Modélisation Spatio-Temporelle des scènes dynamiques 3D à partir de données visuelles

Kiran Varanasi

UNIVERSITÉ DE GRENOBLE

THÈSE

pour obtenir le grade de
Docteur de l'Université de Grenoble

Spécialité : Mathématique et Informatique
préparée au **Laboratoire Jean Kuntzmann**
dans le cadre de l'Ecole Doctorale :
Mathématiques, Sciences et Technologies de l'Information, Informatique

———

présentée et soutenue publiquement par

Kiran Varanasi

le 2 Décembre 2010

Modélisation Spatio-Temporelle des scènes dynamiques 3D à partir des données visuelles

———

Directeur de thèse : Edmond Boyer

———

JURY

Président :	**Georges-Pierre Bonneau**
Rapporteurs :	**Christian Theobalt**
	Renaud Keriven
Examinateurs :	**Radu Horaud**
	Slobodan Ilic
	Edmond Boyer

Spatio-Temporal Modeling of Dynamic 3D Scenes from Visual Data

Kiran Varanasi

Amma, Nanna ki premato ankitam

Acknowledgments

Firstly, I offer my thanks to my adviser Edmond Boyer for his guidance and support during my Ph.D. He has introduced me to a fascinating problem and gave me the confidence to analyze it from multiple angles. I have learnt from him the art of choosing the right problems, exploring them methodically, and presenting my results in a lucid manner. I am also greatly indebted to my colleagues in the PERCEPTION team of INRIA for the illuminating discussions they offered me, both for understanding the problem and for improving my methods. I should particularly acknowledge the involvement of Andrei Zaharescu and Radu Horaud in the work presented in chapters 4 and 5. I offer my thanks to the students and engineers who have built the Grimage room at INRIA, and who helped me acquire the datasets used in this thesis. Without them, none of my work would have been possible. A special thanks should also go to my colleague David Knossow for several brainstorming sessions, and for helping me around in Grenoble during my initial days. I am thankful to the wider research community who have released their source-code and datasets for public usage. My list of thanks shall not be complete without acknowledging the influences of P.J. Narayanan and other mentors at IIIT Hyderabad, who instilled in me a love for research during my formative years.

Apart from a rich academic life, my friends at INRIA gave me a colourful bonhomie and camaraderie that built several fond memories through these four years of my life. I am equally thankful to my friends in Grenoble outside INRIA for the same. I should also thank my beloved sister Renu and brother Stalin for their love and support.

Finally, I would like to offer my deepest love and gratitude for my parents - Rani and Subrahmanyam.

Contents

Chapter 1

Introduction

Contents

Space-time models (or 4D models) are descriptions of dynamic activities of the real world, that can be stored, analyzed by computers and visualized at distance. Digital imaging and computing technologies of today have matured to a level capable of recording human activities in rich three dimensional detail, and across a length of time. Such photographic 4D models shall in the future be the equivalents of photographic video of today, and produce equivalent artifacts of human culture and heritage. In this thesis, we are concerned with the problem of building such space-time models from the bottoms up through a multi-camera environment. Our contributions are primarily towards estimating motion, both at a coarse and at a dense level from surfaces reconstructed in this fashion.

1.1 Spatio-Temporal Modeling

The world we live in is a dynamic world, and we often ascribe meaning to natural objects based on how they behave over time. The evolution of intelligence across different living organisms can be understood as the evolution of the speed with which the dynamics in the environment are modeled. Such modeling is essential for the survival of an organism and hence determines its evolutionary success. The visual system remains the most important system for sensing the environment, and to address this need, the eye has evolved independently at different branches of the phylogenic tree. This evolution has been not towards more faithful means of capturing the 3D layout, but towards more effective means of capturing the dynamics of objects in the surroundings. Thus, the evolution of the eye is overwhelmingly coupled with the evolution of the neural system that *senses* these dynamics in the scene. This knowledge about dynamics is also represented in a modular fashion, where the applicability of a concept increases owing to its *repeated observation in time*.

Neuroscientists observe that this ability to detect repetitions in time is an essential step in the development of human visual cognition. In congenitally blind children who are given corrective surgery to restore vision, the ability to *see* the world doesn't arrive immediately but gradually. The process of visual learning often takes several days. Processing scene dynamics, especially detecting repetitions of objects, is found essential for the child to derive meaning from visual input [Bouvrie & Sinha 2007]. In light of these findings, we may imagine the need for a computer or for a robot to also learn and represent scene knowledge through the dynamics of objects, in order to operate intelligently in the environment. We call this process the spatio-temporal modeling of the scene. A spatio-temporal model represents both the 3D spatial structure of the objects and their temporal behavior in a combined manner.

1.1.1 Spatio-Temporal Modeling in Human Vision

The human brain is one of the most complex systems that has evolved in nature to model scene dynamics. More than half the neurons in the human brain are engaged directly in visual processing. The eyes are a calibrated stereoscopic pair of cameras, that are only the tip of the iceberg that is the human visual system. The human brain reads the 3D layout of these objects in the scene very rapidly, and focuses the eyes to inspect certain areas of interest, through a series of eye movements known as the saccadic movements. These movements are particularly targeted to sense the scene dynamics, both as a reaction to motion and as an anticipation of motion. Apart from the saccadic eye movements that scan within the field of vision, humans can also orient their eyes and move them around through various muscles and actuators, including skeletal joints in the neck and the limbs. Also unlike many animals which can interpret the dynamics of the scene only in the *"now"*, humans can also interpret them through a variety of past experiences. This is achieved by

the linguistic capabilities of the human brain that permit abstract and symbolic reasoning. So even areas of the brain that are not conventionally considered part of the visual system do help in the processing of visual scene dynamics.

Human vision is pre-occupied with recognizing and interpreting activities of the daily lives of human beings. These often concern other human beings and interactions amongst them. In this thesis, we address the spatio-temporal modeling of similar scenes, however by a computer through visual information acquired from a static multi-camera setup. This has neither the active visual processing capabilities of the human visual system, nor the abundant knowledge about visual scene dynamics that is learnt by a human being since childhood. However, we build certain primitive spatio-temporal models by observing the very *"now"* of the scene. We target scenes of interactions amongst human beings. There can be various other natural and man-made objects in the scene, that may be used for the interaction. This makes the problem very generic and hard, as we explain in the next section.

1.1.2 What Constitutes a Dynamic Scene ?

We refer to a *scene* as composed of various solid objects. A static scene contains two types of visual information - *structure* and *appearance*. Structure refers to the 3D layout of different objects in the scene, the geometry on their surfaces and the inherent volume enclosed within them. Structure may also be referred to as *shape*. Appearance refers to the colour and the texture of a surface and how it reflects light towards the eye. A dynamic scene may be described simplistically as a sequence of static scenes, but it is not correct. The structure and appearance of several objects in the scene remain the same over time, and the scene dynamics are defined over these temporally re-occuring objects. The motion in the scene indeed needs to be described as the relative change of structure *between* such objects. With respect to these object dynamics, a dynamic scene can be decomposed in a modular fashion.

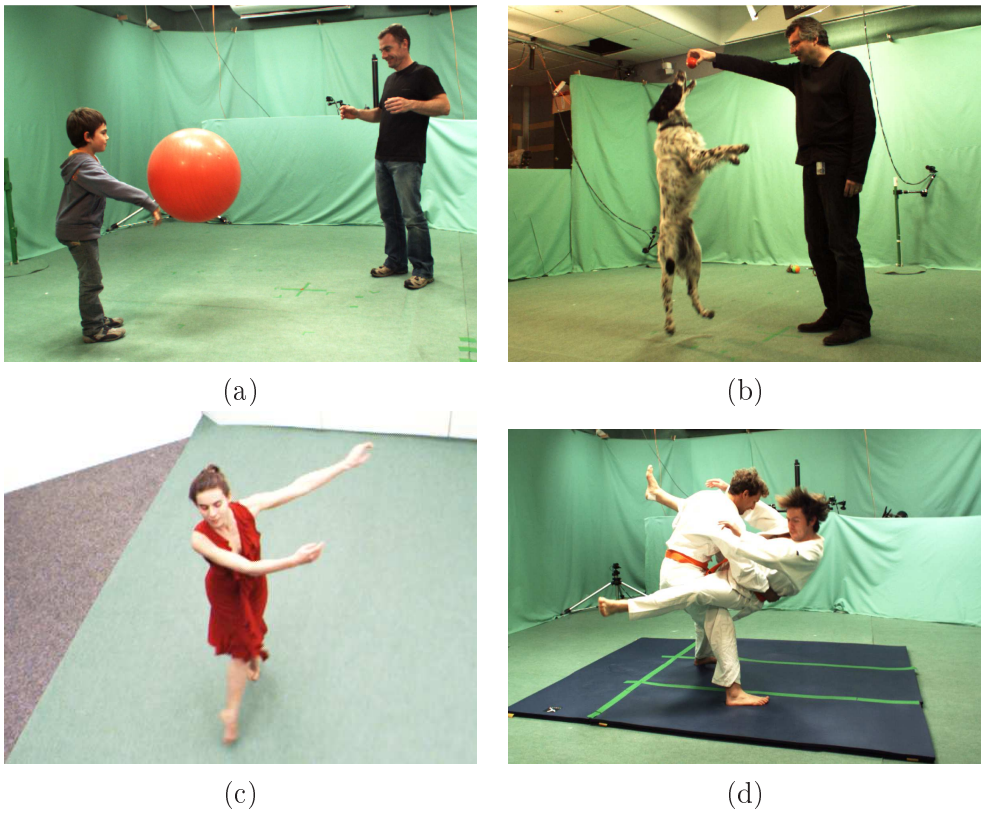
actors : independent agents in the scene with a connected structure. These actors can be human or non-human.

props : objects that are set into motion by these actors.

background : consisting of completely static objects.

flutter : minor motion disturbances in the scene due to environmental factors such as wind, or due to physical properties of the objects such as elasticity. The latter can be vividly observed in scenes of human activity - on the clothes, skin and hair of the actors.

We are concerned with scenes of human visual interest. Such scenes may be composed of any of the above elements, and whose dynamics are completely arbitrary. There may be any number of independent actors. The actors may be of different sizes - figure 1.1-(a) shows a boy and a man playing with a ball. They need not be even human - figure 1.1-(b) shows an example scenario where a man plays with his



(a) (b)

(c) (d)

Figure 1.1: Examples of Dynamic 3D Scenes : (a) two actors of different size playing with a ball (b) a person playing with a dog (c) a girl dressed in a loose robe dancing (d) two actors juggling with clubs

pet dog. The actors may be dressed in very diverse types of clothing, which occlude the limbs and flutters greatly - figure 1.1-(c) shows a dancer dressed in a long robe with a belt. Different kinds of props may be used for interaction. The type of the interaction between the actors can be of close contact - figure 1.1-(d) shows two martial artists enacting a judo move. And finally, actors need not be continuously present in the scene - some actors may disappear and new actors may appear into the scene.

Depending on the nature of an application, certain assumptions can be made about the number, size or type of the objects in the scene or on the nature of their movements. But in the general case, no such assumptions whatsoever can be made.

1.1.3 What is a Spatio-Temporal Model ?

We take a short philosophical interlude to understand what a spatio-temporal model means. Philosophers have argued that visual perception consists of grasping two different notions from reality - concrete and abstract [Dretske 1988]. The *concrete* is composed of *objects* that have a specific locus in space, and by *events* that have a specific locus in time. The *abstract* is composed of various *properties* that are

possessed by objects or events, and that could be shared between them - such as colour, size, texture, speed etc. In computer vision, it has been recognized that the modeling of the abstract properties depends highly on the context of the concerned application and on other objects that exist in the environment. Thus, the *abstract* and the *concrete* are tied together. But there still remains a division between modeling the *objects* and *events* of the concrete world. Such division may not be justified. This is because meaning is related to causality, and causality in nature is intimately tied to the directionality of time (as is attested by the second law of thermodynamics in physics). Thus, spatial knowledge may indeed deemed to have no *meaning* without the temporal dimension. Any model of reality needs to have both spatial and temporal aspects - the lack of one or the other makes it incomplete.

We now present three elements of knowledge that relate to both spatial and temporal dimensions. A dynamic scene can be thought of as defined in these terms.

Temporally consistent structure This is given by a division of the scene into *objects*, the surface geometry of each of these objects and the intrinsic volume enclosed by these surfaces.

Temporally consistent appearance This is defined as a function over the surfaces of objects - as simple colour information or as more complex surface-reflectance properties.

Motion over time This can be given either *coarsely* as a series of transformations for each object, or *densely* as a function over the surface of each object.

Spatio-temporal modeling means deriving these elements of knowledge by observing the scene. These three elements need to be represented at different scales, depending on the type of application we put them to use. A spatio-temporal model is categorized both by the power of *modularity* and by the richness of *detail* in describing the different objects in the scene. An ideal model is not only concise and semantically organized, but also possesses rich detail in all the above three components. But often these two objectives are conflicting with each other. The temporal length of the dynamic sequence is often important in judging the trade-offs that need to be made between these two objectives. In this thesis, we address sequences ranging from several seconds to a few minutes. We represent structure in terms of triangular meshes, and appearance in terms of colour values defined on the mesh facets. We present two methods for spatio-temporal modeling. We estimate (a) a coarse model which decomposes a scene into objects of interest and computes motion at the level of objects and (b) a dense model which traces the subtle changes of structure on objects and computes motion as a dense function over their surfaces, identifying flutter of clothes and skin. We build these models completely from the bottoms up, without making any assumption on the scene being captured. Our models can be improved with respect to both the criteria of *modularity* and *detail*. Specifically, challenges remain in the modeling of the second element - temporally consistent appearance. In the concluding chapter 6, we identify these richer spatio temporal models as avenues for future work in the community.

1.1.4 Supervised Learning vs. Unsupervised Modeling

If we have certain knowledge about the scene being observed, we can impose a model to fit the observation in a supervised learning framework. Then the problem of estimating structure, appearance or motion becomes easier. An imposed model makes assumptions (at varying degrees) about the structure of the objects in the scene, their appearance or the type of motion they exhibit. In certain applications, some such assumptions might be justified. For example, if we know that it is a single human actor that is being observed and that there are no other objects in the scene, a human structural model can be imposed to track the movements. Using such a structural model, motion can be estimated densely and to a high accuracy.

We review the related work on different such approaches for model based performance capture in section 1.4.2. We present them with respect to the nature and the degree of assumptions they make on the scene. But with any of these assumptions, the generality of the scenes that can be addressed shall be compromised. For example, a supervised approach may require that only a single person be present in the scene, or that the person be dressed in tight clothing, or that the motion be restricted to within a predefined set of motions. Such conditions may prove restrictive to the type of scenes that can be analyzed. Especially, when the scene contains multiple actors (as illustrated through examples in section 1.1.2), very few assumptions can be made about its structure or topology. In this thesis, we do not subscribe to any type of external knowledge for motion estimation. As we have argued in earlier sections, the richness of the real world may only be captured through *dynamic models* and no static model may be expected to fit the data perfectly. When we can build and jointly represent all the three elements of a spatio-temporal model, we believe they provide a stronger framework for learning and betterment.

1.1.5 Acquiring Data : Types of 3D Sensors

There exist different types of position and motion sensors to acquire raw data, and help model the dynamics of the scene. We differentiate these sensors into three types based on their level of intrusiveness.

Wearable Sensors : These are the most intrusive sensors. A user, normally a trained human being, can wear an electromagnetic tracking device that transmits the position and orientation information from any body part in a fast and accurate manner. These devices usually have 6 degrees of freedom : X,Y,Z, yaw, pitch and roll. In terms of hardware, they generally require the following three components : a source that generates a signal, a sensor that receives the signal, and a control box that processes the signal and communicates with the computer. Depending on the technology used, either the source or the sensor is attached to the body, with the other placed at a fixed spot in the environment, serving as a reference point. Alternatively, some devices have no transmission capabilities but inbuilt MEMS electronics that record either the orientation (gyroscopes) or speed (accelerometers).

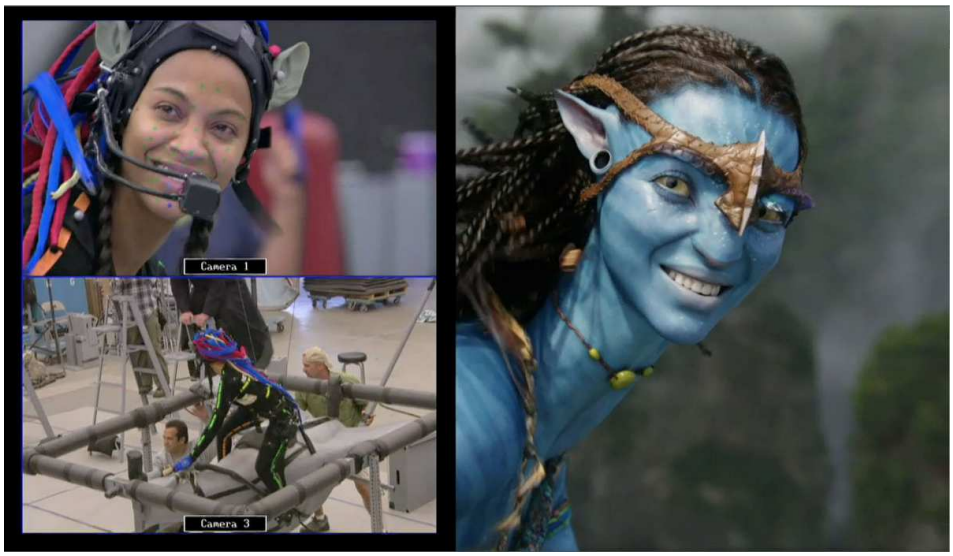


Figure 1.2: The performance capture system developed by Giant Studios for the Hollywood film "Avatar" : optical markers are tracked by a camera placed in front of the actor's face - subtle muscle deformations are transposed to a virtual avatar. Skeletal motion is transferred using a mocap suit. ©Giant Studios & 20th Century Fox

Such devices can be installed in a head mounted display or in a human body suit (also called a motion-capture suit or "mocap suit"). Due to their speed and accuracy, these sensors have conventionally been deployed in the entertainment industry to transfer motion from a human actor to a 3D avatar, but are now being disparaged owing to their cost and bulkiness. Such gadgets are being replaced by optical tracking devices - LEDs are tracked remotely from cameras and their 3D positions are obtained from triangulation. Under good lighting, a user may also wear a specified make-up of dots that are tracked by cameras. For the recent movie *Avatar* (see figure 1.2), a camera is attached to a helmet and positioned in front of the face of an actor to track the facial make-up and obtain a highly accurate performance capture.

Active Sensors : Active sensors do not require the user to wear specific tracking devices or markers. Instead they project light into the scene and read the reflection from the surface of objects through cameras. Triangulation 3D laser scanners (also termed as range-scanning devices) project a laser dot into the scene and obtain the 3D position of the object it strikes by triangulation in the camera's field of view. As the 3D scene needs to be scanned by projecting multiple dots, it is difficult to track fast-moving objects by this method. Another kind of sensors operate by measuring the time of flight of the reflected light to reach back to the camera. They can use laser point sources (known as *LIDAR scanners*) or 2 dimensional light sources emitting modulated light (known as *time-of-flight* or *TOF cameras*). Laser scanners can be used for precise scanning of human bodies to obtain highly detailed surface geometry

Figure 1.3: The Microsoft *Kinect* game controller : the sensor consists of a projector of infrared light patterns and a CMOS array for capture. It senses the 3D depth in real time, and the skeletal dynamics of human actors are modeled through statistical learning ©Microsoft Corporation

[bod 2010a, bod 2010b]. Their accuracy and range of measurements are influenced by the accuracy in keeping the time and by the spectrum of the light used. Time-of-flight devices are similar to acoustic devices (for example, *SONAR*) which emit sound and compute its time-of-flight of return, in a manner similar to how certain animals (particularly bats) sense in 3D. Time-of-flight cameras are a very recent development and have found remarkable commercial success. Finally, the method of structured light sensing projects a light of known pattern, and reconstructs the surface in 3D from the deformation of the light pattern as seen in the captured images. This can be done very fast, and used for tracking highly deformable objects, such as human faces or cloth. The downside of this method is that performance deteriorates under ambient lighting; hence, the subject should be captured in a relatively dark room. An alternative is to use project light patterns beyond the visible spectrum. *Microsoft Kinect* (earlier called *Project Natal*), the new game controller from Microsoft Corporation, uses a structured light system developed by *PrimeSense* that uses infrared light. The use of infrared light makes this system less intrusive and operable in normal ambient lighting conditions.

Passive Sensors : Passive sensors do not emit any light but only read the ambient light as reflected from the surface of objects. Amongst these sensors, stereoscopy

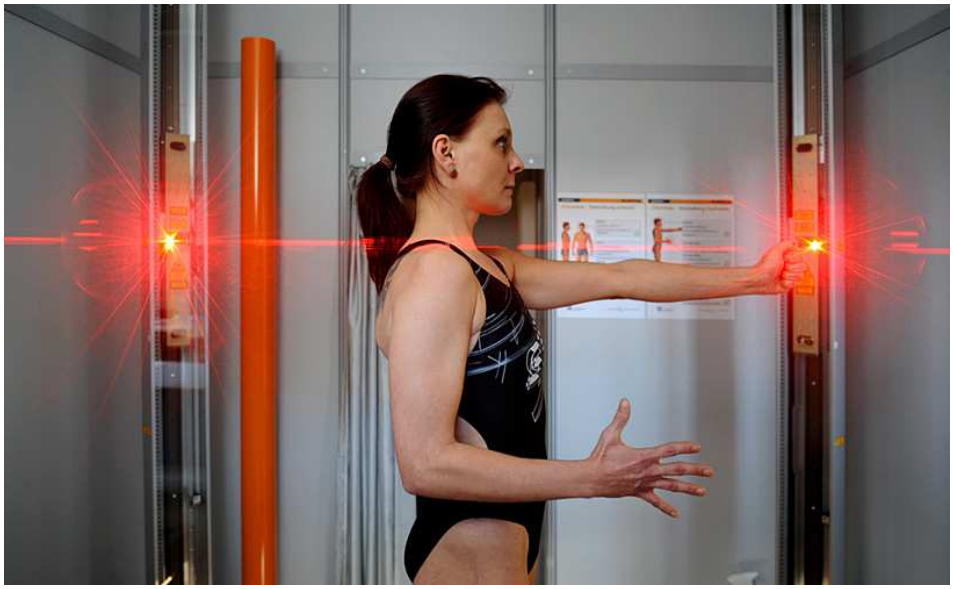
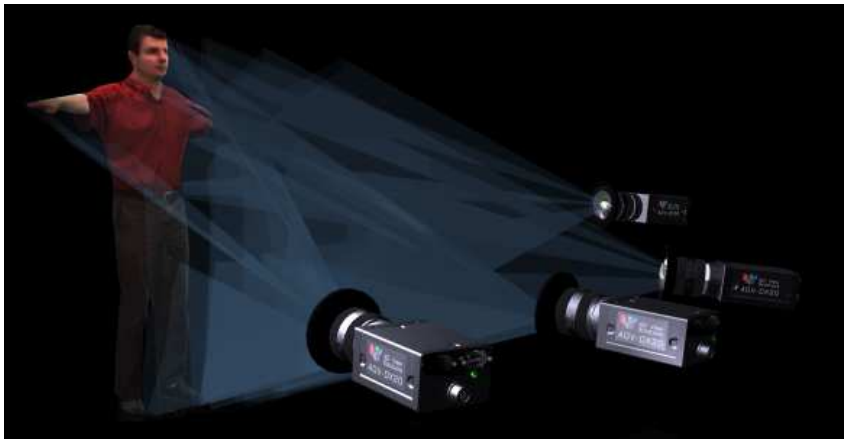


Figure 1.4: A Laser body-scanner : highly detailed surface geometry can be obtained by such scanners. Such detailed geometry is used for modeling textiles, medical analysis or for performance capture in the entertainment industry ©Human Solutions

operates on a principle similar to human vision - by simultaneously imaging from two cameras, estimating correspondences between points and computing depth of a 3D point based on the disparity in its imaged positions. This is a computationally hard problem and normally cannot be used to capture dynamic scenes in real time. Bifocal stereo can be generalized to multiple-views where such geometric information is integrated from several cameras placed around the scene. Instead of using multiple cameras, one can also reconstruct shape by using multiple lights. This method is known as *photometric stereo* - it captures the object at different lighting conditions and tries to estimate the surface orientation. This method has historically been used for capturing static scenes, but a few works have been proposed in the recent past to capture dynamic scenes under controlled lighting conditions, through using coloured lights or time multiplexed lighting (described in greater detail in section 1.4.1). *Visual hull* methods are similar to multi-view stereo, but avoid the correspondence problem by using only the silhouettes of the object. They capture the object silhouettes from multiple views (by subtracting the background) and integrate these silhouettes into a 3D representation. These methods can be quite fast, especially if implemented on a parallel computing system, and can be used to vividly capture dynamic scenes into 3D. The geometric information in such data can also be improved through post-processing by multi-view stereo algorithms. In this thesis, we use data obtained from such an environment. We describe this multi-view capture in detail in section 1.4.1 and compare its relative advantages and disadvantages with respect to other types of sensors. We review the related works for 3D surface reconstruction using such systems in section 1.4.1.



(a)

(b)

Figure 1.5: Passive sensing systems : (a) A multi-camera environment ©4DView technologies (b) A photometric stereo system ©Paul Debevec, University of Southern California

Sensor	Cost	Accuracy	Intrusive	Generic	Outdoor
Electromagnetic Tracking Devices	high	high	high	no	yes
Optical Marker-based Tracking	low	low	high	no	no
Structured Light	medium	medium	high	no	no
Time-of-flight camera	medium	medium	low	low	may be
Multi-Camera Silhouettes	low	low	low	yes	may be
Photometric Stereo	medium - high	high	medium	low	no

Table 1.1: Comparison of different kinds of sensors with respect to various attributes : The Multi-camera environment that we use is good for spatio-temporal modeling of arbitrary scenes in an indoor setting

Comparison of Sensors In table 1.1.5, we compare the different kinds of sensors we listed above. Photometric stereo (through time-multiplexed lighting or through coloured lights) captures the geometry of the scene at the highest resolution. This makes it suitable for applications where such high resolution geometry is necessary; for example, capturing the facial performance of an actor. But controlled lighting limits its applicability towards capturing generic scenes under natural illumination. The multi-camera setup (using silhouettes and possibly wide baseline stereo) has a crucial advantage in this aspect. This setup is good for capturing generic and arbitrary scenes in a non-intrusive manner. One good example is capturing the interactions of multiple human beings in a natural setting.

1.1.6 Potential Applications

Spatio-temporal modeling from visual information has several immediate applications. In the following, we list some of the important ones. Each of them permit different levels of intrusiveness in sensing, and different types of assumptions on the activity in the scene.

3D video : 3D video is a new way of observing real world scenes that permits a user to navigate the scene in 3D and view the action from any arbitrary angle [Carranza *et al.* 2003], in a fashion similar to the interaction in 3D computer games. This should be differentiated from stereoscopic video (sometimes misleadingly also called as 3D video, or as *3D movies* by the film industry) which a person views through an eye-gear of polarizing glasses to appreciate a 3D stereoscopic effect of the rendered scene. True 3D video not only permits such stereoscopic viewing, but also arbitrary navigation in 3D. It is now possible to capture such video from real world

activities, through image based reconstruction methods (reviewed in detail in section 1.4.1). 3D video of sports, dance performances, music concerts etc. have already been a staple for success in the entertainment industry. Such video information from multiple views can be stored in several formats, ranging from 2D image-streams at several views to a sequence of textured 3D meshes. The latter can be considered a type of space-time models (or 4D models). However, there is a lot of redundancy in the storage of such information - both geometry and texture information. This redundancy results in bulky files which take a lot more bandwidth for transmission. Better spatio-temporal models (similar to what are proposed in this thesis) help in achieving better compression without sacrificing the visual quality of the rendering.

Performance evaluation of skilled motion : Swinging a golf club, holding a baseball bat in position, doing a correct *fouetté en tournant* in ballet etc. are all skilled motions. They are learnt by experts through a lot of practice, trial and error. It is possible to evaluate the accuracy of a particular motion by a computer, and provide constructive suggestions for improving the performance of a user. Various wearable sensors such as gyrometers or accelerometers can be strapped onto the user's body to read the motion, but they are intrusive and affect the natural ease of motion. Such equipment may also be too expensive for amateur athletes and dancers to purchase. Spatio-temporal models as built by computer vision algorithms provide a cheap and easy alternative for analyzing such performances - through images taken from normal cameras. In particular, the methods proposed in this thesis are very general, and are applicable even to performances involving multiple persons (for example, two judo martial artists performing an *Obi otoshi* throw). However, generality and accuracy are often conflicting objectives - bootstrapping the motion estimation method for the exact application in question often produces better accuracy.

Monitoring for signs of weakness : A similar application to the above is the evaluation of normal and unskilled motions, for monitoring the signs of muscle weakness in the elderly. Currently, there exists certain methods where sensors are strapped onto the bodies of the patients and their motions read for evaluation by a doctor. But such methods are intrusive and discourage the patients to take the tests oftener. Similar evaluation through spatio-temporal modeling is cheaper and can be deployed in daily situations, where the users are not required to wear tight clothing. Through such non-intrusive sensing, a doctor shall have a lot more data to make the assessment, and dispense the appropriate care for the patient.

Novel ways of human-machine interaction : Interacting through visual information by gestures and body language is one of the most natural and powerful means of communication between human beings. If a computer can visually estimate motion information, that can be read as input from human beings to facilitate several new ways of interaction. A camera can then be used as a versatile input de-

vice to replace other tactile input devices that may not be available in the situation (for example, a cellphone camera can provide a virtual keyboard for typing). But the modes of visual interaction are much higher than those offered by tactile devices. In the future, we can conceive of a dancer automatically playing music through the movement of the limbs of her body. Visual gestures can also be used to animate virtual objects of a similar type - for example, a human being can animate a humanoid avatar. Such modes of interaction are already being realized in the domain of computer games (for example, *Microsoft Kinect*). Spatio-temporal modeling of generic scenes, involving multiple persons, enlarges the scope of applicability of such modes of interaction.

Activity recognition and surveillance : As cameras become cheaper and ubiquitous, it becomes possible that they produce a lot more visual information than can be processed by any set of human beings. Most such visual data would be consumed not by human beings, but by computer algorithms, which detect activities of interest and signal them to human beings. For example, surveillance cameras of today are coupled with computer algorithms that detect suspicious activity in a given and well-defined region, such as a parking area for vehicles. Spatio-temporal models of generic scenes broaden the applicability of such algorithms to generic and outdoor scenes, involving multiple persons in an arbitrary interaction. Such applications shall be geared not only towards surveillance but also towards entertainment. For example, the 3D video of a soccer game can be monitored through a spatio-temporal model, which detects the possibility of a goal and draws the attention of users.

Applications for other visual data : In this thesis, we primarily address scenes of human activities. But there exists several other applications that can benefit from the simultaneous modeling of spatial and temporal information. For example, analysis of medical tomoraphic images of heart, lungs etc. can monitor for the signs of weakness in the organs. Cameras deployed under water at ocean floors can monitor the activity of marine animals, and help throw light on various species and their metabolism. However, it should be remembered that each application needs to represent the visual information in a different manner and that it demands different kinds of trade-offs between the *generality* and *accuracy* in spatio-temporal modeling.

1.2 Modeling by Multi-view Capture

Any type of visual sensor projects the 3D information about the scene onto a 2D plane. Due to visual occlusions, this inevitably results in loss of information, some of which might be vital for interpreting the scene. Human visual system compensates for these occlusions through its active visual capabilities and through the vast amount of visual knowledge that is learnt from childhood. But there is another way to compensate for this loss, and that is through simultaneous imaging from multiple viewing angles. This is achieved by arranging multiple cameras around the scene of

interest and capturing video of the activity. Several algorithms are proposed in the field of computer vision to integrate the visual information from multiple views into a single and coherent three dimensional representation (these methods are reviewed in greater detail in section 1.4.1). These methods require arranging and calibrating a set of cameras, and are applicable for capturing indoor activities. However, when large amounts of data are thus captured, more general spatio-temporal models may be learnt that have a wider applicability - even for outdoor scenes. We now describe the nature of a multi-camera environment, and its relative advantages and disadvantages as compared to other types of visual sensors.

1.2.1 Description of a Multi-Camera Environment

A multi-camera environment is defined by a fixed zone of interest around which various cameras are placed, facing it at different angles. This camera configuration is fixed throughout the capture. The internal and external parameters of calibration of each camera (or equivalently, its projection matrix) are estimated beforehand. This is normally done by waving a rod of LED lights, which can be detected easily in all the captured images. The imaged positions of these lights are used for recovering the calibration parameters.

The background of the scene, as seen by the cameras, is then captured and stored into a computer. This background is later eliminated from the captured images, providing the silhouettes of various objects in the scene. The background elimination is normally helped by setting it to a constant colour (for example, green). This process is known as *chroma-keying*. In figure 1.6, we show the GrImage room at INRIA Rhône-Alpes, France, as an example multi-camera environment.

The various cameras are controlled by a timer which synchronizes the process of taking images. The captured images and their corresponding silhouettes are named with a time-stamp and stored in a set of computers, connected to the cameras by firewire cables. These synchronized multi-view images, taken at several instants over time, are used to build a sequence of textured meshes known as a *raw photographic 4D model* of the captured dynamic scene.

1.2.2 Building of Photographic 4D Models from Multi-view Images

The silhouettes of the object captured from multiple views contain information pertaining to the geometric structure of the object. The *visual hull* of the object is defined as that 3D shape that projects exactly into the silhouettes of the object as taken from the several views. Such a 3D shape can be obtained by producing a *viewing cone* for each silhouette with the corresponding camera center as the apex, and by taking the 3D intersection of all such viewing cones. The visual hull completely encloses the object from all the sides, but it may be bigger at certain places. Specifically, object concavities are lost in the visual hull representation. It helps to place the cameras around the scene in such a way that all possible viewing angles for observing the scene are covered. Self-occlusions will still cause further

(a)

(b)

Figure 1.6: The multi-camera environment : (a) the studio with fixed cameras and background to facilitate chroma-keying (b) the production software for building 3D meshes out of multi-view silhouettes ©4DView Technologies

inaccuracies in the reconstruction.

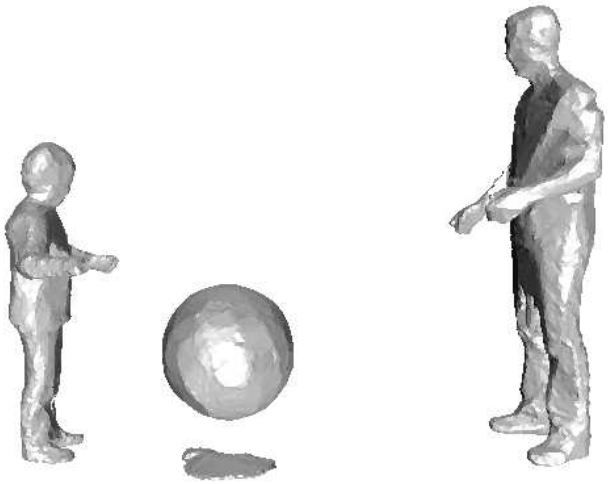
In our system, we represent silhouettes as discrete polygons, which induce polyhedral viewing cones, and thence, a polyhedral visual hull of the object. We use the method of [Franco & Boyer 2003] to efficiently compute this exact polyhedral visual hull (EPVH) of the silhouette polygons in 3D. Example EPVHs reconstructed into 3D meshes are shown in figures 1.7, 1.8, 1.9. An EPVH mesh thus obtained is sensitive not only to the self-occlusions between the objects in the 3D scene, but also to the process of discretization of the silhouettes. Errors in segmenting the object from the image background produce other artifacts. These artifacts of visual reconstruction are detailed in section 1.2.3. It has been observed by [Franco & Boyer 2003] that increasing the number of viewpoints (cameras) need not offer better results in the 3D modeling of the shape - the reconstructed visual hull inherently becomes more sensitive to image calibration and discretization noise. It has been noted that 10 to 20 viewpoints are sufficient to obtain decent results for real world scenes.

This process of the construction of visual hull can be parallelized and performed in a rapid manner [Franco *et al.* 2004]. This makes it possible to capture snapshots of a given dynamic scene as 3D meshes at multiple frames. These 3D meshes are trivially associated with texture information that is obtained from the multi-view images. Such a sequence of textured 3D models is termed as a raw photographic 4D (space + time) model of the scene. The main problem with these reconstructed meshes is that they are not temporally coherent - either with respect to surface geometry or mesh topology. So they cannot be termed as a good spatio-temporal model (as detailed earlier). But they are a crude spatio-temporal model that is sufficiently good for certain applications (for example: 3D video). In this thesis, we aim at building better spatio-temporal models from this raw 4D data.

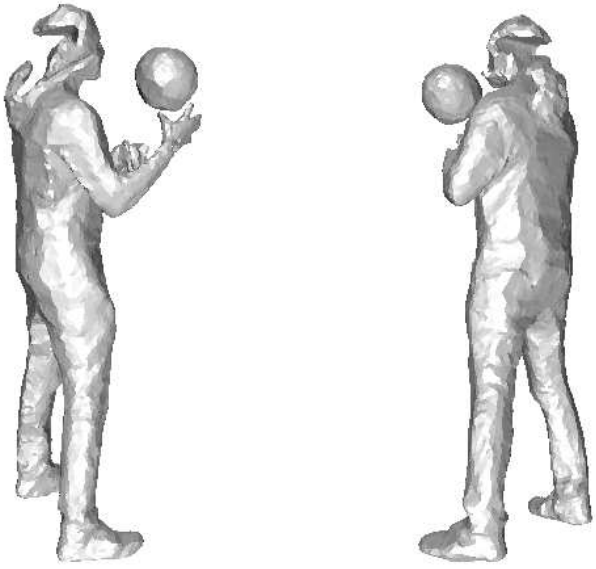
Please note that the visual hull (spatial) reconstruction can be improved within each frame. The first step of improvement is to simplify the mesh by re-triangulating it into equilateral triangles of roughly equal size. Such a regular mesh has several nice surface properties, however it should be noted that it need not project exactly into the all the image silhouettes. In this thesis, we perform such regularization as pre-processing for all the meshes that we take as input. This step is very rapid and doesn't take more than several milliseconds on a computer armed with a 2 G.Hz processor. The second step of improvement is to use multi-view stereo to recover the 3D shape of the object in greater detail, particularly over the concavities in the visual hull. This step, however, is time-prone and needs several minutes to produce a smooth reconstruction of the shape. We call such improved multi-view stereo reconstructions also as raw 4D models : they suffer from artifacts very similar to the EPVHs. We detail those artifacts in the following section.

1.2.3 Reconstruction Artifacts in Photographic Models

There are different causes for geometric and topological inconsistencies in visually reconstructed meshes. Several of these artifacts occur in other types of sensors as well.

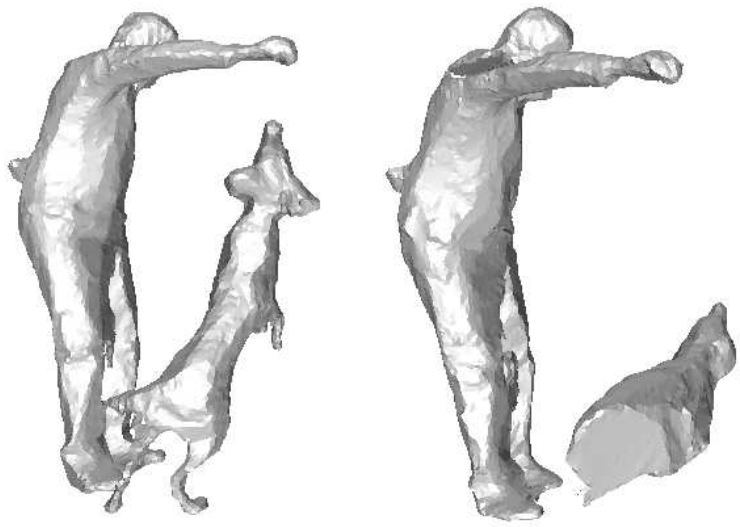


(a)

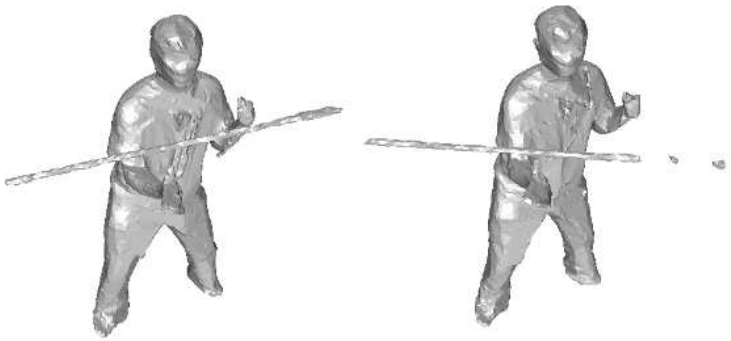


(b)

Figure 1.7: Artifacts in visual reconstruction : (a) inaccurate silhouette extraction creates a phantom 3D blob from the shadow of the ball (b) inaccurate silhouettes produce a poor reconstruction of the head as hair of the person was segmented out



(c)



(d)

Figure 1.8: Artifacts in visual reconstruction (continued) : (c) the dog is not completely imaged in the 2nd frame due to limited field of view (d) the thin stick is reconstructed in pieces due to limited resolution and rapid motion

Errors in Silhouette Extraction : The process of segmenting the object from the background is not perfect, even if it is aided by chroma-keying. This segmentation can be improved by using various other factors - such as optical flow in the images, simultaneous segmentation from all views, learning a prior on background and objects and obtaining the segmentation with the best posterior probability (a process known as Bayesian matting) etc. But none of these approaches can ensure 100% accuracy, and moreover are harder to be implemented in real-time for capturing a dynamic scene. Problems occur particularly due to the shadows of objects, which are hard to be distinguished from genuinely darker objects like a person's hair or clothes. Errors in the silhouette extraction process cause spurious objects to appear (figure 1.7-(a) the shadow of the ball is reconstructed into a 3D blob) or large holes to be introduced into the object (figure 1.7-(b) shows holes in the 3D model of the reconstructed head, as portions of the hair are eliminated as background).

Discretization of Silhouettes : Silhouettes are discretized into polygons independently at each frame, and such discretization may be inconsistent over time. This means the polygonal meshes that are reconstructed at each instant are vastly different from each other with respect to surface information. Also, the surface information is accurate only up to the scale of discretization of silhouettes.

Insufficient View-points : An object concavity is represented into the visual hull only when there exists a view-point from which the concavity is imaged. If there exists no camera placed in such a viewpoint, or if the resolution of the image is not sufficient to capture the concavity, it is not represented in the 3D reconstruction.

The above three problems can be partially rectified if stereoscopy algorithms are used : the appearance of the object (photometric information) can be used to rectify the limitations in object silhouettes. But such algorithms take longer time for reconstruction.

Topological Collapse due to Occlusions : A dynamic object causes several self-occlusions as it is captured by any camera. Sometimes, such a self-occlusion cannot be distinguished from any of the placed cameras, in which case the occluded and occluding objects get merged together in the reconstruction. We call this artifact a *topological collapse* - this is the most serious problem for visual reconstructions as it drastically alters the mesh geometry. Such collapses are more probable as two objects approach each other. In figure 1.9, we show a reconstruction of a man juggling with clubs, and several such artifacts are present : the juggling clubs are merged with the hands, the right hand is pasted to the stomach of the person, and the two feet of the legs get merged into a single blob.

Finite Field of View : Since the multi-camera environment is not an active visual system, objects cannot be tracked as they move along. If an object is partially imaged by any of the cameras, only that part of the object that is visible in the

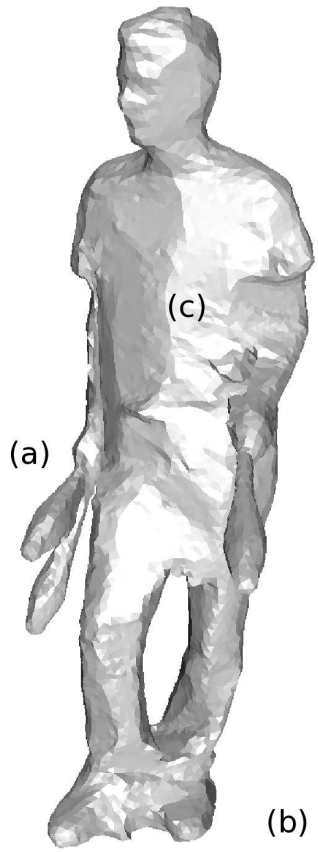


Figure 1.9: Artifacts in visual reconstruction (continued) : A 3D mesh of a juggler holding clubs, reconstructed from multiple view silhouettes. Visible artifacts of reconstruction (a) the clubs merged with the hands (b) the two feet merged together (c) spurious noise on the surface

cameras is reconstructed. This produces the artifact of an object chopped by a plane (figure 1.8-(c) shows how only part of a prancing dog is reconstructed into 3D). In a dynamic scene, several new actors might move into the field of view and several old actors might disappear.

The above two problems can be partially rectified by utilizing temporal information - such as motion estimates. Even crude motion estimate such as optical flow in images may help in correcting the problems. In this thesis, we provide more powerful means of spatio-temporal modeling that accept such challenging data and produce temporally coherent models.

Rapid Motions : The limited frame-rate in the video cannot capture rapid motions of objects. The reconstructions at succeessive frames can be quite distant from each other, and purely local approaches of motion estimation, such as optical flow, would not perform well in such cases. There are also practical limitations on the synchronization between the cameras, and a rapidly moving objects may not be imaged at the same 3D position by all the cameras, in which case the reconstructed shape would suffer from severely altered geometry (figure 1.8-(d) shows how a stick is broken into several pieces as it is being turned rapidly). Such artifacts are commoner for thin objects. An active visual system which can focus on an object of interest and zoom into it, can overcome some of these problems as it identifies and uses the appropriate scale of imaging for a dynamic motion. It may also image important details for capturing motion (for example : the face and hands of an actor) more attentively than the rest of the scene. However, the multi-camera environment which we use does not have such active visual capabilities.

Inaccurate Photometry : The lighting information in the scene need not be constant over time, or even across space as imaged from the different cameras. So we cannot use algorithms which rely on exact photometric correspondence for spatial or temporal reconstruction. Self-shadows of the objects (called as *umbra*), softer self-shadows (called as *penumbra*), specular reflections on the surface etc. alter the photometric information in the captured images. Reconstructing the reflectance of the objects over the changes of lights and over viewing angles is a hard problem, that is still being investigated in computer vision. Transparent and translucent objects pose more severe problems towards the modeling of shape and appearance.

1.2.4 Spatio-Temporal Information in Photographic Models

Despite the artifacts we described so far, a raw model captured in a multi-camera environment has four kinds of information that can be analyzed to build a better spatio-temporal model of the scene : the first three pertain to space and the last pertains to time.

Photometric information The visual appearance of an object is imaged into several cameras at multiple frames, and captured as RGB colours of the images.

Surface information This can also be termed as *local geometry*. Properties such as local surface curvature at various points are captured into the 3D representation of the mesh.

Volumetric information This can also be termed as *non-local geometry*. The gross 3D structure of the scene : the number of disjoint objects in the scene, the protrusions of the shapes, the thickness of the shape at different places etc. can be computed at each frame of the reconstructed scene.

Temporal locality The reconstructed meshes at successive frames are related as temporally being close, and thus their 3D reconstructions are not very far from each other.

In a raw photographic 4D model, all these types of information are present in a very redundant manner. The size (in memory storage) of a raw model is primarily determined by the captured images, which may be very bulky owing to this redundancy. Various methods can be used for compressing this information, so that it can be transmitted easily across a network and visualized at distance. A spatio-temporal model can be viewed as a method for compressing such data. An ideal spatio-temporal model (as discussed in section 1.1.3) achieves the best compression and visualization capabilities. The methods that we present in this thesis should be understood as steps towards this end, on starting from the raw models as input.

1.3 Major Contributions : Thesis Outline

In this thesis, we make four principal contributions towards the spatio-temporal modeling of arbitrary scenes. They are presented in the following chapters.

Chapter 2 We present a method for segmenting a visually reconstructed mesh into a set of approximately convex segments. We review in detail how existing methods for mesh segmentation are not suited for handling visual reconstructions, due to the presence of various artifacts. The segments that we detect roughly identify the body-parts of the actors and are used in later chapters. In this step, we exploit only the volumetric information in the scene.

Chapter 3 We present a method for segmenting a sequence of independent visual reconstructions in a temporally coherent manner. This provides a crude spatio-temporal model as a set of objects possessing temporally consistent structure, and rough motion estimates for each object. We achieve this by tracking the various convex segments, and refining their shape through this. We obtain a set of segments that best explains the whole sequence of observation. In this step, we exploit the volumetric and temporal information in the scene.

Chapter 4 We present a method to detect features on visual reconstructions and match them across time. We present two types of features on the surface - local

features in small neighborhoods on the mesh, and global features which take the whole shape into consideration. We provide mathematical background for feature detection through scale-space theory, and provide a method to derive geometrically consistent feature matches between two shapes. In this step, we exploit all the types of spatial and temporal information in the scene.

Chapter 5 We present a method for estimating dense motion on a temporally varying shape, without assuming a given mesh topology or geometry. We first obtain partial surface alignment between shapes through the sparse feature matches obtained in the earlier chapter. We then present a method for aligning the two independent meshes in an exact manner and evolving mesh topology to connect the two shapes. We thus present a method to not only handle the various topological artifacts in visual reconstruction, but in principle to also capture a dynamic activity with topologically varying shapes. In this step, we exploit all the spatial and temporal information available in the scene.

Our contributions are towards identifying temporally consistent structure (with respect to the parts that compose the scene) and spatially consistent dynamics (both part-based motion and dense motion on the surface).

Important related works concerning spatial and temporal reconstruction are described in the following section. We review additional related work, wherever appropriate, in each of the chapters to relate to the contributions. Finally in the concluding chapter 6, we relate how each of these contributions fit into the broader goal of spatio-temporal modeling of dynamic scenes, and outline avenues for future work.

1.4 Related Work

The past works in computer vision have mostly treated the spatial and temporal reconstructions of the scene independently. By spatial reconstruction, we mean the estimation of 3D structure from visual information captured from one or more camera views. By temporal reconstruction, we mean the estimation of 3D motion in the scene, by relying on various assumptions about the underlying structure of the shape. Each of these problems have been investigated extensively in the past, but are also very alive due to the difficulty of the challenges faced. We briefly review these works in sections 1.4.1 and 1.4.2 respectively. In comparison to these approaches, fewer works exist on joint and unsupervised spatio-temporal reconstruction (or spatio-temporal modeling) of the scene. We review them in section 1.4.3. Our approach falls into this final category.

Apart from the works reviewed here, we review concerned related works on subtopics attempted in the thesis (mesh segmentation, feature detection and matching etc.) in the respective chapters themselves.

1.4.1 Spatial Reconstruction / 3D Video

The estimation of 3D depth from calibrated or uncalibrated images is one of the fundamental problems of computer vision. Various methodologies exist for solving these problems, and they are collectively known as *shape-from-X* ('X' can be stereo, shading, texture, focus, defocus, silhouettes etc.) An extensive review of these methods is beyond the scope of this thesis, the reader is advised to consult a book such as [Szeliski 2010] (chapters 12, 7).

In the recent past, these diverse methodologies have been combined to realize the novel application of *3D video* or *free viewpoint video*. This application permits the user to navigate arbitrarily into the scene and visualize it from arbitrary viewpoints. One of the first works to appear in this topic was by [Kanade & Narayanan 2007] - called *virtualized reality*. In this work, multiple cameras placed around the scene are used to perform dense stereo reconstruction [Narayanan *et al.* 1998].

Volumetric information, as available from image silhouettes, can substitute for stereo based depth estimation [Davis 2001]. The object silhouette in a calibrated image defines a generalized 3D cone within which the object must lie. The intersection of these generalized cones associated with a set of cameras defines the volume of the scene that encloses the object. [Laurentini 1994] proposes the *visual hull* as the best such approximation afforded by placing an infinite number of viewpoints outside the convex hull of the object. This visual hull of a 2D scene coincides with its convex hull. However for a 3D scene, its visual hull is contained within the convex hull, where concavities are not removed but hyperbolic regions are. In practice, only a finite number of cameras can be placed around the scene and only an approximate visual hull be computed. [Matusik *et al.* 2000] propose a method for efficiently computing this approximation by taking advantage of the epipolar geometry between the camera views. [Franco & Boyer 2003] propose a method to compute the exact visual hull as a polyhedral mesh by considering the image silhouettes as discrete contours. These silhouette based methods are generally very fast and can reconstruct a dynamic scene in real time. The principal advantage of these methods (especially the ones which produce a polyhedral mesh as output) is that texture information is readily available from the captured images and can be used for rendering the scene realistically from an arbitrary viewpoint. However, the downside is that the reconstructed geometry is not of high quality.

Other shape cues such as stereo, shading etc. can be used to improve the reconstructed geometry. [Furukawa & Ponce 2006] carve the visual hulls by using stereo information between the views to obtain reconstructions of high resolution. [Hernandez & Schmitt 2004] provide another alternative for 3D object modeling by fusing silhouette and stereo information. [Pons *et al.* 2007b] propose the multi-view stereo reprojection error for estimating the 3D shape of the scene, a potential initialization for this reconstruction can be obtained from the visual hulls. These methods operate by iteratively refining the mesh reconstruction until the error falls within a bound. Such evolution often leads to topological changes and they need to be addressed properly. One option for this evolution is by refining a higher dimen-

sional (4D) object and reconstructing the scene as the zero-level set of this object at every iteration. Such methods are known as Lagrangian methods - they suffer from a higher computational cost associated with maintaining the higher dimensional object. Eulerian methods provide another alternative for this evolution, by working directly with the 3D scene and by detecting and addressing the topological changes as they happen. These methods are often faster and more efficient. [Zaharescu *et al.* 2007] propose *TransforMesh*, a mesh-based approach for scene refinement and evolution. [Pons *et al.* 2007a] propose another alternative through Delaunay re-triangulation of a given set of points. These mesh-evolution methods can incorporate various types of information - [Delaunoy *et al.* 2008] propose minimizing the stereo reprojection error. [Starck & Hilton 2007b] propose blending a variety of cues - stereo, silhouettes and edge information. [Labatut *et al.* 2007] propose a method which does not require image silhouettes (visual hull) for initialization of the shape. Instead, they match keypoints across frames and generate a quasi dense point cloud by optimizing a photo-consistency metric; the point cloud is remeshed at each instant by adaptive 3D Delaunay triangulation.

One persistent problem with stereo based surface reconstruction is on handling regions with no significant texture variations, and over which point correspondences cannot be estimated reliably. Structured light sensing overcomes this problem by projecting coded light patterns into the scene and imaging them from a single camera [Batlle *et al.* 1998]. Dynamic scnes can be captured using this method by projecting a simple grid pattern [Kawasaki *et al.* 2008], or dynamic adaptive patterns [Konickx & Van Gool 2006]. Stereo information (by imaging from multiple cameras) can be used to improve the reconstructions [Weise *et al.* 2007].

One of the most efficient ways to capture surface geometry at high detail is through photometric stereo [Woodham 1980] [Ikeuchi & Horn 1981] [Horn 1986]. In this approach, a 3D object is imaged under multiple *known* lighting conditions. The orientations of the surface normals at various points can be recovered by comparing the images obtained. On a smooth and continuous surface, these surface normals can be integrated to recover the 3D shape of the object. Since the surface normals are captured at fine detail, the 3D shape can be constructed at high resolution. In the recent past, the photometric stereo method has been used with multiple coloured lights to reconstruct a dynamic 3D shape at a fast frame rate [Hernandez *et al.* 2007]. This method has been extended to multi-camera systems in [Hernandez-Esteban *et al.* 2008]. [Wenger *et al.* 2005] propose another alternative for capturing dynamic 3D scenes through photometric stereo, by using a *light stage* (an arrangement of LED lights around the object) and rapidly alternating the illumination conditions by time-multiplexing. This method not only captures the 3D shape and surface normals, but also the surface reflectance in terms of the Bi-directional Reflectance Distribution Function (BRDF). This enables relighting of the object under an illumination different from the one it is captured in. Photometric stereo methods have a promising future, but key challenges remain towards capturing scenes under a natural (instead of controlled) lighting.

1.4.2 Temporal Reconstruction / Performance Capture

The 3D reconstructions obtained using the methods described in the earlier section are not temporally consistent. Each individual reconstruction is partially inaccurate and these errors are not temporally consistent either. The successive frames in a 3D video exhibit flutter which is disturbing to the eye. This is particularly apparent when only the geometry is visualized without the associated texture information. Further, no motion information is yet captured that connects the reconstructions. In this section, we study various approaches for recovering such motion information that assume an underlying model. Several approaches have been proposed in computer vision on model based tracking. This model can be locally rigid [Gavrila & Davis 1996, Kakadiaris & Metaxas 2000, Carranza *et al.* 2003], or deformable [DeCarlo & Metaxas 2000, Salzmann *et al.* 2007]. [Montagnat *et al.* 2000] provides a review on 3D deformable models.

The model is deformed and made to fit to the data available in the new frame, either to the reconstruction obtained or directly to the captured images and silhouettes. Such methods recover motion information, and by construction, produce temporally consistent reconstructions. The recovered motion can be used to animate a virtual avatar through skeletal motion retargetting [Gleicher 1998] or through surface deformation transfer [Sumner & Popovic 2004]. These methods have come to be known as *performance capture*. The assumed model for recovering motion can either be given by hand, or captured entirely from reality. We discuss both these cases in this section, and term these methods as *temporal reconstruction* methods, as they decouple spatial reconstruction and treat it as an independent problem.

The structure of a scene can be specified at four different levels, with increasing level of detail.

1. topology of the shape
2. skeletal structure of the shape
3. surface geometry of the shape
4. statistically augmented motion model

In fact, the richer the description of the model along the four levels, the greater the detail at which motion can be estimated on the shape. But also greater is the compromise on the generality of the scenes that can be addressed by the model.

Topology : If the object under consideration is known to be topologically equivalent to a sphere (genus-0), its motion in 3D can be restricted to a 2D diffeomorphism on a sphere [Starck & Hilton 2005]. This immensely reduces the complexity of motion estimation. Similar assumptions can be made for objects of higher genus, if we know the topology of the shape to be conserved over time.

Skeleton : If we know the object under consideration to be a human being, the motion can be estimated within the space of articulation of the limbs [Knossow *et al.* 2008]. This estimates motion at a coarse level, and can serve as an initialization for a denser estimation of motion. [Vlasic *et al.* 2008] use a hand-given skeleton to deform the model so that it fits multi-view silhouettes.

Surface geometry : If we know exactly the surface that is conserved over time, its motion can be estimated so as to conserve these surface properties. For example, geodesic distances on the surface shall be preserved when the motion is estimated within the space of isometric deformations of the surface [Mateus *et al.* 2008]. Detailed surface geometry for a human actor may be obtained by scanning the subject in a precise configuration using an accurate laser sensor [de Aguiar *et al.* 2007b]. The same for the face of a person can be obtained by using reflectance data from photometric stereo [Bickel *et al.* 2007]. Alternatively, one of the spatial reconstructions at a frame can be chosen as the model [Vlasic *et al.* 2008] [Cagniart *et al.* 2009], but this captures the shape at lesser detail. [Cagniart *et al.* 2010a] parametrize the surface motion along a hierarchy of surface patches, whose estimation they later expressed in a probabilistic framework [Cagniart *et al.* 2010b]. This method generalizes well to estimating motion in a complex scene with multiple actors, provided that there exists a frame where the geometry of all the actors and their limbs is well-separated and accurate. On the other hand, using a laser-scanned mesh as a template needs to be initiated for each actor separately but can produce accurate and high-resolution capture of peroformance [de Aguiar *et al.* 2008a]. Motion on the surface geometry can also be parametrized adaptively, with higher sampling on textured regions where feature matches can be obtained [Ahmed *et al.* 2008]. [Li *et al.* 2009] provide another method for obtaining a template model by registering multiple 3D scans of an object obtained from structured light sensing. The object is assumed to be approximately rigid as these scans are being taken (for example, moving a stiff face around in front of the sensor), though their system is robust to minor non-rigid deformations in this step. The constructed template model is later used to estimate generic non-rigid deformations. In principle, this kind of methods are capable of reconstructing fine performance details such as modeling the wrinkles on the skin or folds on the cloth, but it is more efficient to model such detail separately [Popa *et al.* 2009]. In a similar spirit, [Bradley *et al.* 2010] capture detailed facial performance by deforming a mesh model of the face (reconstructed from stereo at one of the frames) and overlaying wrinkles that are tracked along each frame.

Statistically augmented motion model : Instead of scanning the exact human actor, the observation can be made to fit within a statistical model of shape and pose that is learnt over a large number of human actors in various poses [Anguelov *et al.* 2005]. Such complex and rich statistical models can be learnt not only for structure but also for motion (pose of the subject). The richness of this

information helps them tackle more complex problems such as identifying the shape and pose of a partially visible person from a single image [Balan & Black 2008]. Motion pose estimation can be posed in a probabilistic framework [Ukita *et al.* 2009]. For a specific set of objects, say human faces, very elaborate models can be learnt and detailed motion estimates be made in a quick manner [Li *et al.* 2010]. The estimated deformation can be transfered across to an avatar in almost real-time [Weise *et al.* 2009].

1.4.3 Unsupervised Spatio-Temporal Modeling

Estimating structure and motion from the scene simultaneously is a complex task, and relatively fewer works have been published in this area. Most works on spatial reconstruction (section 1.4.1) can be augmented with optical flow computations from images. Instead of estimating optical flow on the image plane, 3D scene flow [Vedula *et al.* 2005b] [Neumann & Aloimonos 2002] can be estimated from multiple images simultaneously. These flow vectors are generally short, and connect reconstructions in neighboring frames with each other. But they provide the means to interpolate the reconstruction along the time axis [Vedula *et al.* 2002] [Vedula *et al.* 2005a]. [Pons *et al.* 2007b] use multi-view stereo reprojection error to simultaneously estimate spatial reconstructions and 3D scene flow. [Vlasic *et al.* 2009] use optical flow within the spatial reconstruction step at a single frame, for aligning images taken under time-multiplexed illumination to compensate for the motion of the subject. Flow based approaches are generally limited to recovering small displacements (as noticed in [Starck & Hilton 2007b]). This makes them unsuitable when the subject motion is large, or when the captured frame-rate is not fast enough.

For generic non-rigid motion estimation, a few approaches have considered it as global point corresondence between the reconstructed meshes. [Anguelov *et al.* 2004a] have treated this as a graph-labeling problem (by considering the entire set of target mesh vertices as potential labels for each vertex in the source mesh graph) and proposed a solution based on loopy belief propogation. They have considered the matching score of purely geometric features (spin images of [Johnson & Hebert 1999]) for assigning local potential, and the elastic stretch and twist between points for assigning pair-wise potential between nodes. They demonstrated results on laser scan data. A similar approach is proposed by [Starck & Hilton 2007a] for registering meshes reconstructed from images where they experimented with a more diverse set of features, including photometric features. The disadvantage of these approaches is that they take more time for computation. RANSAC like approaches estimate a subset of point matches which are geometrically consistent and use them to derive more matches. [Tevs *et al.* 2009] estimate a subset of point matches between which the intrinsic (geodesic) distances are preserved across the scans. Our approach is similar in spirit. The point correspondence problem is discussed in greater detail in chapter 4.

Motion information, when estimated in an unsupervised manner, can be used to derive a structural model that connects the individual reconstructions. [Anguelov *et al.* 2004b]

estimate the underlying skeleton of the shape, by using the point correspondences from [Anguelov *et al.* 2004a]. [Wand *et al.* 2009] reconstruct simultaneously shape and motion by aligning structured light data. They estimate a so-called *urshape* (the shape for cross-parametrization of scans) as a set of points between which the topological edges can be repeatedly detected over all the scans. The individual scans are aligned to deformations of the urshape. They show results in aligning scans of a hand over non-rigid deformations, their method fails in the extreme case of the hand closing into a fist where the requirement for a template model becomes apparent ([Li *et al.* 2009] use an external template model, and successfully register such problem cases). [Popa *et al.* 2010] hierarchically assemble the urshape by considering neighboring frames and topologically connecting surface patches that are adjacent. This method of bottom-up assembly is claimed to have an advantage over [Wand *et al.* 2009] that the approxmiate surface urshape does not contain high frequency detail that does not recur in all the frames ([Popa *et al.* 2010] : figure 3).

[Courchay *et al.* 2009] simultaneously reconstruct shape and motion directly from multi-view images. They adopt a variational approach that models photo-consistency across views, spatial smoothness of the surface and temporal smoothness of the motion. This yields a complex optimization problem that is solved by a multi-resolution scheme. They show good reconstruction results for richly textured surfaces. [Aganj *et al.* 2009] target the estimation of a 4D object - a *hyper surface* that encodes both shape and motion from similar input data. This is seeked as the global optimum of an energy minimization scheme. The 4D object can be sliced across time to obtain individual 3D meshes - the motion information connecting them is smooth, but the meshes themselves are not ensured to be of the same topology.

Unsupervised spatio-temporal modeling is still an open problem, considering its novelty and relative lack of literature for cross-comparison. The latter chapters of this thesis explain our contributions towards solving certain sub-problems in this direction. But we would not have yet solved completely the problem of obtaining the ideal spatio-temporal model (section 1.1.3) - modular, highly detailed and temporally consistent along both surface geometry and appearance. We discuss prospects for future work to this end in the concluding chapter 6 of the thesis.

Chapter 2

Shape Segmentation by Visibility

Contents

In this chapter, we describe a method for segmenting a visually reconstructed 3D scene into a set of approximately convex segments. Visually reconstructed meshes, as detailed in the earlier chapter, exhibit severe artifacts due to silhouette extraction and visual occlusion that present challenges to existing approaches of mesh segmentation. We review a few related works from this perspective, and motivate the need for a new approach.

Our method is based on the computation of the *visible volume*, *i.e,* the portion of the mesh volume that is visible (without getting occluded) from an interior point. We build the notion of visibility on the basis of the well-known medial axis transform (MAT), and show how it is connected to the convex decomposition of the shape. We present algorithms for computing the *visible volume* from a given interior point (termed a *viewpoint*), and for decomposing it into a set of approximately convex segments. These algorithms are applied from multiple *viewpoints* to segment the

entire 3D scene. It can be argued that convex segments identify rigid body-parts in the scene that are useful for analyzing motion and dynamics. The later chapters of the thesis exploit this connection, and widely borrow from the terminology and concepts introduced in this chapter. A portion of the work presented in this chapter is accessible through our publication [Varanasi & Boyer 2010].

2.1 Introduction

Polygonal meshes can be rendered easily from arbitrary views. This makes them a very useful and popular representation in computer graphics. Traditionally, such meshes were hand-modeled by artists; they were smooth and composed of well-defined components. Recently, several alternative methods have come into vogue for obtaining such mesh representations by using 3D sensing devices (also known as *range scanning* devices). This has necessitated the development of algorithms to segment such meshes into constituent parts (a process known as *mesh segmentation* or *mesh partitioning*). These algorithms have several applications - mesh reparametrization, compression, multi-resolution modeling, texture mapping, mesh editing, shape matching, retrieval, morphing etc. Amongst these key applications is also mesh animation. Traditionally, animation is performed using bone skeletons that were additionally supplied by artists and embedded into the meshes (through a process known as *rigging*). If the mesh is segmented into constituent parts, the process of fitting such skeletons becomes easier and robust to surface noise. Also, bounding boxes defined around the body parts can assist in fast calculations of collision detection during animation. Thus segmentation is a crucial element in making a 3D mesh come alive in motion. This is not surprising, as there is some evidence from neurological studies [Marr 1977] [Siddiqi & Kimia 1995] [Biederman 1987] that decomposing a shape into its constituent parts plays a strong role in human visual cognition.

In this thesis, we are concerned with 3D meshes obtained by passive 3D sensing, particularly those which are reconstructed from multiple-view images using computer vision algorithms. As argued in chapter 1, such reconstructions have the advantage of being true to reality at every instant, and whose texture information is mapped trivially to the reconstructed geometry. In this context, mesh segmentation is useful towards understanding the temporally consistent shape behind a dynamic scene (explored in chapter 3). Visually reconstructed meshes present certain unique challenges to existing algorithms on mesh segmentation, because of their artifacts (as discussed in section 1.2.3).

Despite these artifacts, such meshes can be segmented into geometrically meaningful components. Convexity is a particularly interesting shape attribute, because the probability of two different body-parts getting together into a single convex segment is usually low. For example, the upper and lower arms can be merged into a single convex component, but only when the arm is stretched. Convexity is also relatively robust to topological artifacts. For example, in figure 1.9, the juggling

clubs are merged together with the juggler's hand in the reconstructed mesh. But they can still be separated out as distinct convex segments. So can also be separated the two legs of the juggler, whose feet were merged together into a single blob in the reconstructed mesh. This robustness ought to be contrasted with the sensitivity of other shape attributes to topological errors. For example, geodesic distances between the mesh points are altered drastically in such events.

In view of these properties, we attempt to decompose a given mesh into approximately convex segments. Since the surface of a reconstructed mesh suffers from a lot of geometric noise, we do not rely heavily on surface properties such as mesh curvature to achieve our goal. Instead, we analyze the interior of the shape through the notion of visibility, which we introduce and elaborate in this chapter. The rest of the chapter is ordered as follows. In section 2.2, we review the related work on mesh segmentation from the perspective of our application domain. In section 2.3, we provide the theoretical formulation of our problem. We introduce the concept of visibility and discuss its properties in continuous and discrete settings. In section 2.4, we provide a practical algorithm based on this background, which can handle the noisy input data of visually reconstructed meshes. In section 2.6, we provide segmentation results as qualitative validation of our approach.

2.2 Related Work on Mesh Segmentation

Mesh segmentation is a well studied problem in computer graphics. [Shamir 2008] provides a good overview of various existing approaches. In this survey, it is argued that most existing segmentation methods can be broadly divided into two classes based on their objectives : *surface-type* segmentation methods which possess a 2D surface view and produce clusters of surface patches, and *part-type* segmentation methods which possess a 3D volumetric view and partition the mesh into semantically meaningful components. Since our objective is clearly the latter, we review related work in this class of methods.

2.2.1 Approximate Convex Decomposition

The work that is most closely related to our objective is that of [Lien & Amanto 2007] on the approximate convex decomposition of 3D polyhedra. Exact convex decomposition has been investigated extensively in computational geometry [Chazelle *et al.* 1995] and proved to be a NP-hard problem in the generic case. [Lien & Amanto 2007] explain how an approximate decomposition is more desirable for many applications, especially on noisy and bumpy meshes. Their algorithm is based on a semi-local metric of concavity illustrated in figure 2.1 for the simpler 2D case. The SL-concavity (or straight line concavity) computes the distance of a surface point to its closest point on the visual hull of the shape. As illustrated in figure 2.1-(b), this metric fails to identify the actual pocket of concavity in certain cases. For this reason, another metric called SP-concavity (or shortest path concavity) is proposed, which maps a pocket of concavity with its corresponding bridge on the convex hull. The depth of

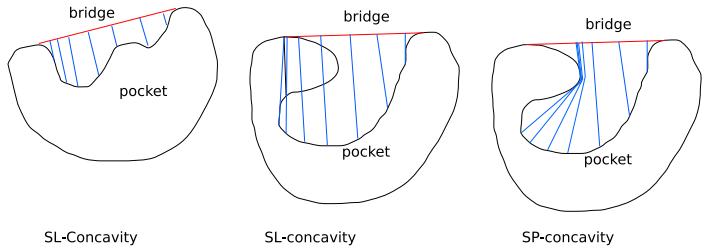


Figure 2.1: (a) SL-concavity measures the length of the straight line joining a point in the pocket of a concavity to its associated bridge on the convex hull (b) SL-concavity is inaccurate when the pocket has bumps (c) SP-concavity computes the shortest path exterior to the surface that connects a point in the pocket of a concavity to its associated bridge on the convex hull, which overcomes this problem

concavity is obtained by the shortest path in the exterior of the mesh volume that connects the pocket to its bridge. The SP-concavity is more accurate but is harder to compute, so in practice, a hybrid measure is adopted by [Lien & Amanto 2007]. Semi-local metrics like these partially account for the global structure of the shape, and hence are more effective than simple local approximations of concavity as obtained by surface curvature. Approximate convex decomposition is driven by the detection of concavities that are deeper than a user-given threshold ε.

Though SP-concavity is a semi-local metric and partially considers the global structure of the shape, it is very sensitive to local surface noise. As illustrated in figure 2.2, it identifies a very non-intuitive point as the depth of the concavity, and thus yields a segmentation that is quite different from what we intuitively seek. We would like to formalize this intuition in a more precise manner. The local surface curvature should be deemed significant only when it is higher than the user-given threshold ε; in other words, the concavities should be detected only after the surface mesh is smoothed by a given factor ε. The metric of SP-concavity as formalized in [Lien & Amanto 2007] does not take this into account. Further, their 2D analogy also doesn't extend easily to 3D, where there exists no unique mapping between pockets of concavity and bridges on the convex hull. In fact, the problem of associating bridges to pockets in 3D is related to the problem of spherical reparametrization of the mesh, which itself is ambiguous for surfaces of genus > 1. So they measure 3D surface curvature using local metrics (the discrete approximation of curvature is obtained from the vertex neighborhood on the mesh) and then spatially cluster the detected pockets of concavity. Purely local metrics like this are unfortunately sensitive to noise on the surface.

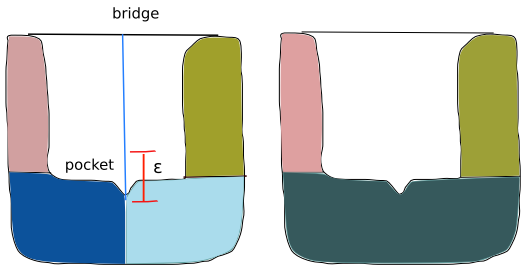
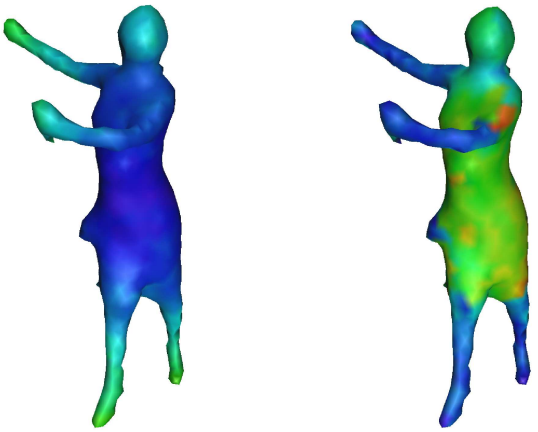


Figure 2.2: The tiny bump in the shape shall be detected as a concavity $> \varepsilon$ while using the shortest-path to convex hull as a measure for concavity. It results in (a) non-intuitive decomposition of the shape. The intuitively correct decomposition is shown in (b)

2.2.2 Spectral Clustering Methods

Spectral graph theory can be used to analyze the mesh structure in terms of graph relationships. Geometrical attributes such as surface curvature can be effectively encoded in terms of edge weights in a graph. For example, we can define a graph such that each mesh facet forms a node and neighboring facets are connected by an edge whose weight is proportional to the dihedral angle between the facets. Alternatively, mesh vertices can be considered as graph nodes and edge weights between adjacent vertices be given in terms of cotangent weights on the mesh. The adjacency matrix of this graph (known as the *Laplacian* matrix) can be analyzed and its Eigen vectors be found. When the surface points of a given mesh are projected into a new space defined by these Eigen vectors as Euclidean basis vectors, the resultant set of points are said to be a *spectral embedding* of the given mesh. Many methods have been proposed that cluster points in this spectral embedding space, and obtain a decomposition of the surface into semantically meaningful parts. One simple approach is to perform k-means clustering in the embedding space with k random seed-points for clustering. These clusters identify the different parts that constitute the shape. What is interesting about these methods is that the 3D volumetric information of the mesh is not explicitly considered, but implicitly encoded in the graph relationships on the surface. The Eigen basis vectors in the embedding space automatically capture the volumetric nature, as long as the surface is closed (the mesh is water-tight).

Two spectral methods are particularly worth mention : [Katz *et al.* 2005] explicitly detect the protrusions (termed by them as feature points) and the core of the shape and use them as initialization for segmentation, [Shapira *et al.* 2008] encode volumetric information using the object thickness at a point (termed the *shape diameter* function, shown in figure 2.3-b) and extract segments with similar thickness. Both of these methods define a graph based on dihedral angles between mesh facets and cluster the facets into segments.



(a) (b)

Figure 2.3: Surface features which identify global shape structure (a) Geodesic eccentricity (b) Shape diameter function. Warmer colors mean higher values in both the figures.

The main problem with using spectral methods is that they are very sensitive to topological artifacts such as those depicted in figure 1.9. These methods are also susceptible to surface geometric noise if they rely on surface attributes such as dihedral angles. Also, it is not straightforward to guess the number of segments beforehand (for example, if we use k-means clustering in the embedding space). [Katz *et al.* 2005] overcome this problem by explicitly detecting protrusions, but their method fails on surfaces of genus > 0. Further, spectral methods are not directly suited to the purpose of extracting convex segments. For example, the two shapes in figure 2.4 are treated similarly by these methods since they possess approximately the same surface perimeter. Extracting such curved segments might be desirable in certain cases (for example, identifying the neck of a camel as a single segment). But, this property is not suited for our problem, which is the analysis of visual reconstructions of inherently articulated shapes (as in figure 1.9).

2.2.3 Segmentation from Deformation Modes

[Huang *et al.* 2009] provide a good alternative to surface metrics such as dihedral angles for the spectral analysis of a mesh. They use the as-rigid-as-possible deformation energy of [Sorkine & Alexa 2007] to compute the non-trivial deformation modes of a shape, through taking the Hessian of the deformation energy. These vibration modes are treated as the basis vectors of a new embedding space, over which surface vertices are projected and clustered. The advantage of this approach is that it is less sensitive to surface noise than earlier methods, but shares other limitations such as the sensitivity to topological noise.

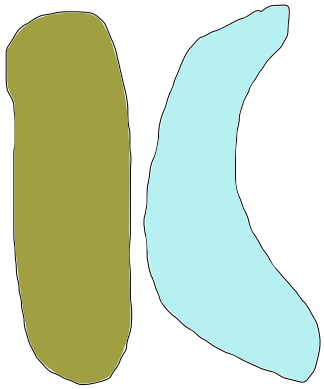


Figure 2.4: The two shapes are treated similarly by spectral methods, even though one is more curved than the other

2.2.4 Segmentation from Visibility Difference

The approach that is most similar in spirit to ours is that of [Liu *et al.* 2009]. They too compute visible volumes from interior points in the mesh. They construct a graph where the edge-weights between neighboring interior points are given by the difference in their visibility. [Liu *et al.* 2009] argue that visibility doesn't change much within a body-part, but changes drastically across body-parts (illustrated in figure 2.5 - a,b). Since the computations are done over the mesh volume, this measure is not as sensitive to noise on the surface. For deriving a final segmentation, [Liu *et al.* 2009] construct a graph of mesh-facets akin to earlier methods, but blend visibility difference with other metrics such as dihedral angles between mesh facets. This helps them derive better segmentations than the other methods.

However, we would like to identify two limitations of their approach. The first is computational, it takes substantial amount of time to estimate visible volumes from all the interior points. This is highlighted in the extreme case of when the object is composed of a single large segment; visible volumes shall be computed from every interior point, only to confirm that they are all identical. The second problem is the inability of their metric to capture, by itself, small and subtle changes in visibility when the segment boundaries are not crisp, as illustrated in figure 2.5. This is the reason why the authors blend it with other measures such as dihedral angles (but which are prone to surface noise disturbances, as discussed earlier). We overcome these limitations by a more elaborate development of the concept of visibility (section 2.3). We base it on the concept the λ-medial axis [Chazal & Lieuter 2005] that is theoretically proven to be stable to noise.

2.3 Theoretical Formulation of Visibility

In this section, we provide the theoretical basis for our work. In section 2.3.1, we introduce certain formal definitions as background. In section 2.3.2, we introduce

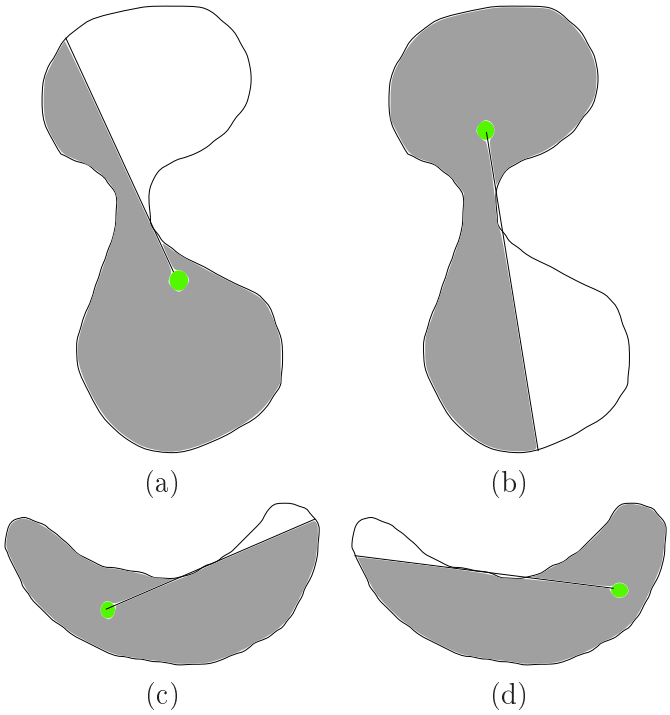


Figure 2.5: Difference of visible volumes near a segment boundary : (a) and (b) show how the visible volume changes drastically as one moves across segments that are clearly separate; (c) and (d) show the weakness of this method for separating segments that don't have a crisp boundary. In this case, visible volumes don't differ much in the vicinity of the segment boundary.

the notion of visibility in the continuous setting. We define the *medial visible volume* and explain a few of its properties. In section 2.3.3, we study the special case of the surface being a discrete polyhedral mesh. We introduce *visibility cells* and relate them to the medial visible volume of the continuous case. The concepts introduced here provide the theoretical justification for our segmentation algorithm (section 2.4). At that stage of practical implementation, we make certain simplifications from the conceptual level.

2.3.1 Definitions

Shape We define a *shape* A as the closure of an open, bounded subset of the Euclidean space $\mathbb{R}^3$. The open subset $\mathcal{O}^A \subset \mathbb{R}^3$ itself is called the *shape interior*. The boundary of the shape, given by $\mathcal{S}^A = A \setminus \mathcal{O}^A$, is called the *surface* (or sometimes, the *shape exterior*). In this thesis, we consider the boundary to be a closed 2-D manifold *i.e,* $\mathcal{S}_A \in \mathbb{S}$ where $\mathbb{S}$ is the set of all closed 2-D manifolds in $\mathbb{R}^3$.

Metric Distances The exterior geodesic distance $d^A_{\mathcal{S}}(u, v)$ between two points $u, v \in \mathcal{S}^A$ is given as the infimum of the lengths of all the paths connecting them on $\mathcal{S}^A$. Similarly, we can define the interior geodesic distance $d^A_{\mathcal{O}}(x, y)$ between two interior points $x, y \in \mathcal{O}^A$ as the infimum of all the paths connecting them within $\mathcal{O}^A$. Both these distances are metric, and thus $\mathcal{S}^A$ and $\mathcal{O}^A$ can be considered as metric spaces. In the rest of the thesis, we refer to $d^A_{\mathcal{O}}(x, y)$ as the *interior distance* and $d^A_{\mathcal{S}}(u, v)$ as simply the *geodesic distance*. Please note that the interior distance $d^A_{\mathcal{O}}(x, y)$ can be defined on the entire shape $A = \mathcal{O}^A \bigcup \mathcal{S}^A$. We denote the standard Euclidean distance in $\mathbb{R}^3$ without any suffix, as $d(x, y)$.

Medial Axis An effective way to link the surface $\mathcal{S}^A$ and the interior $\mathcal{O}^A$ would be through the medial axis transform (MAT) [Choi *et al.* 1997]. MAT represents each surface point by its distance from a select subset of interior points, known as the medial axis, which is formally defined as the set of points in $\mathcal{O}^A$ with at least two closest points in $\mathcal{S}^A$. Let $\Gamma(x)$ denote the set of closest surface points from $x \in \mathcal{O}^A$ *i.e*,

$$\Gamma(x) = \{u \in \mathcal{S}^A \mid d(x, u) = d(x, \mathcal{S}^A)\}$$

Then the medial axis $M(A)$ of the shape is given as

$$M(A) = \{x \in \mathcal{O}^A \mid |\Gamma(x)| \geq 2\}$$

The medial axis transform (MAT) is formally defined as the set of maximal balls inside $\mathcal{O}^A$ centered on the medial axis. Let $B(x, r)$ be an open ball centered at x with radius r *i.e*,

$$B(x, r) = \{y \in \mathbb{R}^3 \mid d(x, y) < r\}$$

Then, the medial axis transform is given by

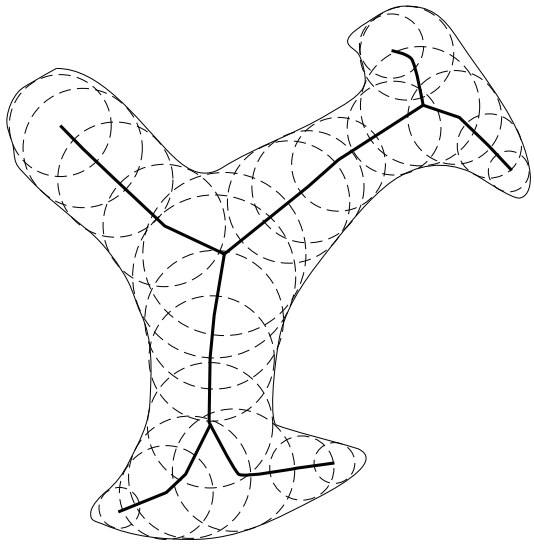


Figure 2.6: The medial axis and the corresponding medial balls of a shape

$$MAT(A) = \{B(x, d(x, \mathcal{S}^A)) \mid x \in M(A)\}$$

The balls in $MAT(A)$ are called medial balls; these medial balls cannot cover each other, and their union is exactly the shape interior *i.e*,

$$\bigcup MAT(A) = \mathcal{O}^A$$

Thus, the shape is completely defined by its MAT and vice-versa. MAT may thus be called the *volumetric representation* of the shape. An important consequence of this property is that the medial axis preserves the topology of the shape. Particularly, the medial axis of a connected shape is connected.

We have seen that every point on the medial axis $x \in M(A)$ is associated with a set of surface points $\Gamma(x)$. But the converse is also true.

Lemma 1 *There exists a medial ball $B(x, r) \in MAT(A)$ tangential to every point on the surface $u \in S^A$.*

The proof can be given easily by construction. We consider an infinitesimal ball $B(x, r)$ tangential to the given surface point u such that $|x - u| = r$. We then grow the radius r of the ball keeping it tangential to the surface point. We stop when this ball touches any other point on the surface, making it a medial ball.

λ-Medial Axis The medial axis is known to be very unstable to surface noise. Minor perturbations to the surface create several large slivers and branches to the medial axis. Thus, the medial axes of two very similar shapes may be widely different

from each other. Further the medial axis of a 3D shape may be composed of several disconnected sheets and doesn't resemble a linear skeleton that is its 2D counterpart. There are a few approximations to the medial axis that trade its exact convergence to the given shape for better stability to noise ([Attali *et al.* 2007] provides a good survey of various methods). The property of stability is important to us, as the input meshes in our problem are very noisy. [Chazal & Lieuter 2005] describe a special approximation, called the λ-medial axis, which is a subset of the medial axis points for which the set of closest surface points $\Gamma(x)$ cannot be enclosed within a ball of radius $< \lambda$ *i.e*,

$$M_\lambda(A) = \{x \in M(A) \mid \nexists B(y,r) \supset \Gamma(x)\ r < \lambda\}$$

$$MAT_\lambda(A) = \{B(x, d(x, \mathcal{S}^A)) \mid x \in M_\lambda(A)\}$$

The parameter λ is a global threshold that prunes medial axis points $\{x\}$ whose closest surface points $\Gamma(x)$ are not sufficiently distant from each other. This has the effect of disregarding medial balls that occur due to surface noise. In fact, [Chazal & Lieuter 2005] prove that the λ-medial axis is stable towards noise in two important ways : it preserves the topology of the shape for a restricted range of values of λ, and it has geometric stability with respect to small perturbations on the surface in terms of Hausdorff distance. However, since $M_\lambda(A) \subset M(A)$, it represents the shape only up to an approximation *i.e*,

$$\bigcup MAT_\lambda(A) \approx \mathcal{O}^A$$

Along with the medial axis $M_\lambda(A)$, the union of the λ-medial balls $MAT_\lambda(A)$ can also be proved to be stable with respect to minor geometric perturbations to the surface. The envelope of the λ-medial balls is a surface $\mathcal{S}^A_\lambda$ that approximates the given surface S^A of the shape, by smoothing out small noise disturbances. We build upon the notions of $M_\lambda(A)$ and $\mathcal{S}^A_\lambda$ so that our segmentation method achieves stability towards surface noise in the reconstruction. The parameter λ can be chosen such that it is large enough to filter out surface noise disturbances, but also small enough so as to preserve significant shape details.

Other Stable Approximations to Medial Axis There exists a problem with using a global threshold λ. The method of [Chazal & Lieuter 2005] does not distinguish noisy bumps on the surface from genuine features of the shape that are thin and tubular. When the medial axis is pruned aggressively by a large value of λ, such thin features of the shape may disappear in the approximation $\mathcal{S}^A_\lambda$. To better preserve such details, [Amenta & Kolluri 2001] propose a pruning method based on the angles subtended by the surface points $\Gamma(x)$ on the medial axis point x. But this method no longer enjoys any guarantees on the preservation of topology *i.e*, even though the shape is connected, its approximated medial axis need not be so. Another pruning method which would suffer from the same downside is to

choose λ adaptively on the surface, for instance, based on SDF [Shapira *et al.* 2008].

[Giesen *et al.* 2009] have recently proposed the scale-axis transform (SAT), which preserves shape details in a better manner even after aggressive pruning. The scale axis $M_s(A)$ is a generalization of the medial axis to different scales s. [Giesen *et al.* 2009] prove that it shares the good properties of M_λ *i.e*, topology preservation and geometric stability with respect to surface noise. The scale axis $M_s(A)$ and the corresponding envelope of scaled medial balls $\mathcal{S}_s^A$ may provide a stronger background for our ensuing discussion of visibility. But for any type of approximation to the medial axis, our argument would remain essentially similar. In the interest of simplicity, we base it here on $M_\lambda(A)$.

2.3.2 Visibility inside a Continuously Smooth Surface

We define the *visible volume* $VV(x) \subset A \subset \mathbb{R}^3$ from a *viewpoint* $x \in \mathcal{O}^A$ as a domain (connected open set) where the interior distance from x equals the Euclidean distance *i.e*,

$$VV(x) = \{y \in A \mid d^A_{\mathcal{O}}(x,y) = d(x,y)\}$$

The visible volume $VV(x)$ can be thought of as the volume spanned by shooting rays from the viewpoint in all directions towards the surface. This simple definition is similar to the one used by [Liu *et al.* 2009] and suffers from problems illustrated in figure 2.5. We provide a more elaborate definition of visibility in the following.

As can be seen in figure 2.7, $VV(x)$ resembles a star-shaped object. If we observe the set of surface points that are part of $VV(x)$, called the *visible surface* and denoted by $VS(x)$, they form a manifold with a boundary. As can be seen in figure 2.7, this boundary is a disconnected set of points that can be distinguished into two parts : *occluding boundary* and *non-occluding boundary*. Points on the occluding boundary have a special property - the viewing rays connecting them to the viewpoint x graze along the surface tangentially. The same viewing ray meets the surface for a second time at a non-occluding boundary point. Thus, the occluding boundary coincides with concavities on the surface, where as the non-occluding boundary is haphazard and provides no information about the shape.

Medial Visible Volume Now we define the *medial visible volume* - a subset of the visible volume $VV_\lambda(x) \subset VV(x)$, as the union of the medial balls $B(x,r) \in MAT_\lambda$ that are completely within $VV(x)$ *i.e*,

$$VV_\lambda(x) = \bigcup \{B(x,r) \in MAT_\lambda(A) \mid B(x,r) \subset VV(x)\}$$

The medial visible volume $VV_\lambda(x)$ is visualized in figure 2.8, for the same shape as displayed in figures 2.6 and 2.7. $VV_\lambda(x)$ is interesting because it is delimited by just the occluding boundary. The viewing ray at this boundary point is tangential to the surface, and also to the medial ball (such a medial ball always exists, according to our construction in lemma 1). The medial axis corresponding to this -

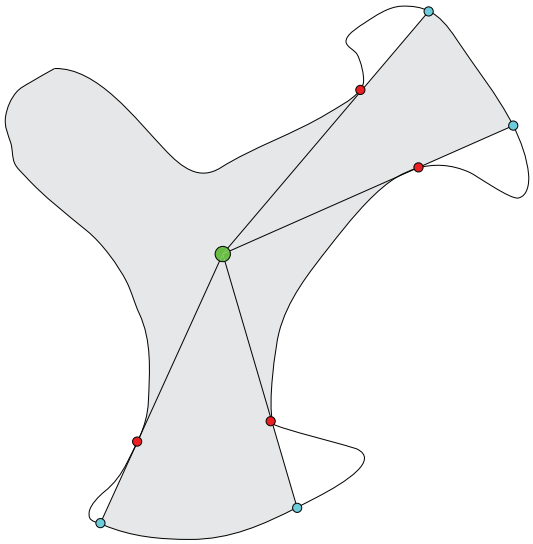


Figure 2.7: Visible volume $VV(x)$ from an interior viewpoint x, conceptualized on the 2D shape introduced in figure 2.6 : the viewpoint is colored green, the occluding boundary points are colored red, the non-occluding boundary points are colored cyan. This definition is very similar to the one used by [Liu *et al.* 2009]

$M(VV_\lambda(x)) \subset M(A)$ is a connected set, since $VV_\lambda(x)$ is connected. This subset of medial axis is an interesting shape attribute which encapsulates two types of information : shape thickness and surface concavities (in terms of occluding boundary). Further, since we consider λ-medial balls, the surface concavities that matter are only those that belong to $\mathcal{S}^A_\lambda$. Surface noise disturbances smaller than the parameter λ are smoothed out, and only concavities at a deeper scale remain on $\mathcal{S}^A_\lambda$.

For a surface with a smooth concavity, the occluding boundary slides gradually along with change in the viewpoint (as shown in figure 2.9). The nature of self-occlusions to visibility in the shape-interior is defined by both the local surface properties (given by curvature) and the global shape properties. We explain these connections in the following.

Tangential Segment Let us define the tangential segment $\tau(u)$ at any surface point $u \in \mathcal{S}^A_\lambda$ as the finite segment of the tangential plane that lies within the shape interior $\mathcal{O}^A$. The visible volume $VV_\lambda(y)$ computed from any point $y \in \tau(u)$ is delimited by the tangential medial ball at u. Thus, the area of the tangential segment $|\tau(u)|$ indicates magnitude of the space from which the surface point u occludes visibility. Please note that $|\tau(u)|$ is a non-local function that is defined by how the surface behaves at a distance from the surface point u. The tangential segments at a few concavities in an example shape are illustrated in figure 2.10

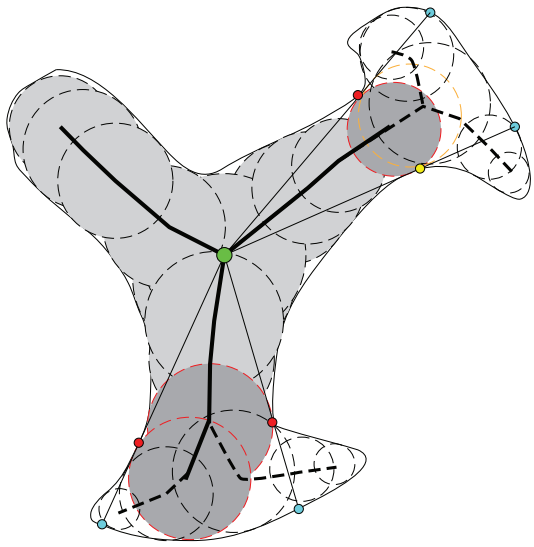


Figure 2.8: The medial visible volume $VV_\lambda(x)$ from the viewpoint x, to be contrasted with $VV(x)$ shown in figure 2.7 : the viewpoint is colored green, the visible medial balls are colored in grey. The occluding boundary points are colored red, the medial balls corresponding to them are colored dark grey. The non-occluding boundary points are coloured cyan. Please note that not all occluding boundary points of figure 2.7 are part of $VV_\lambda(x)$. One particular point (colored yellow in the figure) is left out because the corresponding medial ball touches non-visible surface portions. The medial axis of the visible volume $M(VV_\lambda(x))$ is shown in thick line, whereas the medial axis of the entire shape is shown in dashed lines.

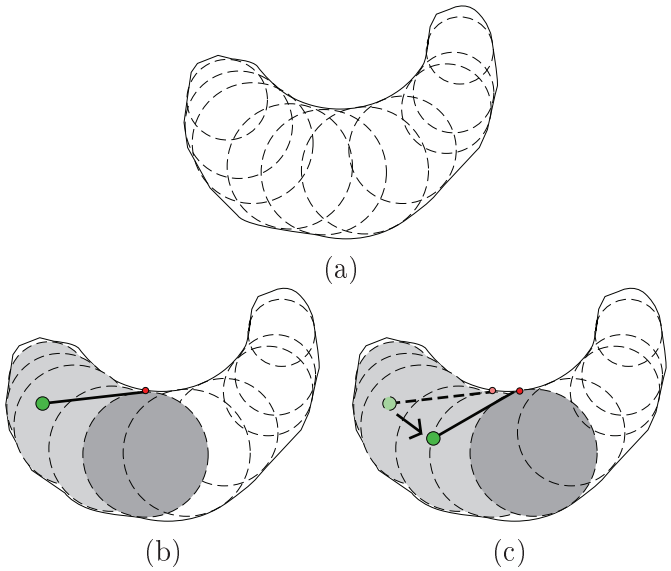


Figure 2.9: Visibility change along a smooth concavity : the medial axis and the medial balls of the shape are shown in (a), visible volumes from neighboring viewpoints are shown in (b) and (c). The viewpoint is colored in green, the visible medial balls are colored grey, the occluding boundary point is colored red, and the medial ball corresponding tangential to the viewing ray is colored dark grey. Please note how the viewpoint change in (b) induces a sliding of the occluding boundary point.

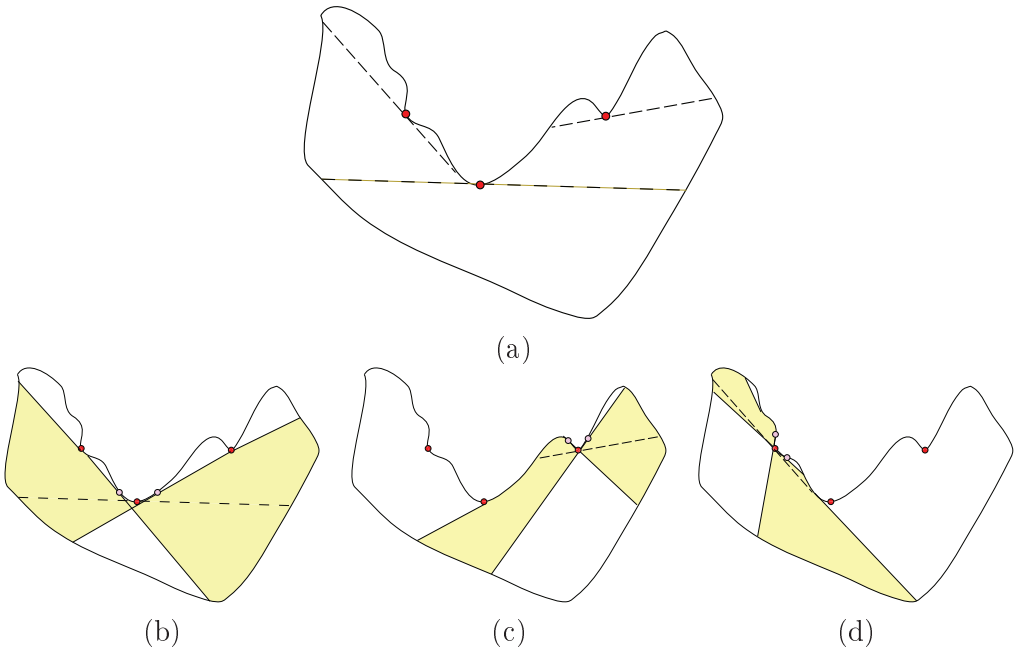


Figure 2.10: Tangential segments (a) and tangential volumes (b,c,d) at a few concavities in an example shape. The concavities are marked in red and tangential segments in dashed lines. The tangential volume, colored in yellow is the area (volume in 3D) across the shape interior by tangent lines (planes in 3D) in the neighborhood of a concavity (whose ends are marked in pink)

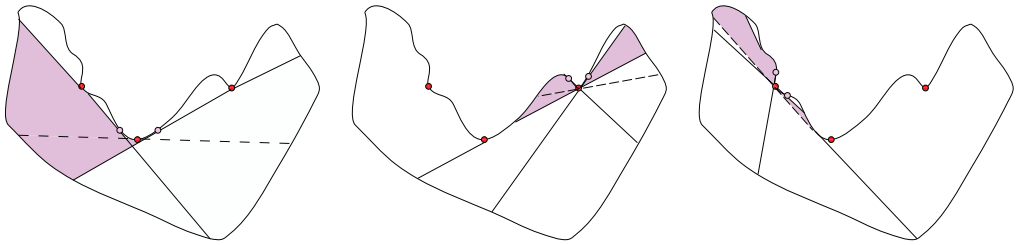


Figure 2.11: Occluded volumes : The area swept in the neighborhood of a concavity by the shorter pieces of the tangential segments. It can be seen that deeper concavities occlude larger regions of space.

Tangential Volume Consider an infinitesimally small patch dS centered at a surface point u. The area of the tangential segments can be integrated on this surface patch, and yield the tangential volume $\int |\tau(u)|dS$ around the neighborhood of the surface point. When this integral is taken over a surface region R and normalized over the entire volume of the shape, it indicates the likelihood that a point in R obstructs visibility from any point in the interior of the shape. Since the curvature at a surface point indicates how rapidly the tangent changes direction in its local neighborhood, the tangential volume is proportional to the magnitude of curvature at a point. (But it is not solely determined by curvature, since it also depends on global shape structure). Points with higher curvature are thus more likely to occlude visibility than points with lower curvatuture. The tangential volumes in the neighborhoods of a few concavities in an example shape are illustrated in figure 2.10-(b,c,d).

Apart from the likelihood of a point to occlude visibility, it sometimes also helps to visualize the amount of space that such a point is occluding. A simple but crude way to visualize this is by considering only a part of the tangential volume - the volume swept by the shorter of the two pieces of a tangential segment on either sides of the concavity. We call this the *occluded volume* and it is visualized for the example shape in figure 2.11. As expected, deeper concavities occlude more space than shallow concavities. However this also depends on the global structure of the shape.

2.3.3 Visibility inside a Discrete Polyhedral Surface

In this section, we consider a special class of surfaces which can be exactly represented (in the Hausdorff sense) by discrete polyhedral meshes; this class of shapes is of particular interest to us since our input data are discrete meshes.

A polyhedral mesh $M = (V, F)$ is a set of vertices and facets. Let f_i, f_j be two planar facets and e_{ij} be the edge between them. Let π_i, π_j be the planes containing

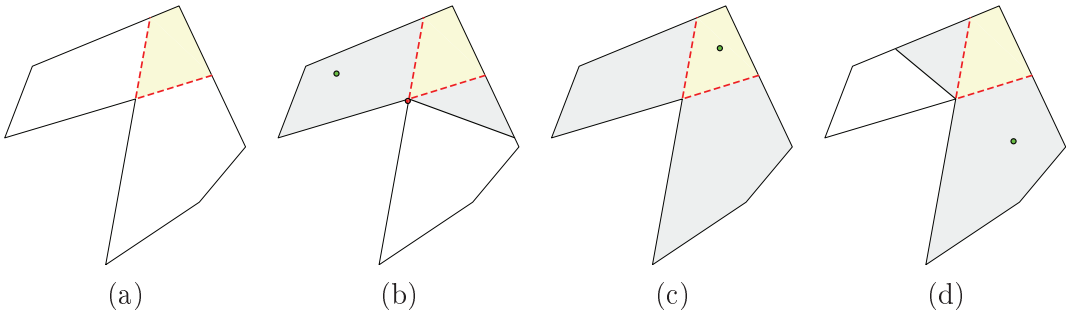


Figure 2.12: The joint cone visualized in 2D : (a) The joint cone (colored yellow) is created by extending the planes (dashed red lines) of the occluding edge into the interior of the shape. (b,c,d) Visible volumes computed from various viewpoints. The concavity occludes visibility on either sides of the joint cone, but not from within.

facets f_i, f_j respectively. The dihedral angle θ_{ij} between the planes is the angle between the outward facing normals of these planes; it gives a measure of surface curvature on the mesh. If this angle is such that $0 < \theta_{ij} < \pi$, then the surface is not concave. Otherwise, there might exist a concavity between the facets f_i, f_j along the edge e_{ij} (if the surface is not hyperbolic in that region).

Occluding Edges In the case of a discrete mesh, the surface concavities are given by a finite set of edges. Since they occlude visibility from the interior, we call them the *occluding edges*. The occluding boundary of the visible volume from an arbitrary interior point is composed of a subset of these edges. Given an occluding edge e_{ij} we extend the planes π_i, π_j of the adjoining facets f_i, f_j into the interior of the mesh. This defines a cone χ_{ij} that we term the *joint cone* of the edge (illustrated in figure 2.12-a). The edge e_{ij} occludes visibility from either sides of the joint cone (figure 2.12-b,d), but not from within (figure 2.12-c).

We illustrate the concepts in 2D, where the facets are line segments and occluding edges between them are just the vertices of the mesh. Please note that in 3D, the facets are planar polygons and the occluding edges are the shared line-segments between the polygons.

Visibility Cells For a closed manifold surface the joint cones do not extend to infinity, but are delimited by the surface of the object. The criss-cross intersections of the various joint cones with each other create cells $\{C_k\}$ that we call the *visibility cells* of the shape. The *joint number* of a cell is given by the number of joint cones that contain it. Cells with high joint number are likely to be at the center of the shape, at the joints between segments.

Lemma 2 *Each of the cells C_k is a convex polyhedron.*

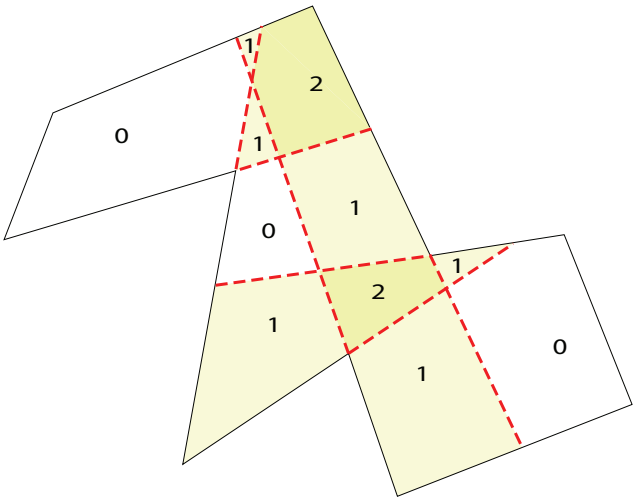


Figure 2.13: Visibility cells : Joint cones from various occluding edges create cells $\{C_k\}$ through their intersections. The *joint number* (marked within each cell) is the number of joint cones that pass through a cell.

The proof is trivial, and derives from the fact that the mutual intersection of conic regions with convex apex is convex.

From within a cell C_k, visibility is occluded by the same set of occluding edges. Thus, visibility does not change by much even if we move the viewpoint within the cell. In the discrete case, it is possible to uniquely seal the visible volume at an occluding edge e_{ij}. This is by considering the wing π_i of the joint cone χ_{ij} that occludes visibility. This might be extended till it meets the surface, or more efficiently by following other joint cones along the shortest convex path to the surface. We call this path the *extended occlusion* at e_{ij}, and it depends only on other occluding edges (not on the viewpoint). These paths are illustrated in figure 2.14. As shown in this figure, these paths define *blocking zones* (colored in yellow).

Cellular Visible Volume We define the cellular visible volume $VV_{\mathcal{C}}(x) \subset VV(x)$ as a union of visibility cells C_i such that it is delimited by the extended occlusion at each occluding edge. With this definition, a visible volume $VV_{\mathcal{C}}(x)$ $VV_{\mathcal{C}}(x)$ does not depend on the non-occluding boundary, and is completely defined based on the set of occluding edges in $VV(x)$. Due to this, $VV_{\mathcal{C}}(x)$ remains constant for different viewpoints in a visibility cell C_k, and can be represented using the notation $VV(C_k)$.

The cellular visible volume is visualized for a few example 2D shapes in figure 2.14.

Convex Decomposition by Visibility Decomposing a discrete mesh into the minimal number of convex segments is a combinatorial problem that is known to be NP-hard in the general case. However, there is a probabilistic interpretation to

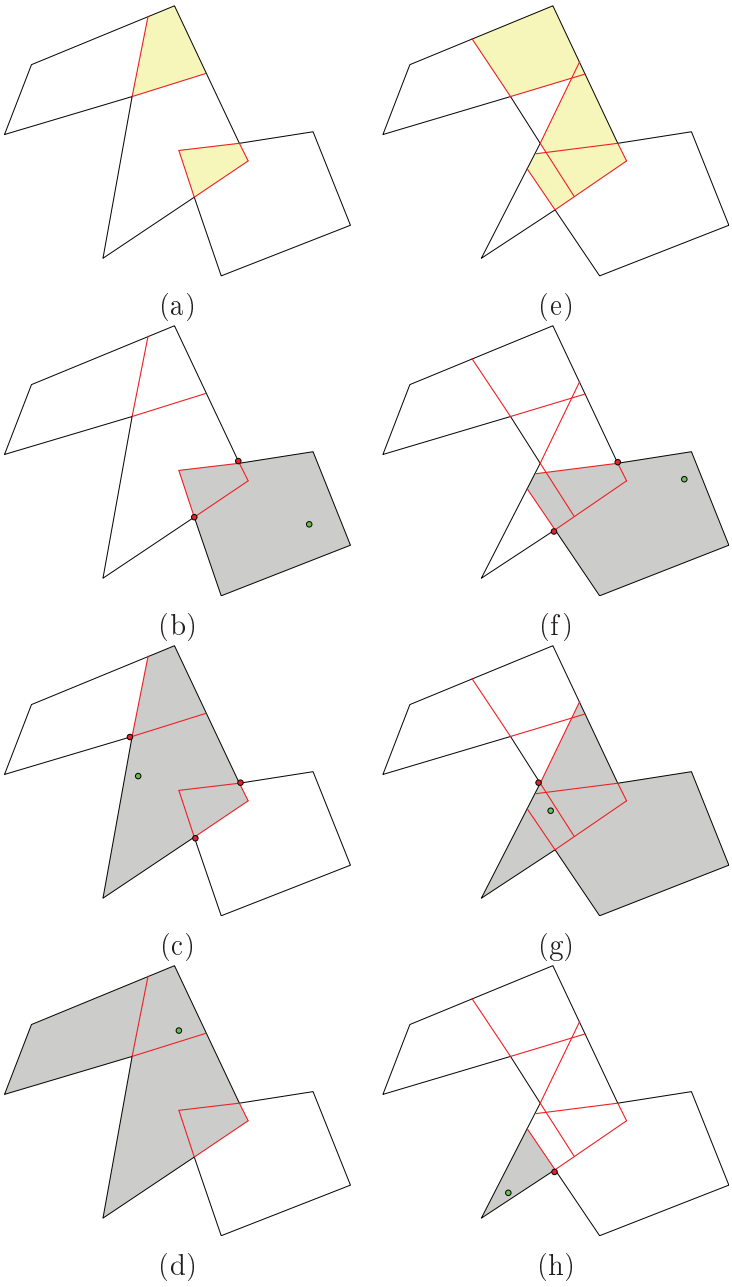


Figure 2.14: Cellular Visible Volume : (a), (e) The paths of extended occlusions (colored red) define blocking zones (colored yellow) that span between the concavities. (b,c,d), (f,g,h) Cellular visible volumes (colored grey) from different viewpoints (colored green) do not spread beyond the blocking zones.

this problem that is more useful practically. Visibility cells C_k of greater volume are generally more important and more likely to denote crucial geometric features of the shape (such as limbs of a human body). Regions of smaller volume are likely to arise due to surface noise on the reconstructed meshes. We consider the problem of finding sufficiently large convex segments which are composed of the visibility cells C_i.

Lemma 3 *$VV(C_k)$ is convex if the joint number of C_k is equal to 0, and if the number of occluding edges is 0 or 1.*

A cell with joint number 0 does not belong to any joint cone. If a surface concavity (occluding edge) is part of the visible volume, then it has to be occluding. The case with 0 occluding edges is trivial - there exists no concavity on the surface and hence the object is convex. The case with 1 occluding edge follows owing to the construction of the *extended occlusion* - it is aligned with the surface at the points of contact and remains convex throughout.

This above lemma provides a base condition for devising a recursive algorithm that decomposes a shape into convex segments. If the number of occluding edges is ≥ 2, then $VV(C_k)$ will not be convex but will be a star shaped object. However, it is possible to recursively decompose this object until the base condition is met. This idea forms the basis of our segmentation algorithm that we will describe in the next section.

Connection with $VV_\lambda(x)$ The medial visible volume $VV_\lambda(x)$ that we defined in the earlier section, is closely related to the cellular visible volume $VV_\mathcal{C}$. Both are delimited by just the occluding boundary, and take the global shape structure into consideration for sealing at the occlusion. So, $VV_\lambda(x)$ can be considered as a good initialization for computing $VV_\mathcal{C}(x)$, or for obtaining convex segments based on $VV_\mathcal{C}(x)$.

Please recall that the viewing ray at a boundary point of $VV_\lambda(x)$ is tangential to the medial ball at that point (visualized in figure 2.15. However, it is not possible to extend our earlier argument that the viewing ray is also tangential to the surface at the occluding boundary, since the gradient of a discrete surface has sharp discontinuities at the edges between the facets, and it is not possible to define a unique tangent at the boundary point. In fact, a large swathe of medial balls slide around an occluding edge and they correspond to a part of the medial axis (visualized in figure 2.16). The medial ball that delimits the visibility from a viewpoint is tangential to the viewing ray, and thus changes with the viewpoint. The size and position of this terminal medial ball depends not only on the concavity at the occluding edge, but also on the global nature of the shape.

To better explain the discrete case, we provide a new definition for the medial visible volume as the union of medial balls whose "points of contact" are visible from a viewpoint x *i.e*,

$$VV_{\lambda\star} = \bigcup \{B(x,r) \in MAT_\lambda(A) \mid \Gamma(B(x,r)) \subset VV(x)\}$$

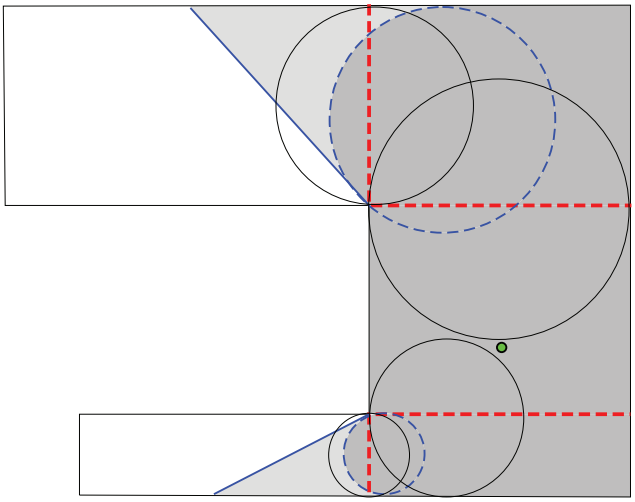


Figure 2.15: The medial visible volume $VV_\lambda(x)$ from a viewpoint in a discrete shape.

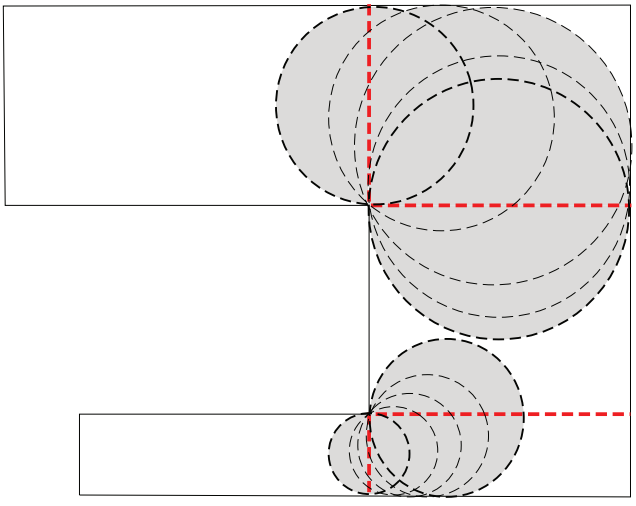
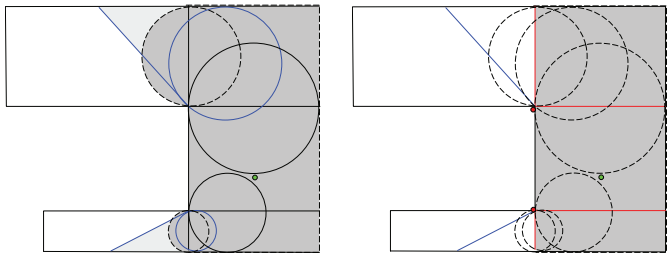


Figure 2.16: The medial balls sliding around the occluding edge (concavity) in a discrete shape



(a) (b)

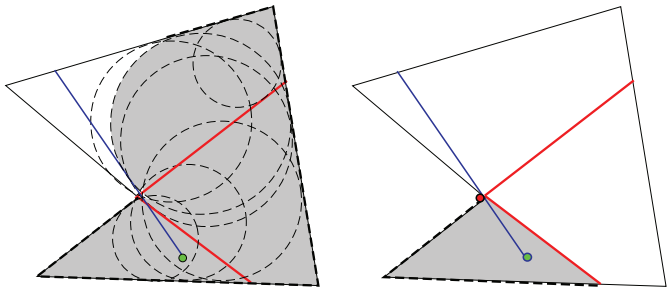
Figure 2.17: An example case where visible surface $VS_\lambda(x)$ (a) coincides with $VS_\mathcal{C}(x)$ (b)

Please note that $VV_{\lambda\star}$ converges to VV_λ in the continuous case with a smooth surface. But in the general case with discontinuities in the surface gradient, $VV_{\lambda\star}(x) \nsubseteq VV(x)$ and $VV_{\lambda\star}(X) \neq VV_\lambda(x)$. There exists a class of shapes for which $VS_\lambda(x) = VS_\mathcal{C}(x)$. This happens when the corresponding visibility cell is cylindrical in shape (with parallel walls), as shown in figure 2.15. Such cylindrical cells occur in natural shapes - for example, the limbs of a human body. In these cases, the delimiting points of $VS_{\lambda\star}(x)$ provide a good initialization to recover the cellular visible volume $VV_\mathcal{C}(x)$. However, there exist cases where $VV_\lambda(x)$ and $VV_\mathcal{C}(x)$ deviate significantly - an example is shown in figure 2.18.

Recall that the medial balls that remain in $MAT_\lambda(A)$ are of sufficiently large size and stable towards surface noise disturbances. In this scenario, with a simple sampling strategy of choosing viewpoints uniformly inside the shape interior $\mathcal{O}_A$ (in practice, just the medial axis $M_\lambda(A)$), the sample viewpoints are more likely to fall into regions of higher volume than the contrary. A visible volume computed from such a sample viewpoint is more likely to be composed of convex segments of large size than small. In section 2.4, we provide an algorithm based on these ideas to derive approximately convex segments from a given shape, that are probabilistically more likely to be large than small. Thus we have a lesser risk of oversegmentation, though that risk is not completely eliminated.

2.4 Segmentation Algorithm

In this section, we devise a practical algorithm for deriving approximately convex segments from an input mesh, using the conceptual background gained in the previous section. Since the reconstructed meshes are very noisy, we don't attempt to produce an optimal or exact algorithm for segmentation. Instead, we make choices to facilitate easy implementation that is sufficiently robust to noise. Particularly, we don't seek to obtain fine segment boundaries. Our objective is to estimate rough shape attributes in terms of segments, so we allow them to overlap. Since we deal with 4D (space-time) data, exact segmentation at each frame is not necessary, but a holistic understanding of the dynamic scene is required. So we use our static seg-



(a) (b)

Figure 2.18: An example case where visible surface $VS_\lambda(x)$ (a) does not coincide with $VS_\mathcal{C}(x)$ (b)

mentation work in this current chapter as a stepping stone for temporally coherent segmentation, which we present in chapter 3.

As explained before, the medial visible volume does not change by much within a segment, and depends solely on the occluding boundary points which lie on the surface. We represent the medial axis of the shape in a very approximate manner. Instead of seeking the exact points on the medial axis, we consider only a set of interior points $\mathcal{X}$. Instead of seeking the exact medial ball and its points of contact on the surface, we consider the *surface ring* of an interior point - which is a set of nearby points on the surface. Given an input mesh $\mathcal{M} = \mathcal{V}, \mathcal{F})$, we describe how to obtain the interior points $\mathcal{X}$ in section 2.4.1. We also introduce the *medial graph* - a data structure that we use to mimic the actual medial axis. We make these simplifications in order to obtain an easier implementation, and also in lieu of the poor quality of the reconstructed surfaces available as our input data. We define the problem of convex segmentation of the surface points $\mathcal{V}$ as the segmentation of the interior point set $\mathcal{X}$. Through the *surface rings* of points in $\mathcal{X}$, we transfer the segmentation to the surface points in $\mathcal{V}$. Our algorithm is schematically presented in figure 2.19.

We initiate the algorithm by marking all the interior points *free* and sort them according to an ordering function ρ for picking up viewpoints. A simple choice for ρ is the random-ordering function, another choice based on heuristics for faster convergence is given in section 2.5.2. We pop out the top element in the ordered free set $\mathcal{X}_F$ as the viewpoint w_i and compute its visible volume $VV_\star(w_i)$. We expect $VV_\star(w_i)$ to be approximately close to $VV_\lambda(w_i)$ and possess the same occluding boundary. We then decompose $VV_\star(w_i)$ into a set of convex segments $\{S_j\}$, tag every point with the corresponding segment, and remove the tagged points from the free set $\mathcal{X}_F$. We repeat the algorithm until the free set $\mathcal{X}_F$ is empty. In the following, we explain these steps in detail.

Algorithm 1: Convex Segmentation of the Surface

Input: a surface mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$

Output: a set of convex segments $\{S_{j\in[1..m]} \subset \mathcal{V}\}$

Obtain a set of interior points $\mathcal{X}$ for the mesh $\mathcal{M}$;

Initialize the set of free points $\mathcal{X}_F = \mathcal{X}$;

Sort $\mathcal{X}_F$ according to a function ρ;

i = 0;

while $\mathcal{X}_F$ *is not empty* **do**

  Select the top element of $\mathcal{X}_F$ as the viewpoint w_i; Compute the visible volume $VV_\star(w_i)$ from w_i;

  Remove all points $x \in VV_\star(w_i)$ from $\mathcal{X}_F$;

  Decompose $VV_\star(w_i)$ into convex segments $\{S_j\}$;

  i = i+1;

end

Figure 2.19: Algorithm for convex segmentation of a mesh

2.4.1 Construction of the Medial Graph

Our objective is to obtain a set of uniformly sampled interior points close to the medial axis. It is to be noted that computing the medial axis exactly is not trivial in 3D [Choi *et al.* 1997] [Dey & Zhao 2002], but we aim only to estimate a cloud of interior points. Our method is similar in principle to the work of [Shapira *et al.* 2008] on the *shape diameter function*. For each point on the surface v , we shoot a ray opposite to the surface normal and identify its point of intersection with the surface. We then trace a vector from v opposite to its normal with this length, and call it the *medial vector* at v. Due to the local surface variations and orientations of the normal, these medial vectors can be widely divergent. We remove the outliers (those which diverge excessively from neighborhood) and perform spatial smoothing such that neighboring points have similar medial vectors. These medial vectors map the set of surface points $\mathcal{V}$ to a set of interior points $\mathcal{X}$, which should roughly be uniformly distributed in the interior of the shape. These interior points for an example mesh are visualized in figure 2.20.

We define the *medial graph* of the shape as the k-nearest neighbor graph on this point set - it is shown for an example shape in figure 2.21-(a). For each interior point x, we create a ball whose radius is slightly larger than the length of the medial vector, and compute the set of surface vertices that fall within the ball. We call this set the *surface ring* of the interior point. For a roughly cylindrical shape in 3D, this picks points on a circular patch on the surface around the given interior point. More generally, for such shapes in n dimensions, the surface ring resembles

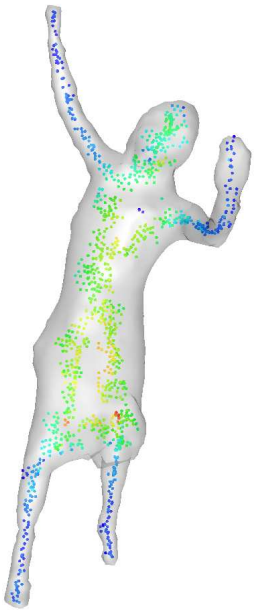
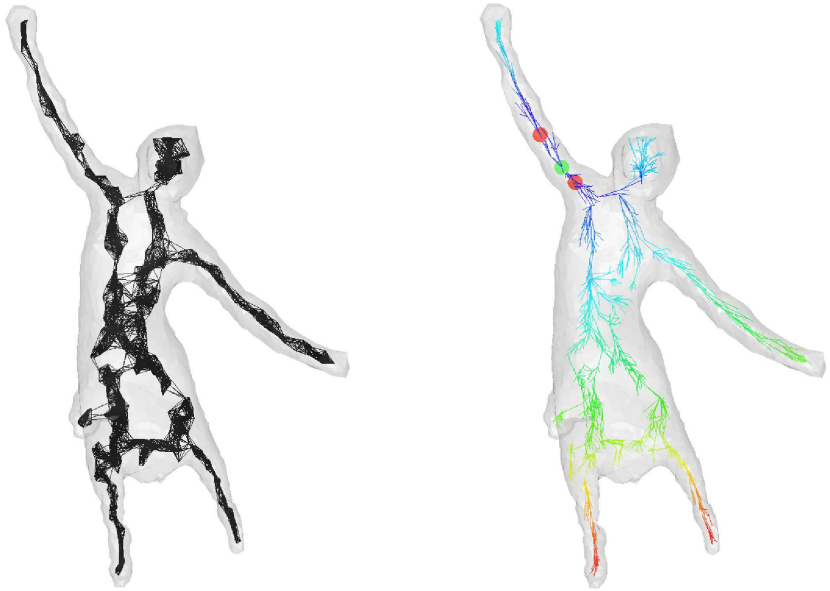


Figure 2.20: The set of interior points for an example mesh : Each of these points is colored according to the value of the *shape diameter* or the thickness of the mesh at that point. Warmer colors indicate greater thickness.



(a) (b)

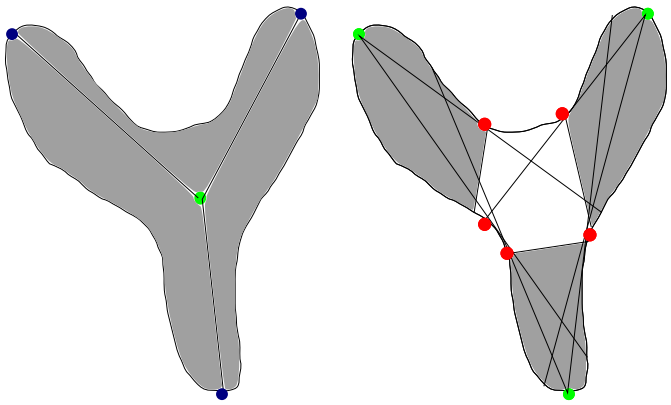
Figure 2.21: (a) The k-nearest neighbor graph on the interior point set, termed as the *medial graph* (b) The *medial tree* constructed by graph distance from a viewpoint, shown in green. The tree-branches in warmer colors are farther away from the viewpoint. The two red points are the *occluding tips* which delimit the visible volume from the given viewpoint.

a $(n-1)$-sphere around the given interior point. Particularly in the 2D case, the surface ring resembles a 1-sphere, which is nothing but a discrete set of two distinct points on either ends of the shape.

2.4.2 Computation of a Visible Volume

Given a specific viewpoint w from which to compute the visible-volume, we first order the set of interior points in a *medial tree* based on the shortest path from w on the *medial graph*. As illustrated in figure 2.21, this tree has w at the root and captures well the underlying structure of the shape as seen from w.

We then incrementally build the visible volume $VV_\star(w_i)$ by traversing the medial tree from w_i. We maintain a set of visible surface facets $F_{VV}(w_i)$ that is started with the surface ring of w_i. We then descend *medial tree* by steps and check if we can add a new interior point x_j to $VV_\star(w_i)$. We test by checking the visibility of all the vertices in the surface ring of x_j from the viewpoint w_i. This visibility test can be performed fast, by considering only the facets within $F_{VV}(w_i)$ for occlusions. If no occlusions are reported, we add x_j to $VV_\star(w_i)$ and its surface ring to $F_{VV}(w_i)$.



(a) (b)

Figure 2.22: Convex decomposition of a visible volume : (a) The visible volume from the view point is a star shaped structure. The viewpoint is colored green, and the protruding tips of the star are colored dark blue. (b) Visible volumes are then computed from each tip of the star, which run into occluding boundaries (colored red). A convex segment is identified per each tip (colored gray), and the remaining stub of the visible volume (colored white) becomes an independent convex segment.

When a non-visible surface point is encountered, we mark the corresponding interior point t as an *occluding tip*. We chop the *medial tree* at that point (figure 2.21-b) and don't grow any further. The algorithm is terminated when the medial tree is chopped on all sides and no more points can be added. The obtained visible volume $VV_\star(w_i)$ is a crude approximation for $VV_\lambda(w_i)$. We compute the *extent* of $VV_\star(w_i)$ as the largest distance across any two visible surface points. If this value is small compared to ε, we reject $VV_\star(w_i)$ to be too small to be significant. Otherwise, we proceed to the next step.

2.4.3 Convex Decomposition of a Visible Volume

A visible volume is shaped as a star, and may contain multiple convex segments (as can be seen in the figures 2.7 and 2.22). Here we describe how to decompose the visible volume into its constituent segments. We first obtain the *tips* of the visible volume by identifying the set of points which are locally maximal in their neighborhood with respect to the distance from the viewpoint w_i. The occluding tips discovered as explained in the earlier section are also added to this set. These tips are shown colored dark blue in the figures.

Instead of checking the convexity on all the surface of $VV_\star w_i$, we test for it approximately by checking the line segments joining the different tips of this star-shaped object. Since the tips lie at the extremities, they are more likely to hold a concavity between themselves than other points on the surface. We compute a Boolean compatibility graph between the tips based on whether the surface rings

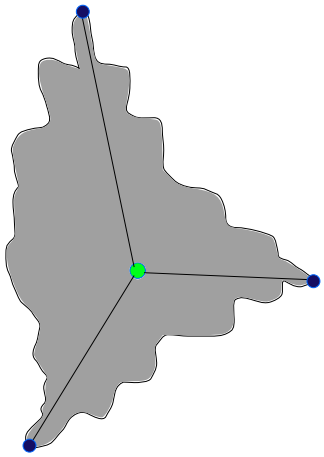


Figure 2.23: An extreme case of a visible volume, which can be decomposed by a recursive application of our algorithm at each of the tips

associated with a pair of tips are completely visible from each other. If all the tips are mutually compatible with each other, we mark the visible volume as convex and terminate the algorithm. Otherwise, we treat each tip as a new viewpoint and recursively compute its visible volume, but this time restricted only to the star-shaped volume visible from the parent viewpoint w. When restricted in such a manner, the new visible volume computed from a *tip* usually comes out as a single convex segment. But there can exist complex shapes for which this procedure should be applied recursively at multiple levels (for example, illustrated in figure 2.23). But we note that such cases are extremely rare in normal scenarios. If all the tips are accounted for and there still remains a stub of unassigned points (as is the case in figure 2.22-b), we mark this stub as an independent convex segment. At the end of this recursive procedure, we obtain a set of convex segments from the visible volume.

2.5 Implementation Details

Using algorithms from sections 2.4.2 and 2.4.3 repeatedly on the shape, we decompose it into a set of approximately convex segments. Our objective is to rapidly segment out roughly convex regions such as the limbs of a human body. We describe a few implementation details towards this purpose.

2.5.1 Testing for Approximate Convexity

Since we seek approximately convex segments, we request the user to set a threshold ε that specifies the crudeness of the apprroximation. We follow the definition of [Lien & Amanto 2007] : an ε-convex shape is one in which the Hausdorff distance between itself and its convex hull is less than ε. This ε is related to the parameter λ of the λ-medial axis that we introduced in the theoretical formulation of the

problem. It provides the scale at which surface details are unimportant and can be smoothened away. The same value of ε can be used for the segmentation of different reconstructions of the scene at various frames.

One possibility for automatically determining ε is by setting it to the length of the smallest medial vector (in section 2.4.1). However, we would like to note that $\mathcal{X}$ need not be as densely sampled as $\mathcal{V}$; interior points can be constructed for only a subset of the surface points. So we first perform a mesh simplification step (using the CGAL library) by requesting the user to provide the average edge length of the simplified mesh. We set ε to three times this value.

We use this threshold is used in two places (1) identifying tips and (2) testing for compatibility between tips. For (1), we perform non-maximal suppression in detecting tips, and don't return a tip within a distance of ε from another tip. For (2), we consider two tips as being incompatible only if there exists a pair of points v_a, v_b in their corresponding *surface rings*, the line $\overline{v_a v_b}$ joining which is farther than a threshold ε from the surface (illustrated in figure 2.24). $\overline{v_a v_b}$ can be thought of as similar to the bridge on the convex hull that is used by [Lien & Amanto 2007], but it has certain differences. The first difference is that it is always a line segment and easy to compute, even in 3D. The second difference is that it need not lie on the convex hull of the shape. It is not susceptible to minor surface noise (as depicted in figure 2.2), and does not subdivide segments unnecessarily.

We discover each convex segment as a set of interior points. Because each interior point is mapped to a *surface ring*, there exists a set of surface points that are associated with each segment. In our results, we visualize segments as these sets of points on the surface. These point-sets can be potentially overlapping. We do not invest any effort in obtaining crisp boundaries between the segments, because our input meshes are themselves noisy and not very reliable. In extracting segments, our objective is only to discover the gross attributes of the shape, that can be used for the broader goal of spatio-temporal modeling.

2.5.2 Viewpoint Selection

Our algorithm correctly identifies convex segments irrespective of where the viewpoints are placed. However the algorithm is faster when all the tips of the visible volume are mutually compatible, thereby returning just one convex segment. In the theoretical formulation given in section 2.3.3, such cases happen when the viewpoint is placed within a visibility cell with joint number = 0. It is difficult to estimate all these regions beforehand, but a subset of them, namely the global protrusions of the shape, can be computed easily. In articulated figures such as human bodies, protrusions such as hands and legs can be detected easily by computing the geodesic eccintricity (figure 2.3-a). The protrusions of a shape come out as local maxima of this function. We use this function as the sorting function ρ in algorithm 2.19. We describe the method of detecting surface protrusions in greater detail in chapter 4, section 4.4.1.

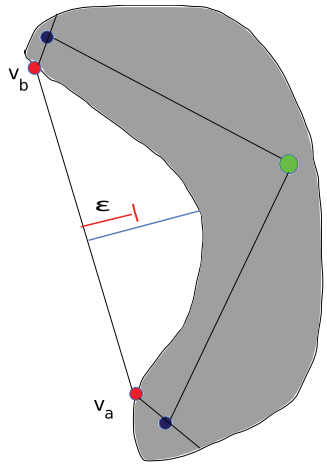


Figure 2.24: Selecting concavities that are only deeper than a user specified threshold δ_2 : The tips v_a and v_b shown in the figure are joined by a bridge from which the depth of the concavity is measured, which in this case is found to be higher than δ_2. Please note that the bridge $\overline{v_a v_b}$ need not lie on the convex hull, in contrast to [Lien & Amanto 2007].

2.6 Results

We tested our algorithm on various 3D meshes reconstructed from real world scenes. In this section, we present example results on 4 different reconstructions.

The *juggle* sequence is introduced earlier in figure 1.9; it captures a juggler juggling with clubs. We chose the visual hull reconstruction at a particular frame, numbered 115, to show the results of our convex segmentation. The results are shown in figure 2.25. In this figure, (a)(b) and (c) show different convex segments that are extracted. We can see that the juggling club and hand are extracted as different segments even though they are topologically clubbed together in the input mesh. The section (d) in the figure shows a color-coded representation of all the convex segments detected on the mesh.

The *flashkick* sequence is the visual reconstruction of a hip-hop dancer doing a backflip. We chose the reconstruction at frame 50 to show the results of convex segmentation. These results are shown in figure 2.26.

The *kids* sequence shows a boy and a girl playing with a large ball. We chose the reconstruction at frame 78, where the hands of the boy are touching the ball, and due to which they are clubbed together in the reconstructed mesh that is input. The identified segments on this mesh are shown in figure 2.27.

The *dance-girl* sequence shows a dancer dressed in a loose fitting robe with a belt. We chose two reconstructions at frames 575 and 585 to highlight the problem we face. The belt of the robe is visible clearly in the frame 585, but is not visible at all in 575. This happens because the belt touches the torso at frame 575 and is clubbed with it in the input mesh. In figure 2.28, we show different individual

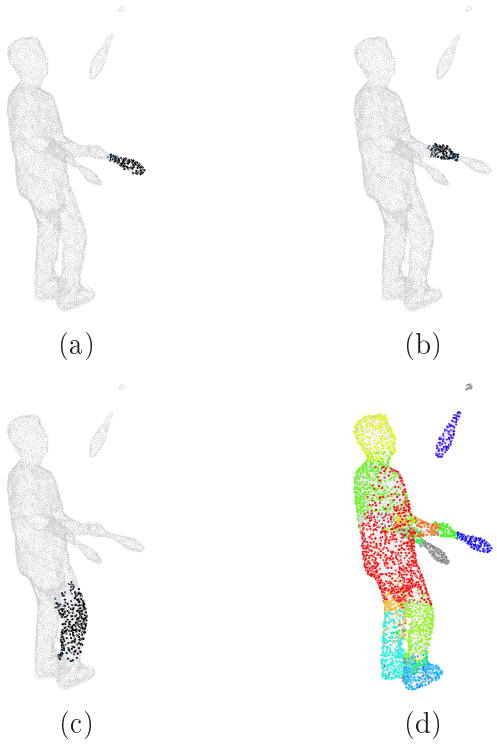
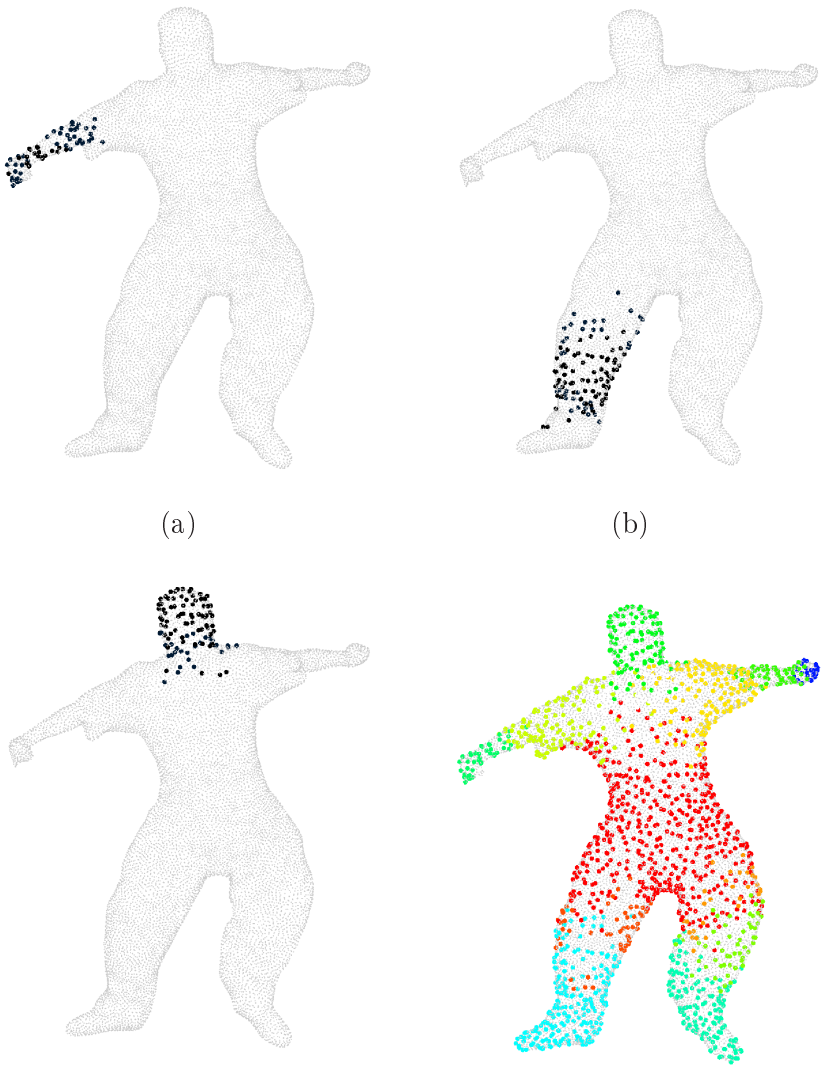


Figure 2.25: Convex segments at frame 115 of *juggle* sequence : (a)(b)(c) individual segments (d) color-coded representation of all the segments found

segments that are identified in both these meshes. In the next figure 2.29, we show the total segmentation of the meshes at both the frames. It can be seen that the extracted segments are not completely identical to each other, because of the underlying visual reconstruction method is not based on any model. The size, thickness and other attributes of the segments are also different. This gives the motivation for our work in chapter 3, where we handle such problems and derive a temporally coherent segmentation of an entire mesh sequence.

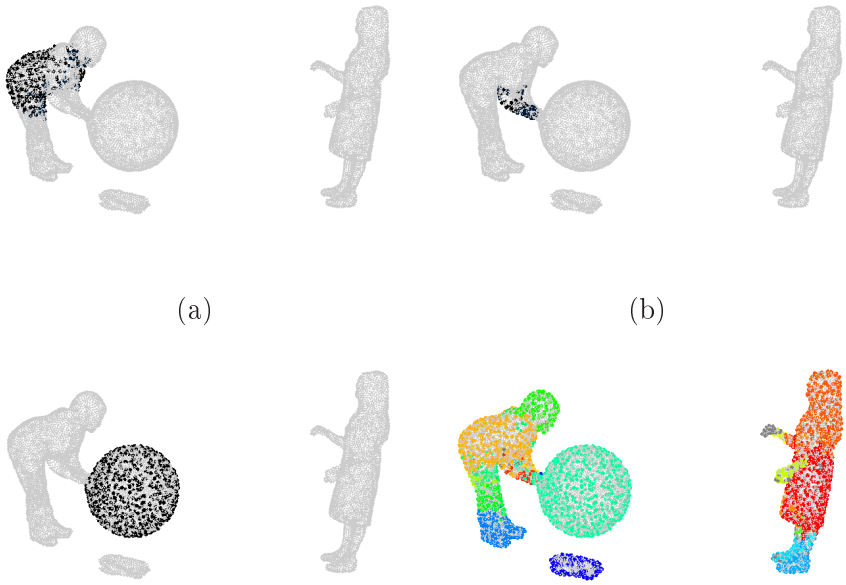
The *flashkick* is made available to us by the SurfCap group at the University of Surrey. The *juggle, kids* and *dance-girl* sequences are capturedby us using the Grimage multi-camera platform at INRIA Grenoble [4dr 2010].



(a) (b)

(c) (d)

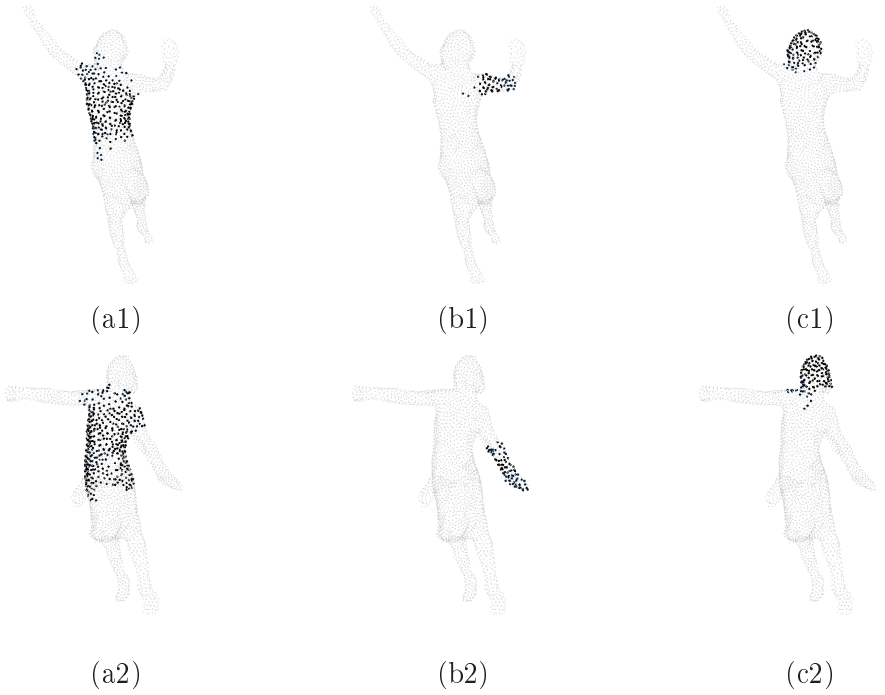
Figure 2.26: Convex segments at frame 50 of *flashkick* sequence : (a)(b)(c) individual segments (d) color-coded representation of all the segments found



(a) (b)

(c) (d)

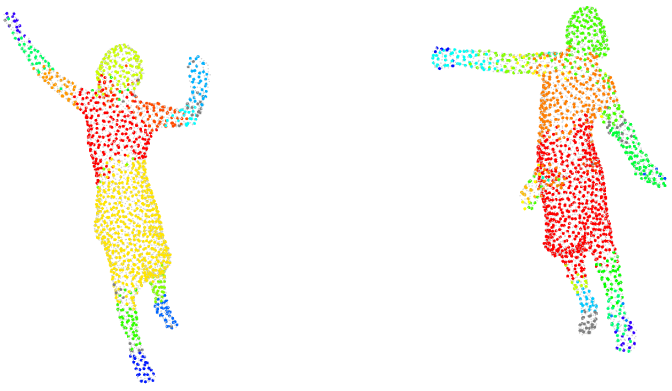
Figure 2.27: Convex segments at frame 115 of *juggle* sequence : (a)(b)(c) individual segments (d) color-coded representation of all the segments found



(a1) (b1) (c1)

(a2) (b2) (c2)

Figure 2.28: Convex segments for the *dance-girl* sequence : (a1), (b1),(c1) belong to the frame 575; (a2),(b2),(c2) belong to the frame 585. (a1), (a2) show the upper torso; (b1),(b2) show the right hand; (c1),(c2) show the head



(a) (b)

Figure 2.29: The static segmentation of the reconstructed meshes from the *dance-girl* sequence (a) at frame 575 and (b) at frame 585

Chapter 3

Temporally Coherent Segmentation of a Mesh Sequence

Contents

In this chapter, we describe a method for segmenting consistently an evolving 3D scene reconstructed individually at different time-frames. Specifically, we seek convex segments that move rigidly over time. The set of segments and their motions (rigid transformations) can be thought of as a crude spatio-temporal model that explains the dynamics of various objects in the scene in a coarse manner.

As our starting point, we use the convex segments derived in chapter 2 from static mesh reconstruction. We perform this convex segmentation at a few randomly sampled frames in the sequence. We track the motion of each of these segments along the sequence through a variation of the ICP algorithm. Through these estimates of rigid motion along the sequence, we refine the segments discovered earlier. We describe a method for obtaining a comprehensive set of segments, defined by us as an ε-cover of the sequence, that completely characterizes the observed scene points at all the frames, up to an approximation. We show the results of this temporally coherent segmentation on several challenging sequences of reconstructed meshes from real world scenes. An earlier version of this work has been published by us in [Varanasi & Boyer 2010].

3.1 Introduction

In chapter 1, we have reviewed various existing methods for spatially reconstructing the geometry of the 3D scene which integrate information captured from multiple cameras and sensors. This class of methods attempt to model the *objects* in reality. We have also reviewed another class of methods that assume the presence of a certain *object* in the scene and try to estimate its motion over time. Such methods attempt to model the *events* in reality. Both these classes of methods observe the traditional distinction between space and time in the concrete world. In this thesis, we eschew this distinction, and attempt to perform a *spatio-temporal* modeling of reality. The current chapter describes our first attempt in this direction. Here, we describe a method of temporally coherent segmentation of a sequence of visual reconstructions, that tries to estimate both the *objects* and *events* in a given scene.

3.1.1 Segmentation as a Spatio-Temporal Model

Temporally coherent segmentation of a mesh sequence has two components - a spatial component which describes the spatial description of each segment and a temporal component which describes the motion of each segment over time. In our case, the spatial component is given by the positions of surface points that belong to the segment, and the temporal component is given by the rigid motion estimates of each segment at each instant of time, in terms of translational and rotational components. Temporally coherent segments are a crude spatio-temporal model of a dynamic scene - the *objects* being described are limited to a discrete set of segments, and the *events* being described are limited to motion of the segments. However, such description is an arguably good representation for modeling inherently articulated motion, as occurs in the motion of human beings or animals. In a generic setting, there can be multiple such actors present in the scene and interacting with each other. We do not intend to impose any restrictions on the type and number of actors in the scene, or on the clothing that they wear. Figure 5.5 shows a result of our method and illustrates what we seek : the sequence consists of two children playing with a ball, and a set of segments are consistently detected on all the frames.

At this stage, we do not try to estimate a globally coherent motion across the shape. Each segment is treated independently and no higher relationships are estimated between the segments and their properties. For example, we do not group all the segments belonging to an actor by marking them as such. The motion information from these segments is also limited to rigid motion estimates for the entire segment - a restriction which renders our model inadequate to capture the true non-rigid motion of real world scenes. The latter chapters deal with modeling these dynamics in greater detail.

3.1.2 Limitations of Static Segmentations

In chapter 2, we briefly discussed how the segments obtained from a static reconstruction at a frame are not reliable over a whole sequence. Here, we list several

reasons why this is the case.

1. **Silhouette Errors :** Segmentation of foreground from background in the images is not perfect, and induces errors in reconstructed meshes, especially when the foreground and background colors are very similar. This is the case with strong shadows that create spurious objects (an example is shown in figure 3.1-d).

2. **Surface noise :** The reconstructed surfaces are not generally smooth, and show significant local variations in the tesselation of triangles, unlike user-modeled meshes. Due to this, local metrics like surface curvature are often unreliable.

3. **Topological Inconsistencies due to Occlusions:** Visual occlusions severely limit the accuracy in reconstructed surfaces. Distinct objects get clubbed together as a single object when they approach or touch each other. An example is shown in figure 3.1-(a).

4. **Degenerate articulations :** Certain poses of articulated shapes may club two body parts as one when the underlying bones are aligned with each other. For example, the upper arm is clubbed together with the lower arm when the arm is stretched.

5. **Loose Clothing :** The clothes of the actors produce significant occlusions, which cannot be overcome by sensors that capture within the visual spectrum. The relative thickness and size of the detected body-parts is affected through artificial bulges introduced into them due to loose clothes. An example is shown in figure 3.1-(b).

6. **Appearance of new actors :** The configuration of the scene might change over time as new actors make appearance. Any spatio-temporal model that is not based on an observation of the entire sequence cannot handle for such new appearances.

Some of these limitations may be rectified (the earlier ones are easier than the latter ones) - by improving the algorithms for silhouette extraction and visual reconstruction, or by explicitly using a model. In this chapter, we present another way to overcome these limitations - by using the temporal information that connects these mesh reconstructions.

3.1.3 Related Work

One approach to derive temporally coherent segments is to base the segmentation on a function that is oblivious to pose-changes and articulations of the shape. Geodesic distances on the surface of the mesh are invariant to isometry, and can be used for matching surfaces which differ by a non-rigid deformation [Bronstein *et al.* 2006b].

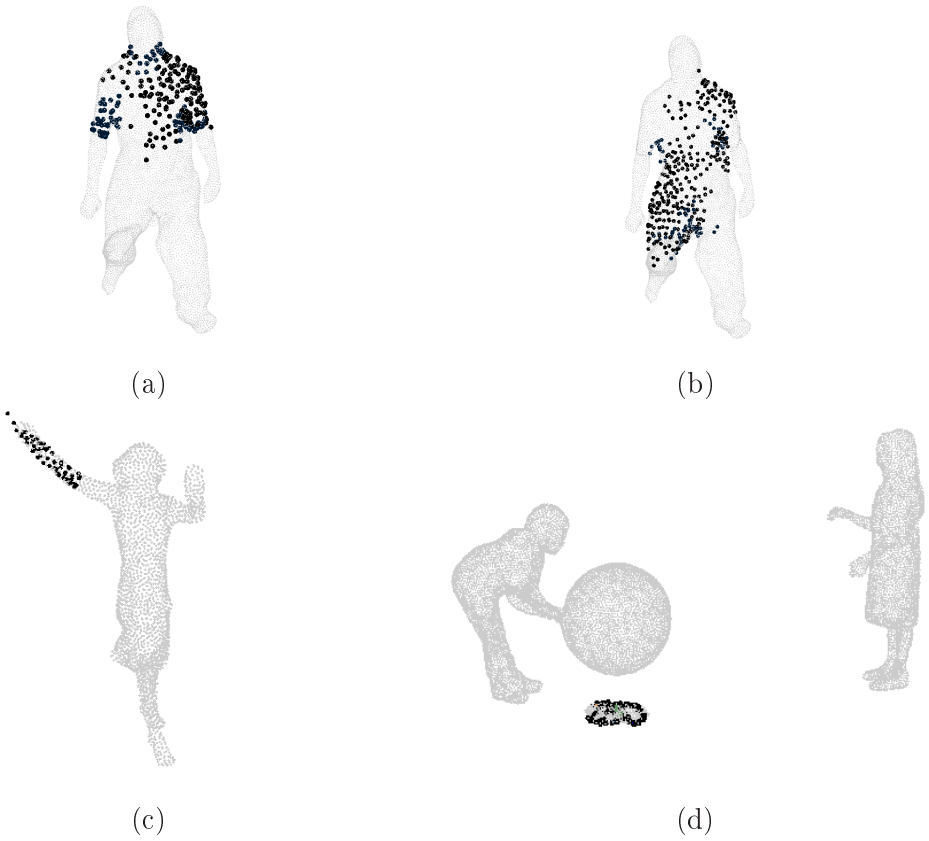


Figure 3.1: Limitations of static segmentation : (a) the upper arms get merged with the torso due to reconstruction artifacts (b) due to loose clothing, the left thigh merges with the torso (c) the upper and lower arms of the dancer are detected as a single convex segment (d) a spurious 3D object arising due to a strong shadow is identified as a segment

However, they are very sensitive to topological changes (which occur often in visually reconstructed meshes). [Bronstein *et al.* 2009b] use a metric based on diffusion over the surface through a heat kernel, which is less sensitive in this regard. Functions defined on the mesh-volume are more resistant to topological changes. [Shapira *et al.* 2008] use the local thickness of the mesh-volume, termed the shape-diameter function, to consistently partition different surfaces over articulations and pose changes. These and other works in the geometry processing community use surface curvature to derive the final segmentation, which is fine for smooth 3D models but is not reliable for visually reconstructed meshes.

Instead of relying on the supposed consistency of a function, a set of objects can be explicitly marked as related and can be segmented consistently. [Golovinskiy & Funkhouser 2009] have proposed a method that consistently segments sets of objects such as chairs and airplanes. They need point-wise correspondence between the objects, which they obtain by doing a global ICP registration and taking the closest point pairs between objects. Though this approach is powerful, it is not suited to the context of a sequence of meshes as it doesn't exploit the temporal information. In the context of a mesh sequence,[de Aguiar *et al.* 2008b] use the point trajectories (computed, for example, using [de Aguiar *et al.* 2008a]) to cluster points into rigidly moving parts.

The problem of coherently segmenting a sequence of reconstructed meshes is closely related to the problem of estimating a motion model for scene dynamics. Traditionally, embedded skeletons are used to animate meshes. So, automatic estimation of skeletons and fitting them into a mesh is related to our problem. A graphical model can be used to explicitly match two surfaces differing from a non-rigid deformation [Anguelov *et al.* 2004a] [Starck & Hilton 2007a], and these point correspondences can be later used to recover the articulated structure (skeleton) of the shape, as proposed by [Anguelov *et al.* 2004b]. [Chang & Zwicker 2008] propose an interesting alternative in the graphical model approach; they compute a putative set of rigid transformations between surface points on two articulated shapes, and treat these transformations (instead of the points themselves) as the labels for the graphical model. These approaches are very powerful but surface matching or motion estimation is a very hard problem, and the results are often inaccurate in the presence of surface occlusions, topological noise and disjoint objects. In this paper, we propose a segmentation algorithm that does not rely on pre-computed correspondences or motion estimates.

Our work is most closely related to that of [Cuzzolin *et al.* 2008]. In this work, the authors use locally linear embedding (LLE) to represent a cloud of points and derive a segmentation in this space. The segments are then propagated across time to obtain a temporally coherent segmentation of a voxel-sequence into protrusions of the shape, such as head, hands and legs. Though very powerful, this method cannot be used directly for identifying rigid body-parts (for example, separating the upper-arm from the lower-arm) and for working on disjoint objects (for example, two interacting persons). To the best of our knowledge, nobody has attempted the segmentation of a mesh-sequence with multiple interacting persons reconstructed visually in a multi-camera environment.

3.2 Approach Outline : ε-cover of a Mesh Sequence

We consider as input a mesh sequence $\mathcal{M}^{t\in[1..n]}$. Each of the meshes $\mathcal{M}^t$ is an individual 3D reconstruction of a real scene composed of several actors and objects, and is composed of a set of vertices $\mathcal{V}^t$ and a set of facets $\mathcal{F}^t$. We further represent the volumetric information of the mesh $\mathcal{M}^t$ by a set of interior points $\mathcal{X}^t$. An individual surface point is denoted as $v_i^t \in \mathcal{V}^t$ and an individual interior point is denoted as $x_i^t \in \mathcal{X}^t$.

The task is to estimate a set of segments $\Psi = \{\mathcal{S}_j\}$ and their rigid transformations over time $\mathcal{T}_j^{t\in[1..n]}$. Each segment $\mathcal{S}_j$ is detected at a specific *seed-frame* $t = k$ and consists of a set of N_j surface points. These points are represented by a matrix $\mathcal{S}_j^k$ of size $N_j \times 4$, with each column representing a surface point in homogeneous coordinates. The segment points at another frame t are then given by the matrix $\mathcal{S}_j^t = \mathcal{T}_j^t * \mathcal{S}_j^k$ (with $\mathcal{T}_j^k = I$). An individual segment point in the segment S_j^t is denoted as s_{ji}^t.

We say that a set of segments $\Psi = \{\mathcal{S}_j\}$ is an *ε-cover* of the mesh sequence if no surface point v_i^t reconstructed at frame t is farther than a distance of ε from the corresponding set of segments $\{\mathcal{S}_j^t\}$. *i.e,*

$$\forall k \quad \forall v_i^t \in \mathcal{V}^t, \quad Min_j \quad Dist(v_i^t, \mathcal{S}_j^t) < \varepsilon \tag{3.1}$$

An ε-cover yields a temporally coherent segmentation of the sequence by a margin ε.

3.3 Tracking a Segment over Time

We use the algorithm described in chapter 2 to obtain convex segments from a 3D mesh reconstruction within a sequence. These segments usually identify body parts, but not all of them are equally reliable. Due to the limitations described in section 3.1.2, we obtain several segments that are non-informative about the scene dynamics. Such segments are technically convex, but remain so only in one or few frames. In this section, we describe a method for rejecting such segments and identifying ones which are consistent with the entire mesh sequence. To achieve this, we estimate the motion of each segment over the entire sequence. We observe that the convex segments of a mesh usually correspond to the articulated parts of a body and thus, their motion can be approximated as rigid.

3.3.1 Reliability Estimation

We take the surface point cloud S_j^k of a convex segment S_j detected at frame k. We estimate the motion of this point cloud as a set of rigid transformations $T^{t\in[1..n]}$ over the entire sequence. To do this, we iteratively register the segment's point cloud to the mesh points in the neighboring frames using the ICP algorithm [Besl & McKay 1992]. As the 3D video is captured at a good frame-rate (around

10 to 20 fps), neighboring frames are sufficiently close to each other, justifying the application of the ICP algorithm.

We accelerate the ICP algorithm by using spatial kd-tree organization. When we select closest points for registration, we reject matches between points with widely discrepant surface normals. As observed by Pulli [Pulli 1997], this is an efficient strategy for eliminating outliers in the registration. Since the ICP algorithm is based on local search, it doesn't find good matches across large movements. Following Shapira *et al.*[Shapira *et al.* 2008], we use the two mesh features discussed in the earlier chapter (a) the geodesic eccentricity, which helps in detecting surface protrusions and (b) the shape diameter which gives the thickness of the volume (depicted in figures 2.3-a and 2.3-b respectively). We reject matches between points with discrepant values for these features, this strategy helps us find better matches than simple closest point search. We used simple geometric registration for ICP, even though more complicated methods exist that account for photometric information or scene-flow.

3.3.2 Refining a Segment

We consider the success of registration along the sequence as an estimate of reliability for a convex segment. For each point s^k_{ji} in S^k_j, we compute its estimated position at frame t as $s^t_{ji} = T^t * s^k_{ji}$. The discrepancy $d(s^t_{ji})$ in this estimate is computed as the least distance to the mesh vertices reconstructed at frame t.

$$d(s^t_{ji}) = Min_a \quad ||T^t_j * s^k_{ji} - v^t_a|| \tag{3.2}$$

If this value is more than ε, we note the point to be lost in registration at this frame. Points that are lost several times during the registration are unlikely to be part of the actual segment. We prune the segment by removing such points from the border of the segment as outliers (figure 3.2). In certain cases, not just a few, but a vast chunk of points in the segment suffer from discrepancies $\geq \varepsilon$. We discard such segments altogether as incorrect. For each remaining segment S_j, we compute its *size* as the total number of mesh vertices in $\{\mathcal{V}^{t\in[1..n]}\}$ that are within the margin ε to its estimated position at the corresponding frame S^t_j. We sort the various segments in $\mathcal{P}$ according to their size in ascending order.

3.4 Achieving a Holistic Segmentation

We now describe a method for achieving a single set of segments for describing the entire sequence of 3D reconstructions. Our approach is to perform segmentation at multiple frames in the mesh sequence. We denote the holistic set of segments obtained as $\mathcal{P}$. We refine the segments within $\mathcal{P}$ using tracking, as described in the previous section. Some segments within $\mathcal{P}$ might be useless, several other segments would be repetitions and overlaps. We detect such cases and choose a subset of $\mathcal{P}$ as the ε-cover Ψ. We describe the steps involved in the following.

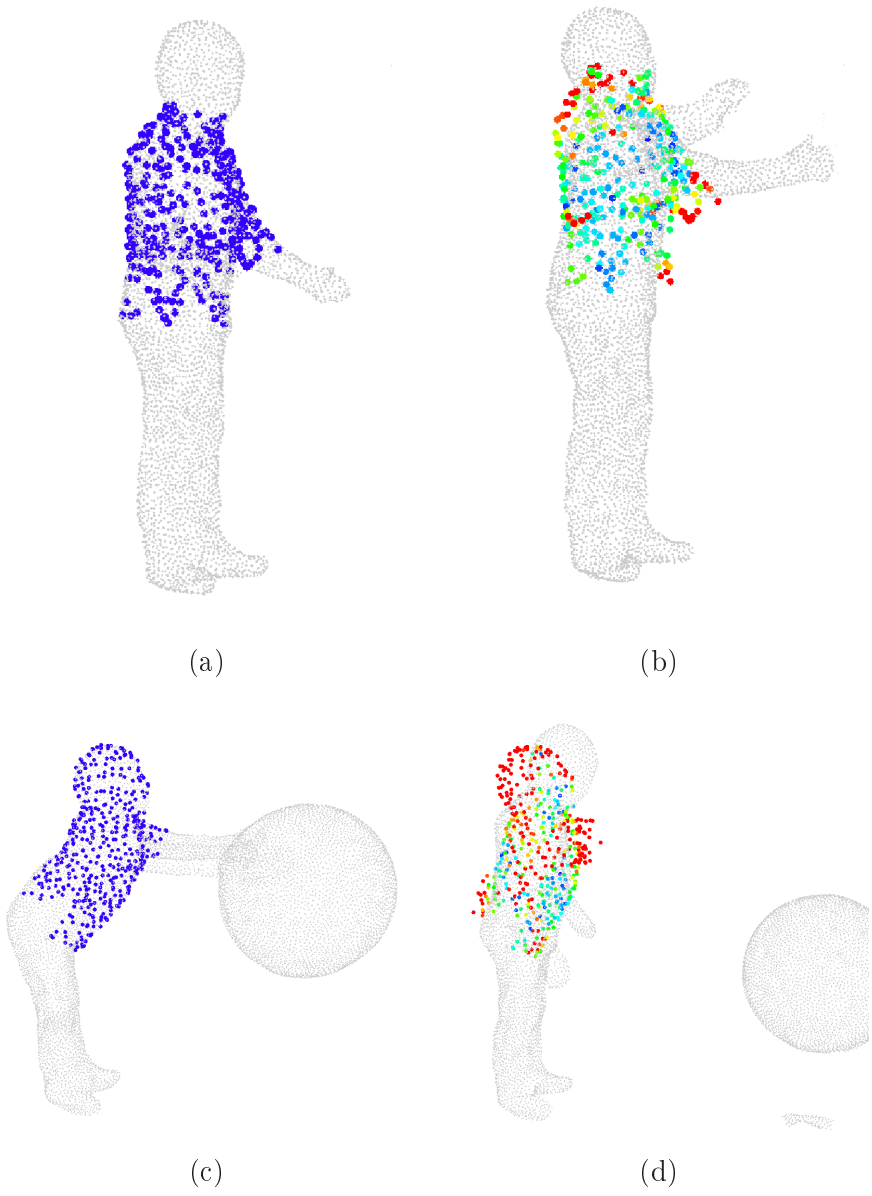


Figure 3.2: Reliability Estimation of Segments : (a) (c) - the segments detected at their seed-frames, (b) - outlier points with high discrepancy are pruned from the borders of (a), (d) segment is discarded after too many discrepancies in registration from (c). Points with higher discrepancy are shown in warmer colors - the red points are the outliers.

3.4.1 Detecting Segment Overlap and Repetition

We construct $\mathcal{P}$ by performing static segmentation on multiple randomly chosen frames. It is normally sufficient to segment just 3 to 5 frames, because many segments shall be detected repeatedly in all the frames. Following equation 3.1, a temporally coherent segmentation is given by an $\varepsilon - cover$ of segments chosen from $\mathcal{P}$. We take a greedy approach to obtain this.

We maintain the current set of accepted segments Ψ and gradually add new segments into it from $\mathcal{P}$ (Ψ is started out as empty). At each stage, we pop out the top element S_j in the sorted set $\mathcal{P}$ and check if it overlaps with any of the accepted segments S_a in Ψ. We call a point s_{ji} in S_j at seed-frame $t = k$ to be within the ε-*margin* of a segment S_a if it is within the distance of ε from the estimated positions of any of the points in S_a^k. We run this test from all accepted segments, and mark the points in S_j that are within the ε-margin of a prior segment. Examples are shown in figure 3.3-a,b. We define the overlap between two segments S_j and S_a as the fraction of points in S_j at its seed-frame $t = k$ that are within an ε-*margin* of S_a^k. If this overlap is large, we detect S_j as a repetition of S_a and proceed to the next segment. Due to the nature of the algorithm, it is the smallest of the segment repetitions that is acknowledged, the rest are discarded. This yields tighter segments, as can be seen in figure 3.3-c,d.

3.4.2 Building an ε-cover

If, on the other hand, no overlap is detected for segment S_j with any of the earlier segments, then Ψ is augmented by adding S_j to it. This process is terminated When every surface point in the mesh-sequence is acknowledged to be within an ε-margin of one or more segments of Ψ. The set of segments Ψ now defines an ε-*cover* for the sequence.

3.5 Results

We tested our approach primarily on visual hull reconstructions from multiple view silhouette data. These silhouettes are extracted from real images taken through a synchronized camera setup in an indoor setting. The *dance-girl* and *flashkick* sequences (figures 3.8 and 3.6-a) present fast movements of limbs that are difficult to be tracked.The *juggle* and *kids* sequences (figure 3.4) show human-object and human-human interactions respectively. In such interactive scenarios, it is very difficult to obtain prior knowledge of the scene, and hence difficult to make assumptions on the number of actors and the topology of these shapes. Our algorithm segments these scenes without making any such assumptions.

There are certain limitations for our approach. One interesting case is detailed in figure 3.9-a. The algorithm fails to track the juggler's club properly. When the club leaves the juggler's hand, our algorithm fails to follow the club and registers its points on the hand of the juggler. Then when a new and different club approaches

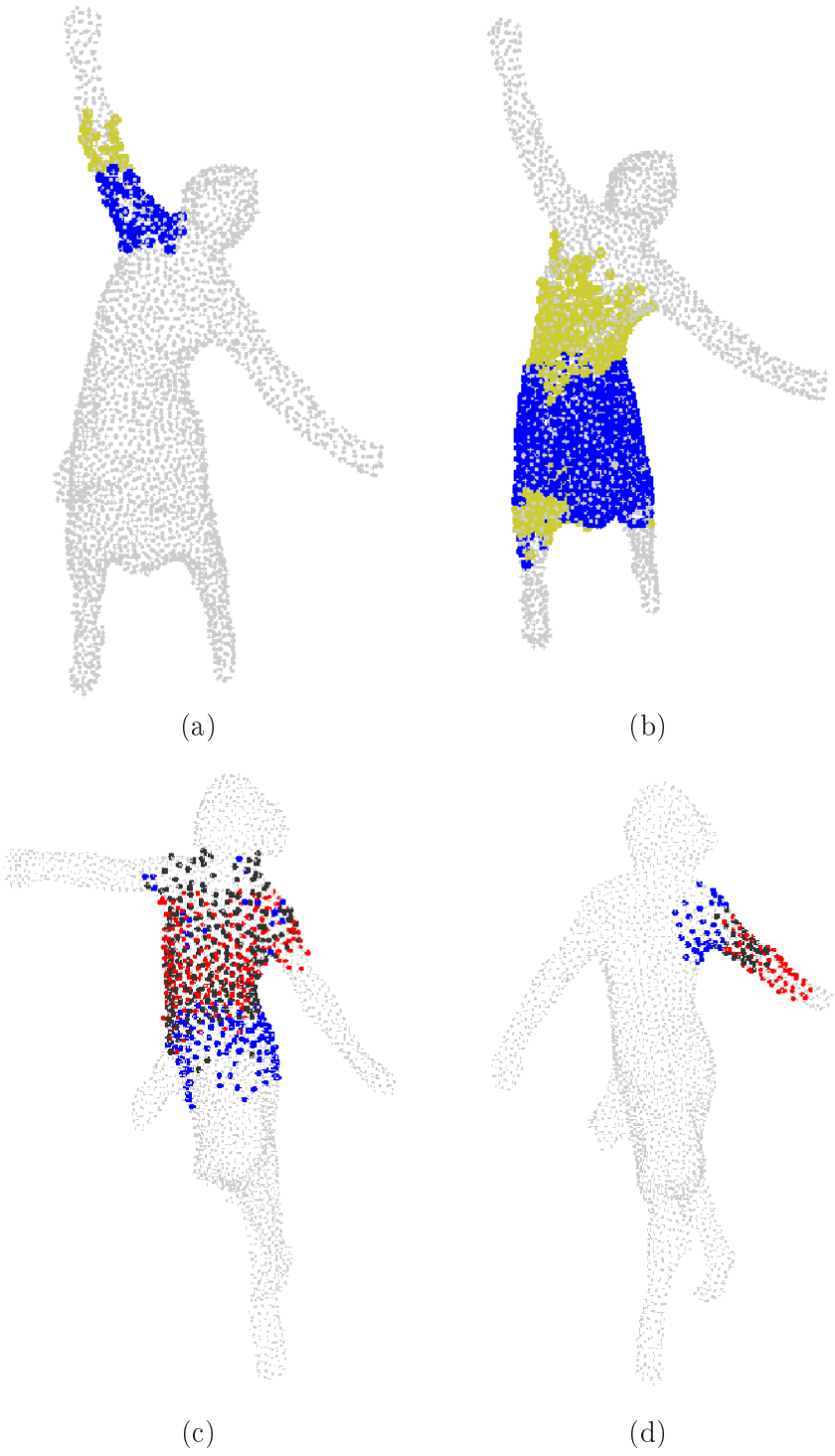
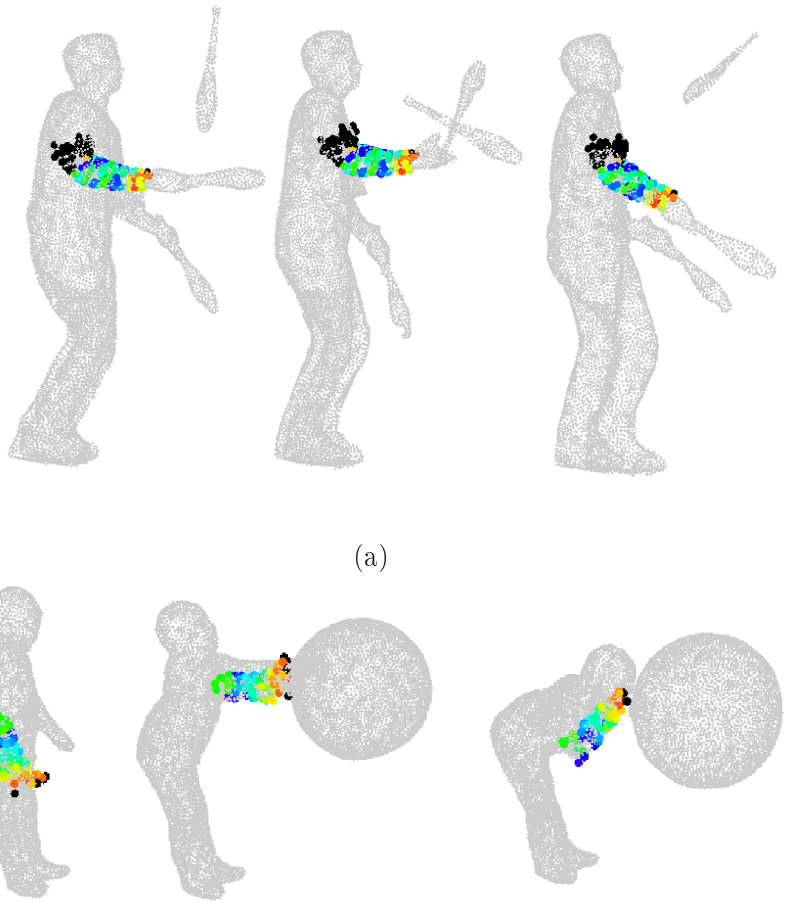


Figure 3.3: Comparing segments by overlap estimation : (a)(b) the points colored yellow are discarded as outliers, for being within an ε-margin of an earlier accepted segment. The remaining segment points are colored in blue. (c)(d) the blue segment is detected as a repetition of the red one, and is discarded



(a)

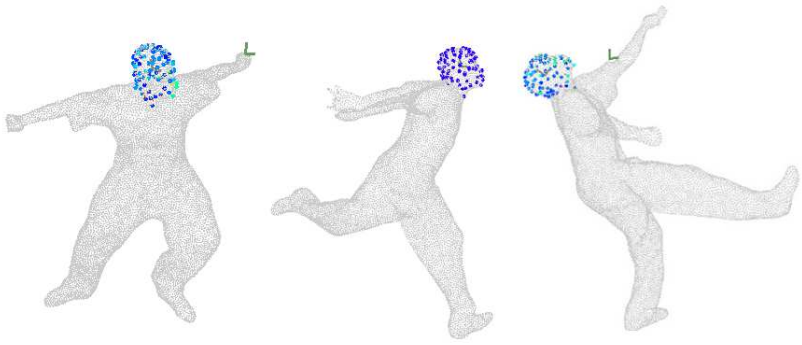
(b)

Figure 3.4: Tracking of a rigid component in a point cloud of diverse objects

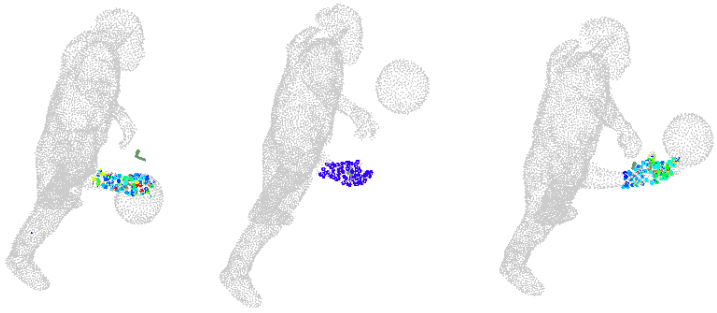
the hand, our algorithm registers these points onto the new club. A similar failure is presented in figure 3.9-b where the right foot of the actor has been confused with the left foot. The features that we use are not discriminative enough to handle cases like these, and the limitations of the ICP registration are manifest. In general, without a global model for tracking, it is difficult to track segments correctly for long sequences. This is especially true for small segments that are not sufficiently distinctive.

3.6 Discussion

In this chapter, we presented a novel algorithm for coherently segmenting a sequence of visually reconstructed meshes without making any assumption on the type, number or topology of the objects in the scene. Once such segments are identified, they



(a)



(b)

Figure 3.5: Tracking of a rigid component in a point cloud of diverse objects

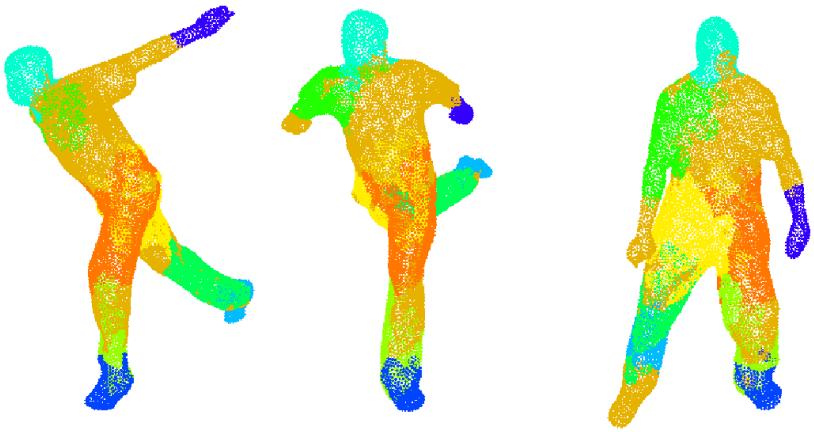


Figure 3.6: Temporally coherent segments on the *flashkick* sequence

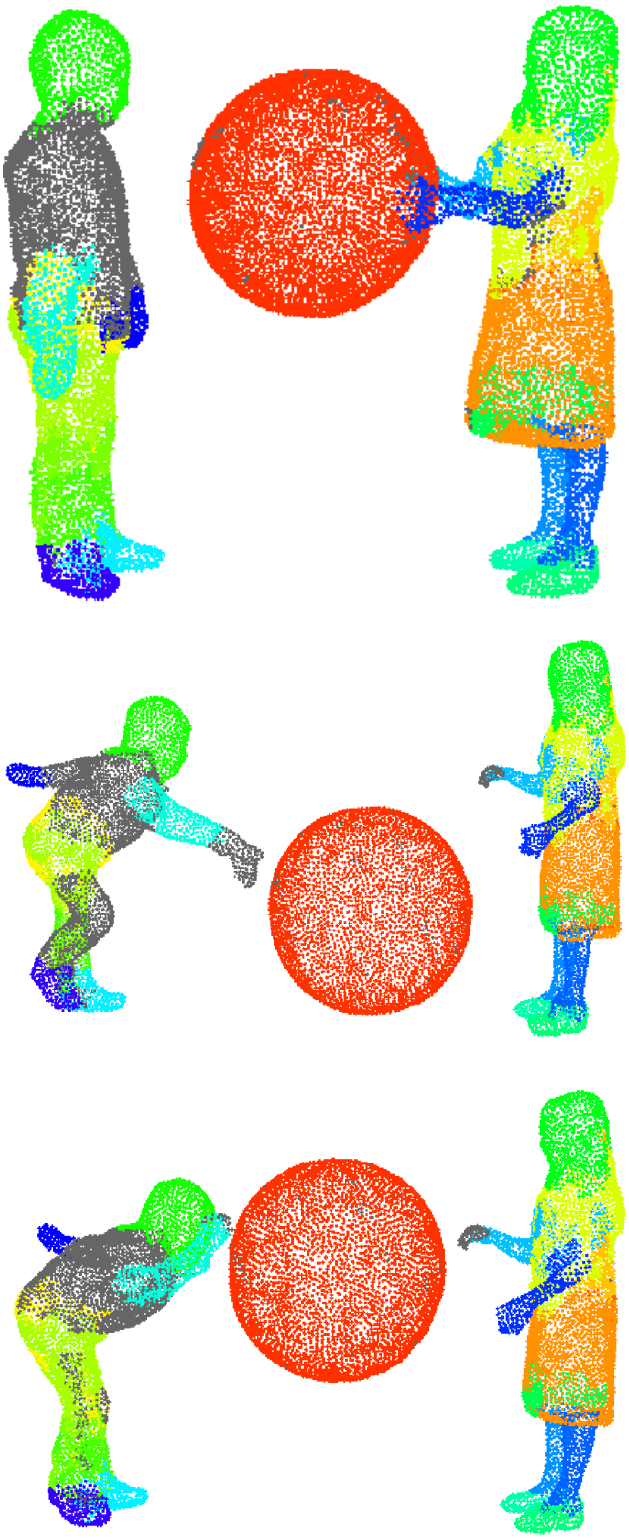
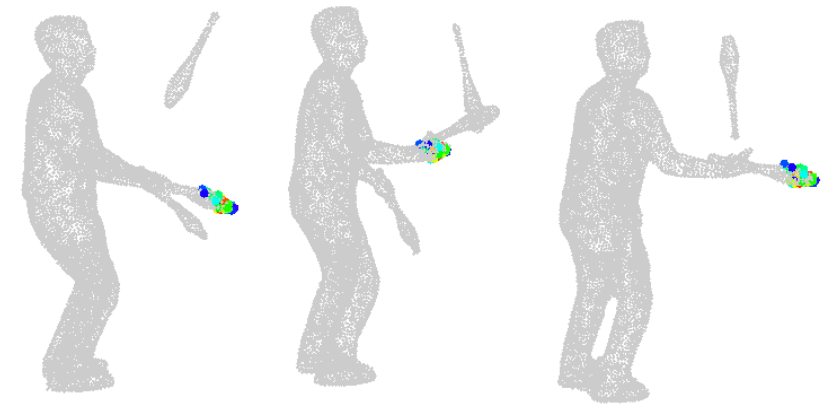
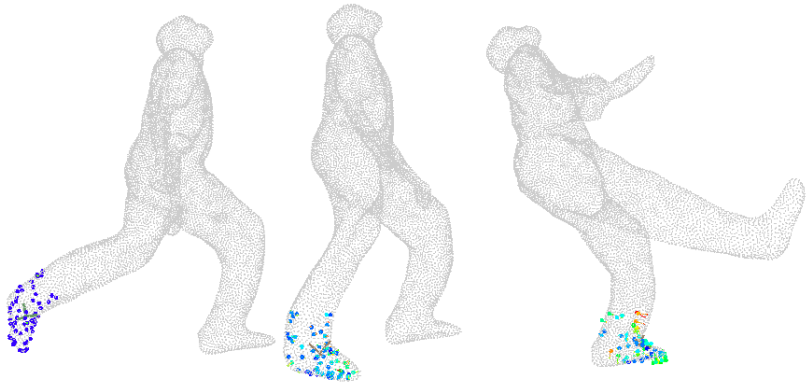


Figure 3.7: Temporally coherent segments on the *kids* sequence

Figure 3.8: Temporally coherent segments on the *dance-girl* sequence



(a)



(b)

Figure 3.9: Limitations of the algorithm : (a) the club is not registered correctly as it is difficult to distinguish it from objects in the neighborhood (b) the right foot of the person is wrongly registered to the left foot in the last frame, due to rapid motion

can provide a basis for learning the spatio-temporal model of the scene. Several potential applications await here to be explored. Our algorithm for registration is currently based on the ICP algorithm, and is thus limited to small displacements. In future work, we would like to overcome this limitation through stronger and more discriminative features for matching.

Chapter 4

Features for Matching Photometric Manifolds

Contents

In this chapter, we present an approach for computing features on visually reconstructed meshes for matching them across time. The underlying dynamic scene is composed of multiple objects that move in a non-rigid manner. The visual reconstructions of such a scene suffer from various artifacts, as explained in detail in chapter 1. In this scenario, it can be argued that motion can be estimated reliably only in a sparse manner, from only parts of the surface. We formulate this sparse motion estimation as a feature matching problem.

We present local and global shape features for photometric manifolds. We detect local features as scale-space extrema, by extending the Difference of Gaussians (DOG) operator from planar images to scalar functions on generic 3D manifolds. We compute local feature descriptors based on the Histogram of Gradients operator. We detect global features as surface protrusions and colour blobs, by analyzing the surface geometry and surface colour distribution respectively. We provide qualitative and quantitative results as validation. A part of the work presented in this chapter has been published earlier by us in [Zaharescu *et al.* 2009].

4.1 Introduction

Most natural shapes exhibit non-rigid deformations. Estimation of such deformations is a highly non-linear problem that is difficult to solve by a computer. However, there is often an interesting sub-space where this deformation of the natural shape can be captured and estimated in a tractable manner. For example, the motion of a human body can be understood with respect to the skeletal structure of the limbs. This motion lies principally in the space of articulation between the skeletal joints. However, non-rigid structures such as muscles, skin and clothes over the body distort this motion by varying degrees. This makes the problem harder. Another interesting subspace for capturing natural deformations is the space of isometric deformations - most natural shapes deform in a way which preserves metric properties on the surface such as areas and geodesic distances between points. Again, subtle non-rigid elements (for example, due to shearing on skin or clothes) distort this deformation from being a pure isometry. Most earlier works on shape matching have attempted to solve the motion estimation problem in a tractable sub-space, while addressing reasonably the subtle distortions caused by non-rigid effects.

However, the process of visual reconstruction of natural shapes into 3D meshes causes distortions more serious than the subtle non-rigid effects mentioned above. For example, topological inconsistencies in the reconstruction completely disrupt the metric properties of the surface. In this scenario, it can be argued that motion can only be estimated in a sparse manner and wherever it is done, it is also necessary to identify the scope - a distance beyond which the deformation is not valid. To this end, we formulate the motion estimation problem as a feature matching problem. The scale at which a feature is detected identifies the scope upto which the estimated deformation is relevant. Apart from overcoming the artifacts of the visual reconstruction process, this approach is suited for estimating relevant motion in cases where only a partial match is available between the surfaces; for example, when an actor enters into or disappears from the scene.

The main contributions of our work are

1. We propose *MeshDOG* - a local feature for photometric manifolds, by extending the Difference of Gaussian (DOG) operator from 2D images to 3D meshes

2. We propose *MeshHOG* - a histogram of gradients (HOG) feature descriptor computed in the surface neighborhood of a feature point on a 3D mesh

3. We propose global features - surface protrusions and color blobs, that can be used for coarse matching of surface regions.

The remainder of the chapter is organized as follows. In section 4.2, we review the related work. In section 4.3.1, we describe our method for detecting local feature points on the shape. In section 4.3.2, we give the method for describing these features using photometric information. In section 4.4, we introduce global shape features - surface protrusions and colour blobs. We provide qualitative and quantitative results in section 4.5. We discuss and conclude in section 4.6.

4.2 Related Work

Our work is related to the estimation of geometric and photometric features on 3D shapes. In this section, we review the related literature in its broader context of finding correspondences between non-rigid shapes. [van Kaick *et al.* 2010] have recently published a good survey of various existing methods on shape correspondence. Broadly these methods can be differentiated into global deformation methods and feature matching methods. We briefly review both classes of methods in the following.

4.2.1 Shape Correspondence by Global Deformation

Shape Invariants for Matching : Most earlier works have treated the problem of matching non-rigid shapes in 3D as a global shape matching problem through the use of invariants. [Zhang & Hebert 1999] use harmonic maps to achieve 3D rotational invariance. [Hilaga *et al.* 2001] discover inherent skeletal structure of the shape as reeb graphs and use them for matching shapes of identical topology. [G *et al.* 2002] use multidimensional scaling on geodesic distances between mesh vertices to project the data into an invariant space for texture mapping. This approach is generalized by [Bronstein *et al.* 2006a, Bronstein *et al.* 2007b] as generalized multi-dimensional scaling (GMDS) to match shapes over near-isometric deformations. [Zhang *et al.* 2007] convert the local mesh structure into the spectral domain and match shapes over isometric deformations. All these approaches require the shapes to have the same topology, which cannot be assumed in our problem setting. In a later work, [Bronstein *et al.* 2007a] discuss using the heat diffusion distance which is less sensitive to topological artifacts.

Global Shape Features : The computation of these distances across all pairs of points results in a computationally expensive task. Further, it produces a dense matrix which contains a lot of redundant information. It is possible to compute these distances only within the local neighborhood of each point, and thus produce a sparse matrix. For example, the local geometry in the 1-ring of a mesh point can be represented by the Laplace-Beltrami operator, and this information can be aggregated into a sparse Laplacian matrix. [Mateus *et al.* 2008] show that the eigen vectors of this matrix (called the Laplace-Beltrami Eigenfunctions) capture certain crucial geometric properties such as the protrusions of the shape. [Hu & Hua 2009] also analyze this spectral space to identify feature points on the shape at several levels. But [Gebal *et al.* 2009] have argued that these Laplacian Eigenfunctions suffer from various limitations, most importantly that they are not local in their scope and often contain information from very distant regions in the shape. Further, they suffer from noise disturbances on the underlying topological structure of the graph.

To avoid such limitations, [Bronstein *et al.* 2009a] compute the *diffusion distance* on the graph, according to the properties of dissipation of heat over time.

The *diffusion distance* is computed as the average time that is required for heat to reach the target point from the source point, over several random walks on the graph. This distance is relatively stable to minor variations in graph topology. Such diffusion distance may be computed over either the 3D mesh or over discrete graph representations of the medial axis points. [Gebal *et al.* 2009] and [Sun *et al.* 2009] have independently proposed a way to reduce the redundancy of the information in the matrix of diffusion distances. The *auto diffusion function* (ADF) or the *heat-kernel signature* (HKS) computes the diffusion distance from a point back to itself by traversing the graph structure. In other words, they analyze the diagonal vector of the diffusion distance matrix, which can be visualized as a scalar function defined over the graph. The local maxima of the ADF identify the protrusions of the shape, because in the words of [Gebal *et al.* 2009], "*... for a family of metric spheres centered at x, the ratio of volume of metric sphere to its surface area will be bigger for a feature point (protrusion) than for a point on a flat part of the shape ...*". These maxima can be analyzed at several scales (simply by varying the parameter of time when computing the diffusion), and they yield shape protrusions at several scales.

Matching as Global Deformation : A few works have formulated the shape matching problem as the estimation of a global deformation between shapes. [Mateus *et al.* 2008] convert the shapes into spectral domain using locally linear embedding (LLE), where they can be aligned with a simple isometry transformation. They estimate this using the spectral features of Laplacian eigenfunctions of the shapes. [Zhang *et al.* 2008] formulate shape matching as the problem of deforming the first shape into the second, and obtain correspondences by estimating the minimum deformation. [Lipman & Funkhouser 2009] provide a more efficient variant for genus-0 shapes through the use of spherical reparametrization, where just 3 point matches are sufficient to obtain the deformation between shapes. They obtain the required 3 feature matches through a RANSAC based voting procedure. [Starck & Hilton 2005] also use spherical parametrization but estimate dense motion for the mapped points using optical flow. [Zeng *et al.* 2010] extend the work of [Lipman & Funkhouser 2009] to higher genus shapes by using conformal parametrization of the shapes to derive dense matching from 3 feature matches. But instead of using RANSAC, they formulate a high-order graph matching. When a shape exhibits symmetry, these approaches might not be able to resolve the ambiguity between the matches and their symmetric counterparts. [Thorstensen & Keriven 2009] couple photometric information with geometric information and solve shape matching through a variational approach. Assuming that the deformation is isometric (or near-isometric) limits the applicability of these works to data exhibiting topological artifacts. One option is to ignore mesh connectivity altogether. Point based approaches, e.g. [Chui *et al.* 2004], register sets of points as a global deformation but do not account for mesh connectivity. This approach is applicable to many cases but ignores useful shape information at places where it is reliable.

4.2.2 Shape Correspondence by Feature Matching

Local 3D Shape Features : Several geometric shape descriptors have been proposed for solving the problem of 3D object retrieval from a database of models, [Bustos *et al.* 2005] provides a good survey. [Johnson & Hebert 1999] have proposed spin images - histograms of point distribution around a point of interest as a feature descriptor. These histograms are computed in cylindrical bins around the normal of the vertex, and are invariant to local rotation. [Mori *et al.* 2001] have proposed shape contexts - computing similar histograms of point distributions in spherical bins. The spherical harmonics of these histograms are invariant to rotation of the shape. Please note that spin images and shape contexts can also be computed globally for the entire shape, with respect to the object center. But such descriptors are invariant only to rigid transformations that limit their applicability. So it is better to compute these features locally at various interest points and match them exhaustively over the candidate shapes. The selection of these interest points can be random, or based on geometric criteria. [Novotni *et al.* 2005] detect such interest points on a 3D voxel grid as local extrema of the scale space Laplacian-of-Gaussian. [Hu & Hua 2009] detect salient mesh regions as extrema in the Laplace-Beltrami spectral domain of the mesh, and possess the advantage of having the localization invariant to isometric deformations. [Schnabel *et al.* 2007] detect geometric primitives such as cylinders, spheres, cones etc. directly in the scene and use them as regions of interest. If the features are used for object retrieval, the retrieval performance of each feature can be measured and used as a criterion for selecting and localizing features, as done by [Shilane & Funkhouser 2006]. More closely related to our approach, [Schlattmann *et al.* 2008] propose scale-space extrema based on the averaged mean curvature flow. Alternatively, [Novatnack & Nishino 2007] define the scale space in a planar parametrization of the surface using the normal information and search for its extrema. Photometric information is not taken into account by these methods.

Local Photometric Features : In the problem of shape correspondence for photometric manifolds, many approaches have conventionally used photometric features computed in the captured images (for instance [Lowe 2004, Dalal & Triggs 2005, Bay *et al.* 2006]) by back-projecting them onto the surfaces. Feature detection on images has been well-investigated in computer vision (see [Mikolajczyk & Schmidt 2005, Aanaes *et al.* 2010] for good reviews). Some spatio-temporal features have also been proposed for video [Wong & Cipolla 2007, Kläser *et al.* 2008]. However, fewer works exist for computing photometric features on 3D shapes. In [Wu *et al.* 2008] a SIFT-based descriptor on 3-D oriented patches is proposed, i.e., VIP (Viewpoint Invariant Patches), which was used for 3D model matching. It constitutes a first attempt to devise a descriptor that includes both geometry (normal orientation) and photometric information. Our approach is similar in spirit to them, but extended to consider full 3D gradients and histograms.

Spatial Relationship between Features : Now we review related works on the 3D spatial grouping of features for matching them across shapes. The simplest method of using spatial information is to specify the relative location of a feature with respect to the object centre, and use this information for matching, as done by [Li *et al.* 2008]. [Shilane & Funkhouser 2006] use an additional term that is given by the angle made by the normal vector at a feature to the vector joining the feature to the object centre. These methods are not applicable for matching non-rigid shapes because both the object centre and the relative location of a feature with respect to it change over non-rigid deformations. Instead of using the object centre, relative geometry can be represented as pairwise relationships between features, such that the features move together rigidly. Normally, these rigidity constraints are modeled using two error terms - *stretch* which denotes the length of the vector joining the features and *twist* which denotes the relative orientation of the normal vectors at the two features with respect to the vector joining them. These pairwise relationships have been used by [Shilane & Funkhouser 2006] to derive geometrically coherent matches.These relationships are preserved exactly only under rigid transformations. If it is possible to approximate a non-rigid deformation as piecewise rigid-transformations, then such relationships can be modeled at a local scale. [Leordeanu & Hebert 2005] propose a spectral approach for obtaining a coherent set of matches that respects pairwise constraints. They detect clusters of strongly connected features that match coherently, and thereby are robust to outliers.

Global Shape Correspondence : [Starck & Hilton 2007a] try to blend local and global approaches, by computing features at every vertex on the shape and trying to obtain dense correspondence between the features. They define a graphical model of points all over the mesh connected with each other by edges preserving local rigidity and surface stretch. They formulate shape matching as a graph labeling problem with labels given by vertices of the second shape; they obtain a locally optimal solution of this problem belief propagation. [Chang & Zwicker 2008] propose another variant of using a graphical model for matching, by estimating a putative set of rigid transformations between points across the shapes, and treating these transformations as labels instead. The disadvantage of both these approaches is that they are slow, and require features defined ubiquitously on the shape, even over uninteresting and monotonous stretches. Secondly, they are even slower when they are set to obtain partial matches, as when new actors enter the scene. They normally require a good initialization lest they get stuck in a local minimum.

Our work in this chapter falls into the class of feature matching methods. We provide efficient features for matching between visually reconstructed meshes. We describe both local and global features that complement each other. Such features may be used by approaches similar to [Starck & Hilton 2007a] to obtain dense shape correspondence. However in this thesis we are more interested in modeling temporal sequences, and we employ these features in chapter 5 to derive dense point trajectories over time.

4.3 Local Shape Features

In this section, we describe our method for detection and description of local shape features. A more elaborate description of our method is given in our paper [Zaharescu *et al.* 2009]. Here, we present it from the specific perspective of matching photometric meshes over non-rigid deformations. The exterior of the shape contains two types of information based on the appearance of the shape and on the local surface geometry. It is possible to treat these information disjointly and detect features on either of them. For example, photometric information can be read from the 2D images captured from multiple-views, and features of interest (for example, SIFT [Lowe 2004] or SURF [Bay *et al.* 2006]) can be detected on the images. Similarly, geometric features such as spin-images or 3D shape contexts can be detected on the surface of reconstructed meshes. However, both photometric 2D features and geometric 3D features have limited informative capabilities with respect to the potential richness of the data available in visual reconstructions. Further, treated independently, both these information have limitations. Image information is only partially robust to motions, and several unstable features may be detected on the 2D contours of a 3D shape. Surface geometric information has several artifacts, as explained in detail in section 1.2.3. Because of these problems, we need to handle both photometric and geometric information in a consistent and simultaneous manner.

Photometric information can be considered as a scalar function ϕ defined on a 2D manifold that is the surface of the mesh. Such photometric manifolds are a generalization of planar image domains (2D images) to non-planar domains. The local properties of the function ϕ can be analyzed at various scales using scale-space theory, and interest points be detected on the surface through this analysis. The *Difference of Gaussians* (DOG) operator is popular for analyzing the photometric scalar-field on a 2D image. We extend this framework to generic 2D manifolds such as meshes and call the detected points of interest as *MeshDOG* features. Such features have two principal attractions as compared to features detected on 2D images (a) they suffer from no perspective distortions (b) they are no false detections due to occlusions. We associate a photometric descriptor for each feature, based on the computation of the histogram of gradients in the neighborhood, which we call *MeshHOG*.

4.3.1 Feature Detection on Photometric Manifolds

We consider the analysis of a scalar field ϕ defined on a 2D manifold in $\mathcal{R}^3$. We consider the discretization of such a manifold as a uniformly triangulated mesh $\mathcal{S}$, whose facets are triangles of approximately the same size and whose vertices' valence is close to 6. Such a uniform mesh can be obtained from a non-uniform mesh through simple mesh operations, as proposed in [Kobbelt *et al.* 2000]. This absolves us of the necessity of complex techniques that ensure proper samplings of the scalar field, while keeping generality. It is interesting to notice that an image can be viewed as a "flat" uniformly sampled mesh, i.e, a grid of vertices with valence 4 and whose

facets are squares or rectangles.

Let $\nabla_S \phi$ denote the gradient operator of ϕ on S, the directional derivative of ϕ at $v \in S$, for any direction $\overrightarrow{u}$ in the tangent plane of S at v, is defined as

$$D_{\overrightarrow{u}}\phi(v) = \nabla_S \phi(v).\overrightarrow{u} \tag{4.1}$$

For a discretized manifold, we define the n-ring neighbourhood $N_n(v_i)$ around a vertex v_i as to consist of vertices $\{v_j\}$ belonging to all the set of rings $\{rg(v,i)\}_{0 \leq i \leq n}$. Then, the discrete version of the gradient operaor $\nabla_S \phi$ is obtained by aggregating the directional components from all the vertices in the neighborhood of a vertex v_i as a weighted sum

$$\nabla_S \phi(\mathbf{v}_i) = \sum_{v_j \in rg(v_i,1)} (w_{ij} D_{\overrightarrow{e_{ij}}} f(\mathbf{v}_i))\ \overrightarrow{u_{ij}}, \tag{4.2}$$

where the w_{ij} weighs the contribution of $D_{\overrightarrow{e_{ij}}}$ and $\overrightarrow{u_{ij}}$ is the normalized projected direction of $\overrightarrow{v_i v_j}$ in the tangent plane at v_i.

If S is uniformly sampled, the various neighbors around v_i are equally spaced and all the weights can be taken as equal $w_{ij} = \frac{1}{val(v_i)}$ where $val(v_i)$ is the valency of v_i. For non-uniformly sampled meshes, the weights are a function of the angles between the directions $\overrightarrow{u_{ij}}$ around v_i in the tangent plane at v_i.

We then define the discrete convolution operator on the mesh of the scalar field ϕ with a kernel κ as

$$(\phi * \kappa)(v_i) = \frac{1}{K} \sum_{v_j \in N_n(v_i)} \kappa(||\overrightarrow{v_i v_j}||)\phi(\mathbf{v}_j), \tag{4.3}$$

where the kernel weighs the participation of neighbouring vertices v_j as a function of their distances from vertex v_i and $K = \sum_{v_j \in N_n(v_i)} \kappa(||\overrightarrow{v_i v_j})$ is a normalization factor. For the case of a discrete uniformly sampled mesh, the contributions of the neighbouring vertices are v_j are equally weighted with respect to their spatial arrangments.

For the scale-space analysis of the scalar field ϕ on the mesh, we use its convolutions with the Gaussian kernel, defined as

$$g_\sigma(x) = \frac{\exp(-x^2/2\sigma^2)}{\sigma\sqrt{2\pi}}.$$

The scale space of ϕ is built progressively: $\phi_0 = \phi$, $\phi_1 = \phi_0 * g_\sigma$, $\phi_2 = \phi_1 * g_\sigma$ etc. This scale-space representation capture both the scalar field (photometric information) and the surface geometry in a succinct fashion. The scale space of an example shape is shown in figure 4.1.

The interest points are defined by using the *Difference of Gaussians (DOG)* operator which takes the difference between different scales of the above scale-space representation $DOG_1 = \phi_1 - \phi_0$, $DOG_2 = \phi_2 - \phi_1$ etc.

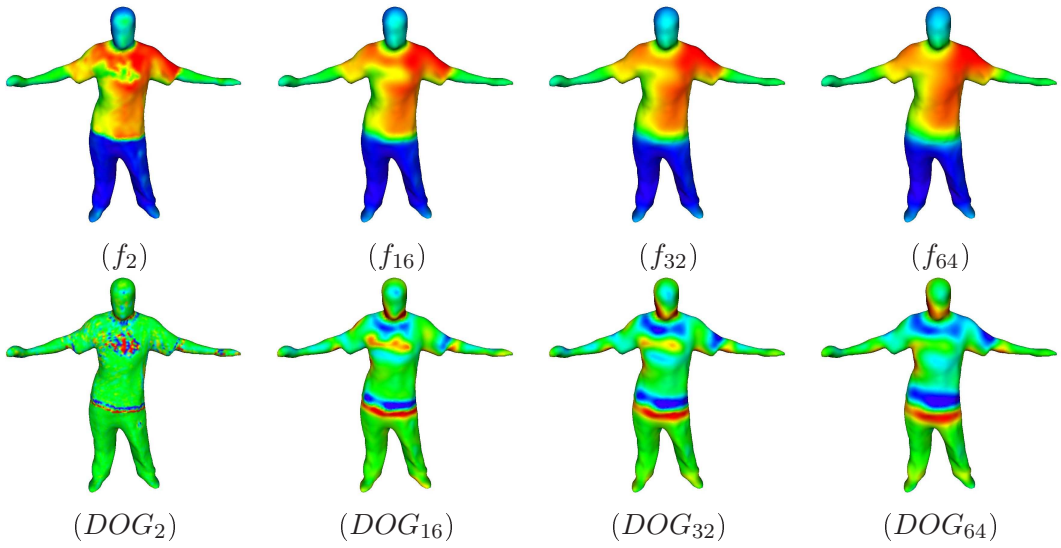


Figure 4.1: Scale space representation with the function ϕ being the color of mesh vertices : the shape is frame 30 from the *pop2lock* sequence.

An important observation is that when building the scale-space representation, the mesh-geometry is not changed, but the different scalar functions defined on the mesh change : $\phi_1, \phi_2, DOG_1, DOG_2 \ldots$. Each octave of $\sigma_0 = e_{avg}$ is subdivided into three steps choosing $\sigma = 2^{\frac{1}{3}}\sigma_0 = 2^{\frac{1}{3}}e_{avg}$. In the 2D image case, after covering an octave, the original image is subsampled from the first, before continuing to the next octave. In triangular meshes, a similar effect can be obtained through mesh simplification operations, by imposing the new average edge to be $e_{avg} * 1.5$. However, due to the cost of the mesh-simplification operation, we choose not to simplify the mesh. Instead we continue convolving with the same original kernel, keeping in mind that, in order to cover another octave, twice as many convolutions are necessary.

The feature points are detected as the maxima of the scale space across scales, followed by non-maximal suppression using the 1-ring neighbourhood in the current and the adjacent scales.

Thresholding From the extrema of the scale space, only the top $\beta = 5\%$ of the maximum number of vertices are considered, sorted by the magnitude of DOG_i. A percentage value for threshold keeps the detector flexible irrespective of the scalar function under consideration, without the need for individual normalizations.

Selection of stable points We retain features that exhibit corner characteristics, which is given by the Hessian operator. We first define the gradient plane at a vertex, as given by the gradient direction for $\overrightarrow{x}$ and its orthogonal direction for $\overrightarrow{y}$. We compute the 2nd order directional derivatives along these directions into the Hessian matrix, and then compute the Eigen values of this matrix. The ratio of the larger

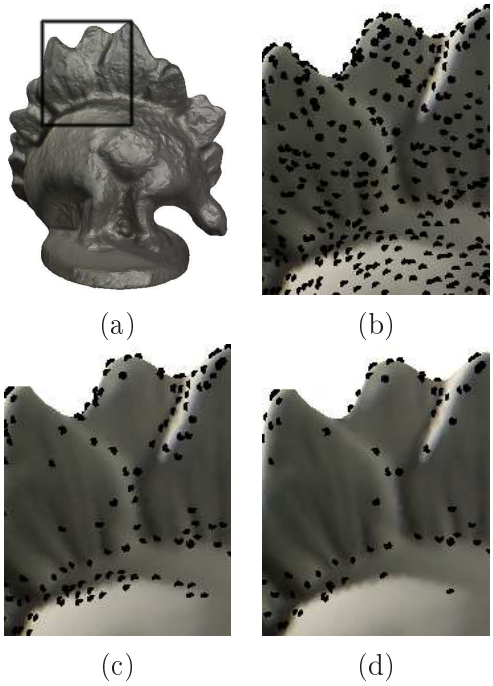


Figure 4.2: Feature detection shown with photometric data. (a) The original mesh has 27240 vertices. (b) There are 5760 extrema detected. (c) After thresholding there are 1360 vertices left. (d) 650 vertices with "cornerness" are eventually retained.

Eigen value λ_{max} to the smaller Eigen value λ_{min} is a good indication for the corner response. We typically use $\frac{\lambda_{max}}{\lambda_{min}} = 10$ as a minimum value to threshold responses.

Notion of Scale The scale at which a feature point is detected is given by the number of rings over which the discrete convolution is computed. The vertices belonging to all these rings provide the region of support for the computation of the corresponding feature descriptor, the process of which is described in section 4.3.2. In a generic non-rigid motion of the shape, there is no guarantee that all the points in this region of support suffer from exactly identical motion over time. This is especially true if the region of support is distributed across neighboring body-parts (such as the limbs of articulation of a human body). However, when the region of support is sufficiently small, this approximation can be justified. So we select most of the feature points at the scale of 1-ring neighborhood. The artifacts of visual reconstruction (illustrated in detail in chapter 1) also serve as caution against using large regions of support for these local features.

However, if the objective is to match the objects only across similarity transformations (induced by rigid motions and scaling), then all the scales can be used for feature detection and the associated regions of support be used for feature description.

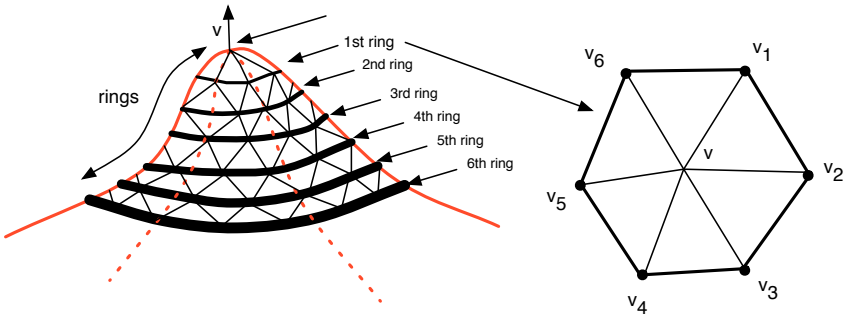


Figure 4.3: A vertex v and its rings (left) and the first ring of v (right).

4.3.2 Photometric Feature Description

As is commonly done in image based feature descriptors, we describe each feature point based on the photometric gradient information around it. We consider the surface neighborhood of a feature point on the 3D mesh. The gradient of a scalar function on such a surface yields a 3D vector. We discretize the 3D space into bins, and compute the histograms of these gradients in the neighborhood region.

This region is defined using a neighbourhood ring size r, as depicted in Figure 4.3. For each vertex from the neighbourhood $v_i \in N_r(v)$, the gradient information $\nabla_S f(v_i)$ is computed using (4.2). As a first step, a local coordinate system is chosen, in order to make the descriptor invariant to rotation. Then, a histogram of gradient is computed, both spatially, at a coarse level, in order to maintain a certain high-level spatial ordering, and using orientations, at a finer level. Since the gradient vectors are 3 dimensional, the histograms are computed in 3-D.

Neighborhood Size The number of rings r for the support region is chosen adaptively based on a more global measure, such that the descriptor is robust to different spatial samplings and to scaling. The value of r is chosen such that it covers a proportion α_r from the the total mesh surface, where $\alpha_r \in (0, 1)$. By denoting A_S as the total surface area of the mesh S, which can be computed as the sum of all triangle areas, the ring size r is:

$$r = round\left(\frac{1}{e_{avg}}\sqrt{\frac{\alpha_r A_S}{\Pi}}\right), \tag{4.4}$$

assuming that the surface covering the ring neighbourhood can be approximated with a circle and that the mesh S is equally sampled, with the average edge size e_{avg}. In practice, we use an r corresponding to $\alpha_r = 1\%$.

Local Coordinate System A local coordinate system can be devised using the normal $\vec{\mathbf{n}}_v$ and two other unit vectors, residing in tangent plane $\mathcal{P}_v$ of v. Given a unit vector $\vec{\mathbf{a}}_v \in \mathcal{P}_v$, the local coordinate system is given by $\{\vec{\mathbf{a}}_v, \vec{\mathbf{n}}_v, \vec{\mathbf{a}}_v \times \vec{\mathbf{n}}_v\}$. Vector $\vec{\mathbf{a}}_v$ is computed as the direction associated to the dominant bin in a polar histogram, with $b_a = 36$ bins. The histogram is computed by considering the projected vertices v_i in $\mathcal{P}_v$ and taking into account their gradient magnitudes. We

weigh $||\nabla_S f(v_i)||$ by a Gaussian with $\sigma = e_{avg} * r/2$, based on the geodesic distance from v. In order to reduce aliasing and boundary effects of binning, votes are interpolated bilinearly between neighbouring bins when computing the histograms. We use the same weighting and interpolation technique for any further binning.

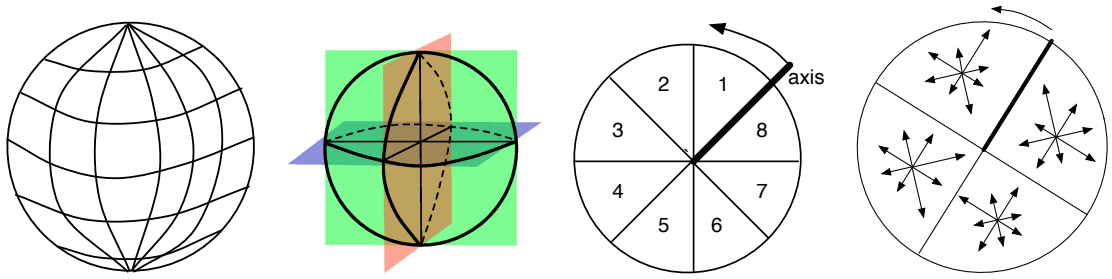


Figure 4.4: (a) 3-D Histogram - polar mapping used for creating histograms via binning of 3-D vectors; (b) Choosing 3 orthogonal planes onto which to project the 3-D Histogram. (c) Polar Coordinate system used for creating histograms via binning of 2-D vectors, shown in this example with 8 polar slices. (d) Example of a typical spatial and orientation histograms, using 4 spatial polar slices and 8 orientation slices.

Histograms Instead of computing full 3-D orientation histograms, as proposed in [Kläser *et al.* 2008], we project the gradient vectors to the 3 orthonormal planes, describing the local coordinate system. This provides us with a more compact representation of the descriptor. For each of the three planes, we compute a 2 level histogram. Firstly, the plane is divided in $b_s = 4$ polar slices, starting with an origin and continuing in the direction dictated by the right hand rule with respect to the other orthonormal axis vector. When projected onto the plane, each vertex v_i will fall within one of the spatial slices. For each spatial slice, we compute orientation histograms with $b_o = 8$ bins for each of the projected gradient vectors $\nabla_S f(v_i)$ of the vertices v_i that projected onto that spatial slice, as shown in figure 4.4(d).

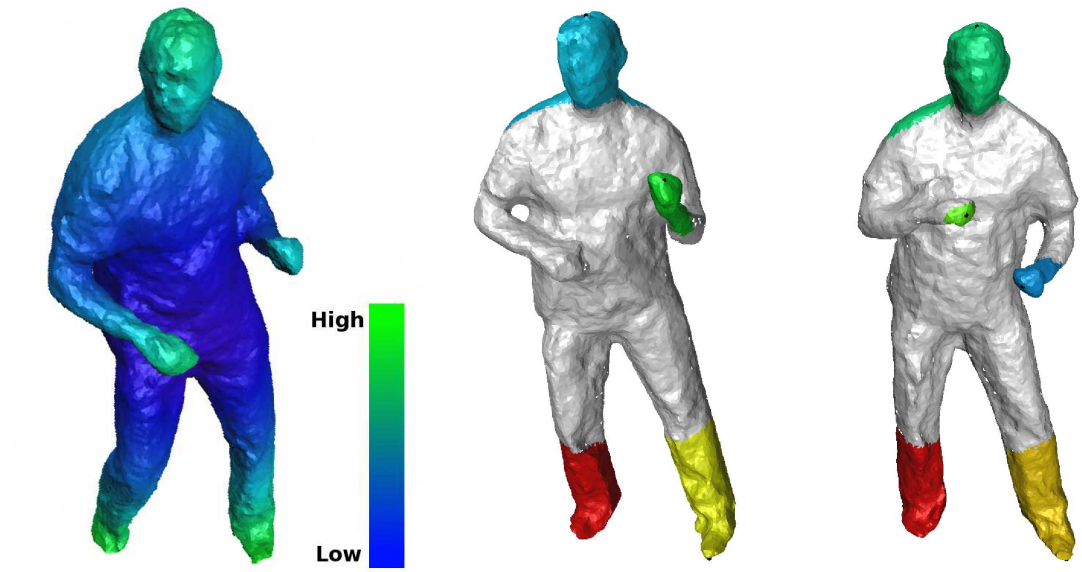
Descriptor The final descriptor is obtained by concatenating $b_s \times b_o$ histogram values for each of the three planes, followed by L^2 normalization.

4.4 Global Shape Features

By global shape features, we mean features at specific locations whose scope contains the entire shape. These features represent pertinent information about the shape at a coarse level. We discuss two such features - surface protrusions and colour blobs.

4.4.1 Surface Protrusions

We have briefly mentioned surface protrusions in chapter 2. They are detected as the local maxima of the geodesic eccentricity function [Hilaga *et al.* 2001] , defined



(a) (b)

Figure 4.5: (a) Distribution of the geodesic eccentricity function $\mu_n(v)$ on a shape (b) Detection of surface protrusions by thresholding $\mu_n(v)$. Protrusions are not detected over topological collapse.

formally as

$$\mu(v) = \int_{u \in \mathcal{S}} d_{\mathcal{S}}(u, v) dS \quad , \quad \mu_n(v) = \frac{\mu(v) - Min_{u \in \mathcal{S}} \mu(u)}{Max_{u \in \mathcal{S}} \mu(u)}$$

where $d_{\mathcal{S}}(u, v)$ denotes the geodesic distance between the points P and V and $\mu(V)$ is defined as the sum of the geodesic distances from V to all points on S. After normalization, $\mu_n(V)$ provides a continuous function whose value indicates the apparent nearness of a point to the center of the object. Figure 4.5-(a) illustrates the distribution of this function over the sample 3D mesh. $\mu(V)$ is alternatively known in the literature as the normalized geodesic integral or as the average geodesic distance.

Our method for estimating $\mu(v)$ is similar to that of [Hilaga *et al.* 2001] but with certain differences. We estimate $\mu(v)$ in an approximate manner, by selecting points through sampling the mesh surface uniformly. We assign a patch neighborhood to each selected point (called a patch-centre). Thus the surface of the mesh is divided into a set of overlapping patches. We compute geodesic distances between the centres of neighboring patches exactly, and then use the graph-distance on the patch graph to approximate geodesic distance between faraway points. We use the well-known Warshall's algorithm to compute graph distances between all pairs of points. However, Warshall's algorithm has a complexity of $O(n^3)$ on the number of patch-centres, and can take a long time to compute for large n. This is particularly true when the reconstructed mesh has a large number of vertices, and we don't want to sample the patch-centres too aggressively. In such cases, we use an alternative

approach by first simplifying the mesh such that all the edges are approximately of a certain length, and then using breadth-first search on the mesh to propagate a front from each patch-centre. This operation can be implemented very efficiently, and scales only linearly with increasing n. Finally, we compute $\mu(v)$ in the above equation, by replacing the continuous integral with a discrete sum over the patches.

The maxima of $\mu(v)$ correspond to the extremities of the object. It is possible to detect them at various scales, in the manner of [Gebal *et al.* 2009] [Sun *et al.* 2009]. But due to the poor quality of the reconstructions, only the features detected at very coarse scales have relevance. We detect global extremities of this function by simply imposing a threshold on $\mu(V)$, and labeling points which lie above this threshold. Such points lie on compact clusters, typically corresponding to the different protrusions of the object. For example, on the shape of a human being, the four limbs and the head are detected in this fashion (as shown in figure 4.5-b). Amongst the labelled points in each cluster, we identify the point with the maximal value of $\mu(V)$ as the locus of the protrusion.We consider a geodesic neighborhood around this point, upto a range that is selected as a fraction of the maximal geodesic distance on the surface. $\gamma * Max_{u,v \in \mathcal{S}}\, d_{\mathcal{S}}(u,v)$. This geodesic neighborhood gives the region of support for the protrusion, over which estimate a feature descriptor using photometric information (described in the next sub-section).

It should be noted that surface protrusions are not detected at all times. Topological inconsistencies in the reconstruction, can cause a protrusion to disappear. For example, in figure 4.5-b, the right hand of the hand is clubbed with the stomach in the reconstructed mesh, and thus not detected as a protrusion.

4.4.2 Color Blobs

To compensate for the cases of missing protrusions, and to take advantage of the photometric information on our meshes, we introduce a global shape feature based on the colour distribution on the mesh. We first compute the key colours in the reconstructed scene as follows. We quantize the RGB space of the colour cube into bins of equal size, and compute the frequency of each bin along the reconstructed vertices of the scene. We associate each mesh vertex to the closest bin, based on the $2-norm$ difference in RGB values (each normalized to between 0 and 1). This yields a histogram describing the probability of occurrence of each colour over the RGB colour cube. When we deal with the problem of matching different reconstructions, we pool together the vertices belonging to different frames and compute a single histogram. In our experiments, we have quantized the RGB space into $10 \times 10 \times 10 = 1000$ bins. To compute key-colours in the scene, we identify the local maxima amongst the various bins in this histogram. To select well-behaved peaks, we smooth the histogram and perform non-maximal suppression by checking the neighborhood of each peak. A subset of the bins are thus selected as key-colours of the scene. These key-colours provide an adaptive quantization of the RGB space for the given scene. Then we label each reconstructed vertex of the scene with the closest key-colour. In our experiments, we have chosen a number of $b_q = 6$ key-colours to label the scene.

Figure 4.6: Colour blobs computed by labeling the surface of the shape through key-colors : the colour-blobs at two frames of reconstruction resemeble each other at a coarse level.

When we deal with shapes without too much texture variation (like people dressed in clothes of single colour), this labeling with respect to the key-colours segments the scene into a set of connected regions that we call *colour blobs*. Figure 4.6 shows an example of two frames of a reconstructed scene with colour blobs.

These colour blobs are coarse region based features. They can be compared and matched across the reconstructed shapes, to provide a rough initialization for computing the actual displacement of points. We use the colour blobs in conjunction with the surface protrusions described above.

We first introduce a photometric region descriptor for surface protrusions, by using key-colours in the scene. We discretize the region of support of a protrusion into radial zones around the feature point. In our experiments, we have used $b_z = 5$ radial zones at increasing distance from the feature point. For each radial zone in the support region, we compute the adaptive colour histogram by measuring how each of the key colours are represented in the zone. The final descriptor is obtained by concatenating $b_z \times b_q$ histogram values along each zone, followed by L^2 normalization. This descriptor is computed over a large region of the surface, and helps well in efficiently discriminating (the relatively small number of) protrusions that are detected.

In the cases where a protrusion is not detected owing to a topological collapse in one of the reconstructions, we replace the protrusion feature with the colour

blob that overlaps the most with its region of support. This is not as informative as the earlier feature which computes the probabilities of all the colours in the region of support. But it retains a good part of the information. Amongst scenes involving human beings, it is often the hands and the feet that go missing because of topological collapses. In many such cases, a colour blob of the skin colour of the actor (for hands) or a colour blob of the shoes (for feet) help us replace the concerned protrusion. An example result of matching is illustrated in figure 4.7. We describe the process of matching in the next section.

4.5 Matching Results

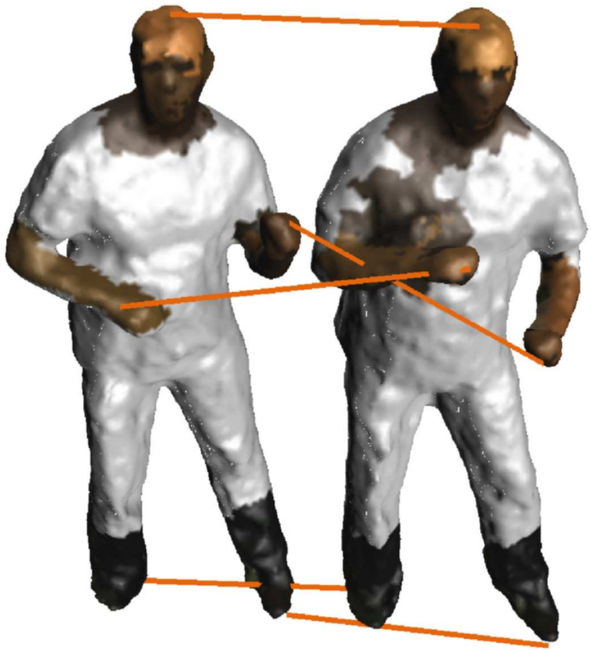
The local and global shape features that we have proposed are independent of each other, and we have evaluated both of them. It is possible to use them for various applications such as shape matching, retrieval or recognition. Depending on the application concerned, it is possible to devise a specific matching algorithm that incorporates assumptions justified about the problem setting and that exploits strengths - such as data available for training. But our objective in this chapter is not to devise a good matching algorithm, but only to propose features that could be used later. So we have performed a basic evaluation of the features using simple algorithms that we describe in the following. In chapter 5, we employ these features in the more specific problem of surface tracking. There, we use a slightly different matching algorithm that combines both the global and local features. The tracking results at the end of chapter 5 may be considered to augment the evaluation that we report here.

4.5.1 Local Feature Matching

We are validating the proposed detector and descriptor in the framework of matching shapes across non-rigid deformations. Let us consider two meshes S_1 and S_2 of the same object. The two meshes do not necessarily have the same number of vertices. Using the proposed approach, n_1 interest points are detected on S_1, which are characterized by descriptors $\mathbf{t}_{(1,i)}$, with $i \in [1..n_1]$. Similarly, n_2 interest points are detected on S_2, characterized by descriptors $\mathbf{t}_{(2,j)}$, with $j \in [1..n_2]$.

Matching. We use a simple Greedy heuristic in order to select the a set of best matches. Let the Euclidean distance $d_{ij} = ||\mathbf{t}_{(1,i)} - \mathbf{t}_{(2,j)}||$ represent the matching score between descriptor $\mathbf{t}_{(1,i)}$ from surface S_1 and descriptor $\mathbf{t}_{(2,j)}$ from S_2. Now, for each descriptor $\mathbf{t}_{(1,i)}$, consider the best two matches $t_{(2,b(1,\mathbf{t}_{(1,i)}))}$ and $t_{(2,b(2,\mathbf{t}_{(1,i)}))}$ from S_2, in terms of the defined matching score, where the function $b(x, \mathbf{t}_{(1,i)})$ returns the index of the x^{th} best match among the descriptors of S_2. The putative match $(\mathbf{t}_{(1,i)}, t_{(2,b(1,\mathbf{t}_{(1,i)}))})$ will be considered a good match only if it is significantly better than the second best match, that is if $d_{i,b(1,\mathbf{t}_{(1,i)})}\gamma > d_{i,b(2,\mathbf{t}_{(1,i)})}$, with $\gamma = 0.7$. Additionally, cross validation is performed, by checking that $t_{(2,b(1,\mathbf{t}_{(1,i)}))}$'s best match among the descriptors of S_1 is indeed $\mathbf{t}_{(1,i)}$. This is not meant to fully solve the

(a)



(b)

Figure 4.7: (a) Photometric region descriptors on the protrusion features at two frames (b) The Protrusion features matched across shapes : the protrusion that is not detected in the first mesh (the right hand of the person) is matched to the nearest color blob

matching problem, as would a global approach [Starck & Hilton 2007a]. It is merely intended for validation and for evaluation of our detector and descriptor.

Datasets.

- Synthetic Data: we consider a synthetically generated dataset entitled *Synth-Dance* of a human mesh with 7061 vertices moving across 200 frames.
- Real Data: additionally, we use frames 515-550 from the INRIA *dance-man* sequence [4dr 2010], where the same reconstruction method [Zaharescu *et al.* 2007] was employed to recover models using 32 cameras. The models have vertices ranging between 16212 and 18332.

Photometric information. The colour of each vertex of the surface is computed by considering the median colour in the visible images. We assume that the colours of a vertex follow a non-Gaussian distribution, due to errors that can occur around occluding contours. In the *Synth-Dance* dataset the vertices are randomly coloured.

Comparison with Back-Projected 2-D Features. We present a comparison between the proposed mesh matching framework using MeshHOG descriptor with another framework, currently employed in a number of mesh matching methods (see Section 4.2), that uses back-projected image descriptors. In the image based framework, the matching is performed in the images and only then is back-projected onto the surface. In our comparisons, we used the SIFT image descriptor. When matching the two surfaces, only matches from the same cameras are considered. In order to be able to carry such a comparison for the *Synth-Dance* dataset, we have generated images for 16 virtual cameras, distributed in a circular pattern around the object.

Synthetic comparative results are presented in Figure 4.8. The mesh in the first frame was matched with the mesh at any of the other 199 frames across the sequence. As it can be observed, the MeshHOG descriptor generates fewer false positives in comparison with the SIFT equivalent, clearly demonstrating the advantages of the proposed approach.

In addition, we present empirical results in Figure 4.9 for for the INRIA *dance-man* sequence. As it can be observed, the second second best match ratio threshold $\gamma = 0.7$ tends to be more aggressive for SIFT. There are only 54 matches found using the SIFT back-projected method between frame 525 and 526, whereas MeshHOG finds 119 matches. Even when matching across distant frames (530 and 550), our proposed method finds 13 correct matches, versus the SIFT descriptor, that fails to find any match. It is to be expected, since most of the inter-frame matches are due to local creases formed by the clothes. The head is the only unique feature that can be robustly matched across time.

In this section, we have presented an evaluation of *MeshDOG* and *MeshHOG* for the problem of matching visually reconstructed meshes across time. We refer the reader to our work [Zaharescu *et al.* 2009] for an evalaution of these features in their wider context. Our framework for feature detection and description is applicable

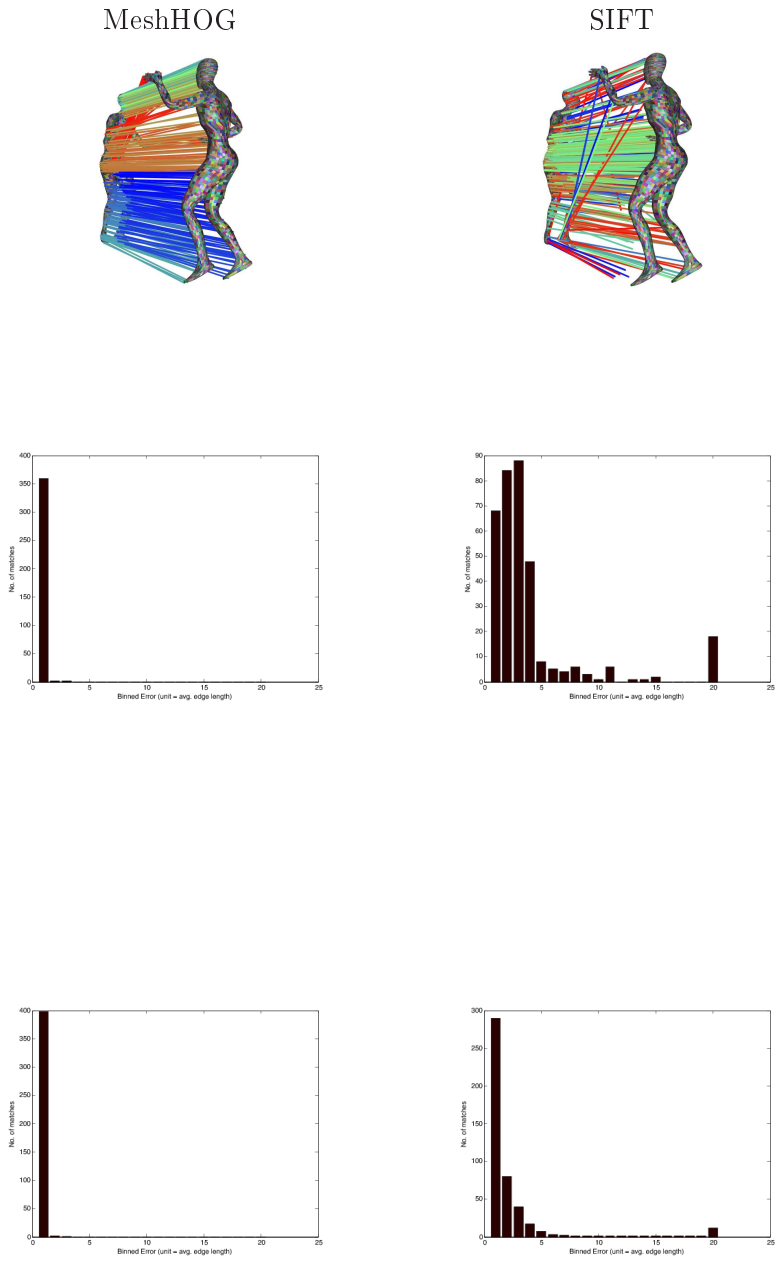


Figure 4.8: Non Rigid matching using synthetic data - *dancer-synth* dataset. Comparison between MeshHOG and SIFT matching results. Matches between frames 1 and 50 are visually depicted in the top row. There are 364 matches for MeshHOG and 343 matches for SIFT. The second row shows the error histograms for matching. The histogram bins are of size equal to e_{avg}. The last bin groups all the errors greater than $20 * e_{avg}$. The thrid row shows the histogram of the average error for matching frame 1 with x, where $x \in [2..200]$. MeshHOG outperforms SIFT in all these comparisons.

for any scalar function ϕ defined on a manifold in 3D - photometric information is not the only possible choice for ϕ. It is possible to consider other functions such as Gaussian curvature or mean-curvature of the shape as ϕ. In [Zaharescu *et al.* 2009], we have experimented with these choices and provided results on multiple problem settings - rigid body registration, rigid body registration under noise and non-rigid registration (the last of which we report here).

4.5.2 Global Feature matching

The surface protrusions that we detect as global shape features are generally few in number. So it is possible to consider the various configurations of their matches, instead of treating them independently. Also since these features are global in scope, it is also not justified to consider them in isolation from each other. We formulate the problem of matching surface protrusions as an error minimization problem with the following error:

$$E = \sum_{i} \Psi(X_i^t, X_i^{t+1}) + \sum_{i,j} |G(X_i^t, X_j^t) - G(X_i^{t+1}, X_j^{t+1})| \tag{4.5}$$

where X_i^{t+1} denotes the match of a protrusion X_i^t, $\Psi(X_i^t, X_i^{t+1})$ denotes the error computed through color features and $G(X_i, X_j)$ denotes the geodesic distance between the protrusions X_i and X_j. Since the number of detected protrusions is typically very small, we proceed to do an exhaustive search to solve the matching problem.

In the particular case of tracking adjacent frames, we add an additional term $||X_i^t - X_i^{t+1}||$ in the error, which denotes the Euclidean distance between the two matched protrusions. This means that the two matched protrusions are not too far from each other - a strong but valid assumption in our case. This helps the algorithm in resolving issues raised by symmetry (such as between the two hands, or between the two legs).

We detect cases of topological collapse through a mismatch in the number of protrusions identified in both the frames of the reconstruction. For example, a human body in a normal condition has 5 protrusions (head + 2 hands + 2 legs). When there is a collapse (hand/leg touching a part of the body), the number of detected protrusions will be 4 or lesser. The detection of the more general case, where a topological collapse happens in both the meshes, is harder. In a tracking scenario, the additional constraint that the matched protrusions be near to each other in position shall help in detecting such cases. These simple heuristics help in handling the majority of the cases.

In the event we detect a topological collapse at time $t+1$ for a protrusion X^t, we augment the global features at $t+1$ by considering colour blobs. We select the most appropriate colour blob X^{t+1} as the one which minimizes $\Psi(X_i^t, X_i^{t+1})$ (or $\Psi(X_i^t, X_i^{t+1}) + ||X_i^t - X_i^{t+1}||$) for the protrusion X^t.

It should be noted that a topological collapses will not damage the geodesic distances between protrusions which did not collapse themselves. Thus, geodesic

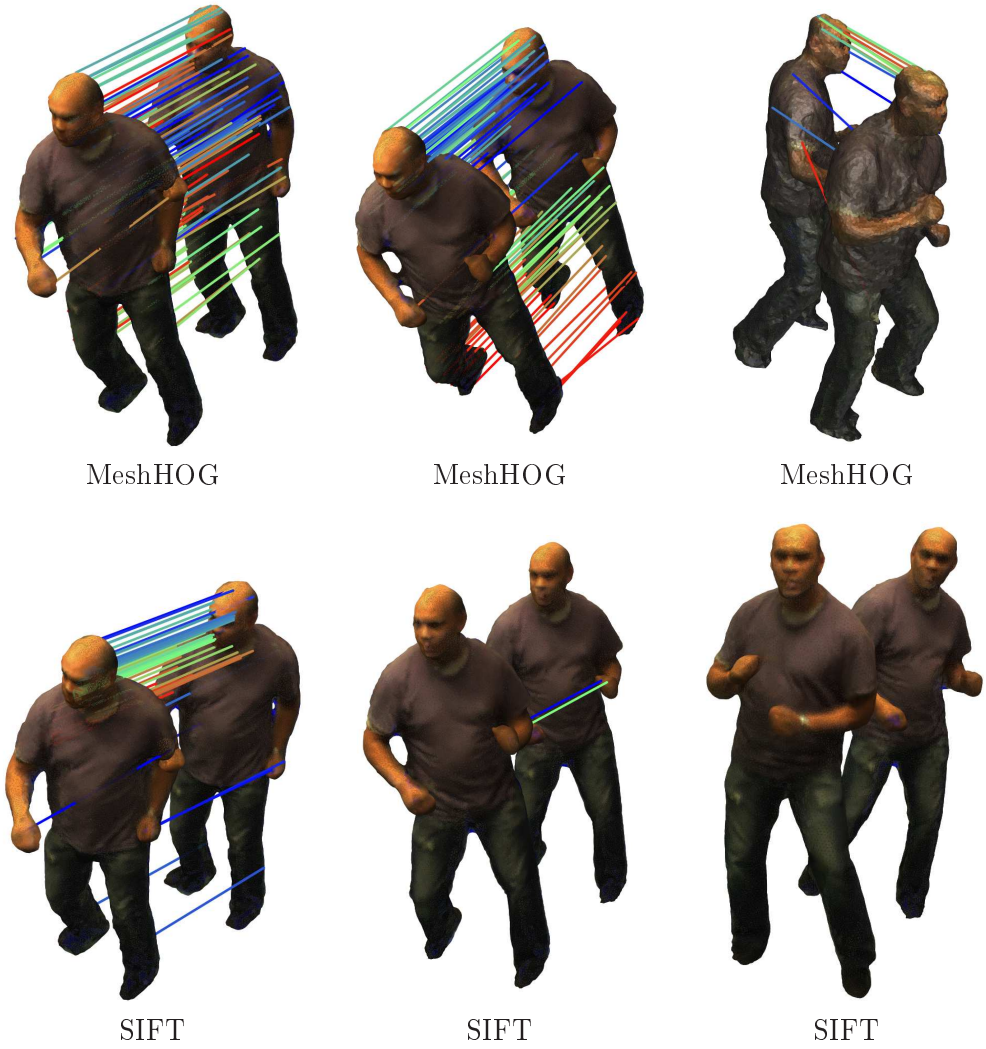


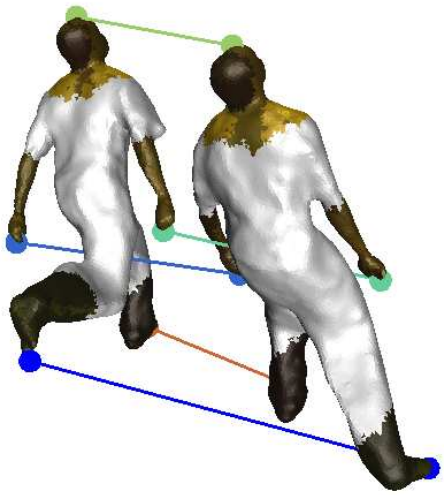
Figure 4.9: Non Rigid matching using real data - *dance-man* sequence. Comparison between MeshHOG and SIFT matching results. Matches between frames 525 and 526 are visually depicted in (a), (d). There are 119 matches for MeshHOG and 54 matches for SIFT. Matches between frames 530 and 531 are visually depicted in (b), (e). There are 122 matches for MeshHOG and 2 matches for SIFT. Matches between frames 530 and 550 are visually depicted in (c), (f). There are 13 matches for MeshHOG and 0 matches for SIFT.

distances in equation 4.5 can still be used for finding the best configuration of matches. In figures 4.10, 4.11 we show qualitative results of matching coarse features across various real-world reconstructions.

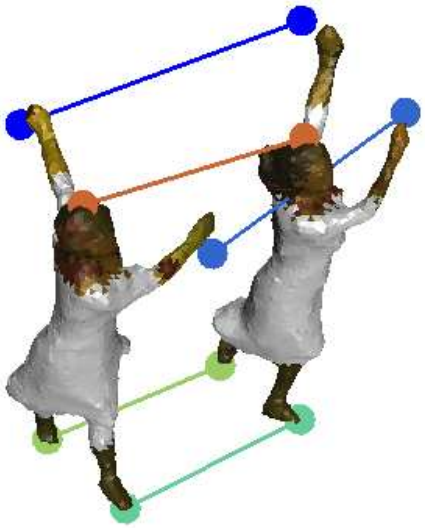
4.6 Discussion

In this chapter, we have proposed local and global shape features for matching visually reconstructed shapes across non-rigid deformations. We have employed geometric and photometric information simultaneously, thereby avoiding the limitations of both when considered individually. We have proposed local shape features as scale space extrema of the photometric function on the manifold mesh. We have proposed global shape features - surface protrusions and colour blobs, as maxima of integral functions that aggregate information from all over the shape.

The local and global shape features are complementary in their utility towards matching. Global features are fewer in number and thus can be matched more effectively across shapes. However, they are sensitive to inaccuracies in the reconstruction, such as topological collapses. On the other hand, local features are robust towards such inaccuracies. But they are far more numerous and matching them in a consistent way is much harder. In this chapter, we have evaluated both these features using simple algorithms that are easy to implement and that do not require any further training. However it is better to exploit the synergy between the complementary types of information available in both these features. Specifically, coarse-matching of global features provides a good initialization for a fine-level matching of local features. We describe this in the next chapter.

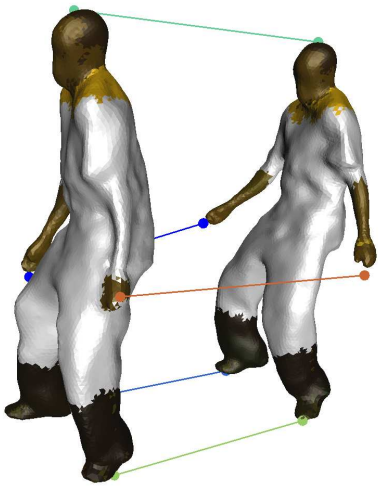


(a)

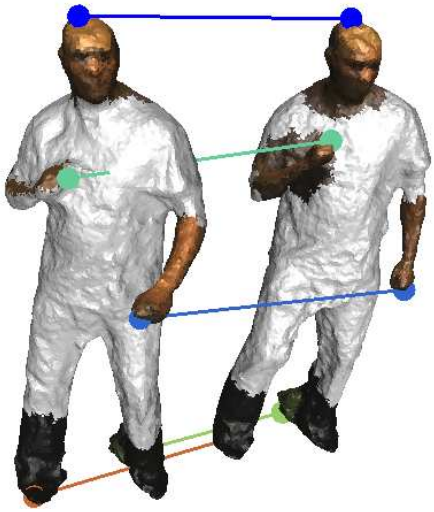


(b)

Figure 4.10: Matching of coarse features across visual reconstructions at different times (a) *flashkick* sequence frames 59, 61 (b) *dance-girl* sequence frames 597, 593



(c)



(d)

Figure 4.11: Matching of coarse features across visual reconstructions at different times (c) *pop2lock* sequence frames 39, 42 (d) *dance-ben* sequence frames 507, 500. The results show matches across topological collapses.

Chapter 5

Surface Tracking by Mesh Evolution

Contents

In this chapter, we address the problem of surface tracking in multiple camera environments and over time sequences. In order to fully track a surface undergoing significant deformations, we cast the problem as a mesh evolution over time. Such an evolution is driven by 3D displacement fields estimated between meshes recovered independently at different time frames. Geometric and photometric information are used to identify a robust set of matching vertices. This provides a sparse displacement field that is densified over the mesh by Laplacian diffusion. Our contribution is a novel mesh evolution based framework that allows to fully track, over long sequences, an unknown surface encountering deformations, including topological changes. Results on challenging and publicly available image based 3D mesh sequences demonstrate the ability of our framework to efficiently recover surface motions. The work presented in this chapter has been published earlier by us in [Varanasi *et al.* 2008].

5.1 Introduction

Tracking the surface of moving objects is of central importance when modeling dynamic scenes using multiple videos. In this chapter we describe a method for

obtaining dense temporal correspondences on the surface. This key step in the modeling pipeline is necessary when considering motion related applications such as motion capture. Furthermore, it allows recovery of improved and consistent descriptions of object shapes and appearances.

In this work we address the problem of capturing the evolution of a moving and deforming surface, in particular moving human bodies, given multiple videos. Our approach is grounded on the observation that natural surfaces are usually arbitrary shaped and difficult to model *a priori*. In addition, shapes can significantly evolve over a time sequence. For instance, human bodies are usually covered by clothes whose topologies can change. To handle such deformations, we use meshes which are morphed from one frame to another. We use our feature matching strategy described in chapter 4 to obtain a set of sparse but robust matches. We then use the associated sparse displacement vectors to drive a full consistent mesh evolution, with possible topological changes. This approach provides both a consistent surface evolution over time and dense point trajectories on the surface.

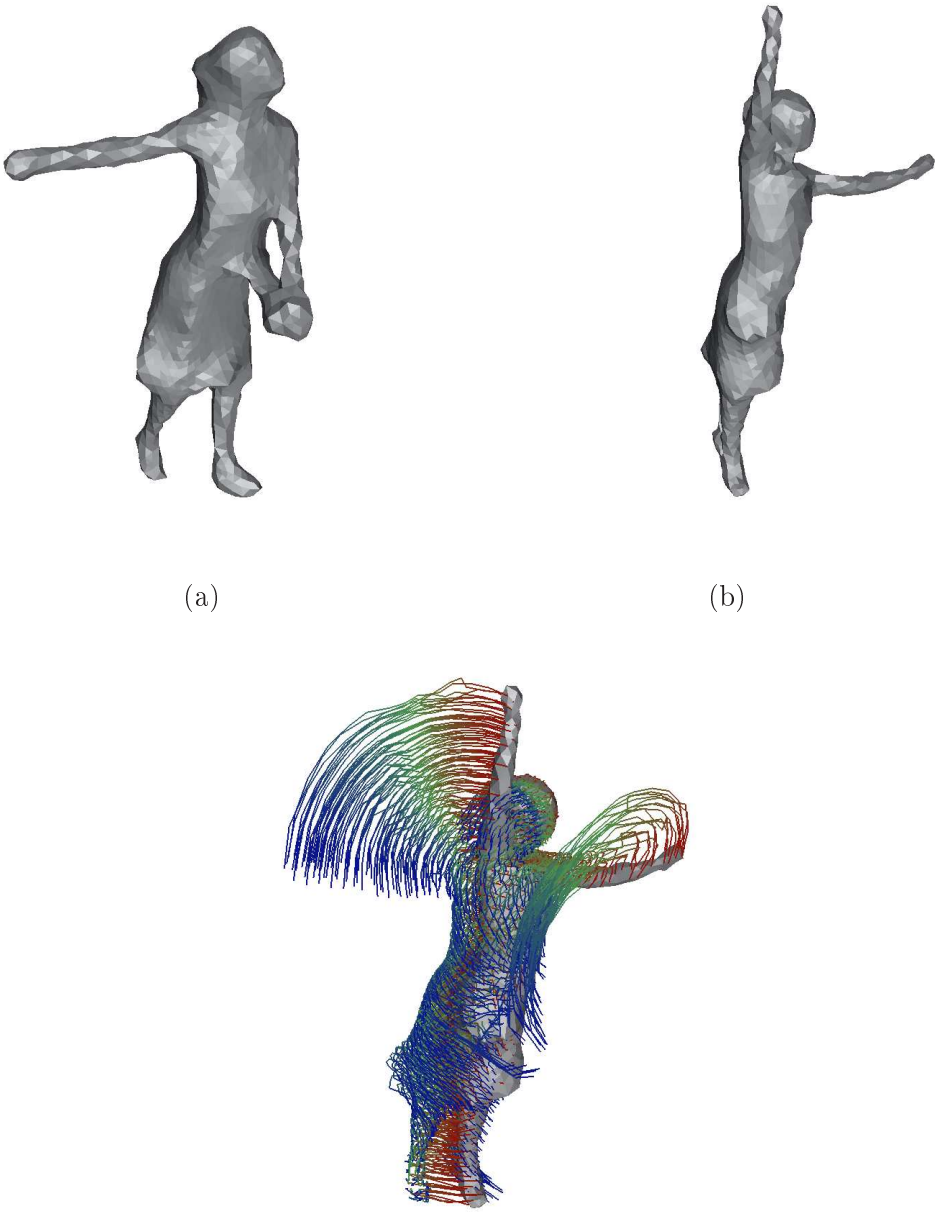
Our work falls under the class of unsupervised methods for spatio-temporal modeling, reviewed in section 1.4.3. As recounted here, methods for the computation of *scene-flow* directly from multi-view images can be considered as unsupervised. However, as observed by [Starck & Hilton 2007a], they are limited to capturing small displacements between successive frames. Capturing long trajectories, in the spirit of model-based approaches (such as [de Aguiar *et al.* 2007a, Vlasic *et al.* 2008, Cagniart *et al.* 2009]) is a harder problem. The main bottleneck is an effective means for evolving a complex geometric object (such as a mesh) that follows the topological changes that occur across the reconstructed shapes of different frames.

Our principal contribution is a framework for shape evolution that is flexible enough to model changes in its topology and structure. Figure 5.1 shows an example result of our method. The reconstructed surface is that of a dancer wearing a loose robe. The topology of the reconstructed surface changes significantly over time - the belt of the robe disappears and appears, sometimes merging with the hand of the dancer. The topology for such a scene can hardly be known in advance. Our method recovers dense motion information from such scenes, though by itself, it doesn't output a single mesh with a fixed temporally consistent structure and topology. We leave the latter problem for future work.

The remainder of this chapter is organized as follows. Related works are reviewed in section 5.2. The proposed approach is outlined in section 5.3. The recovery of displacement vector fields is described in section 5.4 and 5.5. The mesh deformation is then explained in section 5.6. Experimental results obtained with publicly available sequences are shown in section 5.7, before concluding in section 5.8.

5.2 Related Work

The problem of recovering dense 3D motion from multi-view images has been significantly popular with the research community in the recent past. In chapter 1,



(a) (b)

(c)

Figure 5.1: An example of a surface for which the topology can hardly be known in advance. The belt of the dress forms a new protrusion that appears and disappears (a)-(b). (c) Dense point trajectories computed from (a) to (b)

we have reviewed several related works in this area. These works can be principally grouped into model-based approaches and unsupervised approaches. Our work falls in the latter class.

Model based approaches (reviewed elaborately in section 1.4.2) require a dense structural model e.g, a detailed 3D mesh to fit to the observations at different instants of time. A principal representative work is [Ahmed *et al.* 2008] which uses sparse motion information obtained by matching image based features across meshes to deform a highly detailed surface across time. Such works have taken advantage of a parallel research effort in the mesh processing community for obtaining plausible shape deformation. Given a few sparse displacement vectors, the objective is to deform a shape (e.g. a 3D mesh) such that local shape properties such as curvature are preserved. This can be done by solving a linear Poisson equation described over the Laplacian operator on the surface. This may be termed as the *vanilla deformation scheme* - our method performs this as a step within its pipeline, and we describe it clearly in section 5.5. However, more complex schemes for mesh deformation exist. [Lipman *et al.* 2005] propose using rotation invariant coordinates - they produce more realistic deformations when the displacements are large. This deformation method still remains linear in complexity and is thus used widely by model-based approaches [Vlasic *et al.* 2008, Cagniart *et al.* 2009]. In a later work, [Sorkine & Alexa 2007] propose a non-linear iterative approach to obtain deformations that are locally as rigid as possible. In practice, this method still converges very rapidly. The deformations can also be constrained to follow a set of given examples [Weber *et al.* 2007]. More elaborate constraints such as preservation of volume, skeletal rigidity etc can also be added as in the *mesh puppetry* work of [Shi *et al.* 2007]. However, such constraints yield a highly non-linear problem which cannot be solved as simply. Another class of deformation techniques do not deform the mesh itself but the space in which it is embedded in. [Lipman *et al.*] use Green coordinates to deform a cage around the mesh. [Sumner *et al.* 2007] use a coarse deformation graph to drive the deformation of the underlying mesh. It should be noted that most of these deformation schemes were proposed keeping in mind the problem of interactive shape editing, where the artist edits a few vertices and the entire shape is deformed in a plausible manner.

The problem of dense motion estimation from videos is not very similar to the above. The two main differences are that

a there is usually more information forthcoming from videos

b this information is less reliable and more likely to be self-contradictory.

To account for these differences, [Cagniart *et al.* 2009] model the motion information at the level of surface patches which may be larger in area than triangles on the mesh, and resolve contradictions in the sparse inputs through a probabilistic voting scheme.

With unsupervised approaches, the problem becomes even harder as there exists no underlying mesh structure that can be relied upon for deforming and fitting to the

observations. The critical difference is the potential within the evolution framework to handle topological changes. The works of [Wand *et al.* 2009, Popa *et al.* 2010] attempt to solve this problem at high resolution detail between mesh vertices, and to obtain a surface of temporally consistent topology. They show results in cases where such high resolution data can be obtained reliably - for example, through active sensing methods such as structured light. Similar high resolution data can also be obtained from methods of photometric stereo [Hernandez-Esteban *et al.* 2008, Wenger *et al.* 2005]. But for the case of passive multi camera systems, the surfaces reconstructed from silhouettes or geometric stereo are of much lower quality. In this scenario, obtaining a shape of temporally consistent topology is a harder problem. Our approach is less demanding, and seeks to obtain only point trajectories that are spatially consistent on the shape. The works of [Courchay *et al.* 2009, Aganj *et al.* 2009, Thorstensen & Keriven 2009] seek a similar objective. There are two ways to evolve a shape along with changes to its topology - *Eulerian* and *Lagrangian*. *Eulerian methods* e.g, level sets [Osher & Fedkiw 2003], represent a function in a higher dimensional space and obtain the shape as its zero level set. These methods had been popular in the past, but suffer from higher computational cost and are inappropriate for modeling interface properties - such as vector displacements. The alternative *Lagrangian* work on the mesh vertices themselves and adjust for the changes in topology, carefully avoiding degenerate cases. This can be done through performing a Delaunay re-triangulation of the point set at each step in the evolution [Pons *et al.* 2007a, Thorstensen & Keriven 2009] or by detecting the degenerate cases and fixing each of them appropriately [Zaharescu *et al.* 2007]. We use the second approach and describe it in greater detail in section 5.6.

5.3 Approach Outline

We consider multiple camera environments and we assume that multiple calibrated videos of an object with closed surfaces are available. We also assume that 3D mesh models $\mathcal{M}^{t\in[1..n]}$ of the object at different time instances $[1..n]$ estimated using multi-view 3D modeling approaches, e.g. [Hernandez & Schmitt 2004, Furukawa & Ponce 2006, Pons *et al.* 2007a], are available. These meshes $\mathcal{M}^{t\in[1..n]}$ correspond to discrete values of the time continuous mesh $\mathcal{S}^t$. In order to recover $\mathcal{S}^t$, the mapping of $\mathcal{S}^t$ onto $\mathcal{M}^{t+1}$ is iteratively estimated using the following 3 consecutive steps, starting with $\mathcal{S}^1 = \mathcal{M}^1$:

1. Geometrically consistent matching : photometric and geometric cues are used to match a set of points between $\mathcal{S}^t$ and $\mathcal{M}^{t+1}$ (see Figure 5.2-c). We expect only a sparse set of correspondences, but we require them to be geometrically valid along the shape 4.

2. Motion diffusion: the identified correspondences define a sparse displacement field over $\mathcal{S}^t$ (cf. figure 5.2-d). This field is propagated over the mesh by Laplacian diffusion hence preserving local shape details [Sorkine 2005] (cf. Figure

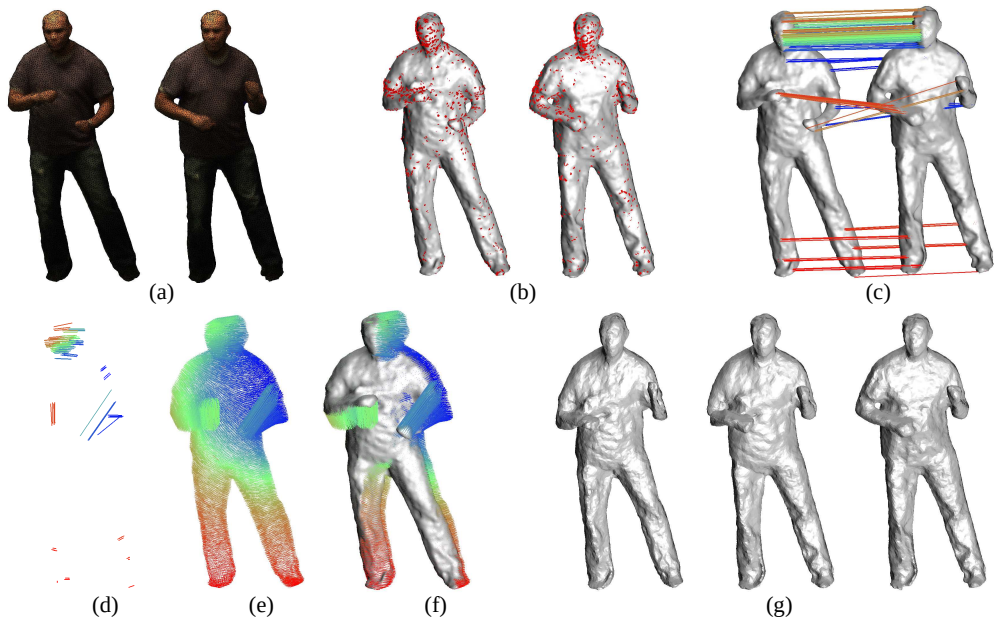
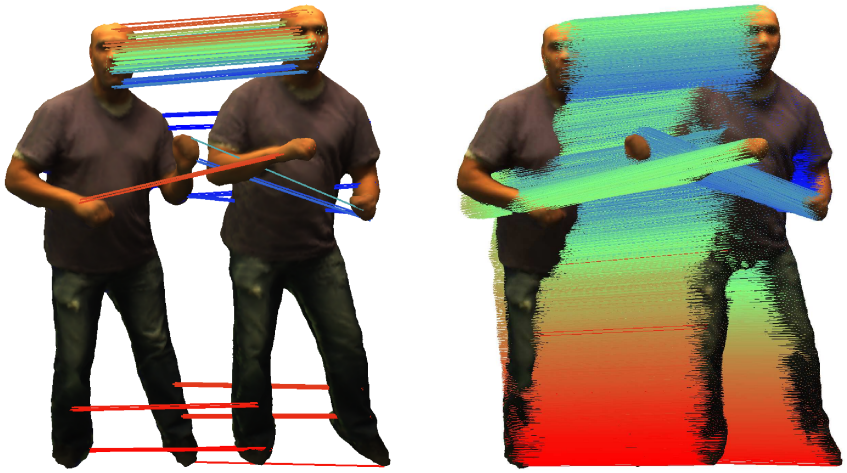


Figure 5.2: The different consecutive steps of the proposed framework: (a) original meshes $\mathcal{S}^t$ and $\mathcal{M}^{t+1}$; (b) features extracted; (c) feature matching; (d) the associated sparse-displacement; (e)-(f) the dense displacement field after Laplacian diffusion; (g) mesh-morphing (observe the topological change that takes place around the right arm of the model - the right elbow de-attaches from the body, creating a genus change).

5.2-e). However, unlike [Sorkine 2005], we conduct the propagation only on portions of $\tilde{\mathcal{S}}^t$ that are aligned in the previous step. We thereby facilitate the global alignment of topologically distinct shapes.

3. Mesh evolution: The dense displacement field is applied to the vertices of $\tilde{\mathcal{S}}^t$ yielding a new mesh. The resulting mesh $\hat{\mathcal{S}}^t$ is then morphed to $\mathcal{M}^{t+1}$ by minimizing the signed distance to $\mathcal{M}^{t+1}$ (cf. Figure 5.2-fg). Mesh consistency within the optimization is enforced using [Zaharescu *et al.* 2007]. The final optimized mesh defines $\mathcal{S}^{t+1}$.

The Laplacian diffusion allows a partial vertex matching only and yields to a good estimation of the motion at all vertex locations. Nevertheless, an additional step is required to guarantee that the resulting mesh fits the observations $\mathcal{M}^{t\in[1..n]}$ and also to guaranty its correctness, e.g. manifoldness. This is in contrast with the work [de Aguiar *et al.* 2007a] which also uses Laplacian diffusion to evolve a reference mesh to the observed posture, but without refinement and therefore without guaranties.



(a) (b)

Figure 5.3: (a) The set of identified feature matches (b) The dense motion field computed by Laplacian diffusion.

5.4 Geometrically Consistent Feature Matching

We use the local and global shape features that we introduced in chapter 4 to match between the meshes. Instead of using the simple algorithms described in section 4.5, we use a slightly more elaborate algorithm that suits the current scenario of tracking. The principal requirement here is to derive matches that are geometrically consistent with each other. By that, we mean that the structural relationships between pairs of points be preserved along the match. This requirement is necessary to ensure that the motion information that we recover is geometrically valid with respect to the shape.

We identify two types of structural relationships between pairs of points.

Euclidean Link When two surface points are *visible* from each other (i.e, the straight line joining them does not intersect any surface facet), the Euclidean distance between them can be taken as the interior distance. These distances are preserved only over *rigid transformations*. As discussed in chapter 3, convex segments often represent the limbs of an articulated shape and can be expected to undergo rigid motions over time. Hence, Euclidean distance can be used for points belonging to the same segment. This is likelier to happen for points lying in the immediate geodesic vicinity of each other.

Surface Geodesic Link When two points are close to each other on the surface, the interior distance between them can be approximated by the surface geodesic distance. In fact, the geodesic distances are preserved over *isometric deformations*, which are more general than rigid deformations. However, they get altered by the artifacts of visual reconstruction such as surface noise and

topological collapses. Surface geodesic distances are a reliable information up to a longer range than Euclidean links between points.

In order to derive geometrically consistent matches, we first employ the simple matching algorithm for local features described in section 4.5. Then we prune the set of matches by allowing only matches that are geometrically consistent within their local neighborhood.

If two feature points P_1^t, P_2^t are found to be geodesically near to each other, we connect them by the Euclidean link which encodes the Euclidean length $|\overrightarrow{P_1^t P_2^t}|$ of the line joining the two points, and the angles it makes with the normals $\hat{P_1^t}, \hat{P_2^t}$ at both the ends. Then the two pairs of matches (P_1^t, P_1^{t+1}) and (P_2, P_2^{t+1}) are checked for mutual spatial consistency in terms of the elastic stretch (γ_s) and twist $(\gamma_{t1}, \gamma_{t2})$ of the link, defined as:

$$\begin{aligned}
\gamma_s &= \Gamma_s(|\overrightarrow{P_1^t P_2^t}| - |\overrightarrow{P_1^{t+1} P_2^{t+1}}|) \\
\gamma_{t1} &= \Gamma_t(\theta(\overrightarrow{P_1^t P_2^t}, \hat{P_1^t}) - \theta(\overrightarrow{P_1^{t+1} P_2^{t+1}}, \hat{P_1^{t+1}})) \\
\gamma_{t2} &= \Gamma_t(\theta(\overrightarrow{P_1^t P_2^t}, \hat{P_2^t}) - \theta(\overrightarrow{P_1^{t+1} P_2^{t+1}}, \hat{P_2^{t+1}}))
\end{aligned}$$

where $\theta(\vec{v1}, \vec{v2})$ denotes the angle between two vectors, and Γ_s, Γ_t denote two Gaussian penalty functions. We employ three thresholds on $\gamma_s, \gamma_{t1}, \gamma_{t2}$ to prune out matches that violate the constraints of geometric consistency. This produces a reduced set of matches that are geometrically valid in local neighborhoods. However, this set still suffers from two drawbacks : (i) the matches might still be geometrically invalid at a larger global scale (ii) the matches need not be defined ubiquitously on the surface - certain key areas of motion, such as the limbs of a human body may not have any matches detected on them due to lack of texture.

In order to overcome these drawbacks, we employ global shape features. We detect surface protrusions on both the shapes. In addition to the photometric feature descriptor computed using colour blobs, we associate each surface protrusion to the set of local feature matches its neighbourhood. Given a pair of protrusions that can be potentially matched, we provide a matching score between two protrusions as a combination of the photometric descriptors and the local features in the vicinity. Since the number of detected protrusions on the two shapes is small, we evaluate all the possible configurations for matches between them, using our algorithm described in section 4.5. Also as described, we detect the event of a topological collapse and match the associated protrusion with the closest colour blob. Since this is a tracking scenario, we prefer the matched protrusions to be close to each other with respect to their positions. The final associated error for a selected match X_i^t, X_i^{t+1} is given as

$$E = \sum_i \Psi(X_i^t, X_i^{t+1}) + \sum_{i,j} |G(X_i^t, X_j^t) - G(X_i^{t+1}, X_j^{t+1})| + ||X_i^t - X_i^{t+1}|| \quad (5.1)$$

When the surface protrusions are thus matched, they enforce geometric consistency at a global scale. We also obtain matches along places most susceptible for large movements *i.e*, surface protrusions. We further prune the local shape features by utilizing this information. We check each matched pair for parity in order (ρ_1) and orientation (ρ_2) with respect to the nearest matched protrusion (X^t, X^{t+1}), defined as:

$$\rho_1 : Sign(|\overrightarrow{P_1^t X^t}| - |\overrightarrow{P_2^t X^t}|) = Sign(|\overrightarrow{P_1^{t+1} X^{t+1}}| - |\overrightarrow{P_2^{t+1} X^{t+1}}|)$$
$$\rho_2 : \theta(\overrightarrow{P_1^t X^t} \times \overrightarrow{P_2^t X^t}, \overrightarrow{P_1^{t+1} X^{t+1}} \times \overrightarrow{P_2^{t+1} X^{t+1}}) < 180^\circ$$

The careful checks that we have explained in this section help us derive a set of sparse but robust matches that are geometrically consistent along the shape. These feature matches sparsely define a set of vectors on the surface of the mesh, which can be considered as samples of a plausible motion field on the surface. In the next section we describe how to recover such a plausible and dense motion field on the surface.

5.5 Surface Deformation by Motion Diffusion

The set of sparse vectors defined by feature matches provide a good initialization for the transformation of the mesh. In order to obtain a dense motion field on the surface, we propagate the sparse vectors across the entire mesh by Laplacian diffusion.

The geometric Laplacian operator is a way of encoding the local curvature of the mesh. This has proven to be useful in a variety of mesh applications [Sorkine 2005], such as interactive mesh editing. This operator provides an efficient approach to deform the mesh while preserving the local shape information. If N_V is the number of vertices, the Laplacian matrix L of size $N_V \times N_V$ is defined by these equations

$$\begin{aligned} L(i,j) &= wt(i,j) \quad \forall_{j \in N(i)} \\ L(i,j) &= 0 \quad \forall_{j \notin N(i)} \\ L(i,i) &= -1 * \textstyle\sum_{j \in N(i)} wt(i,j) \end{aligned}$$

where $N(i)$ is the set of vertices sharing an edge with the vertex i, and $wt(i,j)$ is the weight of the edge as defined by mean-value coordinates [Sorkine 2005]. We compute the differential coordinates of the mesh at time t into three vectors $\delta X^t, \delta Y^t$ and δZ^t, where $\delta X^t = L * X^t$ (similarly for δY^t, δZ^t).

The feature matches computed earlier as initialization, we now define 3 matrices L_x, L_y and L_z, corresponding to the three dimensions X, Y and Z. If the number of feature matches is N_F, these matrices shall be of order $(N_V + N_F) \times N_V$. The first N_V rows shall be identical to the L matrix. The later rows are defined by constraints ($\forall i \in \{features\}$) (similarly for L_y, L_z):

$$\begin{aligned} L_x(i,j) &= 0 \quad \forall j \neq i \\ L_x(i,i) &= \lambda \end{aligned}$$

where λ is a weighting factor we set to 4000.

Similar to the matrix L_x, we append the vector δX^t by adding N_F new elements $\{\lambda * X_F^{t+1}\}$ where X_F^{t+1} are the X-coordinates of the feature matches in the frame $t+1$. The diffusion of the matches is done as a matrix inversion.

$$X^{t+1} = (L_x^\top L_x)^{-1} L_x^\top * \delta X^t$$

The matrix L_x being extremely sparse, this inversion can be efficiently implemented using Cholesky factorization.

Thus we propagate the mesh S^t via Laplacian diffusion to $\hat{\mathcal{S}}^t$. An example of the dense motion field obtained from a sparse set of feature matches is shown in Figure 5.3-b.

As mentioned in the related work of section 5.2, more elaborate techniques exist for shape deformation which encode a gamut of various powerful constraints in order to make the deformation appear smoother and more geometrically plausible. However, since the reconstructed geometry in our case is not of high quality, we do not avail of such elaborate techniques. Instead we spend most of our effort in deriving a set of geometrically consistent matches - picking them up on regions susceptible for large motions such as protrusions. When this set of feature matches is sufficiently well-sampled on the shape, even the most basic technique for shape deformation provides reasonable results.

5.6 Evolution of Mesh Connectivity and Topology

The matching and diffusion steps presented in the previous sections provide us with a dense displacement field over the mesh $\mathcal{S}^t$. As mentioned before, such motion field is a good estimate of the true motion field between time t and $t+1$. However, it will not guarantee the exact overlap with the mesh observed at $t+1$, i.e. $\mathcal{M}^{t+1}$, nor the correctness of the resulting mesh. Therefore, a final step is needed in order to ensure both convergence to the observations and correctness. Our approach is motivated by the fact that the solution mesh $\mathcal{M}^{t+1}$ and the propagated Laplacian mesh $\hat{\mathcal{S}}^t$ are different, but nearby. Thus a solution is to perform surface-morphing, that is starting from the source surface, i.e. $\hat{\mathcal{S}}^t$, and evolving it towards the destination surface $\mathcal{M}^{t+1}$. To this purpose, we have used [Zaharescu *et al.* 2007] as an explicit surface evolution approach which handles self-intersections and topological changes and guarantees correctness. To drive the surface evolution, we adopt here a simple morphing scheme introduced in [Breen & Whitaker 2001] and described below.

Consider an open set $O_A \subset \mathbb{R}^3$ representing the source object, enclosed by surface $S_A = \partial O_A$, and similarly the open set $O_B \subset \mathbb{R}^3$ representing the target object, enclosed by $S_B = \partial O_B$. Consider the signed distance u_B of S_B, as defined by:

$$u_B(x) = \begin{cases} -d(x, S_B) & \forall x \subset O_B, \\ d(x, S_B) & \text{otherwise}, \end{cases} \tag{5.2}$$

where $d(x, y)$ is the Euclidean distance between x and y in $\mathbb{R}^3$. Following [Breen & Whitaker 2001], the surface motion that maximizes the overlap between the morphed object and S_B is defined by:

$$\frac{\partial S}{\partial t} = -u_B(x)\mathbf{N}(x), \tag{5.3}$$

where x is a point on the surface S and $\mathbf{N}(x)$ is the normal to S at x. The strategy described above will converge to the desired solution if the surface of departure S_A and the destination surface S_B overlap.

5.6.1 Discussion

In a few cases, certain tracks are temporarily lost, due to the non-overlap of certain parts of the surface between the propagated Laplacian and the next frame. Such an example can be observed in Figure 3.6-f, where the left hand had the fist properly propagated (due to the protrusion region matching), but not the forearm (due to the lack of features). This caused the signed distance function based evolution to collapse a sub-part of the forearm and regrow it from the upper-arm and the fist. These rare cases can be addressed by interpolating the trajectories from the neighboring vertices which are tracked correctly.

If we are not satisfied with the propagated Laplacian, we can also try to increase the number of the matches by exploring the neighborhoods of the sparse matches detected by our method. These increased matches are then diffused in a similar fashion using the mesh Laplacian. At a minor additional computational cost, this process produces a better initialization for the mesh deformation step.

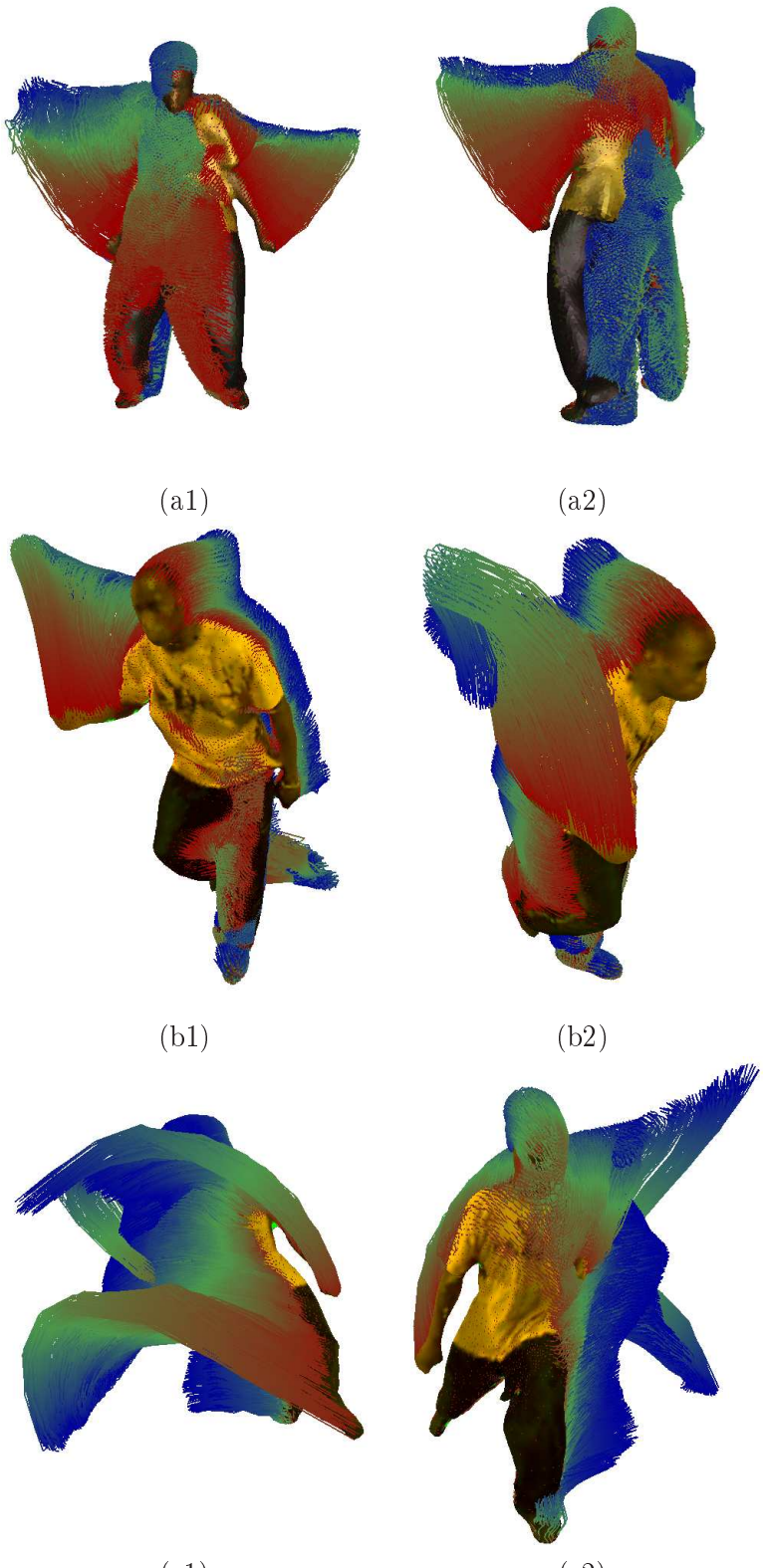
Another remark is that instead of surface morphing, one could also consider other functions. One such choice is multi-view stereo photo-consistency. We have experimented with such a distance function [Pons *et al.* 2007a], observing that the optimizer could not easily handle situations where the source mesh is relatively far from the destination mesh. This is in part due to its coarse to fine nature. Another benefit of the mesh morphing approach is that, assuming there is some overlap between the source $\hat{\mathcal{S}}^t$ and the destination $\mathcal{M}^{t+1}$ meshes, it is guaranteed that the approach will converge, with potential topological changes. In addition, every vertex will reach the destination mesh. Once reached, it will neither move nor oscillate.

5.7 Results

5.7.1 Qualitative Evaluation

For our evaluation we have been using sequences from two sources: the *dance-man* (used to exemplify the method) and the *dance-girl* sequences are available publicly on our website [1]. The *pop2lock* and *flashkick* sequences were made available to us by the Surface Motion Capture project at the University of Surrey [Starck & Hilton 2007b].

[1] https://charibdis.inrialpes.fr/html/sequences.php

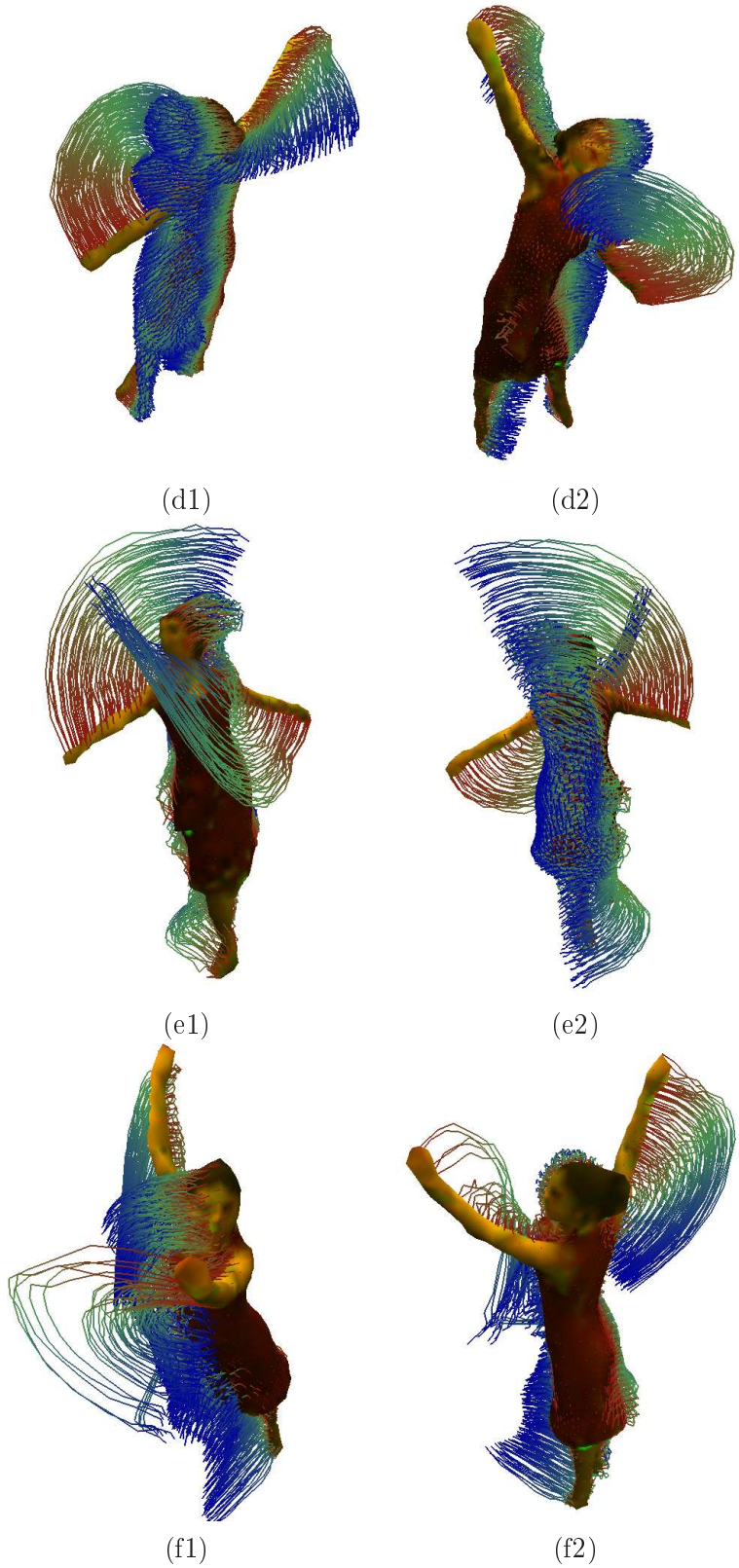


(a1) (a2)

(b1) (b2)

(c1) (c2)

Figure 5.4: Dense 3D trajectories are presented in a colour coded scheme - cooler colours represent earlier frames. (a1,a2) *pop2lock* sequence frames 20-40 (b1,b2) *pop2lock* sequence frames 78-87 (c1,c2) *flashkick* sequence frames 50-60. The data is courtesy Univ. of Surrey. ©



(d1) (d2)

(e1) (e2)

(f1) (f2)

Figure 5.5: Dense 3D trajectories are presented in a colour coded scheme - cooler colours represent earlier frames. The data is from the *dance-girl* sequence from INRIA. ©(d1,d2) frames 567-582 (e1,e2) frames 620-635 (f1,f2) frames 607-622

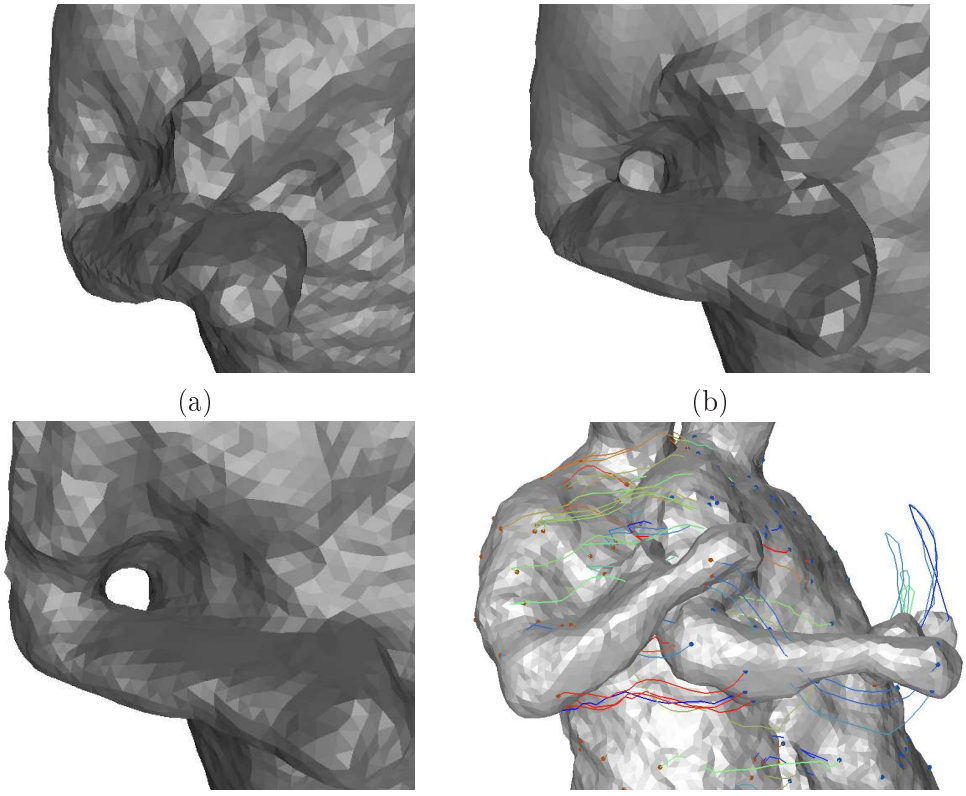


Figure 5.6: Mesh deformation over topological change : (a) initialization (b) intermediate step (c) final step (d) overall algorithm behavior (sparse set of matches shown only for ease of visualization purposes).

The *pop2lock* sequence provides us with full 3-D reconstruction results, together with the camera calibration, input images and silhouettes, using 8 cameras (1920x1080). The *dance-girl* sequence is captured using 8 cameras (780x582). For this last sequence we decided to test the limits of the algorithm. We have used rougher 3-D surfaces approximation obtained via a fast visual hull reconstruction (exclusively based on silhouettes). Despite their coarse nature and topological changes, we still obtain consistent point trajectories. We ensured proper mesh sampling via edge collapses and edge swaps, such that each edge is around 3 pixels when projected onto the image (*pop2lock* meshes - 12,000 vertices; *dance-girl* - 3,000 vertices). Coarser meshes were used for computing geodesics (1,500 vertices).

Our results are presented in Figure 3.6, with a close-up of a topological change illustrated in Figure 5.6. The tracks are colour-coded, where cooler colours represent earlier frames. Additional convincing results are available as a video [2], the natural way of displaying temporal information.

We were able to successfully track without problems *long* sequences of over 100

[2] https://perception.inrialpes.fr/Publications/2008/VZBH08/ECCV08.mp4

frames with *large* inter-frame shifts. The running times are satisfactory, depending a lot on the mesh density and the number of images used within each frame. As an example, in the *dance-girl* sequence, an inter-frame surface tracking is produced in about 30 seconds, whereas for the *pop2lock* dataset, it takes about 2.5 minutes.

5.7.2 Numerical Evaluation

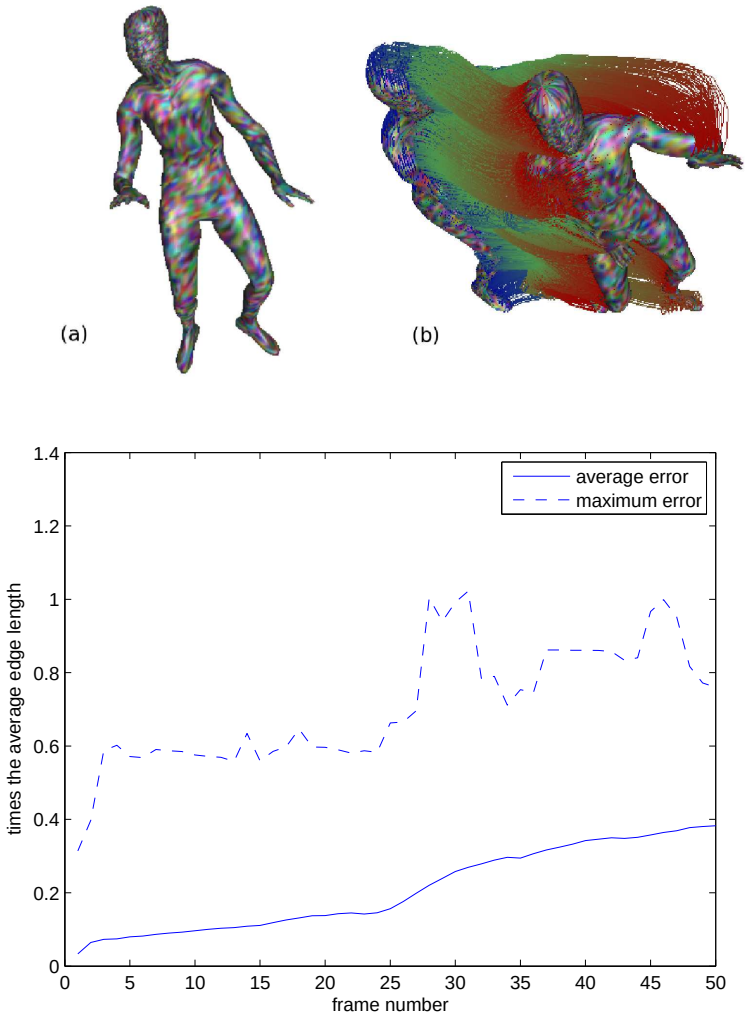
Lack of proper ground truth makes quantitative assessment of 3D tracking algorithms difficult. A manual labeling could be inconsistent because the accuracy of the tracks needs to be measured with high precision. Due to the absence of real world test data, we evaluated the trajectories of our algorithm against known deformations of a 3D graphical model. We used an artificially textured female humanoid model (figure 5.7-a), and the multi-view video is captured using a 16 camera setup. An example trajectory computed by our algorithm is visualized in figure 5.7-b.

We evaluated the error in point trajectories with respect to the average edge length, which defines the resolution for temporal correspondences. Figure 5.7-c shows such errors for 600 points randomly distributed over the mesh and as obtained with independent estimations of the surface evolutions between frames. We observe that the error is less than half the average edge length after 50 frames. In the same duration, the average true deformation encountered by each point is about 10 times the average edge length. Thus we stay within reasonable limits of accuracy in producing our tracks.

5.8 Conclusion

In conclusion, we have presented a robust algorithm for temporal mesh tracking that incorporates the following key ingredients: it uses both geometric and photometric information in a coarse to fine fashion in order to efficiently solve for a sparse set of matches; it uses Laplacian propagation to obtain a dense match set; it ensures proper evolution using a mesh-morphing approach that is capable of dealing with topological changes. Thus, we are able to perform surface tracking with *large displacements* of surfaces with *topological changes* over *long sequences* in the context of multiple camera environments. In addition, our algorithm performs gracefully even when provided with inexact surfaces.

Several avenues exist for future work. The drift in trajectories can be reduced by adopting the common practice of time integration from a chosen frame. This however requires the chosen frame to be of good quality. A more principled approach would be to build a globally optimal shape from the bottoms up, and use this as the canonical frame for tracking. We elaborate on this research theme in the concluding chapter 6. Further improvements to our algorithm would be towards making it faster and real time.



(c)

Figure 5.7: Numerical Evaluation on the synthetic *dance-synth* sequence : (a) Example mesh with texture (b) Computed trajectories (c) Error over the sequence

Chapter 6

Conclusion

Contents

6.1 Summary

In this thesis, we have addressed the problem of spatio-temporal modeling of generic dynamic scenes by capturing them in a multi-camera environment. We have made two principal contributions towards this direction : (i) identifying the scene-components that behave as individual units of motion (ii) estimating dense motion on the surface of the shape without relying on any model about the scene. We have developed methods that are suited to the peculiarities and that overcome artifacts of visual reconstruction. In order to make these contributions, we have developed two methods of scene analysis : (a) segmentation of a visually reconstructed mesh into approximately convex components (b) feature detection and matching on photometric manifolds. We summarize the key-aspects of each of our contributions below.

Segmentation by visibility We have proposed a new method for convex decomposition through the notion of visibility from an interior point in the shape. Since we consider the interior of the shape, our method is not as sensitive to surface noise as similar decomposition methods that are in vogue. Our method is suited for decomposing any shape irrespective of its topology. Thus, it is applicable to real world scenes in all their generality, and is robust towards the artifacts that inevitably occur in any reconstruction process. The notion of visibility is very powerful, and we are currently exploring its utility beyond shape segmentation. Our method produces approximately convex segments, which are likelier to be large in size than small. However, it does not have any guarantees on optimality. This limitation will be a focus for improvement in future work.

Temporally coherent segmentation We have proposed a method for deriving temporally coherent segments from a mesh sequence. We have considered both spatial and temporal information simultaneously, and thus overcome the limitations of both when considered individually. We have discussed how to use temporal

information to yield better segments than what can be observed at a single frame. We have also proposed a method to aggregate multiple such segments into a single set that describes the dynamic scene in a concise manner. Our method is robust towards the artifacts of visual reconstruction - the topological collapse of distinct objects into one being the most important. The main limitation of our algorithm is that it does not analyze spatial relationships between segments at a global level. This needs to be considered in future work. The set of temporally coherent segments that we derive provides a good starting point for a more elaborate scene analysis. We are currently exploring implications of this towards other problems.

Features on photometric manifolds We have proposed local and global shape features that can be computed directly on photometric manifolds *i.e*, 3D meshes visually reconstructed and texture-mapped from images. We have considered both geometric and photometric information simultaneously. Considering either of these information individually is prone to artifacts of visual reconstruction, that would restrict the applicability of features computed on either of them. We have developed a principled method for local feature detection through scale-space analysis. These local features succinctly combine geometric and photometric information. We have also introduced global shape features to complement local features. The surface protrusions and colour blobs that we introduced are useful towards quickly and efficiently match two shapes at a coarse level. Since these global features are far fewer in number, a more extensive search strategy can be devised for their matching. Our local and global features are complementary, and have different kinds of strengths and drawbacks. Our strategy for feature detection and matching is a good first step, but there exists scope for better integration of photometric and geometric information. There also exists a need for bringing local and global features into a more common footing. These can be addressed in future work.

Surface tracking by shape evolution We have proposed a method for obtaining dense point trajectories on a shape by dynamically evolving its surface connectivity and topology. Our method is applicable to the very generic case in which such topological changes do happen - such as a cloth being torn. But more practically, it efficiently estimates motion information in the scene when no assumptions can be made about its topology. We treat the reconstructed meshes at each instant of time as equally plausible explanations of the scene, and do not introduce bias by assuming one more than the other. We also do not assume any model about the kind and number of objects in the scene, or the nature of their dynamics. The dense motion information that we compute shall be useful for building such scene-knowledge in an unsupervised manner. The main limitation of our approach is that the point trajectories accumulate drift over time. This needs to be addressed by estimating motion information on a common structural model (built from the bottoms-up) instead of just sequentially matching pairs of reconstructed meshes. Building such a rich spatio-temporal model remains in the future work, and its various directions

for research are detailed in the next section.

The above contributions of our thesis can be thought of as stepping stones towards deriving a stronger and richer spatio-temporal model of the scene from visual data. The earlier chapters 2 3 describe a crude spatio-temporal model at the level of segments and their motions. The latter chapters 4 5 describe a finer model that gives detail about every point on the scene. We have presented these chapters in this manner for easier reading. However, we have made these contributions in the reverse chronological order. As such, we have not yet exploited the synergy between our various contributions in the fullest manner. We describe these and other avenues for future work in the next section.

6.2 Future Work

Wide time-frame correspondence In this thesis, we have concentrated on exploiting temporal information in 3D video with respect to the nearness of frames. We have not investigated thoroughly the more general problem of wide time-frame matching. Many of our contributions - convex segments, local and global shape features have a wider applicability towards matching shapes separated widely apart in time. In such a scenario, we need to use a more sophisticated algorithm for matching on similar lines of [Starck & Hilton 2007a], that minimizes a global error defined on pairwise potentials. This is a promising line of research with applications towards 3D shape retrieval and recognition.

Deriving the globally optimal shape from a mesh sequence Our results of temporally coherent segments and dense motion trajectories can be used to building a globally optimal shape from the various meshes that are reconstructed visually across time. There are two ways in which this problem can be attacked - (i) removing spurious points that appear on the meshes due to silhouette noise, and reducing flicker as reconstructed meshes are played in a video (ii) recovering the subset of points that have a well-defined position from evidence in all the frames. The first objective is related to synthesis (*computer graphics*) and the second is related to analysis (*computer vision*). In the ideal scenario where we discover the globally optimal shape, both the objectives (i) and (ii) shall be fulfilled. But in practice, it is better to attempt solving the problem from both sides independently. There are certain critical decisions that need to be made when building such a shape - on the mesh topology and connectivity between neighboring vertices. We believe our work in this thesis provides a good starting point for reaching such decisions.

Deriving the globally optimal appearance (reflectance) An ideal spatio-temporal model contains not only the globally optimal surface geometry, but also the globally optimal appearance for the shape. In order to model changes in lighting and shadows, this appearance needs to be modeled as surface reflectance. An example is through the bidirectional reflectance distribution function (BRDF) of the

shape. Similar to the earlier problem, deriving the global optimal appearance properties can be attacked from two ways : (i) derive an optimal appearance model that can help relight the scene under widely different illumination conditions (ii) view the problem as that of compression of texture data acquired from multiple cameras, without degradation in the quality of the video rendered as compared to the original. The first objective is related to the synthesis (*computer graphics*) and the second is related to analysis (*computer vision*). Though in the ideal case, both these objectives coincide with each other, it helps to attempt solving the problem from both the sides independently. Particularly, the second formulation of the problem as the compression of multi-view video has immediate practical utility in a multi-camera acquisition system, because images form the bulk of the data in 3D video. Compressing them shall be useful towards more efficient storage and transmission of such data across networks.

Super-resolution capture of 3D video Steps towards finding the globally optimal shape and appearance, as described above, can help integrate the various data captured at multiple frames into a common space of representation. This has the potential to capture the scene at a resolution superior to those of the individual captures. Video super-resolution through tracking has already been a well-studied topic in computer vision. In a multi-camera environment, there is a deeper scope for such applications. Both the surface geometry (with respect to the mesh vertices and connectivity) and appearance (with respect to the number of pixels on which it can be rendered on the screen) can be improved to a super resolution through the spatio-temporal modeling of a 3D video. This is a promising application whose potential needs to be explored.

Tracking longer sequences In this thesis, we have attempted analyzing relatively short sequences of visual reconstructions - ranging from 10 to 200 frames. To analyze longer sequences would require a more modular representation of motion as events and sub-events. Higher level semantics about the agents, theme and instruments of action need to be used - akin to similar tools in human language processing. Spatio-temporal modeling such long events in an unsupervised manner is a new area for research. The use of 3D data, as captured from multi-camera environments rather than a single camera, has a lot of potential in producing high quality results in this area. There shall be several practical applications for this in visual surveillance, sports analysis, medical analysis etc.

Statistical spatio-temporal models : In this thesis, we have treated the various captured sequences as independent and tried to analyze each of them separately. Analyzing these sequences in a united manner is possible through statistical learning. Our framework of unsupervised modeling reduces the level of human intervention for labeling the data. Thereby, much longer and varied sequences can be used for training. The statistical models, thus learnt, can be used for a variety of applica-

tions and help provide a bias (*prior*) for solving several under-constrained problems. One example is the analysis of video captured from a single view, by fitting the parameters of the statistical model to each frame in the observation. The use of such statistical models will increase the applicability of these techniques to more general scenarios, such as the analysis of images and videos on the world wide web.

Bibliography

[4dr 2010] *http://4drepository.inrialpes.fr*, September 2010. 62, 100

[Aanaes *et al.* 2010] Henrik Aanaes, Anders Lindbjerg Dahl and Kim Steenstrup Pedersen. *On recall rate of interest point detectors.* In Proceedings of the 5th International Symposium on 3D Data Processing, Visualization and Processing, Paris, (France), 2010. 87

[Aganj *et al.* 2009] E Aganj, J. P Pons and R Keriven. *Globally optimal spatio-temporal reconstruction from cluttered videos.* In accv, Sept 2009. 29, 111

[Ahmed *et al.* 2008] Naveed Ahmed, Christian Theobalt, Christian Roessl, Sebastian Thrun and Hans-Peter Seidel. *Dense Correspondence Finding for Parametrization-free Animation Reconstruction from Video.* In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, (USA), 2008. 27, 110

[Amenta & Kolluri 2001] N Amenta and R Kolluri. *The medial axis of a union of balls.* Computational Geometry, vol. 20, no. 1-2, pages 25–37, 2001. 41

[Anguelov *et al.* 2004a] D Anguelov, P Srinivasan, D Koller, S Thrun, H Pang and J Davis. *The Correlated Correspondence Algorithm for Unsupervised registration of Non-rigid Surfaces.* In Neural Information Processing Systems, 2004. 28, 29, 71

[Anguelov *et al.* 2004b] Dragomir Anguelov, Daphne Koller, Hoi-Cheung Pang, Praveen Srinivasan and Sebastian Thrun. *Recovering Articulated Object Models from 3D Range Data.* In Uncertainty in Artificial Intelligence, 2004. 28, 71

[Anguelov *et al.* 2005] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun and J. Rodgers. *SCAPE : Shape Completion and Animation of People.* In ACM Transactions on Computer Graphics, pages 408–416, 2005. 27

[Attali *et al.* 2007] Dominique Attali, Jean-Daniel Boissonnat and Herbert Edelsbrunner. *Stability and computation of the medial axis - a state-of-the-art report.* In Mathematical foundations of scientific visualization, computer graphics, and massive data exploration, 2007. 41

[Balan & Black 2008] A Balan and M Black. *The Naked Truth: Estimating Body Shape Under Clothing.* In eccv08, 2008. 28

[Batlle *et al.* 1998] J Batlle, E Mouaddib and J Salvi. *Recent progress in coded structured light as a technique to solve the correspondence problem: a survey.* Pattern Recognition, vol. 31, no. 7, pages 963–982, 1998. 25

[Bay *et al.* 2006] H. Bay, T. Tuytelaars and L. van Gool. *SURF : Speeded up robust features.* In Proceedings of the 9th European Conference on Computer Vision, Graz, (Austria), 2006. 87, 89

[Besl & McKay 1992] Paul J. Besl and Neil D. McKay. *A Method for Registration of 3-D Shapes.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, February 1992. 72

[Bickel *et al.* 2007] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister and M. Gross. *Multi-Scale Capture of Facial Geometry and Motion.* In ACM Computer Graphics (Proceedings SIGGRAPH), 2007. 27

[Biederman 1987] I Biederman. *Recognition-by-components: A theory of human image understanding.* Psychological Review, vol. 94, pages 115–147, 1987. 32

[bod 2010a] *http://www.human-solutions.com*, August 2010. 8

[bod 2010b] *http://www.tc2.com/index_ 3dbodyscan.html*, August 2010. 8

[Bouvrie & Sinha 2007] Jake V. Bouvrie and Pawan Sinha. *Visual object concept discovery: Observations in congenitally blind children, and a computational approach.* Neurocomputing, vol. 70, pages 2218–2233, January 2007. 2

[Bradley *et al.* 2010] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa and Alla Sheffer. *High resolution passive facial performance capture.* In siggraph, volume 29, page 3, 2010. 27

[Breen & Whitaker 2001] D. E. Breen and R. T. Whitaker. *A level-set approach for the metamorphosis of solid models.* IEEE Transaction on Visualization and Computer Graphics, vol. 7, no. 2, pages 173–192, 2001. 116, 117

[Bronstein *et al.* 2006a] Alexander M. Bronstein, Michael M. Bronstein and Ron Kimmel. *Efficient computation of isometry-invariant distances between surfaces.* SIAM Journal of Scientific Computing, vol. 28/5, page 1812=1836, 2006. 85

[Bronstein *et al.* 2006b] Alexander M. Bronstein, Michael M. Bronstein and Ron Kimmel. *Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching.* Proceedings of the National Academy of Sciences, vol. 103/5, pages 1168–1172, January 2006. 69

[Bronstein *et al.* 2007a] Alexander M. Bronstein, Michael M. Bronstein and Ron Kimmel. *Rock, Paper and Scissors: extrinsic vs. intrinsic similarity of non-rigid shapes.* In Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, (Brazil), 2007. 85

[Bronstein *et al.* 2007b] A.M. Bronstein, M.M. Bronstein and R. Kimmel. *Calculus of non-rigid surfaces for geometry and texture manipulation.* IEEE Transaction on Visualization and Computer Graphics, vol. 13(5), pages 902–913, 2007. 85

[Bronstein *et al.* 2009a] A. M Bronstein, M. M. Bronstein, Ron Kimmel, M Mahmoudi and G Sapiro. *A Gromov-Hausdorff framework with diffusion geometry for topologically robust non-rigid shape matching.* ijcv, 2009. 85

[Bronstein *et al.* 2009b] Alexander M. Bronstein, Michael M. Bronstein and Ron Kimmel. *Topology-invariant similarity of nonrigid shapes.* International Journal of Computer Vision, vol. 81/3, pages 281–301, March 2009. 71

[Bustos *et al.* 2005] B. Bustos, A. D. Keim, D. Saupe, T. Schreck and D. V. Vranic. *Feature-based similarity search in 3D object databases.* ACM Computing Surveys, vol. 34, no. 4, pages 345–387, 2005. 87

[Cagniart *et al.* 2009] C Cagniart, E Boyer and S Ilic. *Iterative Mesh Deformation for Dense Surface Tracking.* In IEEE International Workshop on 3-D Digital Imaging and Modeling, October 3-4, Kyoto, Japan, 2009. 27, 108, 110

[Cagniart *et al.* 2010a] C Cagniart, E Boyer and S Ilic. *Free-Form Mesh Tracking : a Patch-Based Approach.* In cvpr, 2010. 27

[Cagniart *et al.* 2010b] C Cagniart, E Boyer and S Ilic. *Probabilistic Deformable Surface Tracking from Multiple Videos.* In eccv, 2010. 27

[Carranza *et al.* 2003] J. Carranza, C. Theobalt, M Magnor and H.-P. Seidel. *Free-Viewpoint Video of Human Actors.* Proc. ACM Siggraph'03, San Diego, USA, pages 569–577, July 2003. 11, 26

[Chang & Zwicker 2008] Will Chang and Matthias Zwicker. *Automatic Registration for Articulated Shapes.* eurographics, vol. 27, no. 5, 2008. 71, 88

[Chazal & Lieuter 2005] Frederic Chazal and Andre Lieuter. *The λ-medial axis.* Graphical Models, vol. 67, no. 4, pages 304–331, 2005. 37, 41

[Chazelle *et al.* 1995] B Chazelle, D. Dobkin, N. Shouraboura and A. Tal. *Strategies for polyhedral surface decomposition: An experimental study.* In 11th Annual ACM Symposium on Computational Geometry, pages 297–305, 1995. 33

[Choi *et al.* 1997] H. Choi, S. Choi and H. Moon. *Mathematical theory of the medial axis transform.* Pacific Journal of Mathematics, vol. 181, no. 1, pages 57–88, 1997. 39, 55

[Chui *et al.* 2004] H. Chui, J. Zhang and A. Rangarajan. *Unsupervised learning of an atlas from unlabeled point-sets.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26(2), pages 160–173, 2004. 86

[Courchay *et al.* 2009] J Courchay, J. P Pons, P Monasse and R Keriven. *Dense and accurate spatio-temporal multi-view stereovision.* In accv, Sept 2009. 29, 111

[Cuzzolin *et al.* 2008] Fabio Cuzzolin, Diana Mateus, David Knossow, Edmond Boyer and Radu Horaud. *Coherent Laplacian 3D protrusion segmentation.* In Computer Vision and Pattern Recognition, 2008. 71

[Dalal & Triggs 2005] N. Dalal and B. Triggs. *Histograms of oriented gradients for human detection.* In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Diego, (USA), pages 886–893, 2005. 87

[Davis 2001] Larry S. Davis, editeur. Foundations of image understanding, chapitre Volumetric scene reconstruction from multiple views (Charles Dyer). Numeéro 16. Kluwer Academic Publishers, 2001. 24

[de Aguiar *et al.* 2007a] E. de Aguiar, C. Theobalt, C. Stoll and H.P. Seidel. *Marker-less Deformable Mesh Tracking for Human Shape and Motion Capture.* In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, (USA), 2007. 108, 112

[de Aguiar *et al.* 2007b] Edilson de Aguiar, C Theobalt, C Stoll and H. P Seidel. *Rapid Animation of Laser-scanned Humans.* In Proceedings of IEEE Virtual Reality 2007, pages 223–226, 2007. 27

[de Aguiar *et al.* 2008a] Edilson de Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, H. P Seidel and Sebastian Thrun. *Performance Capture from Sparse Multi-view Video.* In ACM Transactions on Computer Graphics, 2008. 27, 71

[de Aguiar *et al.* 2008b] Edilson de Aguiar, Christian Theobalt, Sebastian Thrun and Hans-Peter Seidel. *Automatic Conversion of Mesh Animations into Skeleton-based Animations.* Transactions of the Computer Graphics Forum (Proceedings of EUROGRAPHICS), vol. 27, no. 2, 2008. 71

[DeCarlo & Metaxas 2000] D. DeCarlo and D. Metaxas. *Optical Flow Constraints on Deformable Models with Applications to Face Tracking.* International Journal of Computer Vision, vol. 38(2), pages 99–127, 2000. 26

[Delaunoy *et al.* 2008] A Delaunoy, E Prados, P Gargallo, J. P Pons and P Sturm. *Minimizing the multi-view stereo reprojection error for triangular surface meshes.* In bmvc, 2008. 25

[Dey & Zhao 2002] T. K. Dey and W. Zhao. *Approximating the medial axis from the voronoi diagram with a convergence guarantee.* In 10th Annual European Symposium on Algorithms (ESA), pages 387–398, Heidelberg, 2002. Springer. 55

[Dretske 1988] Fred Dretske. Seeing and knowing. Numeéro 0226162454 de 978. University of Chicago Press, April 1988. 4

[Franco & Boyer 2003] Jean-Sebastien Franco and Edmond Boyer. *Exact Polyhedral Visual Hulls.* In bmvc, volume 1, pages 329–338, 2003. 16, 24

[Franco *et al.* 2004] Jean-Sebastien Franco, Clement Menier, Edmond Boyer and Bruno Raffin. *A distributed approach for real-time 3D modeling.* In CVPR : Workshop on Real-Time 3D Sensors and their Applications, July 2004. 16

[Furukawa & Ponce 2006] Y. Furukawa and J. Ponce. *Carved Visual Hulls for Image-Based Modeling.* In Proceedings of the 9th European Conference on Computer Vision, Graz, (Austria), 2006. 24, 111

[G *et al.* 2002] G. Zigelman G, R.Kimmel and N. Kiryati. *Texture mapping using surface flattening via multidimensional scaling.* IEEE Transactions on Visualization and Computer Graphics, vol. 8(2), pages 198–207, 2002. 85

[Gavrila & Davis 1996] D. Gavrila and L. Davis. *3-D model-based tracking of humans in action: a multi-view approach* . In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, (USA), 1996. 26

[Gebal *et al.* 2009] K. Gebal, J. A. Baerentzen, H Aanaes and R. Larsen. *Shape analysis using the auto diffusion function.* In sgp, volume 28, 2009. 85, 86, 96

[Giesen *et al.* 2009] J Giesen, B Miklos, M Pauly and C Wormser. *The scale axis transform.* In In Proceedings of the Annual Symposium on Computational Geometry, pages 106–115, 2009. 42

[Gleicher 1998] M. Gleicher. *Retargetting Motion to New Characters.* In ACM Transactions on Computer Graphics, pages 33–42, 1998. 26

[Golovinskiy & Funkhouser 2009] Aleksey Golovinskiy and Thomas Funkhouser. *Consistent Segmentation of 3D Models.* In ACM Transactions on Computer Graphics (Proceedings of SIGGRAPH), 2009. 71

[Hernandez & Schmitt 2004] C. Hernandez and F. Schmitt. *Silhouette and stereo fusion for 3D object modeling.* Computer Vision and Image Understanding, vol. 96, no. 3, pages 367–392, 2004. 24, 111

[Hernandez-Esteban *et al.* 2008] C Hernandez-Esteban, G Vogiatzis and R Cipolla. *Multiview Photometric Stereo.* pami, vol. 30, no. 3, pages 548–554, March 2008. 25, 111

[Hernandez *et al.* 2007] C Hernandez, G Vogiatzis, G. J. Brostow, B Stenger and R Cipolla. *Non-rigid photometric stereo with colored lights.* In iccv, pages 1–8, 2007. 25

[Hilaga *et al.* 2001] M. Hilaga, Y. Shinagawa, T. Kohmura and T.L. Kunii. *Topology matching for fully automatic similarity estimation of 3D shapes.* In ACM Computer Graphics (Proceedings SIGGRAPH), 2001. 85, 94, 95

[Horn 1986] B. K. P. Horn. Robot vision. MIT Press, 1986. 25

[Hu & Hua 2009] J Hu and J. Hua. *Salient spectral geometric features for shape matching and retrieval.* The Visual Computer, vol. 25, no. 5-7, pages 667–675, 2009. 85, 87

[Huang *et al.* 2009] Qi-Xing Huang, Martin Wicke, Bart Adams and Leonidas Guibas. *Shape Decomposition using Modal Analysis.* Transactions of the Computer Graphics Forum (Proceedings of EUROGRAPHICS), vol. 28, no. 2, 2009. 36

[Ikeuchi & Horn 1981] K Ikeuchi and B. K. P. Horn. *Numerical shape from shading and occluding boundaries.* Artificial Intelligence, vol. 17, no. 1-3, pages 141–184, August 1981. 25

[Johnson & Hebert 1999] Andrew E. Johnson and Martial Hebert. *Using spin images for efficient object recognition in cluttered 3D scenes.* pami, vol. 21, no. 5, pages 433–449, 1999. 28, 87

[Kakadiaris & Metaxas 2000] I. Kakadiaris and D. Metaxas. *Model-Based Estimation of 3D Human Motion.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pages 1453–1459, december 2000. 26

[Kanade & Narayanan 2007] T Kanade and P Narayanan. *Virtualized Reality : Perspectives on 4D Digitization of Dynamic Events.* IEEE Computer Graphics and Applications, vol. 27, no. 3, pages 32–40, 2007. 24

[Katz *et al.* 2005] Sagi Katz, George Leifman and Ayellet Tal. *Mesh segmentation using feature point and core extraction.* The Visual Computer, vol. 21, pages 649–658, 2005. 35, 36

[Kawasaki *et al.* 2008] Hiroshi Kawasaki, Ryo Furukawa, Ryusuke Sagawa and Yasushi Yagi. *Dynamic scene shape reconstruction using a single structured light pattern.* In cvpr08, 2008. 25

[Kläser *et al.* 2008] Alexander Kläser, Marcin Marszałek and Cordelia Schmid. *A Spatio-Temporal Descriptor based on 3D Gradients.* In bmvc, pages 995–1004, 2008. 87, 94

[Knossow *et al.* 2008] David Knossow, Remi Ronfard and Radu Horaud. *Human Motion Tracking with a Kinematic Parametrization of Extrema Contours.* ijcv, vol. 79, no. 2, pages 247–269, 2008. 27

[Kobbelt *et al.* 2000] L Kobbelt, T Bareuther and H. P Seidel. *Multiresolution shape deformations for meshes with dynamic vertex connectivity.* In eurographics, pages 249–260, 2000. 89

[Konickx & Van Gool 2006] Thomas P. Konickx and Luc Van Gool. *Real-Time Acquisition by Adaptive Structured Light.* pami, vol. 28, no. 3, pages 432–445, 2006. 25

[Labatut *et al.* 2007] P Labatut, J. P Pons and R Keriven. *Efficient multi-view reconstruction of large-scale scenes using interest points, Delaunay triangulation and graph cuts.* In iccv07, 2007. 25

[Laurentini 1994] A Laurentini. *The visual hull concept for silhouette-based image understanding.* pami, 1994. 24

[Leordeanu & Hebert 2005] Marius Leordeanu and Martial Hebert. *A Spectral Technique for Correspondence Problems using Pairwise Constraints.* In Proceedings of the International Conference on Computer Vision, pages 1482–1489, 2005. 88

[Li *et al.* 2008] X. Li, A. Godil and A. Wagan. *Spatially enhanced bags of words for 3D shape retrieval.* In Proceedings of the International Symposium on Visual Computing (ISVC), pages 349–358, 2008. 88

[Li *et al.* 2009] Hao Li, Bart Adams, Leonidas Guibas and Mark Pauly. *Robust single-view geometry and motion reconstruction.* In ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia), 2009. 27, 29

[Li *et al.* 2010] Hao Li, Thibaut Weise and Mark Pauly. *Example-Based Facial Rigging.* In siggraph, 2010. 28

[Lien & Amanto 2007] Jyh-Mig Lien and Nancy M. Amanto. *Approximate Convex Decomposition of Polyhedra.* In Proceedings of the ACM Symposium on Solid and Physical Modeling, pages 121–131, 2007. 33, 34, 59, 60, 61

[Lipman & Funkhouser 2009] Yaron Lipman and Thomas Funkhouser. *Mobius Voting for Surface Correspondence.* In ACM Transactions on Computer Graphics, 2009. 86

[Lipman *et al.*] Y. Lipman, D. Levin and D. Cohen-Or. 110

[Lipman *et al.* 2005] Y. Lipman, O. Sorkine, D. Levin and D. Cohen-Or. *Linear Rotation-Invariant Coordinates for Meshes.* In ACM Transactions on Computer Graphics, volume 24, pages 479–487, 2005. 110

[Liu *et al.* 2009] Rong Liu, Hao Zhang, Ariel Shamir and Daniel Cohen-Or. *A Part-Aware Surface Metric for Shape Analysis.* In Transactions of the Computer Graphics Forum (Proceedings of EUROGRAPHICS), 2009. 37, 42, 43

[Lowe 2004] David Lowe. *Distinctive image features from scale-invariant keypoints.* ijcv, vol. 60, no. 2, pages 91–110, 2004. 87, 89

[Marr 1977] David Marr. *Analysis of occluding contour.* In Proceedings of the Royal Society of London, pages 441–475, 1977. 32

[Mateus *et al.* 2008] D. Mateus, R. Horaud, D. Knossow, F. Cuzzolin and E. Boyer. *Articulated shape matching using Laplacian Eigenfunctions and unsupervised point registration.* In cvpr, 2008. 27, 85, 86

[Matusik *et al.* 2000] W Matusik, C Buehler, R Raskar, S Gortler and L McMilan. *Image based visual hulls.* acmcg, 2000. 24

[Mikolajczyk & Schmidt 2005] K. Mikolajczyk and C. Schmidt. *A performance evaluation of local desecriptors.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pages 1615–1630, 2005. 87

[Montagnat *et al.* 2000] J. Montagnat, H. Delingette, N. Scapel and N. Ayache. *Representation, shape, topology and evolution of deformable surfaces. Application to 3D medical image segmentation.* Rapport technique 3954, INRIA, 2000. 26

[Mori *et al.* 2001] G. Mori, S. Belongie and J. Malik. *Shape contexts enable efficient retrieval of similar shapes.* In cvpr, 2001. 87

[Narayanan *et al.* 1998] P Narayanan, P Rander and T Kanade. *Constructing Virtual Worlds from Dense Stereo.* In iccv, 1998. 24

[Neumann & Aloimonos 2002] J. Neumann and Y. Aloimonos. *Spatio-Temporal Stereo Using Multi-Resolution Subdivision Surfaces.* International Journal of Computer Vision, vol. 47, pages 181–193, 2002. 28

[Novatnack & Nishino 2007] J. Novatnack and K. Nishino. *Scale-dependent 3D geometric features.* In Proceedings of the International Conference on Computer Vision, 2007. 87

[Novotni *et al.* 2005] M. Novotni, P. Degener and R. Klein. *Correspondence generation and matching of 3D shape subparts.* Rapport technique CG-2005-2, ISSN 1610-8892, Friedrich-Wilhelms-Universitat Bonn, June 2005. 87

[Osher & Fedkiw 2003] S. Osher and R. Fedkiw. Level set methods and dynamic implicit surfaces. Springer, 2003. 111

[Pons *et al.* 2007a] J. P Pons, R Keriven and 0 Faugeras. *Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score.* International Journal of Computer Vision, vol. 72(2), pages 179–193, 2007. 25, 111, 117

[Pons *et al.* 2007b] J. P Pons, R Keriven and O Faugeras. *Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score.* ijcv, vol. 72, no. 2, pages 179–193, April 2007. 24, 28

[Popa *et al.* 2009] T Popa, Q Zhou, D Bradley, V Kraevoy, H Fu, A Sheffer and W Heidrich. *Wrinkling captured garments using space-time data-driven deformation.* In eurographics, 2009. 27

[Popa *et al.* 2010] Tiberiu Popa, Ian South-Dickinson, Derek Bradley, Alla Sheffer and Wolfgang Heidrich. *Globally consistent space-time reconstruction.* In sgp, 2010. 29, 111

[Pulli 1997] K Pulli. *Surface Reconstruction and Display from Range and Color Data.* PhD thesis, University of Washington, 1997. 73

[Salzmann *et al.* 2007] M. Salzmann, J.Pilet, S.Ilic and P.Fua. *Surface Deformation Models for Non-Rigid 3–D Shape Recovery.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, pages 1481–1487, 2007. 26

[Schlattmann *et al.* 2008] M. Schlattmann, P. Degener and R. Klein. *Scale space based feature point detection on surfaces.* Journal of WSCG, vol. 16, no. 1-3, Feb 2008. 87

[Schnabel *et al.* 2007] R. Schnabel, R. Wahl and R. Klein. *Efficient ransac for point-cloud shape detection.* cgf, vol. 26, pages 214–226, June 2007. 87

[Shamir 2008] Ariel Shamir. *A Survey on Mesh Segmentation Techniques.* Computer Graphics Forum, vol. 27, pages 1539–1556, 2008. 33

[Shapira *et al.* 2008] Lior Shapira, Ariel Shamir and Daniel Cohen-Or. *Consistent mesh partitioning and skeletonisation using the shape diameter function.* The Visual Computer, vol. 24, pages 249–259, 2008. 35, 42, 55, 71, 73

[Shi *et al.* 2007] Xiaohan Shi, Kun Zhou, Yiying Tong, Mathieu Desbrun, Hujun Bao and Baining Guo. *Mesh Puppetry : Cascading Optimization of Mesh Deformation with Inverse Kinematics.* In ACM Transactions on Computer Graphics, 2007. 110

[Shilane & Funkhouser 2006] P. Shilane and T. Funkhouser. *Selecting distinctive 3D shape descriptors for similarity retrieval.* In smi, 2006. 87, 88

[Siddiqi & Kimia 1995] K. Siddiqi and B. B. Kimia. *Parts of visual form: Computational aspects.* pami, vol. 17, no. 3, pages 239–251, 1995. 32

[Sorkine & Alexa 2007] Olga Sorkine and Marc Alexa. *As-Rigid-As-Possible Surface Modeling.* In Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing, 2007. 36, 110

[Sorkine 2005] O. Sorkine. *Laplacian mesh processing.* In Eurographics Conference, 2005. 111, 112, 115

[Starck & Hilton 2005] J. Starck and A. Hilton. *Spherical matching for temporal correspondence of non-rigid surfaces.* In iccv, pages 1387–1394, 2005. 26, 86

[Starck & Hilton 2007a] J Starck and A Hilton. *Correspondence Labelling for Wide-timeframe Free-form Surface Matching.* In Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, (Brazil), 2007. 28, 71, 88, 100, 108, 125

[Starck & Hilton 2007b] J Starck and A Hilton. *Surface capture for performance-based animation.* IEEE Computer Graphics and Applications, vol. 27, no. 3, pages 21–31, May 2007. 25, 28, 117

[Sumner & Popovic 2004] Robert W. Sumner and Jovan Popovic. *Deformation Transfer for Triangle Meshes.* In ACM Transactions on Computer Graphics, 2004. 26

[Sumner *et al.* 2007] Robert W. Sumner, Johannes Schmid and Mark Pauly. *Embedded Deformation for Shape Manipulation.* In ACM Transactions on Computer Graphics (Proceedings of SIGGRAPH), 2007. 110

[Sun *et al.* 2009] Jian Sun, Maks Ovsjanikov and Leonidas Guibas. *A concise and provably informative multi-scale signature based on heat diffusion.* In sgp, volume 28, page 2009, 2009. 86, 96

[Szeliski 2010] Richard Szeliski. Computer vision : Algorithms and applications. Kluwer Academic Publishers (unprinted yet), 2010. 24

[Tevs *et al.* 2009] A Tevs, M Bokeloh, M Wand, A Schilling and H-P. Seidel. *Isometric registration of ambiguous and partial data.* In cvpr, 2009. 28

[Thorstensen & Keriven 2009] N Thorstensen and R Keriven. *Non-rigid shape matching using geometry and photometry.* In accv, Sept 2009. 86, 111

[Ukita *et al.* 2009] N Ukita, M Hirai and M Kidode. *Complex Volume and Pose Tracking with Probabilistic Dynamical Models and Visual Hull Constraints.* In iccv, 2009. 28

[van Kaick *et al.* 2010] Oliver van Kaick, Hao Zhang, Ghassan Hamarneh and Daniel Cohen-Or. *A survey on shape correspondence : state of the art report (STAR).* In eurographics, 2010. 85

[Varanasi & Boyer 2010] K. Varanasi and E. Boyer. *Temporally coherent segmentation of 3D reconstructions.* In Proceedings of the 5th International Symposium on 3D Data Processing, Visualization and Processing, Paris, (France), 2010. 32, 67

[Varanasi *et al.* 2008] K. Varanasi, A. Zaharescu, E. Boyer and R. Horaud. *Temporal Surface Tracking using Mesh Evolution.* In Proceedings of the 10th European Conference on Computer Vision, Marseille, (France), volume 2, pages 30–43, 2008. 107

[Vedula *et al.* 2002] S Vedula, S Baker and T Kanade. *Spatio-temporal view interpolation.* In eurographics, 2002. 28

[Vedula *et al.* 2005a] S Vedula, S Baker and T Kanade. *Image-based spatio-temporal modeling and view interpolation of dynamic scenes.* ACM Transactions on Graphics, vol. 24, no. 2, pages 240–261, April 2005. 28

[Vedula *et al.* 2005b] S Vedula, S Baker, P Rander, R Collins and T Kanade. *Three-Dimensional Scene Flow.* pami, vol. 27, no. 3, pages 475–480, March 2005. 28

[Vlasic *et al.* 2008] Daniel Vlasic, Ilya Baran, Wojciech Matusik and Jovan Popovic. *Articulated Mesh Animation from Multi-view Silhouettes.* In ACM Transactions on Computer Graphics, pages 97:1–97:9, 2008. 27, 108, 110

[Vlasic *et al.* 2009] D Vlasic, P Peers, I Baran, P Debevec, J Popovic, S Rusinkiewicz and W Matusik. *Dynamic shape capture using multi-view photometric stereo.* acmcg, vol. 28, no. 5, page 174, 2009. 28

[Wand *et al.* 2009] M. Wand, B. Adams, M. Ovsjanikov, A. Berner, M. Bokeloh, P. Jenke, L. Guibas, H-P. Seidel and A. Schilling. *Efficient Reconstruction of Nonrigid Shape and Motion from Real-Time 3D Scanner Data.* ACM Transactions on Computer Graphics, vol. 28, no. 2, April 2009. 29, 111

[Weber *et al.* 2007] Ofir Weber, Olga Sorkine, Yaron Lipman and Craig Gotsman. *Context-Aware Skeletal Shape Deformation.* In Transactions of the Computer Graphics Forum (Proceedings of EUROGRAPHICS), volume 26, 2007. 110

[Weise *et al.* 2007] Thibaut Weise, Bastian Leibe and Luc Van Gool. *Fast 3D Scanning with Automatic Motion Compensation.* In cvpr, june 2007. 25

[Weise *et al.* 2009] Thibaut Weise, Hao Li, Luc Van Gool and Mark Pauly. *Face/Off: Live Facial Puppetry.* In ACM Siggraph/Eurographics Symposium on Computer Animation, 2009. 28

[Wenger *et al.* 2005] A Wenger, A Gardner, C Tchou, J Unger, T Hawkins and P Debevec. *Performance Relighting and Reflectance Transformation with Time-Multiplexed Illumination.* acmcg, vol. 24, no. 3, pages 756–764, July 2005. 25, 111

[Wong & Cipolla 2007] S. F. Wong and R. Cipolla. *Extracting spatiotemporal interest points using global information.* In Proceedings of the International Conference on Computer Vision, 2007. 87

[Woodham 1980] R Woodham. *Photometric method for determining surface orientation from multiple images.* Optical Engineering, vol. 19, no. 1, pages 139–144, 1980. 25

[Wu *et al.* 2008] C. Wu, B. Clipp, X. Li, J. M. Frahm and M. Pollyfeys. *3D model matching with viewpoint invariant patches (vips).* In Computer Vision and Pattern Recognition, 2008. 87

[Zaharescu *et al.* 2007] A. Zaharescu, E. Boyer and R. Horaud. *TransforMesh: a topology-adaptive mesh-based approach to surface evolution.* In Proceedings of the 8th Asian Conference on Computer Vision, Tokyo (Japan), 2007. 25, 100, 111, 112, 116

[Zaharescu *et al.* 2009] A. Zaharescu, E. Boyer, K. Varanasi and R. Horaud. *Surface feature detection and description with applications to mesh tracking.* 2009. 83, 89, 100, 102

[Zeng *et al.* 2010] Y. Zeng, C. Wang, Y. Wang, X. Gu, D. Samaras and Paragios N. *Dense non-rigid surface registration using high-order graph matching.* In cvpr, 2010. 86

[Zhang & Hebert 1999] Dongmei Zhang and Martial Hebert. *Harmonic Maps and Their Applications in Surface Matching.* In Computer Vision and Pattern Recognition, pages 2524–2530, 1999. 85

[Zhang *et al.* 2007] Hao Zhang, Oliver van Kaick and Ramsay Dyer. *Spectral Methods for Mesh Processing and Analysis.* In State of the Art Report, Proceedings of Eurographics, 2007. 85

[Zhang *et al.* 2008] Hao Zhang, Alla Sheffer, Daniel Cohen-Or, Qingnan Zhou, Oliver van Kaick and Andrea Tagliasacchi. *Deformation-driven Shape Correspondence.* In Proceedings of the 5th Eurographics Symposium on Geometry Processing, 2008. 86

Spatio-Temporal Modeling of Dynamic 3D Scenes from Visual Data

Abstract: This thesis addresses the problem of modeling the time-varying shape of a dynamic scene, as seen from a sparse multi-camera system. No restrictions are placed on the scene - either on the type and number of the actors in the scene, or on the nature of their interactions. *A priori*, computer vision algorithms of surface reconstruction based on silhouettes and multi-view stereo are performed individually at various frames, to obtain a sequence of meshes in 3D. These meshes are not consistent with each other, either geometrically or topologically.This thesis makes two contributions towards obtaining a coherent spatio-temporal model of the underlying scene : (1) obtaining a temporally coherent segmentation of the mesh sequence, identifying parts that are moving in an approximately rigid manner (2) estimating dense 3D motion on the surface through a mesh evolution framework, that handles topological and geometrical inconsistencies amidst individual mesh reconstructions. In order to make these contributions, various sub-problems are solved. A new method of mesh segmentation is presented that is suited for visually reconstructed meshes, which is robust to the geometric artifacts that occur in such reconstructions. Segments extracted at various frames are then refined and repositioned into a consistent segmentation of the mesh sequence. As a by-product of this scheme, articulated motion is estimated on the scene, as a set of rigid body transformations of each segment over the sequence. This pair of motion estimates and temporally consistent shape, constitute the underlying spatio-temporal model of the scene, which is captured at a coarse level by the segmentation scheme. The latter chapters of the thesis explain how to estimate motion densely, to capture true non-rigid deformation of surfaces. In order to achieve this, sparse features are detected on the surfaces and matched across time, in a geometrically valid manner. These sparse matches provide an initialization for estimating motion densely over the surface. A mesh evolution framework maps meshes reconstructed at two frames completely, accounting for the topological and geometrical inconsistencies, providing a means for transferring motion estimates from one surface to the other. Dense 3D trajectories of surface points are thereby estimated over long sequences.
The proposed methods are systematically tested on diverse and challenging datasets, which are made publicly available on the web. Qualitative and quantitative experimental results are presented in each chapter to validate the approach adopted.

Keywords: Spatio-temporal modeling, scene dynamics, 3D shape over time, motion segmentation at various scales, dynamic scene analysis, mesh evolution, mesh matching, mesh segmentation, non-rigid registration, non-rigid motion estimation, spatio-temporal reconstruction, 4D reconstruction, 4D segmentation