the surface. Furthermore, choosing the scale at which these transformations are sampled allows to solve problems in a *coarse-to-fine* manner, which helps minimize low-frequency errors in the deformation more rapidly.
- Optimizing explicitly on local transformations of the surface allows to define simple and usable regularization terms. In the next section, such a regularization energy will be presented. Because this regularization energy is uniquely defined, and does not depend on current estimates of the local rotations (as in [62] for example), it is possible to check whether each step of the minimization effectively decreases it. This comes at a cost: there are more variables to be solved for, and the minimized energies are explicitly non-linear. However the continuing increase in available computational power allows to handle this additional complexity for the type of geometry that we wish to deform.
- As we have discussed, one of the key issues with embedded deformation is to respect the topology of the deformed object. We have seen how TPS deforms the whole domain, and how the control lattices and basis functions of FFD can cause artifacts. The deformation model that we present in the following sections focuses on respecting the topology of the deformed shape. This is achieved by adopting a strictly *surface-based* approach that builds a deformation graph from the connectivity of the original geometry.

In that sense our work is closer to [11] (surface-based) than it is to [12] or [65] (volumetric). However [11] was formulated in a way that did not decouple the control graph from the deformed mesh. In our formulation, the density of the deformation graph can be controlled, allowing to sample the deformations of the object with a chosen spatial precision.

In this chapter, we present an embedded deformation method based on small surface elements we refer to as *patches*. These patches resemble the *matrix palette skinning* used in real-time rendering engines in that every patch is associated with a rigid transform and that the positions of the vertices are computed by blending the transformations locally. We also present details on the robust numerical machinery that was built around this representation to optimize energy functions involving elastic regularization and other terms defined from visual data. The resulting deformation framework can be used for the automatic inference of 3D shape deformation from visual data in a large range of potential applications. To support this idea, we discuss the integration of our framework for two applications and present the corresponding results.

2 A Patch-based Approach to Data-driven Mesh Deformation

2.1 Patches

A rigid transformation with respect to the world coordinates is associated to each patch P_k. It is parametrized by the position of the patch center $\mathbf{c}_k$ and a rotation matrix $\mathbf{R}_k$ (or equivalently by a unit-length quaternion $\mathbf{q}_k$). This rigid transform yields for every vertex v of the mesh a predicted position $\mathbf{x}_k(v)$:

$$\mathbf{x}_k(v) = \mathbf{R}_k(\mathbf{x}^0(v) - \mathbf{c}_k^0) + \mathbf{c}_k, \tag{3.13}$$

where $\mathbf{c}_k^0$ is the center of P_k in the reference pose and $\mathbf{x}^0(v)$ the position of the considered vertex in the reference pose. The deformed mesh is recovered by linearly blending the predictions made by the different patches for each vertex. The weighting functions α_k are Gaussians of the Euclidean distance to the center of mass of P_k and their support is restricted to the union of P_k and its neighboring patches $\mathcal{N}_k$. These Gaussians have isotropic covariances and their standard deviation is taken as the maximum patch radius parameter of the patch seeding stage. At each vertex v, the values $\alpha_k(v)$ are normalized to add up to 1.

$$\mathbf{x}(v) = \sum_k \alpha_k(v)\mathbf{x}_k(v). \tag{3.14}$$

Figure 3.2: Greedy patching procedure evolution on the Stanford Armadillo model (170k vertices) with a maximum patch radius of 40. From left to right: patching after 1,3,4,30,143 patches have been seeded.

Thus the complete deformation of the mesh is encoded by one parameter vector $\mathbf{\Theta} = \{\mathbf{R}_k, \mathbf{c}_k\}_{k=1:N_p}$, where N_p is the number of patches seeded on the surface.

$$\mathbf{\Theta} = \{\mathbf{R}_k, \mathbf{c}_k\}_{k=1:N_p} \tag{3.15}$$

The distribution of patches on the surface should ideally follow the intrinsic nature of the shape, e.g., rigid parts. However, in the absence of prior knowledge on this structure, and to handle non-articulated objects, the patches are preferably regularly distributed over the surface, with a density that defines the spatial frequency at which we wish to sample the

surface's stretching and bending with respect to the reference pose. One existing approach to obtain a sub-sampling of the shape is the Farthest Point Sampling method [28]. This idea consists in randomly picking the first sample and iteratively placing the next sample in the middle of the least-sampled area of the domain. For our application, this translates into picking new patch centers one at a time by finding the vertex that maximizes the maximum geodesic distance to all already picked centers until the mesh is sufficiently covered. However this requires to build for each added center the complete field of geodesic distance over the mesh, which is computationally involved.

Our approach also automatically computes the patches from the embedded surface, but is much more local in nature. Our patching method considers geodesic distances, takes a maximum patch radius as parameter and seeds patches greedily. The idea is to randomly choose a vertex to be the center of the first patch and then to grow this patch until the maximum radius is reached. The subsequent patch centers are chosen among the unassigned vertices which lie on the most existing patch boundaries. The front of a new patch is propagated from the center until the maximum radius is reached or until the processed vertex is closer to the center of another patch. In all of our experiments we have assumed the sampling density of the mesh to be homogeneous and therefore approximated the geodesic distance with the number of edges of the shortest path linking two vertices. We illustrate in figure 3.2 the behavior of this greedy patch seeding algorithm.

This procedure also organizes the patches in a deformation graph that respects the topology of the embedded object. This means that the deformation graph has an edge between patches P_k and P_l if and only if there exists a vertex in P_k that is a neighbor of a vertex of P_l in the original mesh. These edges are crucial as they are used to define the rigidity of the structure which will be the key to regularizing the object's deformation.

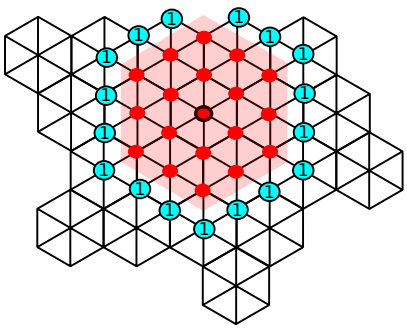
2.2 Optimization

In the previous section and our discussion of extrinsic regularizer for surface deformation, we have seem that driving the deformation of the mesh from visual data can be viewed as an optimization problem balancing data terms and extrinsic regularization terms. All these functions are assumed to be squared functions of the vertices positions, so that the problem can be formalized as an unconstrained non-linear least-squares minimization:

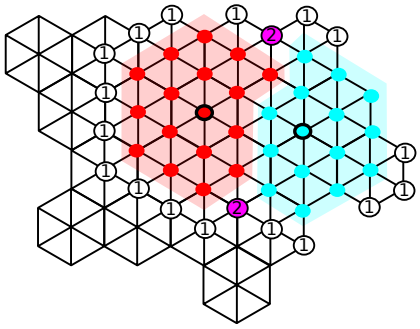
$$\begin{aligned} &\underset{\boldsymbol{\Theta}}{\operatorname{argmin}}\ E(\boldsymbol{\Theta}), \\ &\qquad \text{where } E(\boldsymbol{\Theta}) = \|\mathbf{r}(\mathbf{x}(\boldsymbol{\Theta}))\|^2. \end{aligned} \tag{3.16}$$

The residual vector $\mathbf{r}$ maps $SE(3)^{N_p} \mapsto \mathbb{R}^{N_r}$. The Gauss-Newton algorithm is commonly used to solve unconstrained least-squares minimizations. It is however not straightforward to apply here because the structure of $SE(3)^{N_p}$ is non Euclidean. We recall briefly that if it were, the Gauss-Newton method would approximate the variation of E with respect to a variation of the parameters $\boldsymbol{\Delta\Theta}$ as:

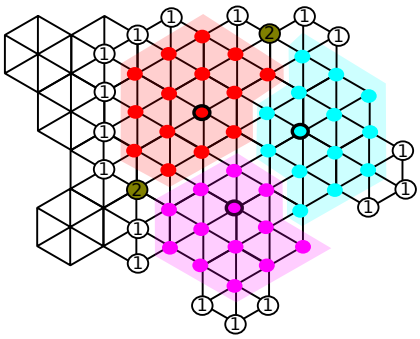
$$E(\boldsymbol{\Theta} + \boldsymbol{\Delta\Theta}) \simeq \|\mathbf{r}(\boldsymbol{\Theta}) + \mathbf{J}_\mathbf{r}\boldsymbol{\Delta\Theta}\|^2, \tag{3.17}$$



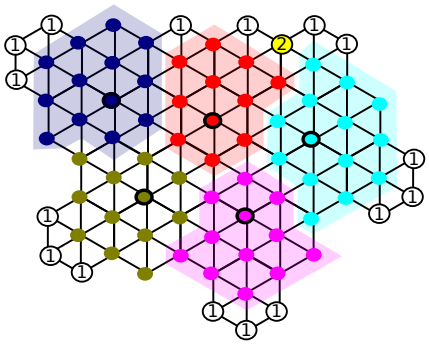
(a) Seeding of the first patch. All the points on the boundary are candidates for the second patch center.



(b) Seeding of the second patch. Only two points are candidates for the third patch center



(c) Seeding of the third patch.



(d) After five patches have been seeded

Figure 3.3: detail of the patching procedure

where $\mathbf{J_r}$ is the Jacobian of r computed at $\mathbf{\Theta}$. This is a quadratic form in $\mathbf{\Delta\Theta}$ whose minimum $\mathbf{\Delta\Theta}^*$ is given by the equation:

$$\mathbf{J_r}^T\mathbf{J_r}\mathbf{\Delta\Theta}^* = -\mathbf{J_r}^T\mathbf{r}(\mathbf{\Theta}). \tag{3.18}$$

The algorithm converges to a local minimum of E by iteratively computing these minima of local quadratic approximations and taking the corresponding steps in the parameter space.

Parameterizing the rotations However elements of $SE(3)^{N_p}$ cannot be directly used in such a framework, mainly because the rotations $\mathbf{R}_k$ can not be simply added. Thus, to use the Gauss-Newton algorithm, the neighborhood of the current approximation $\mathbf{\Theta}$ needs an adequate parametrization that will allow to compute an energy-minimizing step. We chose to use exponential maps [68] to do so. As shown in figure 3.4, this means that for each patch the update step is parametrized with a vector $\boldsymbol{\theta}_k = \begin{bmatrix}\mathbf{u}_k \\ \mathbf{v}_k\end{bmatrix} \in \mathbb{R}^6$.

$$\mathbf{R}_k \mapsto e^{[\mathbf{u}_k]_\times}\mathbf{R}_k$$
$$\mathbf{c}_k \mapsto \mathbf{c}_k + \mathbf{v}_k$$

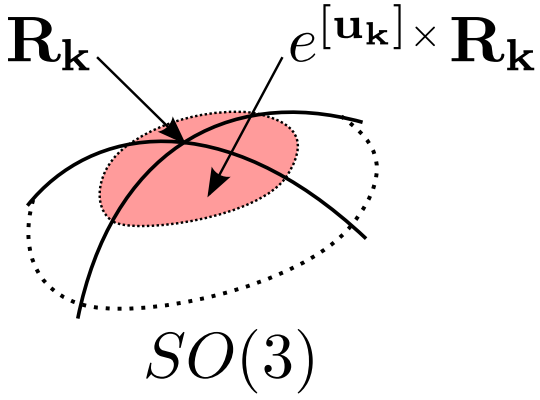


Figure 3.4: On a small neighborhood around the current approximation $\mathbf{R}_k$, the manifold of rotations $SO(3)$ is locally parametrized with an exponential mapping

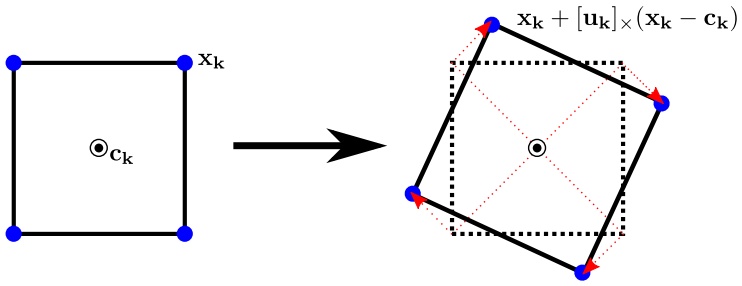


Figure 3.5: In this example, the points are in the plane $z = 0$ and $\mathbf{u}_k$ points along the z direction. The first order approximation of the exponential map adds to each vertex $\mathbf{x}_k$ the cross product of $\mathbf{u}_k$ with $\mathbf{x}_k - \mathbf{c}_k$. This approximation of the transformation is not a rigid motion and rapidly induces scaling effects when $\|\mathbf{u}_k\|$ (the angle) increases.

Using the first-order expansion of the exponential mapping of $\mathbf{u}_k$ to $SO(3)$, the update of $\mathbf{R}_k$ can be approximated as $\mathbf{R}_k \mapsto (\mathbf{I} + [\mathbf{u}_k]_\times + \ldots)\mathbf{R}_k$. Figure 3.5 and equation (3.19) show how this update of the parameters affects the vertices coordinates. The coordinate $\mathbf{x}_k$ varies linearly in $\boldsymbol{\theta}_k$, and thus $\mathbf{x}$ also varies linearly in $\{\boldsymbol{\theta}_k\}_{k=1:N_p}$.

$$\begin{aligned} \mathbf{x}_k(v) &\mapsto \mathbf{x}_k(v) + [\mathbf{u}_k]_\times(\mathbf{x}_k(v) - \mathbf{c}_k) + \mathbf{v}_k. \\ &\mapsto \mathbf{x}_k(v) + \mathbf{K}_k(v)\,\boldsymbol{\theta}_k \\ &\text{with } \mathbf{K}_k(v) = \begin{bmatrix} [\mathbf{c}_k - \mathbf{x}_k(v)]_\times & I \end{bmatrix} \end{aligned} \tag{3.19}$$

Computing the Jacobian For all scalar components r in the residual $\mathbf{r}$, the gradient with respect to the $\{\boldsymbol{\theta}_k\}_{k=1:N_p}$ can be expressed using the chain rule and the fact that $\mathbf{K}_k(v)$ is

precisely the Jacobian of $\mathbf{x}_k(v)$ with respect to $\boldsymbol{\theta}_k$.

$$\left[\frac{\partial r}{\partial \boldsymbol{\theta}_k}\right] = \underbrace{\left[\frac{\partial r}{\partial \mathbf{x}_k}\right]}_{1\times 3}\underbrace{\left[\frac{\partial \mathbf{x}_k}{\partial \boldsymbol{\theta}_k}\right]}_{3\times 6} = \left[\frac{\partial r}{\partial \mathbf{x}_k}\right]\mathbf{K}_k(v). \tag{3.20}$$

Taking an energy-minimizing step These first order approximations can be used to compute the gradient, but the actual energy must be evaluated once the rotations in $\boldsymbol{\Theta}$ have been properly updated with the exponential maps. In practice, the rotations $\mathbf{R}_k$ are encoded as quaternions, and their update is performed with a multiplication:

$$\mathbf{q}_k \mapsto [\cos\frac{\|\mathbf{u}_k\|}{2}, \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}\sin\frac{\|\mathbf{u}_k\|}{2}]\mathbf{q}_k$$

The resulting quaternion is normalized after each update to limit the accumulation of numerical error.

The corresponding step on the manifold does not however necessarily decrease the energy. This step was only computed to minimize the quadratic approximation of the function, and this approximation induces non-rigid effects on each patch, as shown by figure 3.5. Therefore, it is necessary to make sure that each step decreases the energy. This is performed by line-search along the direction set by the step, which maps to a trajectory on $SE(3)^{N_p}$. In practice, we follow [12] and simply halve the step's length by powers of two until the corresponding energy evaluated on the manifold is actually decreased with respect to the current state.

2.3 Regularization

The deformation method as it has been described up to now forces each patch to undergo rigid motion through the choice of parametrization. In that respect, it is intrinsically regularized. However, these rigid motions are not yet linked together and nothing prevents neighboring patches from collapsing or from drifting apart. Therefore, an extrinsic regularizer needs to be introduced to maintain coherence in the structure and prevent the deformation from reaching implausible states.

Definition We define a regularization energy that emulates elastic behavior with respect to a reference pose. As discussed in section 1 this is a reasonable deformation prior given a reference shape to be deformed, when no further information is available on the nature of the object. The energy that we define resembles these proposed by Botsch et al. [11, 12] and Sumner et al. [65]. These approaches embed the shape in deformation graphs similar to our patch structure and emulate elastic behavior by defining pairwise energy terms between nodes of the graph. These pairwise energies are simplified "elastic glue" [12], designed to

induce plausible behavior rather than to estimate a proper strain energy from the deformed object. Let us consider two neighboring patches P_k and P_l, and name $E_{kl}(\mathbf{\Theta})$ the rigidity energy that links them. The idea is that these two patches should agree on their predictions of a number of representative vertices $v \in V_{kl}$.

$$\begin{aligned} E_{kl}(\mathbf{\Theta}) &= \sum_{v \in V_{kl}} w_{kl}(v) \|\mathbf{x}_k(v) - \mathbf{x}_l(v)\|^2. \\ &= \sum_{v \in V_{kl}} w_{kl}(v) \|[\mathbf{R}_k(\mathbf{x}^0(v) - \mathbf{c}_k^0) + \mathbf{c}_k] - [\mathbf{R}_l(\mathbf{x}^0(v) - \mathbf{c}_l^0) + \mathbf{c}_l]\|^2. \end{aligned} \tag{3.21}$$

This type of regularization term integrates perfectly within the energy minimization framework because it consists of squared residuals, and because these residuals can be linearized using equation (3.20). This linearization of the residual yields a quadratic approximation of the energy, presented here in matrix form:

$$\begin{aligned} E_{kl}(\boldsymbol{\theta}_k, \boldsymbol{\theta}_l) &\simeq \sum_{v \in V_{kl}} w_{kl}(v) \|\mathbf{K}_k(v)\boldsymbol{\theta}_k - \mathbf{K}_l(v)\boldsymbol{\theta}_l - (\mathbf{x}_l(v) - \mathbf{x}_k(v))\|^2 \\ &\simeq \sum_{v \in V_{kl}} w_{kl}(v) \| \underbrace{\begin{bmatrix} \dots & \mathbf{K}_k(v) & \dots & -\mathbf{K}_l(v) & \dots \end{bmatrix}}_{3 \times 6N_p} \underbrace{\begin{bmatrix} \vdots \\ \boldsymbol{\theta}_k \\ \vdots \\ \boldsymbol{\theta}_l \\ \vdots \end{bmatrix}}_{6N_p \times 1} - \underbrace{(\mathbf{x}_l(v) - \mathbf{x}_k(v))}_{3 \times 1} \|^2 \end{aligned} \tag{3.22}$$

This type of elastic energy is numerically well behaved because it does not rely on the interpolated shape to make any computation. The set of representative vertices V_{kl} is always transformed by $(\mathbf{R}_k, \mathbf{c}_k)$ and $(\mathbf{R}_l, \mathbf{c}_l)$ and the energy only depends on these rigidly transformed point sets, and not on the interpolated shape. This means that even if the deformation is driven to an extreme case where interpolation artifacts occur (see [38] for a recent discussion of the limitations of linear blend skinning), the energy will stay well behaved and consistently pull back towards the rest state. This is illustrated by one experiment of Botsch et al. [12] where the deformation graph is collapsed to a single point, and where minimizing the deformation energy brings back to the rest state. Such consistence in the behavior of regularization terms is particularly important when dealing with visual data, as the computed advection terms are noisy.

Choosing a set of representative vertices There are several possible choices for the set of representative vertices. In the following, two existing works are presented and their transposition to our patch structure is discussed. Our proposed regularization energy is then introduced.

- The regularization energy proposed by Sumner et al. [65] applies each node's transformation to the centers of the neighboring nodes and penalizes quadratically the distance

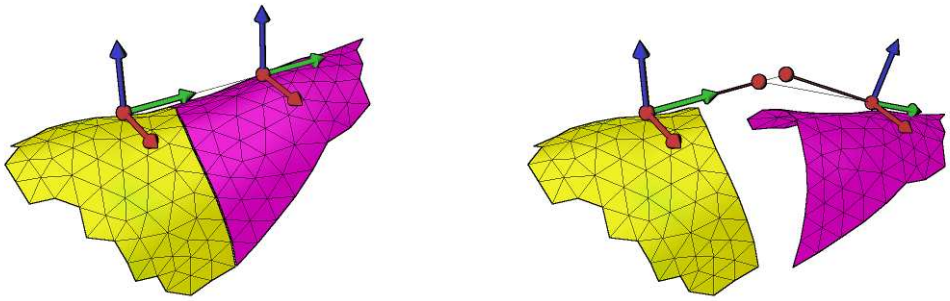


Figure 3.6: Rigidity energy as defined in the work of Sumner et al. [65]. Each patch tries to preserve the center of its neighboring patches in its own local frame. The relative rigid transformations are not well conditioned, as all rotations around the axis between the centers yield a null energy.

of these prediction to the actual positions of the neighbors's centers. In the formulation of equation (3.21), this amounts to have only the two centers $\mathbf{c}_k$ and $\mathbf{c}_l$ in V_{kl}.

$$E_{kl} = \|\mathbf{R}_k(\mathbf{c}_l^0 - \mathbf{c}_k^0) + \mathbf{c}_k - \mathbf{c}_l)\|^2 + \|\mathbf{R}_l(\mathbf{c}_k^0 - \mathbf{c}_l^0) + \mathbf{c}_l - \mathbf{c}_k)\|^2$$

An illustration of this method is displayed in figure 3.6 and helps see that this regularization is not well conditioned. Indeed, it yields 0 energy for all the relative rotations of the two patches around the axis that joins their centers. In practice, this means that this energy does not penalize twisting at all. Despite this remark, this energy has been successfully applied for interactive deformations [65] and mesh registration [43]. This can be explained by the fact that when all the nodes of the deformation graph are considered, other pairwise energy terms from other neighbors help constrain the rotations in this direction. Furthermore, it should be noted that these works do not intrinsically force the node's transformations to be rigid motions but instead let them be affine transformations that are extrinsically regularized by another energy term. In that respect, the singularity of the rigidity term could be absorbed by this additional regularization. More simply, it can be fought by penalizing changes with respect to the current estimate, that is by adding a small diagonal term on ${\mathbf{J}_\mathbf{r}}^T\mathbf{J}_\mathbf{r}$ from equation (3.18). In effect, this then adds some curvature along the direction of the energy valley, or singularity, and allows for the linear system to be solved. However, the condition number of ${\mathbf{J}_\mathbf{r}}^T\mathbf{J}_\mathbf{r}$ still can be very big in some extreme cases such as a graph made of a single 1 dimensional line of nodes.

- Choosing other representative points helps solve this singularity issue. In the works by Botsch et al. [11, 12] the meshes are embedded in control structures made of volumetric cells. In [11] these cells are prisms extruded from the faces of the surface and they are linked by an energy that forces two neighboring prisms to agree on the face that they shared in the rest pose. In [12], the cells are cubes that are linked by an energy that forces two neighboring cubes to agree on the union of their volumes. In both cases, it can be shown that these error integrals over faces -resp. volumes- can be expressed as weighted sums of the errors on the corners of the face -resp.volume-, thus fitting in the

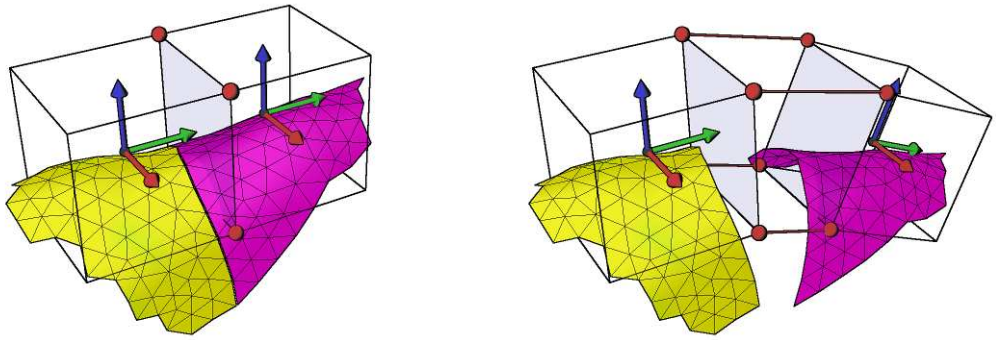


Figure 3.7: Rigidity energy inspired from the work of Botsch et al. [11]. Each node needs to agree with its neighbor on 4 vertices that lie on the face shared by polyhedra build around them.

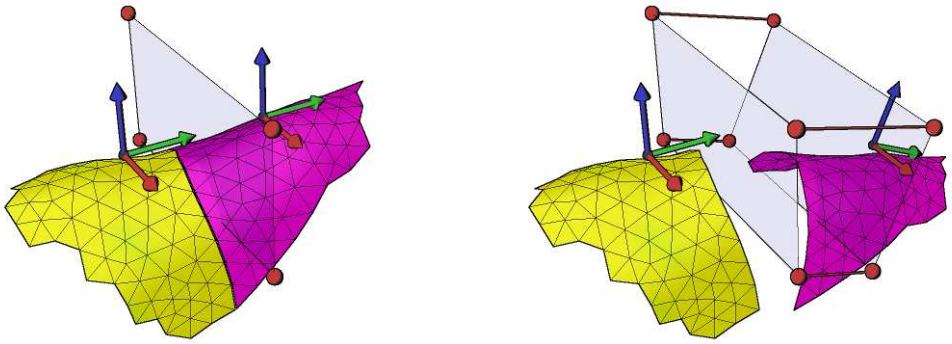


Figure 3.8: Rigidity energy inspired from the work of Botsch et al. [11]. Varying the ratio of dimensions of the common face allows to control the relative penalizations of bending, shearing and twisting.

formulation from equation (3.21). Figure 3.7 illustrates how these ideas transpose to our patch-based representation. In figure 3.8, we illustrate how varying the shape of the representative face -resp. volume- allows to control the relative penalizations of bending, shearing and twisting.

- Our rigidity energy, as defined in Eq. (3.23) and shown in figure 3.9, simply tries to enforce the predicted positions $\mathbf{x}_k(v)$ and $\mathbf{x}_l(v)$ of a vertex v by two neighboring patches P_k and $P_l \in N_k$ to be consistent.

$$E_r(\mathbf{\Theta}) = \sum_{l=1:N_p} \sum_{P_k \in N_l} \sum_{v \in P_k \cup P_l} w_{kl}(v) \|\mathbf{x}_k(v) - \mathbf{x}_l(v)\|^2. \tag{3.23}$$

The choice of the weights $w_{kl}(v)$ is of importance as allows to encode for material properties. We chose to use the product of the blending basis functions. This choice is motivated by the following equations. If we leave the discrete settings and the sum over

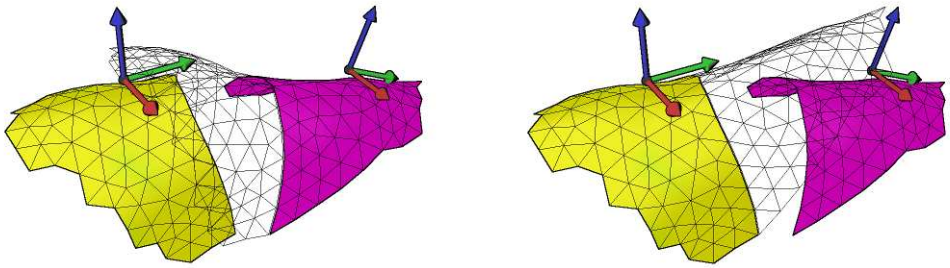


Figure 3.9: Rigidity energy as defined in the work of Cagniart et al. [16]. Both patches predict positions for the mesh points associated to themselves and their neighboring patches. The energy penalizes discrepancies in these predictions.

the vertices v, we can rewrite the previous energy as:

$$\begin{aligned}
E_r(\mathbf{\Theta}) &= \int_S \sum_k \sum_l \alpha_k \alpha_l \|\mathbf{x}_k - \mathbf{x}_l\|^2 dS \\
&= \int_S \sum_k \alpha_k \sum_l \alpha_l (\mathbf{x}_k^T \mathbf{x}_k^T - 2\mathbf{x}_k^T \mathbf{x}_l + \mathbf{x}_l^T \mathbf{x}_l) dS \\
&= \int_S [\sum_k \alpha_k \mathbf{x}_k^T \mathbf{x}_k - 2 \sum_k \alpha_k \mathbf{x}_k^T \sum_l \alpha_l \mathbf{x}_l + \sum_l \alpha_l \mathbf{x}_l^T \mathbf{x}_l] dS \\
&= 2 \int_S [\sum_k \alpha_k \mathbf{x}_k^T \mathbf{x}_k - \mathbf{x}^T \mathbf{x}] dS \\
&= 2 \int_S \sum_k \alpha_k \|\mathbf{x}_k - \mathbf{x}\|^2 dS
\end{aligned} \tag{3.24}$$

The transition between the two last lines was simply a conversion from $E[X^T X] - E[X]^2$ to $E[(X - E[X])^T (X - E[X])]$. It appears with this formulation that E_r is measuring the variance of the different rigid transformations $\mathbf{x}_k$ of the original surface, weighting this variance locally with the compactly supported blending functions α_k.

2.4 Numerical considerations

Structure of the linear system With the regularization energy defined, and given external data terms, the Jacobians for the data and rigidity residuals are computed using equation (3.20). These residuals and Jacobians could be fed to a minimization software which would compute ${\mathbf{J_r}}^T \mathbf{J_r}$ and solve the linear system of equation (3.18). However, there are 6 variables per patch and this system becomes rapidly large. Moreover, the sparse nature of ${\mathbf{J_r}}^T \mathbf{J_r}$ problem must be taken into account. To illustrate this sparseness, let us consider the Jacobian of a pairwise energy term E_{kl} linking patches P_k and P_l. If we index the vertices

v_i in V_{kl}, the approximation of equation (3.22) can be rewritten as:

$$E_{kl} \simeq \| W \underbrace{\begin{bmatrix} & \vdots & & \vdots & \\ \dots & \mathbf{K}_k(v_i) & \dots & -\mathbf{K}_l(v_i) & \dots \\ & \vdots & & \vdots & \end{bmatrix}}_{\mathbf{J}_{kl}} \begin{bmatrix} \vdots \\ \boldsymbol{\theta}_k \\ \vdots \\ \boldsymbol{\theta}_l \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ \mathbf{x}_l(v_i) - \mathbf{x}_k(v_i) \\ \vdots \end{bmatrix} \|^2 \quad (3.25)$$

This energy term contributes a symmetric matrix $\mathbf{J_{kl}}^T\mathbf{J_{kl}}$ to the general $\mathbf{J_r}^T\mathbf{J_r}$.

$$\mathbf{J}_{kl}^T\mathbf{J}_{kl} = \begin{bmatrix} \ddots & & & \\ & \sum_i w_{kl}(v_i)\mathbf{K}_k^T(v_i)\mathbf{K}_k(v_i) & \dots & -\sum_i w_{kl}(v_i)\mathbf{K}_k^T(v_i)\mathbf{K}_l(v_i) \\ & & \ddots & \vdots \\ & & & \sum_i w_{kl}(v_i)\mathbf{K}_l^T(v_i)\mathbf{K}_l(v_i) \\ & & & & \ddots \end{bmatrix} \quad (3.26)$$

These $6N_p \times 6N_p$ matrices are composed of 6×6 blocks, and their sparse structure mirrors the connectivity in the graph of patches. Other data terms depending on blended vertice positions yield similar matrices, but have a positive sign on the off-diagonal terms.

In our implementation, we never store the Jacobians of the energy terms. Instead, we directly accumulate on the $\mathbf{J_r}^T\mathbf{J_r}$ matrix and the $\mathbf{J_r}^T\mathbf{r}$ vector that constitutes the right hand side of equation (3.18). This has the advantage of keeping the memory footprint low and independent of the number of energy terms in the minimization.

Sparse Cholesky factorization Finding a minimizer of the quadratic approximation at each step of the Gauss-Newton algorithm can then be tackled by any available sparse solver, either direct or iterative. We present in this paragraph details on the direct solver. We have however confirmed that a simple Conjugate Gradient algorithm with a diagonal preconditioner was a functional alternative that requires a much less involved implementation effort. For more details, the interested reader can refer to the survey by Botsch et al. [10] on the solving of large linear systems for mesh processing.

We use a sparse Cholesky factorization package [53] to solve equation (3.18). When the deformation graph has several connected components, each one is processed independently and has its own $\mathbf{J_r}^T\mathbf{J_r}$ and $\mathbf{J_r}^T\mathbf{r}$ accumulators. Furthermore, the nodes are re-indexed in each connected component to reduce fill-in of the sparse structure during the solving stage [36]. In the same spirit, the sparse Cholesky factorization package [53] that is used for solving precomputes the symbolic part of the factorization. This symbolic decomposition is reused at each Gauss-Newton step and allows for a consequent speedup of the overall procedure.

Dimensional inhomogeneity and importance of scale As concluding remark on the numerical behavior of the presented deformation framework, we mention the conditioning of

the matrix ${\mathbf{J_r}}^T\mathbf{J_r}$, as well as the impact of scale on the numerical behavior of this framework. Both the regularization energy and the data energies accumulate 6×6 blocks on the matrix that are multiples of $\mathbf{K}_k^T(v)\mathbf{K}_k(v)$. If we name $\mathbf{dx}_k(v) = \mathbf{x}_k(v) - \mathbf{c}_k$ and omit v for the sake of clarity, we see that these terms have the form:

$$\mathbf{K}_k^T\mathbf{K}_k = \left[\begin{array}{c|c} -[\mathbf{dx}_k]_\times[\mathbf{dx}_k]_\times & [\mathbf{dx}_k]_\times \\ \hline -[\mathbf{dx}_k]_\times & \mathbf{I} \end{array}\right] \tag{3.27}$$

It can be seen that the top left 3×3 sub-matrix, which corresponds to the rotation variable $\mathbf{u}_k$, involves elements with magnitudes that are quadratic in the length of $\mathbf{dx}_k$, which varies with the scale of the mesh. In comparison, the bottom right part that corresponds to the translation variable $\mathbf{v}_k$ is independent on the scale of the mesh and is always $\mathbf{I}$. This means that a same mesh, scaled a thousand times could yield a ${\mathbf{J_r}}^T\mathbf{J_r}$ matrix with a condition number of roughly a million times the condition number that was given by the original mesh. This dependence on scale is due to the dimensional inhomogeneity of the parameters used for optimization, that is of rotations and translations.

Provided a limited numerical error in the accumulation of terms on ${\mathbf{J_r}}^T\mathbf{J_r}$ and ${\mathbf{J_r}}^T\mathbf{b_r}$, this conditioning issue has very limited impact on the behavior of the framework. A more important issue appears when regularizing Gauss-Newton type optimizers by adding a diagonal term λI to the $\mathbf{J}_r^T\mathbf{J}_r$ matrix, where λ is a small scalar. This adds some curvature to the local quadratic approximation of the energy function, which can effectively remove singularities and damp the steps taken by the optimizer. Using λI amounts to performing this regularization isotropically. Therefore we must be conscious of what isotropy means, that is of the correspondence established between lengths units and radians. The damping can be adapted to account for this correspondence by replacing the identity matrix with a diagonal matrix whose values reflect the proper ratio. This makes the framework behave similarly in all scales.

3 Applications

The presented mesh deformation framework is a generic tool for two reasons: first, it can be used to parametrize and regularize the deformation of just about any mesh, regardless of the object it actually represents. Second, we have presented how any data term that depends on vertex positions can be integrated in the energy minimization. We show in this section how different problems can be solved by simply integrating data terms in our patch-based framework, and how the same framework and code can be reused across this variety of problems.

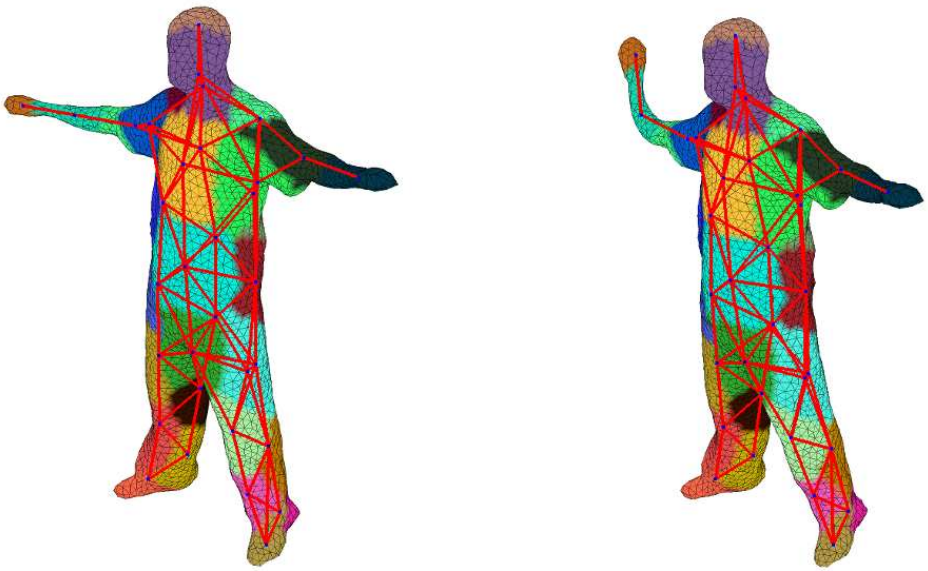


Figure 3.10: Illustration of the deformation graph inferred from the patch structure.

3.1 Interactive deformation

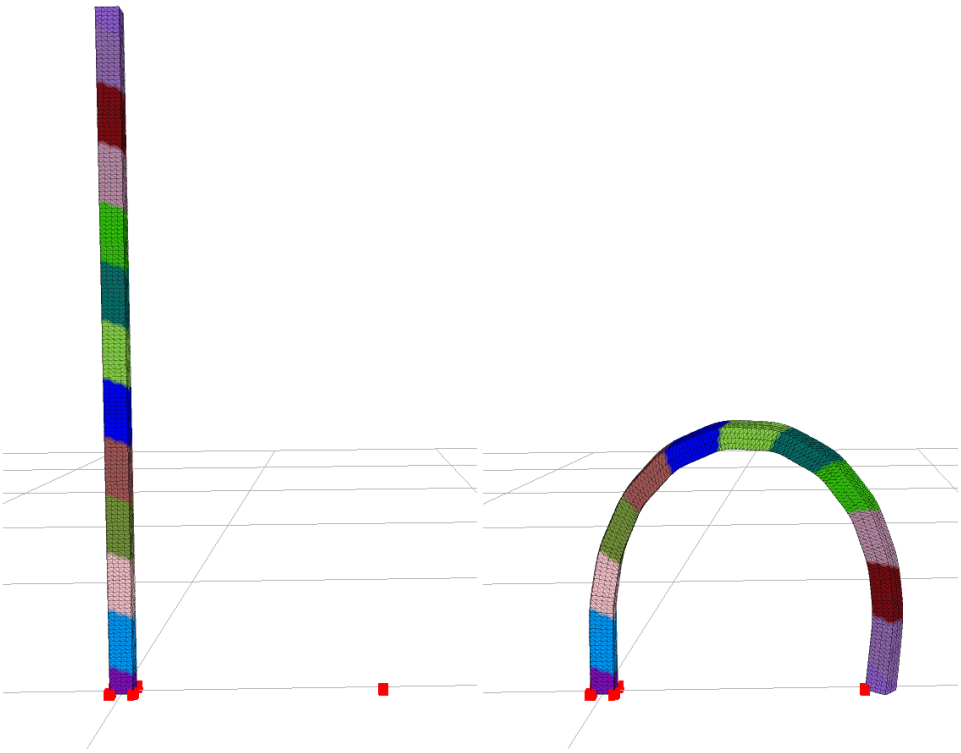
To evaluate the behavior of our patch-based mesh deformation framework and quickly test against regressions in the code base of the solver, we implemented a simple interactive application where 3D constraints on the vertices positions could manually be set by the user. For example if a vertex v on the mesh is constrained to go to position $\mathbf{y}$ in space, the energy term $\|\mathbf{r_{3D}}(v)\|^2$ is added to the energy to be minimized.

$$\mathbf{r_{3D}}(v) = \beta(\mathbf{x}(v) - \mathbf{y}) \tag{3.28}$$

Integrating these residuals in the framework only requires to compute their Jacobians, whose components are simply:

$$\underbrace{\left[\frac{\partial \mathbf{r_{3D}}(v)}{\partial \boldsymbol{\theta}_k}\right]}_{3\times 6} = \beta\alpha_k(v)\mathbf{K}_k(v) \tag{3.29}$$

The results displayed in figure 3.11 illustrate two important facts on the method. Firstly, even though the number of patches is low, the resulting interpolated deformation of the mesh is reasonably smooth. Secondly, the constraints are set on vertices of the mesh and not on patch centers. This shows that even though the deformation is computed on the control graph, the data terms can still be sampled on the original geometry.



(a) Object in its rest pose (b) Deformed object

Figure 3.11: Behavior of the patch deformation framework in an interactive deformation application. The target positions for the constrained vertices of the original mesh are indicated by red boxes.

3.2 Recovering cloth deformation

Recovering the evolutions of 3D surfaces from 2D information is a highly underconstrained problem. Prior knowledge is therefore required to ensure consistent deformations. We ran a simple experiment to show that our framework, equipped with the simple surface rigidity priors of equation (3.21), performs well in this situation. We use the data made available by the EPFL computer vision group for that purpose [55]. It consists of a reference 3D mesh model of a piece of cloth and of a list of correspondences for each video frame. Each of these correspondences maps a 3D position on the reference mesh, expressed in barycentric coordinates, to a 2D position in the image. We use these correspondences in our framework by simply defining a residual function $\mathbf{r_{2D}}(v)$ that measure the reprojection

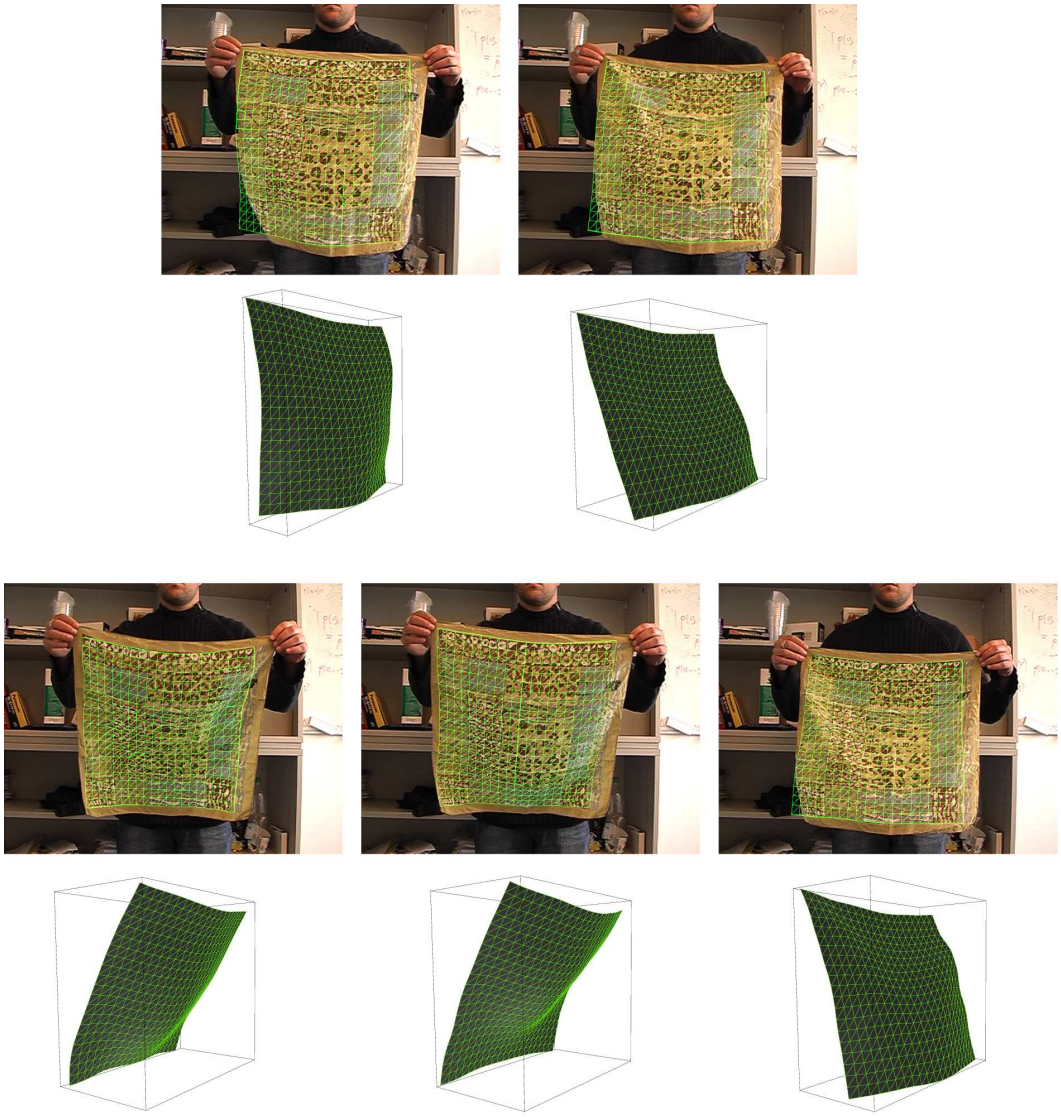


Figure 3.12: Results on the cloth dataset [55]. On the top row, the mesh overlay on the original data demonstrates a low reprojection error, while the bottom row shows that the recovered 3D deformations are physically plausible. Note on the two first images that in absence of matches, the rigidity constraints make the mesh return locally to its rest pose (flat cloth).

error of a 3D vertex on the image.

$$\mathbf{r_{2D}}(v) = \beta \left[\frac{\begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \end{bmatrix} \begin{bmatrix} \mathbf{x}(v) \\ 1 \end{bmatrix}}{\begin{bmatrix} p_{20} & p_{21} & p_{22} & p_{23} \end{bmatrix} \begin{bmatrix} \mathbf{x}(v) \\ 1 \end{bmatrix}} - \begin{bmatrix} t_u \\ t_v \end{bmatrix} \right], \tag{3.30}$$

where $\mathbf{P} = \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \end{bmatrix}$ is a 3×4 matrix that contains the parameters of the camera. The Jacobian of these residuals is more complex than in the interactive deformation example but remains straightforward to compute:

$$\begin{aligned} \underbrace{\left[\frac{\partial \mathbf{r_{2D}}(v)}{\partial \boldsymbol{\theta}_k} \right]}_{2\times 6} &= \left[\frac{\partial \mathbf{r_{2D}}(v)}{\mathbf{x}(v)} \right] \alpha_k(v) \mathbf{K}_k(v) \\ &= \frac{\beta}{z_c(v)^2} \begin{bmatrix} z_c(v) \begin{bmatrix} p_{00} & p_{01} & p_{02} \end{bmatrix} - x_c(v) \begin{bmatrix} p_{20} & p_{21} & p_{22} \end{bmatrix} \\ z_c(v) \begin{bmatrix} p_{10} & p_{11} & p_{12} \end{bmatrix} - y_c(v) \begin{bmatrix} p_{20} & p_{21} & p_{22} \end{bmatrix} \end{bmatrix} \alpha_k(v) \mathbf{K}_k(v), \end{aligned} \tag{3.31}$$

where

$$\begin{aligned} x_c(v) &= \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \end{bmatrix} \begin{bmatrix} \mathbf{x}(v) \\ 1 \end{bmatrix} \\ y_c(v) &= \begin{bmatrix} p_{10} & p_{11} & p_{12} & p_{13} \end{bmatrix} \begin{bmatrix} \mathbf{x}(v) \\ 1 \end{bmatrix} \\ z_c(v) &= \begin{bmatrix} p_{20} & p_{21} & p_{22} & p_{23} \end{bmatrix} \begin{bmatrix} \mathbf{x}(v) \\ 1 \end{bmatrix} \end{aligned}$$

Figure 3.12 presents overlays of the our result over the input images. These overlays show that there is an overall low residual reprojection error. The figure also shows side views of the corresponding 3D shape illustrating that the recovered 3D deformations are actually plausible states for a piece of cloth. This simple experiment confirms that our deformation model, although originating mostly from the field of computer graphics, can be used as deformation prior for the data-driven recovery of shape.

3.3 Silhouette fitting

The last application that we present also uses 2D data as input. We show that our deformation model can be used to optimize silhouette reprojection error when deforming shapes observed in a multi-view studio.

For many approaches to markerless performance capture, this is an important step because silhouettes are usually the most reliable information available in a multi-view studio. For example, Vlasic et al. [72] first deform the reference mesh using a skeletal pose they

optimized for, then fit the silhouettes by refining the deformation. Their method samples the silhouette contours in each image by shooting rays through the image plane every 10 pixels on the contour. For each ray, they find the vertex on the mesh that is closest to the ray (and consider some additional heuristics to take surface normals into account). They then compute for each of these vertices the closest point on the ray and use it as a 3D positional constraint in a Laplacian-preserving mesh deformation framework. This framework basically minimizes the sum of the regularization energy of equation (3.12) and of the weighted residual 3D distances of the constrained points to their targets. Gall et al. [29] also use silhouette fitting to refine the surface deformation after optimizing a skeletal pose. They regularize this refinement with a Laplacian-preserving energy also, but differ from [72] in that they constrain the projections of the points to match a given 2D position in the image, instead of setting 3D positional constraints.

As we have already remarked in our conclusion on deformation models (1.5), the Laplacian-preserving regularization energy is difficult to work with in because it depends on the current approximation of the local rotations of the surface. As such, there is no objective function whose decrease can be checked during the optimization and the convergence does not appear as provable, especially if for some reason local surface flips occur and perturb the regularization energy.

Silhouette energy We propose to explictely minimize the weighted sum of the shape rigidity energy $E_r(\mathbf{\Theta})$ defined in equation (3.21) and of the silhouette reprojection error. This silhouette reprojection error is simply taken for each camera as the XOR of the observed silhouette and the projected silhouette from the deformed mesh. This effectively defines a unique energy for which we can check that every step of the optimization indeed decreases the residual error. This does not guarantee that the algorithm will converge to the global minimum. However, it allows to stop the algorithm when the energy decrease rate becomes too small and thus guarantees that the algorithm will converge.

$$E_{SIL}(\mathbf{\Theta}) = \sum_{ci \in cameras} \|SIL_{ci} - SIL_{ci}(\mathbf{\Theta})\|, \tag{3.32}$$

where SIL_{ci} is the observed silhouette for camera i, and $SIL_{ci}(\mathbf{\Theta})$ is the silhouette generated by the mesh that was deformed with parameters $\mathbf{\Theta}$.

Energy gradient The challenge in this formulation is to compute the gradient of the data term analytically. Our approach is to compute an approximation of the gradient, and to rely on the line search in the optimization to reduce the size of the step until that approximation is valid enough to take an energy decreasing step. We first find *contour generators*, that is the set of vertices that are on the border of $SIL_{ci}(\mathbf{\Theta})$. Then, for each of these contour generating vertices v_g, we consider the line in the image plane defined by its projection and the projection of its normal. For each v_g, we scan along this line in SIL_{ci} for a corresponding point $\mathbf{t}$ that has a compatible gradient. Finally, we use the gradient of the 2D residual function $\mathbf{r_{2D}}(v_g)$ defined in the previous section to approximate the variation

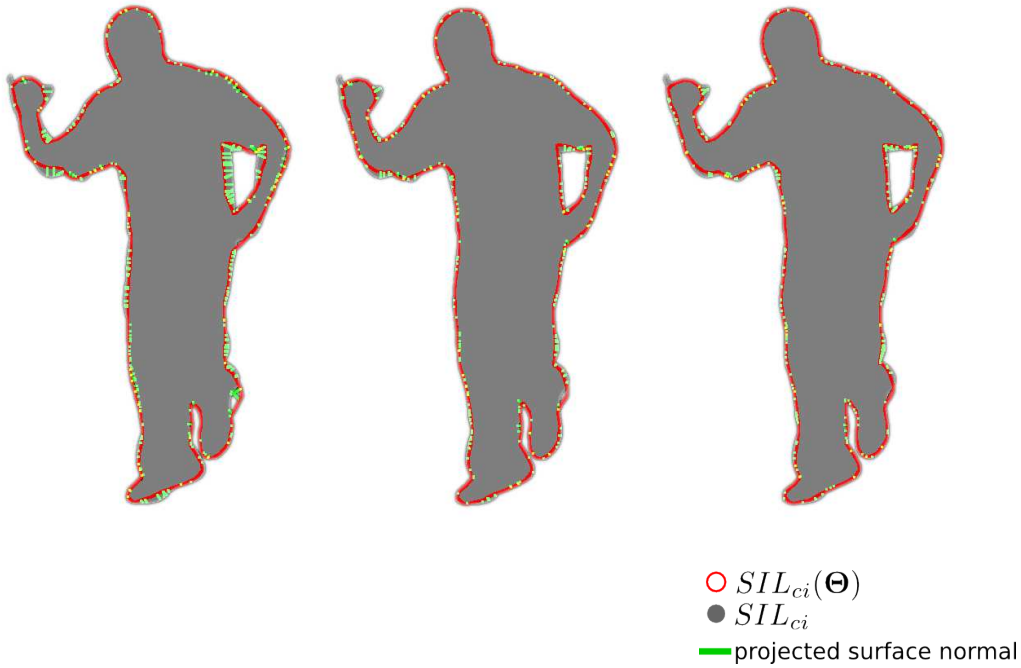


Figure 3.13: The observed silhouette SIL_{ci} is displayed in gray. The silhouette generated by the deformed mesh $SIL_{ci}(\mathbf{\Theta})$ is displayed in red. The small green lines indicate the found correspondences between silhouette generators in the red contour and point of the observed silhouette.

of the silhouette error with respect to small variations of $\mathbf{\Theta}$

$$E_{SIL}(\mathbf{\Theta}) \simeq const + \sum_{v_g} \|\mathbf{r_{2D}}(v_g)\|^2 \tag{3.33}$$

A simple interpretation for this approximation is that moving the contour generator one pixel towards the real silhouette will reduce the silhouette overlap error by one pixel. A more principled approximation would need to account for the neighboring contour generators along the silhouette boundary. We however keep the approximation of equation (3.33) because it gives good results in practice.

Results We show in figure 3.14 how silhouette optimization can help refine the results of mesh registration. When the quality of the segmented silhouettes is good enough, we use the described optimization as a post-process to improve the quality of the deformation recovered by the method that we will present in chapter 4.

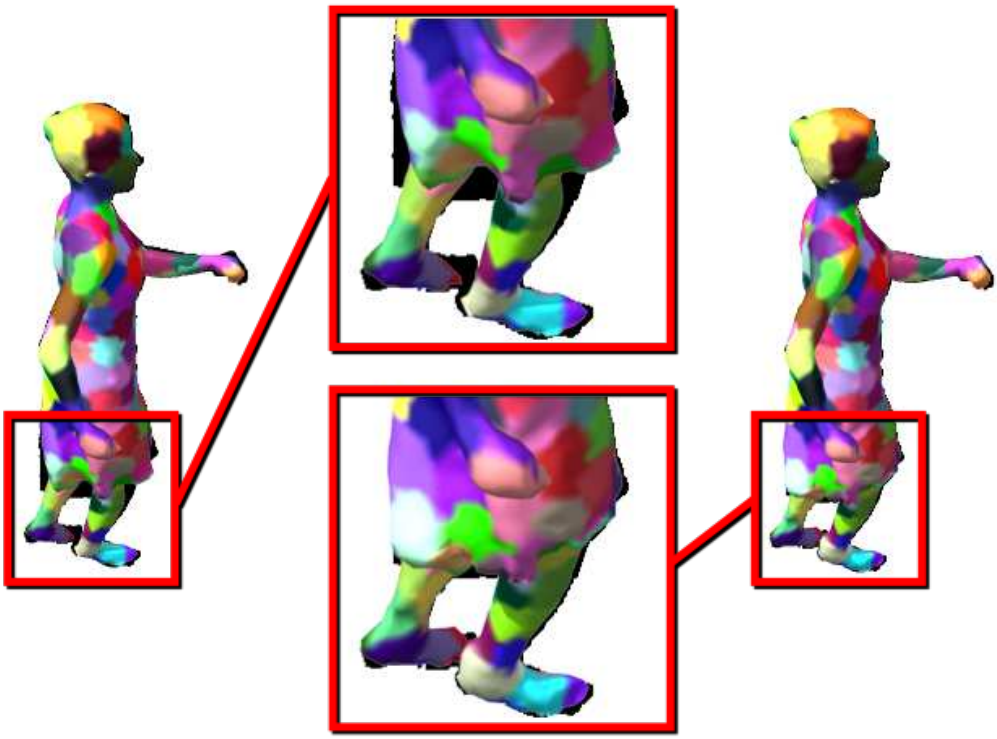


Figure 3.14: Silhouette optimization allows to better fit the silhouette data as a post-process. The mesh on the left is the result of the probabilistic registration algorithm presented in Chapter 4. It can be observed that after the mesh registration step, the overlap with the silhouette (displayed in black underneath) is not necessarily optimal. After a couple of iterations of silhouette optimization, the overlap is much more precise.

4 Conclusion

In this chapter, we have presented a broad survey of the state-of-the art on mesh deformation models. Among the reviewed methods, we have identified two interesting ideas for data-driven deformation: first we have seen that embedded deformation methods allowed to decouple the parametrization of the deformation from the complexity of the deformed geometry. Second, we have seen that elasticity intrinsically carried non-linearities in its formulation as soon as large deformations need to be represented. We therefore have decided to explicitly optimize on local rotations of the surface, when most other methods optimize on the locations of control vertices and implicitly recover the local rotations from this sparse set of vertex locations.

Building on these ideas, we have presented a deformation model and an optimization framework. The deformation model is strictly surface-based an builds a deformation graph that matches the connectivity of the deformed mesh. We have proposed to divide this mesh into small surface elements whose size depends on the precision at which we wish to sample the deformation. We have emulated an elastic behavior of the surface by defining a

well conditioned rigidity force between neighboring patches. Finally, we have presented an optimizer that approximates variations of the local rotations of the surface with exponential maps, and thus has access to precise analytical derivatives of any energy term that depends on vertex positions.

We have presented a number of such energy terms through three applications. The results we have obtained show that this deformation model is a useful and generic tool for the recovery of surface deformation from visual data. We will show in the following chapter how it can be used as a prior in the inference for the deformation of arbitrary shapes observed in multi-camera studios.

References

[1] Anguelov, D., P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis
2005. Scape: shape completion and animation of people. *SIGGRAPH*.

[2] Baran, I., D. Vlasic, E. Grinspun, and J. Popovic
2007. Automatic rigging and animation of 3D characters. In *SIGGRAPH*.

[3] Barr, A.
1981. Superquadrics and angle-preserving transformations. *CGA*.

[4] Barr, A. H.
1984. Global and local deformations of solid primitives. *SIGGRAPH*.

[5] Bartoli, A., M. Perriollat, and S. Chambon
2010. Generalized thin-plate spline warps. *IJCV*.

[6] Bickel, B., M. Lang, M. Botsch, M. A. Otaduy, and M. Gross
2008. Pose-space animation and transfer of facial details. In *EUROGRAPHICS*.

[7] Blanz, V., C. Basso, T. Vetter, and T. Poggio
2003. Reanimating Faces in Images and Video. In *EUROGRAPHICS*.

[8] Blanz, V. and T. Vetter
1999. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*.

[9] Bookstein, F.
1989. Principal warps: thin-plate splines and the decomposition of deformations. *PAMI*.

[10] Botsch, M., D. Bommes, and L. Kobbelt
2005. Efficient linear system solvers for mesh processing. In *IMA Conference on the Mathematics of Surfaces*.

[11] Botsch, M., M. Pauly, M. Gross, and L. Kobbelt
2006. Primo: Coupled prisms for intuitive surface modeling. *EUROGRAPHICS*.

[12] Botsch, M., M. Pauly, M. Wicke, and M. H. Gross
2007. Adaptive space deformations based on rigid cells. *EUROGRAPHICS*.

[13] Botsch, M. and O. Sorkine
2008. On linear variational surface deformation methods. *TVCG*.

[14] Brochu, T. and R. Bridson
2009. Robust topological operations for dynamic explicit surfaces. *SIAM Journal on Scientific Computing*.

[15] Cagniart, C., E. Boyer, and S. Ilic
2009. Iterative Mesh Deformation for Dense Surface Tracking. In *3DIM*.

[16] Cagniart, C., E. Boyer, and S. Ilic
2010. Free-from mesh tracking: a patch-based approach. In *CVPR*.

[17] Cagniart, C., E. Boyer, and S. Ilic
2010. Iterative Deformable Surface Tracking in Multi-View Setups. In *3DPVT*.

[18] Carmo, M. P. D.
1976. *Differential Geometry of Curves and Surfaces*.

[19] Chang, W. and M. Zwicker
2009. Range scan registration using reduced deformable models. *EUROGRAPHICS*.

[20] Chen, M. B., O. Weber, and C. Gotsman
2009. Variational harmonic maps for space deformation. In *ACM SIGGRAPH*.

[21] Chui, H. and A. Rangarajan
2003. A new point matching algorithm for non-rigid registration. *CVIU*.

[22] Cootes, T. F., G. J. Edwards, and C. J. Taylor
1998. Active Appearance Models. *ECCV*.

[23] Cootes, T. F., C. J. Taylor, D. H. Cooper, and J. Graham
1995. Active shape models-their training and application. *CVIU*.

[24] Coquillart, S.
1990. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. *SIGGRAPH*.

[25] de Aguiar, E., Sigal Leonid, A. Treuille, and J. K. Hodgins
2010. Stable Spaces for Real-time Clothing. In *SIGGRAPH*.

[26] de Aguiar, E., C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun
2008. Performance capture from sparse multi-view video. In *SIGGRAPH*.

[27] Ekman, P. and W. Friesen
1978. Facial action coding system: A technique for the measurement of facial movement: Investigator's guide 2 parts.

[28] Eldar, Y., M. Lindenbaum, M. Porat, S. Member, and Y. Y. Zeevi
1997. The farthest point strategy for progressive image sampling. *IEEE Trans. on Image Processing*.

[29] Gall, J., C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel
2009. Motion capture using joint skeleton tracking and surface estimation. In *CVPR*.

[30] Girshick, R., J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon
2011. Efficient regression of general-activity human poses from depth images. In *ICCV*.

[31] Greminger, M. A. and B. J. Nelson
2004. Vision-based force measurement. *PAMI*.

[32] Huang, J., X. Shi, X. Liu, K. Zhou, L.-Y. Wei, S.-H. Teng, H. Bao, B. Guo, and H.-Y. Shum
2006. Subspace gradient domain mesh deformation. In *SIGGRAPH*.

[33] Huang, P., A. Hilton, and J. Starck
2010. Shape similarity for 3d video sequences of people. *IJCV*.

[34] Ilic, S. and P. Fua
2007. Non linear beam model for tracking large deformations. In *ICCV*.

[35] James, D. L. and C. D. Twigg
2005. Skinning Mesh Animations. *SIGGRAPH*.

[36] Karypis, G. and V. Kumar
1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*

[37] Kass, M., A. Witkin, and D. Terzopoulos
1988. Snakes: Active contour models. *IJCV*.

[38] Kavan, L., S. Collins, J. Zara, and C. O'Sullivan
2008. Geometric skinning with approximate dual quaternion blending.

[39] Kircher, S. and M. Garland
2008. Free-form motion processing. *SIGGRAPH*.

[40] Ladislav Kavan, Peter-Pike Sloan, C. O.
2010. Fast and efficient skinning of animated meshes. *EUROGRAPHICS*.

[41] Lewis, J. P., M. Cordner, and N. Fong
2000. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *SIGGRAPH*.

[42] Li, H., P. Roivainen, and R. Forcheimer
1993. 3-D Motion Estimation in Model-Based Facial Image Coding. *PAMI*.

[43] Li, H., R. W. Sumner, and M. Pauly
2008. Global correspondence optimization for non-rigid registration of depth scans. *Comput. Graph. Forum*.

[44] Lipman, Y., O. Sorkine, D. Levin, and D. Cohen-Or
2005. Linear rotation-invariant coordinates for meshes. In *SIGGRAPH*.

[45] Meyer, M., M. Desbrun, P. Schröder, and A. H. Barr
2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *VSM*.

[46] Nealen, A., M. Mueller, R. Keiser, E. Boxerman, and M. Carlson
2006. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*.

[47] Osher, S. and J. A. Sethian
1988. Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*.

[48] Parke, F. I.
1974. *A parametric model for human faces.* PhD thesis.

[49] Pons, J.-P. and J.-D. Boissonnat
2007. Delaunay deformable models: Topology-adaptive meshes based on the restricted delaunay triangulation. In *CVPR*.

[50] Pons, J.-P., G. Hermosillo, R. Keriven, and O. Faugeras
2006. Maintaining the point correspondence in the level set framework. *Journal of Computational Physics*.

[51] Prautzsch, H., W. Boehm, and M. Paluszny
2002. *Bezier and B-Spline Techniques*.

[52] Rydfalk, M.
1987. CANDIDE, a parameterized face. Technical report.

[53] S. Toledo
2003. Taucs: A Library of Sparse Linear Solvers, Version 2.2. Technical report, .

[54] Salzmann, M. and P. Fua
2010. Deformable surface 3d reconstruction from monocular images. *Synthesis Lectures on Computer Vision*.

[55] Salzmann, M., F. Moreno-Noguer, V. Lepetit, and P. Fua
2008a. Closed-form solution to non-rigid 3d surface registration. In *ECCV (4)*, D. A. Forsyth, P. H. S. Torr, and A. Zisserman, eds.

[56] Salzmann, M., J. Pilet, S. Ilic, and P. Fua
2007. Surface deformation models for nonrigid 3d shape recovery. *PAMI*.

[57] Salzmann, M., R. Urtasun, and P. Fua
2008b. Local deformation models for monocular 3d shape recovery. In *CVPR*.

[58] Sederberg, T. W. and S. R. Parry
1986. Free-form deformation of solid geometric models. In *SIGGRAPH*.

[59] Sheffer, A. and V. Kraevoy
2004. Pyramid coordinates for morphing and deformation. In *3DPVT*.

[60] Shotton, J., A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake
2011. Real-Time Human Pose Recognition in Parts from a Single Depth Image. In *CVPR*.

[61] Solina, F. and R. Bajcsy
1990. Recovery of parametric models from range images: The case for superquadrics with global deformations. *PAMI*.

[62] Sorkine, O. and M. Alexa
2007. As-rigid-as-possible surface modeling. In *EUROGRAPHICS*.

[63] Sorkine, O., D. C. Or, Y. Lipman, M. Alexa, C. Rössl, and H. P. Seidel
2004. Laplacian surface editing. In *EUROGRAPHICS*.

[64] Stoiber, N., R. Seguier, and G. Breton
2010. Facial animation retargeting and control based on a human appearance space. *Computer Animation and Virtual Worlds*.

[65] Sumner, R. W., J. Schmid, and M. Pauly
2007. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007*.

[66] Sumner, R. W., M. Zwicker, C. Gotsman, and J. Popovic
2005. Mesh-based inverse kinematics. In *SIGGRAPH*.

[67] Szeliski, R. and J. Coughlan
1997. Spline-based image registration. *IJCV*.

[68] Taylor, C. and D. J. Kriegman
1994. Minimization on the lie group so(3) and related manifolds. Technical report, Yale University.

[69] Terzopoulos, D. and D. Metaxas
1991. Dynamic 3d models with local and global deformations: deformable superquadrics. *PAMI*.

[70] Terzopoulos, D., J. Platt, A. Barr, and K. Fleischer
1987. Elastically deformable models. In *SIGGRAPH*.

[71] Terzopoulos, D. and A. Witkin
1988. Physically based models with rigid and deformable components. *CGA*.

[72] Vlasic, D., I. Baran, W. Matusik, and J. Popović
2008. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH*.

[73] Wang, X., G. Ananthasuresh, and J. Ostrowski
2001. Vision-Based Sensing of Forces in Elastic Objects. *Sensors and Actuators*.

[74] Wardetzky, M., S. Mathur, F. Kälberer, and E. Grinspun
2007. Discrete laplace operators: no free lunch. In *EUROGRAPHICS*.

[75] White, R., K. Crane, and D. Forsyth
2007. Capturing and Animating Occluded Cloth. In *SIGGRAPH*.

[76] Wojtan, C., N. Thürey, M. Gross, and G. Turk
2009. Deforming meshes that split and merge. In *SIGGRAPH*.

[77] Yu, Y., K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, and H. Y. Shum
2004. Mesh editing with poisson-based gradient field manipulation. In *SIGGRAPH*.

[78] Zaharescu, A., E. Boyer, and R. P. Horaud
2011. Topology-adaptive mesh deformation for surface evolution, morphing, and multi-view reconstruction. *PAMI*.

[79] Zayer, R., C. Rössl, Z. Karni, and H.-P. Seidel
2005. Harmonic guidance for surface deformation. In *EUROGRAPHICS*.

CHAPTER 4

Surface Tracking by Probabilistic Mesh Registration

In this chapter we address the problem of automatically recovering temporally consistent animated 3D models of arbitrary shapes observed in multi-camera setups. The approach we propose considers a sequence of independently reconstructed surfaces and iteratively deforms a reference mesh to fit these observations. To effectively cope with the parasite and missing geometry that are inherent to reconstructions from visual data, we build on two ideas. Firstly, we use the patch-based deformation and numerical optimization framework that were presented in the previous chapter. This framework increases robustness by providing natural integration domains over the surface for noisy data and by enabling to express simple patch-level rigidity constraints. Secondly we propose a Bayesian method for mesh registration that accounts for the uncertainty in the data acquisition process by embedding the optimization in an Expectation-Maximization formulation. Extensive experiments on various 4D datasets show that these two ideas allow to process complex scenes involving several objects of arbitrary nature, while robustly handling missing data and relatively large reconstruction artifacts.

1 Introduction

In chapter 2, the study of prior art established that finding correspondence between two deformed instances of an object is a challenging problem. We discussed how most of the available feature-matching methods were designed to match temporally uncorrelated observations of deformable objects, and how this fails to exploit fully the strong temporal

redundancy in scene appearance and structure that exists between two successive frames of a sequence captured in a multi-view studio. Furthermore, we noticed that when trying to infer dense motion in this context, many approaches rely heavily on silhouettes to ensure of the correctness of correspondences computed by other means such as optical flow or sparse feature matching. This is a strong indication that the information brought by silhouettes on the scene geometry for a given frame can be considered a more reliable cue than the matches established between successive images or successive reconstructions during the sequence. Not only is this information relatively reliable, it is also sufficient to infer motion in some cases. For example, Vlasic et al. [32] use silhouettes exclusively as input data to perform dense surface tracking of humans.

The question addressed by this chapter is that of inferring the motion of 3D surface points from a series of snapshot of the scene structure alone. From one frame to the next, these independent 3D reconstructions vary in their sampling of the scene structure. However, given a high enough frame-rate, the independently reconstructed surfaces represent shapes that are both similar and spatially close to one another. We aim at exploiting this proximity to establish dense correspondence between frames. We cast the problem as geometric registration of a reference mesh to the sequence of reconstructed meshes, as displayed in figure 4.1. This reference mesh, once deformed to fit every observed mesh, constitutes a temporally coherent sampling of the scene geometry across the sequence.

The first key aspect of our approach is its generality. In contrast to other works that compute motion from geometric cues exclusively [32, 16, 10], we do not assume a strong articulated structure and instead use the robust mesh deformation framework presented in chapter 3 to regularize the tracking. In that respect, our work is closer to that of [12, 11]. Because our regularization energy is purely surface based and does not make assumptions on the nature of the object, our approach can process scenes involving several objects of arbitrary nature in a single mathematical formulation.

The second key aspect of our work lies in the *probabilistic formulation* of the registration problem that accounts for potential errors in the silhouette segmentation and 3D reconstruction stages, and increases the robustness to artifacts in the geometry. As we will discuss in this chapter, modeling for the uncertainty of the acquisition process is a crucial component of our work because of the more generic and thus less constraining regularization that we use.

In the next section, we briefly recall several important ideas on geometric registration, as well as a review of the works directly related to ours in terms of probabilistic modeling. The remainder of the chapter presents our developments in details and particularly focuses on our proposal of probabilistic formulation for mesh registration. We present our results and demonstrate the performance and versatility of our algorithm. This evaluation is performed qualitatively and quantitatively on several sequences, including scenes that involve several objects of different nature. We show that our approach not only allows to process such scenes with a single mathematical formulation and deformation model, but also performs comparably well to previous art on simpler scenes.

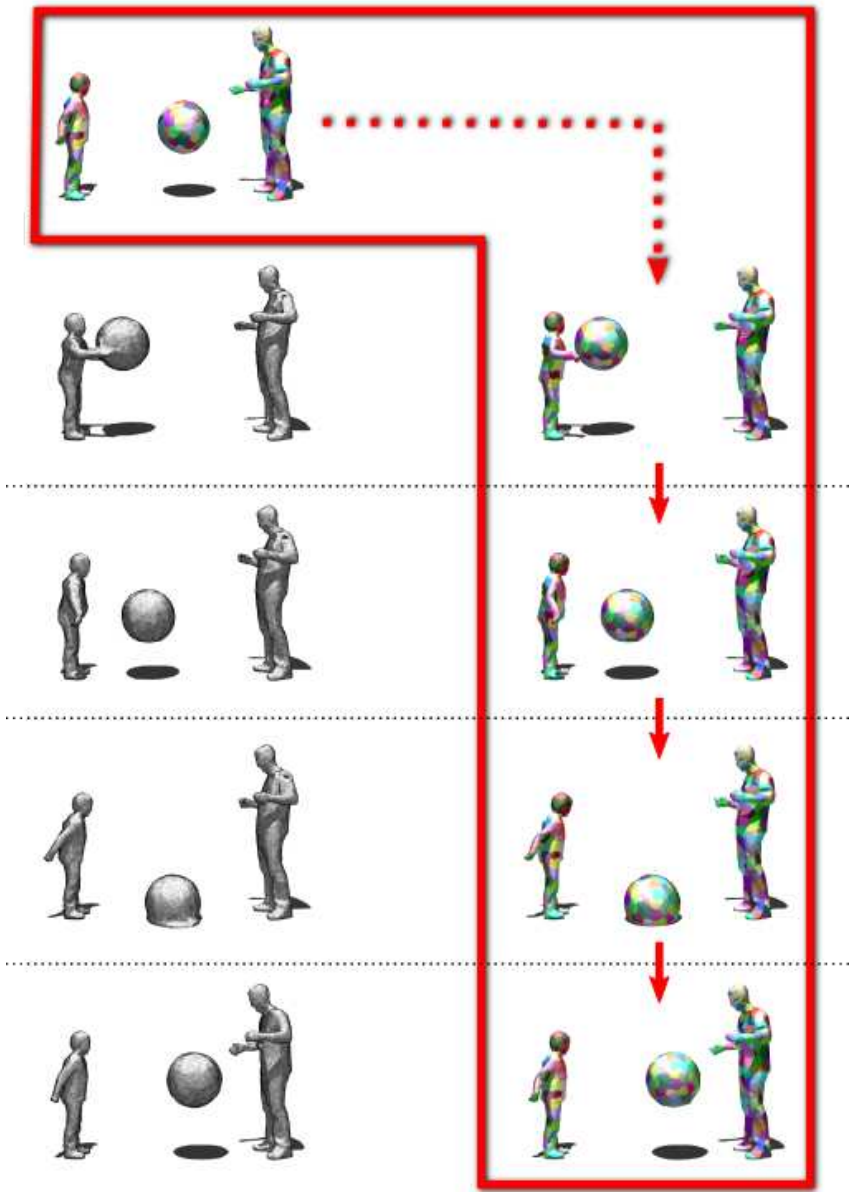


Figure 4.1: Our method performs the geometric registration of a reference mesh to a temporal series of independently reconstructed meshes. The left column contains the meshes that were independently reconstructed from the image data. A patch structure is associated to one of these meshes (usually the first). This reference mesh is deformed to fit the rest of the sequence. The resulting animated mesh is a 4D model of the sequence.

2 Related Works on Geometric Registration

The following discussion of geometric registration methods considers that the two shapes that need to be registered are already be roughly aligned. This hypothesis is consistent with our setting where the registration happens between two successive frames of a temporal sequence. For a discussion of coarse alignment methods, the interested reader can refer to the section on matching in chapter 2 and its listed references.

2.1 Rigid registration

For simplicity purposes, we first illustrate the idea of geometric registration within the restricted case of rigid registration of point clouds. In that case, the goal is to find an optimal rigid transformation $\mathbf{R}, \mathbf{t}$ that aligns two sets of 3D points $\mathcal{X} = \{\mathbf{x_i}\}$ and $\mathcal{Y} = \{\mathbf{y_j}\}$. This is a relatively constrained problem as there are only 6 degrees of freedom. The issue is that we don't know the correspondence between the points of $\mathcal{X}$ and the points of $\mathcal{Y}$.

ICP In the literature, the prevailing way to address rigid registration when the two shapes are almost aligned is the Iterative Closest Point (ICP) method by Besl and McKay [1]. The idea is to minimize the following registration error with respect to $\mathbf{R}, \mathbf{t}$:

$$\underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_i \min_j \|\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y_j}\|^2. \tag{4.1}$$

This minimization happens by iterating two steps until convergence:

1. given the current approximation $\mathbf{R}, \mathbf{t}$, find $\forall \mathbf{x_i} \in \mathcal{X}$ the point $\mathbf{y}(i)$ closest to $\mathbf{R}\mathbf{x_i} + \mathbf{t}$.
2. given these closest points $\mathbf{y}(i)$, update $\mathbf{R}, \mathbf{t}$ to the rigid transformation minimizing $\sum_i \|\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y}(i)\|^2$

The first step is a simple nearest neighbor search and can be tackled by brute force or with more sophisticated space-partitioning methods. The second step has a closed-form solution that can be found with the method by Horn [20], or other methods such as polar decomposition of the coordinate covariance matrix. Overall, ICP is proven to converge to a local minimum.

Different error functions Since the original ICP algorithm was presented, numerous extensions have been proposed. Rusinkiewicz and Levoy [28] provide a good overview, as well as a comparison of different strategies.

A notable modification was presented by Chen and Medioni [6] who propose to minimize the *point-to-plane distance* in place of the *point-to-point distance*. Their idea is to use information on the surface normals when it is available to get a better approximation of the residual distance of each transformed $\mathbf{x_i}$ to the target surface. Once the closest point $\mathbf{y}(i)$ is found for all $\mathbf{x_i}$, the second step minimizes $\sum_i \left[\mathbf{n_i}^T(\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y}(i)\right]^2$. This first-order

approximation of the target surface, that considers more than the points of $\mathcal{Y}$ only, was shown to significantly improve the convergence rate by Pottmann et al. [25].

Another path to improve ICP's performance and robustness is to weight the individual contributions of the correspondences, and to minimize $\sum_i w_i \|\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y}(i)\|^2$. For example, [14] uses robust estimators to weight the contributions of correspondences depending on the residual distance between points. This prevents the quadratic penalization of outlying correspondences that can significantly impact the update of the rigid transformation in the second step.

Handling outliers As noted by [28], an extreme case of weighting correspondences is to ignore some of them completely. This rejection can happen in the matching stage, when the closest point $\mathbf{y}(i)$ in the target set is computed for each model point $\mathbf{x_i}$, by making sure that the two points are compatible with respect to some criterion. A common criterion is to check that the angles between the normals are below some threshold.

The rejection of correspondence can also happen when forming the objective function in the second step, by pruning correspondences involving points that are too distant, or rejecting a percentage of the correspondence after sorting them by their residual error [7]. The rejection of a percentage of worst correspondences can for example improve the registration of two partial reconstructions which are only expected to have a partial overlap. The idea comes from more conventional least-square regression tasks where it is used to limit the influence of outliers. It integrates naturally in the ICP framework and it is shown in[7] that this does not affect the convergence properties of the algorithm.

2.2 Probabilistic formulation

The first step of the ICP iterations makes a hard decision when computing the corresponding point $\mathbf{y}(i)$ for every model point $\mathbf{x_i}$. This correspondence is unfortunately rarely correct, especially in the early stages of the registration, and overall because of the different sampling of the shape by $\mathcal{X}$ and $\mathcal{Y}$. In other words, if $(\mathbf{R}^*, \mathbf{t}^*)$ is the optimal registration, there is no guarantee at a given point of the inference that $\mathbf{y}(i)$ will be the point closest to $\mathbf{R}^*\mathbf{x_i} + \mathbf{t}^*$. Even if it were, the fact that it is the closest point does not mean it is the same point on the surface.

As such, some works evaluate the likelihood of assignments between points of $\mathcal{X}$ and points of $\mathcal{Y}$. Instead of making a hard choice on the assignment of $\mathbf{x_i}$ to a unique point of $\mathcal{Y}$, the *softassign* approach of Rangarajan et al. [26] builds a $|\mathcal{X}| \times |\mathcal{Y}|$ assignment matrix W that weights the correspondence between every point of the two clouds. These weights encode fuzzy assignments between the two point sets. The weights w_{ij} are computed as Gaussians of the distance between $\mathbf{x_i}$ and $\mathbf{y_j}$ and normalized to favor one-to-one correspondences. The variance of the Gaussians, which determines how fuzzy the assignments are, is controlled by a temperature term. The algorithm alternates between the estimation of the

now *soft* correspondences, and the update of transformation parameters that minimizes:

$$\sum_i \sum_j w_{ij} \|\mathbf{R}\mathbf{x_i} + \mathbf{t} - \mathbf{y}_j\|^2. \tag{4.2}$$

The variance parameter and its multi-scale interpretation Just like ICP, the algorithm alternates the two steps until convergence, but does so by decreasing progressively the temperature, performing what is called *deterministic annealing*. To understand what the algorithm does, we must first notice that the higher the temperature, the fuzzier the assignments are. Given a fixed very high variance for example, every point will be associated to every other point with roughly the same weight, and the optimization will progressively align the centers of the two point clouds, as well as their moments of inertia. For a very small variance, the assignment becomes very selective and points get assigned to one point much more than the others, bringing back to the behavior of ICP.

Granger and Pennec [17] propose to discuss the annealing of the variance parameter as the *scale* parameter in a coarse-to-fine approach. This is justified by the fact that a high temperature speeds-up the coarse registration process and prevents the algorithm from getting trapped in local minima caused by erroneous associations. Inspired by [26], they look at the registration problem from a probabilistic standpoint. However, they advance to a fully Bayesian formulation by embedding it in an Expectation-Maximization (EM) framework, coining the name EM-ICP. We will return to the reasons the EM formalism fits the problem particularly well when presenting our approach.

Horaud et al. [19] remark that annealing scheduling on the temperature parameter fails to consider the variance as a parameter of the optimization. As such, deterministic annealing prevents from estimating the characteristics of the noise in the data and prevents from fully benefiting from the convergence properties of the EM-algorithm.

Handling outliers In an extension of softassign named Robust Point Matching (RPM), Chui and Rangarajan [9] deal with outliers by adding a cluster to each point cloud and associating it with the points from the other cloud that can not be matched. In practice, this amounts to adding a virtual point to each cloud, but to give it a much bigger variance in his search for correspondence. This adds a row and a column to the assignment matrix W. After normalization, outliers tend to have high weights in these additional rows or columns. During the update step, these rows and columns are not included in the error function, which effectively diminishes the impact of outliers on the estimation of the rigid transformation.

Horaud et al. [19] essentially do the same, but the idea integrates much more elegantly into the EM-framework than it does into softassign. They model the outliers as the effect of a random point-generating process that creates points in 3D space with a uniform probability distribution function distributed on the scene's bounding box. We use a similar model in our work and will present the details in section 3.

2.3 Non rigid registration

Extending ICP to the non-rigid case is a challenging task, mostly because of the additional degrees of freedom. The transformation being much less constrained than in the rigid case, it becomes difficult to regularize the correspondence information. Because the parametrization of the deformation and its regularization were already discussed in previous chapters, we focus here our discussion on the handling of uncertainty in the data term.

2.3.1 Deterministic approaches

Choi and Szymczak [8] tackle roughly the same problem that we are addressing. They consider a reference volumetric mesh in its rest state and treat it as an elastic body that needs to be fitted to a series of temporally consecutive 3D reconstructions. They do not formulate it exactly as the registration of point clouds but instead pre-compute signed distance volumes to the target surface. Then the residual energy is computed as the sum of sampled residual distances at the locations of the boundary vertices of the deformed model. This is a common way to speed up registration tasks and avoid nearest neighbor searches when approximating the fitting error (see the distance transform used to speed up ICP in [14] for example). Their approach introduces a measure of confidence in the data term by weighting the residuals of every vertex using a heuristic that trusts error terms more when the normal to the currently deformed model is aligned with the gradient of the distance volume. Their method however does not consider the possibility of outliers in the scene data. These would undoubtedly cause problems because they would significantly modify the signed distance volume. As this distance volume is computed once and for all at the start, it collapses all the information brought by the scene points in a single representation, forgetting their individual and localized uncertainty. Furthermore, the presented results suggest that their energy contains numerous local minima in which their optimizer can get trapped, despite some additional and computationally costly heuristics preventing self collisions of the deformed model.

de Aguiar et al. [11] insist on their use of image features in the registration process, but as discussed in chapter 2, a number of implementation details confirm the prevailing influence of silhouettes in their registration algorithm. Indeed, they do not trust the sparse correspondences directly but instead progressively deform the mesh along the direction of these matches, making sure that the silhouette error decreases at each step. Finally they refine the pose using the silhouette information alone to account for regions where photometric feature matches could not be established. As such, it appears that they in fact minimize the silhouette reprojection error in their method. They therefore can be classified as a geometric registration method. However, they do not account at all for possible errors in the segmented silhouettes.

The work by Li et al. [22] on the registration of range scans is the closest to ours in terms of parametrization and regularization of the deformation. There are slight differences between this approach and ours in that respect, and we have discussed them in chapter 3.

In contrast, the differences in the data-term of the optimization are significant. First of all, they do not sample the geometric information densely and only look for correspondence for the graph nodes of their optimization structure, which correspond to the patch centers in our formulation. Secondly, these correspondences are established by projecting each node on the range image of the target shape (corrected of an average rigid transformation between the two shapes) and is not established like ICP by nearest neighbors search in 3D. The last difference is related to the problem of outliers: because part of the surface in the first range image might not be seen in the second range image, there might be nodes that will not find a reliable match. The way they solve this problem is by weighting the correspondences with weights that are also variables in the optimization: $E_{data} = \sum_i w_i \|\mathbf{x_i} - \mathbf{y}(i)\|^2$, and introducing an energy term favoring non-zero w_is: $E_w = \sum_i (1 - w_i)^2$. In effect, if the correspondence is bad and $\|\mathbf{x_i} - \mathbf{y}(i)\|$ is large, the optimizer will benefit from the slack offered by w_i and progressively reduce it to zero. This idea is related to the variable weights and outlier class of softassign or EM-ICP, and offers the same slack to the optimizer. However, while EM optimizes these weights *implicitly* in the E-step, [22] make it an explicit variable of the optimization and make its complexity grow. This probably explains why they only sample data at control nodes and not more densely on the model.

Shinya [29] propose to register a mesh to subsequent observations for the purpose of sequence compression. Similar to our setting, they make no assumptions on the nature of the observed objects and aim at designing a generic non-rigid mesh registration algorithm. They display result on very clean data involving limited motion. They display failure cases that occur when the deformations become to large. These failures could be the result of their regularization energy which operates at the vertex level by penalizing changes of angles between triangles, changes of edge lengths and changes of triangle areas. In the end, they use user-specified positional constraints to guide the deformation.

Our early experiments on mesh registration [2, 4, 3] were fully deterministic and only associating the closest point in the target set to each point of the deformed model. This had two major consequences: first, the convergence rate was considerably worse than in the proposed method, because hard assignments do not benefit from the fuzzy multi-scale effects that were just discussed for probabilistic approaches in the rigid case. The second issue was that reconstruction artifacts and outliers were not accounted for and impacted the deformation as strongly as correctly reconstructed points.

2.3.2 Probabilistic approaches

The probabilistic approaches to non-rigid registration are directly related to the approaches of the rigid case in their handling of data uncertainty. They mostly differ on their choice of deformation model.

Articulated Horaud et al. [19] extend their rigid registration method based on EM to tackle the alignment of articulated structures. Their method iteratively registers the different parts that constitute the articulated body: first, the root part is registered to the tar-

get data using their *Expectation Conditional Maximization Point Registration* (ECMPR) method. Then, the points of the target cloud that are explained by the root in its optimized position are removed from the data set. This procedure is repeated iteratively for all the remaining parts of the articulated body.

Thin-plate splines and other RBFs Chui and Rangarajan [9] extend the softassign algorithm to the non-rigid case and name their method *Thin Plate Spline Robust Point Matching* (TPS-RPM). As this name indicates, this approaches parametrizes the non-affine part of the deformation using the thin-plate-spline (TPS) as basis functions. The optimization thus happens on the parameters of an affine warp plus as many warping coefficients as there are dimensions to the TPS Kernel. As noted by Myronenko and Song [24], the TPS in 3D uses basis functions that are non differentiable at control points. Therefore, [24] presents the *Coherent Point Drift* algorithm, based on Gaussian basis functions. It is worth noting that both approaches use deterministic annealing on the variance instead of estimating it as a variable. The main issue with these methods however, already discussed in chapter 3, is that these parametrization enforce smoothness over the whole scene's volume, and not only on the object itself. They therefore can present artifacts when two geodesically distant parts of the surface are close in 3D space, and they have limited applicability to large deformations.

Mesh deformation Our work shares common points with the work by Starck et al. [31] who deform a generic human template to fit a visual hull reconstruction. They start by roughly aligning the template model with the visual hull, then perform deformable registration. To this end, they adapt the TPS-RPM [9]. Like TPS-RPM, they rely on the fuzzy assignments of softassign. However they establish this smooth correspondence between the voxels of the visual hull reconstruction and the vertices of the deformed template. They also use deterministic annealing on the variance, that they refer to as temperature parameter. The main resemblance with our work lies in the introduction of a regularization based on a mesh deformation energy to replace the TPS. However this deformation method optimizes directly on the vertices of the template, which results in a very high-dimensional non-linear minimization. Moreover, and just like the TPS-RPM method, the approach lacks a probabilistic formulation.

Early reference to deformable EM-ICP We finish this overview of related works by showing that the idea of associating EM with an elastic model is far from new. In the earliest (1992) reference we could find, Hinton et al. [18] propose a method for fitting deformable templates of letters to handwritten characters. Their deformable model is a 1D spline on which "Gaussian ink-generators" are placed (see figure 4.2). The EM inference then optimizes the position of these ink generators while estimating the soft assignments between observed black pixels and each of the ink generators. Additionally, the authors include a uniform noise process in the model so that pixels that were erroneously segmented as ink can be rejected. The affine component of the deformation energy is absorbed by always computing an optimal affine transform between the reference pose and the current

Figure 4.2: Illustration of the idea presented in the work by Hinton et al. [18]. The digit model is represented by a spline on which Gaussian ink-generators are placed. Each pixel of the observed image can be explained by each of these ink-generators. The registration and assignment problems are simultaneously solved for in an EM framework.

approximation of the deformed spline. In following work [27] the variance is correctly updated.

Interestingly, this work was published concurrently to the ICP algorithm. The authors therefore do not refer to their method as some sort of EM-ICP, as the term ICP had not been coined yet. This is simply a geometric registration method, solved with EM, that uses a deformable elastic model and a probabilistic model of the data generation process. In that respect, and even though handwritten digit recognition is seemingly very far from our problem, we believe this work to be the closest to ours.

3 Probabilistic Mesh Registration

Our work addresses data-driven mesh deformation, and we cast the problem as the geometric registration of 3D point sets. In a Bayesian context, this means that given a set of observed 3D points and an estimate of the current pose of the mesh, we are faced with a maximum-a-posteriori (MAP) estimation problem where the joint probability distribution of data and model must be maximized:

$$\max_{\mathbf{\Theta}} \ln P(\mathcal{Y}, \mathbf{\Theta}), \tag{4.3}$$

where $\mathcal{Y} = \{y_i\}_{i=1:m}$ is the set of observed 3D points $\{\mathbf{y_i}\}_{i=1:m}$ associated with the surface normals at these points. In the absence of knowledge on the nature of the shape, we model a probability distribution over the range of shape deformations by seeding patches

on a reference surface and making the approximation

$$P(\mathbf{\Theta}) \propto e^{-E_r(\mathbf{\Theta})}, \tag{4.4}$$

where $E_r(\mathbf{\Theta})$ is the rigidity energy defined in equation (3.23). This energy emulates elastic behavior with respect to the patched reference mesh.

Because our patching approach infers the connectivity of the object from the vertex connectivity, this reference mesh has to be topologically suitable, that is it has to be split wherever the surface might split during the sequence. For example if multiple objects are present in the scene the number of connected components in the reference mesh should match the number of objects. The pose defined by this reference surface is of less importance provided that the number of patches is high enough to finely sample changes of curvature with respect to the rest pose (see figure 4.6 for example).

3.1 Bayesian Model

Given this model for $P(\mathbf{\Theta})$, the likelihood $P(\mathcal{Y}|\mathbf{\Theta})$ remains to be approximated to complete the generative model. This is done with a mixture of distributions parametrized by a common variance σ^2, where each component corresponds to a patch. This requires to introduce latent variables z_i for each observation $y_i \in \mathcal{Y}$, where $z_i = k$ means that y_i was generated by the mixture component associated with P_k. We also increase the robustness of our model to outliers by introducing a uniform component in the mixture to handle points in the input data that could not be explained by the patches. This uniform component is supported on the scene's bounding box and we index it with $N_p + 1$.

$$P(y_i|\Theta, \sigma) = \sum_{k=1}^{N_p+1} \Pi_k P(y_i|z_i = k, \Theta, \sigma), \tag{4.5}$$

where the $\Pi_k = p(z_i = k|\Theta, \sigma)$ represent probabilities on the latent variables marginalized over all possible values of y_i. In other words they are prior probabilities on model-data assignments. We define them as constants $p(z_i = k)$ that add up to 1, using the expected proportion of outlier surface in the observations and the ratios of patch surfaces in the reference mesh.

The patch mixture component with index k must encode a distance between the position $\mathbf{y_i}$ and the patch P_k while accounting for the alignment of normals. For computational cost reasons, we model this distance by looking for each patch P_k in its different predicted poses (this means the positions $\{\mathbf{x_l}(v)\}_{l\in\{k\}\cup N_k, v\in P_k}$ and corresponding normals as shown in figure 4.3 for the closest vertex with a compatible normal v_i^k. We consider two points and normals to be compatible when their normals form an angle smaller than a threshold. In practice this threshold was set to $45°$ in all of our experiments. This leads to the following

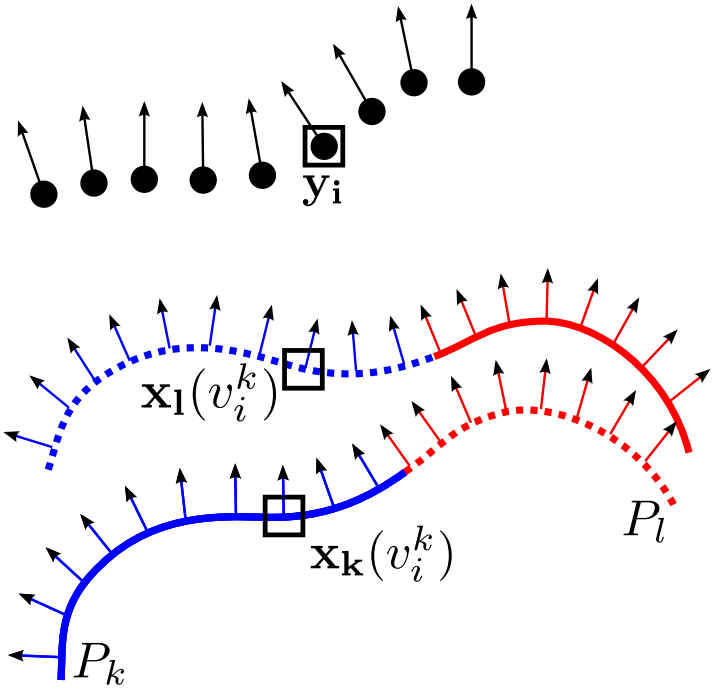
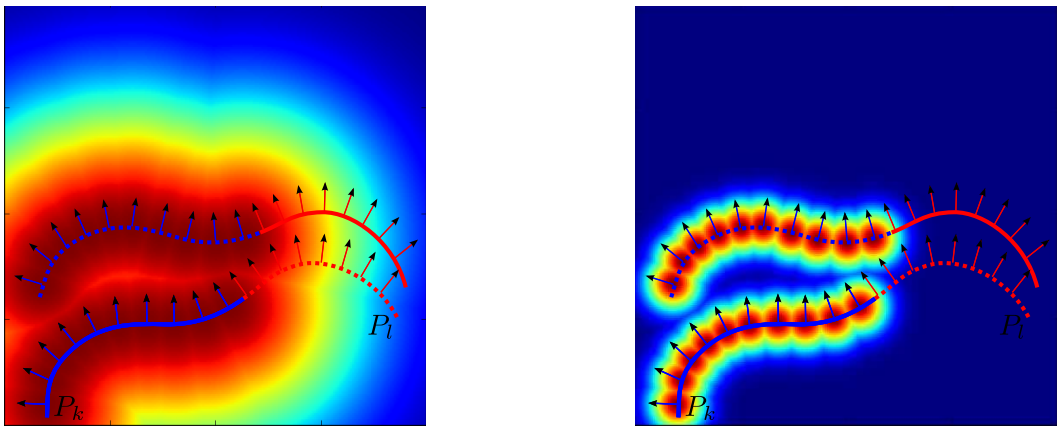


Figure 4.3: A point/normal y_i with position $\mathbf{y_i}$ from the observed data is associated to v_i^k, the closest vertex with a compatible normal among all the predictions for the patch P_k. In this case v_i^k is selected because of its position and normal in the prediction made by the neighboring patch P_l.



(a) (b)

Figure 4.4: $P(y_i|z_i = k, \mathbf{\Theta}, \sigma)$ as defined in Eq. 4.6 is a Gaussian of an approximate distance to the patch. It forms a sort of halo in space around the different predicted positions for the vertices of patches k. The subfigures display the probability density function for two values of σ. Please note that we ignore here the normal compatibility condition.

model for each component of the mixture:

$$\forall k \in [1, N_p],$$

$$P(y_i|z_i = k, \mathbf{\Theta}, \sigma) \propto \begin{cases} \mathcal{N}(\mathbf{y}_i|\mathbf{x}(v_i^k), \sigma) & \text{if } v_i^k \text{ exists} \\ \epsilon & \text{otherwise,} \end{cases} \tag{4.6}$$

where ϵ encodes for a uniform distribution defined on the scene's bounding box, and $\mathcal{N}(\mathbf{y}|\mathbf{x}, \sigma)$ is a multivariate Gaussian distribution centered in $\mathbf{x}$ with an isotropic covariance.

3.2 Expectation-Maximization

The variables z_i can not be observed, but the probabilistic model built until now allows to express their posterior distributions:

$$P(z_i = k|y_i, \mathbf{\Theta}, \sigma) = \frac{\Pi_k P(y_i|z_i = k, \mathbf{\Theta}, \sigma)}{\sum_{l=1}^{N_p+1} \Pi_l P(y_i|z_i = l, \mathbf{\Theta}, \sigma)}. \tag{4.7}$$

In other words,if we have an estimate for the parameters $\mathbf{\Theta}$ (the deformed mesh) and σ (the variance), we can evaluate for every point y_i how likely it is that it was generated by patch P_k. This places us in a position where we can use the Expectation Maximization (EM) algorithm that was originally formalized by Dempster et al. [13]. We have seen that if we have an estimate for the unknowns, we can compute posterior distributions on the latent variables. We will see that given these posterior distributions, we can optimize for the unknowns. The idea of EM can be roughly described as alternating between these two stages until convergence. For the sake of completeness, and to build some insight on the convergence properties of the algorithm, we recall in the following paragraphs a derivation of EM. We follow the derivation by Frank Dellaert [15] and adopt here the interpretation of EM as bound minimization that seems to us as the easiest to understand.

Building a bounding function We recall that we aim at maximizing $P(\mathcal{Y}|\mathbf{\Theta}, \sigma)$, which we rewrite as the marginalization over the hidden variables of the joint probability.

$$\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) = \ln \sum_Z P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma), \tag{4.8}$$

Following [15], we can write for any positive real valued function $q(Z)$ defined on the space of latent variables:

$$\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) = \ln \sum_Z q(Z) \frac{P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)}{q(Z)}, \tag{4.9}$$

If we add the constraint that the values of $q(Z)$ must sum up to 1, the concavity of the log function allows to write a bound on the function of interest:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq -\sum_Z q(Z) \ln \frac{P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)}{q(Z)}. \tag{4.10}$$

Finding an optimal bounding function Let us consider that we have a current estimate $(\mathbf{\Theta}^t, \sigma^t)$, of the variables we wish to optimize. To take a step that decreases $-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma)$, it suffices decrease the bounding function, after carefully choosing $q(Z)$ so that the bounding function will touch the bounded function at $(\mathbf{\Theta}^t, \sigma^t)$. [15] finds $q(Z)$ by minimizing the difference between the two while enforcing $\sum q(Z) = 1$ with Lagrange multipliers.

He shows that choosing $q(Z) = P(Z|Y, \mathbf{\Theta}^t, \sigma^t)$ satisfies the requirements: it indeed sums up to 1 and yields a bounding function that touches the bounded function at the point of interest. This means that the bounding function should be the *expected complete-data log-likelihood conditioned by the observed data*.

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq const - E_Z\left[\ln P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)|Y, \mathbf{\Theta}^t, \sigma^t\right]. \tag{4.11}$$

A practical expression of the bound The challenge is now to find an expression of the bounding function in equation (4.11) that we will be able to minimize. We start by rewriting $P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma)$ by making the approximation that the observation process that gave $\mathcal{Y}$ draws the y_i's from this distribution in an independent identically distributed way:

$$P(\mathcal{Y}, Z|\mathbf{\Theta}, \sigma) = \prod_{i=1}^{m} P(y_i, z_i|\mathbf{\Theta}, \sigma) \tag{4.12}$$

$$= \prod_{k=1}^{N_p+1} \prod_{i=1}^{m} \left[P(y_i, z_i = k|\mathbf{\Theta}, \sigma)\right]^{\delta_k(z_i)}. \tag{4.13}$$

The rewriting that appeared in equation (4.13) allows to make the double product move out of the logarithm and obtain sums:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq const - \sum_{k=1}^{N_p+1} \sum_{i=1}^{m} \left[E_Z[\delta_k(zi)|Y, \mathbf{\Theta}^t, \sigma^t] \ln[\,\Pi_k P(y_i|z_i = k, \mathbf{\Theta}, \sigma)\right], \tag{4.14}$$

which finally leads to the expression of the bounding function we need to minimize:

$$-\ln P(\mathcal{Y}|\mathbf{\Theta}, \sigma) \leq const - \sum_{k=1}^{N_p+1} \sum_{i=1}^{m} P(z_i = k|y_i, \mathbf{\Theta}^t, \sigma^t) \ln P(y_i|z_i = k, \mathbf{\Theta}, \sigma). \tag{4.15}$$

For practical purposes, we can rename the current estimates of the posterior probability density functions of the latent variables. These are not at all dependent on the variables of the bounding function $\mathbf{\Theta}$ and σ and are simply scalars that act as weights in the equations.

$$w_i^t(k) = P(z_i = k|y_i, \mathbf{\Theta}^t, \sigma^t) \tag{4.16}$$

We also rewrite the other terms of the equation for $k \neq N_p + 1$ by reminding that the covariance is assumed isotropic:

$$\ln P(y_i|z_i = k, \mathbf{\Theta}, \sigma) = -\frac{\|\mathbf{y_i} - \mathbf{x}(v_i^k)\|^2}{2\sigma^2} - 3\ln\sigma + const \tag{4.17}$$

Reintroducing the prior If we reintroduce the prior and move from $P(\mathcal{Y}|\mathbf{\Theta},\sigma)$ to the joint-probability $P(\mathcal{Y},\mathbf{\Theta},\sigma)$, we obtain a new bounding function $B^t(\mathbf{\Theta},\sigma)$ to minimize given a current approximate of the variables $\mathbf{\Theta}^t$ and σ^t:

$$B^t(\mathbf{\Theta},\sigma) = E_r(\mathbf{\Theta}) + 3\left[\sum_{k=1}^{N_p}\sum_{i=1}^{m} w_i^t(k)\right]\ln\sigma + \frac{1}{2\sigma^2}\sum_{k=1}^{N_p+1}\sum_{i=1}^{m}\left[w_i^t(k)\|\mathbf{y_i} - \mathbf{x}(v_i^k)\|^2\right]. \quad (4.18)$$

Although $\mathbf{\Theta}$ does not appear explicitly in the equation, it controls the deformation of the mesh, and thus of the $\mathbf{x}(v_i^k)$ vectors.

3.3 Practical minimization

The previous section recalled the EM algorithm and its mathematical derivation. In this section, we focus on how to translate these ideas in a practical implementation.

E - Step In the E-Step of iteration t, the posterior $P(z_i|y_i,\mathbf{\Theta}^t,\sigma^t)$ distributions are evaluated using the current estimation $\mathbf{\Theta}^t$, σ^t and the corresponding predicted local deformations of the mesh. As defined in equation (4.7), these functions require to find for each target vertex y_i and patch k the vertex index v_i^k of its nearest neighbor in the different predicted configurations of the patch.

The complete E-Step amounts to the computation of a $m \times (N_p+1)$ matrix whose lines add up to 1, as shown in figure 4.5. This is an very parallel operation as all the elements of this matrix can be evaluated independently, except for the normalization of each line that takes place afterwards. In theory it would be tempting to use space partitioning techniques to speed up the nearest neighbor search. However the dependency of this search on the orientation of vertex normals makes this cumbersome. In practice we run a brute-force search, and show in section 5.4 CPU/GPU timings that indicate that the computation time remains reasonable for practical uses.

M - Step The M-Step of iteration t requires to minimize the bounding function $B^t(\mathbf{\Theta},\sigma)$ defined as in equation (4.18) by the the soft data - model assignment weights that were computed in the E-Step. In this bounding function, both data terms and rigidity terms are made of weighted squared distances between 3D points. This fits exactly in the mesh deformation framework defined in chapter 3 and equation (3.16). We do not solve minimize $B^t(\mathbf{\Theta},\sigma)$ directly with respect to both $\mathbf{\Theta}$ and σ but instead follow the Expectation Conditional Maximization (ECM) approach (Meng and Rubin [23]) that shares the convergence properties of EM while being easier to implement. The idea is to replace the M-Step by a number of CM-steps in which variables are optimized alone while the others remain fixed. Thus in the M-step, we first use the mesh deformation framework and obtain $\mathbf{\Theta}^{t+1}$, then

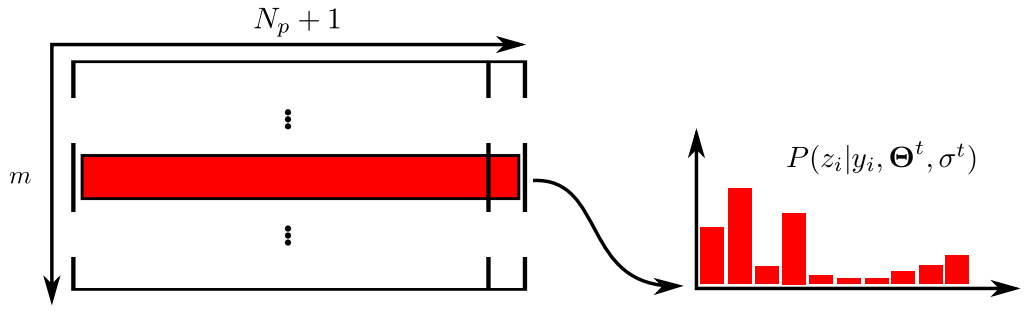


Figure 4.5: The soft assignment matrix holds the posterior patch-assignment distributions for every vertex of the target point cloud. As such, the lines are normalized to add up to 1. The last column of the matrix corresponds to the outlier class.

we update σ^{t+1}. The update of σ is simply done by setting the derivative $\frac{\partial}{\partial\sigma}B^t(\mathbf{\Theta}^{t+1}, \sigma)$ to zero.

$$\mathbf{\Theta}^{t+1} = \underset{\mathbf{\Theta}}{\operatorname{argmin}}\, B^t(\mathbf{\Theta}, \sigma^t) \tag{4.19}$$

$$\sigma^{t+1} = \frac{1}{2}\sqrt{\frac{\sum_{k=1}^{N_p+1}\sum_{i=1}^{m} w_i^t(k)\|\mathbf{y_i} - \mathbf{x}^t(v_i^k)\|^2}{3\sum_{k=1}^{N_p}\sum_{i=1}^{m} w_i^t(k)}} \tag{4.20}$$

To avoid degenerate mesh configurations, we however do not completely minimize the bounding function. Instead we just run one iteration of the Gauss-Newton algorithm, which amounts to minimizing the quadratic approximation of the objective function around (Θ^t, σ^t).

4 Results

In this section we present the results of our algorithm on several multi-view datasets. We first show our results on sequences involving several objects to emphasize the benefit of our approach with respect to methods that use very constraining deformation models. Then we show with a quantitative evaluation that our work produces comparable results to existing methods when dealing with less complex sequences.

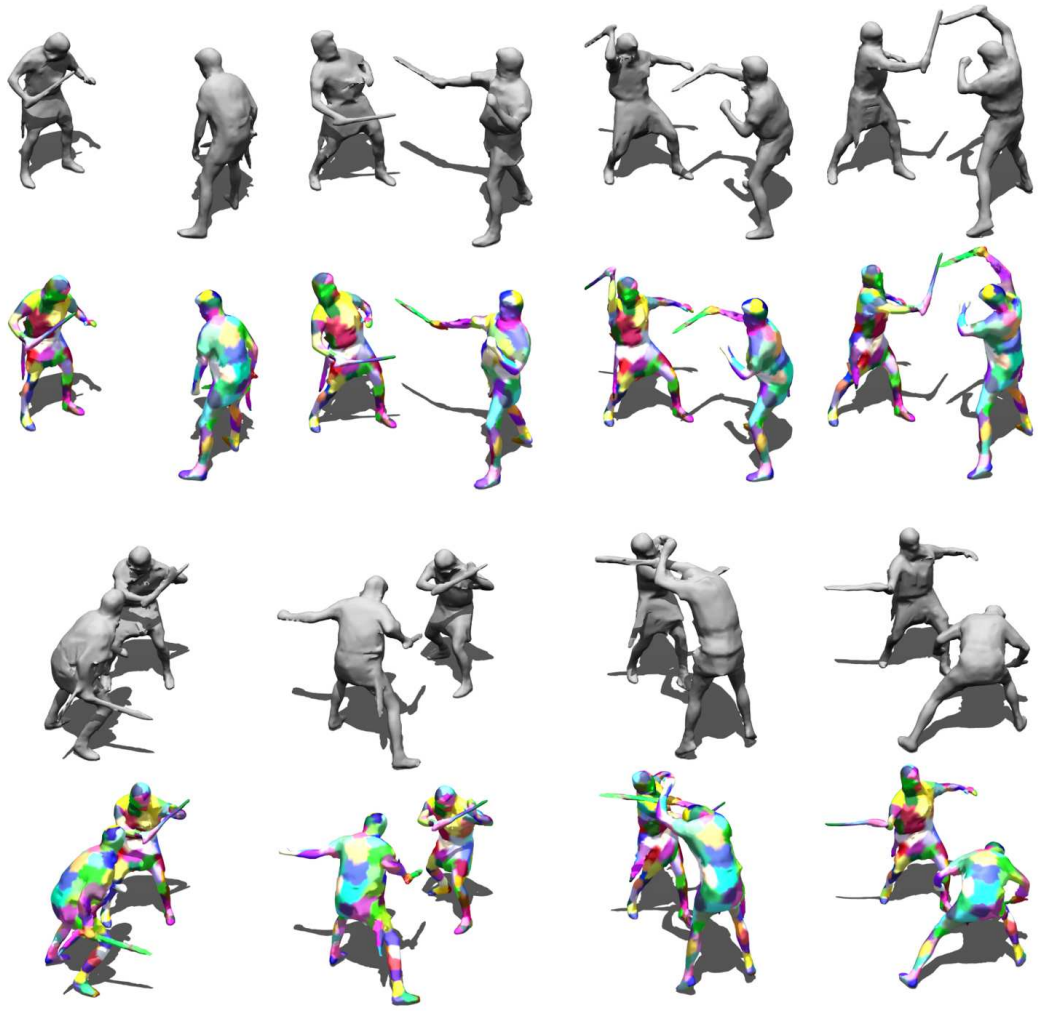


Figure 4.6: Results on the Romans sequence that exhibits two interacting objects occluding each other in fast paced action. The reference mesh is on the top left. The temporally inconsistent input geometry is displayed in gray. The corresponding colored surfaces are the deformed reference mesh.

4.1 Multi object tracking

The Romans sequence (1050 frames - about 42 sec) in figure 4.8 was kindly provided by Indigenes Productions and 4D View Solutions. It involves two soldiers fighting with swords. The action is fast paced and there are many intersections and occlusions of geometry. This type of sequence highlights the advantage of our approach. No initialization was required, as the reference mesh was simply the first reconstruction of the sequence. Even with our simple rigidity model that treats the swords exactly as the loose clothing of the actors, the algorithm recovers meaningful deformations throughout the sequence.

The Ball sequence shown in figure 4.7 was made available by INRIA Rhône-Alpes

and involves a father and his son playing with a ball. Here again, our approach does not require the tedious definition and fitting of an articulated model for each of the objects. A topologically suitable reconstruction where the three objects are distinct is simply used as reference geometry. This sequence illustrates the handling of some outlying geometry that was created by the erroneous segmentation of shadows as foreground. The outlier class introduced in the Bayesian framework allows to limit the impact of such geometry in the inference of deformation by progressively reducing the weight of points that can't be explained by the model.

The Basketball sequence (1364 frames - about 55 sec) in figure 4.6 was recorded in our own multi-camera studio. This scene is interesting for a number of reasons. Firstly, the ball bounces between the legs and is sometimes held close to the body for many frames. As such the data involves two distinct objects whose interaction is fast, complex and contains a lot of occlusion. Secondly, the simple shape from silhouette method that was used to recover the temporally inconsistent meshes was quite coarse. It produced geometry exhibiting occlusions and numerous artifacts such as missing limbs. Finally, the reference mesh that was deformed across the sequence was simply the first mesh, which is barely more than a blob. The results presented in figure 4.8 show that our algorithm can recover meaningful estimates of these difficult motions and deformations using a coarse model of the surface, even when confronted with numerous artifacts in the input data such as missing limbs, occlusions and self intersecting geometry.

4.2 Evaluation of the silhouette reprojection error

We also ran our algorithm on standard datasets available to the community to compare it to previous works. We used as input the results of a precise 3D reconstruction algorithm in one case, and rudimentary shape from silhouette in the other. As we show in this section, our algorithm performs consistently well in both these situations. In the presented results we additionally optimized the silhouette reprojection error in a post-processing step, with the method presented in subsection 3.3. This procedure relies on the very same numerical framework defined in chapter 3, uses extremely small patches and minimizes an energy that is the residual error in silhouette overlap.

Tracking Using Photo-consistent Meshes As Input The Surfcap Data from University of Surrey consists of a series of temporally inconsistent meshes obtained by the photo-consistency driven graph-cut method of Starck et al.[30]. Except for some rare reconstruction artifacts in a couple of frames, these are overall very clean and smooth meshes. Because of their extremely high resolution, these meshes were down-sampled to roughly 10k vertices and fed to our algorithm. We present our results on six sequences. They show a hip-hop dancer whose moves are very challenging to track because they contain fast motions and large deformations. In figure 4.9, our results on the *Flashkick* dataset show that we can cope with extremely fast deformations such as a backflip. In figure 4.10 we present our results on the *Pop* sequence in which the intricate and ambiguous motion of crossing arms is handled properly. Additionally figure 4.13 shows a quantitative evaluation of the

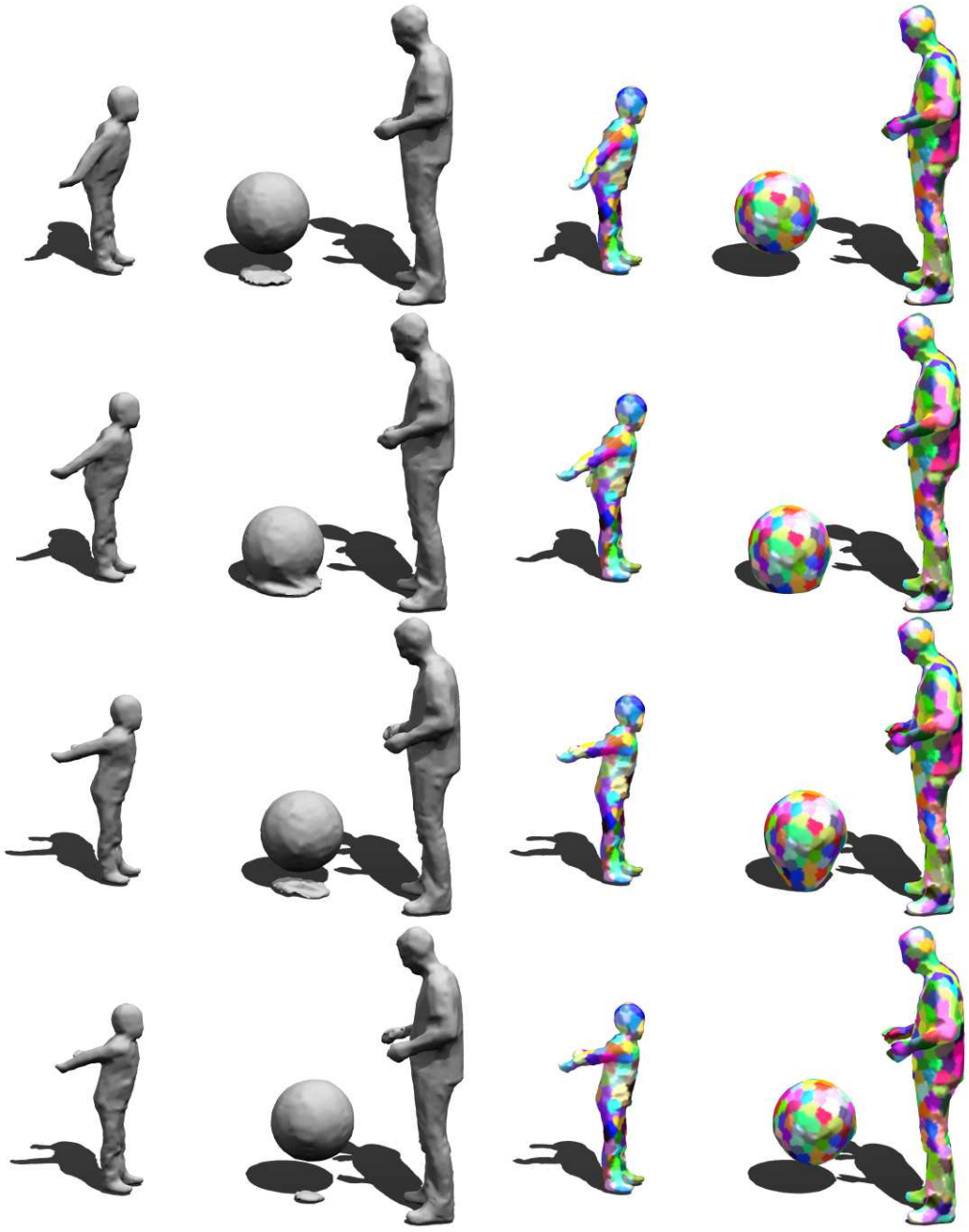


Figure 4.7: The frames 292-295-296-297-298 of the Ball sequence show the effects of outlying geometry and the benefit of the outlier class in the Bayesian model. The shadows were erroneously segmented as foreground, which resulted in outlying geometry. It is observable that as the ball goes down, this outlying geometry is correctly handled by the EM framework and does not impact the estimation of the ball's deformation. As the ball bounces, the algorithm tries to find a compromise between rigidity and data while progressively reducing the weight of the erroneous points. It quickly converges to the proper estimate.

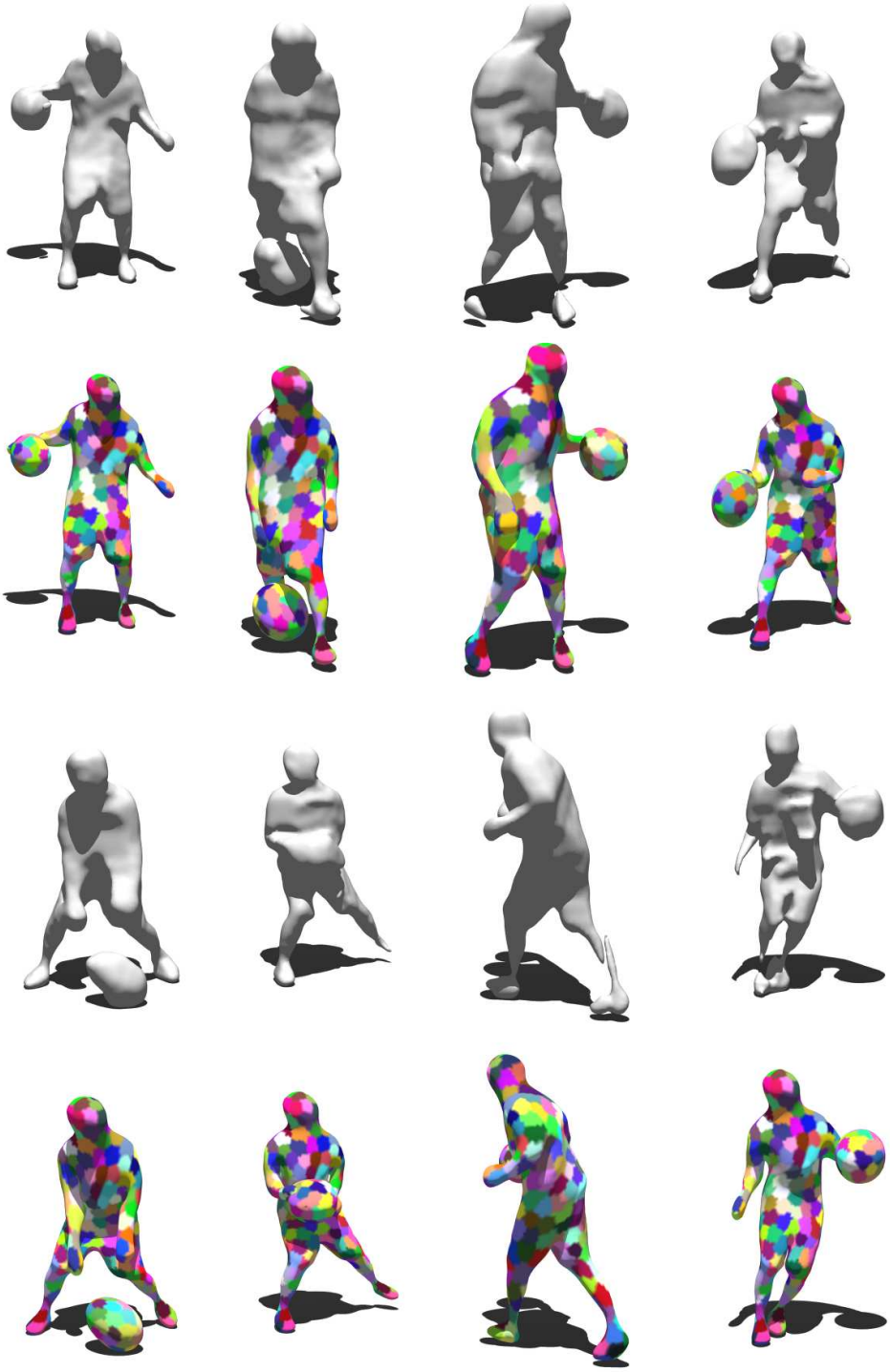


Figure 4.8: Results on the Basketball Sequence. The reference mesh is displayed on the top left. The hand and ball were manually separated for this initial mesh in a modeling software so that the deformation model would be topologically suitable. Note that despite a very coarse reference surface, wrong geometry, missing data and fast motion have a limited impact on our tracking algorithm.

Figure 4.9: Results on the Flashkick sequence. The kickflip itself consists of extremely fast motion as it spans over 15 frames.

Figure 4.10: Results on the Pop sequence. Note how geometrically ambiguous the arm crossing is, and the strong self-occlusions it produces.

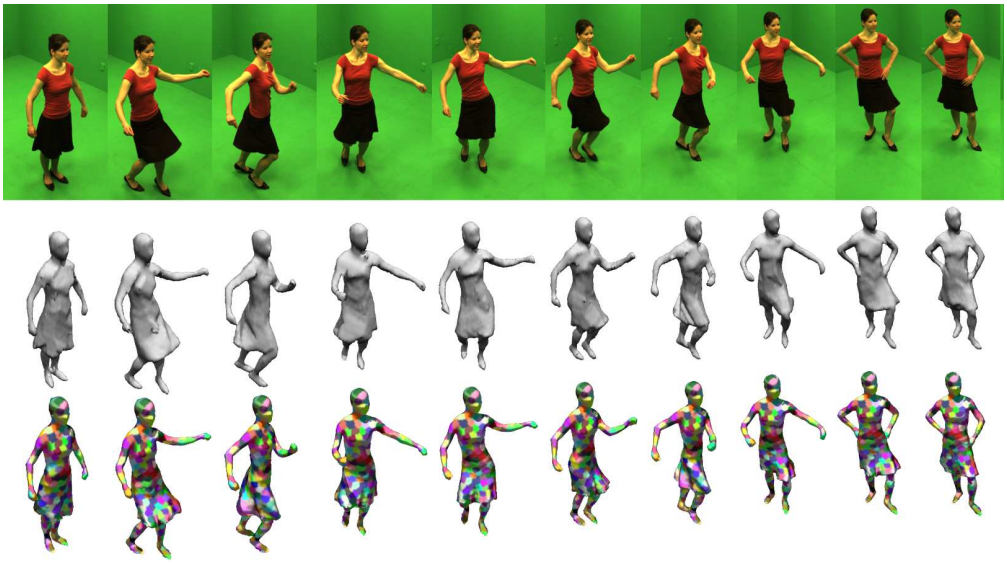


Figure 4.11: Results on the Samba sequence. The approach yields visually convincing results on the tracking of a skirt.

overlap error between the reprojected silhouettes from our result and the original silhouettes. The error is given as the ratio of erroneous pixels and total number of pixels in the original silhouette and stays approximately constant at a value of 5%.

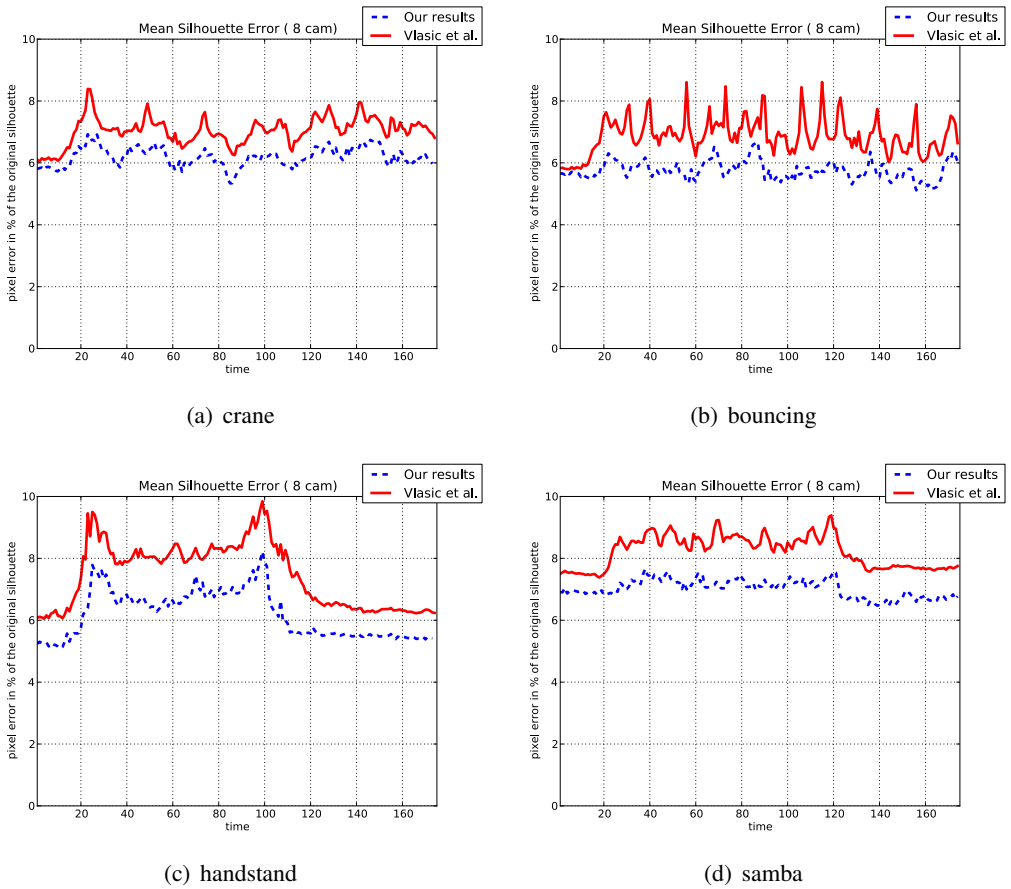
Tracking Using The Visual Hull As Input We used the multi-view image data made public by the MIT CSAIL group to run a very simple volumetric shape from silhouette algorithm. The resulting visual hulls, although only a coarse approximation of the true shape, were enough to drive the deformation of the provided template mesh through the sequences. We show our results after silhouette fitting on the *Samba* dataset. In this specific sequence, a woman in a skirt dances. Skirts are difficult to handle for methods deforming a reference mesh as the interpolated surface between the bottom of the skirt and the legs has to undergo severe compression and stretching. We show in figure 4.11 that our approach still manages to produce visually convincing results. We ran our algorithm on four of the available sequences and compared the silhouette re-projection error after silhouette fitting to the meshes obtained by Vlasic et al. [32]. The results in figure 4.12 show that our approach yields a similar precision despite its much weaker underlying deformation model. Furthermore, our results were obtained without manual intervention, while these of [32] required help from the user in ambiguous frames.

5 Discussion

In this section, we present a variety of experiments designed to determine more precisely the impact of the different components and parameters of the algorithm. We start by looking at the probabilistic model [5] that was presented in this chapter and compare it to the simpler non-rigid ICP that we presented in earlier work [3]. Both approaches use the same patch-based regularization. We show that the probabilistic model, complemented with the patch prediction mechanism, gives better convergence rate and overall tracking accuracy. In another part of this section we recall the parameters that need to be set to run the algorithm and discuss their influence. Finally we present timing results that confirm the computational tractability of the presented method and that open perspectives for turning it into an actual tool that could be used for production purposes.

5.1 Probabilistic and deterministic assignments compared

The reason why we experimented with the Bayesian model in the first place was because it offered a principled way to handle of outliers. It is indeed one of its major benefits, and the Ball sequence in figure 4.7 illustrated how this allows to increase robustness with respect to parasite geometry in the scene. We however found that this approach also gave



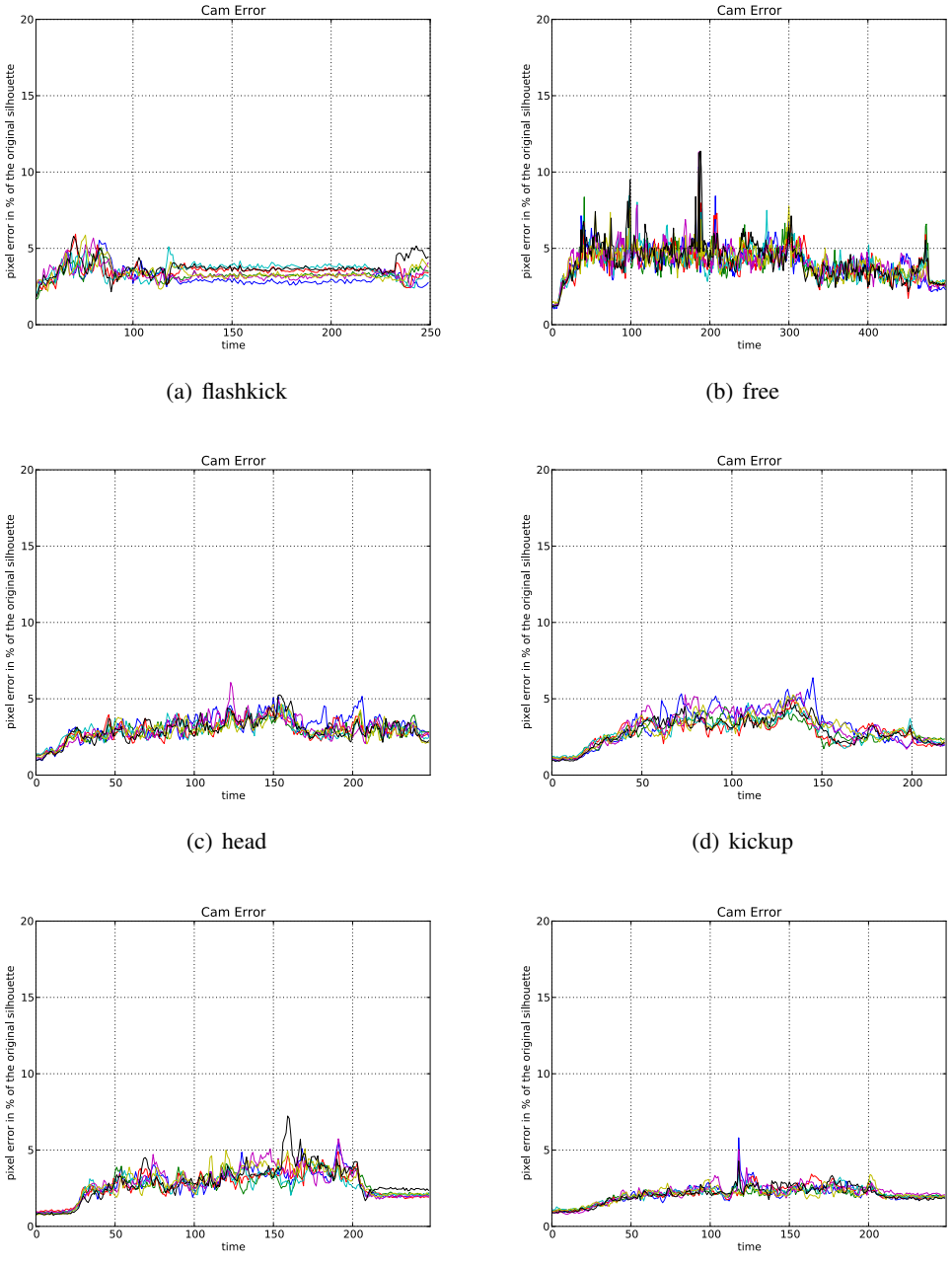
(a) crane (b) bouncing

(c) handstand (d) samba

Figure 4.12: Numerical comparison of our silhouette reprojection error to that of Vlasic et al. [32]. This graph shows a consistent good behavior of our approach despite the much weaker underlying deformation model. The slight improvement in performance we get is most likely due to the temporal smoothing they apply.

better results on sequences that were relatively free of parasite geometry. On the MIT sequences (figure 4.14) as well as on the U. of Surrey sequences (figures 4.15, 4.16, 4.17, 4.18, 4.19).

We ran the registration in four configurations. In these configurations, the Bayesian model with the smooth assignments was either used or replaced by a simpler nearest neighbor search. Similarly, the prediction mechanism (described in figure 4.3) could be turned off. In all cases, the optimizer was allowed to run for at most 30 iterations of E-Step (or one of its variations) and M-Step. We ran the silhouette optimization on every frame independently as a post-process. The goal of this experiment was to identify cases where the inference would get stuck in a local minimum. The idea was that in extreme cases of bad registration, the silhouette optimization would also fail, while it would improve greatly the numerical results for successful registrations. This accentuates the differences between



(a) flashkick (b) free

(c) head (d) kickup

(e) lock (f) pop

Figure 4.13: Silhouette reprojection error of our deformed model in percentage of the original silhouette area. Each color represents a camera.

incorrectly and correctly registered frames in our results.

An number of remarks can be made on the already mentioned figures, and the results for the roman sequence in figure 4.20:

- the Bayesian modeling combined with neighbor prediction performs better than the other configurations in the vast majority of cases.
- most configurations actually give reasonable results except for small time periods where they lose track. However they tend to recover quickly, which indicates that the shape prior maintained by our deformation model is quite strong.
- the worst combination seems by far to be to combine smooth assignments with no neighborhood prediction.

5.2 Importance of neighboring patch prediction

To investigate the reasons behind the previous results, we focused on the initial frames of the flashkick sequence (see figure 4.18) in which it can be observed that the probabilistic/prediction and deterministic/prediction perform much better than the configurations without patch prediction. These are interesting frames because they involve fast motion, and exhibit no changes of topology.

In figure 4.21, we look at the evolution of the average distance between the two surfaces during the iterated E and M-Steps. In each plot, we used the first mesh of the sequence as reference mesh for the rigidity and started the inference from the same patch configuration for all four curves.

The presented results show that the prediction mechanism greatly improves the rate of convergence. It also appears that without this prediction, the probabilistic registration has a sharp decrease in its rate of convergence after 3 or 4 iterations. Our interpretation is that most of the surface is registered properly after these iterations, and that this causes a sharp decrease in the evaluation of σ. Then the slack introduced by the outlier class in the model starts to dominate in badly registered part of the surface, which without the help of the prediction mechanism severely hurts the convergence rate.

5.3 Influence of parameters

Now that we have evaluated the impact of the components in our algorithm, we move to discussing the influence of parameters. In its full version, with smooth assignments and neighbor prediction, the algorithm requires the following parameters to be defined:

- the maximum patch size.
- the balancing between the data term $\ln P(\mathcal{Y}|\mathbf{\Theta})$ and the rigidity regularization $E_r(\mathbf{\Theta})$.
- an initial value for the variance σ.
- the expected outlier proportion $e_{outlier}$.

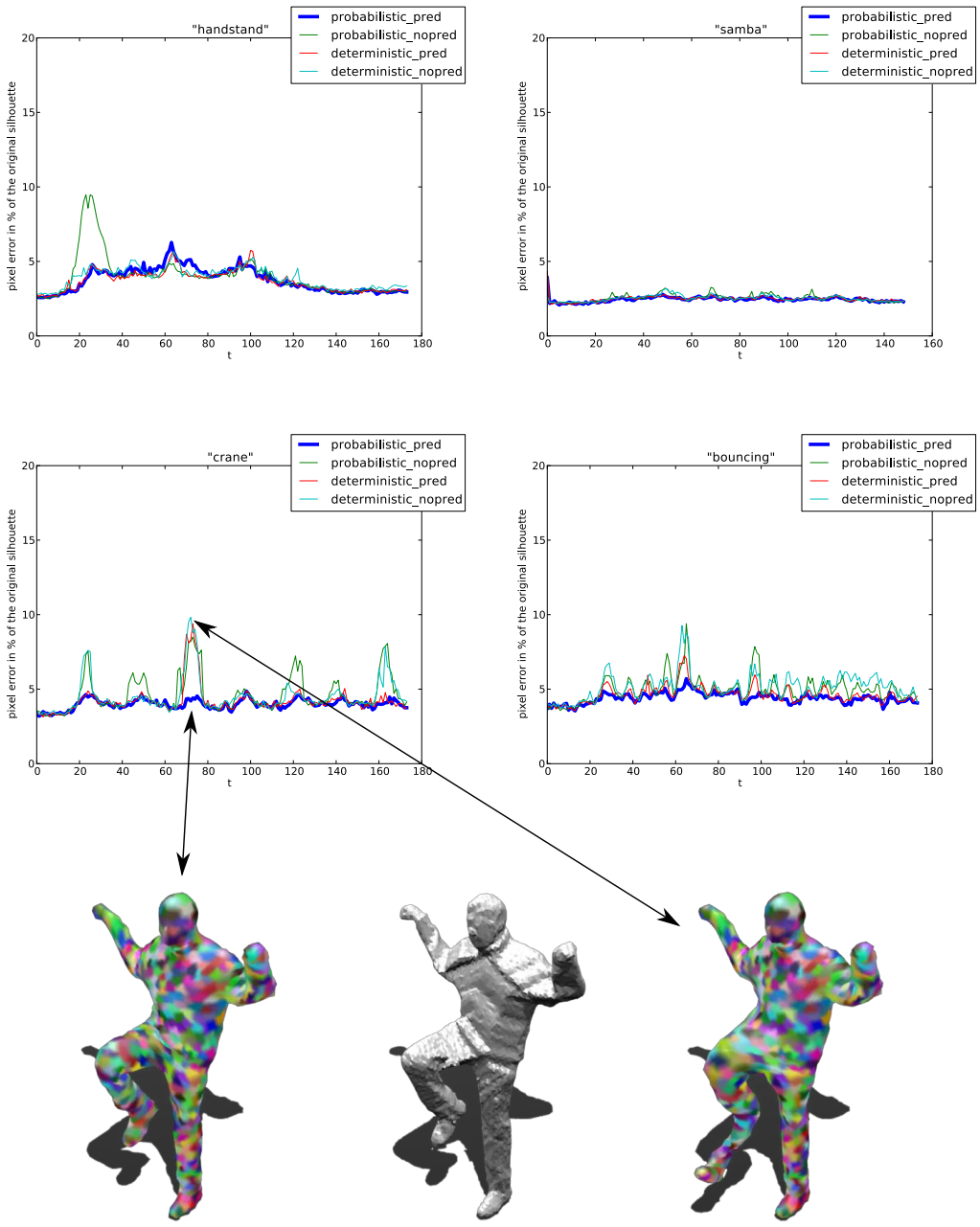


Figure 4.14: Probabilistic/deterministic comparison: the U. of Surrey sequences. These results are obtained after an additional silhouette fitting step. Residual errors indicate that a limb was not fitted properly in the registration step and that the silhouette optimization stayed stuck in a local minimum. On the bottom row we compare the result with probabilistic assignments and prediction, the target geometry, and the result with deterministic assignments and no prediction.

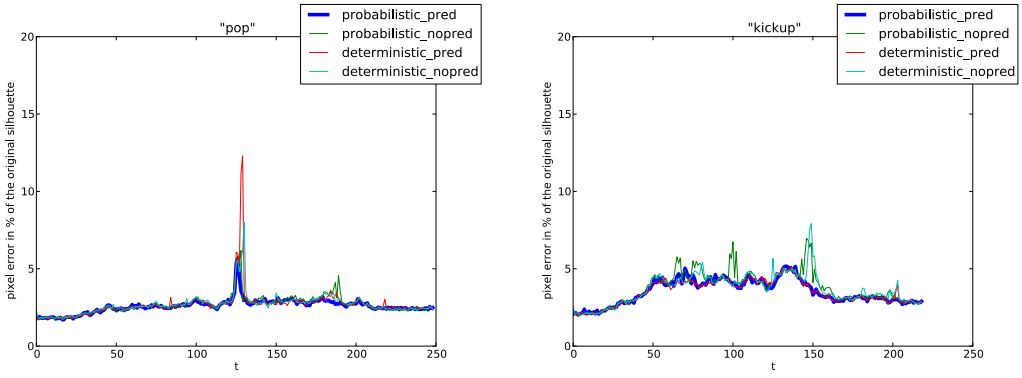


Figure 4.15: The pop and kickup sequences behaved well in all configurations.

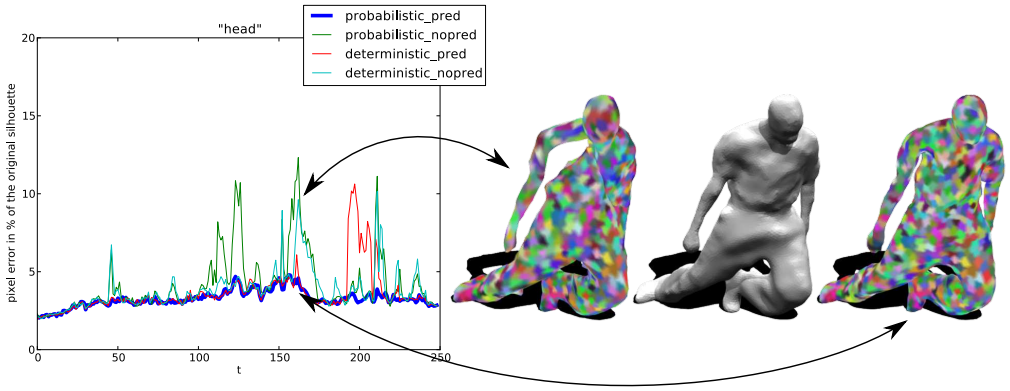


Figure 4.16: In the head sequence the registration converged to a state from which the silhouette fitting could not get out. This is most likely due to a local surface flip. Such flips combined with the normal compatibility condition in the nearest neighbor search are difficult to recover from when there is no neighboring patch prediction.

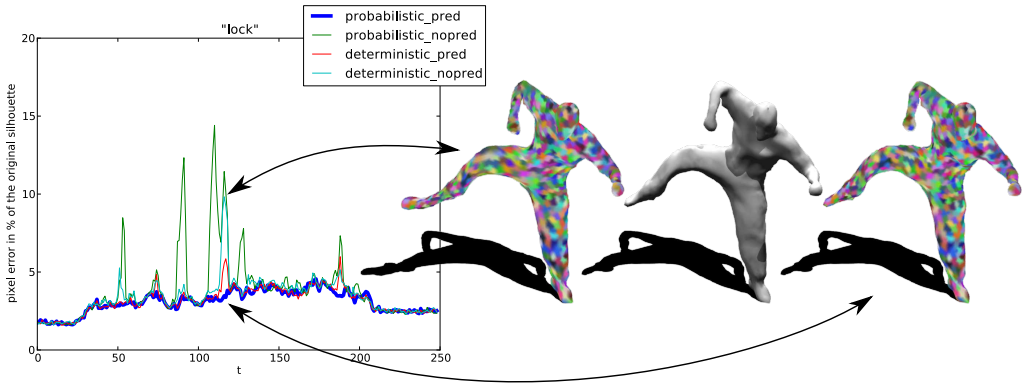


Figure 4.17: In the lock sequence the leg is stretched, probably because the strong deterministic assignments did not allow the surface to "slide" back into place with the rigidity force.

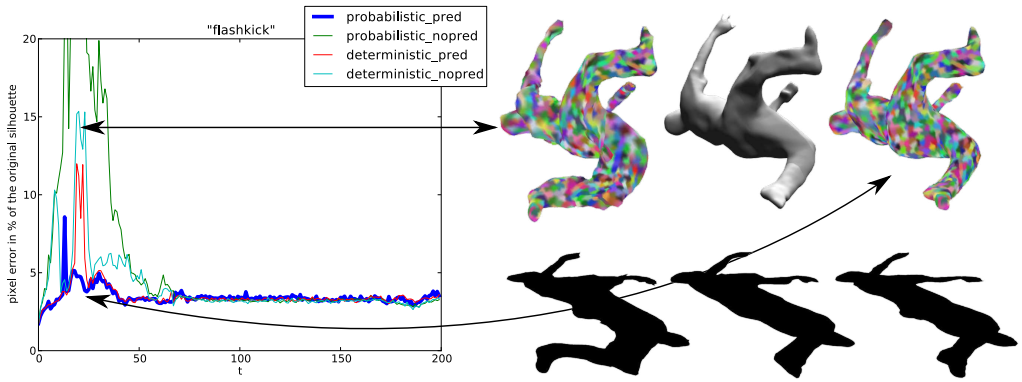


Figure 4.18: In the flashkick sequence, similarly to the lock sequence, it appears that with deterministic assignments and without prediction, the surface did not manage to slide back into place during the fast motion of the flashkick, and that the inference led to a local minimum involving stretching the thigh.

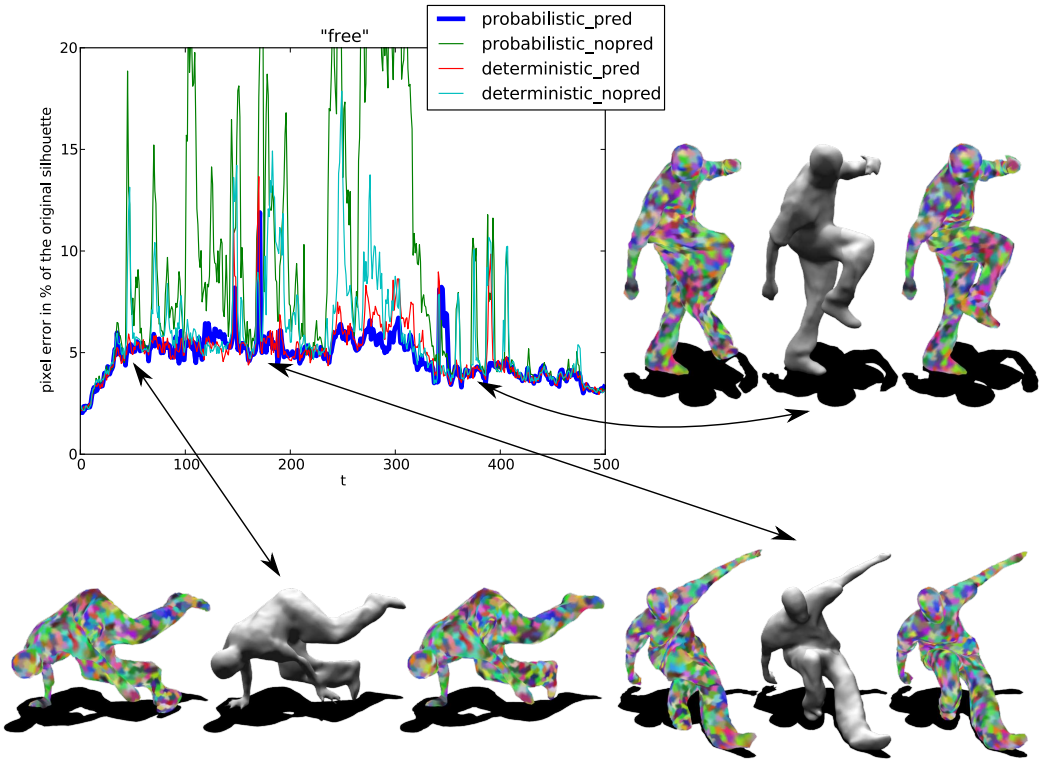


Figure 4.19: Probabilistic/deterministic comparison: the free sequence. The artifacts at frames 45 and 391 are similar to the ones observed in the lock and flashkick sequences. At frame 188 however, the problem is that the legs are switched. It appears that without soft assignments, the inference converges prematurely to a local minimum.

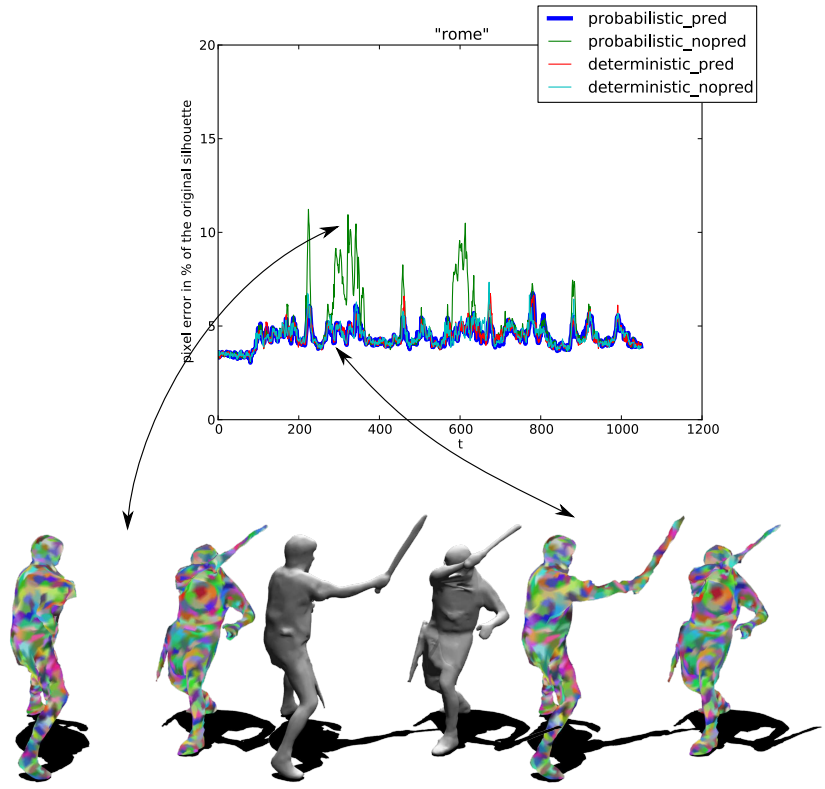
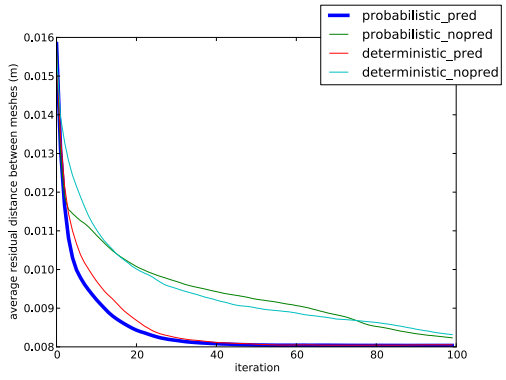
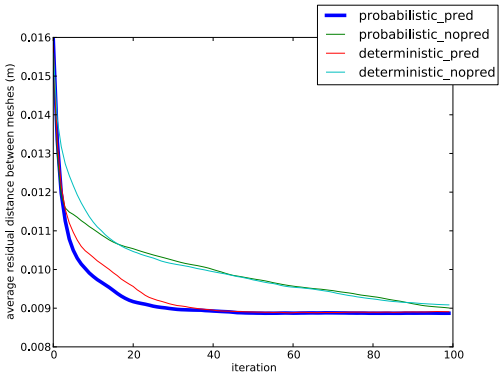


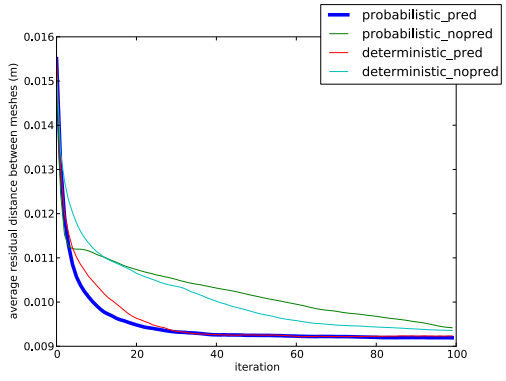
Figure 4.20: The roman sequence, as shown in figure 4.6 involves two actors with props, but is in fact not very ambiguous.The soldiers have there arms stretched out most of the time and very little self intersection happens. Therefore, most methods perform correctly. In this sequence like in most others, the worst performance is that of the probabilistic/no prediction configuration. On the bottom row we show how the arm of the left soldier remained stuck in the body. Our interpretation is that with the probabilistic modeling, the outlier class offers some slack to the optimizer that can choose to ignore the full right arm of the soldier. Without the prediction mechanism to propagate the information from correctly registered parts of the surface to their neighborhoods, this is a dangerous mechanism.



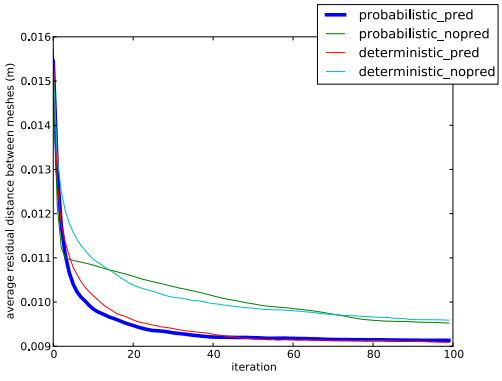
(a) flashkick $50 \mapsto 51$



(b) flashkick $51 \mapsto 52$



(c) flashkick $52 \mapsto 53$



(d) flashkick $53 \mapsto 54$

Figure 4.21: The first 5 frames of the flashkick sequence involve fast motion and are therefore interesting to evaluate the convergence behavior of the registration. Furthermore, as they do not involve changes of topology, we can use the average residual 3D distance between the two meshes as error measure. For each of these plots, the 4 configurations start off with the same deformation of the reference mesh and try to register it observed mesh of the next frame. The first observation that can be made on these 4 frames is that the prediction mechanism greatly improves convergence speed. The second observation is that without prediction, the probabilistic registration has a sharp decrease in its convergence rate after 3 or 4 iterations.

Our experience indicates that the algorithm is actually rather resilient to variations of these values. Since all of the sequences used similar mesh resolutions, the size of patches was common to all sequences an yielded 150-200 patches per human. The balancing between data and regularization term was empirically fixed on one of the sequences so that the residual energies would have comparable magnitudes. All the other sequences ran with the same value. The starting variance was always set to 2 times the mean edge length.

In figure 4.22 we show the result of an experiment we ran to study the influence of the expected outlier proportion. In this experiment, seven pairs of consecutive meshes (M_t, M_{t+1}) from the Free sequence were considered. Outlier points were distributed around M_{t+1} by duplicating a percentage of its vertices and perturbing them with a Gaussian noise of standard deviation 4 edge length. Then we ran the deformable registration of M_t to M_{t+1} with different outlier proportions. The figure shows the average residual registration error as a function of $e_{outlier}$ and the actual proportion of added outliers. We conclude from this experiment that this parameter does not require to be finely tuned and that it simply needs to be non zero to give the optimization enough slack to progressively ignore outliers and converge to a proper solution.

5.4 Computational cost

We give in table 4.1 experimental timings on numerous sequences that give an idea of the complexity of the method. These measurements were obtained by looking at times when files were written to the hard-drive. As such they are only an indication on the computational load of our method and do not constitute a precise performance evaluation. The computational cost is largely dominated by the nearest neighbor search and the sparse linear system solver. The nearest neighbor search is straightforward to parallelize on the GPU. It is to be expected that coming up with a smarter space partitioning approach or using more computational resources should make this step negligible. The remaining bottleneck is therefore the sparse linear solver. Preliminary experiments on the CPU indicate that Conjugate Gradient is a viable alternative to the direct solver we currently use. Porting this approach to the GPU and evaluating its interest definitely constitutes future work, on the way to making our algorithm usable for interactive purposes.

5.5 The i.i.d. assumption and limitations.

We conclude the discussion of our approach by presenting its main theoretical limitation and the practical consequences that ensue. The i.i.d. assumption that led to equation (4.12) is to be considered with care in that the observation process is a multi-camera setup in which parts of the surface, thus patches occlude each other. This clearly biases the drawing of samples in the distribution of 3D data. For example in figure 4.8, when the arms and body are joined, the local density of points in the input data does not double, which clearly indicates that the data generation by two overlapping patches on the arm and the body is not independent. In that sense our method and equation (4.12) are only approximations.

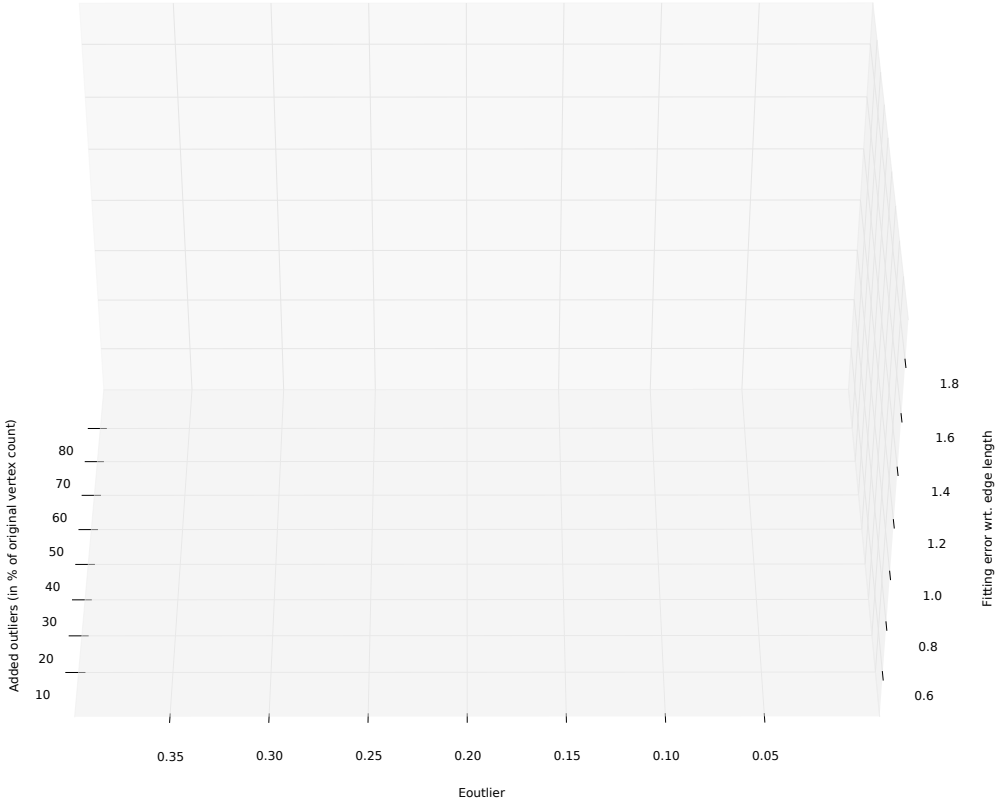


Figure 4.22: Average fitting error on 7 random frames of the Free sequence. The x axis represents the expected outlier parameter. The x axis represents the actual proportion of added outliers to the target point cloud. The z axis shows the fitting error with respect to the mean edge length.

The first and obvious practical consequence to this assumption of independence is that our EM framework can not be easily be used to track an object observed from a single range scanner. Occlusion in this case becomes much too severe for the hypothesis of independence to hold. Consider for example a human facing the range scanner and pointing his arm towards the camera. In the observed point cloud, there are only couple of points corresponding to the fist and no points at all for the arm. This makes the optimization impossible within our framework.

The second practical consequence and main artifact that can be produced by our method we did not encounter in the sequences from the community. On a sequence that we had to process for an applicative paper [21] however, we had an actor that kept both his arms along his body most of the time. The reconstruction was of very low quality and the severe self-occlusion caused by the arms merged the arm and torso reconstructions. We observed that the arms would pop-out and would sometime briefly return to the rest pose for a couple of frames. Our interpretation is that because the local density of points did not double, each point of the merged arm-torso reconstruction in the data would be explained by both the

Table 4.1: Average timings on standard sequences for the EM procedure (max. 10 EM steps - without silhouette refinement) with target point clouds of roughly ten thousand vertices. The CPU implementation was run on a 2.5Ghz quad-core machine. The CUDA implementation was run on a NVIDIA Geforce GTX260.

Sequence	Length	Ref. Mesh Vertex#	Average Time Per Frame	
			CPU	GPU
Flashkick	200	5445	24 s	3.60 s
Free	500	4284	25 s	3.16 s
Head	250	5548	29 s	3.79 s
Kickup	220	5580	23 s	3.69 s
Lock	250	5301	24 s	3.52 s
Pop	250	5596	16 s	3.44 s
Handstand	174	5939	29 s	4.11 s
Bouncing	174	3848	29 s	3.70 s
Crane	174	3407	11 s	2.72 s
Samba	150	5530	12 s	2.03 s

arm and the torso. Because of the normalization of posterior probability density functions and thus of weights in EM, this meant that the energy function would only pull the model towards the data with half the force. The regularization term would then pull the arm further away from the torso at each iteration, which would in turn diminish the weight of the association between the data points and the arm.

This behavior could be expected because our approach is designed to look for a solution that explain the data with as little deformation as possible. Our method maximizes the explanation of *occupied* space but does not model at all *empty space*. Interestingly, this problem was also considered by Revow et al. [27] in an extension of their work on handwritten digit recognition [18] (figure 4.2). In [27], the problem is described as "A significant drawback of our generative model is that it does not treat the un-inked pixels as evidence." They address the issue by suggesting to introduce a penalty for all the pixels that are not inked in the data but inked by the deformed model. The results of their attempt appear inconclusive: They account for white space violation to evaluate the final fits but do not use it in the inference of the deformation itself. Indeed, penalizing the occupation of unoccupied space in the final fit with such a method also induces penalizing going through that same unoccupied space during the inference. In our case one could understand it as having attractive vertices on the observed surface and repulsive vertices everywhere else in the scene's bounding box. These repulsive vertices would make it difficult to go from a configuration where the arm is against the torso to a configuration where the arm is away from the torso for example. For small values of σ, they would create impassable energy ridges in parameter space that would prevent the arm in the deformable model from going through the newly formed empty space between the observed arm and torso. We leave as future work the study of this problem and the evaluation of this particular solution.

6 Conclusion

In this chapter, we have proposed an approach to recover temporally consistent animated models from sequences of surfaces acquired in multi-camera setups. In contrast to previous works, this approach makes no assumption on the articulated nature of the tracked object and actually allows to process scenes involving several arbitrary objects.

Our developments have integrated the rigidity model of chapter 3 as a deformation prior in a Bayesian formulation of the mesh registration problem. Although the idea of embedding deformable registration in an EM framework had already been explored several times during the past 20 years, we have made two contributions that have allowed to perform significantly more challenging deformable mesh tracking tasks.

- First, we have used the patch-based parametrization that was presented in chapter 3. Instead of computing intractable point-point soft assignments, we have computed point-patch soft assignments. Our results show that this was sufficient to handle parasite geometry gracefully. They also show that this approaches allowed to retain some smoothness in the convergence compared to the deterministic assignments of ICP. Finally, our timing results prove that this approach runs fast enough to be usable on commodity hardware.

- Our second contribution is the patch-prediction mechanism. This mechanism maintains in the inference several hypothesis for the location and orientation of every patch. For each patch, the considered hypothesis are obtained by assuming that the patch and one of its neighbors moved together rigidly. In effect, this gives the inference an opportunity to quickly propagate the information from correctly registered parts of the surface to their neighbors. We have shown that this approach significantly improves the convergence rate of the mesh registration.

Our experimental validation has been run on several scenes of different nature. Some of them included loose clothing, skirts, swords or bouncing balls. The 3D data that we have used as input was of variable quality. Some meshes were obtained with state of the art photometric reconstruction methods, while others simply consisted of the visual hull and exhibited significant reconstruction artifacts. For all of these scenes, we have used the same deformation prior, the same mathematical formulation and the same set of parameters. There was no user intervention before the inference to indicate rigid parts or joint locations. There was no user intervention during the inference to correct the estimated pose. Our algorithm has managed on all of these sequences to perform comparatively well with respect to previous works that use more priors on the deformation or user intervention to correct errors. The major contribution however is that our algorithm has allowed to process scenes more general and complex than these that had been addressed by prior art.

References

[1] Besl, P. J. and N. D. McKay
1992. A method for registration of 3-d shapes. *IEEE PAMI*.

[2] Cagniart, C., E. Boyer, and S. Ilic
2009. Iterative Mesh Deformation for Dense Surface Tracking. In *3DIM*.

[3] Cagniart, C., E. Boyer, and S. Ilic
2010. Free-from mesh tracking: a patch-based approach. In *CVPR*.

[4] Cagniart, C., E. Boyer, and S. Ilic
2010. Iterative Deformable Surface Tracking in Multi-View Setups. In *3DPVT*.

[5] Cagniart, C., E. Boyer, and S. Ilic
2010. Probabilistic deformable surface tracking from multiple videos. In *ECCV*.

[6] Chen, Y. and G. Medioni
1991. Object modeling by registration of multiple range images. In *ICRA*.

[7] Chetverikov, D., D. Stepanov, and P. Krsek
2005. Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm. *Image and Vision Computing*.

[8] Choi, J. and A. Szymczak
2009. Fitting solid meshes to animated surfaces using linear elasticity. *ACM Trans. Graph.*

[9] Chui, H. and A. Rangarajan
2003. A new point matching algorithm for non-rigid registration. *CVIU*.

[10] Corazza, S., L. Mündermann, E. Gambaretto, G. Ferrigno, and T. P. Andriacchi
2010. Markerless motion capture through visual hull, articulated icp and subject specific model generation. *IJCV*.

[11] de Aguiar, E., C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun
2008. Performance capture from sparse multi-view video. In *SIGGRAPH*.

[12] de Aguiar, E., C. Theobalt, C. Stoll, and H.-P. Seidel
2007. Marker-less deformable mesh tracking for human shape and motion capture. In *CVPR*.

[13] Dempster, A. P., N. M. Laird, and D. B. Rubin
1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society, series B*.

[14] Fitzgibbon, A. W.
2001. Robust registration of 2d and 3d point sets. *BMVC*.

[15] Frank Dellaert
2002. The Expectation Maximization Algorithm. Technical report, Georgia Tech.

[16] Gall, J., C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel
2009. Motion capture using joint skeleton tracking and surface estimation. In *CVPR*.

[17] Granger, S. and X. Pennec
2002. Multi-scale em-icp: A fast and robust approach for surface registration. In *ECCV*.

[18] Hinton, G. E., C. K. I. Williams, and M. Revow
1992. Adaptive elastic models for hand-printed character recognition. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*.

[19] Horaud, R. P., F. Forbes, M. Yguel, G. Dewaele, and J. Zhang
2011. Rigid and articulated point registration with expectation conditional maximization. *PAMI*.

[20] Horn, B. K. P.
1987. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*.

[21] Ladikos, A., C. Cagniart, R. Gothbi, M. Reiser, and N. Navab
2010. Estimating radiation exposure in interventional environments. In *MICCAI*.

[22] Li, H., R. W. Sumner, and M. Pauly
2008. Global correspondence optimization for non-rigid registration of depth scans. *Comput. Graph. Forum*.

[23] Meng, X.-L. and D. B. Rubin
1993. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*.

[24] Myronenko, A. and X. Song
2010. Point-set registration: Coherent point drift. *PAMI*.

[25] Pottmann, H., Q.-X. Huang, Y.-L. Yang, and S.-M. Hu
2006. Geometry and convergence analysis of algorithms for registration of 3d shapes. *IJCV*.

[26] Rangarajan, A., H. Chui, and F. L. Bookstein
1997. The softassign procrustes matching algorithm. In *IPMI*.

[27] Revow, M., C. K. I. Williams, and G. E. Hinton
1996. Using generative models for handwritten digit recognition. *PAMI*.

[28] Rusinkiewicz, S. and M. Levoy
2001. Efficient variants of the icp algorithm. In *3DIM*.

[29] Shinya, M.
2004. Unifying measured point sequences of deforming objects. In *3DPVT*.

[30] Starck, J. and A. Hilton
2007. Surface capture for performance based animation. *CGA*.

[31] Starck, J., A. Hilton, and J. Illingworth
2002. Reconstruction of animated models from images using constrained deformable surfaces. In *DGCI*.

[32] Vlasic, D., I. Baran, W. Matusik, and J. Popović
2008. Articulated mesh animation from multi-view silhouettes. In *SIGGRAPH*.

CHAPTER 5

Articulated Models for Mesh Registration

The developments presented in the previous chapters were focused on the core contribution of this thesis, that is on a generic surface tracking method that does not make assumption on the nature of the tracked object. In this chapter we study the applicability of the developments from the previous chapter to the tracking of articulated objects. This allows us to evaluate to which extent considering prior knowledge on the way the surface deforms helps in the inference for motion.

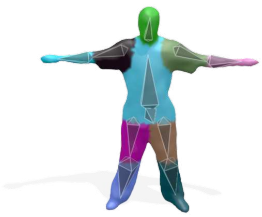
The goal of this chapter is to explore the use of different rigidity models with the Bayesian mesh registration framework that was developed in chapter 4. In other words, we want to introduce more knowledge on how the surface deforms in the inference for motion. We explore the case of articulated objects, and look into two distinct ways to account for the prior we have on the way they deform.

- One possibility is to use an *intrinsically regularized* deformation model that enforces articulated motion through its parametrization. We therefore adapt the mesh registration framework so that the motion would be parametrized by kinematic chains. It is expected that the reduced number of degrees of freedom in the optimization should make the registration process faster.
- The other possibility is to extend the deformation model presented in chapter 3 so that it will account for *clusters* of patches that tend to deform rigidly. There the parametrization is the one that was used in chapters 3 and 4, but a new *extrinsic regularizer* is introduced to quadratically penalize non rigid motion within the clusters.

This study can at first appear to go against the general motivation of this thesis, that is the development of a generic tracking method. It is however necessary for several reasons: firstly, one can reasonably imagine interfaces where artists could guide the inference for motion by manually feeding information on the rigidity or the topology of the tracked objects. Secondly, we need to evaluate the benefits and drawbacks of more constrained deformation models.

(a) GM: The patch-based rigidity of chapter 3.

(b) GMa: The patch-based rigidity of chapter 3 with an additional energy term enforcing the rigidity of clusters.



(c) KB: Parametrization of the deformation with a kinematic tree.

Figure 5.1: There are three deformation models compared in this chapter: the generic patch-based model of the previous chapter, an extrinsically regularized model that additionally enforces rigidity in clusters of patches, and an intrinsically regularized articulated model that parametrizes motion as joint angles in a kinematic tree.

1 Related Works on Articulated Tracking

Marker-less motion capture has received considerable amounts of attention over the past two decades, and several surveys have been dedicated to the subject [17, 28, 21, 25]. Our goal here is not to overview the whole field, but to give some background on key ideas that are directly related to the problems that we address in this chapter, namely tracking by geometric registration and the comparison of articulated models with our more generic deformation models.

1.1 Parameterizing articulated motion.

The first option to parametrize articulated motion is to represent the articulated object with a kinematic tree made out of joints and bones, and to parametrize its pose with angle values at the joints. This has the evident advantage of intrinsically constraining the range of motions to these that preserve the length of bones. Furthermore the joint angles constitute a relatively low dimensional set of parameters that fully describe object's motion. Kinematic chains and trees are well studied structures because of their crucial importance for robotic manipulation, and the interested reader can turn to the book by Murray et al. [23] for a meticulous study of the subject. Two interesting mathematical tools presented in this book are the product of exponential maps and twist motion. These tools were introduced for articulated tracking in the work of Bregler and Malik [4], Bregler et al. [5] who were tracking human poses in monocular videos. As it was discussed in chapter 3, exponential maps are indeed the parametrization of choice when descending the gradient of energies that are functions of rotations. In the next section, we will recall as simply as possible the mathematics required in our context, and go over the specificities of different joint types and kinematic chains. This line of work corresponds to the first rigidity model that we examine: the *intrinsically regularized* deformation model.

The second option when parameterizing articulated motion is to use *soft-constraints* in place of intrinsically constrained parameter spaces. This type of parametrization is for example very used for human detection in monocular images. Pictorial structures [13] are graphical models that divide the object in parts, where each part correspond to local visual characteristics that can be searched for by feature detectors. The parts are linked by soft spring-like constraints that model the relative locations or orientations of the two parts. Typically the parameters of the model for the relation between two parts (or conditional probabilities) are learned from data, and these models can be as simple as as: "the head tends to be 20cm above the neck." The recent book chapter by Ramanan [26] provides an interesting overview on these models. However pictorial structures are originally 2D entities. In the work by Sigal et al. [27], they are transposed to 3D and called loose-limbed models, where each limb is parametrized by a full rigid transformation. That idea was also employed by Corazza et al. [11] and in following work by Mundermann et al. [22]. This line of ideas is closer to the second rigidity model that we examine: the *extrinsically regularized* deformation model.

1.2 Modeling the skin.

The previous paragraphs have presented possible definitions for the skeleton. For model-based tracking, we now need to define the shape, or flesh, deformed by this skeleton. In other words, we need to go from the variation of joint angles to the deformation of actual surfaces in 3D.

A common and simple solution is to use geometric primitives to model the limbs: examples include ellipsoids (Bregler et al. [5], Horaud et al. [20]), cylinders (Sigal et al. [27]) or even tapered super-quadrics (Gavrila and Davis [18]). Implicit surfaces were also used by Plänkers and Fua [24] who attach metaballs to the skeleton. Each of these metaballs generates an ellipsoidal field functions in 3D, and the surface of each limb is obtained as a level-set of the added field functions of the metaballs corresponding to that limb. This formulation allows the author to parametrize the shape with the coefficients of the ellipsoids. Because the goal of this particular paper was to optimize both for shape and motion of the tracked person, this low dimensional parametrization of the shape was required.

Subject specific models are however usually recovered before tracking. The work by Corazza et al. [11] builds a model by using a method based on the SCAPE model [1] and iteratively optimizes for shape and pose of the subject. Another type of body shape model is used in the work by Bandouch et al. [2], where the anthropometric coefficients of the RAMSIS [6] model (originally created in the fields of ergonomics) are tuned by hand before tracking. Model generation is not the focus of our study. We therefore choose to use a coarse yet simpler way to generate a subject specific model: we attach a reference mesh of the tracked person to a skeleton. This could be done automatically with a method such as the one by Baran et al. [3]. We will present in the next section how we use the semi-automatic method of a 3D content production software to perform the rigging of the reference mesh.

1.3 Pose optimization

The last part of our review is focused on how articulated body models can be used in an optimization for the pose of the tracked object. Because we focus on generic geometric registration in 3D, we leave aside from our review a number of ideas. For example we do not detect endpoints such as heads, hands or feet to constrain the inference and neither use motion priors nor activity recognition. Our interest goes exclusively to geometric registration and to the different available optimization schemes. In all model-based approaches, an objective function measures the fitness of any considered pose by evaluating how well this pose explains the observed data. In the case of human motion capture, three things make the optimization of such a function challenging: first, the space of human poses is high dimensional. Second, occlusion and self similarities of objects make visual data ambiguous, and the fitness functions are therefore multi-modal with many local minima. Finally, evaluating the fitness of a pose is quite costly when it requires to synthesize visual data to be compared with the observation.

Stochastic search Stochastic sampling and filtering approaches are designed to handle the multi-modality of fitness measures. This is especially important in the monocular case. Consider for example an actor observed from the side. It is really difficult to differentiate on appearance alone whether this actor holds his left arm, or his right arm, or both arms horizontally in front of him. Stochastic search methods evolve populations of discrete samples in the parameter space and try to explore this space to escape local extrema, but also to increase the sampling density in areas where the fitness measure has peaks to get results with acceptable precision. This is challenging because of the high dimensionality of the parameter space and the restricted number of points that can be sampled in reasonable time.

If the process underlying the observed data is modeled as Markovian, stochastic filtering methods allow to more efficiently explore the parameter space by accounting for the dynamic nature of the process with a simple transition model between subsequent poses. However this transition model may not yield a proposal density of samples that matches the real posterior probability. To further refine the inference, the annealed particle filter of Deutscher and Reid [12] performs several iterations to evolve the particles while gradually reducing a temperature term that controls the smoothing of the fitness function. This increases the sampling in narrow peaks of the fitness function, and the initial high temperature tends to prevent premature convergence of the particle population to local maxima of the fitness function. The paper by Gall et al. [14] gives a more in depth analysis of the mathematical derivation and behavior of this method.

Another interesting idea in the work of Deutscher and Reid [12] is the adaptive diffusion that reduces the number of particles along dimensions where the algorithm is doing well to increase the sampling of the uncertainty along ambiguous dimensions. Furthermore, they introduce a crossover operator inspired by genetic algorithms. This operator generates new sample points from two parent samples by using the coordinates of one parent along some dimensions and the coordinates of the other along the rest. This is a very effective way of

seeding new samples in yet unexplored modes of the fitness function, and to encourage the survival rate of the subparts of particles that correspond to a partial good fit of the data. While [12] tries not to commit to a fixed partition of the parameter space, Gall et al. [15] propose two human specific mutation operators that swap kinematic branches such as the right and left leg.

Local optimization Local optimization methods are unimodal, and usually assume the local differentiability of the fitness function to perform some kind of gradient descent. Their advantage is their better precision and their convergence when these hypothesis on the fitness function are valid. Their drawback is that the inference can easily get stuck in a local maximum of the fitness function.

We have illustrated the multiple modes of an error function with the example of an actor seen from the side from a single point of view. When multiple point of views are available however, many situations can be disambiguated. For example, with an additional camera looking at the actor from the top, it is no longer ambiguous whether the right arm, left arm or both arms are held horizontally. One possible error function in a multi-view environment is the sum of silhouette reprojection errors over all cameras. However in this case each component of the error term contributed by a single image can be highly multi-modal. The hope is that the average of these functions will present prominent peaks where modes coincide, and therefore resolve the ambiguity. Previous works that use the sum of silhouette reprojection errors tend to show that this does not work very well and have to resort to hybrid methods mixing local optimization with stochastic search. For example the method by Carranza et al. [8] first fits the torso to the silhouettes using some numerical gradient descent (Powell's method) then solves for the limb's poses independently by performing a grid search on the 4 dimensions of the limb's parameter spaces. In a similar spirit, the work by Gall et al. [16] initially drives the inference with silhouette contour matches and SIFT matches. In a second step, stochastic searches are run on the subspaces corresponding to misaligned parts to optimize the silhouette overlap. Although this approach is not explicitly hierarchical as [8], it can be expected that the misaligned parts will be limbs and that the two approaches will have similar behavior.

Strictly local optimization methods seem however usable when the fitness function is expressed in 3D. This hypothesis is supported by our results in the previous chapter, as well as by a number of works that we will now review. The research by Mundermann et al. [22], Corazza et al. [11] is one of the closest to ours in terms of methodology and a good example of local optimization for articulated tracking. In these works, the problem is formulated as articulated ICP, and the optimization is performed with the Levenberg-Marquardt method. One issue with this approach however is that the correspondences computed at each iterations ICP are fully trusted.

Other methods perform some kind of articulated ICP, but are less trusting with correspondences. For example in the early work of Cheung et al. [10], the body is coarsely represented with 6 ellipses, and an EM-like procedure alternates voxel labeling and ellipse estimation where the ellipse parameters for a body part are only computed from the voxels that were assigned to this body part. This idea can also be found in the work of Caillette

and Howard [7] who additionally combine color and 3D distance in their computation of soft-assignments. Cheng and Trivedi [9] proposes a kinematically constrained Gaussian mixture model that uses soft priors on the motion to avoid hard angle limits and retain the convergence properties of EM. Finally the recent works (already mentioned in chapter 4) by Horaud et al. [20] and Horaud et al. [19] explored articulated registration using EM. The research presented in this chapter belongs to this line of work and actually naturally extends the results of chapter 4 by examining the effects of the introduction of rigidity priors in the inference for motion.

2 Pose Optimization for Kinematic Trees

In this section, we briefly recall how kinematic chains can be parametrized, and how analytical gradients of point positions can be computed with respect to the parameters of these chains. This allows to solve Inverse Kinematics problems, that is to optimize the configuration of the chain so that points attached to it will fulfill as well as possible a number of soft positional constraints.

However, in contrast with usual applications of Inverse Kinematics where a few positional constraints at most are used at a time, the mesh registration framework of chapter 4 that we adapt here can potentially involve millions. Therefore, the following paragraphs focus on the uniformity of the mathematical formulations and on numerical efficiency.

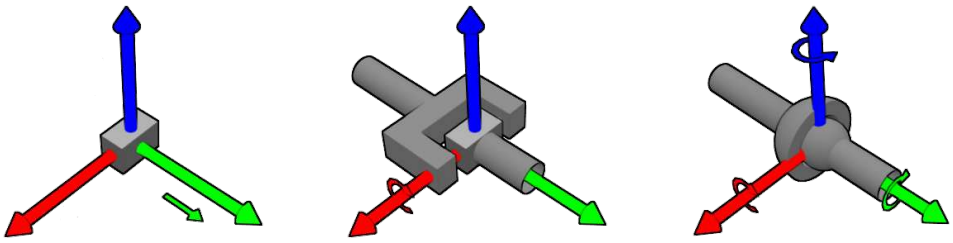
In particular, the following derivations will underline that the Jacobian of a residual positional error with respect to the update of a chain parameter can be factorized as the product of a matrix that is only dependent on the position of the 3D point and a vector that only depends on the state of the kinematic chain. This will lead us discuss how the Gauss-Newton algorithm can be efficiently implemented by processing positional constraints in batches instead of explicitly computing one Jacobian per constraint.

2.1 Joints in their local frame

In its local frame, a joint induces a rigid transformation that we write with a 4×4 matrix $\mathbf{T}$. This allows to describe the rigid transformation applied to a 3D point $\mathbf{x}^0$ this way:

$$\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \mathbf{T} \begin{bmatrix} \mathbf{x}^0 \\ 1 \end{bmatrix}$$

.



(a) Prismatic joint - 1 DOF. (b) Hinge joint - 1 DOF. (c) Ball joint - 3 DOF.

Figure 5.2: The three types of joint considered by our optimizer, along with the number of degrees of freedom (DOF) associated.

Prismatic joint A prismatic joint has 1 DOF. It is defined by an axis $\mathbf{w}$. A scalar parameter θ controls the magnitude of the translation.

$$\mathbf{T} = \begin{bmatrix} \mathbf{I} & \theta\mathbf{w} \\ 0 & 1 \end{bmatrix}$$

. This means that $\mathbf{x}(\theta) = \mathbf{x}^0 + \theta\mathbf{w}$, and therefore that the derivative of $\mathbf{x}$ with respect to θ is simply

$$\frac{d\mathbf{x}}{d\theta} = \mathbf{w}. \tag{5.1}$$

Hinge joint A hinge joint has 1 DOF. It is defined by a 3D point $\mathbf{c}$ and a unit-length axis $\mathbf{w}$. A scalar parameter θ controls the magnitude of the rotation. In its local frame, it induces the transformation:

$$\mathbf{T} = \begin{bmatrix} e^{[\theta\mathbf{w}]_\times} & 0 \\ 0 & 1 \end{bmatrix}$$

This means that $\mathbf{x}(\theta) = e^{[\theta\mathbf{w}]_\times}\mathbf{x}^0$. If we perform a Taylor expansion of the exponential, the derivative of $\mathbf{x}$ with respect to θ appears as:

$$\frac{d\mathbf{x}}{d\theta} = [\mathbf{w}]_\times\mathbf{x} \tag{5.2}$$

2.2 Joints in kinematic chains

We index the kinematic chain linking the root of tree with the joint to which the point is attached with indices $1, \ldots, n$. Each joint k along the chain induces a transform written as a 4×4 matrix $\mathbf{T}_k$. We define the result transformation induced by the whole chain as $\mathbb{T}_n = \mathbf{T}_1 \ldots \mathbf{T}_n$.

Let us consider a vertex v attached to the frame of joint n, and let $\mathbf{x}^n$ be its local coordinates in this frame. Because v is attached to this frame, this value is constant. The

world coordinates of the point $\mathbf{x}$ for a given value of the parameters is then:

$$\begin{aligned}\begin{bmatrix}\mathbf{x}\\1\end{bmatrix} &= \mathbf{T}_1 \dots \mathbf{T}_n \begin{bmatrix}\mathbf{x}^n\\1\end{bmatrix}\\ &= \mathbb{T}_n \begin{bmatrix}\mathbf{x}^n\\1\end{bmatrix}\end{aligned} \tag{5.3}$$

The state of the kinematic chain is described by a vector $\mathbf{\Theta} = \{\theta_1, \dots, \theta_n\}$, where θ_k holds the angle values for the hinge joints, or a translation value for the prismatic joints.

To compute derivatives, we fix all the chain parameters $\{\theta_1, \dots, \theta_n\}$, except the one associated to the joint k and parametrized by θ_k. We define $\mathbf{x}^k$ as the local coordinates of the vertex v in the frame of joint k. These local coordinates account for the transforms of all the joints down the chain from k, that is of the current value of $\{\theta_{k+1}, \dots, \theta_n\}$.

$$\begin{bmatrix}\mathbf{x}\\1\end{bmatrix} = \mathbb{T}_k \begin{bmatrix}\mathbf{x}^k\\1\end{bmatrix}. \tag{5.4}$$

We equivalently write equation (5.4) as $\mathbf{x} = \mathbb{R}_k\mathbf{x}^k + \mathbf{c}_k$, where $\mathbb{R}_k$ and $\mathbf{c}_k$ are respectively the rotational and the translational part of $\mathbb{T}_k$. Note that $\mathbf{c}_k$ holds the world coordinates of the origin of joint k's frame.

Prismatic joint We know from equation (5.1) that $\frac{\partial \mathbf{x}^k}{\partial \theta_k} = \mathbf{w}_k$. In the context of the kinematic chain, this translates to:

$$\frac{\partial \mathbf{x}}{\partial \theta_k} = \mathbb{R}_k\mathbf{w}_k \tag{5.5}$$

Intuitively, this simply rotates the axis of the joint with the rotation induced by the previous joints in the chain. For all the points attached to joints down the chain, a variation of θ_k will result in a translation along this axis.

Hinge Joint We know from equation (5.2) that $\frac{\partial \mathbf{x}^k}{\partial \theta} = [\mathbf{w}_k]_\times \mathbf{x}^k$. Knowing that $\mathbf{x} = \mathbb{R}_k\mathbf{x}^k + \mathbf{c}_k$, we deduce that $\frac{\partial \mathbf{x}}{\partial \theta} = \mathbb{R}_k[\mathbf{w}_k]_\times\mathbb{R}_k^T(\mathbf{x} - \mathbf{c}_k)$. Finally this leads to:

$$\begin{aligned}\frac{\partial \mathbf{x}}{\partial \theta_k} &= [\mathbb{R}_k\mathbf{w}_k]_\times(\mathbf{x} - \mathbf{c}_k)\\ &= \begin{bmatrix}-[\mathbf{x}]_\times & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbb{R}_k\mathbf{w}_k\\ -[\mathbb{R}_k\mathbf{w}_k]_\times\mathbf{c}_k\end{bmatrix}\end{aligned} \tag{5.6}$$

2.3 Inverse Kinematics

To reduce the amount of clutter in the notation, we assume that there is a single kinematic chain whose joints are indexed by $1 \dots n$ and that every point $\mathbf{x}_i$ in the following equations is transformed by all the joints of the chain. This means it is attached to the frame of joint n.

Energy The notations are similar as these in chapter 4. We write target coordinates as $\mathbf{y}_i$. Solving an inverse kinematics problem on a given chain then amounts to minimizing an energy of the form:

$$E_{IK} = \sum_i \|\mathbf{x}_i(\mathbf{\Theta}) - \mathbf{y}_i\|^2. \tag{5.7}$$

The optimization consists in minimizing E_{IK} with respect to the state of the kinematic chain $\mathbf{\Theta}$. Because the energy is a sum of squared functions whose gradients can be computed analytically, we perform this optimization using the Gauss-Newton algorithm.

Efficient computation The EM formalism developed in the previous chapter involves a very large number of residuals. For a typical scene with one human, an observed cloud of 10^4 vertices and a model with 100 patches, this formalism leads to an energy function involving the squared norm of 10^6 3D residuals. Computing a Jacobian matrix $\mathbf{J}$ of size $3.10^6 \times N_{joints}$ is impractically slow in these conditions. Even accumulating directly on $\mathbf{J}^T\mathbf{J}$ and $\mathbf{J}^T\mathbf{b}$ remains very involved for that much data.

This paragraph presents how this matrix and this vector can be computed efficiently for a large number of residuals. We first notice that the gradients of these residuals can be written as:

$$\frac{\partial \mathbf{x}_i - \mathbf{y}_i}{\partial \mathbf{\Theta}} = \frac{\partial \mathbf{x}_i}{\partial \mathbf{\Theta}}. \tag{5.8}$$

Thus entry $[k, l]$ of $\mathbf{J}^T\mathbf{J}$ is given by:

$$(\mathbf{J}^T\mathbf{J})[k, l] = \sum_i \frac{\partial \mathbf{x}_i}{\partial \theta_k}^T \frac{\partial \mathbf{x}_i}{\partial \theta_l} \tag{5.9}$$

We can rewrite equation (5.6) to have a similar form to that of equation (5.5). If for prismatic joints $\boldsymbol{\omega}_k = \begin{bmatrix} 0 \\ \mathbb{R}_k\mathbf{w}_k \end{bmatrix}$ and for hinge joints $\boldsymbol{\omega}_k = \begin{bmatrix} \mathbb{R}_k\mathbf{w}_k \\ -[\mathbb{R}_k\mathbf{w}_k]_\times \mathbf{c}_k \end{bmatrix}$, then both partial derivatives write as the multiplication of a 6×3 matrix that only depends on $\mathbf{x}$, and this 6×1 vector that is different for each joint but only depends on the state of the chain and the type of joint:

$$\frac{\partial \mathbf{x}_i}{\partial \theta_k} = \begin{bmatrix} -[\mathbf{x}_i]_\times & \mathbf{I} \end{bmatrix} \boldsymbol{\omega}_k \tag{5.10}$$

This means that the entries of $\mathbf{J}^T\mathbf{J}$ can be written as:

$$(\mathbf{J}^T\mathbf{J})[k, l] = \boldsymbol{\omega}_k^T \left(\sum_i \begin{bmatrix} -[\mathbf{x}_i]_\times[\mathbf{x}_i]_\times & [\mathbf{x}_i]_\times \\ -[\mathbf{x}_i]_\times & \mathbf{I} \end{bmatrix} \right) \boldsymbol{\omega}_l \tag{5.11}$$

The entries for $\mathbf{J}^T\mathbf{b}$ contain the same type of factorization:

$$(\mathbf{J}^T\mathbf{b})[k] = \boldsymbol{\omega}_k^T \left(\sum_i \begin{bmatrix} [\mathbf{x}_i]_\times \\ \mathbf{I} \end{bmatrix} (\mathbf{x}_i - \mathbf{y}_i) \right) \tag{5.12}$$

Equations 5.11 and 5.12 show that for a simple chain where all the points are attached to the last joint, all the entries of $\mathbf{J}^T\mathbf{J}$ and $\mathbf{J}^T\mathbf{b}$ can be computed from the same 6×6 matrix and a 6×1 vector, multiplied by the correct $\boldsymbol{\omega}_k$ and $\boldsymbol{\omega}_l$. Pre-computing this accumulator transforms the complexity of building $\mathbf{J}^T\mathbf{J}$ and $\mathbf{J}^T\mathbf{b}$ from $O(N_{joints} \times N_{residuals})$ to $O(N_{residuals})$.

In the case where points are attached to different joints, we can easily keep this improved complexity. The idea is to accumulate this 6×6 matrix and this 6×1 vector per joint, using only the constraints set on points $\mathbf{x}_i$ attached to that joint. Because the accumulators use the world coordinates of points, and because skeletons are kinematic trees, one pass from the end joints to the root allows to add each joint's accumulator onto its parent's. This simple method allows the $\sum_i$ that happens in each accumulator to account only for the points that depend on the associated joint.

Furthermore, one can notice that these 6×6 and 6×1 accumulators only contain elements from the matrices $\sum_i \begin{bmatrix}\mathbf{x}_i^T & 1\end{bmatrix}^T \begin{bmatrix}\mathbf{x}_i & 1\end{bmatrix}$ and $\sum_i \begin{bmatrix}\mathbf{x}_i^T & 1\end{bmatrix}^T \begin{bmatrix}[\mathbf{x}_i - \mathbf{y}_i] & 1\end{bmatrix}$. These covariance matrices can be very efficiently accumulated and propagated down the kinematic chain, allowing to further speedup the dominating $O(N_{residuals})$ part of the computation.

One possible drawback of this method is that the covariance matrices are accumulated with world coordinates, which can cause some precision issues in practice. We used world coordinates here to describe the idea as simply as possible. In a practical application one might want to use local coordinates for each joint's accumulator, and perform a transform of the covariance matrix as it gets added to the accumulator of the parent's joint.

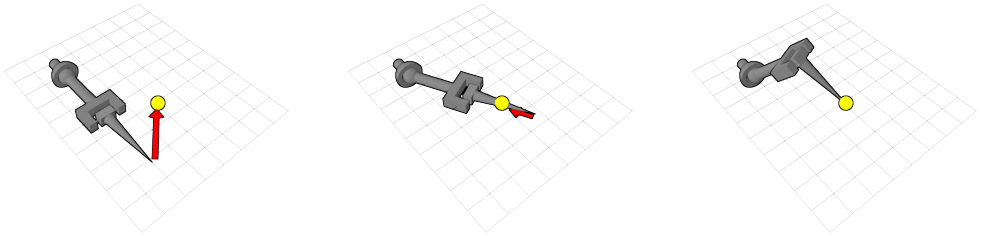
2.4 Numerical considerations

Regularization One of the drawbacks inherent to the use of the Gauss-Newton algorithm appears in the cases where the $\mathbf{J}^T\mathbf{J}$ matrix is singular. This usually happens when the system is under-constrained. For example, given a kinematic chain and an energy corresponding to one positional constraint on its root, the rest of the joints remain completely free. This means that the derivatives of the residual error with respect to their parameters are 0 and that the whole submatrix of $\mathbf{J}^T\mathbf{J}$ corresponding to these parameters is 0.

Confronted with such cases, a common method consists in lightly penalizing changes with respect to the current state of the kinematic chain, and to choose an update that optimizes the following function:

$$E_{damped}(\boldsymbol{\Theta} + \boldsymbol{\Delta\Theta}) = \sum_i \|\mathbf{x}_i(\boldsymbol{\Theta} + \boldsymbol{\Delta\Theta}) - \mathbf{y}_i\|^2 + \lambda\|\boldsymbol{\Delta\Theta}\|^2 \tag{5.13}$$

This has the effect of adding $\lambda\mathbf{I}$ to the $\mathbf{J}^T\mathbf{J}$ matrix and thus to make it positive definite. This method is called *damped* least-squares and its convergence behavior varies from that of Gauss-Newton ($\lambda = 0$) to that of a simple gradient descent for large values of λ. However, and as already discussed at page 75, the dimensional inhomogeneity between translations and rotations requires to use different damping coefficient λ_{rotate} and $\lambda_{translate}$ if we want



(a) Initial (b) Saddle point (c) A solution

Figure 5.3: Illustration of a singular case. The target for the tip of the kinematic chain is displayed in yellow. In both (a) and (b) the derivative of the residual with respect to the angle of the revolute joint is orthogonal to the residual error vector (in red).

the optimizer to behave similarly for scaled versions of the same problem (this issue is also briefly mentioned by Corazza et al. [11]).

This issue brings also to the more general issue of the problem's ill conditioning caused by the hierarchical parameterization of the articulated structure. In the extreme case where the human skeleton would have joints for finger, a variation of angle on the root has a significantly bigger impact than a variation of the same angle on the finger joint in terms of 3D motion of the fingertip. In other words the parameter space's Euclidean norm has no physical meaning. It is interesting to note that the patch-based structure presented in the previous chapters does not suffer from these issues, as it is local in nature and does not involve a hierarchy.

Perturbation Another issue with gradient descent or Gauss-Newton methods occurs at local extrema of the energy function. Figure 5.3 displays such a case, where the kinematic chain and the target position all lie in a 2D plane. The problem here is that the Jacobian of the position of the tip of the arm with respect to the angle of the hinge joint is *orthogonal* to the residual error vector. This means that the corresponding entry in $\mathbf{J}^T\mathbf{b}$ will remain 0 as long as the arm stays in the plane. The optimizer can get stuck at this saddle point (b). A perturbation is needed to kick the optimizer off this local optimum so that a solution (c) can be reached. In practice, and when dealing with real, noisy data in a geometric registration problem, we have not encountered this problem.

3 Two Models for Articulated Bodies

In this section, we present the two models of articulated bodies that we compare in this chapter. We begin by describing the skeleton-based articulated model that builds on the previous section. We justify our choice of ball joints to represent all the rotational joints with an example of human motion. We also present how the skeleton is attached to a 3D mesh. In the second part of this section, we briefly describe the second model, which builds on the patch-structure presented in chapter 3 but additionally maintains the rigidity of clusters of patches that correspond to the rigid parts of the articulated body.

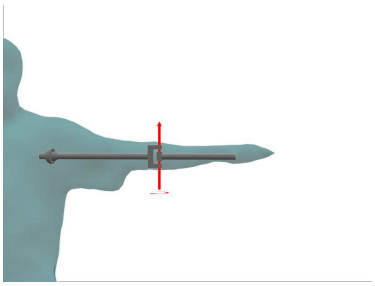
3.1 Skeleton-based articulated model

As shown in figure 5.4, our parametrization of the human body is rather coarse and uses only 11 joints. At the root we place three prismatic joint that controls the hips location. All the other joints are ball joints. This means that the model has 33 degrees of freedom. We did not present the mathematical details of ball joints in the previous section to keep the presentation concise. They nonetheless fit in the presented framework easily and allow to avoid the problem of gimbal lock caused by the Euler-angle parametrization of 3 DOF rotational joints.

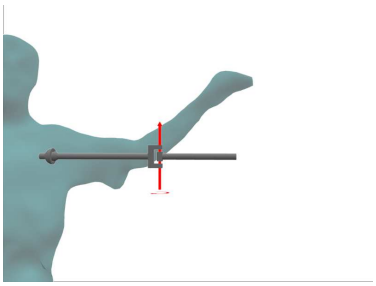
Our choice of ball joints for all rotational joints can seem surprising at first, since articulations like the knee look like they would be advantageously represented by hinge joints. The obvious benefit would be the reduction of the number of degrees of freedom in the optimization. In figure 5.5 we show a simple kinematic chain that models a human arm using a ball joint at the shoulder and a hinge joint at the elbow. In a mesh registration application, we might try to go from a straight horizontal arm (a) to a pose where the arm is horizontal and the forearm points up (b). In the figure the hinge is misaligned and only allows to move the forearm horizontally, so that the arm needs to rotate the arm around its axis first. However, all the residual point upwards, and are orthogonal to their derivatives with respect to the hinge angle. Their is no torque created either on the hinge or on the ball to rotate the arm around its axis. The optimizer might find a compromise where the arm stays straight but points diagonally, or maybe will eventually rotate the hinge a little which will start creating torque on the ball. The bottom line is that the convergence will be slow at best.

In the end, using hinges significantly changes the shape of the parameter space. This space is already challenging to work with because it is largely made of rotations and is therefore a curved manifold. The hinge behavior depends greatly on the parent joint orientation. This hurts the locality of the inference and introduces dependencies in the parameter space. Going back to the example of figure 5.5, if the arm is registered correctly, we would like to be able to optimize the forearm without having to modify the registration of the arm. We therefore choose to simplify the mesh registration by using ball joints, and delay the inference for the elbow orientation to a post process.

Figure 5.4: Sample kinematic tree with 11 joints to parametrize human motion. The root joint is a translate joint, and all the rotation joints are ball joints, which results in a total of 33 degrees of freedom for the kinematic chain.



(a) Reference mesh and reference skeletal pose.



(b) Deformed target mesh

Figure 5.5: This example displays a hinge at the elbow that is not aligned with the actual articulation. When the target geometry moves, the residual vectors are orthogonal to their derivative with respect to the joint angle.

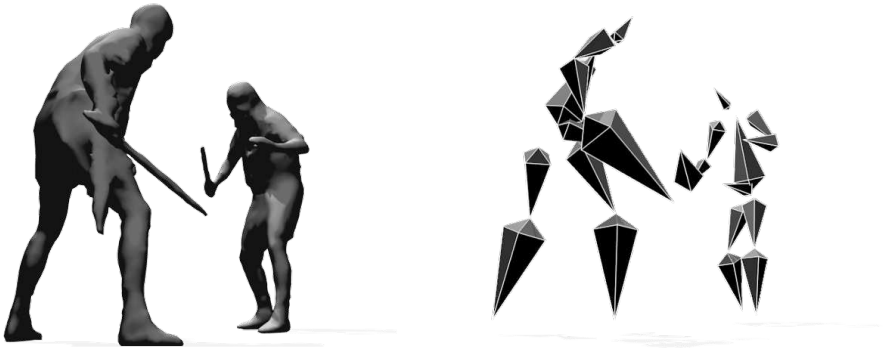
Mesh skinning The skin attachment step consists in attaching a mesh to the skeletal structure. This type of procedure is a well studied problem in computer graphics as it is allows to synthesize animated mesh sequences from motion capture data. Therefore, there is a wide variety of ready-to-use software packages to perform the task. We chose to use the open-source Blender 3D content creation suite. This software provides an implementation of the skinning method described in the 2007 SIGGRAPH paper of Baran et al. [3]. This method turns bones into heat sources, and attaches every vertex to its closest bone with a thermal conductivity proportional to the inverse squared distance to the bone. Then it solves for the equilibrium of the heat diffusion over the mesh using the surface Laplace-Beltrami operator. Figure 5.6 displays the articulated structure and the blending weights on the reference frame of the Roman sequence.

Patch prediction and soft assignments. The skeleton-based articulated model does not associate a patch structure to the reference mesh. We however transpose the ideas of neighboring patch prediction and patch-vertex assignments because the results presented in the previous chapter showed that they improved the behavior of the registration process. To transpose these ideas to the skeleton-based model, we overlay a patch structure on the reference mesh and associate each patch to one bone. So now each vertex is associated to a patch that is associated to a bone. Because bones move rigidly, the effects of the prediction are only felt at the boundaries of bones, that is when two neighboring patches depend on distinct bones.

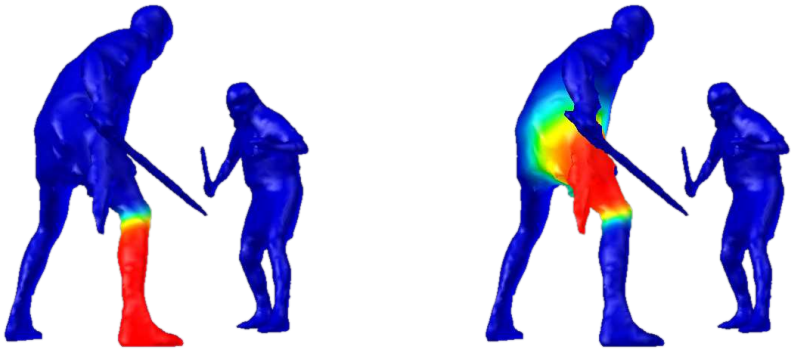
3.2 Patch structure with rigid clusters

The vertex blending weights obtained in the mesh skinning process, and shown in figure 5.6 can be used without the skeletal model. Our second articulated deformation model uses these weights to assign each patch to a cluster that corresponds to one of the body parts. For example figure 5.1(b) highlights all the patches that were assigned to the left knee. The idea is then to encourage all the patches that belong to the same cluster to move rigidly together. To do so, we simply extend the approach of chapter 3 but modify the notion of neighborhood: instead of having each patch be linked with its neighbors only, we link it with all the patches of the same cluster. The rigidity energy that we use is the one displayed page 73 in figure 3.7.

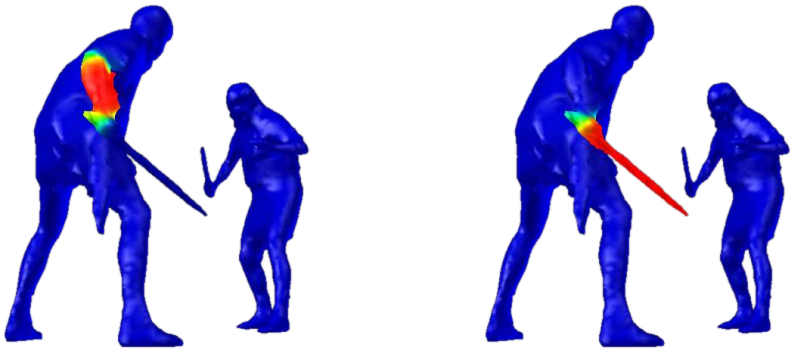
The major advantage of this approach is that it was simple to implement and integrate in the registration framework that was developed through this thesis. The only significant impact it has on the code is that it modifies the topology of the graph of patches and therefore the sparseness of the large linear system that is solved at each Gauss-Newton iteration. This matrix is still sparse but now has large strongly connected clusters of variables that form dense blocks. The link between these blocks is only ensured by the regular patch-based rigidity E_r that was used throughout the thesis. However, if the weight of the cluster rigidity energy is high enough, the blocks become numerically independent and the registration behaves as if there were 11 independent rigid bodies. To maintain coherence between the body parts, it is therefore necessary to limit the weight of the cluster rigidity.



(a) Reference mesh. (b) Articulated model.



(c) Skinning weights for the right leg. (d) Skinning weights for the right hip.



(e) Skinning weights for the right forearm. (f) Skinning weights for the sword.

Figure 5.6: Articulated body skinning on the initial frame of the Romans sequence. The skinning weights are automatically computed by a mesh animation software (here we used Blender) once the skeleton has been roughly aligned with the mesh. We display here these weights for 4 of the bones. Note that the articulated model is not limited to a biped, and contains bones for the swords, as well as one root for each of the soldiers.

4 Results

Equipped with the two models for articulated mesh deformation defined in the previous section, we proceed to their evaluation. We start this section with a number of qualitative results that show the overall good behavior of the skeletal-based approach. We then proceed to a comparison of how the two models for articulated motion affect the convergence properties of the registration by considering a difficult registration task between two non-consecutive frames of the lock sequence. We finish our study by varying the weight of the articulated prior for the model that uses cluster of patches, and discuss how stronger priors on articulated motion do not necessarily improve the behavior of the registration.

4.1 Qualitative results.

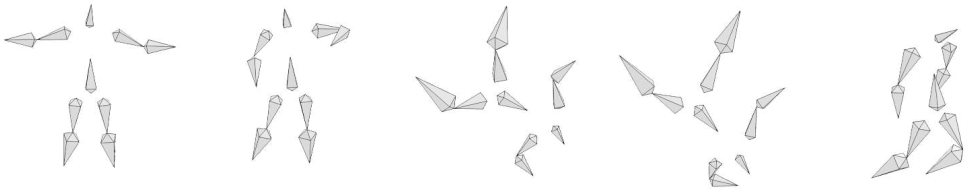
We evaluated both approaches on all the sequences that we had access to and obtained results that were comparable to these presented in chapter 4. We display in figure 5.7 the results obtained for the Free sequence. The skeleton in its recovered pose is overlaid on the original images from one of the cameras. In spite of fast and complex motion, the tracking performed well and the bones are correctly located. In figure 5.8 similar results are shown for some other sequences that involved complex motion.

4.2 Comparison of 3 rigidity models.

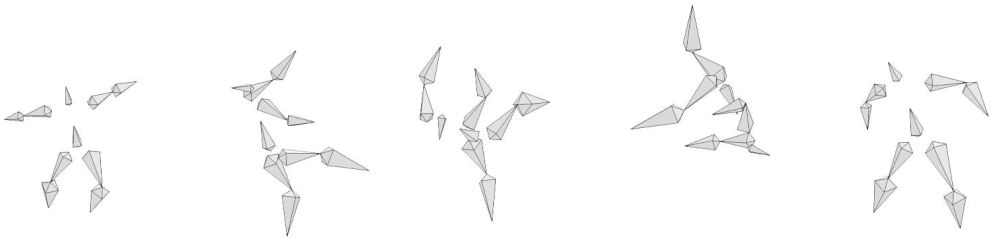
Our comparison of the three rigidity models is built on one experiment that we sum up in figure 5.9. In this experiment, we perform the geometric registration of the lock reference model between two non consecutive frames of the lock sequence. This means that the registration starts with a deformed version of the reference model that corresponds to the state at frame 124, and deforms it further to fit the independently reconstructed mesh of frame 128. This choice of frame was motivated by several reasons: first, there is no self occlusion or change of topology, which allows us to evaluate the convergence with the residual 3D distance between surfaces. Second, the deformation with respect to the reference model is relatively large. Third, the motion is fast with some of limbs changing orientation of more than 90 degrees between the two frames. Finally, the motion is articulated, with both legs going from an extended to a bent position, and the right arm going from a bent to an extended position.

The results of figure 5.9 display for all three deformation models the residual 3D distance between the two surfaces as a function of the number of EM iterations performed. These results were obtained for a registration with soft, probabilistic assignments and neighboring patch prediction. It can be seen that in this case all three rigidity model yield similar convergence times. Furthermore, and as one would expect, the residual error for more constrained deformation models after convergence is higher, because the deformation prior limits the fitting of the data.

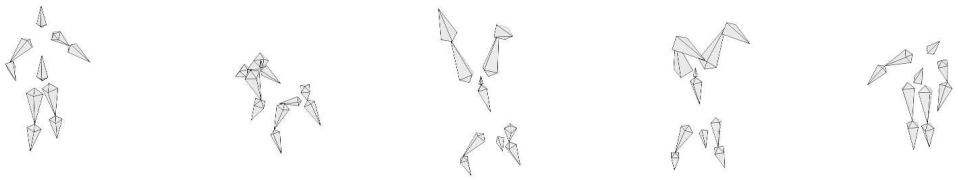
Figure 5.7: Results of the skeleton-based articulated tracking -with probabilistic assignments and patch-prediction- on the Free sequence, overlaid on the original images of camera 2.



(a) Head sequence.



(b) Flashkick sequence.



(c) Handstand sequence.

Figure 5.8: Results of the skeleton-based articulated tracking with probabilistic assignments and patch-prediction.

To further study the convergence behavior with these different deformation priors, we now consider their effects when combined with the different options for the data term. These options, already presented in chapter 4 yield 4 configurations obtained by deciding whether to use soft assignments, and whether to use neighboring patch prediction. We present in figure 5.10 compared convergence rates for each rigidity model and each data term configuration. We also show in the same figure the deformed meshes that correspond to the convergence point of each curve. We make the following observations on these experimental results.

- The most striking fact is that for the skeleton-based articulated model, the only configuration that converged properly was the one with soft assignments and neighboring patch prediction. The consistent and satisfactory performance on the upper body can be explained by the relatively little movement for these parts. On the legs. deterministic assignments appear to fail. A probable explanation for these errors is that the orientation of the vertices on the top of the upper right thigh in the starting pose match the orientation of the vertices on the top of the left thigh in the target mesh. Because the assignments are deterministic, and these points are the closest compatible matches, the optimizer immediately pulled the right thigh towards the left one and created the leg switch from which

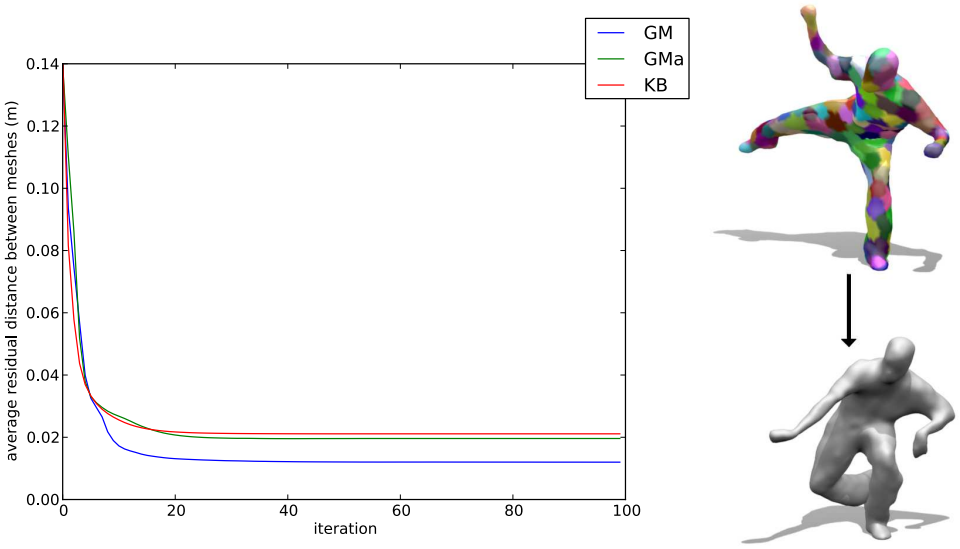
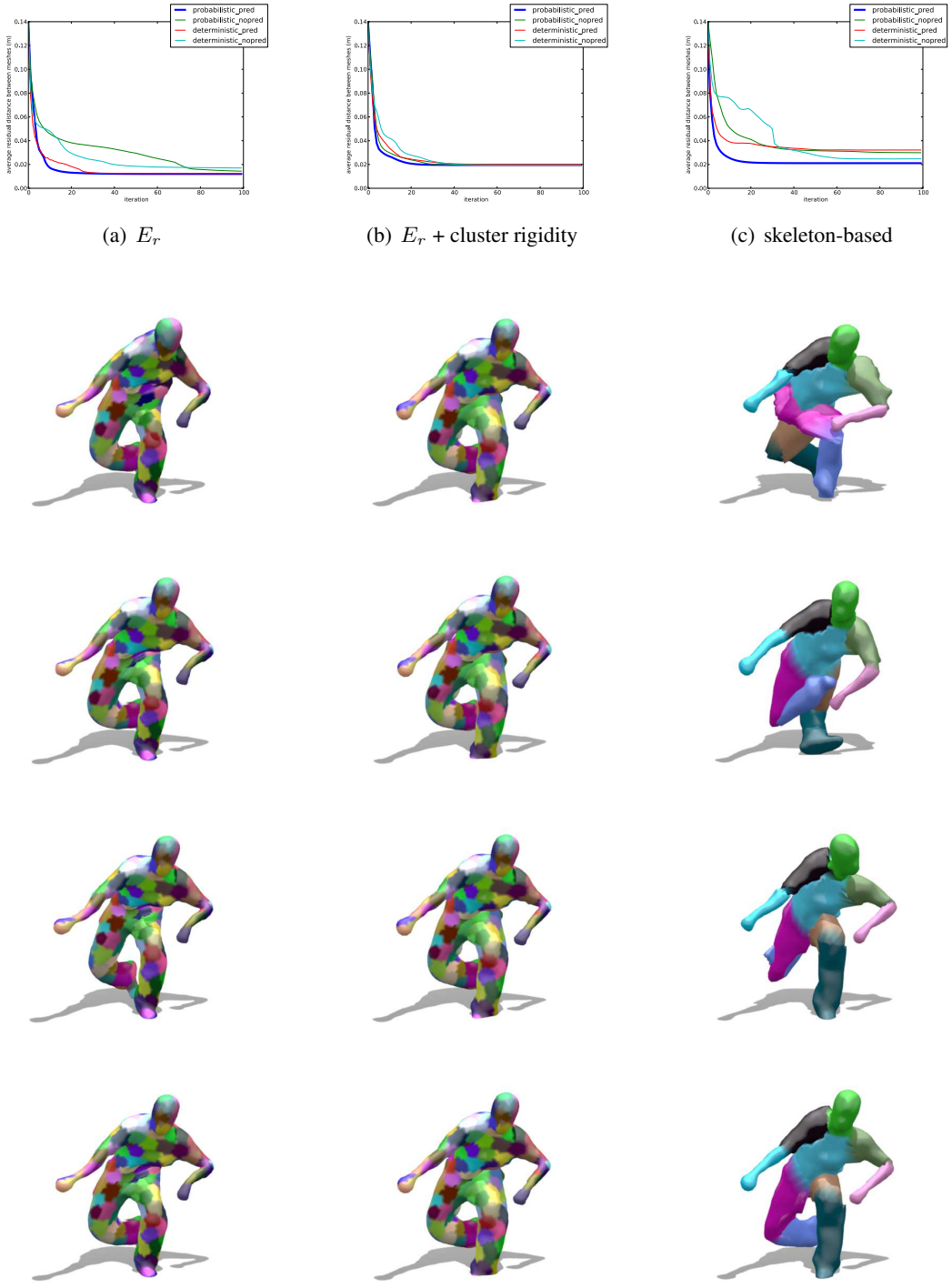


Figure 5.9: Compared convergence properties for different rigidity models. These results are obtained with soft assignments and neighboring patch prediction. On the lock sequence, we chose two frames, separated by 4 frames that exhibited a significant articulated motion, without topology changes so that we could use the 3D residual distance between surfaces as error measure. The results compare the convergence properties of the rigidity model of chapter 3 (GM), the same rigidity model complemented with a force maintaining cluster cohesion (GMa), and the kinematic tree (KB). It can be seen that the patch-based rigidity fits the surface more accurately than the two articulated methods.

he could not recover.

- For the skeleton-based model with soft assignments (two last rows), the neighboring patch prediction mechanism helped prevent the wrong registration of the right calf. Having the right thigh introduce hypothesis for part of the right calf in the optimization apparently helped the optimizer find a new mode where the knee was not bent 180 degrees and the calf actually could explain some observed data.
- The compared convergence rates for the model used in chapter 4 and the same model augmented with cluster rigidity seem to indicate better and more consistent convergence rates when using cluster rigidity. This is a coherent an expected behavior. Penalizing deformation inside the clusters prevents the optimizer from wandering in some modes of deformation. More importantly it immediately propagates registration information through the block of the deformation graph that corresponds to a limb. This means that low frequency registration error is attenuated much faster. While the E_r force penalizes local deformation and operates at a fine level, the cluster rigidity operates at a coarser level.
- Finally, it is very interesting to note that for both the cluster rigidity and the skeleton-based deformation models, the left foot goes through the floor, which indicates an imprecise registration. Our explanation, confirmed by looking at the skinning of the reference mesh is that the joint location for the knee was placed too high. Because both articulated



(a) E_r (b) E_r + cluster rigidity (c) skeleton-based

Figure 5.10: Comparison of the convergence properties of different rigidity models. The top row displays the evolutions of the residual mesh distance in 3D for different registration methods. The following rows show the point of convergence for each of the curves in the top row. (respectively deterministic no prediction, deterministic prediction, probabilistic no prediction, probabilistic prediction).

models impose one unique place for bending, the foot ended up under the floor. In comparison, the non-articulated model relying on the E_r force correctly registered the foot, which indicates that this model is not only more generic, but also more robust to model imprecisions.

4.3 Limitations of the rigid clusters model

In the previous paragraph we have discussed how both articulated models forced more or less strictly the bending to happen at one location. We saw with the presented example and how this could affect the final results. More importantly, constrained articulated model also affect the convergence of the registration. We present in figures 5.11 and 5.12 details of the convergence for the three compared rigidity models. These figures show the state of the registration through iterations 1 to 9, as well as the final convergence state after 100 iterations. These meshes correspond to the error curves of figure 5.9.

It can be observed that the patch-based deformation model of chapter 3 goes through very deformed states during the inference, while both articulated models preserve the rigidity of the body parts.

- The skeleton-based articulated model converges pretty quickly to the correct pose. It appears that the data corresponding to the right forearm is classified as outlier for the first two iterations, and therefore has a negligible weight in the registration. The right forearm of the model stays in a constant orientation with respect to its parent bone for the first two iterations. Because the neighboring patch predictions test a configuration where the arm is extended, some patches of the forearm around the elbow are predicted at the right location, get associated with data, and the whole forearm is rapidly reintegrated in the registration.
- The cluster-based articulated model converges more rapidly than the one with the simple patch-based rigidity for most of the body. The notable exception is the left knee which is still incorrectly registered after 9 frames. It is bent backwards. Because most of the body is correctly registered, the σ parameter is evaluated as small, and the weight of most of the left leg data points is diminished in the registration. Because the left thigh and left foot are closer to data-points, their association to data is stronger. Where the patch-based rigidity could locally deform the template to fit the data and then later on propagate this information to the rest of the surface, the cluster-based rigidity immediately has to solve a non-local problem: the weak associations around the knee must force a reevaluation of the strong (and in this case erroneous) associations at the foot and hip because the regularization forces clusters to move rigidly.

That example shows that articulated models are a double-edged sword: on one side they can quickly propagate good registration information from one end of a bone to the other, resulting in quicker convergence. On the other side, an erroneous registration of one end (the hip in our example) also propagates to the other end and can limit the convergence speed , or simply prevent the inference from reaching a global minimum. To confirm this behavior, we varied the weight of the cluster rigidity relatively to the simple patch-based

rigidity and confirmed that when the cluster rigidity was too strong, the registration simply failed to converge to the proper minimum.

5 Conclusion

In this chapter, we have examined the case of articulated objects, and presented a comparison of rigidity models used with the Bayesian mesh registration framework of chapter 4. The first articulated model that we have developed simply enforces the rigidity of clusters of patches on the deformation graph that was defined in the previous chapters. The second articulated model parametrizes the motion with a kinematic chain, and we have presented a number of developments to efficiently transpose the ideas of soft-assignments and patch predictions so that they could be used with this parametrization of the deformation.

Our comparison of the three models on a challenging registration task has revealed that within our probabilistic framework, using more specific deformation priors can speed up the convergence. However, these priors do not necessarily improve the results. We have observed convergences towards erroneous configurations, as well as imprecise registration results due to an imprecise location of the joint in the model.

This study opens the way for future work on rigidity models. In particular, a promising area of research would be to simultaneously perform the tracking of the surface and estimate the local rigidity of different parts of the surface. In other words, the challenge would be to initiate the inference with our generic deformation model and to have the true rigidity of the observed material be a variable in the optimization.

References

[1] Anguelov, D., P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis
2005. Scape: shape completion and animation of people. *SIGGRAPH*.

[2] Bandouch, J., F. Engstler, and M. Beetz
2008. Accurate Human Motion Capture Using an Ergonomics-Based Anthropometric Human Model. In *Proceedings of the Fifth International Conference on Articulated Motion and Deformable Objects (AMDO)*.

[3] Baran, I., D. Vlasic, E. Grinspun, and J. Popovic

2007. Automatic rigging and animation of 3D characters. In *SIGGRAPH*.

[4] Bregler, C. and J. Malik
1998. Tracking people with twists and exponential maps. In *CVPR*.

[5] Bregler, C., J. Malik, and K. Pullen
2004. Twist based acquisition and tracking of animal and human kinematics. *IJCV*.

[6] Bubb, H., F. Engstler, F. Fritzsche, C. Mergl, O. Sabbah, P. Schaefer, and I. Zacher
2006. The development of RAMSIS in past and future as an example for the cooperation between industry and university. *International Journal of Human Factors Modelling and Simulation*.

[7] Caillette, F. and T. Howard
2004. Real-time markerless human body tracking with multi-view 3-d voxel reconstruction. In *BMVC*.

[8] Carranza, J., C. Theobalt, M. A. Magnor, and H. peter Seidel
2003. Free-viewpoint video of human actors. In *SIGGRAPH*.

[9] Cheng, S. Y. and M. M. Trivedi
2007. Articulated Body Pose Estimation from Voxel Reconstructions using Kinematically Constrained Gaussian Mixture Models: Algorithm and Evaluation. In *2-nd Workshop on Evaluation of Articulated Human Motion and Pose Estimation (EHuM2)*.

[10] Cheung, K. M., T. Kanade, J.-Y. Bouguet, and M. Holler
2000. A real time system for robust 3d voxel reconstruction of human motions. In *CVPR*.

[11] Corazza, S., L. Mündermann, E. Gambaretto, G. Ferrigno, and T. P. Andriacchi
2010. Markerless motion capture through visual hull, articulated icp and subject specific model generation. *IJCV*.

[12] Deutscher, J. and I. Reid
2005. Articulated Body Motion Capture by Stochastic Search. *IJCV*.

[13] Fischler, M. A. and R. A. Elschlager
1973. The representation and matching of pictorial structures. *IEEE Trans. Comput.*

[14] Gall, J., J. Potthoff, C. Schnörr, B. Rosenhahn, and H.-P. Seidel
2007. Interacting and annealing particle filters: Mathematics and a recipe for applications. *Journal of Mathematical Imaging and Vision*.

[15] Gall, J., B. Rosenhahn, T. Brox, and H.-P. Seidel
2010. Optimization and filtering for human motion capture. *IJCV*.

[16] Gall, J., C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel
2009. Motion capture using joint skeleton tracking and surface estimation. In *CVPR*.

[17] Gavrila, D. M.
1999. The visual analysis of human movement: A survey. *CVIU*.

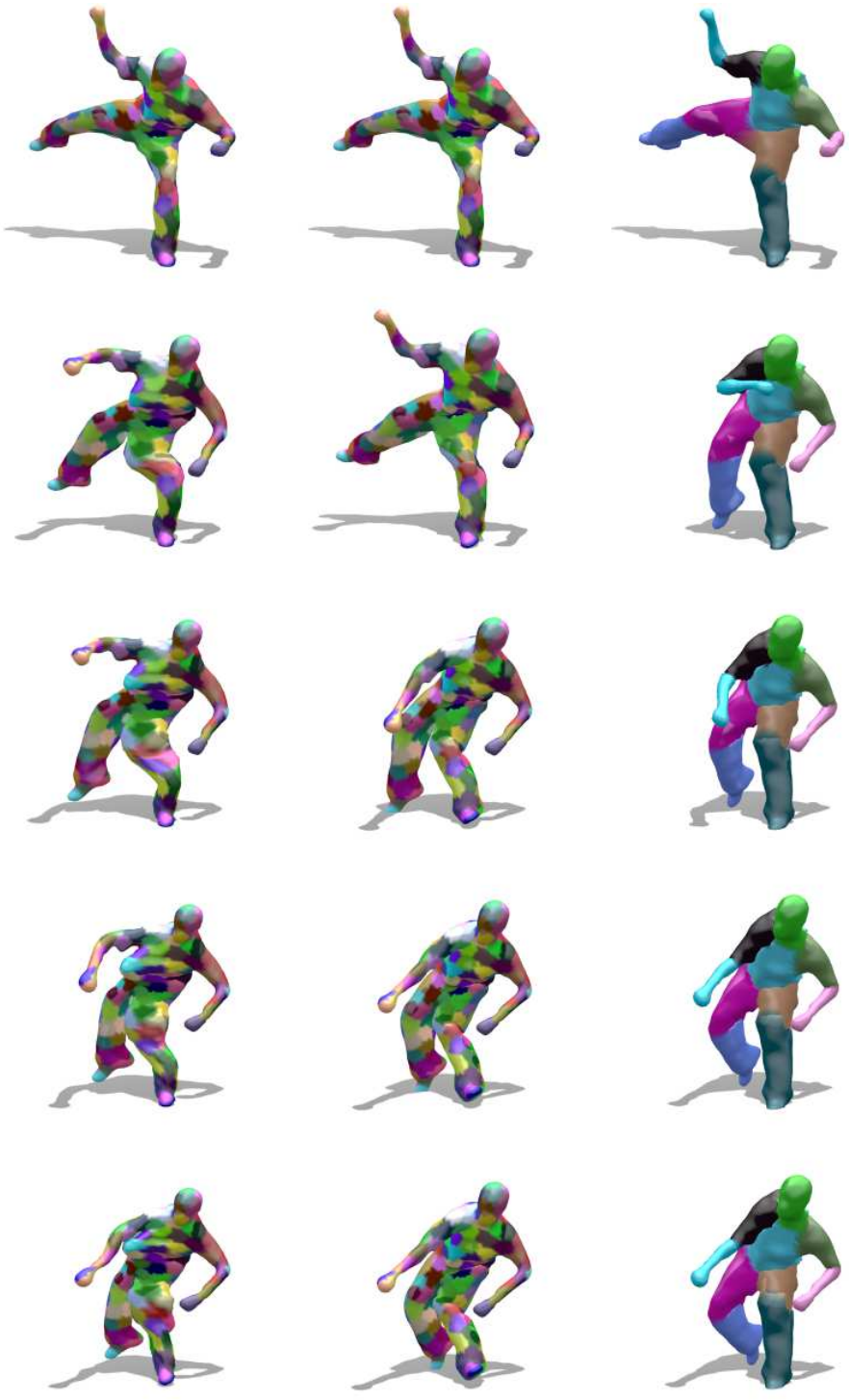


Figure 5.11: The rows show iterations 1 to 5 of the registration for the three rigidity models. The left column corresponds to the patch-based rigidity of chapter 3. The middle column corresponds to the same rigidity complemented with a force maintaining the rigidity of cluster of patches. The last column shows the results for the kinematic tree parametrization.

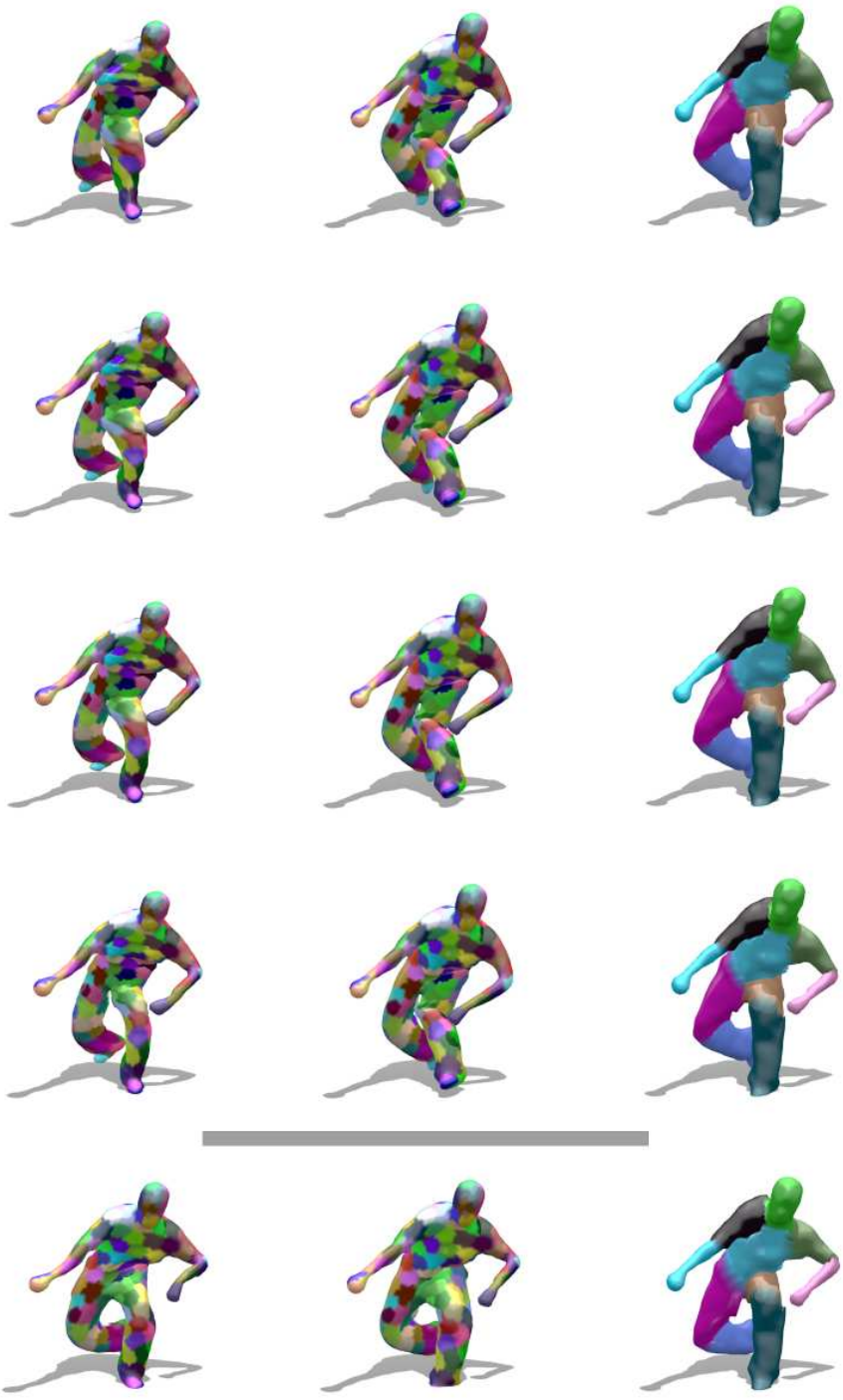


Figure 5.12: Iterations 6 to 9 of the registration for the three rigidity models. The final row shows the final result after 100 iterations.

[18] Gavrila, D. M. and L. S. Davis
1996. 3-d model-based tracking of humans in action: a multi-view approach. In *CVPR*.

[19] Horaud, R. P., F. Forbes, M. Yguel, G. Dewaele, and J. Zhang
2011. Rigid and articulated point registration with expectation conditional maximization. *PAMI*.

[20] Horaud, R. P., M. Niskanen, G. Dewaele, and E. Boyer
2009. Human motion tracking by registering an articulated surface to 3-d points and normals. *IEEE PAMI*, 31.

[21] Moeslund, T. B., A. Hilton, and V. Krüger
2006. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*.

[22] Mundermann, L., S. Corazza, and T. P. Andriacchi
2007. Accurately measuring human movement using articulated icp with soft-joint constraints and a repository of articulated models. In *IEEE CVPR*.

[23] Murray, R. M., S. S. Sastry, and L. Zexiang
1994. *A Mathematical Introduction to Robotic Manipulation*.

[24] Plänkers, R. and P. Fua
2003. Articulated soft objects for multiview shape and motion capture. *PAMI*.

[25] Poppe, R.
2007. Vision-based human motion analysis: An overview. *CVIU*.

[26] Ramanan, D.
2011. *Visual Analysis of Humans*, chapter Part-based Models for Finding People and Estimating Their Pose.

[27] Sigal, L., M. Isard, B. H. Sigelman, and M. J. Black
2003. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *NIPS*.

[28] Wang, L., W. Hu, and T. Tan
2003. Recent developments in human motion analysis. *Pattern Recognition*.

CHAPTER 6

Conclusion and Perspectives

1 Conclusion

In this thesis, we have explored novel techniques to capture the motion and deformation of surfaces observed in multi-camera environments. Our research was motivated by two main ideas: firstly, we wanted the process to be unintrusive so that motion capture would not get in the way of the concurrent digitization of appearance and shape of the observed objects. Our second motivation, and essential contribution with respect to previous art on markerless motion capture is that that we have tackled this problem without making strong assumptions on the nature of the observed object. In fact, we have shown on several examples that the genericity of our approach allows to track several objects of arbitrary nature, given a topologically suitable frame in the sequence to be used as model. This is in sharp contrast with the vast majority of the literature that tends to focus on human motion capture and mostly limits itself to capturing the performance of a single actor in a multi-view studio. This thesis has reported on our progress and contributions to the ongoing research on the use of generic deformation models for the purpose of motion capture.

In chapter 3 we have presented a robust mesh deformation model that has the following traits. It is purely surface based and creates a control graph for the deformation that respects the topology of the reference frame. It emulates elastic behavior with respect to the reference frame. It explicitly optimizes the local rotations of the surface and parametrizes updates of these rotations with exponential maps contrary to previous work on similar models. We have furthermore shown on several applications that this deformation framework is an interesting tool for data-driven mesh deformation in computer vision.

In chapter 4 we have introduced a Bayesian formulation of deformable mesh registration. Our work has explored further a path opened by previous works on rigid, articulated and deformable registration (e.g. EM-ICP, Softassign, TPS-RPM). We have shown that this formulation was a mathematically principled and effective way to handle parasite geometry thanks to the introduction of an outlier class in the Bayesian model. We also have shown

that the variance parameter σ allowed to benefit from a coarse-to-fine behavior that improved convergence properties in ambiguous situations. Furthermore, and in contrast with prior art on deformable registration we have optimized this variance parameter explicitly instead of forcing convergence by performing deterministic annealing. We have additionally introduced a local prediction mechanism of neighboring patches that maintains several hypothesis on the relative locations and orientations of neighbors in the control graph, allows the optimizer to quickly favor one over the others, and therefore greatly improves the convergence rate of the registration. Finally, we have produced experimental results that confirm that this single mathematical formulation constitutes a generic tool that performs consistently well for a given set of parameters on various datasets involving several objects of different nature.

In chapter 5, we have looked at the case of articulated objects and have proposed two models to account for prior knowledge on their articulated nature in the inference for motion. We have illustrated on a challenging registration task that although this prior knowledge can make the convergence faster, it can also lead to imprecise or even erroneous results on tasks where the generic deformation model of chapter 4 performs well.

2 Perspectives

We conclude this dissertation by first presenting how we used our work in a new application for marker less motion capture in the medical field. We then discuss the fundamental limitation of our work that is the fixed topology of the model, and go over the current research effort from the computer vision community on the subject. This allows us to discuss the perspectives opened by our work for further research on these topics.

New applications of motion capture The developments presented in this thesis were focused on the production of 4D content in multi-view systems. They also have implications for research on model-based motion capture, and open perspectives for new applications. We cite as an example the work that we presented at MICCAI 2010 [5]. In this paper, we installed a multi-view system in the operating room and looked at how motion capture could be used to evaluate the accumulated exposition of the medical personnel to the radiation from the X-ray imaging device used during interventions. Although the dose received by the staff over a single intervention is limited, the repeated exposure can have important long term consequences on their health. Motion capture is challenging in this setting for several reasons: first, the operating room is a complex environment that involves several people and moving objects made out of different materials. There is therefore a need for a generic tracking method that can build a complete spatio-temporal description of the surgical procedure that will later be fed in a physics simulation package to evaluate the actual

radiation exposure of the staff. The second difficulty is lead vest and collar usually worn by the physician. The lead vest being rather long, articulated body models are not well suited for tracking. Although our experimental conditions were simplified, we showed in our results that multi-camera systems and our generic marker less motion tracking method allow interesting new applications in interventional environments, and that further development of such ideas will most likely be integrated into the operating room in the future.

Segmentation of an animated mesh The temporal trajectories encoded in 4D models can be used to recover information on the nature of the observed object. For example, it is interesting to learn from these trajectories that that some parts tend to bend or stretch more than others, or maybe that the object tends to deform in an articulated way. Shape segmentation consists in identifying a partition of a shape such that each segment fulfills a defined criterion. A considerable amount of literature already exists on segmenting static shapes from static geometric criteria (e.g. curvature, geodesic integral, convexity). If motion information is added to the shape, the temporal trajectories of points can be used to define new segmentation criteria.

For example de Aguiar et al. [3] automatically compute a skeleton and skinning weights from a animated mesh obtained with the technique developed in their previous work [2]. The animation can then be modified by an artist who edits the skeletal structure instead of having to edit meshes directly. Another example is the recent work by Stoll et al. [7], who begin by animating an articulated structure from the visual data, then segment out pieces of the reference mesh whose deformation can not be well explained by an articulated motion and re-synthesize convincing cloth simulation for these pieces of the surface.

The mesh deformation framework presented in chapter 3 performs some kind of segmentation when it seeds the patches on the reference mesh. However this segmentation is only constrained by the patch size and regularity heuristics for the patch seeding. As noted in the conclusion of chapter 5, it would be much more interesting for the patch structure and regularization energy to adapt so that they would match the actual rigid parts of the tracked body. Rigid parts could be identified from the mesh animation with a method such as [2]. Knowing which parts tend to move rigidly and adapting the regularization energy to favor rigid behavior of these parts would in turn help to constrain the inference of the mesh animation. However the improvement on the available datasets would be marginal with respect to our current results, and difficult to evaluate as our only quantitative evaluation metric is the silhouette re-projection error.

Topology changes Identifying rigidly moving parts on an animated mesh of constant topology is interesting but limited in its applications. The more general challenge is to handle gracefully changes of topology between the reconstructed surfaces of a same deforming object at two different times. Topology changes were discussed in chapter 2 and an typical case presented in figure 2.4. This example illustrates perfectly the challenge at hand: the problem is not to assert whether the hand and the thigh deform together as part of a rigid segment. The problem is to assert whether they move together at all, as we need this information to regularize the deformation.

In our work, this problem was left aside by postulating the existence of a reference frame that is topologically suitable for tracking. This is usually a valid hypothesis when dealing with a human character, as it is likely that at some time in the sequence, the actor will have both arms and both legs correctly separated from his body. Furthermore, if our work is used to build an animated mesh of a performance in the context of content production, it is reasonable to make sure that the actor will start in a suitable pose. However, finding a reference frame becomes more complicated as the scene complexity increases. For example in the Ball sequence presented in figure 4.7, there are very few topologically suitable frames in the sequence because there are more conditions to be met: the ball should not be in the hands of either actor and both actors should not have their arms merged with their torso.

Essentially, the question is whether letting go of model-based deformation is feasible. In our study of prior art in chapter 2, the model-based deformation of a reference template all along the sequence had been identified as the key to long term precision and robustness. Our work in this thesis has explored to which extent assumptions on the model's rigidity could be relaxed to increase generality without losing either precision or robustness. However, we preserved the assumption of constant topology.

Among the recent works that have explored how the constant topology constraint could be relaxed, Wand et al. [10] propose a technique to build a template model from range scans of a deformable object and compute at the same time a deformation field registering this reference template to every observation of the sequence. Franco and Boyer [4] hypothesize a number of rigid clusters in the shape and simultaneously estimate rigid motions for these clusters, as well as which points of the reference shape belong to which cluster. They effectively estimate the rigid motions (deformation) and cluster assignments (topology) simultaneously in an EM framework. Other works let go of the notion of template altogether. For example, Varanasi and Boyer [9] first notice that the convexity of a volumetric part of object is a strong indication that it might deform rigidly. Then they try to track each convex part independently across the sequence to validate the hypothesis that it is a rigid cluster. They add validated clusters to a list until they have enough valid clusters to explain all the observed geometry in the sequence. Popa et al. [6] process the sequence hierarchically by first registering frames with their direct adjacent frame, and handling the potential topological changes there. They then register at each new level the resulting moving shapes of the previous level together and solve for topological inconsistencies. They proceed recursively until they obtain a unique piece of geometry that can be deformed to any shape of the sequence by moving through the tree of correspondences.

Our future work on the subject will be guided by three ideas. First, we will explore methods that do not process the sequence linearly, inspired by the work of Popa et al. [6] that we just mentioned or the recent work by Budd et al. [1]. Second, we will persist in parameterizing the deformation with a deformation graph but will investigate how its connectivity can be updated during the inference. Finally, we will investigate how the mesh representation can be abandoned, to avoid the bookkeeping that is required when meshes change topology. Attaching simple surfels to the deformation graph or small generative elements encoding geometry or color [8] should be promising paths of research.

References

[1] Budd, C., P. Huang, and A. Hilton
2011. Hierarchical Shape Matching for Temporally Consistent 3D Video. In *3DIMPVT*.

[2] de Aguiar, E., C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun
2008a. Performance capture from sparse multi-view video. In *SIGGRAPH*.

[3] de Aguiar, E., C. Theobalt, S. Thrun, and H.-P. Seidel
2008b. Automatic Conversion of Mesh Animations into Skeleton-based Animations. In *EUROGRAPHICS*.

[4] Franco, J.-S. and E. Boyer
2011. Learning temporally consistent rigidities. In *CVPR*.

[5] Ladikos, A., C. Cagniart, R. Gothbi, M. Reiser, and N. Navab
2010. Estimating radiation exposure in interventional environments. In *MICCAI*.

[6] Popa, T., I. South-Dickinson, D. Bradley, A. Sheffer, and W. Heidrich
2010. Globally Consistent Space-Time Reconstruction. In *Computer Graphics Forum*.

[7] Stoll, C., J. Gall, E. de Aguiar, S. Thrun, and C. Theobalt
2010. Video-based reconstruction of animatable human characters. *SIGGRAPH ASIA*.

[8] Stoll, C., N. Hasler, J. Gall, H.-P. Seidel, and C. Theobalt
2011. Fast Articulated Motion Tracking using a Sums of Gaussians Body Model. In *ICCV*.

[9] Varanasi, K. and E. Boyer
2010. Temporally coherent segmentation of 3d reconstructions. In *3DPVT*.

[10] Wand, M., B. Adams, M. Ovsjanikov, A. Berner, M. Bokeloh, P. Jenke, L. Guibas, H.-P. Seidel, and A. Schilling
2009. Efficient reconstruction of nonrigid shape and motion from real-time 3d scanner data. *ACM Trans. Graph.*

List of Figures