1-1-2011

Deformable meshes for shape recovery: models and applications

Chang Liu
Wayne State University,

DEFORMABLE MESHES FOR SHAPE RECOVERY: MODELS AND APPLICATIONS

by

CHANG LIU

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2011

MAJOR: COMPUTER SCIENCE

Approved by:

____

Advisor Date

____

____

____

DEDICATION

This dissertation is dedicated to whoever has loved and supported me during my years of Ph.D. study including but not limited to the people mentioned in the next chapter.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Dr. Jing Hua. Dr. Jing Hua is an excellent educator and researcher. His dedication and positive attitude for research has greatly encouraged me in my own research work especially during the difficult times. He has been available to me all the time whenever I needed his feedback. This dissertation would not haven been possible without him.

I also thank Dr. Jie Shen, Dr. Ming Dong, and Dr. Farshad Fotouhi for serving in my prospectus and dissertation committee and giving constructive suggestions and comments on the dissertation.

Last but not least, I thank all my colleagues in Graphics & Imaging Lab: Dr. Guangyu Zou, Dr. Yunhao Tan, Dr. Zhaoqiang Lai, Jiaxi Hu, Darshan Pai, and Vahid Taimouri, for having insightful discussions with me.

On the personal front, I deeply acknowledge the love and encouragement from my parents and my wife. I hope I have made them all proud.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In computer graphics, mesh is widely used in modelling and visualization of 3D objects. It is a simple and versatile representation of 3D geometries compared to splines and other analytical models. In particular, deformable meshes are meshes which can take the deformation of deformable objects. They extend the modelling ability of meshes. Although the research of meshes has been active for decades the latest development of computer graphics continues to bring new challenges for mesh based applications where further work is needed.

1.1 Problem Statement

With the advance of scanning and imaging technology, more and more 3D objects become available. In design and manufacture industry, 3D models are used as making prototypes and for simulation. Scanned models are also important resources for reverse engineering. In entertainment industry, such models are also prevailing in producing special effects and making virtual worlds/characters. In health and medical fields, accurate quantitative analysis is key to better dose prescription and surgery path planning; compared to traditional 2D images, 3D reconstruction of certain organs and lesion areas can provide field experts not only with better visual feedback for diagnosis but also with accurate information such as location, size, and shape of the region of interest (ROI) for historical and cross-subject comparison. Among these 3D data, deformable objects have gained increasing interests. They include medical instances such as organs, a sequence of objects in motion, and objects of similar shapes where a meaningful correspondence can be established between each other. On one hand, it is because the traditional expert hardware such as 3D cameras and tagged MRI equipment is becoming prevailing in recent years; thus a significant number of deformable objects are captured and stored, which provides good testing databases and easy-to-fetch construction components. On the other hand, there is an increasing demand of more advanced applications of 3D techniques in graphics and imaging involving deformable objects; with one dimensional increase, a lot of research begins to realize some of the algorithms which are successful on 2D images or other types of data on these new 3D models in an efficient and robust way. However, before any high level application can be put into practice, with so many 3D objects digitalized, it requires tools to store, compare, and retrieve them in the first place. In another word, we need "Google Shape" to start the journey. Rather than just searching, a more general and fundamental group of operations called shape recovery is needed. Shape recovery is the task to retrieve an object from the environment where its geometry is hidden or implicitly known. The shape information can be hidden in images or a group of similar shapes. Shape recovery usually begins with a query that is a description of a target shape. It can be a similar shape of the target or an initial guess based on prior knowledge. The recovering process is the searching for the target based on the query. The main challenges involved are:

1. Measure the similarity between two shapes or between a shape and the prior knowledge about a shape.

2. Increase the efficiency of the searching based on a certain similarity measurement.

Both of the problems can be approached through a shape deformation framework. The shape similarity can be measured in the process of deforming one shape to another while the efficiency issue will be addressed by designing proper deformation strategies.

It is worth noting that meshes are ideal tools for modelling 3D objects under general deformation. Meshes can model both manifold and non-manifold. It can also satisfy many different smoothness constrains. It can handle surface deformation in a direct and most "natural" way. In fact meshes are widely used every field mentioned above even in CAD where parametrized surfaces take up a significant portion. In this dissertation, deformable meshes are used to solve the aforementioned challenges in shape recovery under different conditions.

1.2 Contributions

A polygon mesh can be used as the explicit representation of a shape. The topology of the shape is modelled as the connectivity of the mesh while the area and internal angles of each polygon reflects its geometry. Besides, the differential measurements can be computed via these quantities [52]. Through the mesh representation, a shape can be easily changed by splitting/merging/transforming mesh elements. In our research, triangle meshes are used instead of general polygon meshes since triangle meshes are the most versatile and well studied. They are especially good for shape editing and deforming due to the simplex property.

Two major methods for modelling deformable meshes are: space modelling, for example, free form deformation (FFD) and surface based modelling. This dissertation mainly adopts the latter approach where the global deformation of a mesh is treated as the result of a series of local deformations of the mesh surface. Different approaches are taken under several different conditions. When the deformation is unknown, it is modelled in a way that the deforming mesh is kept regular so that the same type of deformation can continue until some criteria are fulfilled. This is useful in segmenting deformable objects where only little prior knowledge is known about the hidden shape. When the deformation is known implicitly, for example, one instance of the object similar to the shape being recovered is given; a feature extraction method is used to pick the shape out of the crowd. The features are organized using a graph which maintains the topological information of the shape and is efficient to compare. When the deformation is explicitly known, a clustering algorithm is designed to figure out the minimum/representative set of transformations that can recover the shape most faithfully for efficiency purposes. The contributions of the included work are listed below.

1. A novel mesh deformation framework based on geometric flow is devised (Chapter 3). It allows geometry changes under internal and external forces as well as topological changes such as splitting and merging. It outperforms other deformable models by its simple implementation using meshes and its versatility of modeling arbitrary non-smooth

shapes. The model is successfully applied to robust medical image segmentation.

2. A novel shape abstraction framework is devised for deformable meshes (Chapter 4). A manifold mesh is simplified to a graph of geometry features of interest. A new similarity measurement based on graph matching is proposed for comparing deformable objects more efficiently. The model is successfully applied to 3D shape retrieval.

3. By extending the shape abstraction algorithm, a generic motion abstraction framework is presented for deformable meshes based on local transformation clustering. It can model detailed geometry changes of the mesh instead of only some properties. Thus, it provides a novel approach for 3D motion compression (Chapter 5). It differs from previous work in that it is totally based on local geometry transformation rather than global vertex displacement. It can also be used on non-skeleton driven motions directly. This work can be seen as an extension of our shape abstraction algorithm where motion abstraction is achieved.

Although most of the applications use two manifold triangle meshes, most of the methods developed in this dissertation can be extended to arbitrary polygon meshes.

1.3 Dissertation Organization

The following contents are organized as below:

- Chapter 2 reviews the related work to the main techniques used in the dissertation.

- Chapter 3 presents a PDE deformable model and its application in medical image segmentation.

- Chapter 4 presents a surface abstraction framework for 3D shape retrieval.

- Chapter 5 describes a generic motion abstraction framework for 3D motion compression framework based on local transformation.

- Chapter 6 concludes the contributions of the dissertation and points out the future work along this research direction.

CHAPTER 2

BACKGROUND

In this chapter, the latest development of mesh deformation methods and some representative work of deformable models are reviewed for the purpose of understanding the techniques used in the following chapters.

2.1 Mesh Deformation Methods

Mesh based deformations are widely used in computer animation since meshes are easy to acquire, store and render. Mesh deformation methods can be classified into space deformation methods and surface deformation methods. Space deformation methods directly deform the embedding space (usually a subspace) of the mesh, i.e., the coordinates. Classical methods in this category use some control nets to manipulate the embedding space. For example, Radial basis function (RBF) based deformation interpolates the displacements of a few handles over the embedded space to achieve global deformation; skeleton based methods associate the surface mesh with a skeleton subspace to make the mesh follow the motion of the skeleton. Spatial deformation methods do not directly work with the mesh so they are versatile with respect to inputs and the embedding space; however they lack control of more precise surface properties, where surface deformation methods offer a remedy. On the contrary, it is challenging for the surface deformation methods to maintain global properties of the mesh. The applications in this dissertation require the surface geometry to be preserved, thus surface deformation methods are favored. Some of the representative works are reviewed in the following section.

2.1.1 Displacement Based Methods

Displacement based methods model the mesh deformation by moving mesh vertices directly. When the displacements cannot be defined for some vertices, they have to be estimated or constrained according to the application. Most of the literature use displacement based methods for mesh editing when only a few vertex displacements are known while the geometry of the shape at certain scales is to be preserved. A popular criterion is to preserve the local differential coordinates of the vertices. The most commonly used differential coordinates are based on the Laplacian operator [4]. This operator results in a differential equation of the following kind:

$$\nabla^2 \mathbf{u} = \delta \mathbf{F(U)u}||_{\partial\Omega} = \mathbf{u}_0||_{\partial\Omega} \tag{2.1}$$

where $\nabla^2$is the vector Laplacian, **u** is an unknown vector function, δ is a known vector, **F** is an unknown vector of linear functions applied to **u**. $\mathbf{u}_0$ is the boundary condition of **u** on Ω.

However, local rotation and scaling will not be reflected by naively preserving the Laplacian coordinates. Thus, Yaron and et al. first explored the following property of differential coordinates [43]:

$$R \cdot L(p_j) = L(R \cdot p_j) \tag{2.2}$$

where L is the transformation from absolute to differential coordinates and R a global rotation(in fact affine transformation) applied to the entire mesh. In order to deform the differential coordinates to reflect the rotation of absolute coordinates, the authors proposed to correct the Laplacian coordinates by explicitly apply a rotation matrix to each vertex. These matrices are estimated by comparing the normals of the original mesh and the deformed mesh. Since the global differential coordinates are not affine invariant in general, Oscar et al. [7] proposed another method which decomposes the vertex coordinates into two parts, namely, geometric part and parametrization part. For a vertex v_i, let q_i be its projection on the plane defined by its one-ring neighborhood. The geometric part is defined as $l_i = v_i - q_i$, and the parametrization is the encoding of q_i with respect to the vertices in the one-ring neighborhood in the plane. The geometric part is actually the Laplacian coordinates calculated using a different weighing scheme, the weights are treated as the parametrization of vertex w.r.t. its neighbors. During the deformation, instead of preserving the Laplacian coordinates directly, these two parts are preserved individually. Another nonlinear variation of vertex based method is depicted in [44]. The paper presents two discrete forms for triangle meshes. The first one encodes the relationship between local frames and the second one encodes the vertex positions in the local frames. The system is similar to that in [7]; however the global shape is constrained by the discrete forms rather than the Laplacian coordinates. The main advantage of [44] is that the global and local deformation can be separated the modeled respectively.

2.1.2 Transformation Based Methods

Transformation based methods model the surface deformation in terms of the transformations of mesh elements (vertices, faces, and etc.). All the transformations have to be known in advance. One representative work is by Yu [82]. In [82], vertex coordinates are used to build three gradient fields on the mesh. The local transformations are then applied to the gradient fields to obtain the new configurations after the deformation. Three poisson equations are solved based on the transformed gradient fields to recover the new coordinates of the vertices. Compared to the Poisson approach, Sumner et al. [66] directly applied the desired transformations to the faces of a triangle mesh. Such transformations called deformation gradients are consistent for each triangle facet. Similar to solving Poisson equations, the transformed faces have to be glued together through a reconnect process by forcing their common vertices to coincide. These methods were mainly used for motion transformation. When we want to use them for shape editing where only a few transformations are known in prior, the rest of them have to be estimated by enforcing certain smoothness constrains. Besides using geodesic distances, Zayer et al uses a harmonic guidance field to interpolate the unknown transformations. Since transformation based methods directly change the shape of mesh elements, the geometry change during the deformation can be controlled by constraining the transformations.

2.1.3 Comparison

A comparison of the two types of mesh deformation method is listed in Table 2.1.

Table 2.1: A comparison of displacement and transformation based mesh deformation methods

Method	Pros	Cons
Displacement based	1. Ideal for modeling arbitrary non-smooth shapes by splitting and merging; 2. Local vertex movement is straightforward to define, connectivity is guaranteed.	1. Difficult to control the local shape, ie., angles and areas of the surface; 2. Easy to generate singularities during deformation due to the lack of local shape constrains.
Transformation based	1. Can constrain the local shape in terms of face transformation parameters; 2. Global shape is well defined due to the reconnect procedure.	1. A reconnect procedure is essential for maintaining the connectivity; 2. Some complex shape changes cannot be easily modeled by solely face transformation.

2.2 Deformable Models

Deformable models have been widely used in graphics and image processing for decades. The main idea is to deform a seed shape to fit certain data mainly for the purpose of segmentation and tracking. Much of related research discussed the dynamic behavior and optimization methods during the deformation process, and these techniques can be used for mesh deformation.

Deformable models can be broadly categorized as parametric deformable models and geometric deformable models. Parametric deformable models represent curves and surfaces explicitly in their parametric forms, and their deformation are modeled by lagrangian formulas. This representation allows direct interaction with the model and can lead to a compact representation for fast real-time implementation. Adaptation of the model topology, however, such as splitting or merging parts during the deformation, can be difficult using parametric models. Parametric deformable models are also called active contours or "snakes" in 2D and "balloons" in 3D. Geometric deformable models, on the other hand, can handle topological changes naturally. These models, based on the theory of curve evolution and the level set method, represent curves and surfaces implicitly as a level set of a higher-dimensional scalar function. Their motions are modeled by Eulerian formulas.

The first revolutionary paper was written by Demetri Terzopoulos and et al. [71, 34] The basic idea in deformable models is to evolve a curve subject to constraints from a given image u_0 in order to detect objects in that image. The classical snakes model involves an edge detector, which depends on the gradient of the image u_0, to stop the evolving curve at the boundary of the object.

Let $u_0(x, y)$ map the square $0 \leq x, y \leq 1$ into R, where u_0 is the image and $C(I) : [0, 1] \rightarrow R^2$ is the parameterized curve. The snake model is to minimize

$$\begin{aligned} F_1(C) &= S(C) + P(C) \\ S(C) &= \alpha \int_0^1 |C'(s)|^2 ds + \beta \int_0^1 |C''(s)|^2 ds \\ P(C) &= -\lambda \int_0^1 | \bigtriangledown u_0(C(s))|^2 ds \end{aligned} \tag{2.3}$$

where α, β, λ are positive parameters. The first two terms control the smoothness of the contour, while the third attracts the contour toward the energy. The curve is located at the points of maximum $\bigtriangledown|u_0|$, which act as an edge detector, while keeping the curve smooth.

An edge detector can be defined by a positive decreasing function $g(\overrightarrow{z})$, depending on the gradient of the image u_0, such that $\lim_{|\overrightarrow{z}|\to\infty} g(\overrightarrow{z}) = 0$

A typical example is

$$g(\bigtriangledown u_0(\overrightarrow{x})) = \frac{1}{1 + |J * \bigtriangledown u_0|^p} \tag{2.4}$$

for $p \geq 1$ where J is a Gaussian of variance σ.

Rather than using the energy defined in (2.4), we can define a compact version as in [9, 36] via

$$F_2(C) = \int_0^1 g(\bigtriangledown(u_0(s)))ds \tag{2.5}$$

The application of deformable model in shape recovery is extensively studied during the last two decades. Typically, users initialize a deformable model near the object of interest and allow it to deform into place. Users can then use the interactive capabilities of these models and fine-tune them later. Furthermore, once the user is satisfied with the result on an initial image slice, the fitted contour model may then be used as the initial boundary approximation for neighboring slices. These models are then deformed into place and propagated again until all slices have been processed.

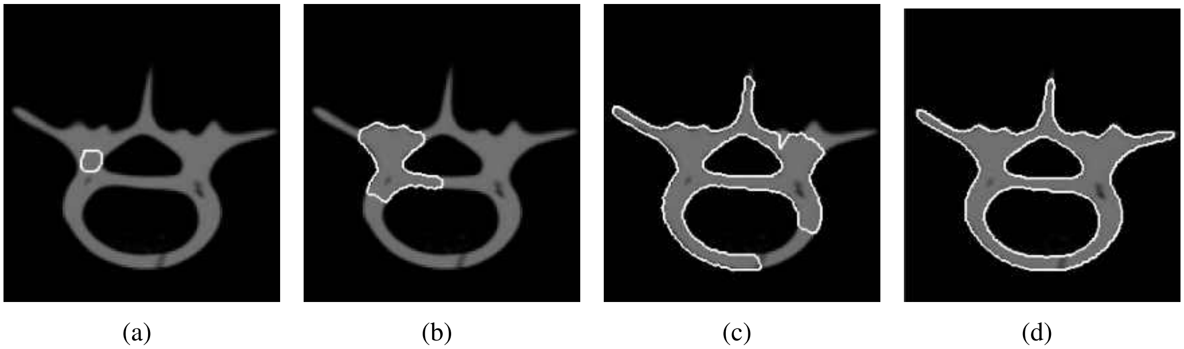


Figure 2.1: (a) (b) (c) (d) show the convergence of a deformable surface to recover a shape, reprinted from [51]

The typical approach is described as follows. Let I be the object domain of dimension n. A level set function φ of dimension $m > n$ is embedded into I. Also a velocity field will be defined to guide the motion of φ. Since the projection of φ onto I will deform as φ changes across time, the zero level set is supposed to rest on the object's boundary. An alternative algorithm is to fix φ and change the level set values across time, and finally the k level set will be the solution. Explicit boundary representation can be obtained using algorithms such as Marching Cubes [46].

The convergent process of the deformable model is illustrated in Figure 2.1.

As applied in shape extraction deformable models do not inherently consider the type of shapes to be extracted. However, as pointed out by McInerney and et al [50]:" A deformable model should be able to easily incorporate added constraints and any other a priori anatomic knowledge of object shape and motion." Many studies aimed at providing the deformable model with prior-shape information which will guide the system's convergence. In [54], a variation called generalized cylinder was devised to extract tube shaped structures in the image. This study added the shape constrain directly on the dynamic behavior of the model so that certain results could be expected. However, it also limited the application of the algorithm to extract tube-like shapes. In [39], a training procedure was added to active contour models to cope with objects with specific shapes. In [81] a snake was designed to optimally separate the values of certain image statistics over a known number of region types. In [26], an active contour model was used for heart ventricle recovery, but with its deformations constrained according to prior knowledge of ventricular shape. When arbitrary shapes are considered, the prior-knowledge has to be provided in a more direct way. In [12, 56], the authors added shape similarity terms into the target functions. Besides threshold parameters, a shape prior given by another level set function φ_M and a shape distortion tolerance factor must be provided. This was done by coding shape into the distance function.

To obtain highly accurate vessel borders for fluid flow simulations, two different deformable models were combined to handle topological changes while a certain degree of interaction was retained to make further refinement [13]. Similar work can be found in [80]. These approaches showed us how to integrate different deformable models when there was a trade off between topology control and geometric flexibility. In the latter paper boundaries of structures were considered as minimal paths, and prior shape knowledge was incorporated into the shape recovery process to achieve more robust and accurate results. The shape priors were implicitly represented and the estimated shapes of the structures could be conveniently obtained. In order to increase speed, the narrow band approach can be adopted. In [9], the authors made the level set rest on locus of high gradient while looking for a solution with minimum surface area. As the algorithm was conducted on 3D image data, a fast march algorithm was used to achieve better efficiency. A fast narrow band algorithm was proposed by Shuntaro and et al. [83] to further make the complexity tractable in case of high dimensions.

Numerical problems are always the main concerns in real world applications. They have to be handled properly in implementations. In [68] a fuzzy model was integrated into the level set framework to prevent leakage caused by weak stopping or clamping forces in noisy images which were common in medical imaging applications. The forces in the level-set approach used four kinds of speed control functions based on shape, region, edge and curvature. Regional and shape speed functions were determined based on the fuzzy membership function computed using the fuzzy clustering technique, while the edge and curvature speed functions were based on gradient and signed distance transform functions, respectively. Similar approaches can also be found in [42]. In order to cope with ambiguous boundaries Kadir and et al. [31] proposed an algorithm which added a region competition process to the original level set method. Adjacent regions generate internal forces to compete for the boundaries. The original idea was borrowed from an statistic model but was implemented using the level set method. Another powerful application used Mumford-Shah model. The original model was proposed by Tony Chan and et al. [10, 11] in the form of an active contour model. Since it was a region based algorithm, a level set approach was used to solve the dynamic system involved. This framework was further studied by other researchers [24, 76] and achieved success in preventing leakage and in situations of high noises. In [72] a training procedure was added to account for the variations within a shape category. Some systematic work with careful implementations can be found in [79, 53].On the other hand, for explicit(parametric) representations, the topological changes of evolving surface must always be handled properly with additional efforts. Besides the reinitialization proposed in [49, 51].

The main difference between traditional deformable models and the deformable mesh model is that latter uses an explicit mesh representation for the surface/volume. Thus, the deformable mesh model inherits all the benefits from parametric models while is able to model any non-smooth shapes. It is also more efficient than geometric models. The following chapters will show how some of the above techniques can be used in mesh deformations with novel insights.

CHAPTER 3

PDE DEFORMABLE MODEL AND ITS APPLICATION IN MEDICAL IMAGING

The recovery of objects from images is a fundamental task in image processing. It is also critical in many graphical visualization systems since the successful rendering of multiple objects usually depends on the existence of their geometric information. This is especially important when we want to highlight a subset of the objects buried in others. Many of these applications can be found in medical imaging where the objects have to be recovered from images. Medical instances such as human organs are often of arbitrary shapes while their topology is usually simple compared to the geometry. Thus, it is often favorable to use parametric deformable models to carry out the segmentation task. This chapter describes a novel PDE deformable model which has a user-controlled topology and is directly in the mesh representation. The algorithm is successfully applied to build a brain tumor visualization system.

3.1 Introduction

The following work is motivated by the recent development of Susceptibility Weighted Imaging (SWI) and Magnetic Resonance Angiography (MRA), while the former highlights the veins in the body and the latter gives a clearer picture of the arteries. Imaging plays a critical role in the diagnosis and treatment planning of brain tumors and also serves as a means for evaluation during or after therapy. Tumors are well visualized thanks to the uptake of the contrast agent into the tumor, but it provides little differentiation of structure within the tumor. By combining instances from T1-weighed MRI and SWI we can extract the information of both the tumor and the corresponding vessels. First we acquire the T1-weighted MRIs taken before and after contrast agent was injected. After registration we can get the difference image of the two, in which the brain tumor is much more highlighted. Using some statistic analysis we are able to fine the approximate location of the tumor in the data. Followed by a PDE dynamic surface flow algorithm we can get the surface mesh of the tumor. With this mesh aa seed and the registered SWI in hand, we then track the vessels that are coming in and out of the tumor. Another tracking algorithm along the vessels is then carried out to find the information of the blood products. After all segmentation tasks have been done, a blended rendering can be realized.

The following contents are organized as follows: Chapter 3.2 shows the previous work of related techniques in use; Chapter 3.3 introduces the basic procedure of the proposed approach; Chapter 3.4 describes the partial differential equation we use to extract the surface of the brain tumor; Chapter 3.5 discusses an automatic pipeline and implementation details; Chapter 3.6 gives the results of the proposed approach in neuroimaging; Chapter 3.7 summarizes this chapter.

3.2 Related Work

Extracting brain tumors and the related structures from 3D volume data is difficult due to the sheer size of the datasets, the complexity and variability of the tumors, and the spatial variability of signal response especially in high fields [35, 64]. In addition, the shortcomings typical of medical data (defect data), such as sampling artifacts, spatial aliasing, partial volume effect, and noise, may cause the boundaries of brain tumors to be indistinct [64, 30].

Many different segmentation algorithms have been devised/applied over the years to make the process more accurate, efficient and robust. Some of the recent representative work will be discussed here. Compared to deformable models, intensity based methods are becoming popular mainly due to accuracy considerations. Statistical models are devised to conquer intensity ambiguities and noise [16, 77]. As a comparison, our method aims at extracting the region and shape of the tumor system in an efficient and robust way. In [37], Kuhnigk et al. devised a morphological algorithm to deal with the partial volume effect, which is considered to be a main cause of inaccurate segmentation. It will affect the segmentation by producing ambiguous boundaries. We use heuristics and geometric interpolation to solve this problem. Fatakdawala et al. [20] incorporated expectationmaximization method into geodesic active contour model to segment overlapping objects. Similarly, our PDE driven deformable model uses different evolution strategies to segment different objects with different shapes. Most recently, Ahmed et al. [2] published their segmentation results based on feature fusion from multiple image modalities. Our framework also employs different modalities include T1 weighted MRI and Susceptibility Weighted Image (SWI) but only with the purpose of segmenting different instances in the tumor system. However, we believe the method in [2] can also be incorporated into our framework to better guide the deformation of the mesh. It is also worth mention that traditional parametric deformable models are not very suitable for tumor segmentation due to the representation limitation of parametric surfaces. Our model can be seen as a variation of deformable model with mesh representation, which is ideal to model irregular geometry such as tumors.

The most related work was proposed in [17]. Delingette devised a deformable model based on simplex meshes and applied it to object reconstruction. Our model differs from [17] in two aspects. First, our model uses a triangle mesh representation, which is more versatile and well studied. Thus, many mature methods can be adopted to maintain its quality during the deformation. Second, we use PDE to guide the mesh deformation rather than Newtonian laws so that it is more efficient. Giving the segmentation task is more challenging than surface reconstruction in terms of shape complexity and efficiency, we believe that our model is more suitable.

3.3 Framework Overview

Our isosurface extraction algorithm is designed to handle general tasks while retaining the adaptivity among different applications. Our experiment is carried out to deal with brain tumor system which includes tumor, vessel, blood product and etc. The scheme consists of a series of surface extraction and rendering tasks. The research work is also motivated by the recent development of Susceptibility Weighted Imaging (SWI) and Magnetic Resonance Angiography (MRA), while the former highlights the veins in the body and the latter gives a clearer picture of the arteries. Imaging plays a critical role in the diagnosis and treatment planning of brain tumors and also serves as a means for evaluation during or after therapy. Tumors can be detected thanks to the uptake of the contrast agent into the tumor, but it provides little differentiation of structure within the tumor. By using modalities such as T1-weighed MRI and SWI we can easily obtain the information of both the tumor and the corresponding vessels. First we acquire the volumes from T1-weighted MRIs taken before and after contrast agent was injected. After registering the two we get the difference volume, in which the brain tumor is much more highlighted. Using some statistic analysis we are able to approximately locate the tumor in the volume. Followed by a PDE dynamic surface flow algorithm the surface mesh of the tumor is obtained. By using the tumor mesh as a seed and the registered volume from SWI , the vessels that are coming in and out of the tumor can be tracked. Another tracking algorithm is then carried out to extract the blood products along the vessels. After all segmentation tasks are done, a composite rendering can be realized displaying all the related visual components.

3.4 PDE Deformable Model

In order to build an robust system which can segment objects of different shapes without much user intervention, we present an adaptive PDE-based deformable model in this section. The model uses a mesh to represent the surface of the object. By carefully designing the dynamic behaviour of the mesh, it is feasible to make its evolution efficient and robust while keeping model versatile. The following subsections describe two types of explicit mesh surface flow which can be used in brain tumor surface extraction as well as surface extraction of the feeding blood vessels. Much like sibling works, in our approach, a triangle mesh with simple geometry is placed into the original data domain as a seed, which is driven by the surface flow to evolve toward the object's boundary.

3.4.1 Normal Surface Flow

The normal surface flow is driven by the gradient force of the image data. The seed model needs to be initially placed either inside or outside the object or partially overlapping the object, and then the model will grow or shrink until it reaches the boundary of the object. Hence, the recovered object has to be a closed manifold so that the deformable model can stop by itself when it reaches the boundary.

Mathematical concepts The normal surface flow is depicted as follows: between two adjacent time steps, every vertex of the mesh moves in the direction of its normal. The mathematical representation is

$$\frac{\partial s}{\partial t} = (g(s)(v + H) - \nabla g(s) \cdot \overrightarrow{n})\overrightarrow{n} \tag{3.1}$$

$$s(0) = s_0 \tag{3.2}$$

$$g(s) = \frac{1}{1 + |\nabla(G_\sigma * I(s))|^2} \tag{3.3}$$

Where $s = s(t)$ is the 3D deformable surface, t is the time variable, and s_0 is the initial shape of the surface. Note that H is the mean curvature of the surface, $\overrightarrow{n}$ is the unit normal of the surface, and v is the constant velocity that will enable the convex initial shape to capture non-convex, arbitrary complicated shapes. The non-zero velocity term is useful to avoid the model getting stuck into the local minimum during the evolution process. Also, g is a monotonic, non-increasing, non-negative function that enables the model to interact with the image and will stop the model deformation when it reaches the object's boundary, which has great gradients. I is the volumetric density function, and G_σ is the smoothed density function by convolution with a Gaussian filter of variance σ.

Implementation Details There are a number of issues that we need to consider in the implementation of normal surface flow on a mesh. In order to obtain good mesh quality, mesh optimization is carried out when necessary. To achieve good distribution of the triangles in the mesh, edge splitting and edge swapping are needed. These are illustrated in Figure 3.1 and Figure 3.2. When an edge is too long we divide it into two halves and when two adjacent triangles ”squeeze” together we swap their common edge to form two new triangles. At certain steps in the surface evolution process the mesh might form a very irregular shape. For example, sharp angles of some mesh triangles will appear at some local areas. These are artifacts which can keep the evolution from convergence. Therefore, after a certain number of iterations, a Gaussian smooth algorithm is applied to “release” some vertices of the mesh.

Due to the nature of most tumors, the topology of PDE surface model is unlikely to change during the mesh evolution process. However, there are still some local determinations the algorithm needs to take care of. The first issue is regarding the moving direction of each vertex at each step: whether a vertex should move in the direction of its normal or negative direction depends on the current properties of the vertex. The data used in this chapter highlights the tumor by using contrast agents, which makes it easy to find a threshold. Chapter 3.5.2 depicts a way to find a good threshold. To avoid the instability of using a single threshold calculated globally, we update it by using the average intensity at the mesh vertices inside the tumor during the evolution. If the intensity at the current location is higher then this threshold, the vertex has a higher possibility to be inside the tumor, it then moves in its normal direction, otherwise it should move in the negative direction of its normal.

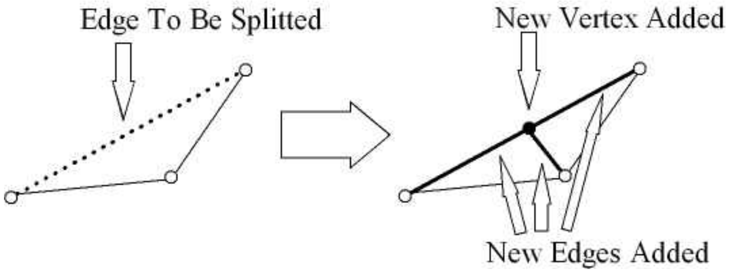


Figure 3.1: Edge splitting. The edge in dashed line on the left exceeds the length limit and is splitted, a new vertex (filled) and three new edges (in bold) are created to form two new faces.

Before updating a vertex, the algorithm needs to check if its intended move will violate the

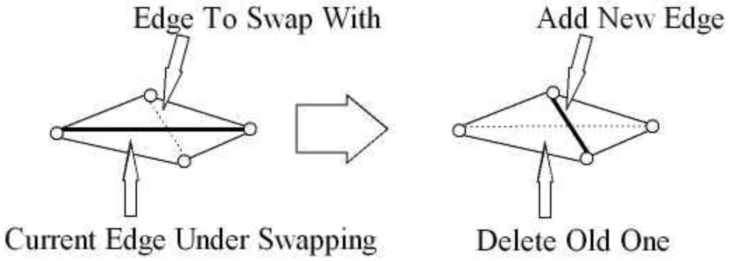


Figure 3.2: Edge swapping. The edge in bold on the left is relatively too long for both adjacent faces and is removed, the bolded edge on the right is added to form two new faces.

current topology of the mesh, i.e. to check if self-collision is at risk. At this stage, self-collision detection is carried out. If there is possible collision the algorithm will have to make a choice based on the topology verification, either collapsing this vertex if not changing topology or freezing it temporarily until its action will not cause topology violation any more. A collapsing is illustrated in Figure 3.3. This kind of collapsing is usually reasonable since it will not affect the following procedures as new vertices will come out at this location if necessary. If at this time the algorithm marks this vertex as frozen, then it will be processed some time later. After a certain number of iterations, all the vertices are either halting at the boundary of the tumor or are frozen due to possible self-collision. The frozen vertices are usually at locations where tiny features occur. In order to differentiate these tiny features the algorithm will adaptively increase the mesh resolution in the corresponding areas by splitting and necessary collapsing. Sometimes a vertex may oscillate around a position as shown in Figure 3.4. This indicates a potential boundary where the vertex speed may not be diminished at either side. In order to increase performance an aging test is implemented in the algorithm. If a certain number of consecutive oscillations is detected, the algorithm forces the corresponding vertex to stop on the point where the smallest speed can be achieved along its path between the current position and last position (see Figure 3.5).

3.4.2 Tangential Surface Flow

For segmentation purposes, the normal flow is universal but not optimal. According to Ye2001[19] a new 2.5D active contour based on tangential flow is suitable for extracting both

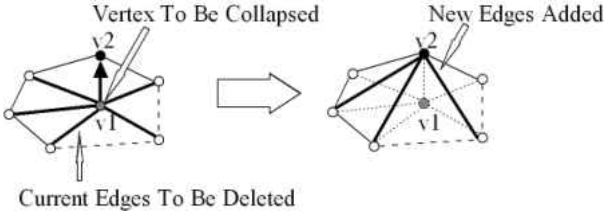


Figure 3.3: Vertex collapse. Due to minimum edge criterion or collision, vertex v_1 is collapsed into v_2, all the associated edges are deleted accordingly and results in the creation of three new edges (in bold) and the corresponding faces on the right.

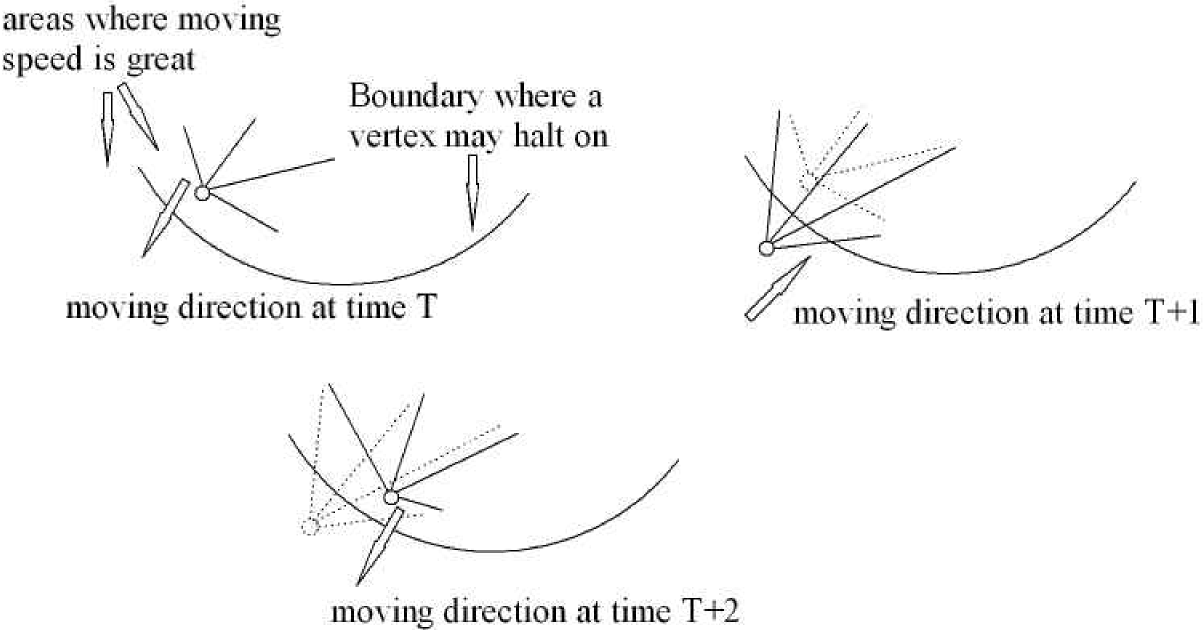

areas where moving speed is great
Boundary where a vertex may halt on
moving direction at time T
moving direction at time T+1
moving direction at time T+2

Figure 3.4: Oscillation. The vertex is oscillating about a boundary due to the direction ambiguity around the boundary.

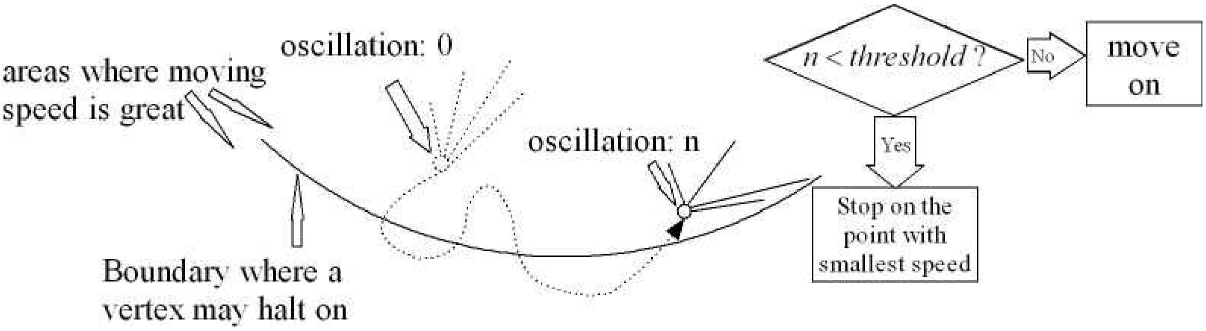

areas where moving speed is great
oscillation: 0
oscillation: n
Boundary where a vertex may halt on
$n < threshold$?
No
move on
Yes
Stop on the point with smallest speed

Figure 3.5: Aging, the dashed curve indicates the oscillation of a vertex, and the formula on the right gives the stop criterion.

manifold and non-manifold structures, which is ideal for segmentation of tube-shaped objects such as blood vessels. In order to achieve faster speed and higher mesh quality we made some modifications to its numerical implementation.

Mathematical Concepts Let us first consider a family of smooth planar curves $C(p, t)$ which evolve according to the evolution equation:

$$\frac{\partial C(p,t)}{\partial t} = Fu(p,t), \tag{3.4}$$

$$C(p,0) = C^0(p) \tag{3.5}$$

where $u(p, t)$ is the unit directional vector for $C(p, t)$, F is the speed function. The family parameter t can be considered as the time duration of the evolution, and p parameterizes the curve. For our tangential flow, the active boundary vertices will evolve along the tangential direction $\overrightarrow{n}$ during each iteration while its speed is determined by the boundary curvature k. The corresponding equation is:

$$\Delta C = \alpha k \overrightarrow{n} \tag{3.6}$$

where α is a constant.

3.4.3 Implementation Details

Implementing the tangential flow on a mesh involves two issues. One is the numerical computation; the other is the topology maintenance.

1. Numerical Calculation

   The tangential direction is decided locally by the current vertex and its two neighbors on the boundary, which is depicted in Figure 3.7. The tangential normal vector $\overrightarrow{n}$ is calculated as

$$\overrightarrow{n} = \frac{\overrightarrow{n_l} + \overrightarrow{n_r}}{\|\overrightarrow{n_l} + \overrightarrow{n_r}\|} \tag{3.7}$$

$\overrightarrow{n_l}$ and $\overrightarrow{n_r}$ are unit vectors perpendicular to VV_l and VV_r. In the tangential direction of an active boundary vertex V, a new vertex V_n' is located. In order to make the mesh attach to the vessel boundary V_n' is further pulled to the nearest grid point with a prominent gradient. After this, a new vertex V_n along with two new faces V_nVV_l and V_nVV_r are added to the mesh. Figure 3.6 depicts this procedure. Furthermore, when adding a certain vertex will cause topology inconsistency, this action will be abandoned and the corresponding boundary vertex will be marked inactive (ex. collision happens). The local curvature is defined as:

$$\overrightarrow{K} = k\overrightarrow{n} \tag{3.8}$$

as shown in Figure 3.7 The scalar of the curvature is calculated as:

$$k = \frac{4 * S_{\triangle V_l V V_r}}{\|VV_l\| * \|V_lV_r\| * \|V_rV\|} \tag{3.9}$$

where $S_{\triangle V_l V V_r}$is the area of the triangle determined by V_l, V and V_r.

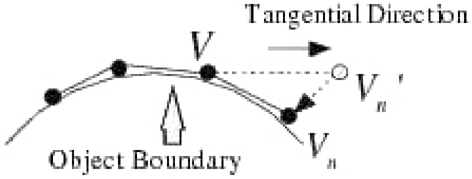


Figure 3.6: New vertex generation, the new vertex is not in the tangential direction but pulled back to the objects boundary

The evolution ends when no active boundary vertex can be found any more or a user-set threshold is met. In our application, the tangential flow is used to extract blood vessels; since the number of vertices is proportional to the length of the vessel, the maximum number of vertices is set to stop the evolution.

2. Topology Maintenance

Topology maintenance is carried out to ensure that the final mesh will best represent

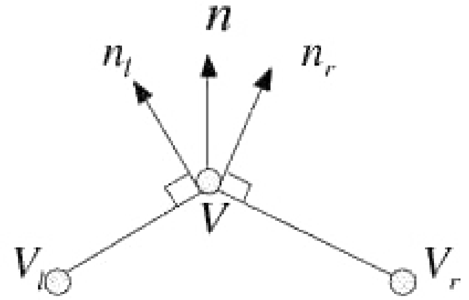


Figure 3.7: Boundary tangential direction

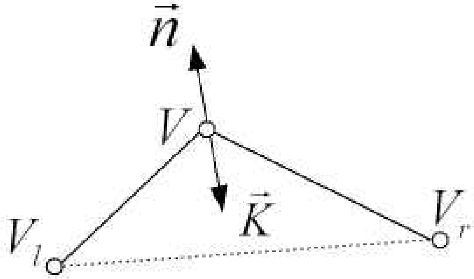


Figure 3.8: Boundary curvature

the geometry of the object. In the case of vessel extraction, the final mesh is a closed manifold of genus one. However due to noise and low space resolution, sometimes it is hard to get the ideal result. Instead we require the resulting mesh to be a manifold or non-manifold with no holes. Two kinds of stitching are used to best achieve this goal. On-site stitching occurs when there is an active boundary vertex that is adjacent to two chains of inactive boundary vertices. The movement of the active boundary vertex and the aforementioned mesh optimization operations will iteratively generate triangles with three adjacent vertices until active boundary vertices are encountered or topology inconsistency occurs. On-site stitching is shown in Figure 3.9.

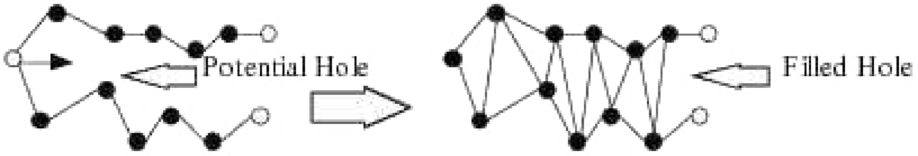


Figure 3.9: On-site stitching

Post stitching will be carried out some inactive boundary vertices are found after the simulation stops. These vertices will form holes in the mesh. Much like the On-site stitching, we patch these holes by iteratively adding faces defined by three adjacent boundary vertices. Post stitching is shown in Figure 3.10.

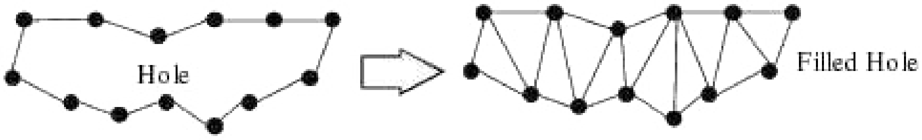


Figure 3.10: Post stitching

3.4.4 Efficiency and Robustness Improvement

The regular deformation of the mesh driven by the PDE is very simple and efficient. It is also very robust for well sampled data; However, experiments show that defect data will increase the occurrence of oscillation and affect the convergence of the evolution, especially for the normal flow. In the general case, some of the vertices will convergent very quickly while others need more steps. Since vertices are updated one by one, an arbitrary updating order will cause the evolution to be unstable, e.x., an oscillating vertex may prevent neighboring vertices from converging. Another issue is regarding when to launch the tangential flow to achieve the best efficiency. The following methods are used to gain improvement.

Geometric Interpolation

When a certain number of vertices (fixed vertices) rest on the object's boundary with high confidence , instead of updating other vertices according to the PDE, we interpolate their positions in a geometric least-squares sense. The following algorithm is inspired by [65] with modifications. In a mesh with n vertices, let the fixed vertices be:

$$v_s = (x_s, y_s, z_s), s \in C \tag{3.10}$$

where $C = \{s_1, s_2, \cdots, s_m\}$. The linear system to solve for the x component of all the vertices is:

$$\mathbf{Ax} = \mathbf{b} \tag{3.11}$$

where $\mathbf{A}$ is a $(n+m) \times n$ matrix of the form:

$$\mathbf{A} = \begin{pmatrix} \mathbf{L} \\ \\ \mathbf{F} \end{pmatrix}, F_{ij} = \{ \begin{array}{cc} 1 & j = s_i \in C \\ \\ 0 & otherwise \end{array} \tag{3.12}$$

$$b_k = \{ \begin{array}{cc} 0 & k \leq n \\ \\ x_{s_{k-n}} & n < k \leq n+m \end{array} \tag{3.13}$$

and $\mathbf{L}$ is the $n \times n$ graph Laplacian matrix defined as:

$$\mathbf{L}_{ij} = \{ \begin{array}{cc} 1 & i = j \\ \\ -\frac{1}{d_i} & (i,j) \in E \\ \\ 0 & otherwise \end{array} \tag{3.14}$$

where d_i is the valence of vertex i and E is the edge set of the mesh.

Solving 3.11 is equivalent to minimizing the following:

$$\parallel \mathbf{Ax} - \mathbf{b} \parallel^2 = \parallel \mathbf{Lx} \parallel^2 + \sum_{s \in C} \|x_s - \mathbf{v}_s^{(x)}\|^2 \tag{3.15}$$

a modification would be:

$$\parallel \mathbf{Ax} - \mathbf{b} \parallel^2 = \parallel \mathbf{Lx} \parallel^2 + \sum_{s \in C} \omega_s^2 \|x_s - \mathbf{v}_s^{(x)}\|^2 \tag{3.16}$$

where $\omega_s, s \in C$ are the weights denoting the confidence of the fixed vertices. In our experiment, the weights are normalized gradients interpolated at the mesh vertices.

Equation 3.11 can be applied to y and z component of the mesh vertices in the same way. Since, the system is sparse, it can be solved very efficiently. The solution will result in a smooth mesh satisfying the constrains (fixed vertices). This algorithm updates all the vertices at the same time according to global smoothing criteria. It resets non-convergent vertices to stable positions that will produce robust results and good mesh quality.

The geometric interpolation is carried when the ratio of oscillating vertices exceeds a threshold. If some vertices do not converge due to data defection, they are forced to rest on the point with maximum gradient along its normal direction within a range, and they are considered as fixed vertices in the next iteration.

Model Selection

We have presented two types of PDE deform models. Compared to normal flow, the simulation of tangential flow is much faster for tube-like shapes such as blood vessels. However normal flow will guarantee to a closed manifold of genus zero, which gives a better representation of many brain structures. Another good aspect of normal flow in our application is that it does not need a strict initialization. For this reason, an ideal automatic scheme is to use normal flow in the initial phase of surface extraction then followed by a model selection procedure to employ appropriate surface flow for extracting different shapes of objects.

The numerical simulation with model selection can be expressed as follows:

Step1 Initially run the simulation with normal flow on a closed seed mesh for a certain number of steps.

Step2 Detect the moving front by tracking all the active vertices. If the moving front is consisting of clusters of adjacent vertices while the majority of vertices are inactive, it is time to switch the model.

Step3 Break the mesh at the moving front by removing the corresponding vertices to form an open mesh. Mark the vertices on the boundary active.

Step4 Resume the simulation by using tangential flow as described above until it ends up with the final mesh.

The model selection proposed here is a good way to combine two major flows while retaining the privileges of both.

3.5 Automatic Segmentation and Visualization of Brain Tumor System

Based on the PDE deformable model, we have developed an automatic software tool to segment and visualize the brain tumor system. Our main targets will be the tumor, the feeding blood vessels and the microhemorrhage. In this section, we want to address some issues regarding preprocessing, data structure, automatic seeding, and the segmentation of multiple objects in the tumor system.

3.5.1 Preprocessing

In order to combine the information extracted from data of different modalities, co-registration is need first. Second, a series of PDE-based surface flow evolutions are carried out to extract object-of-interests and the corresponding visualization can be performed afterwards.

Accurate localization of tumor is known to be a very difficult task in general. Instead, we only estimate the position of the tumor automatically since our PDE model is not sensitive to initialization. This is achieved by multiple-scale searching. The idea is to do octree-based subdivision of the image volume. It is practical under the assumption that the regions around the tumor should have some properties distinguishable from those of other regions. The algorithm first obtains 4 by 4 by 4 equally spaced octree nodes. If no sign of possible tumors can be found it goes to the next level with 8 by 8 by 8 nodes. To carry out a valid statistical analysis with enough samples, the octree partition goes on until node contains no less than $16 \times 16 \times 5$ voxels for a volume of $256 \times 256 \times 100$. To determine if a certain partition contains tumor tissues, we perform the histogram analysis.

3.5.2 Histogram Analysis for Seeding

Histogram analysis tries to identify the tumor by searching for distinctive intensity distributions among the octree nodes. In order to highlight the tumor, the intensity value of T1 weighed MRI is subtracted from that with contrast agent injected to form a difference image data. In the difference image, experiments show that the intensity distribution function in the area around the tumor tends to have two peaks. The comparison between histograms of partitions with and without tumor tissues is shown in Figure 3.11(e) and (f). A data fitting algorithm is used to automatically detect this bi-normal distribution in the intensity histograms. The partition and histogram analysis process is shown in Figure 3.11. The false-alarmed partitioning nodes at this stage can be eliminated by the later PDE surface evolution and shape analysis. When neighboring cells are identified to contain the tumor tissues, they are merged.

Besides, the histogram analysis also provides a threshold of the tumor. The intensity of the second peak in the histogram is used as the initial threshold. The thresholds for the blood vessels and microhemorrhage is pre-defined by the domain experts.

3.5.3 Segmentation and Visualization Pipeline

After the location of the tumor is identified, a seed mesh is initialized into this region in the form of a sphere whose radius is determined by the size of the cell. A normal flow it carried out to drive the seed to rest on the boundary of the tumor. Even if the initial seed mesh does not lie inside or contain the tumor, we can still obtain a robust result due to the fine numerical solution of our PDE based algorithm stated above. When the tumor is extracted, we seed several meshes around it according to the intensity of the corresponding SWI data. They will then evolve based on the model selection procedure described in 3.4.4 to ensure the best performance. Though the microhemorrhage has a different intensity in SWI, they are hard to segment as the lesion usually contains many clusters.

Using a voxel-based tracking approach, it is easy to segment the major cluster of the microhemorrhage. The algorithm works as follows: check for voxels with intensities below a certain

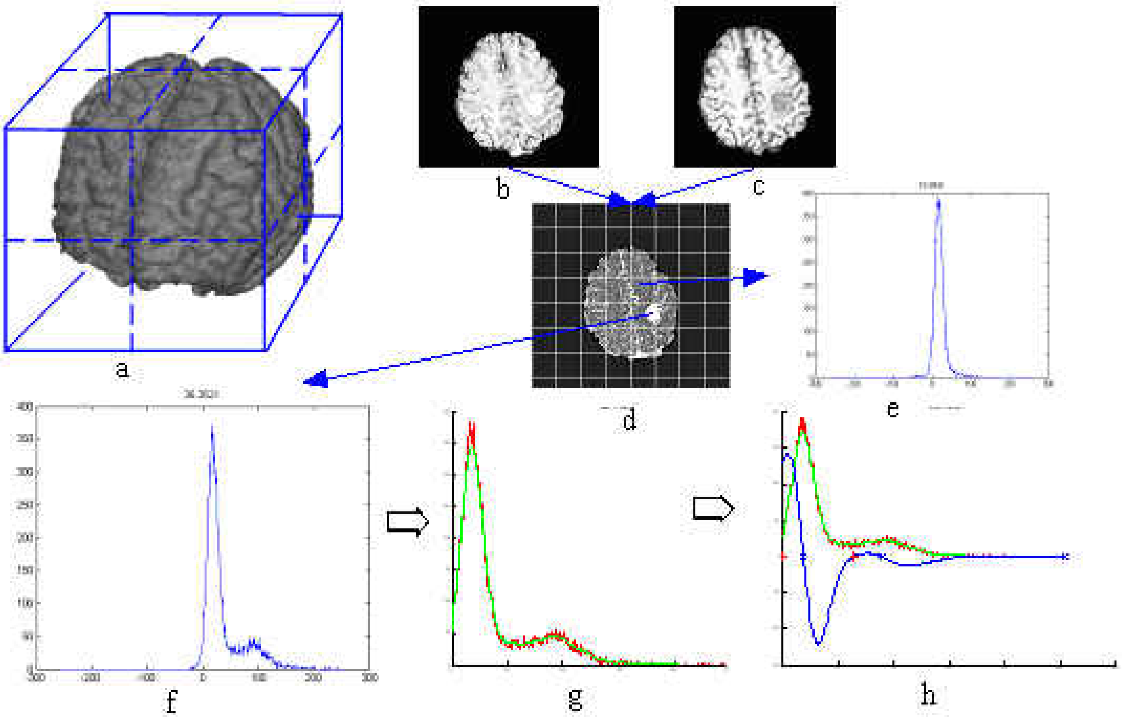


Figure 3.11: Octree-partition and histogram analysis. (a) depicts an 2 by 2 by 2 octree-based subdivision; (b), (c) and (d) are MRI with and without contrast agents and the difference volume respectively; (e) and (f) are histograms of a normal partitioning cube and one containing tumor tissues respectively; analysis of (g) and (h) can give us many information about the tumor and the surrounding tissues.

threshold around the vessel and tumor, push them in a queue and check their neighborhood voxels with the same criteria until no more voxels can be found. Then a volume with only microhemorrhage can be obtained. The major cluster is then enlarged by including neighboring voxels with similar intensities. The result is good enough for visualization purposes. In our current approach, the microhemorrhage is visualized using volume rendering offered by the Visualization ToolKit (VTK) [1].

The pipeline of our system is shown in Figure 3.12.

3.6 Experiments

This section will show some results from our segmentation and visualization tool. The prototype system is implemented in Pentium 4 PC with 3.0GHz CPU and 2GB memory. The

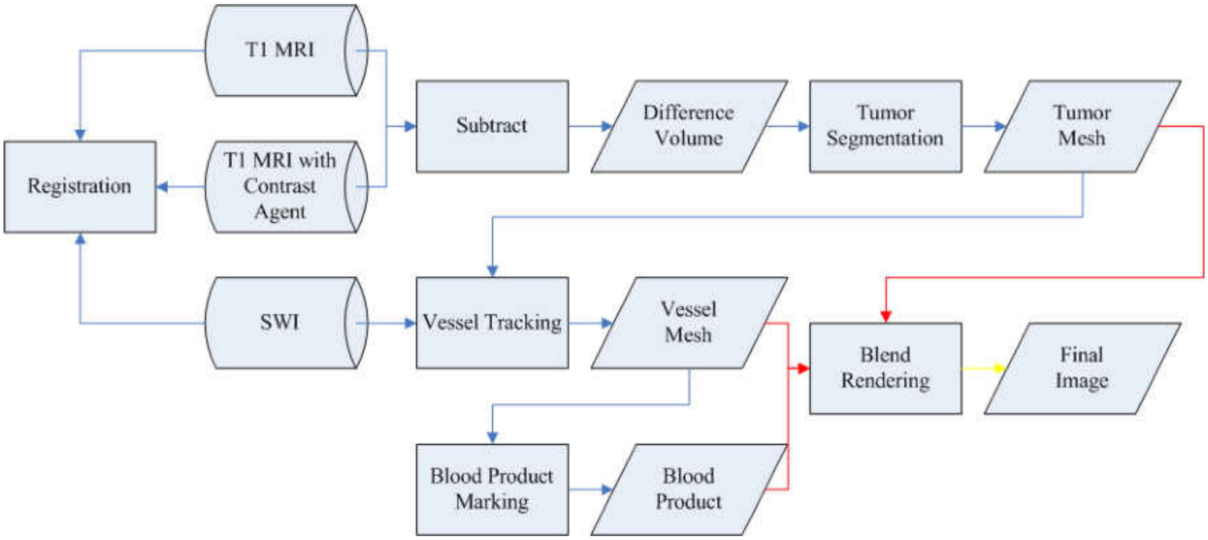


Figure 3.12: Segmentation and visualization pipeline

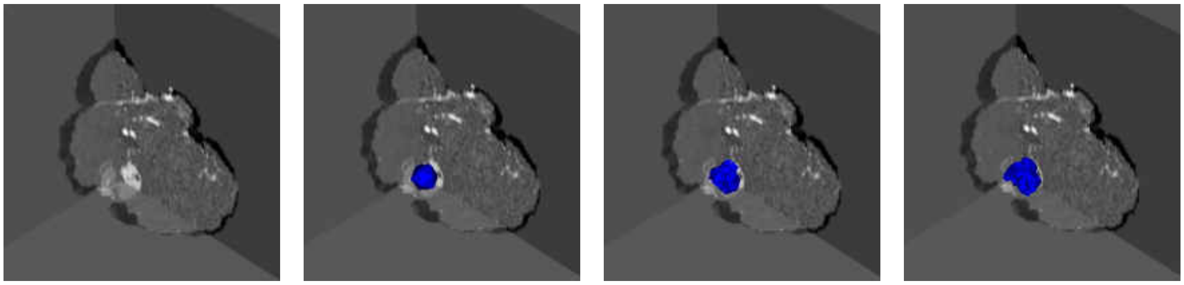
visualization tool is implemented with OpenGL and VTK. Figure 3.13 shows how the seed grows to reach the boundary of the tumor. By a standard triangle test, it is easy to extract the region in side the tumor. We tested our algorithm on 18 patient data sets and compared the results with those from manual segmentation by domain experts. Figure 3.14 shows one case where the segmentation result using the automatic tool is compared with that from a doctor. Table 3.1 shows the tumor segmentation results for all 18 data sets. The accuracy is calculated by $\frac{\|C_{al} \cap C_{gd}\|}{\|C_{al} \cup C_{gd}\|}$, where C_{al} is the voxel set segmented by the algorithm, and C_{gd} is the voxel set segmented by domain experts. The results have very promising clinical application according to domain experts.

The vessel tracking process is shown in Figure 3.15.

Based on the segmentation results, the visualization tool creates an illustrative image by overlaying the surface rendering of the tumor/blood vessels and the volume rendering of the mircohemorrhage. The results are shown in Figure 3.16. All the users in the radiology department gave highly positive feedback on the composed rendering which improves the tumor diagnosis greatly. Quantitative evaluation of the visualization system is being conducted.

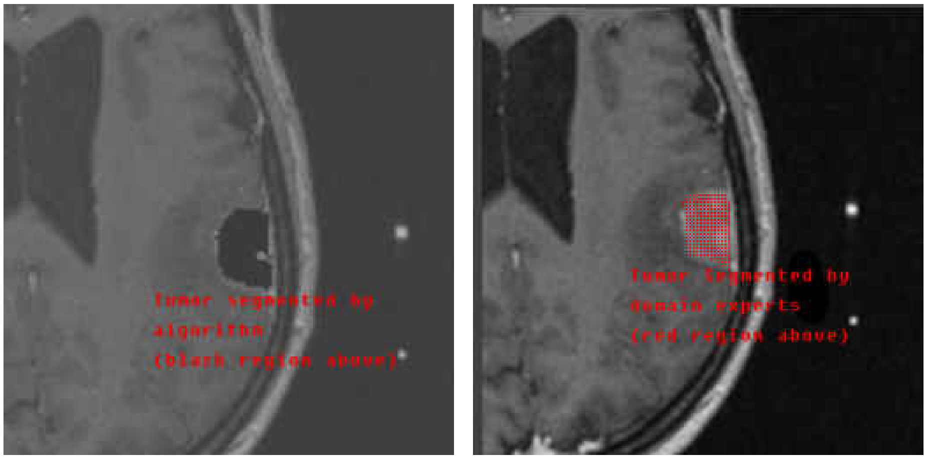
Table 3.1: Tumor segmentation results: the accuracy is calculated by $\frac{\|C_{al} \cap C_{gd}\|}{\|C_{al} \cup C_{gd}\|}$, where C_{al} is the voxel set segmented by the algorithm, and C_{gd} is the voxel set segmented by domain experts.

Data set	1	2	3	4	5	6	7	8	9
# of tumors	1	1	2	1	1	1	2	1	2
Accuracy	0.87	0.73	0.82	0.81	0.91	0.88	0.82	0.79	0.82
Data set	10	11	12	13	14	15	16	17	18
# of tumors	1	2	1	1	1	2	1	2	1
Accuracy	0.86	0.85	0.84	0.90	0.76	0.87	0.82	0.89	0.83



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

Figure 3.13: Visualization of the tumor segmentation process. The bright region in the middle of the cross-sections indicates where the tumor occurs, (a) through (d) illustrates the formulation of the tumor mesh form the initial seed (an Icosahedron).



(a) (b)

Figure 3.14: The dark region in the left image shows the tumor segmented using our automatic tool; the red mask in the right image shows the ground truth marked by a doctor.

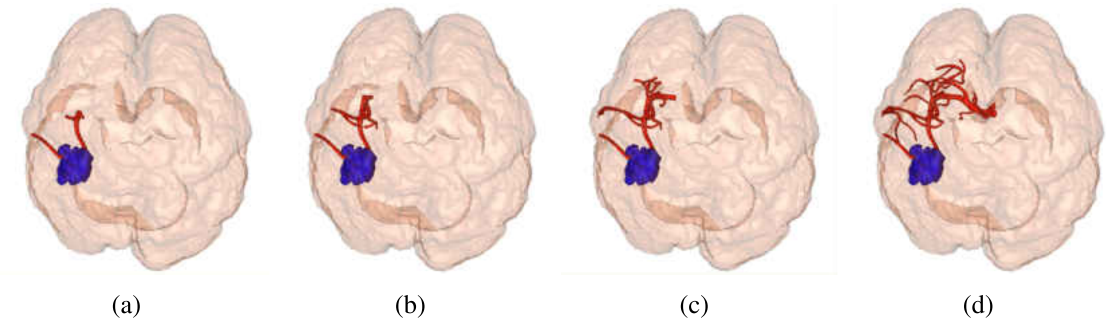


Figure 3.15: Visualization of the vessel tracking process. The process starts from the tumor and simultaneously extends to find all the vessels attached to it. Depending on the vessel length needed, we may obtain different results shown from (a) to (d).

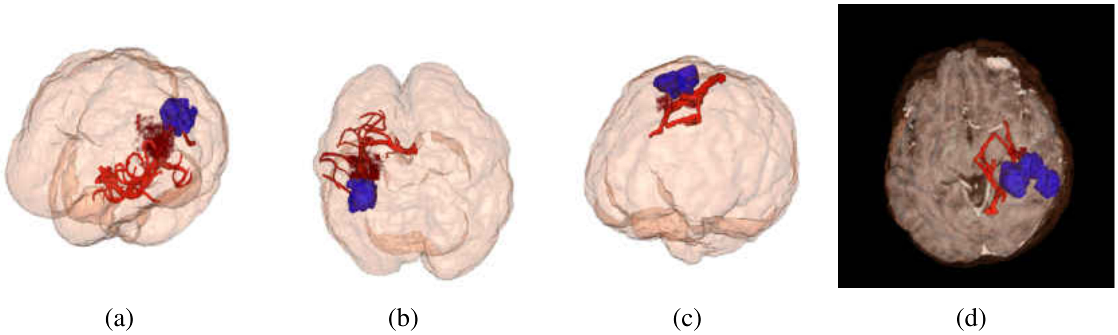


Figure 3.16: Visualization of the tumor system. (a) and (b) are different views of a brain tumor system. (c) and (d) show another example. (d) also shows a slice of the original data

3.7 Summary

In this chapter, a novel iso surface extraction method based on PDE surface flows is presented. The technique is displacement based and is very effective in recovery arbitrary non-smooth shapes from images. Based on this technique, a new visualization framework is proposed to facilitate neuroimaging. The application in brain tumor visualization demonstrates the effectiveness of our framework.

CHAPTER 4

HIERARCHICAL SURFACE ABSTRACTION FOR DEFORMABLE MESHES

The PDE model presented in Chapter 3 is very versatile for arbitrary non-smooth shape recovery and can be “blind” of the shape itself. However, when more information is known about the object, it does not provide the optimal solution. For example, when the initial seed is provided as a similar shape to the object being recovered, the PDE model is not able to utilize this information, therefore, it will not be efficient. This can be the case in shape retrieval and matching. The rapid development of 3D object digitalization equipments and ever increasing demand of shape animation applications have driven the pertinent shape retrieval and matching to become a very active research field in computer graphics. Giving a template shape, the main task of shape retrieval and matching is to recover the similar/partially similar shapes from a bank of shapes. When the shapes are represented by meshes, similar shapes can be modelled by deformable meshes. Consider the large shape bank, the algorithm in Chapter 3 will be slow and hard to measure. In this chapter, a novel framework is proposed to simplify the meshes so that the deformation between them can be efficiently carried out by comparisons of graphs. To differentiate the work from classical mesh simplification, it is referred to as surface abstraction since it abstracts surface patches into graph nodes.

4.1 Introduction

Shape matching becomes increasingly popular in many animation applications in the past few years. For example, it was used in motion synthesis [5] and motion capture [75]. Tangelder et al. classified all existing 3D shape retrieval methods into several categories [70]. It was shown that even though global feature-based methods are more efficient, only algorithms based on local features can support accurate detailed matching. The prior work again raises the question of how to effectively integrate local and global features in a single framework. The solution to this question will certainly lead to a better understanding of a given shape. To facilitate most shape comparison tasks, global appearance, local variance and topological features of shapes should be considered together in a systematic way. Taking efficiency into consideration, redundancies should be hidden from execution. Therefore, an automatic abstraction algorithm is a very appealing solution for shape description, which can retain both global and local shape characteristics.

In most cases, a complex shape is composed of relatively simple components. Recent human perception study proved that visual content is recognized by components or parts [55, 74]. We observed that, on one hand, the global feature of a shape can be treated as the composition of the features from its components, on the other hand, partial matching is usually documented as the task of identifying non-trivial, similar components of more complex shapes. These components either have similar semantical meaning from a global point of view or simply share similar local properties. Overall, it requires a systematic approach to represent global composition and local feature information of the components properly. In this chapter, we present a hierarchical surface abstraction framework based on adaptive Mean Shift for automatic construction of such a shape representation. Through hierarchical abstraction, we can describe the shape in different levels-of-details. At each level, we only focus on a specific scale of components. This hierarchical abstraction approach is an ideal way to organize shape characteristics towards a representation for all-purpose retrieval tasks. The surface abstraction is achieved through an adaptive mean shift-based decomposition and a graph-based organization, which together form a powerful description of the shape even in case of deficient and noisy features.

The main contributions of this work are summarized as follow:

1. We propose a novel adaptive manifold mean shift approach to automatically decompose the surface into meaningful subregions, which are called Feature Enriched Components (FECs), based on its global and local statistics in the feature space. Since geometry and

topology information can both be encoded into the feature space, FEC decomposition is a more general way to characterize the shape. It can handle surfaces of complex geometry and arbitrary topology.

2. We present a hierarchical surface abstraction framework. The surface subregions are hierarchically organized into an attributed relational graph (ARG) which reflects both global structures and local levels-of-details of the shape in terms of topology and geometry. The compact shape description facilitates efficient matching and comparison.

3. The shape representation enables many different shape matching and retrieval tasks. Both the partial matching and deformable model retrieval can be achieved using FECs and ARG. Objects with high-genus shapes can be handled without extra efforts.

The rest of the chapter is organized as follows. Chapter 4.2 reviews the major work that is related to our approaches. In Chapter 4.3, we present the adaptive mean shift method and demonstrate how to use it to decompose a surface shape represented by triangular mesh into FECs. In Chapter 4.6, we explain how to use an attributed relational graph based representation to achieve global and local abstraction of a shape. In Chapter 4.7, we show the results of different shape retrieval applications based on our techniques. Finally, in Chapter 4.8, we discuss the advantages and disadvantages of our framework with a summary of this chapter

4.2 Related Work

Our work is motivated by the state of art in the shape matching field. This section reviews some most related work.

The basic operation of partial matching is to match similar geometries between a query input object and samples in the database. Therefore, the subregions of the sample must be properly defined. To avoid exhaustive search, some research work focused partial matching on "salient" region detection and comparison. One of the typical work was done by Gal and Cohen-Or [23]. They tried to find subregions that are "salient" based on local curvature variances. However, without considerable modification, it may not give the same discriminative power when features other than curvatures are considered. As an alternative, Shilane et al. [62] focused on finding distinctive regions on 3D surfaces. Unlike the salient region method, which measures how much a region sticks out from the rest of the object rather than how important the region is for defining the object type, the distinctive regions are obtained by searching for unique regions among the whole query set. The main problem of this method is its efficiency, the distinctive regions need to be recomputed when the query set is changed. Some other work proposed to extract salient points rather than subregions [45, 78]. Although these points can also take properties such as scales [38], they seem to be too local and not semantic enough for matching purposes. And they are error prone without careful local feature calculation.

Deformable shape retrieval methods are usually skeleton based or graph-based [67, 47, 48]. Although these approaches achieved certain success when models with similar skeletons are considered as in the same class, the skeleton is good at representing shape topology but not at describing surface geometry, thus, they are not accurate for detail matching. Furthermore, the nodes defined on the skeletons are too local to have enough representative power, thus intra-class dissimilarity can not be measured precisely. The idea of combining both topological and geometric surface representations of a shape had motivated Gary et al. to build a "surface skeleton" based on Level Set Diagram (LSD) [69]. In their implementation, the nodes are defined as surface subregions, which form an atlas of the surface. Geometry features are extracted from these subregions, so that they are more suitable for intra-class comparison. However, the algorithm can only handle genus-zero manifold, and may perform poorly when noise is present.

To overcome the difficulties of previous methods, we developed a method to extract FECs of a shape, which can support both partial and deformable model retrieval.

4.3 Surface Decomposition Based on Adaptive Mean Shift

We believe that the existing component-based shape representations are not suitable for universal tasks, since the component definitions are either not simple enough to be stable, or not having enough discriminative power.

As a summery, the components should have at least two properties to form a good shape representation for common applications such as matching:

1. The component should not be too small or too local. Otherwise, it is statistically unstable, and not semantically meaningful. The component should not contain a large and complex region either.

2. The samples inside a component should share the same geometric or physical features, thus, the components can be easily found by clustering. This also implies certain simplicities of the components.

These two properties ensure strong discriminative power, good robustness, and high efficiency.

Although in our implementation, the decomposition is base on general geometric features, such as curvature and normal, other features can be easily embedded to make the patches feature enriched. Thus, we name components satisfying the above two properties as feature enriched components (FECs). A modified mean shift algorithm is used to extract FECs from a shape.

4.4 Manifold Mean Shift and Mode Seeking

Mean shift is a robust approach for feature space analysis [15, 14]. It is widely applied in computer vision applications such as filtering ,segmentation and tracking. It is a statistical approach which is independent of resolution and noise competing. Shamir et al. in [60] extended the mean shift method for feature space analysis of triangular meshes.

The basic idea of mean shift is to find the statistical modes of the sample data, which are usually determined in a high dimensional feature space. The probability density function (PDF) of the data is estimated using a multivariate kernel density estimator:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} K(x - x_i), \tag{4.1}$$

where K is the d-variate kernel which usually takes the form of $K(x) = c_{k,d}k(\|x\|^2)$, where $c_{k,d}$ is a normalization constant.

To find the mode of each sample, in each step, the current sample point moves toward the mean value of a certain neighborhood along the gradient direction. The offset is calculated as:

$$\delta(x) = \frac{\sum_{i=1}^{n} x_i g(\|\frac{x-x_i}{h}\|^2)}{\sum_{i=1}^{n} g(\|\frac{x-x_i}{h}\|^2)} - x, \tag{4.2}$$

where $g(x) = -k'(x)$ and h is the bandwidth. In [15], Comaniciu and Meer further modified the d-variate kernel to be $K_{h_s,h_r}(x) = \frac{C}{h_s^2 h_r^p} k(\|\frac{x^s}{h_s}\|^2) k(\|\frac{x^r}{h_r}\|^2)$, so that the spatial part x^s (stands for locations) can be separated from the range part x^r (stands for properties) for respective controls.

Due to the sound robustness of mean shift, Shamir et.al had extended this framework onto triangular meshes [60]. However, directly move the metric for meanshift calculation onto a closed manifold will cause bad localization of the modes, and generate sibling modes 4.1. Furthermore, kernel size should be determined wisely since neighborhood query becomes expensive on a nonparametric surface.

To solve the above problems, we present a two-stage approach based on our novel adaptive mean shift, and the normalized cut.

4.4.1 Adaptive Manifold Mean Shift for Shape Decomposition

The adaptive manifold mean shift is designed to find the best kernel size for each sample (vertex) as follows:

1. Run manifold mean shift with a small window size w_{MIN}, which determines the smallest

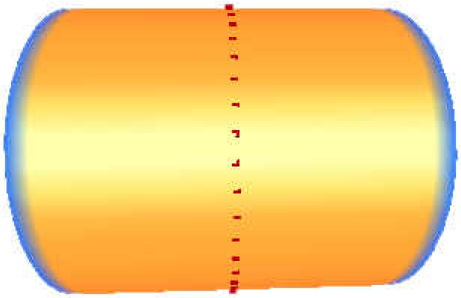


Figure 4.1: Sibling modes detected by manifold mean shift: the cylinder is colored by mean curvature, multiple modes are found on its mother lines, these modes represent the same cluster.

possible clusters in the final atlas. Set the repeatability r of each mode to be 1.

2. Increase the window size by δw and center windows at the modes found in step 1. Calculate the stability (will be discussed later) of the mode under the current kernel setting. If the corresponding mode is still a "basin of attraction", ie. an extremum, increase its repeatability r by 1.

3. Repeat step 2 until the mode is not stable any more, or the window size reaches the predefined maximum w_{MAX}, which determines the largest possible clusters. For each mode, store the maximum window size w_{max} under which the mode is still stable.

In order to decide the stability of a mode given a certain truncated kernel function, two terms are calculated:

1. The mean shift offset $\delta(m)$ from equation 4.2 by centering the window at the mode.

2. The estimated probability density $f(m)$ calculated by equation 4.1.

If the mode is still stable, $\delta(m)$ should not be too large and $f(m)$ shouldn't be too small. The decision is made by thresholding both of them.

After the above stability test, each vertex will be covered by windows with sizes, $dw^1_{max}, dw^2_{max}, \cdots,$ and dw^p_{max}, determined by adjacent modes $m_1, m_2, \cdots, m_p$. The largest window size is selected at the vertex for its mean shift calculation. In this first stage, the initial FECs are obtained by grouping vertices converging to the same mode.

4.4.2 Second Stage Clustering and Hierarchical FEC Decomposition

To deal with the bad localization of modes, a second stage clustering is carried out. After the adaptive manifold mean shift, we further cluster the modes according to their Euclidean distances between each other. This problem could be easily transformed to a graph partition problem, which has a robust solution by normalized cut [61]. Given an initial FEC decomposition with M clusters represented by M' modes, a weight matrix W for normalized cut is constructed as follows:

$$\begin{aligned} &W = \{w_{i,j} | i = 0, 1, \cdots, M-1; j = 0, 1, \\ &\cdots, M-1\} \\ &w_{i,j} = exp(-dis(C_i, C_j)), \end{aligned} \tag{4.3}$$

where $dis(m_i, m_j)$ is the average Euclidean distance between modes in cluster C_i and C_j. Some FECs are merged after the second stage clustering to eliminate bad mode localization.

Another important aspect of using normalized cut is that, by implementing the recursive n-way cut [61], it is easy to achieve a hierarchical FEC decomposition. By simply adjusting the Ncut value, the number of levels and the number of FECs in each level will be automatically determined, which is ideal for our automatic hierarchical surface abstraction.

4.5 Shape Descriptors for Matching

Once the decomposition is completed, the FECs can be compared for matching purposes. One way to do it is based geometry hashing [23]. In order to facilitate fast matching, we define a feature vector for each FEC. Two types of feature vector are designed and tested in our experiments, namely:

1. Type 1 feature vector (FV1) is defined as:

$$\begin{aligned} Vc =&\{n_{dev}, gc_{dev}, gc_{mean}, mc_{dev}, \\ &mc_{mean}, gd_{dev}, gd_{mean}\}, \end{aligned} \tag{4.4}$$

where n_{var} is the deviation of the surface normal, gc_{dev} and mc_{dev} are the deviations of Gaussian curvature and mean curvature respectively, gc_{mean} and mc_{mean} are the mean of Gaussian curvature and mean curvature respectively, gd_{mean} is the mean geodesic distance of all vertices from the center of the patch normalized by patch size, and gd_{dev} is the corresponding deviation of the geodesic distances. Note that Vc is affine and scale invariant.

2. Type 2 feature vector (FV2) is the heat kernel signature defined in [28]:

$$Vc = \{\sum_{i=1}^{m} exp^{-\lambda_i t}\phi_i | t = t_1, t_2, \cdots, t_n\} \tag{4.5}$$

where $\lambda_i, i = 1, 2, \cdots, m$ and $\phi_i, i = 1, 2, \cdots, m$ are the first smallest m eigen values and corresponding eigen vectors of the mesh Laplacian matrix, which is isometric invariant; t is the discrete time variable.

From the definition we can see that FV1 combines intrinsic (mean curvature) and extrinsic (Gaussian curvature) properties of a surface while FV2 only contains intrinsic properties. As a comparison, FV1 is more computational efficient and FV2 is more stable. Experiment shows that although the FEC decompositions are different based on FE1 and FE2, the retrieval results are comparable.

4.6 Surface Abstraction Based on Attributed Relational Graph

The last step of the shape abstraction procedure is to organize the FECs to form a complete and hierarchical description of the shape. An attributed relational graph is a good choice [85, 84]. Attributed Graph (AG) or Attributed Relational Graph (ARG) is a kind of part-based representation. An ARG is a graph $G = (V, E, A)$, where V is the vertex set, E is the edge set, and A is the attribute set. A consists of unary attribute a_i attaching to each node $v_i \in V$ and binary attribute a_{ij} attaching to each edge $e_{ij} = (v_i, v_j) \in E$. ARG can convert shape matching problems into graph matching problems.

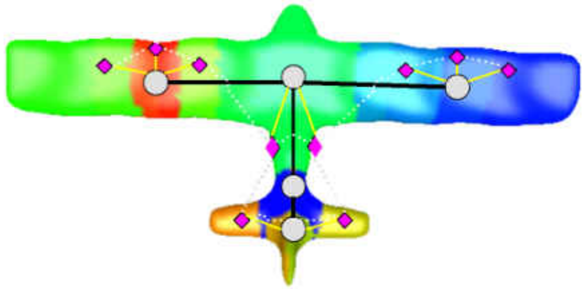


Figure 4.2: Splitting of vertices to build a hierarchical ARG: circles are graph nodes in the current level, diamonds are graph nodes in the child level. Solid and dashed lines indicate the edges in the current and child level.

In our framework, each node of the ARG is an FEC, and the associated attribute is the feature vector Vc defined in Chapter 4.5. Adjacent FECs are connected by an edge with e_{ij} being the average geodesic distance between node i and j. The abstraction is achieved by splitting the vertices of the ARG, from the coarsest level to the finest level. This process is illustrated in Figure 4.2.

The abstraction is extremely useful in tasks such as shape matching and retrieval, since this representation systematically integrate geometric and topological information, which makes it easy to handle deformable models and models of high genus. Matching scores from different levels are stored for further queries, ie., surface shapes exhibit different extents of similarity in different levels, which is implied by the abstraction hierarchy. The similarity of two shapes can be measured using the following standards for two different matching purposes:

1. Partial Matching: The highest matching score among different levels of FECs. The level producing the maximum score is called the best matching level.

2. Global Matching: The summation of the scores from all levels.

The following section demonstrates some matching results in order to show the superiority of our method.

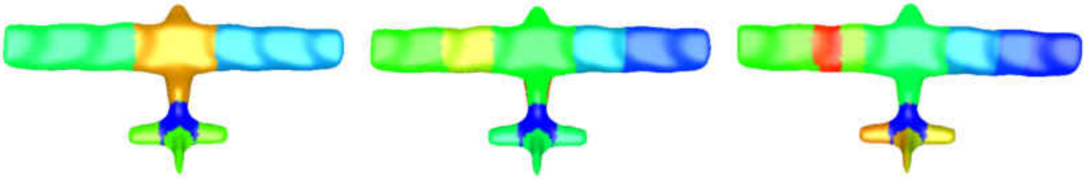


Figure 4.3: FEC decomposition at 3 different levels

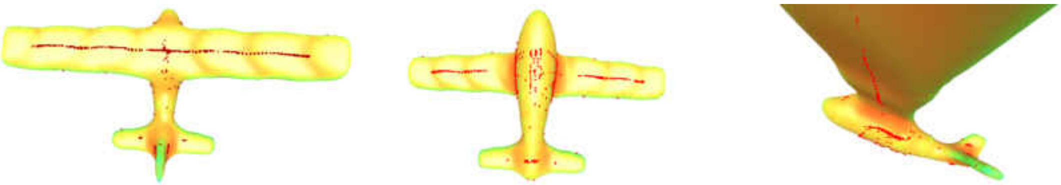


Figure 4.4: The red points are modes detected by the adaptive manifold mean shift. The model is viewed in three different angles in order to show all the detected modes.

4.7 Experiments

We have evaluated our algorithm on SHREC2008 dataset. Both rigid model and deformable model are tested. Figure 4.3 shows the decomposition of an airplane model at different levels using FE1.

Figure 4.4 demonstrates modes found on an airplane model (rigid model) with our adaptive manifold mean shift algorithm. Note that this model is quite noisy, even areas on the wings are quite bumpy. Figure 4.5 shows the result of FEC-based decomposition and matching, where two bird models are automatically decomposed into FECs and generate the matching correspondence. As for deformable model matching, we demonstrate the FEC decomposition on two horse models with different poses. They are matched based on their ARG representations. The FEC decomposition and matching correspondence is illustrated in Figure 4.6. Figure 4.7 shows that our FEC based graph can handle models with high genus as well, which is very difficult for many algorithms, e.g., the one presented in [69].

We have tested inter- and intra- class matching scores. We selected totally 50 objects from 5 categories. Each category contains 10 objects. We did the inter- and intra-class matching using the hierarchical surface abstraction based on adaptive mean shift and ARGs. The average

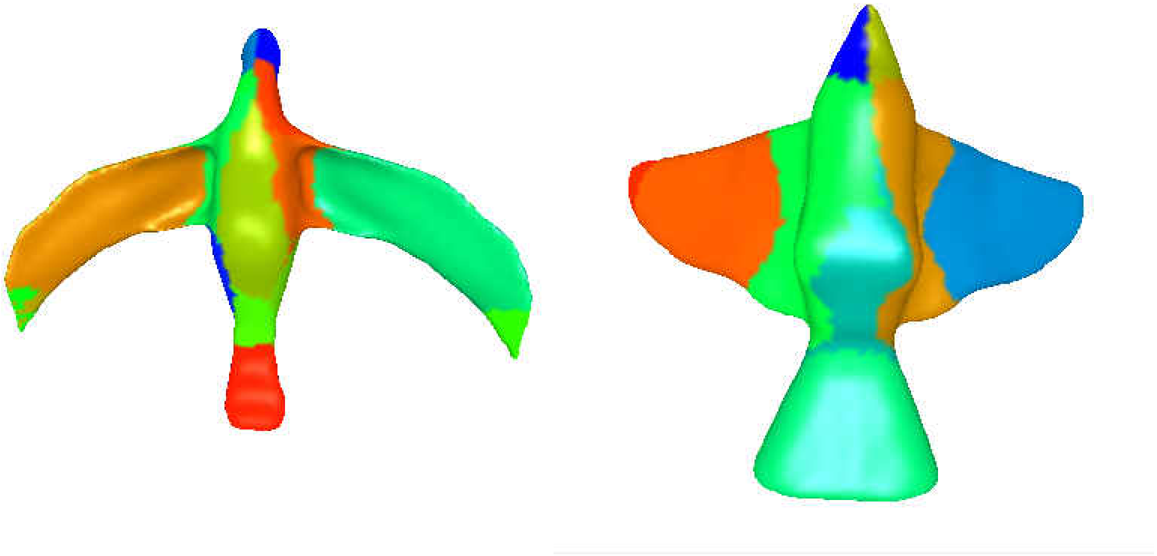


Figure 4.5: Two bird models matched based on hierarchical FEC decomposition and ARGs. The colors show the decomposition.

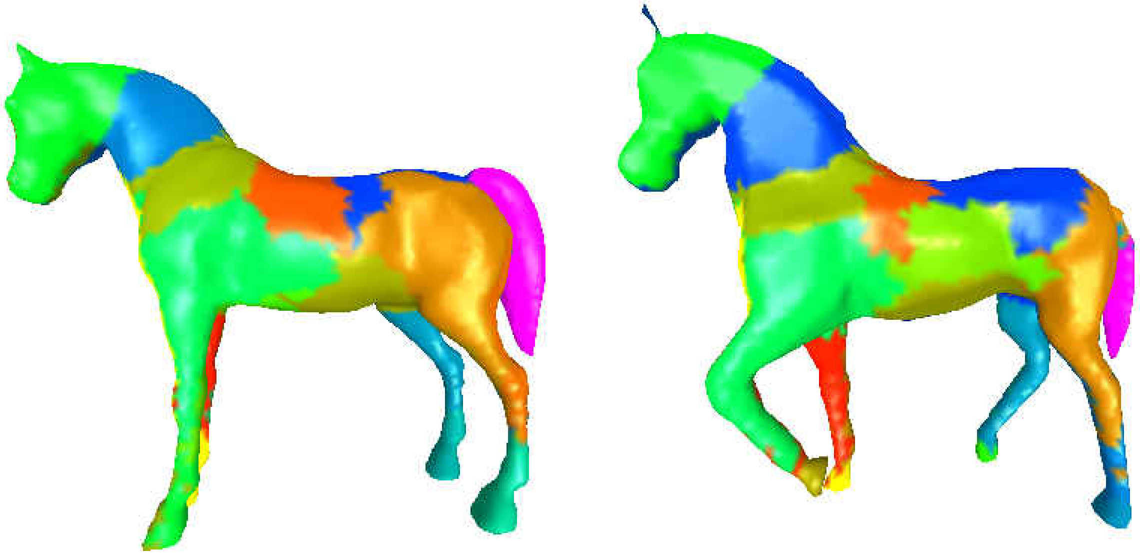


Figure 4.6: Two horse models with different poses matched with FEC decomposition and ARGs. The FEC decomposition and matching correspondence is illustrated (the matched FECs have the same color).

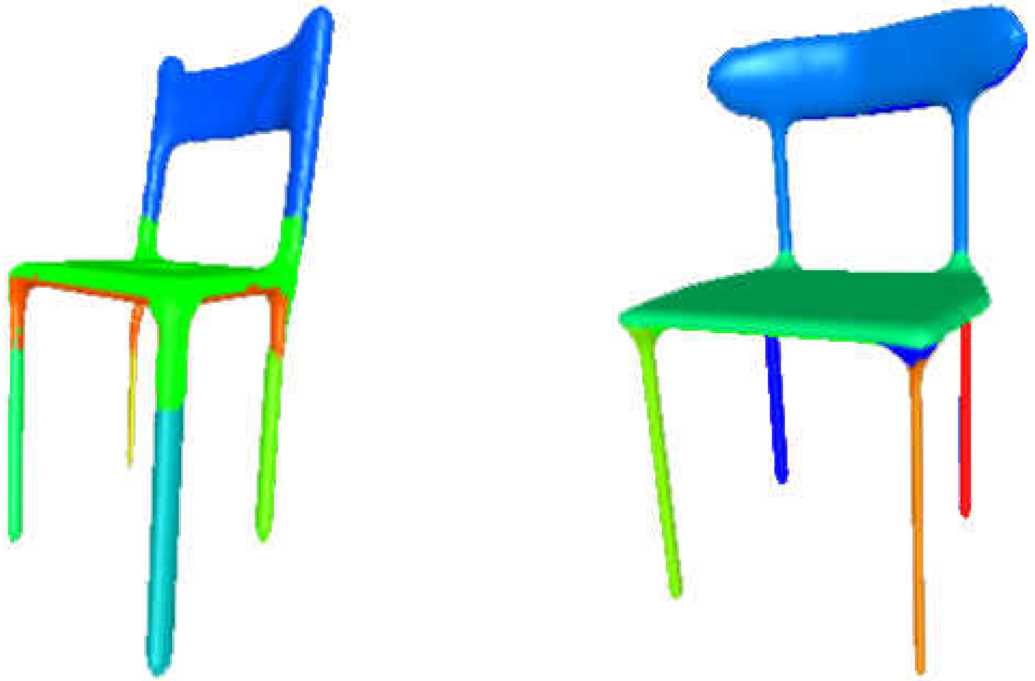


Figure 4.7: Two objects of genus one matched based on FECs and ARGs.

matching scores between each pair of categories are shown in Figure 4.8. From Figure 4.8 we can see that similar shapes have higher matching scores. We have also collected the average recall and precision rate for the whole SHREC2008 dataset with 20 categories with 20 objects in each category. The two rates are defined as $precision = \frac{\|S_l \cap S_t\|}{\|S_t\|}$ and $recall = \frac{\|S_l \cap S_t\|}{\|S_l\|}$, where S_l is the set of relevant objects and S_t is the set of matched objects. Figure 4.9 shows the change of precision and recall rate against the number of objects retrieved. Table 4.1 shows the best precision and recall rate from our algorithm and compared to the results from SHREC2008. Note that all the data used in the experiments contain partial shapes that are considered to be in the same category with the complete shapes. The results of our algorithm are comparable to those from SHREC2008, and we also retrieved many partial matched shapes.

4.8 Summary

In this chapter, we have designed a novel framework for surface abstraction which is well suitable for shape matching and retrieval. Compared to [69], our decomposition is in the com-

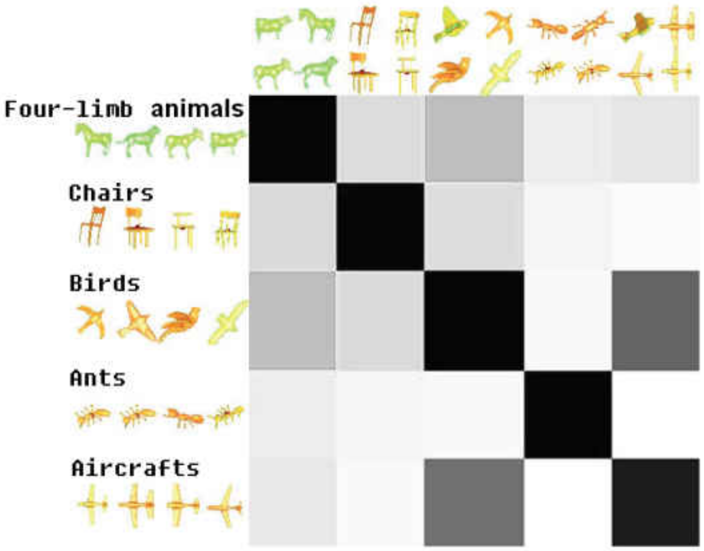


Figure 4.8: Matching scores placed in a matrix. Each row/column stands for one category, each cell of the matrix indicates the average matching score between two categories. Darker squares stand for higher scores.

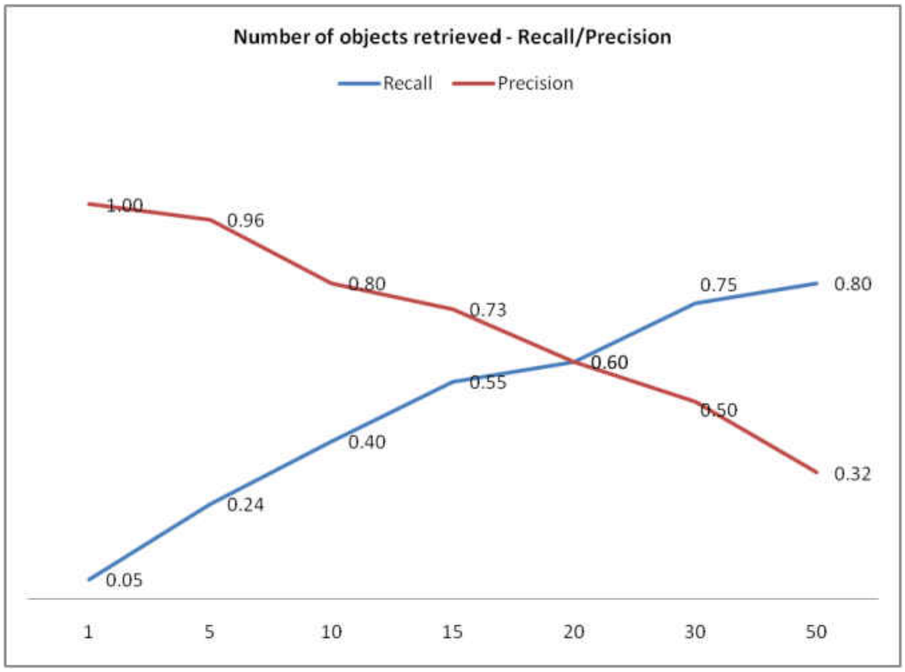


Figure 4.9: The change of precision and recall rate against the number of objects retrieved based on FV1.

Table 4.1: Comparison of the best precision and recall rate from our algorithm and the results from SHREC2008

Methods	FV1	FV2	SHREC (non-supervised)	SHREC (supervised)
Average Precision	60.01	62.80	54.36	63.10
Average Recall	60.01	62.80	60	65.14

posite feature space rather than in the physical space. This gives the possibility to make the components geometric and topological feature enriched instead of only topologically meaningful. In addition, our method can handle objects with high-genus topologies, and is insensitive to resolution and noise. Furthermore, our ARG is built in a hierarchical way towards a full abstraction of the input surface. The surface objects can be compared at different levels for fast computation.

CHAPTER 5

GENERIC MOTION ABSTRACTION FOR DEFORMABLE MESHES

The abstraction framework in the Chapter 4 provides an efficient way to approach mesh deformation by using a graph at the cost of losing the explicit geometry of the original shape in the meantime. Furthermore, by using feature vectors as graph nodes, the surface details are only represented statistically from the abstraction. Consider the case when the dense correspondences between shapes are available, It is feasible to achieve an abstraction of the deforming sequence where this abstraction should not only retain the original shape for recovery purposes but also provide a way to control the loss of geometric details. In this chapter we extend the surface abstraction framework to carry out motion abstraction by addressing the problem of motion compression based on deformable meshes.

5.1 Introduction

With the advance of 3D acquisition techniques [22, 25], more and more 3D models are available. On one hand, high resolution range scanners make 3D large terrain and cityscape data available; on the other hand, fast depth camera arrays enable us to capture 3D motion or facial expressions at a comparable speed with traditional video recorders [41]. Skeleton based animation and skinning animation are prevalent since they are easy to carry out and intuitive for manual editing in motion synthesis. However, compared to scanned mesh animation [73], they are less realistic since it is difficult to model the change of surface details during the motion and it is laborious to design sophisticated motions. Motion acquisition based on agents attached to humans provides an alternative, but it is still hard to capture the surface details especially when the motion is non-skeleton driven. The latest technology gives us a better stand. In [40], the subject in motion can be captured completely without using agents. The surface details are preserved at the same time. With this extra detail information acquired, 3D scanning motion data can be too large to store and process; thus, it prevents the data from being fast distributed to ordinary end users for entertainment or educational use. However, like 2D videos, the 3D scanned motion data usually contain large amounts of redundancies which can be reduced from the continuous sequence using proper algorithms. The core part of the algorithm is to use a limited number of shapes sampled at certain time stamps to predict the rest shapes in the series. In MPEG, this process is called motion compression. This term will be used in this chapter to address the same problem but on 3D scanned data. In MPEG, motion is in the same structured domain as other properties such as color and shading; thus they can be compensated at the same time. However, it is not straightforward to use the same technique to compensate motion and surface details in general since the canonical domain of the surface is changing during the motion. Furthermore, most conventional methods working on a single 3D object are not suitable for motion data analysis or need to be verified. This chapter mainly discusses how to implement efficient motion compression algorithms based on the new type of 3D motion data. The main contributions are:

1. Several algorithms derived from the state-of-the-art are implemented and compared.
2. A novel motion compression framework based on clustering and local transformation is presented.
3. The proposed algorithm is very easy to implement and high compression rates can be achieved without loss of surface details.
4. The algorithm is suitable for general motion including both skeleton driven and non-skeleton driven motion. The quality control is also easy to carry out.

The remaining contents are organized as follows: Chapter 5.2 surveys some work related to our proposed algorithms; Chapter 5.3 compared the results of some durable algorithms for motion compensation; Chapter 5.4 discussed our motion compression framework in detail;

Chapter 5.5 lists some experimental results based on our framework. Finally, Chapter 5.6 makes a summary of this chapter.

5.2 Related Work

To date, relatively less compression research work is carried out on real-world captured motion data than on image data. One reason is that using traditional techniques it is hard to retrieve the correlation between 3D data frames without proper post-registration; fortunately, more recent techniques have overcome the difficulty by incorporating tracking techniques and a deformable template into the scanning process. In [73], Vlasic et al. tried to match the deformable template to image silhouettes captured by a high definition camera array. Not a coincidence, Li et al. [40] used a template to match the partial scans from a depth camera. Their methods are efficient and easy to apply. It is expected that more and more this kind of 3D motion data will be available in the near future. The registration-on-the-fly gives us a consistently meshed sequence to directly analyze the motion on the surface, which provides us with better details than skeleton based methods.

For the purpose of 3D motion data storage, some techniques can be considered. Mesh streaming [27] stressed the compression of connectivity information of triangle meshes were proposed. Compared to other methods, it is out-of-core and is aimed at processing a very large single mesh. Geometric compression can be considered in either the spatial or the frequency domain. Traditional methods use position prediction [57] and ad hoc data structure to remove visual redundancies. An MPEG style codec framework based on linear prediction was proposed in [3]. More recently, spectral analysis has introduced new ways to geometry representation [32]. The challenge of spectral analysis on the manifold is the lack of a common domain. Thus it is usually nontrivial to find a set of analytical basis for an arbitrary shape [33]. The latest research on Laplacian-Beltrami eigen analysis [58] has provided us with tools to decompose the geometry into details at different scales. However, all these methods has little concern on motion data and direct extension to motion compression needs validation.

The most relevant work was from Yasmine et al. [8] who modelled the motion using local transformations. However, their method is preliminary with no surface detail preserving and smoothness control presented. Another similar work based on clustering was proposed in [6]; however, the motion is still modelled as the displacement of vertex coordinates which is lack of the power to describe local shapes. A thorough survey of dynamic mesh compression can be found in [63]. According to [63], our proposed work can be classified as multi-resolution geometry compression while existing topology compression methods can be overlaid with no extra effort. Compared to existing motion compression methods based on clustering, an analog in MPEG, our method does not rely on a predefined motion model but considers general local shape changes instead. The vertex prediction is replaced by transformation prediction as well. The low frequency information is well preserved by the decoding process.

Our work is motivated by the advancement of motion transfer and mesh editing. We can get one shape from another by applying the transformation between the two. Oscar et al. [7] have found that the face-based method has less distortion than vertex-based method since the neighborhood of each face form a simplex on which the local geometry can be easily defined. Fu et al. [21] further developed a method based on local vertex transformations, which preserves local shapes. However, this method needs certain perturbation on flat surfaces since the local transformation for a vertex cannot be estimated using its neighborhood in this case. Sumner et al. [66] solved this problem by adding a vertex in the normal direction of each face. Besides, Guo et al. [59] used a spectrum based method to deform the shape in a coarse scale and the details were added back afterwards. This makes the editing more efficient; However, to model the details as the hight function along the vertex normal will not be sufficient for complex patterns such as garment folds. The proposed work is also very related to [29] and [18] by chance where the techniques were used to build the skeletal structure of shapes. In contrast, our work focuses on motion data compression and is not restricted to articulated motions. Our framework is more rigorous and has more control over local geometry. In the following section we will compare some techniques which can be directly surveyed and used in 3D motion compression.

5.3 Comparison of Spatial and Spectral Methods

5.3.1 Notations and Terms

We'll use the following terms to refer to certain objects or operations:

1. Frame: a shape captured at one time stamp.

2. Key frame(s): the shapes captured at certain time stamps used to predict the rest shapes for compression purposes.

3. Intermediate frames: the shapes captured excluding key frames.

4. Predicted frame(s): the shapes that are predicted from the key frames to approximate intermediate frames.

5. Source mesh: the mesh before the current motion occurs. The terms, source mesh and target mesh, are used when we describe a general algorithm. When the algorithm is applied to motion compression, the source mesh is referred to as the key frame, and the target mesh becomes the intermediate frame.

6. Target mesh: the mesh as the result of the current motion.

In order to demonstrate our framework unique and essential, two types of techniques are implemented and compared in this section.

5.3.2 Motion Compensation Based on Spectral Decomposition

It is well known that the eigen functions of Laplacian-Beltrami operator form a complete basis on a manifold [58]. Furthermore, these bases are isometric invariant and can be used as a signature for similar shapes [58]. It is a widely used global representation of shapes. When the geometry is treated as a function defined on the manifold, the vertex positions can be mapped onto these basis; thus, compression can be achieved by quantizing the coefficients or discarding some of them in the same way as traditional discrete cosine transformation (DCT) encoding. Since articulated motions are near isometric, we can map vertex positions onto the basis of the key frames. For compression purposes, only the first 1000 coefficients are stored and used to recover the shapes. Two adjacent frames of a human body motion are used to test this algorithm. In Figure 5.1, the key frame is on the upper left and the intermediate frame is on the upper right. Using the first 1000 coefficients we can recover the key frame as shown on the lower left. Based on the same basis and the first 1000 coefficients of the intermediate frame, the corresponding predicted frame is shown on the lower right. Although the two original frames are similar to each other, the recovered shapes are quite different. While the recovered key frame is a smoothed version of the original shape, the difference is visually significant between the intermediate frame and the predicted frame.

It is obvious that when the metric of the surface changes slightly, the original basis are not suitable for decomposing the geometry into different levels of detail anymore. As an alternative, we also tried to use the same basis to decompose the differences between the intermediate frames and the key frames. This approach works when the two shapes are very similar. However, it fails when they differ from each other. We keep the key frame unchanged as in Figure 5.1(a) and use another intermediate frame shown in Figure 5.2(a). The predicted frame is shown in Figure 5.2(b). As can be seen, the distortion is dramatic. The above experiments show that an arbitrary motion can not be predicted precisely when the shape is considered using only global functions.

5.3.3 Motion Compensation Based on Spatial Transformation

As an alternative to the global spectral representation, the motion of a shape can be modelled as a set of transformations, i.e., use a spatial decomposition instead of a spectral decomposition. For example, each vertex bears a transformation during the motion, when the vertices with a similar motion are clustered, a concise representation can be achieved. This idea was

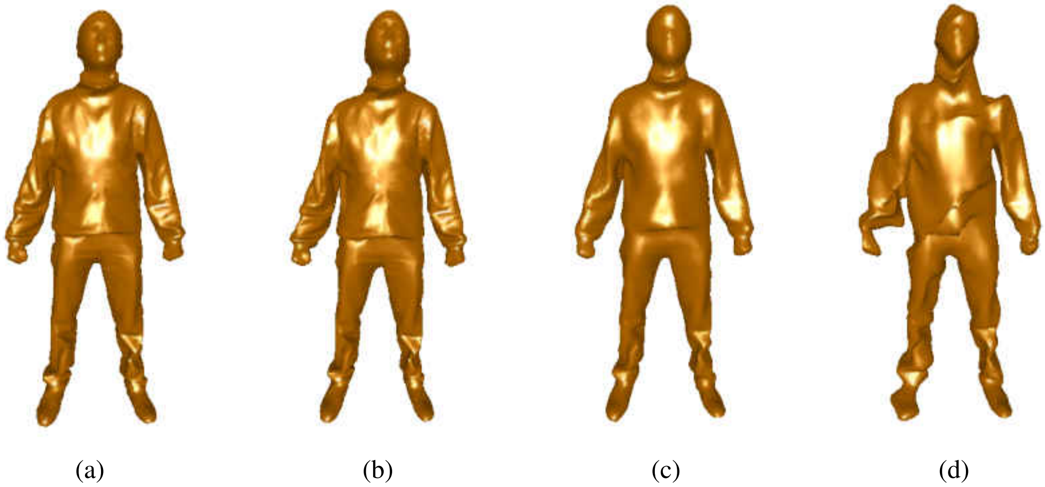


Figure 5.1: Recovered shape from spectrum decomposition: (a) - key frame; (b) - intermediate frame; (c) - recovered key frame; (d) - recovered intermediate frame (predicted frame).

explored in [8]. A matrix in its homogenous representation is used to depict the motion of each cluster of vertices. However, representing the transformation in a global coordinate system will introduce unnecessary translation terms and a single transformation of such does not contain any information of the original shape; Thus, the algorithm in [8] is not ideal for shape preserving motion compression. This is extremely clear when only a few number of clusters are used. In Figure 5.3, the algorithm is implemented and tested on two adjacent frames. The model contains 10000 vertices, and 256 matrices are used to recover the motion of the same number of clusters. The discontinuity on the predicted frame can be easily perceived. For this kind of real world scanned data, the garment motion is very complex. Each vertex will undergo a very unpredictable different motion from others.

In order to solve this problem, local transformations can be used instead. In [21], a vertex and its neighboring vertices are used to estimate the transformation. This algorithm not only successfully preserves local details to a certain extend but also faithfully picks up the global motion. The underline assumption is that when the neighboring vertices bear the same one transformation as the current vertex, the estimation error will be minimized. However, the correlation between a vertex and its neighborhood is not consistent among all the vertices.

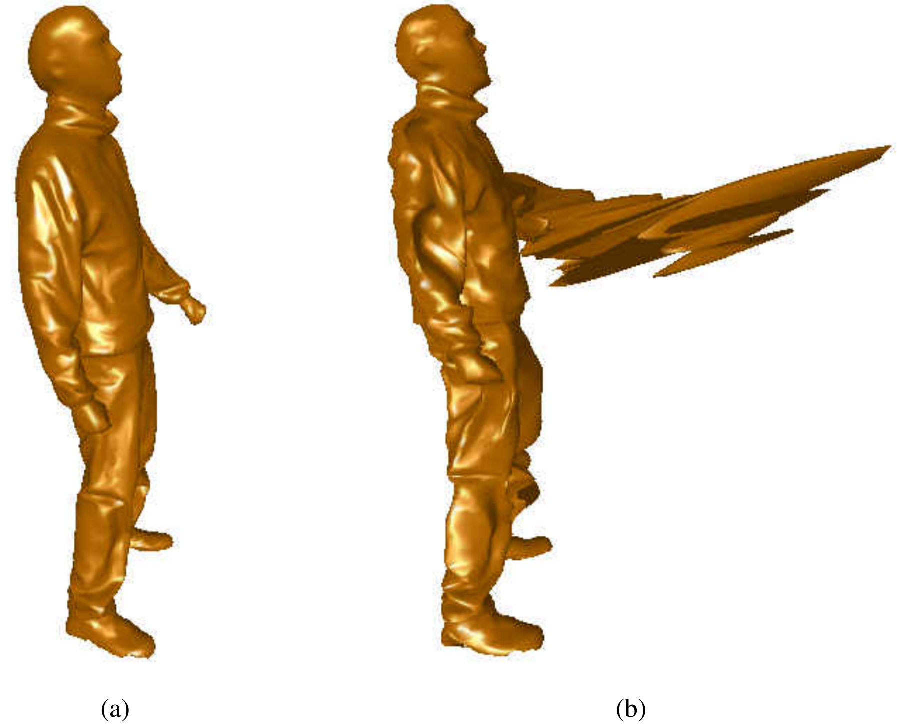
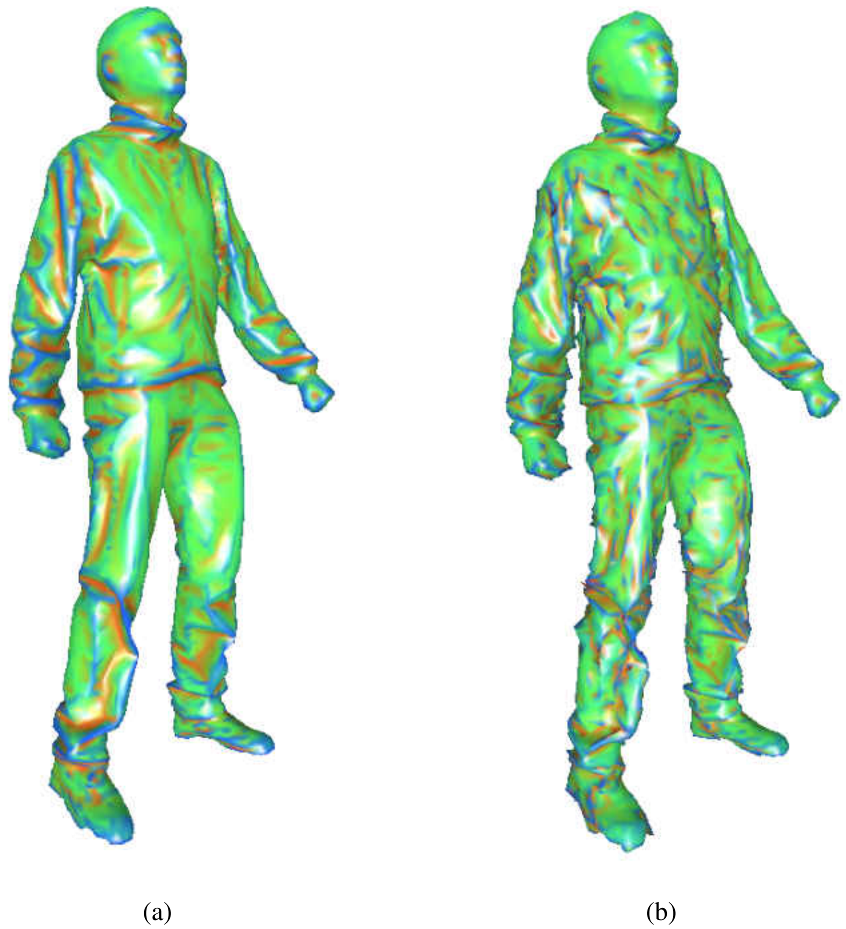


Figure 5.2: Another intermediate frame and the predicted frame based on the key frame in Figure 5.1(a).

When they are highly correlated, eg, co-planar, the transformation is not unique. In such cases, some local disturbance is introduce to regularize the local motion [21], which limits the application of the algorithm. One solution to this would be considering the transformation for each triangular face instead of each vertex as suggested in [7]. We will explore this idea in detail.

5.4 A Novel Motion Compression Framework Based on Local Transformation

Earlier research shows that a set of local transformations can be used to recover the global motion. However, as shown in the previous section, the local transformations based on vertices are not informative enough to describe local shape changes. The fundamental cause can be found in geometry saying that the shape of a primitive is determined by its angles and edge



(a)

(b)

Figure 5.3: (a) an intermediate frame; (b) the predicted frame based on [8] with 256 clusters, the details are distorted. The models are colored according to the mean curvature.

lengths. Thus, when trying to preserve the local geometry of a triangular mesh, we are particularly looking at the transformation for each triangular facet. The algorithm is briefly introduced as below.

In order to estimate the local transformation of a triangular facet, an additional vertex is required. Let $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i, i \in 1, \ldots, 3$, be the vertices of the triangle before the motion and after the motion, respectively. For a triangle in the mesh which has the configuration of $\mathbf{v}_i, i \in$

$1, \ldots, 3$, we compute the fourth vertex as,

$$\mathbf{v}_4 = \mathbf{v}_1 + (\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}1)/\sqrt{|(\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}1)|} \tag{5.1}$$

and the same computation for $\tilde{\mathbf{v}}_4$. An affine transformation defined by the 3×3 matrix **Q** and displacement vector **d** transform these four vertices as follows:

$$\mathbf{Q}\mathbf{v}_i + \mathbf{d} = \tilde{\mathbf{v}}_i,\ i \in 1, \ldots, 4. \tag{5.2}$$

If we subtract the first equation from the others to eliminate **d** and rewrite them in matrix by treating the vectors as columns, we obtain $\mathbf{QV} = \tilde{\mathbf{V}}$ where

$$\begin{aligned} \mathbf{V} &= [\mathbf{v}_2 - \mathbf{v}_1\ \mathbf{v}_3 - \mathbf{v}_1\ \mathbf{v}_4 - \mathbf{v}_1] \\ \tilde{\mathbf{V}} &= [\tilde{\mathbf{v}}_2 - \tilde{\mathbf{v}}_1\ \tilde{\mathbf{v}}_3 - \tilde{\mathbf{v}}_1\ \tilde{\mathbf{v}}_4 - \tilde{\mathbf{v}}_1]. \end{aligned} \tag{5.3}$$

So **V** can be calculated by

$$\mathbf{Q} = \tilde{\mathbf{V}}\mathbf{V}^{-1}. \tag{5.4}$$

Compared to other work, this local transformation Q contains solely the shape change of a triangular facet during the global motion. By clustering of Q for all the facet between intermediate and key frames, it is possible to group similar surface changes and remove redundancies accordingly.

The following algorithm carries out the motion compensation and recovers the shape based on all the known local transformations. We use the terms source mesh and target mesh as defined in Sec. 5.3.1. Since the one-to-one correspondence is known between two arbitrary frames. Therefore, there are pairs of transformation $\{\mathbf{S}_i,\mathbf{T}_i\}$ for the source mesh and the target mesh. In order to maintain consistency, additional constraints are added to the transformation:

$$\mathbf{T}_j\mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k\mathbf{v}_i + \mathbf{d}_k, \forall i, \forall j, k \in p(v_i), \tag{5.5}$$

where $p(\mathbf{v}_i)$ is set of all triangles that share vertex $\mathbf{v}_i$. In order to solve the transformation, the difference between the source and target transformation under the consistency constraint has to be minimized:

$$\min_{\mathbf{T}_1+\mathbf{d}_1,\ldots,\mathbf{T}_{|T|}+\mathbf{d}_{|T|}} \sum_{j=1}^{|T|} \|\mathbf{S}_j - \mathbf{T}_j\|_F^2$$

$$\text{subject to}$$

$$\mathbf{T}_j\mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k\mathbf{v}_i + \mathbf{d}_k, \forall i, \forall j, k \in p(v_i), \tag{5.6}$$

where $\|\cdot\|_F$ is the Frobeniu norm.

For the target mesh, the $\mathbf{V}$ depends on the known, the elements of of $\tilde{\mathbf{V}}$ are the coordinates of the unknown deformed vertices. Thus, the elements of $\mathbf{T}$ are linear combinations of the coordinates of the unknown, deformed vertices $\mathbf{T} = \tilde{\mathbf{V}}\mathbf{V}^{-1}$. Based on this fact, the minimization problem can be rewritten as

$$\min_{\tilde{\mathbf{v}}_1,\ldots,\tilde{\mathbf{v}}_n} \sum_{j=1}^{|T|} \|\mathbf{S}_j - \mathbf{T}_j\|_F^2. \tag{5.7}$$

The solution to this problem is the solution to a system of linear equations. Rewriting the problem in matrix form yields

$$\min_{\tilde{\mathbf{v}}_1,\ldots,\tilde{\mathbf{v}}_n} \sum_{j=1}^{|T|} \|\mathbf{c} - \mathbf{A}\tilde{\mathbf{x}}\|_2^2, \tag{5.8}$$

where $\tilde{\mathbf{x}}$ is a vector of the unknown deformed vertex locations, $\mathbf{c}$ is a vector containing entries from the source transformations, and $\mathbf{A}$ is a matrix that relates $\tilde{\mathbf{x}}$ to $\mathbf{c}$. The final solution is the following form:

$$\tilde{\mathbf{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{c}. \tag{5.9}$$

Before the above algorithm is applied to motion compression, we need to notice that a global translation may occur during the motion besides the shape change between frames. It is effortless to store the position of an arbitrary chosen vertex (called anchor vertex) for each intermediate frame and translate the predicted frame to meet this criteria. As an alternative, we can feed the information into Eq. 5.8 by moving the corresponding column from $\mathbf{A}\tilde{\mathbf{x}}$ to $\mathbf{c}$ to form another linear system as suggested by Sumner [66]. Furthermore, if the model has open boundaries such as the facial model we used, more than one anchor vertex is needed to regularize the motion along the boundary.

By utilizing the above techniques, our motion compression framework is described as follows:

Algorithm 1: Encoding:

1. Determine the key frames and intermediate frames from a motion series. All the intermediate frames prior to the next key frame will be predicted by the current key frame. The intermediate frames and predicted frames are paired with the current key frame.

2. For each key frame and intermediate frame pair, the local transformations between the couple are calculated according to Eq. 5.4.

3. Group the transformations for each pair using k-means clustering. Generate one transformation for each cluster and store them together with the cluster labels for each intermediate frame. Store an additional translation vector or a known vertex position for the intermediate frame as well.

Algorithm 2: Decoding:

1. Each predicted frame is obtained by assigning a transformation to each facet of the paired key frame accordingly to the stored information using Eq. 5.9.

2. Apply the global translation to each predicted frame if needed.

The transformation Q_k for each cluster $\mathbf{C}_k, k = 1, 2, \cdots$, can be generated by minimizing the prediction error.

$$\min_{\mathbf{Q}_k} \sum_{n=1}^{\|\mathbf{C}\|} \|\mathbf{Q}_k \mathbf{v}_n^k - \tilde{\mathbf{v}}_n^k\|^2, \mathbf{v}_n^k \in C_k. \tag{5.10}$$

The vectors $\mathbf{v}_n^k$ and $\tilde{\mathbf{v}}_n^k$ are defined as in Eq. 5.3.

In fact, since the consistent constrain will regularize the transformations, in practice, we can use the transformation of an arbitrary triangular facet in the cluster as $\mathbf{Q}_k$. Our experiments show that this algorithm works well for all the data we tested.

5.5 Experiments

We have examined our framework based on MIT's articulated mesh animation data sets [73]. It includes simple human motions such as marching (Figure 5.4) and jumping as well as complicated ones like handstand (Figure 5.6) and dancing (Figure 5.5). Each data set contains 148 to 250 frames. The characters for scanning all wear loose clothes which deform very randomly and create a lot of garment movements. The scanned models are triangulated with 10002 vertices and 20000 faces. We also tested our framework on a facial sequence with expression change (Figure 5.7), which is consist of triangular meshes with 1500 vertices and 3000 faces. For most of the body motion data, 256 clusters are created using $\mathfrak{k}$-means clustering on the 20000 faces, where the errors are bounded within 0.5% for all the data we have tested. For the facial data , 64 clusters are used. Thus, 5/8 byte is used to store the cluster label for each face and 1 transformation matrix of 3 by 3 is stored for each cluster. For each intermediate frame predicted, the bit rate is calculated according to the following formula:

$$\frac{B_M * N_C + \log N_C * N_F}{N_V} (bits/vertex/frame), \tag{5.11}$$

where B_M is the number of bits used to represent a matrix, N_C is the number of face clusters, N_F is the number of faces and N_V is the number of vertices. If 8 bytes are used for a floating point number, the bit rate will be $30.74 bits/vertex/frame$ and $20.29 bits/vertex/frame$ for the body motion data and facial data, respectively, without further compression. Compared to uncompressed data of $192 bits/vertex/frame$, very high compression rates can be achieved. This result is comparable to that reported in [6] with the same error bounds. However, our work outperforms in terms of the visual errors for low compression rates. Thus, it is good for entertainment use and potentially real time visual processing applications such as 3D video surveillance. In most data sets we have tested, only one key frame (the first frame is used as the key frame) is needed to obtain visually acceptable predictions, i.e., all the rest frames in a sequence are intermediate frames and are compressed by using **Algorithm 1**. Thus, the average bit rate of a sequence will be approximately the same as the numbers reported above.

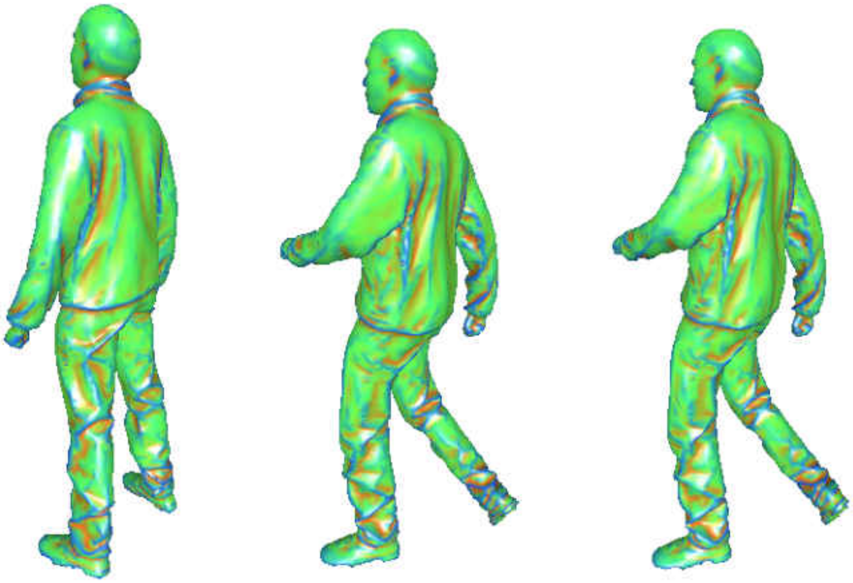


Figure 5.4: Two frames from a marching sequence. Left: key frame; middle: intermediate frame; right: predicted frame. The models are colored according to the mean curvature.

Since the compression is lossy, the following formula is used to calculate the prediction

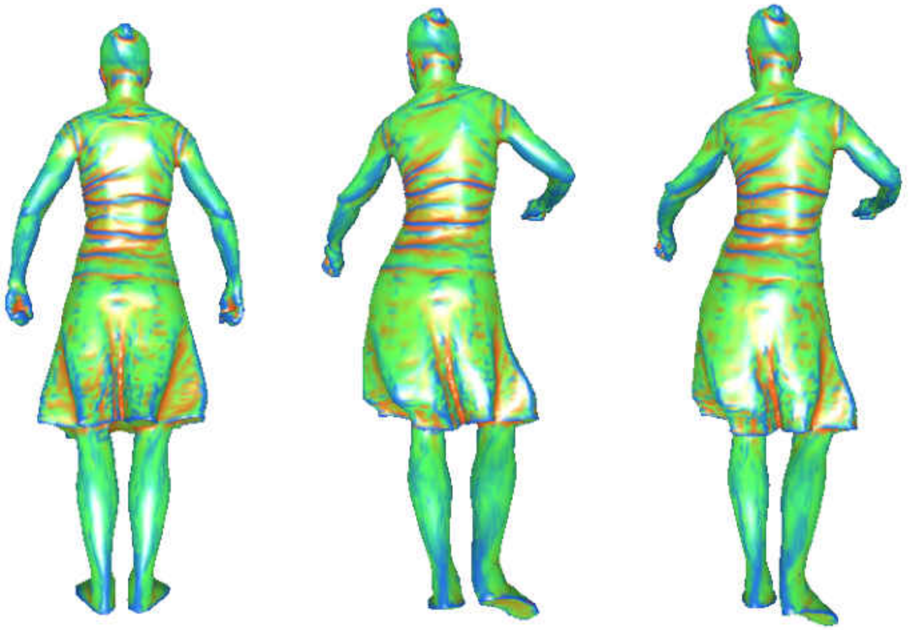


Figure 5.5: Two frames from a dancing sequence which contains fine garment motion. Left: key frame; middle: intermediate frame; right: the predicted frame based on 256 clusters. The models are colored according to the mean curvature.

error for a predicted frame:

$$Err_{predict} = (\sqrt{\sum_i \|\mathbf{v}_i - \tilde{\mathbf{v}}_i\|_2^2 / N_v}) / \overline{L_e}, \tag{5.12}$$

where $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i$ are vertices of the intermediate frame and the predicted frame, respectively, for $i = 1, 2, ..., N_v$, and $\overline{L_e}$ is the average edge length of the model over all the frames. This measurement mainly captures the variance of surface details. Table 5.1 shows the expectation of $Err_{predict}$ for all the predicted frames. The prediction error is proportional to the number of clusters used for encoding.

We have also calculated the prediction errors when different numbers of clusters are used. The $Err_{predict}$-N_C curve is show in Figure 5.8. Figure 5.9 shows the clusters calculated between the paired key frames and the intermediate frames in Figures 5.4, 5.6, 5.5, and 5.7, respectively.

Table 5.2 shows performance of the algorithms in terms of speed. All the computations are conducted on a Core2 Quad CPU with 6G DDR2 memory.

Table 5.1: Prediction error

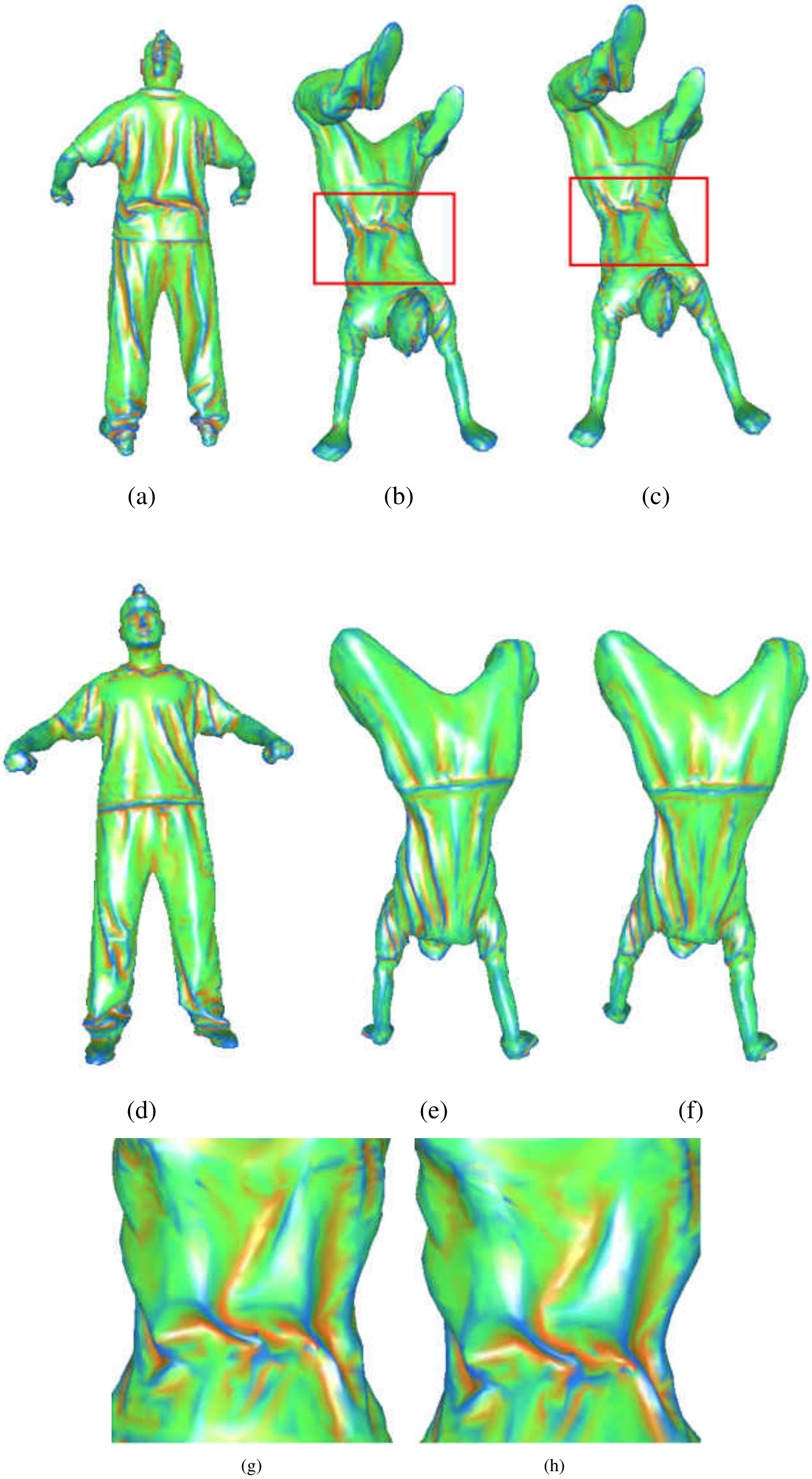
Data set	bouncing	crane	jumping	March	samba	squad	swing	handstand	face
Error	5.1E-05	1.7E-05	0.000145	2.2E-05	1.1E-5	2.1E-05	1.4E-05	0.000215	0.01

Table 5.2: Time taken for computing one frame (in seconds)

Data set	bouncing	crane	jumping	March	samba	squad	swing	handstand	face
Encoding	109	130	156	145	134	128	189	178	12
Decoding	22	23	21	22	25	21	22	20	0.3

5.6 Summary

In this chapter, a novel 3D motion compression framework is presented. It works on registered triangular mesh sequences. The core algorithms are based on clustering of the local transformations of the triangular faces. Besides, we implemented and tested several state-of-the-art techniques on motion compression. Compared to other work, our approach can preserve the surface details during very complicated motions while achieving very high compression rates. The framework can also be used on non-skeleton driven motions directly. Our experiments show that our motion compensation algorithm successfully captures human motion with garment movements and facial motion with expression changes. This work can be seen as a generic motion abstraction for deformable meshes. It provides a potential way similar to that presented in Chapter 4 to compare and retrieve mesh deformation sequences. This is the future research direction when more motion data become available for testing.



(a) (b) (c)

(d) (e) (f)

(g) (h)

Figure 5.6: Two frames from a handstand motion where the surface changes more significantly than others. (a): key frame (b): intermediate frame; (c): the predicted frame based on 512 clusters, The models are colored according to the mean curvature; (d) another view of the key frame in (a); (e) another view of the key frame in (b); (f) another view of the key frame in (c); (g) zoom in view of the rectangular area in (b); (h) zoom in view of the rectangular area in (c).

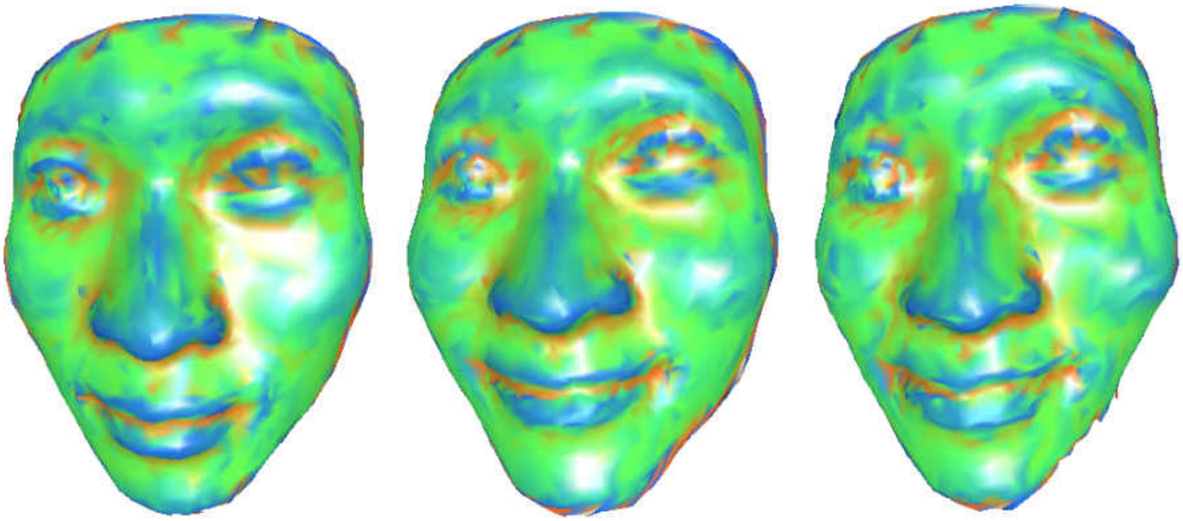


Figure 5.7: Two frames from a facial motion which is non-skeleton driven. Left: key frame; middle: intermediate frame; right: predicted frame.

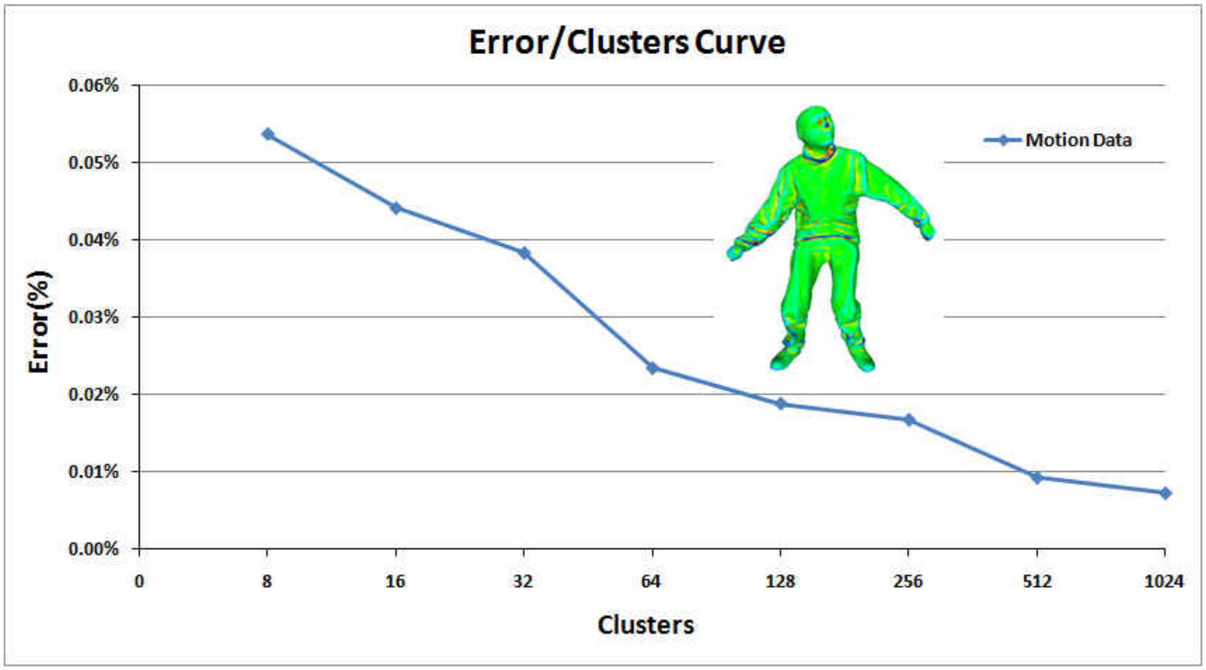


Figure 5.8: The prediction error $Err_{predict}$ change against the number of clusters used.

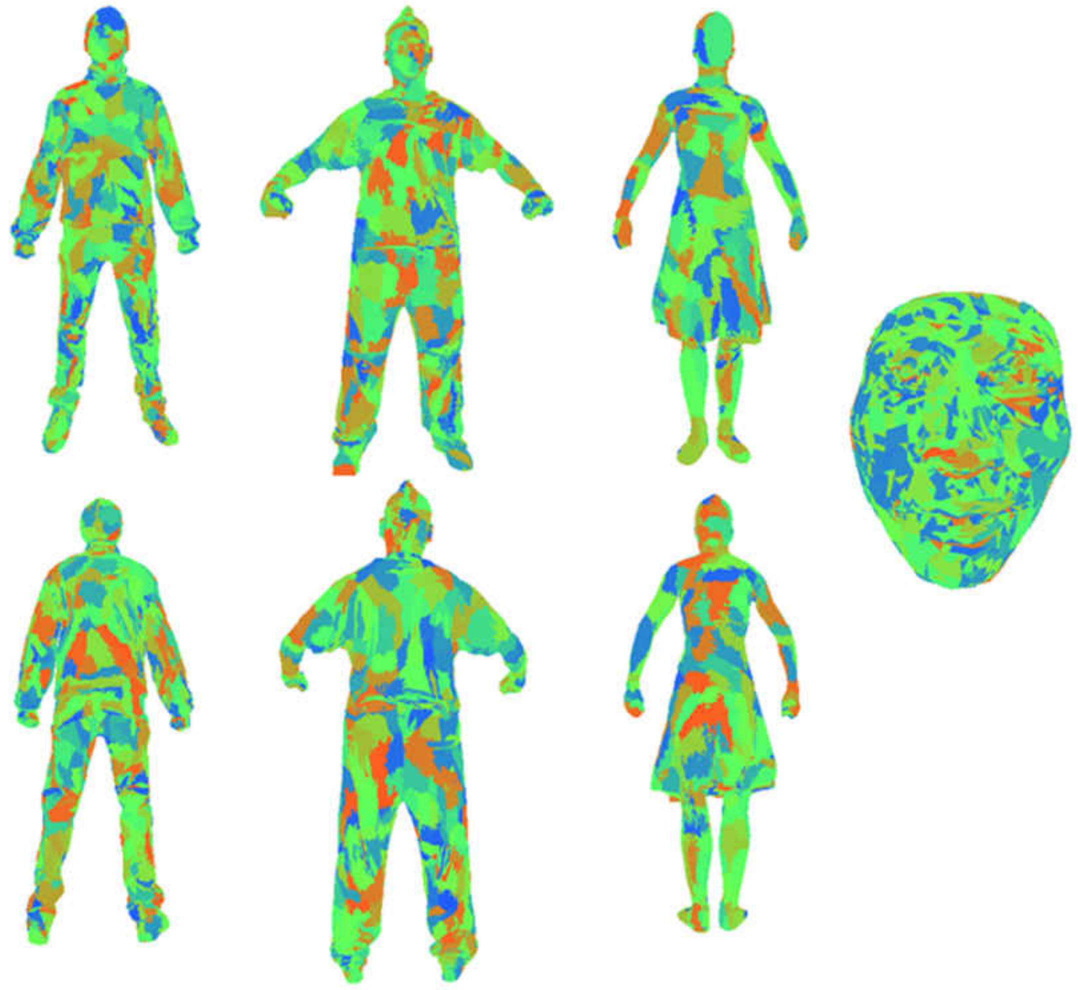


Figure 5.9: The clusters of local motion (calculated for each face) between the key frames and intermediate frames for the data shown in Figures 5.4, 5.6, 5.5, and5.7. Different colors represent different clusters for each data set. The bottom row shows different views of the top row.

CHAPTER 6

CONCLUSION

In the previous chapters, deformable meshes are used to solve typical problems in computer graphics regarding shape recovery. Proper deformation strategies are devised for various applications. A displacement based deformation is used to recover arbitrary non-smooth shapes from images; it is both faster and more versatile compared to traditional methods. The mesh deformation is simplified to graph matching for the purpose of recovering similar shapes more efficiently when the explicit geometry of the target is known by example. The method is presented as an abstraction of the shape. Besides, a generic motion abstraction framework for deformable meshes is devised in terms of local transformation clustering so that it retains the explicit geometry of the shape. A novel 3D motion compression algorithm is devised based on the generic framework. The above work covers some of the most important tasks in shape recovery: modelling the deformation and solving the deformation. Many successful applications are presented with still a lot more left to be discovered. Limited to the scope of this dissertation, this chapter will make a conclusion and leave some openings for future work.

6.1 Contributions

The major contributions of this dissertation are re-listed here.

1. A novel mesh deformation framework based on geometric flow is devised (Chapter 3). It allows geometry changes under internal and external forces as well as simple topological changes such as splitting and merging. Two types of geometric flows, namely normal flow and tangential flow, are proposed for recovering generic and tube like shapes respectively. The model is displacement based, it outperforms other deformable models by its simple implementation using meshes and its versatility of modelling arbitrary non-smooth shapes. These features are ideal for modelling deformable medical instances

that have non-smooth surface geometry but relatively simple topologies. The model is designed to implicitly recover shapes from their background such as images, and it is successfully applied to build a robust segmentation and visualization system of brain tumor systems.

2. A novel shape abstraction framework is devised for deformable meshes (Chapter 4). A manifold mesh is simplified to a graph of geometry features of interest. It generalizes the mesh deformation into a relatively simple graph problem. A two-stage clustering algorithm based on manifold mean-shift and normalized graph cut is devised to build the graph. Besides, a new similarity measurement based on graph matching is proposed for comparing deformable objects more efficiently. Giving a representative, the system can be used to recovery similar/partially similar shapes in a bank of other shapes. We apply the method to 3D shape retrieval, and it gains more partial matching while retaining comparable performance in terms of precision and recall rate than previous work giving the same testing data.

3. By extending the shape abstraction algorithm, a generic motion abstraction framework is devised for deformable meshes based on local transformation clustering. It can model detailed geometry changes of the mesh instead of only some of its properties. Thus, it provides a novel approach for 3D motion compression (Chapter 5). It utilizes the latest development in mesh deformation to minimize the prediction error of the surface geometry and obtains a global optimization in both fine and large scales. The framework differs from previous work in that it is totally based on local geometry transformation rather than global vertex displacement. It can also be used on non-skeleton driven motions directly. Experiments show that our work outperforms previous work in terms of the visual errors for low compression rates. Thus, it is good for entertainment use and potentially real time visual processing applications such as 3D video surveillance. This work can be seen as an extension of our shape abstraction algorithm where motion abstraction is

achieved.

For more details, the reader can refer to the corresponding chapters.

6.2 Future Work

As a fundamental task in computer graphics, shape recovery is still popping up new challenges to the researchers. The models presented in the completed work can be easily refined to adapt to new applications arisen. Some of the future work are listed below.

1. Volumetric (3D) deformable mesh. Up to now, our models all use surface (2D) meshes. Some applications such as human heart motion recovery require modelling of the object's inner structure instead of only its surface since the volume change of the object is key to the analysis. Some of the deformation formulas can be directly extended to have another dimension, but it is still challenging to design proper dynamical behavior for the volumetric model since high dimensional mesh structures are difficult to maintain. Also, the additional degree of freedom introduces new regularization problem. At a glance, proper heuristics based on expertise and ad hoc algorithms for specific cases are needed. We will be looking into these issues in our research.

2. Learning based deformation. The idea to introduce prior knowledge to guide the mesh deformation. Besides the mathematical formulation of the latest discovery in kinetics and medical imaging, existing results from domain experts and sparse landmarks can be used for supervised learning. The deformable model will gain better convergence and accuracy by doing so. For example, the graph model in Chapter 4 can be trained to gain better matching results; however, only efficient enough for simplified meshes such as those after our abstraction. A potential research direction would be to regularize the movement of each mesh vertex by feeding examples. By observing the parameter change during the deformation, it is possible to identify "abnormal" shape occurrences. This could be very useful in abnormality detection.

APPENDIX

Publications

- Journals

1. C. Liu, and J. Hua, Robust Segmentation and Visualization of Brain Tumor System Based on Dynamic Differential Surface Propagation, IEEE Transactions on Information Technology in Biomedicine(Submitted).
2. V. Taimouri, X. Liu, Z. Lai, C. Liu, D. Pai, and J. Hua, Colon Segmentation for Prepless Virtual Colonoscopy, IEEE Transactions on Information Technology in Biomedicine, 2011, vol. 15, no. 5, pp. 709-715.

- Conferences

1. C. Liu, Z. Lai, J. Hu and J. Hua, "Detail Preserving 3D Motion Compression Based on Local Transformation." In Proceedings of The Fourth Pacific-Rim Symposium on Image and Video Technology(PSIVT) , 2010.
2. C. Liu, J. Hu, J. Hua, and H. Qin, "Hierarchical Surface Abstraction Using Adaptive Mean Shift." In Proceedings of the 21 st International Conference on Computer Animation and Social Agents (CASA) , 2008.
3. C. Liu and J. Hua, "An Integrative Neural Network System with Feedback Control for Classification." In Proceedings of International Joint Conference on Neural Network(IJCNN) , 2006.
4. Z. Lai, J. Hu, C. Liu, V. Taimouri, D. Pai, J. Zhu, J. Xu, and J. Hua, "Intra-patient Supine-Prone Colon Registration in CT Colonography Using Shape Spectrum." In Proceedings of the 13th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) , 2010.

5. D. Wu, C. Liu, G. Zou, J. Hua and O. Muzik, “Conformal Contour Mapping for Neurosurgery Outcome Evaluation.” In Proceedings of IEEE International Symposium on Biomedical Imaging (ISBI), 2007.

BIBLIOGRAPHY

[1] *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics*. Kitware, Inc. publishers.

[2] AHMED, S., IFTEKHARUDDIN, K., AND VOSSOUGH, A. Efficacy of texture, shape, and intensity feature fusion for posterior-fossa tumor segmentation in mri. *IEEE TITB* (2011).

[3] AHN, J., KIM, C., AND HO, Y. Predictive compression of geometry, color and normal data of 3D mesh models. *IEEE Transactions on Circuits and Systems for Video Technology 16*, 2 (February 2006), 291–299.

[4] ALEXA, M. Differential coordinates for local mesh morphing and deformation. *The Visual Computer* (2004), 105–114.

[5] ALLEN, C. B., AND B., Z. P. Exploring the space of human body shapes: Data-driven synthesis under anthropometric control. In *Proceedings of Conference on Digital Human Modeling for Design and Engineering* (2004).

[6] AMJOUN, R., AND STRABER, W. Efficient compression of 2d dynamic mesh sequences. *Journal of WSCG* (2007), 99–106.

[7] AU, O. K.-C., TAI, C.-L., LIU, L., AND FU., H. Dual laplacian editing for meshes. *IEEE Transaction on Visualization and Computer Graphics 12*, 3 (May-June 2006), 386–395.

[8] BOULFANI-CUISINAUD, Y., AND ANTONINI, M. Motion-based geometry compensation for dwt compression of 3D mesh sequence,. In *IEEE International Conference on Image Processing (ICIP)* (2007), vol. 1, pp. 217–220.

[9] CASELLES, V., KIMMEL, R., AND SAPIRO, G. Geodesic active contours. *International Journal of Computer Vision 22* (1997), 61–79.

[10] CHAN, T., AND VESE, L. Image segmentation using level sets and the piecewise-constant mumford-shah model. CAM Report 00-14, UCLA, 2000.

[11] CHAN, T., AND VESE, L. Active contours without edges. *IEEE Trans on Image Processing 10* (2001), 266–277.

[12] CHAN, T., AND ZHU, W. Level set based shape prior segmentation. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2* (2005), 1164–1170.

[13] CHEN, J., AND AMINI, A. A. Quantifying 3-d vascular structures in mra images using hybrid pde and geometric deformable models. *IEEE Transactions on Mediacal Imaging 23*, 10 (2004), 1251–1263.

[14] CHENG, Y. Mean shift, mode seeking, and clustering. *TPAMI 17*, 8 (1995), 790–799.

[15] COMANICIU, D., AND MEER, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence 24*, 5 (2002), 603–619.

[16] CORSO, J., SHARON, E., DUBE, S., EL-SADEN, S., SINHA, U., AND YUILLE, A. Efficient multilevel brain tumor segmentation with integrated bayesian model classification. *IEEE TMI* (2008).

[17] DELINGETTE, H. General object reconstruction based on simplex meshes. *International Journal of Computer Vision 32*, 2 (1999), 111–146.

[18] DER, K. G., SUMNER, R. W., AND POPOVI, J. Inverse kinematics for reduced deformable models. In *ACM SIGGRAPH* (2006), pp. 1174–1179.

[19] DUAN, Y., HUA, J., AND QIN, H. Hapticflow: Pde-based mesh editing with haptics. *Computer Animation and Virtual World 15*, 3-4 (July 2004), 193–200.

[20] FATAKDAWALA, H., XU, J., BASAVANHALLY, A., BHANOT, G., GANESAN, S., FELDMAN, M., TOMASZEWSKI, J., AND MADABHUSHI, A. Expectation-maximization-driven geodesic active contour with overlap resolution (emagacor): Application to lymphocyte segmentation on breast cancer histopathology. *IEEE TBE* (2010).

[21] FU, H., AND TAI, C.-L. Mesh editing with affine-invariant laplacian coordinates. Tech. Rep. HKUST-CS05-01, Hong Kong University of Science and Technology, 2005.

[22] FURUKAWA, Y., AND PONCE, J. Dense 3D motion capture for human faces. In *IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 1674–1681.

[23] GAL, R., AND COHEN-OR, D. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics, January 25*, 1 (2006).

[24] GAO, S., AND BUI, T. Image segmentation and selective smoothing by using mumford-shah model. *IEEE Transactions on Image Processing 14*, 10 (2005), 1537–1549.

[25] GOLOVINSKIY, A., KIM, V. G., AND FUNKHOUSER, T. Shape-based recognition of 3D point clouds in urban environments. In *IEEE International Conference on Computer Vision (ICCV)* (2009), pp. 2154–2161.

[26] HAMARNEH, G., AND GUSTAVSSON, T. Combining snakes and active shape models for segmenting the human left ventricle in echocardiographic images. *Computers in Cardiology* (2000), 115–118.

[27] ISENBURG, M., LIU, Y., SHEWCHUK, J. R., AND SNOEYINK, J. Streaming computation of delaunay triangulations. *ACM Transactions on Graphics 25*, 3 (2006), 1049–1056.

[28] J., S., M., O., AND J., G. L. A concise and provably informative multi-scale signature based on heat diffusion. *Computer Graphics Forum* (2009).

[29] JAMES, D. L., AND TWIGG, C. D. Skinning mesh animations. *ACM Transactions on Graphics 24*, 3 (2005), 399–407.

[30] JIANG, C., ZHANG, X., HUANG, W., AND MEINEL, C. Segmentation and quantification of brain tumor. In *IEEE International Conference on Virtual Environments, Human-Computer Interfaces, and Measurement Systems* (2004).

[31] KADIR, T., AND BRADY, M. Unsupervised non-parametric region segmentation using level sets. In *Proceedings of IEEE International Conference on Computer Vision* (2003), vol. 2, pp. 1267–1274.

[32] KARNI, Z., AND GOTSMAN, C. Spectral compression of mesh geometry. In *ACM SIGGRAPH* (New York, NY, USA, 2000), pp. 279–286.

[33] KARNI, Z., AND GOTSMAN, C. 3D mesh compression using fixed spectral bases. *Graphics Interface* (2001), 1–8.

[34] KASS, M., WITKIN, A., AND TERZOPOULOS, D. Snakes: Active contour models. *International Journal of Computer Vision 1* (1988), 31–43.

[35] KAUS, M. R., WARFIELD, S. K., NABAVI, A., BLACK, P. M., JOLESZ, F. A., AND KIKINIS, R. Automated segmentation of mr images of brain tumors. *Radiology 218* (2001), 586–591.

[36] KICHENASSAMY, S., KUMAR, A., OLVER, P., TANNENBAUM, A., AND YEZZI, A. Conformal curvature flows: From phase transisitions to active vision. *Archive for Rational Mech. and Anal. 134* (1996), 275–301.

[37] KUHNIGK, J., DICKEN, V., BORNEMANN, L., BAKAI, A., WORMANNS, D., KRASS, S., AND PEITGEN, H. Morphological segmentation and partial volume analysis for volumetry of solid pulmonary lesions in thoracic ct scans. *IEEE TMI* (2006).

[38] LEE, C. H., VARSHNEY, A., AND JACOBS, D. W. Mesh saliency. In *SIGGRAPH* (2005).

[39] LEVENTON, M., AND E. GRIMSON, F. Statistical shape influence in geodesic active contours. In *Computer Vision and Pattern Recognition* (2000), pp. 75–78.

[40] LI, H., ADAMS, B., GUIBAS, L. J., AND PAULY, M. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009) 28*, 5 (December 2009).

[41] LIAO, M., ZHANG, Q., WANG, H., YANG, R., AND GONG, M. Modeling deformable objects from a single depth camera. In *IEEE International Conference on Computer Vision (ICCV)* (2009), pp. 167–174.

[42] LIN, Y., AND CHEN, W. An adaptive speed term based on generalized fuzzy operator for level set segmentation. In *Proceedings of IEEE International Symposium on Biomedical Imaging: Macro to Nano* (2004), vol. 1, pp. 141–144.

[43] LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., RÖSSL, C., AND SEIDEL, H.-P. Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International* (2004), IEEE Computer Society Press, pp. 181–190.

[44] LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. Linear rotation-invariant coordinates for meshes. In *Proceedings of ACM SIGGRAPH 2005* (2005), ACM Press, pp. 479–487.

[45] LIU, Y., LIU, M., KIHARA, D., AND RAMANI, K. Salient critical points for meshes. In *Proceedings of the ACM symposium on Solid and physical modeling* (2007), pp. 277–282.

[46] LORENSON, W., AND CLINE, H. Marching cubes: A high resolution 3D surface reconstruction algorithm. *Computer Graphics 21* (1987), 163–169.

[47] Lyer, N., Kalyanaraman, Y., K.Lou, Janyanti, S., and Ramani, K. A reconfigurable 3D engineering shape search system parti: shape representation. In *DETC* (2003).

[48] Lyer, N., Kalyanaraman, Y., K.Lou, Janyanti, S., and Ramani, K. A reconfigurable 3D engineering shape search system partii: database indexing, retrieval and clustering. In *DETC* (2003).

[49] McInerney, T., and Terzopoulos, D. Topologically adaptable snakes. In *Proceedings of International Conference on Computer Vision* (1995).

[50] McInerney, T., and Terzopoulos, D. Deformable models in medical image analysis: A survey. *Medical Image Analysis 1*, 2 (1996), 91–108.

[51] McInerney, T., and Terzopoulos, D. Topology adaptive deformable surfaces for medical image volume segmentation. *IEEE Transactions on Medical Imaging 18*, 10 (1999), 840–851.

[52] Meyer, M., Desbrun, M., Schröder, P., and Barr, A. H. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, Heidelberg, 2003, pp. 35–57.

[53] Museth, K., Breen, D., Zhukov, L., and Whitaker, R. Level-set segmentationfrom multiple non-uniform volume datasets. In *Proceedings of IEEE International Conference on Visualization* (2002), pp. 179–186.

[54] ODonnell, T., Boult, T., Fang, X., and Gupta, A. The extruded generalized cylinder: A deformable model for object recovery. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (1994), pp. 174–181.

[55] Palmer, S. E. Hierarchical structure in perceptual representation. *Cognitive Psychology 9* (1977), 441–474.

[56] PARAGIOS, N. A level set approach for shape-driven segmentation and tracking of the left ventricle. *IEEE Transactions on Medical Imaging 22*, 6 (2003), 773–776.

[57] PENG, J., KIM, C., AND KUO, C. Technologies for 3D mesh compression: A survey. *Journal of Visual Communication and Image Representation (JVCIR) 16*, 6 (December 2005), 688–733.

[58] REUTER, M., WOLTER, F.-E., AND PEINECKE, N. Laplace-beltrami spectra as ”Shape-DNA” of surfaces and solids. *Computer-Aided Design 38*, 4 (2006), 342–366.

[59] RONG, G., CAO, Y., AND GUO, X. Spectral mesh deformation. *The Visual Computer 24*, 7-9 (2008), 787–796.

[60] SHAMIR, A., SHAPIRA, L., AND COHEN-OR, D. Mesh analysis using geodesic mean-shift. *The Visual Computer 22*, 2 (2006), 99–108.

[61] SHI, J., AND MALIK, J. Normalized cut and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 8 (2000).

[62] SHILANE, P., AND FUNKHOUSER, T. Distinctive regions of 3D surfaces. *ACM Transactions on Graphics* (2007).

[63] SMOLIC, A., SONDERSHAUS, R., STEFANOSKI, N., VASA, L., MULLER, K., OSTERMANN, J., AND WIEGAND, T. A survey on coding of static and dynamic 3d meshes. *Three Dimensional Television-Capture, Transmission, Display* (2008), 239–312.

[64] SOLOMON, J., BUTMAN, J. A., AND SOOD, A. Data driven brain tumor segmentation in mri using probabilistic reasoning over space and time. In *MICCAI* (2004), vol. 1, pp. 301–309.

[65] SORKINE, O., AND COHEN-OR, D. Least-squares meshes. In *Proceedings of Shape Modeling International* (2004), IEEE Computer Society Press, pp. 191–199.

[66] SUMNER, R. W., AND POPOVIC, J. Deformation transfer for triangle meshes. In *ACM SIGGRAPH* (2004), pp. 399–405.

[67] SUNDAR, H., SILVER, D., GAGVANI, N., AND DICKINSON, S. Skeleton based shape matching and retrieval. In *SMI* (2003), pp. 130–139.

[68] SURI, J. Leaking prevention in fast level sets using fuzzy models: An application in mr brain. In *Proceedings of IEEE International Conference on EMBS* (2000), pp. 220–225.

[69] TAM, G. K., AND LAU, R. W. Deformable model retrieval based on topological and geometric signatures. *IEEE Transactions on Visualization and Computer Graphics 13*, 3 (2007), 470–482.

[70] TANGELDER, J. W., AND VELTKAMP, R. C. A survey of content based 3D shape retrieval methods. In *Proceedings of the Shape Modeling International* (2004), vol. 0, pp. 145–156.

[71] TERZOPOULOS, D., AND FLEISCHER, K. Deformable models. *The Visual Computer 4*, 6 (1988), 306–331.

[72] TSAI, A., YEZZI, A. J., WELLS, W., TEMPANY, C., TUCKER, D., FAN, A., GRIMSON, W., AND WILLSKY, A. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Transactions on Medical Imaging 22*, 2 (2003), 137–154.

[73] VLASIC, D., BARAN, I., MATUSIK, W., AND POPOVIC, J. Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics 27*, 3 (2008), 1–9.

[74] WACHSMUTH, E., ORAM, M. W., AND PERRETT, D. I. Recognition of objects and their component parts: reponses of single units in the temporal cortex of the macaque. *Cerebral Cortex 4* (1994), 509–522.

[75] WANG, H., AND LI, H. Cartoon motion capture by shape matching. In *Proceedings of 10th Pacific Conference on Computer Graphics and Applications* (2002), pp. 454–456.

[76] WANG, J., LI, P., AND WU, B. An improved approach to image segmentation based on mumford-shah model. In *Proceedings of IEEE International Conference on Machine Learning and Cybernetics* (2006), pp. 3996–4001.

[77] WANG, T., CHENG, I., AND BASU, A. Fully automatic brain tumor segmentation using a normalized gaussian bayesian classifier and 3d fluid vector flow. In *2010 17th IEEE International Conference on Image Processing (ICIP)*.

[78] WESSEL, R., NOVOTNI, M., AND KLEIN, R. Correspondences between salient points on 3D shapes. In *proceedings of Vision, Modeling, and Visualization* (2006), pp. 365–372.

[79] WHITAKER, R., BREEN, D., MUSETH, K., AND SONI, N. Segmentation of biological volume datasets using a level-set framework. *Volume Graphics* (2001), 249–263.

[80] YAN, P., AND KASSIM, A. Medical image segmentation using minimal path deformable models with implicit shape priors. *IEEE Transactions on Information Technology in Biomedicine 10*, 4 (2006), 677–684.

[81] YEZZI, A., TSAI, A., AND WILLSKY, A. A fully global approach to image segmentation via coupled curve evolution equations. *Journal of Viausl Communication and Image Representation 13* (2002), 195–216.

[82] YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. Mesh editing with poisson-based gradient field manipulation. *ACM TRANS. GRAPH 23* (2004), 644–651.

[83] YUI, S., HARA, K., ZHA, H., AND HASEGAWA, T. A fast narrow band method and its application in topology-adaptive 3-d modeling. In *Preceedings of IEEE International Conference on Pattern Recognition* (2002), vol. 4, pp. 122–125.

[84] ZHANG, D. Statistical part-based models: Theory and applications in image similarity, object detection and region labeling. *PhD Thesis Graduate School of Arts and Sciences, Columbia University* (2005).

[85] ZHANG, D.-Q., AND CHANG, S.-F. Detecting image near-duplicate by stochastic attributed relational graph matching with learning. In *ACM Multimedia* (2004), pp. 877–884.

ABSTRACT

DEFORMABLE MESHES FOR SHAPE RECOVERY: MODELS AND APPLICATIONS

by

CHANG LIU

December 2011

Advisor: Dr. Jing Hua

Major: Computer Science

Degree: Doctor of Philosophy

With the advance of scanning and imaging technology, more and more 3D objects become available. Among them, deformable objects have gained increasing interests. They include medical instances such as organs, a sequence of objects in motion, and objects of similar shapes where a meaningful correspondence can be established between each other. Thus, it requires tools to store, compare, and retrieve them. Many of these operations depend on successful shape recovery. Shape recovery is the task to retrieve an object from the environment where its geometry is hidden or implicitly known. As a simple and versatile tool, mesh is widely used in computer graphics for modelling and visualization. In particular, deformable meshes are meshes which can take the deformation of deformable objects. They extend the modelling ability of meshes. This dissertation focuses on using deformable meshes to approach the 3D shape recovery problem.

Several models are presented to solve the challenges for shape recovery under different circumstances. When the object is hidden in an image, a PDE deformable model is designed to extract its surface shape. The algorithm uses a mesh representation so that it can model any non-smooth surface with an arbitrary precision compared to a parametric model. It is more computational efficient than a level-set approach. When the explicit geometry of the object

is known but is hidden in a bank of shapes, we simplify the deformation of the model to a graph matching procedure through a hierarchical surface abstraction approach. The framework is used for shape matching and retrieval. This idea is further extended to retain the explicit geometry during the abstraction. A novel motion abstraction framework for deformable meshes is devised based on clustering of local transformations and is successfully applied to 3D motion compression.

AUTOBIOGRAPHICAL STATEMENT

CHANG LIU

Chang Liu is a PhD candidate in Computer Science at Wayne State University, where he is also a research assistant in the Graphics and Imaging Laboratory. He received his BE degree (2003) in Electronics and Information Engineering and ME degree (2005) in Pattern Recognition and Intelligent System, from Huazhong University of Science and Technology (HUST), Wuhan, China. His research interests include shape retrieval, visualization, geometric processing, medical imaging.