The Medial Scaffold for 3D Shape Modeling and Recognition

by
Ming-Ching Chang
B. S., National Taiwan University, Taipei, Taiwan, 1996
M. S., National Taiwan University, Taipei, Taiwan, 1998

A dissertation submitted in partial fulfillment of the
requirements for the Degree of Doctor of Philosophy
in the Division of Engineering at Brown University

Providence, Rhode Island
October 2008

This dissertation by Ming-Ching Chang is accepted in its present form by the Department of Engineering as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date _____ _____

Benjamin B. Kimia, Director
Division of Engineering

Recommended to the Graduate Council

Date _____ _____

Gabriel Taubin, Reader
Division of Engineering

Date _____ _____

Frederic Fol Leymarie, Reader
Department of Computing, Goldsmiths College, University of London, UK

Approved by the Graduate Council

Date _____ _____

Sheila Bonde
Dean of the Graduate School

Curriculum Vitae

Ming-Ching Chang was born on the 22nd of June, 1974, in Chung-Li, Taiwan. From 1989 to 1992, he attended the Taipei Municipal Chien-Kuo High School, the most prestigious high school in Taiwan. For his undergraduate education, he attended the National Taiwan University (NTU) in Taipei, where he majored in Civil Engineering. He is interested in computer programming and published a book on object-oriented programming in C++ in 1994. In 1996, He attended the Graduate School of Computer Science and Information Engineering at NTU and graduated with a M.S. degree in 1998. Simultaneously, he worked as a research assistant on several automatic visual inspection and pattern identification projects in the Mechanical Industry Research Laboratories department within the Industrial Technology Research Institute (ITRI), the leading national research organization in Hsin-Chu, Taiwan. Then, he worked as a full-time teaching assistant in the Computer Center at the College of Management in NTU. Since 2001, he has been working at the Laboratory for Engineering Man/Machine Systems (LEMS) in the Division of Engineering at Brown University. Though he has primarily been working on 3D shape modeling and recognition problems at LEMS as a research assistant, he has also been a teaching assistant for a number of graduate and undergraduate courses in computer vision. Additionally, he has published several papers in leading national and international conferences in computer vision and graphics. His research interests include computer vision and medical imaging, mainly in the topics of shape representation, modeling, visualization, registration, and recognition.

Acknowledgements

First and foremost, I would like to thank my parents, Sheng-Chu Chang and Lan-Teng Yu, to whom I dedicate this thesis, for their infinite love and support through the thirty years of my life as a student. From very early on, they encouraged me to set high goals and to strive to reach them. I owe it to them for instilling in me the desire to be all I can be. I would also like to thank my grandfather, grandmother, aunts and uncles, brother, and cousins. My positive personality traits are products of your love; I am very lucky to have been born into such a lovely family.

Next, I would like to thank my advisor, Prof. Benjamin Kimia, for his continual guidance and support throughout the years of my graduate study. I will especially cherish the one-on-one meetings where he made me "think abstractly" and "communicate effectively" while journeying through the realm of research. Above all, I am grateful for all the insight he gave me, my favorite being to see and think through all the problems that I could encounter in my life and to find a way out before the fact.

I would also like to thank Prof. Gabriel Taubin, Prof. Joseph Mundy, and Prof. David Cooper for their advice, guidance, and for their stimulating classes. I would like to thank them for providing valuable comments about my research, being on my progress review committee, and being on my thesis committee. I would especially like to thank Prof. Frederic Fol Leymarie as a mentor and a friend, who provided an immense amount of helpful guidance regarding my research in addition to being on my thesis committee.

Next I would like to thank all my colleagues at LEMS, Thomas and Amir, for their kindness and friendship from the first day I arrived at Brown, Andrew, Kong-bin, Dongjin, Vishal, John, Ozge, Matt, Dan, Nhon, Ricardo, Yong, Shubao, Ibo, Can, Doug, Eduardo, Isa, Fatih, and everyone else in the group, for making the journey interesting and fun. I am grateful for all the stimulating discussions, the friendship and social interactions, and the enriching international culture.

Next, I would like to thank the Taiwanese community at Brown, especially Mei-Hsiu and Peter, Chun-Chia and Phoebe, Ping, and everyone else, who embraced me with open arms since my first day at Brown. I would especially like to thank Roger, Chih-Hsun and Chia-Hsuan, Chia-En, Tai-Sen, Doug, Feng-Hao, Ming-Li, Lenora, Eunice, and everyone else, who declared me as their "godfather" and who created a community, a home away from home at Brown. Lastly, I would like to thank my house-mate, Sirod, for his friendship and help since the beginning of my stay.

Finally I would like to thank my wife, Yu-Ju, for her undivided love throughout all these years and for her continual encouragement and support. I would especially like to thank her for leaving home half a world away to start a new life with me, and for the years of commuting to and from work in order to spend every sweet night with me. I owe her everything and hope to reciprocate her devotion to me for the rest of my life.

Abstract

In this thesis, we develop the *medial scaffold* ($\mathcal{MS}$) technology, a graph-based medial axis ($\mathcal{MA}$) representation, which is applied to model 3D shapes in a range of applications and lead toward the ultimate goal of recognition. The $\mathcal{MA}$ is with great promise as a universal model for shape, since it outlines many features explicitly in a hierarchy and allows to organize them completely (enabling an exact reconstruction). A key issue in using the 3D $\mathcal{MA}$ is that, how the complex $\mathcal{MA}$ structure can be regularized so that similar, within-category 3D shapes yield similar 3D $\mathcal{MA}$ that are distinct from the non-category shapes. My work extends a line of research which *(i)* organizes the 3D $\mathcal{MA}$ into a hierarchical hypergraph by studying its singularity typology, and *(ii)* classifies the instabilities of the $\mathcal{MA}$ structure, or *transitions* (sudden topological changes due to a small perturbation). A set of $\mathcal{MS}$ *transforms* is defined in a case-by-case analysis to model the $\mathcal{MS}$ across all generic transitions, thus allows manipulating the underlying shape and regularizing it toward a near-by degenerate transition point. The simplified $\mathcal{MS}$ preserves with-in-category similarity, thus enables various applications including shape analysis, robust feature detection, and manipulation. It is also useful in establishing a similarity measure in matching shapes, by adopting a graph matching scheme for the $\mathcal{MS}$ hypergraph. Results of the proposed framework demonstrate its state-of-art in $\mathcal{MA}$-based modeling and promises its potential in shape-based matching.

Contents

Chapter 1

Introduction

1.1 A Brief Overview of this Thesis

* Problem setup: 3D shape representation for modeling and recognition.

Many fields of technologies such as computer vision and graphics require sophisticated interaction with objects using some **representation** of their shapes [125]. In shape modeling, typical tasks include surface meshing, detecting locally salient features (such as high curvature points) and globally salient features (such as elongated axes, necks, limbs, and other primitive regions), as well as the parametrization and compression of the model. The seeking of a suitable representation to "process" the underlying shape is a key issue in these problems, and it is closely related to the **matching** between shapes and toward the ultimate goal of *recognition* [136]. This thesis mainly focuses on three-dimensional (3D) shapes, where the analysis is a generalization of a simpler two-dimensional (2D) cases.

* What is "shape"?

A **shape** could be defined as an object's physical surface, which is *simple*, *continuous*, and *enclosing* the object's volume [136]. A more general definition is all geometric information of an object invariant to Euclidean transforms (*e.g.*, translation and rotation [1]) [112]. In this generalized view, a shape need not necessarily enclose a volume, thus making it more useful in considering a *sampled* object under discrete observation in practice. For example, an edge map in 2D or a set of scan points in 3D contains sufficient geometric information pertinent to the objects. Koenderink considers shapes as being the operational probing of the data (which can be the outline) rather than the outline itself [118]. In this context, the "processing" of the outline of an object, such as edge linking/grouping in 2D and the meshing of scan points in 3D can be viewed as processing one object data to retrieve its "shape". Leyton [131] further considers shape as the dynamics of deformation, *i.e.*, the history that links one outline to another: shape is "memory". A compendium of traditional and alternative viewpoints is presented in a recent book chapter by Leymarie [129].

* Representing shapes to measure "similarity".

The shape **representation** is about *how* and *in what way* the information of a shape is processed and compared. Typically, the geometry of a shape is given as a set of sample points or a polygonal mesh, commonly available from 3D scanners or in graphics. While the geometry of shape is exactly

[1] The *scaling* is not considered as an invariant property in the *object*-centered approach, since shapes of different sizes are not equivalent; however, this might as well be included as an invariant property in the *viewer*-centered approach.

stored, these primitive forms are not capable for further *understanding* of shape, *e.g.*, in the context of distinguishing similar shapes from non-similar ones. In recognition, two shapes within a category should be similar when evaluated under some *similarity measure*, and distinct shapes should have large dissimilarity. How to achieve a suitable similarity measure between shapes is an intricate and open problem [70]. The key question is, what is the suitable representation for shapes and how to match them using such representation? A typical solution is to extract a shape *descriptor* from the shapes, usually with a great deal of simplification, to enable efficiency. The choice of the descriptor is often domain-specific and usually requires manual tuning to achieve a better *discrimination* performance. In the above context, the shape descriptor is a *reduced* form of representation in matching shapes.

* Shape representations suitable for recognition.

The search for a suitable representation to effectively match shapes is a challenging task, still under active investigation. Many representations (shape descriptors) have been proposed, which can be roughly classified into three categories: the *(1)* feature-based, *(2)* view-based, and *(3)* graph-based representations. (Refer to § 2.1 for more details.) [2]

The *feature*-based methods are the current mainstream, where a shape feature (signature) is extracted to describe the (3D) shape and distinguish it from others. The feature vector could be either local or global. Typical problems of feature-based methods are: *(i)* the representation is not *complete*, in that a shape is not *reconstructible* from the set of feature vectors; *(ii)* the *coherence* of shape is not preserved; *(iii)* the representation is not perceptually intuitive; and *(iv)* there is no notion of the interior of the shape.

In contrary, the *view*-based methods represent the 3D objects by using a set of distinct 2D views (appearance) and match the views instead of matching the object. While there is no agreement on whether the view-based or object-centered representation is more suitable for the recognition task [198], the viewpoint-dependent representation is not intrinsic and nor complete (*e.g.*, the number of views to sufficiently describe an 3D object is not know *a priori* in general).

Finally, the *graph*-based methods aim at extracting a graph-like structure out of the shape, thus enabling to analyze the shape by components and the relation between the components. The usual difficulty is that a graph-like structure is hard to define and extract from a shape. As a next step, how to effectively compute and *stabilize* such graph representation is the next bottleneck.

* Lack of a suitable "generic" representation.

While domain-specific shape descriptors can be used to distinguish shapes, they are limited by their representative capability and reaching a bottleneck in recognition. In addition, shape descriptors vary largely from one to another, both in the philosophy and overall matching framework. Their development is facing a *dilemma* that the representation is either too *rough*—ignoring too much information of the shape, or too *complex*—such that the representation is too redundant and unstable. A crucial topic in shape recognition is that whether or not a "generic" representation can be defined to recognize shapes. Several criteria have been proposed that a preferred shape representation ought to have [108, 141]:

1. *Scope*: able to describe all shapes.
2. *Uniqueness*: one-to-one mapping from representations to shapes.

[2] The graph-based representations can be viewed as feature-based or view-centered representation with the information organized in a structural way *e.g.* as a graph. Other classifications have been proposed. Methods 'hybrid' in nature are classified to the most relevant category.

3. *Stability*: stable to tiny changes of the shape.
4. *Sensitivity* (discriminative power): able to capture details and subtle features.
5. *Efficiency*: easy to compute and compare.
6. *Multi-scale support*: hierarchical, coarse-to-fine that describes both global (coarse) and local (fine) features.
7. *Local support*: can be computed and compared locally (to describe partial shapes).

Our goal is to seek to a general shape representation to *exploit the* full *shape information, which could be preferably organized structurally and coupled* intrinsically *with the shape, such that the* qualitative *structure of shape can be modeled, to achieve a better recognition rate.* On the other hand, as the problem of matching rigid closed shapes is generalized into matching *partial* or articulated shapes, how to extract the qualitative structure of shape remains an important issue.

The development of a proper shape representation involves several other issues. First, the representation closely depends on the *input*. Typically the input is in a form of sampled points, polygonal meshes, or volumetric voxels in 3D (which could contain a certain noise/outliers). Second, the shape representation can be either *local* or *global* depending on the allowable shape *topology*. A local representation allows to handle partial shapes (with boundary), while a global representation assumes the shape is a 'solid' (closed surface enclosing a volume). Third, the representation could be either *complete* or not, whether it is sufficient to fully reconstruct the shape. A complete representation enables an one-to-one mapping of the shape to the representation and thus could intuitively increase the recognition/discrimination capability, if its large amount of information can be well organized.

* Medial axis to extract the *qualitative* representation of shapes.

A major branch in shape representation is the symmetry-based **Medial Axis** ($\mathcal{MA}$), which has been long considered with great promise as an universal model for shape. The $\mathcal{MA}$ of a shape is the closure of centers of maximal spheres that are at least tangent to the surface at two places. Typically, three medial branches intersect at a junction point, and these junction points together with the end points of medial curves can be organized into a graph structure with nodes and links, Figure 1.1. The $\mathcal{MA}$ can also be viewed as the singularities (collisions) of the quenching wavefronts originated from the shape boundary in a *"grass-fire"* mode [31], which induces a notion of *'flow'* on it. The $\mathcal{MA}$ is a *generic* representation addressing many desirable characteristics listed above [125]:

1. It is intuitively appealing in representing the essence structure of a shape such as *elongated* and *branching* objects; the generalized axis are explicitly captured (*qualitative* description).
2. A radius function along the $\mathcal{MA}$ trace encodes the varying width of the shape (*quantitative* description); the structural information is robustly preserved along the axis.
3. Important shape *features* such as curvature extrema (ridges, corners) [130, 176], necks and limbs [181], thin/thick parts are made explicit.
4. A *hierarchy* of scales is built-in via the combination of spatial and width properties, *i.e.*, the *scale* is represented: small features can be distinguished from large ones and ranked accordingly [32] (*coarse-to-fine*).
5. It is *complete* in allowing an exact reconstruction of the shape [86, 215].
6. The $\mathcal{MA}$ provides an intrinsic parametrization of the whole embedded space of the shape, Figure 1.1(e).
7. It provides a powerful framework to model deformation and generate shapes [203, 88, 131, 214] in studying shape *dynamics*.

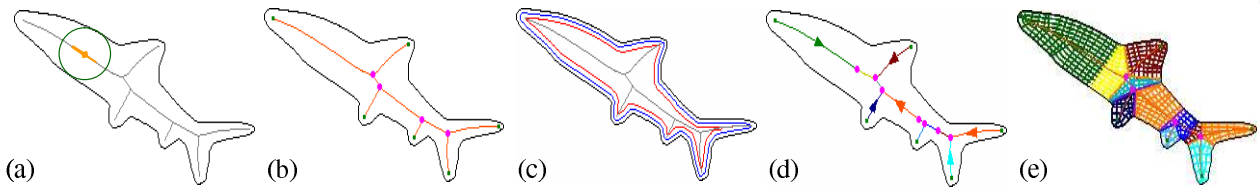


Figure 1.1: (Adapted from [169] and [115, Fig.4].) The $\mathcal{MA}$ and the *shock graph* of a 2D fish shape. (a) The $\mathcal{MA}$ is the centers of maximal spheres bi-tangent to the shape and captures features and structure of the shape. It can be organized into a graph (b) and induced with a notion of *"flow"* by viewing it as collision of wavefronts propagating from the boundary (c), thus being organized into a directed graph, the *shock graph* ($\mathcal{SG}$) in (d). (e) The $\mathcal{MA}$ / $\mathcal{SG}$ is an *intrinsic* representation of the shape, where each point of the embedded space is mapped onto some point on the $\mathcal{MA}$ with a certain radius.

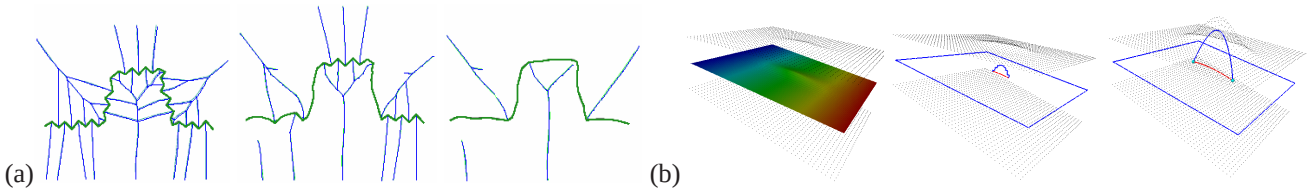


Figure 1.2: (Adapted from [200, (a): Fig.1] and [125, (b): Fig.22].) The *instabilities* of the $\mathcal{MA}$: slight perturbation of the shape induces large structural change in the $\mathcal{MA}$ in 2D (a) and 3D (b).

Despite these advantages, there exist several barriers in using the 3D $\mathcal{MA}$ in shape modeling and matching. First, the structure of the $\mathcal{MA}$ is *not* trivial to characterize and represent, especially in the 3D case with additional dimensionality [12]. Second, the well-known *instability* of the $\mathcal{MA}$ (sensitivity to small perturbations) yet needs to be handled in the more complex scenario. A main contribution of this thesis is to address these issues.

* Use $\mathcal{MA}$ *transitions* to partition the shape space and characterize shape deformations.

We continue the research track of Kimia *et al.* in modeling the $\mathcal{MA}$ to recognize shapes. The following key ideas constitute the philosophy of this thesis in modeling 3D shapes using their $\mathcal{MA}$ in using the $\mathcal{MA}$ to represent shapes and match them:

- *$\mathcal{MA}$ instabilities modeled as transitions.* The $\mathcal{MA}$ is unstable in that slight perturbation of the shape can cause abrupt structural change of the $\mathcal{MA}$, Figure 1.2. Giblin and Kimia analysis this $\mathcal{MA}$ structure change for a sequence of deforming shapes case-by case and classify the $\mathcal{MA}$ topology change into generic types of **transitions** [88]. This set of transitions can be used to model $\mathcal{MA}$ instabilities in all cases.
- *$\mathcal{MA}$ transitions to partition the shape space.* The transitions of the $\mathcal{MA}$ is also useful in separating distinct shapes and grouping similar shapes into categories. Observe that the topology of the $\mathcal{MA}$ does not change for most shape deformations (except at the transitions), the collection of all shapes with the same $\mathcal{MA}$ topology is grouped into a *"shape cell"*. The collection of all shapes which compose of the *"shape space"* is then partitioned into shape cells by the $\mathcal{MA}$ transitions, Figure 1.3(a), where the transition shapes essentially form the *boundary* between shape cells.
- *Transition shapes to characterize a deformation path.* The transition shapes are *perceptually* more significant [115] that they characterize the sequence of discrete events where the $\mathcal{MA}$

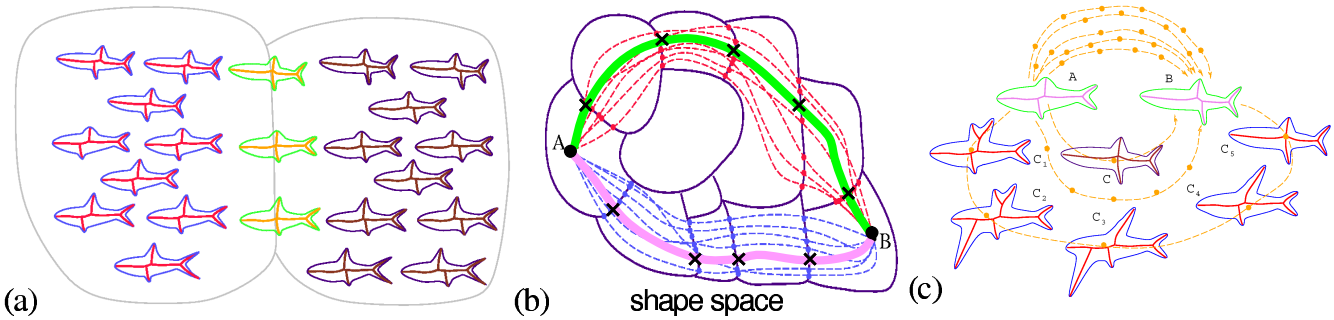


Figure 1.3: (Adapted from [169, Fig.11,5,13].) Partitioning of shape space. (a) The *shape cell* is the collection of shapes with same $\mathcal{MA}$ topology. Transition shapes (green) form the boundary between shape cells. (b) The shape space (collection of all possible shapes) is then partitioned into shape cells, where the *deformation path* between arbitrary two shapes A to B across numerous cells is then characterized by a set of discrete events of $\mathcal{MA}$ transitions (black crosses). (c) Two deformation paths among many are highlighted. The sequence $(\mathsf{A}, \mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_3, \mathsf{C}_4, \mathsf{C}_5, \mathsf{B})$ is not optimal since features are first added and then removed. The pair of "simplifying" deformation paths starting from A and B, respectively, and leading to a common shape C is optimal.

structure changes as the shape is deforming. Consider the path of deforming an arbitrary shape A to shape B (which navigating the shape space) in Figure 1.3(b), the deformation path is essentially partitioned by the set of transitions into a set of *piecewise-continuous* segments across several shape cells, where within each shape cell the $\mathcal{MA}$ topology remains constant [169].

- *Optimal deformation as a pair of simplifying deformation paths.* Consider the set of all possible deformation paths between two arbitrary shapes, the path with *complexity increasing* deformation is clearly not optimal, since features are first added (to make complex the shape) and then removed, Figure 1.3(c). The optimal deformation path is then among the ones where two shapes A and B are both *simplified* toward a common shape C with the least cost. The cost of the optimal deformation path is crucial and can be used to define a *metric* in matching shapes. For 2D shapes, it can be found by restricting the search of the "simplifying" edits of the $\mathcal{MA}$ (across transitions), Figure 1.4, by adopting an edit-distance algorithm [169]. [3]

* Goal: regularize the $\mathcal{MA}$ toward a qualitative representation to model and match 3D shapes.

The goal of this thesis is to develop a *generic qualitative* representation of 3D shapes by modeling the $\mathcal{MA}$. We aim to address the following issues:

- *On computing a graph-like representation of the 3D $\mathcal{MA}$.* Leymarie and Kimia [125] have organized the 3D $\mathcal{MA}$ into a hypergraph structure—the **medial scaffold** ($\mathcal{MS}$), based on a study of the local form of the 3D $\mathcal{MA}$ [87]. A major goal of this thesis is to *implement* the $\mathcal{MS}$ and address the computational issues. Specifically, we explicit develop computational representation to model the inter-connectivity between the medial sheets, curves, and nodes in the $\mathcal{MS}$ hypergraph and organize their topology and geometry into a hierarchical structure.

[3] The cost of this optimal path defines the dissimilarity between two shapes and is used to index into a database of shapes with excellent recognition rates: for a 1032 shape database the recognition rate is 97%, remaining impressively flat in the precision-recall curve [169].

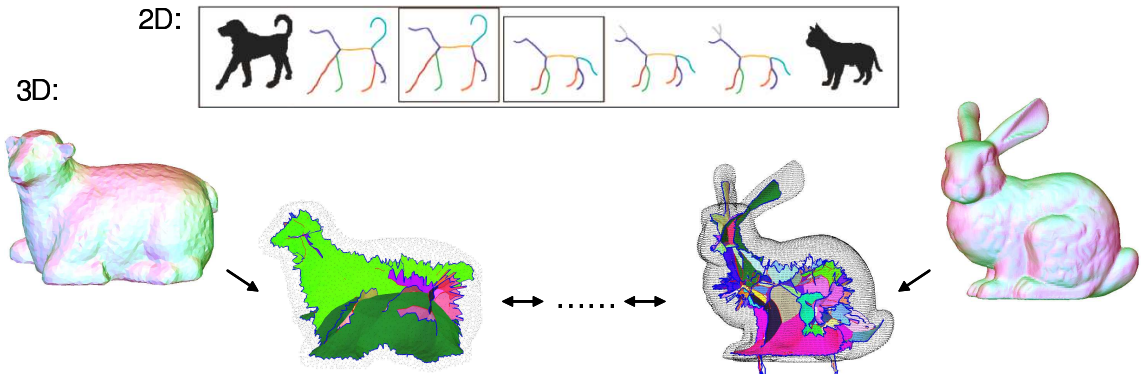


Figure 1.4: An illustrative example in matching the $\mathcal{MA}$ of 2D shapes (upper, adapted from [169, Fig.23]) and 3D shapes (bottom). The simplifying $\mathcal{MA}$ graphs with the same topology in the middle suggest an optimal deformation path to define a *similarity metric* between two shapes in 2D. Our goal is to do the same in 3D.

- *On regularizing the 3D $\mathcal{MA}$ toward a qualitative representation.* The second goal is to remove the $\mathcal{MS}$ instabilities to extract a qualitative structure of the shape. We define a set of **transforms** ("edit" of the $\mathcal{MA}$) to make equivalent the $\mathcal{MA}$ across transitions and use them to regularize the $\mathcal{MA}$, Figure 1.5(a).
- *On matching 3D shapes.* We seek for a matching scheme to recognize objects by matching their qualitative $\mathcal{MA}$ structure. Specifically, we adopt a graph matching scheme to match the $\mathcal{MS}$ and define the similarity between the shapes, Figure 1.5(b).
- *In the scope (of what types of shapes to handle).* We aim to take general inputs such as the *unorganized points* or *partial* meshes. [4] We do not restrict the input shape topology *e.g.*, they can be surfaces with boundary or *non-manifold* surfaces; and we do not impose strict assumption on the sampling condition of the point cloud (see Chapter 6 for details). This makes easy to convert shapes from other representation to ours.

* Potential applications.

The qualitative structure of the $\mathcal{MS}$ addresses the key bottleneck of the $\mathcal{MA}$ thus unlocks a wide range of applications. *(i)* In **shape modeling**, the 'tip' of the $\mathcal{MA}$ corresponds to salient surface features such as the *ridges* [100] and corners of the shape. The $\mathcal{MS}$ is useful in detecting other features such as the flat and tubular regions (generalized cylinders) [96]. This qualitative skeleton is useful in morphing of shapes in animation [219]. It is also useful in surface meshing (Chapter 6), simplification [194], and segmentation of shapes [159, 62]. *(ii)* In **shape matching**, the $\mathcal{MA}$ is promising for recognition [169, 183], retrieval [183, 106], and registration of shapes [44], in that: *(a)* it allows to organize the abundant shape information hierarchically and intrinsically in a graph-like topology [125], which enables matching parts of deformed shapes naturally, and *(b)* such information captured with the $\mathcal{MA}$ is *complete* in that a full shape reconstruction is always possible [86]. Refer to Chapter 10 for a detailed survey in the applications.

[4] The partial mesh can be viewed as augmenting the sample points with a certain connectivity information between them (into a mesh-like structure on top of the points).

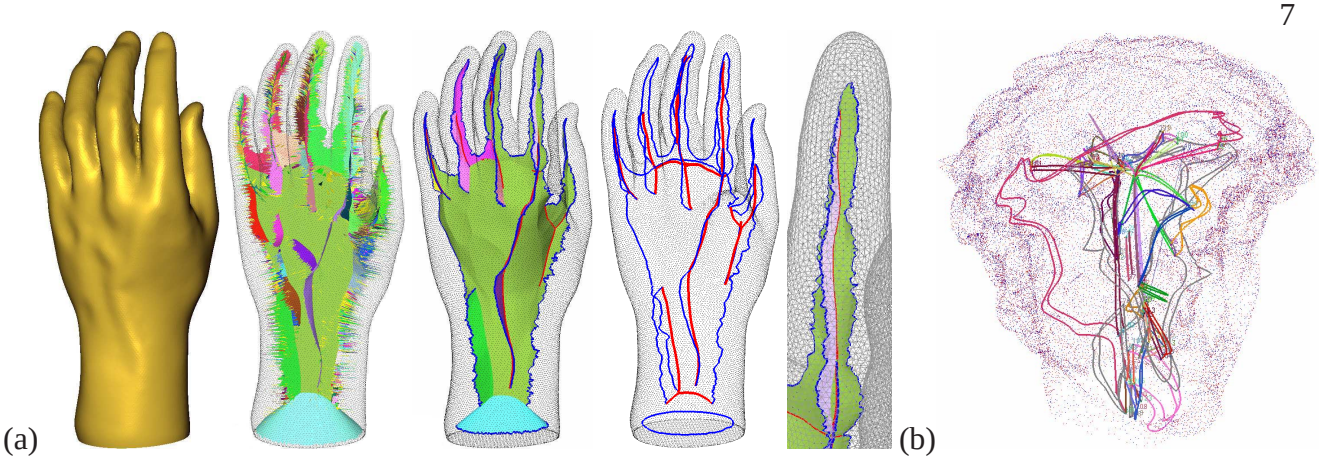


Figure 1.5: Regularized $\mathcal{MS}$ for shape modeling and matching. (a) Regularizing the $\mathcal{MS}$ of a scan of a hand ($\approx$ 38k points, from Polhemus) toward simplification, together with a re-meshed surface ($\approx$ 76k faces) [43]. The initial noisy $\mathcal{MS}$ ($9,574$ sheets) is greatly simplified via a series of *transforms* into a *hypergraph* of only 10 sheets, 33 curves, and 23 nodes. The regularized $\mathcal{MS}$ retains the qualitative structure of the shape, which can be further reduced into a *graph* form (with the sheets implicit). The fore finger is zoomed in the last picture. (b) (From [44, Fig.1].) Graph matching of the $\mathcal{MS}$ structures of two scans of David's head (data from Stanford), matching curves shown in colors) indicates the application in shape registration and recognition [44].

1.2 From Medial Axis to Medial Scaffold: Computation, Modeling, and Regularization

* Difficulties in modeling (processing) the 3D $\mathcal{MA}$.

Compared to the well-studied 2D case, the extraction of a stable, structural, and simplified 3D $\mathcal{MA}$ encounters *three* significant issues: *(i)* How to extract medial axes accurately from input shapes (in the form of unorganized points or polygonal meshes)? *(ii)* How to effectively handle the complex *inter-connectivity* between medial sheets, curves, and nodes (considering the large varieties of them in topology and geometry)? *(iii)* The omnipresent *instabilities* (slight perturbation inducing large changes in the $\mathcal{MA}$) become worse, when the 3D shapes are approximated (in sample points or surface meshes), inevitably producing errors. These issues affect largely how to design a computational algorithm to model the $\mathcal{MA}$.

* From the *medial axis* to the *medial scaffold* ($\mathcal{MS}$).

The modern definition of the $\mathcal{MA}$ originated from Blum [31] has been evolved into a hierarchial *"shocks"* structure in the last decade, Figure 1.7. Giblin and Kimia have studied the local form of the 3D $\mathcal{MA}$ and classified them into one type of $\mathcal{MA}$ sheet, two types of $\mathcal{MA}$ curves, and two types of isolated $\mathcal{MA}$ points [87]. This classification has been used by Leymarie and Kimia [125] to organize the 3D $\mathcal{MA}$ components (sheets, curves, and isolated points) hierarchically into a hypergraph form and propose the notion of a *medial scaffold* ($\mathcal{MS}$), Figure 1.7. [5] A key insight on organizing the $\mathcal{MA}$ toward to $\mathcal{MS}$ is two-fold: *(i)* in preserving all important properties of the $\mathcal{MA}$ (structural, coarse-to-fine, and complete) and *(ii)* enabling one to regularize its structure to remove the instabilities and preserve the "true" qualitative structure of the shape.

[5] The term "scaffold" is used in analogy to building constructions, where a set of metallic beams supports relatively weaker materials, so as to indicate the relative significance of (medial) curves over (medial) sheets, and (medial) nodes over curves, in describing the *qualitative* structure of the 3D $\mathcal{MA}$.

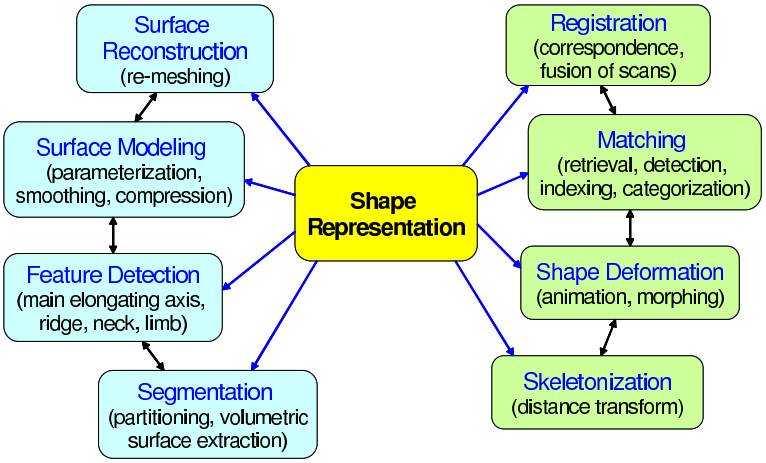


Figure 1.6: Shape *representation* is the key for a large range of 3D shape modeling and recognition problems.

* From the *medial scaffold* toward the *shock scaffold* ($\mathcal{SC}$).

While the $\mathcal{MS}$ handles the topological structure of the 3D $\mathcal{MA}$, it can be further refined according to additional shock flow analysis. The **shock scaffold** ($\mathcal{SC}$) is the refinement of the $\mathcal{MS}$ into a *directed* hypergraph, to classify each sheet and curve segment into regions of monotonic shock flows. While the $\mathcal{MS}$ can be viewed as an extension of the 2D $\mathcal{MA}$ graph in Figure 1.1(b), the shock scaffold is then the extension of the 2D shock graph in Figure 1.1(c). Leymarie and Kimia have classified the 3D shock points into *18* types of shock sources, relays, and sinks [125] (detailed in § 3.1.2). However, a complete topological structural analysis similar to what has been done for the $\mathcal{MS}$ (Chapter 4) remains unexplored. This requires a complete shock flow analysis on the interior of shock sheets to extract a flow network (the *surface network*, which needs to be investigated (§ **??**). [6]

* $\mathcal{MA}$ *transitions* to model the $\mathcal{MA}$ instabilities.

The omnipresent *instabilities* of the $\mathcal{MA}$ occurs when a shape is slightly perturbed as the structure of the $\mathcal{MA}$ changes abruptly, Figure 1.2. Giblin and Kimia have studied the $\mathcal{MA}$ structural changes of a sequence of deforming shapes under a *one-parameter* family of deformations [7] and classify the $\mathcal{MA}$ topology changes into generic cases, which are referred to as the $\mathcal{MA}$ **transitions**. In 2D, there are *six* generic $\mathcal{MA}$ transitions (Figure 5.1 [89]) and in 3D there are *seven* generic $\mathcal{MA}$ transitions for simple closed shapes (Figure 1.8 [88]), detailed in § 5.1.

The significance of $\mathcal{MA}$ transitions is that all use of the $\mathcal{MA}$ structure must explicitly or implicitly handle the transitions/instabilities. In our approach, we *explicitly* exploit the $\mathcal{MA}$ transitions to relate shapes: *shapes immediately across $\mathcal{MA}$ transitions are equivalent.* As aforementioned, the $\mathcal{MA}$ transition is useful in partitioning the shape space and characterizing a shape deformation path for recognition (§ 1.1). The $\mathcal{MA}$ transitions is also useful in shape regularization (detailed below), shape modeling [203], smoothing shape [200] and perceptual grouping [109, 196].

* $\mathcal{MS}$ *transforms* to handle the $\mathcal{MA}$ transitions.

[6] We believe the topological structure developed in Chapter 4 can be applied directly to represent the $\mathcal{SC}$, since the refinement (subdivision) of the $\mathcal{MS}$ components does not create new topologies.

[7] A *one-parameter* family of deformation of shapes can be informally defined as a continuous sequence of shapes where each adjacent pair of shapes differs only in a tiny local change, such that the sequence of shapes is a *morphing* from one to the other.

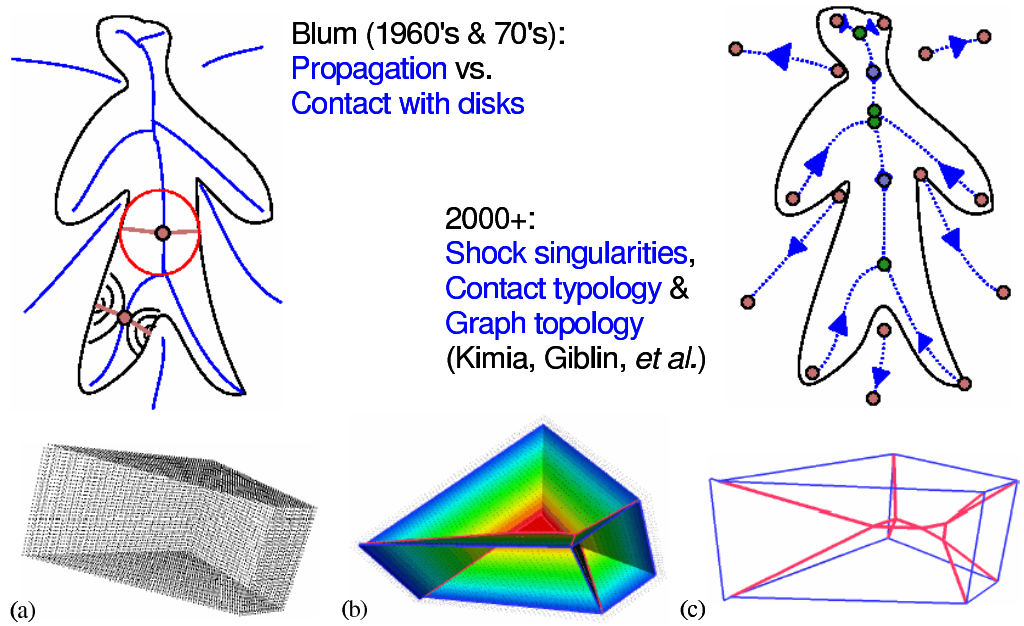


Figure 1.7: (From Leymarie [126, 128].) From the $\mathcal{MA}$ to the $\mathcal{MS}$. Main idea of the Medial Scaffold ($\mathcal{MS}$) and the Shock Scaffold ($\mathcal{SC}$) is to keep only singular points of the 3D $\mathcal{MA}$ radius flow to build a graph [125].

We define a set of $\mathcal{MS}$ **transforms** to explicitly make equivalent shape across transitions. Specifically, as a shape is close to a transition point, we move it **toward** the transition point. We emphasize that we are doing the contrary to other approaches, which try to move *away* from the transition point to ensure stability. For example, Pizer *et al.* fix the topology of their medial representation (the m-Rep) for use in applications such as the segmentation of medical images [188]. In contrary, we believe our approach of explicitly handling the transitions is a strong point in modeling the symmetry. This is based on the idea of Kimia *et al.* [115] that *transition points are perceptually more significant*. To illustrate, observe the shapes in Figure 1.3(a) that a transition shape can be perturbed in many ways around the transition point, while the transition shape itself should be a "characteristic" one to stably represent this "neighborhood" of shapes.

In defining the $\mathcal{MS}$ transform to move a close-to-transition $\mathcal{MS}$ toward a nearby transition point, we observe that in the *seven* generic 3D $\mathcal{MA}$ transitions in Figure 1.8, some involve two ways of deformations while others involve only one, resulting in a total of *eleven* generic $\mathcal{MS}$ transforms. Chapter 5 will explore more on this.

* $\mathcal{MA}$ regularization by applying the $\mathcal{MS}$ transforms.

We regularize the $\mathcal{MS}/\mathcal{MA}$ prior to any practical use of it for two reasons: *(i* First, the instabilities of the $\mathcal{MA}$ can overwhelm its use since the shape is sampled inevitably with errors and other (computational) inaccuracy. *(ii)* Second, the *completeness* of the $\mathcal{MA}$ as a representation (by definition) demands for arbitrary high accuracy, which results in a complex and redundant representation. The goal of the $\mathcal{MS}$ regularization is to filter out such redundant information and keep only the qualitative structure of the $\mathcal{MA}$, Figure 1.5(a), such that similar, within-category shapes share a similar $\mathcal{MA}$ topology.

Our $\mathcal{MS}$ regularization approach is based on the $\mathcal{MA}$ transitions. As aforementioned that degenerate transition shapes are more salient and *simpler*, we transform all shapes to their close-by

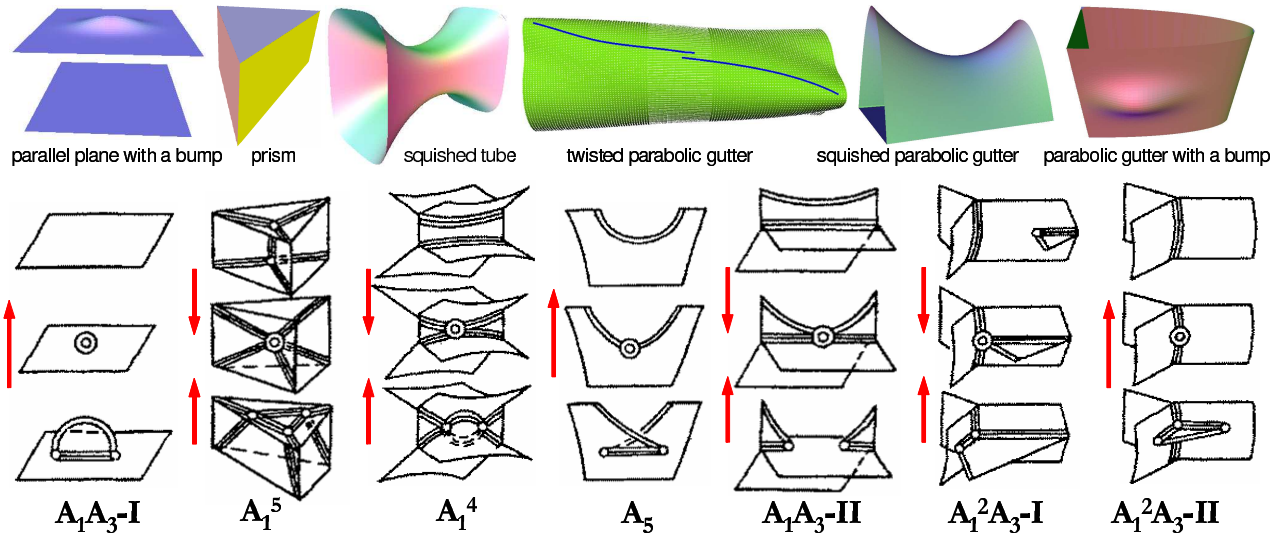


Figure 1.8: (Adapted from [88, Fig.13].) The **seven** generic 3D $\mathcal{MA}$ *transitions* across a one-parameter family of deformations [88]. The synthetic shapes to simulate the transitions are shown in the top row. In considering the transitions in *simplifying* the shape and the $\mathcal{MA}$, some transitions involve only a single way of deformation while others involve two, resulting in totally **eleven** types of *transforms* (in red arrows) [43].

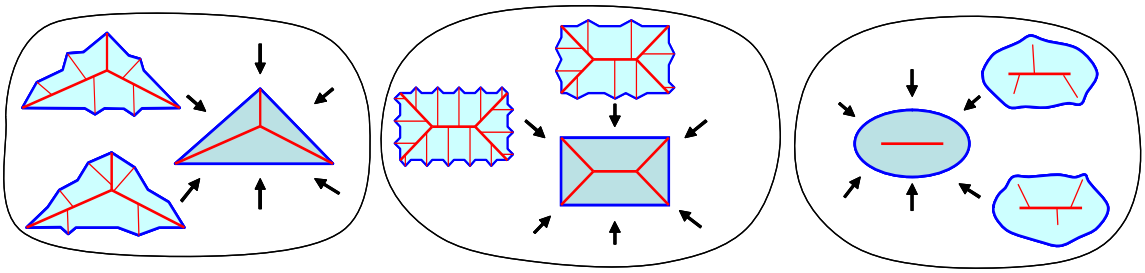


Figure 1.9: Stable $\mathcal{MA}$ regularization by grouping similar shapes under perturbation into equivalence classes and simplifying them toward a *representative* shape (center of arrows) of each class.

more degenerate shapes with simpler $\mathcal{MA}$ indicated by the near-transition configurations. We essentially group together similar shapes into an equivalent class and simplify them toward a "representative" shape of each class. Figure 1.9 illustrates this point, where three categories of shapes (triangular, rectangular, and circular shapes) are depicted and can be characterized by a representative shape in each category. The boundary of each category consists of more complex shape in between the categories.

* Separating the $\mathcal{MA}$ topology and geometry in representation.

Several ideas motivates to separate the (structural) *topology* of the $\mathcal{MA}$ from its fine-scale *geometry*: *(i)* First, the structure of the medial curves/nodes where three or more medial sheets meet is a significant feature of the shape, which can be explicitly represented as a topological graph structure. Such coarse-scale topological structure is by itself important, independent of its underlying geometry. *(ii)* Second, the $\mathcal{MS}$ transforms can be explicitly performed on this coarse-scale structure (to simplify its topology), while keeping the fine-scale geometry intact. This enables an explicit implementation of the $\mathcal{MS}$ transforms as well as a practical approach to remove instabilities. *(iii)* Third,

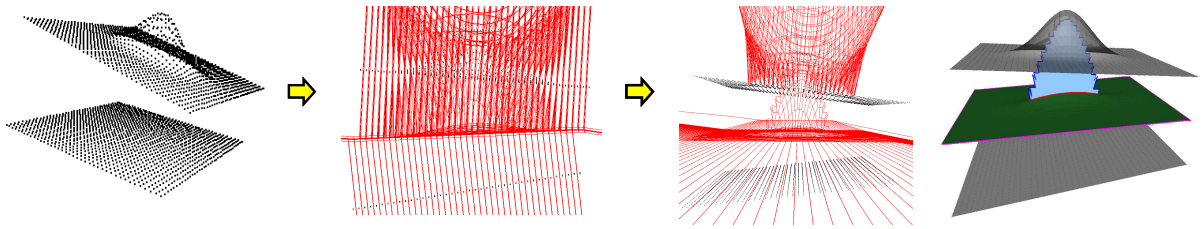


Figure 1.10: (From [45, Fig.6].) After [128, Ch.6], $\mathcal{MS}$ computation and surface meshing: illustration of the *shock segregation* process (detailed in Chapter 6). (a) A set of 3,200 points uniformly sampling a pair of planes, one of which is deformed by an elongated Gaussian kernel. (b) Side-view of the corresponding *full* $\mathcal{MS}$. (c) The remaining $\mathcal{MS}$ after undergoing a series of *gap transforms*. (d) The results are two-fold: *(i)* the reconstructed surface mesh and *(ii)* its corresponding $\mathcal{MS}$ organized into a hypergraph form.

additional information pertinent to the $\mathcal{MS}$ such as the shock radius r, *dynamics* (speed $v = \frac{dr}{ds}$, s as arc-length along the shock flow, *etc.*), associated boundary points of the $\mathcal{MS}$ are also stored in the fine-scale. *(iv)* Forth, the coarse-scale $\mathcal{MS}$ structure can be reduced into a succinct one-dimensional graph structure (by making the sheets implicit), Figure 1.5(a), which allows to adopt graph-based algorithms for effective matching. [8]

* Computational approach: from point cloud to surface — surface meshing and $\mathcal{MS}$ computation.

Our computational approach starts with unorganized point samples, thus requires to reconstruct a surface mesh out of the points to approximate the underlying shape. This meshing process is also related to the computation of the $\mathcal{MA}$ itself. [9] Specifically, we first compute the $\mathcal{MS}$ of the point clouds via an exact computation detailed in [125]. We then *"segregate"* this $\mathcal{MS}$ (referred to as the *full* $\mathcal{MS}$ into two parts, Figure 6.5: one corresponding to the true $\mathcal{MA}$ of the underlying shape, and the other corresponding to sampling artifacts. As a medial branch between two close-by sample points is removed, we close the gap between them (by inserting a surface interplant) and essentially produce a surface mesh after this process. Chapter 6 will elaborate this in details.

* Computational aspects of the $\mathcal{MS}$ regularization.

The regularized $\mathcal{MS}$ after the segregation process can be further regularized by applying the $\mathcal{MS}$ transforms. While the transforms can be applied in any arbitrary order, finding the *optimal* order is computationally expensive and not practical. We consider all transforms in a *greedy* fashion based on their costs (saliency corresponding to the amount of shape change). Ideally, we can start with any initial $\mathcal{MS}$ and apply all transforms (defined in Chapter 5) until finish, but this is also not practical, since initially the number of medial elements can be large and only a few of them shall remain. Our strategy is to *(i)* group together transforms of the same type and with similar costs (to perform them in iterations) and *(ii)* perform the *most-effective* transforms (in simplifying the $\mathcal{MS}$ topology) in prior to others. The detailed computational strategy will be discussed in Chapter 7.

[8] We will elaborate in Chapter 9 that the coarse-scale $\mathcal{MA}$ graph can be augmented to improve the matching rate by adding "virtual links" to handle unexplored transitions.

[9] Our approach fits into the class of *Voronoi filtering* methods in computing the 3D $\mathcal{MA}$. Refer to § 2.2 for details.

1.3 Matching 3D Shapes by Matching their Regularized Medial Scaffolds

* Difficulties in applying the edit-distance matching of the $\mathcal{MS}$ hypergraphs.

While the edit-distance matching of the 2D shock graphs has been successful, the extension of it in matching the $\mathcal{MS}$ remains challenging: The additional dimensionality of the $\mathcal{MS}$ hypergraphs turns the matching of medial curves into matching medial sheets (surfaces). It as well increase the complexity of the $\mathcal{MA}$ transitions and corresponding transforms (as the 'edits'). Although the $\mathcal{MS}$ can be reduced into an (1D) graph structure, it contains loops in general (in bordering medial sheets) thus complicates the edit-distance exploration. (In comparison, the 2D shock graph edit-distance approach in [169] only handles *acyclic trees*.) Finally, a complete shock flow analysis (*i.e.*, the $\mathcal{SC}$) need to be investigated on top of the $\mathcal{MS}$.

* The *graduated assignment* approach to match the $\mathcal{MS}$ hypergraphs.

We implement a practical solution to match the $\mathcal{MS}$ hypergraph by adopting a graph-matching scheme. Specifically, we extend the *graduated assignment* ($\mathcal{GA}$) graph matching algorithm [95] to match the $\mathcal{MS}$ hypergraphs in three aspects. *(i)* First, in addition to the matching of the graph nodes (1st-order assignment) and curves (2nd-order assignment), we introduce a 3rd-order assignment to match the medial sheets by summing up the matching compatibilities at the sheet corners (intersection of two curves). *(ii)* Second, we define the compatibility between pairwise medial nodes, curves, and sheets by matching both their (graph) *structural* similarity and their *parametric* similarity, which captures detailed shape variations. Chapter 9 will elaborate more in details.

1.4 Main Contributions

* Context for this thesis: a continuation from a line of work on using 2D/3D $\mathcal{MA}$ in shape recognition.

This thesis is a direct extension of Leymarie's original work of the medial scaffold [125, 128]. While Leymarie's contribution is three-fold: *(i)* originating the notion of the $\mathcal{MS}$, *(ii)* computation of the (full) $\mathcal{MS}$ from unorganized points, and *(iii)* the early version of the segregation of the $\mathcal{MS}$ and meshing a surface, this thesis continues and extends Leymarie's result on multiple fronts: *(a)* defining the system of $\mathcal{MS}$ transforms [43], *(b)* continuing to refine the surface meshing methodology [45, 46], *(c)* expanding the framework in regularizing the $\mathcal{MS}$ [43], and *(d)* matching the $\mathcal{MS}$ [44]. This thesis is also consistent with a continuation from a line of work of Kimia *et al.* on using the 3D $\mathcal{MA}$ in shape representation and recognition (See Table 11.1 for a summary). Specifically, this thesis is based on: (i) Giblin and Kimia's classification of the 3D $\mathcal{MA}$ [87] and analysis of the $\mathcal{MA}$ transitions [88], (ii) Leymarie and Kimia's notion of the $\mathcal{MS}$ [125], (iii) Sebastian *et al.*'s edit-distance matching of the 2D shock graphs [169], (iv) Sharvit *et al.*'s graduated assignment matching of the 2D shock graphs [175]. The main contributions are summarized as follows.

* Contributions on modeling the $\mathcal{MA}$ as a qualitative 3D shape representation.

We present a comprehensive study of the 3D $\mathcal{MA}$ toward a qualitative shape representation for modeling and recognition. Main contributions are list as follows: The result is a set of $\mathcal{MS}$

1. *Implement the medial scaffold hierarchy.* We provide a practical implementation to explicitly represent the 3D $\mathcal{MA}$ structure as a **topological hypergraph** and separate it from its fine-scale geometry (represented as a polygonal mesh). This *dual-scale* structure enables an explicit

implementation of the $\mathcal{MS}$ hierarchy from the complete representation to the reduced ones with only the succinct structure, detailed in Chapter 3. This *dual-scale* representation also enables explicit simulation of the $\mathcal{MS}$ transforms. (Chapter 4)

2. *Define and implement a complete set of $\mathcal{MS}$ transforms.* We define a set of $\mathcal{MS}$ transforms based on a case-by-case analysis of the seven *generic* $\mathcal{MA}$ transitions of Giblin and Kimia [88]. We also define additional transforms to handle *non-generic* transitions and other cases to complete the system of transforms, detailed in Chapter 5.
3. *Analyze high-order degenerate configurations of the $\mathcal{MS}$.* While the original $\mathcal{MS}$ is studied (by Giblin, Kimia, and Leymarie) in the *generic* cases, we provide an analysis of the *degenerate* $\mathcal{MA}$ points observed in practice, by decomposing them into generic point using the $\mathcal{MS}$ transform.
4. *Compute the $\mathcal{MS}$ from unorganized points and re-mesh a shape surface.* We take the general form of input as unorganized sample points and essentially handle all surface topologies in practice. We compute the $\mathcal{MS}$ together with a reconstructed surface mesh out of the input points. (Chapter 6)
5. *Implement a practical $\mathcal{MS}$ regularization scheme.* We implement a practical computational scheme to regularize the $\mathcal{MS}$. Our system takes input up to $300k$ points and computes the regularized $\mathcal{MS}$ efficiently with full automation. (Chapter 7)

* Contributions on feature detection and 3D shape modeling.

The regularized $\mathcal{MS}$ captures the qualitative structure of the shape in *two* main ways. First, the 'tip' (boundary) of the $\mathcal{MS}$ corresponds to high curvature region of the shape, such as the *ridges* and corners. Second, the medial curve where medial sheets intersect corresponds to a *generalized cylinder axis* of the shape. We list two main contributions in shape modeling:

6. *Ridge detection.* We detect ridges by extrapolate the medial sheets to intersect the shape surface, where the intersection estimates the ridge curves on the surface. (§ 10.2)
7. *A coupled $\mathcal{MA}$-shape representation.* Our $\mathcal{MS}$ regularization scheme not only simplifies the $\mathcal{MS}$ but also produce a tightly-coupled shape. Specifically, we associate all sample points on the shape with their corresponding $\mathcal{MS}$ element(s) and maintain them consistently during all transforms. This coupled skeleton-shape structure is useful in further modeling of the shape such as producing a morphing of shapes in animation. (§ 7.3)

* Contributions on 3D object recognition.

8. *Adopt a graph-matching algorithm to match the 3D $\mathcal{MS}$ hypergraphs.* We implement a practical 3D shape matching system by matching the regularized $\mathcal{MS}$ hypergraphs. A graph matching algorithm is adopted to match the $\mathcal{MS}$ by components (nodes, curves, sheets) and the resulting match (*i.e.*, assignment between graph nodes) estimates a *similarity measure* between the shapes (Chapter 9). We also show that this similarity *metric* approximates the theoretical edit-distance metric with a bounded error (Chapters 8).

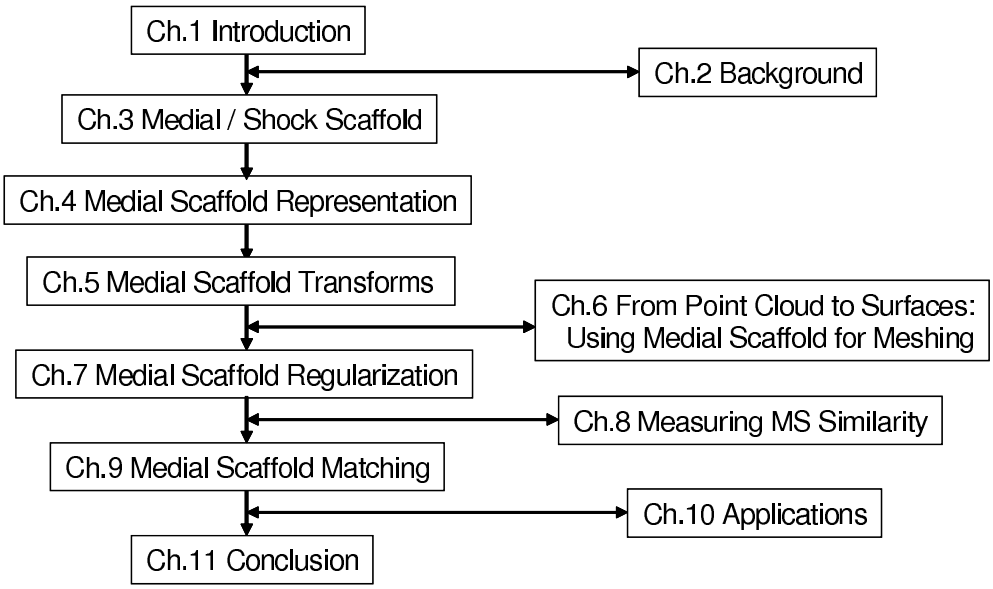


Figure 1.11: Recommended flow of reading. The sequence of essential chapters (on the left) constitute the *core* of the thesis, where other auxiliary chapters (on the right) may only be of interest to a narrower group of readers.

1.5 Organization of Thesis

The thesis is organized hierarchically as a *core* track of essential chapters, together with other auxiliary chapters and appendices to facilitate partial reading at different levels of depth. Figure 1.11 illustrates the recommended flow of reading. At the coarsest level, the introduction (Chapter 1) gives an overview of the thesis and main contributions. Beyond this, the core track of the thesis (Chapters 3, 4, 5, 7, 9, 11) can be read in sequence. Specifically, Chapter 3 elaborates the qualitative representation of the 3D $\mathcal{MA}$ organized as two hypergraph forms—the $\mathcal{MS}$ and the $\mathcal{SC}$. An important special case of the $\mathcal{MS}$ / $\mathcal{SC}$ of a point set is discussed as well, which leads to a practical computational scheme relying on a shock flow analysis. Chapter 4 describes a *dual-scale* representation to separate the structural topology of the $\mathcal{MS}$ from its fine-scale geometry: The structural of the $\mathcal{MS}$ is represented as a topological hypergraph, and the geometry is represented as a polygonal mesh. The data structure for both representations will be elaborated as well. Chapter 5 addresses the $\mathcal{MA}$ *transitions* and defines a set of $\mathcal{MS}$ *transforms* in a case-by-case analysis. Chapter 7 implements the $\mathcal{MS}$ regularization scheme and experiments with practical shapes. Finally, Chapter 9 matches shapes by matching their $\mathcal{MS}$ hypergraphs to define a correspondence and measure their similarity. Chapter 11 covers concluding remarks and future directions.

The remaining chapter elaborate topics complement to the core track and complete the thesis. Chapter 2 reviews the background on symmetry-based shape representation and their use in matching. Chapter 6 addresses the computational aspect of the $\mathcal{MS}$ from a sampled dataset: a *segregation* process of the $\mathcal{MS}$ to retrieve an initial $\mathcal{MS}$ and to mesh the shape surface from the input points. Chapter 8 discuss a theoretical (hypergraph edit-distance) framework as a direct extension of the edit-distance matching of 2D shock graphs, to measure 3D shape similarity as the optimal deformation. The contribution of this chapter is two-fold: The pairwise similarities between the $\mathcal{MS}$ curves and sheets are used in estimating *(i)* the $\mathcal{MS}$ transform costs in Chapter 5 and *(ii)* the matching compatibility between $\mathcal{MS}$ curves in Chapter 9.

We have followed an unorthodox convention as in Leymaire's Ph.D. thesis [128] to include a short header for each paragraph (lead by an asterisk "*"). These summarize the content of most paragraphs and provide an quick overview of the thesis.

Chapter 2

Background on Medial Axis, Graph-based Representation and their Matching

* Overview: survey of graph-based representations, 3D $\mathcal{MA}$, and their use in matching.

This chapter surveys background literature of the structural representation of 3D shapes and their use in matching. Shape representation and matching is a deep research topic with abundant literature. We focus on the skeleton-based representation (for the structure) and the graph matching (on the structure) for two major reasons: *(i)* First, graph has been widely acknowledged as a powerful approach in recognition [68, 81, 40]. *(ii)* On the other hand, as mentioned in Chapter 1, the $\mathcal{MA}$ provides a generic methodology in extracting a skeletal graph structure out of a shape with many desirable properties. In this chapter, we focus on three main topics in surveying the background for this thesis: *(i)* topological graph-based representations such as the Reeb graph, *(ii)* skeletal representations such as the $\mathcal{MA}$ and the curve skeletons, and *(iii)* graph-based shape matching based on the above context. The focus of our survey is then on the property of each representation, their computation and regularization issues, and the use of them in matching.

* Organization of chapter.

This chapter is organized as follows. Section 2.1 starts with the topic of general 3D shape representation and review recent works in the context of shape modeling and matching. Section 2.2 focuses on the study on 3D $\mathcal{MA}$ computation, including the popular Voronoi-based line of works. Section 2.3 continues the topic of 3D $\mathcal{MA}$ on an important issue—the regularization (simplification) to make it robust against the omnipresent instabilities of the skeleton. Finally, Section 2.4 covers the matching of the graph-based matching in order to match the underlying shapes. At the end, we include a table summarizing the main abbreviated symbols frequently used in this thesis.

2.1 Survey of 3D Shape Representation and Matching

Shape representation is a fundamental problem in computer vision with an abundant literature. Refer to [28, 41, 107, 83, 197, 207] for surveys in matching 3D shapes. We focus on the representative ability of each method in two aspects, namely, in how they *(i)* capture the 3D shapes for the *modeling* purpose and *(ii)* provide a proper similarity measure for the *matching* purpose, and briefly organize recent approaches into two main categories: [1]

[1] Note that we do not intend to cover a full treatment of the subject (of shape representation). Other classification schemes are possible; *e.g.*, the addition of "hybrid" methods combining the main feature of two or more categories.

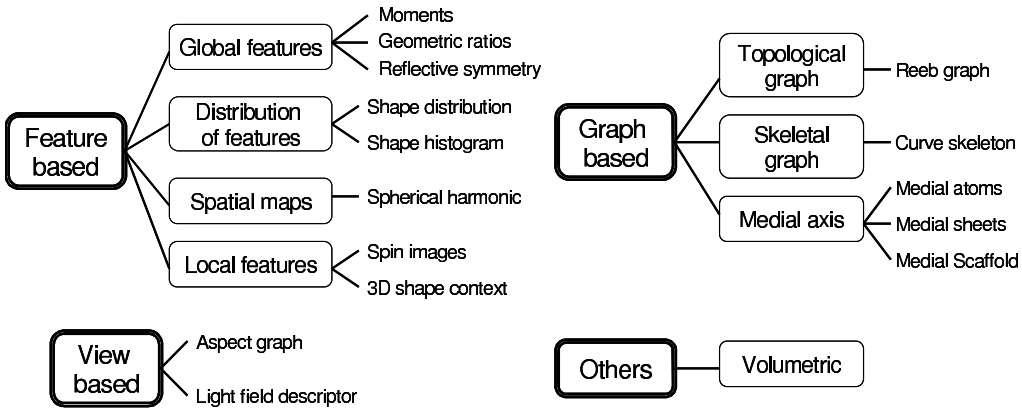


Figure 2.1: A classification of 3D shape representations in an overview.

(i) the popular **descriptor**-based representations, which extract features from the shape and use them to represent the underlying shapes in matching, including the *feature-vector* based and *view* based methods, *etc.*; and

(ii) the **graph**-based representations, which aim to obtain a structural description of the shape, including the symmetry-based $\mathcal{MA}$ and other skeletal representations.

Figure 2.1 summarizes the classification of the 3D shape representations in an overview.

* Descriptor-based 3D shape representations.

The **descriptor**-based methods are the current mainstream, where a shape descriptor (feature, signature) is extracted to describe the shape and to distinguish it from others. The descriptor-based methods are useful in shape retrieval, and a large variety of such descriptors has been proposed, which can be briefly classified into *five* sub-categories: *(i)* local feature based, which relies on local salient geometric features [110, 85] such as the curvature or primitive of flat regions [105, 167], or bending invariant signatures [73], *(ii) spherical* functions, such as the spherical harmonic [111, 151], shape histogram, shape context [119], or transform-based [59], *(iii) statistical measure*-based, such as the shape distribution [147] and other generalizations, *(iv) view*-based, by matching 2D views of the 3D objects [178, 123, 10, 48, 223, 57], and *(v) voxel*-based, assuming the input is a solid volume such that a distance transform [111] or a skeleton can be effectively computed.

We mention a few methods which are capable to handle *partial* shape matching and *non-closed* shapes such as unorganized points or partial meshes [2]. The spin-image [110] takes point based input and models local salient geometric features in matching. In [85], salient intrinsic geometric feature such as the curvature is indexed and matched via a voting scheme accelerated by geometric hashing. In [105], CAD models are retrieved by segmenting the partial 3D point clouds into surface patches for matching. In [167], primitive features (planes, cylinders) are extracted from point cloud and a topological graph is built for matching in architectural applications. In [62], the Delaunay transform after segmentation is used as signature for matching.

* Graph-based 3D shape representations.

[2]The processing of general 3D data is much involved than those simple closed (water-tight) models. The main difficulty is that their topology is arbitrary and can contain boundary/holes, missing data, or artifacts such as intersecting or disjoint polygons.

The **graph**-based methods employ a graph to represent the connectivity between *parts* of a shape. Such structural layout enables matching *partial* or *deformed* shapes more naturally. Recent works are organized by the *graph type* in use into four categories as follows:

(i) Reeb graph: The Reeb graph is a topological graph originated from a mathematician George Reeb (1946) based on the Morse analysis on a continuous scalar function defined on an object [26]. Typically, a height function or a geodesic function is chosen, Figure 2.2, to fulfill the condition that the critical points are not degenerate on such function, thus the Morse analysis can be applied to extract the Reeb graph. Note that the Reeb graph edges are not necessarily skeletal/symmetric. A recent work of Shi *et al.* propose to bridge the gap between the Reeb graph and the skeletal graphs [177]. The Reeb graph has been applied in shape analysis and matching in [99], where a multi-resolution Reeb graph is defined using geodesic distance for matching. In [27], an extended Reeb graph is matched via graph matching. In [25, 23], the Reeb graph is applied for the retrieval of 3D CAD models.

(ii) Skeletal graph: The *skeletal* graph of Sundar *et al.* [192] encodes topological signature vectors for matching, which is computed by thinning a volumetric representation (*i.e.*, voxel-based) via a distance transform. The work of Brennecke and Isenberg [37] constructs an internal skeletal graph for a 3D object by iteratively simplifying a polygonal mesh using an edge-collapse algorithm (until only edges not attached to any surface polygons are left). In [208], the *ik-skeletons* is automatically generated for use as control skeleton in animation.

(iii) Curve skeleton ($\mathcal{CS}$): The $\mathcal{CS}$ is an one-dimensional centerline roughly central inside a 3D shape to capture its structure [54, 55, 53, 66]. Although the $\mathcal{CS}$ is simplified than the $\mathcal{MA}$ (which consists of 2D sheets in general), a suitable mathematical definition for such 'centerness' still needs investigation . We will provides a detailed survey and comparison of the $\mathcal{CS}$ and $\mathcal{MA}$ in § 2.3.2.

(iv) Medial sheets: Pizer *et al.* pioneer in using the fixed-topology (2D) sheet-like medial model for segmentation [188]. Their computation of the $\mathcal{MA}$ sheets is summarized in Figure 2.6. Siddiqi *et al.* [183] employ a directed acyclic graph of the *medial sheets* to retrieve articulated 3D models. Their main results are summarized in Figure 2.9.

* Remark on our approach in comparing to existing shape representations.

In the context of the above survey of 3D shape representations and their use in matching. Our approach is one of the few that explicitly exploit the 3D $\mathcal{MA}$ sheets. Furthermore, our method features an explicit use of the connectivity between the $\mathcal{MA}$ sheets and we aim to regularize such connectivity to preserve the essence structure of the shape (handled as $\mathcal{MA}$ *transitions* mentioned in § 1.1). On the other hand, in comparing to the Reeb graph, skeletal graph, and the $\mathcal{CS}$, our representation can be also reduced into an 1D graph-like structure (the $\mathcal{MS}$ graph described in Chapter 3), which also allows for an adopting of the graph matching scheme. Below we survey the background on the 3D $\mathcal{MA}$ computation and regularization to compare our approach in details.

2.2 Survey of the 3D Medial Axis Computation

* A summary of major 3D $\mathcal{MA}$ computation approaches.

"The idea of representing shapes and their interaction with the surrounding space by medial symmetries is not new, and can be traced back to ancient times in the arts and architecture [127], from the stick figures of the primitive cave drawings to the Vitruvian man and Descartes analysis of the skies, which is a precursor of Voronoi diagrams [145]", quoted from [129]. A more modern definition of

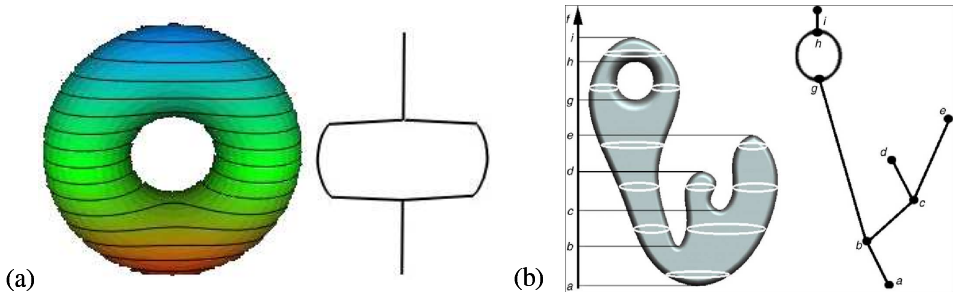


Figure 2.2: Example of Reeb graphs: (a) (Adapted from Hilaga *et al.* [99, Fig.1].) Torus and its Reeb graph using a height function. (b) (Adapted from Biasotti *et al.* [26, Fig.6].) A 2D manifold and its Reeb graph using a height function. The letters from *a* to *i* denote the correspondence between critical values and Reeb graph nodes.

the $\mathcal{MA}$ as a shape representation is originated from Blum [31]. Existing works in 3D $\mathcal{MA}$ computation have been diverse and differ in the philosophy as well as in the representation. We borrow from Leymarie and Kimia [125] the organization of the major $\mathcal{MA}$ computation methods into *seven* categories:

1. *Thinning* by peering layers of elements until the $\mathcal{MA}$ is reached, such as in mathematical morphology [171],
2. *Ridge following* on the distance map using a discrete grid [134],
3. Solving the partial differential equations (PDE) simulating the propagating wavefronts in the spirit of Blum's *grass-fire* model [31], where the background space is lit up by fires initiated at shape boundary loci and where the singularities of collision of wavefronts denote the $\mathcal{MA}$ loci [180],
4. *Refining* the Voronoi diagram ($\mathcal{VD}$) of sample points toward to the $\mathcal{MA}$ [8],
5. Computing the interior $\mathcal{MA}$ from a solid polyhedral shape [56, 176],
6. Full *bisector* computations are followed by trimming operations to define generalized descriptions of the $\mathcal{MA}$ [103, 74], and
7. *Primitive* shapes for which known medial representations are directly available and can be retro-fit to the data [225].

For a detailed survey on the $\mathcal{MA}$ computation, refer to [128, Ch. 2]. In the remaining of this section, we focus on the methods pertinent to our, *i.e.*, computing 3D $\mathcal{MA}$ from point-sampled shapes. Also we focus on comparing key properties of the methods such as those suggested by Attali *et al.* [14], that a "good" $\mathcal{MA}$ approximation should have:

1. *Convergence*: as the sampling density tends to infinity, the computed $\mathcal{MA}$ should converge to the exact one.
2. *Homotopy*: the resulting $\mathcal{MA}$ should preserve topology, *i.e.*, has the same number of connected components, holes, *etc.*
3. *Reversibility* (reconstruction ability): the $\mathcal{MA}$ should be able to recover the underlying shape.

* *Voronoi-filtering* to compute 3D $\mathcal{MA}$ from a point set; the meshing of points.

A major branch of $\mathcal{MA}$ computation is Voronoi-based which typically takes point-sampled shapes as input, partly due to the reason that for other forms of shape such as polygonal meshes, the true $\mathcal{MA}$ is in general difficult to compute. Observe in Figure 2.3 for the well-known fact that the Voronoi diagram ($\mathcal{VD}$) and the $\mathcal{MA}$ are closely related. The development of this branch of $\mathcal{MA}$ computation (from point-based inputs) is then closely related to the problem of *filtering* the Voronoi-Delaunay structure to obtain the $\mathcal{MA}$ as well as producing a surface meshing of the points. [3] In the next section, we survey the *Voronoi-filtering* methods whose initial goal is to obtain a surface mesh from the input points, while their development leads to a major approach to compute the $\mathcal{MA}$. [4] Section 2.3 will re-examinable some of these methods in focusing on how they continue to regularize the resulting $\mathcal{MA}$.

2.2.1 Survey of Voronoi-filtering methods for $\mathcal{MA}$ computation and surface meshing

The Voronoi filtering methods can be organized into *two* categories, depending on how the 'filtering' of the $\mathcal{VD}$ (in meshing the surfaces) is done; it can be either *(i) surface-oriented* or *(ii) volume-oriented*.

* The incremental surface-oriented meshing and $\mathcal{MA}$ computation methods.

(I) First, *incremental surface-oriented* methods select suitable Delaunay triangles interpolating sample points either in a batch or in a greedy incremental fashion. A popular recent set of such methods was initiated by the works of Amenta *et al.*, Figure 2.3, who proposed a Voronoi filtering method [5] where they first consider furthest apart Voronoi vertices of a Voronoi cell of a sample point p to define *poles* by pairs, to approximate the local surface normal. They also define the *local feature size* as the minimum distance from a sample p to the *theoretical* $\mathcal{MA}$ (for a smooth surface with bounded curvatures), which is used to derive theoretical constraints on the sampling and guarantees on the resulting meshes. The method requires a second pass of Delaunay computation using the poles as additional vertices. Triangles through triplets of original sample points are kept to construct a final mesh called the *crust*. Difficulties occur as the *poles* do not always approximate well surface normals, and, thus, a post-processing step is needed to trim results and fix such problems. This was further refined by defining a local neighborhood in the vicinity of a sample p taken as the complement to a cone intersection with its Voronoi cell, called "co-cone," giving a heuristic to approximate the local tangent space at p to restrict the search for neighboring samples to create candidate triangle interpolants. This improved *cocone* method [6] has for main advantage the bypassing of the second Delaunay computation in computing a crust mesh. However, it still requires heuristics to try repair the computed mesh. Subsequently, [61] detect under-sampled regions near boundary and shape features such as ridges, where topological errors such as holes in the mesh are frequent. Note that the theoretical guarantee of obtaining a correct reconstruction in the above *crust* and *cocone* methods is only valid for a strict requirement of densely sampled points, for a smooth surface where the $\mathcal{MA}$ nowhere reaches the surface, a condition which is not necessarily true in all practical situations, such as is the case of objects found in CAD-CAM (with sharp surface features and boundaries).

Aware of these issues, Petitjean and Boyer proposed another approach by defining a notion of r-regularity measured from the samples alone and combined with a discrete (rather than theoretical)

[3] A more comprehensive survey of the surface reconstruction problem in a more general setup, *e.g.*, including using *implicit surfaces*, is in Appendix **??**.

[4] Our approach also fits into this framework, that the *shock segregation* process in Chapter 6 corresponds to the filtering step and Chapter 7 continues to regularize the $\mathcal{MS}$.

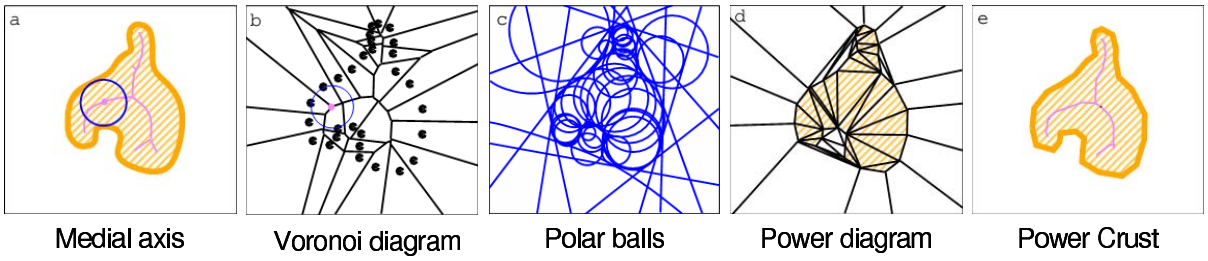


Figure 2.3: (Adapted from [7, Fig.2].) A 2D example of N. Amenta *et al.*'s Power Crust construction [8]. (a) An object with its $\mathcal{MA}$, with one maximal interior ball shown. (b) The $\mathcal{VD}$ of the boundary sample points, with the Voronoi ball surrounding one *pole* shown. (c) The inner and outer polar balls. Outer polar balls with centers at infinity degenerate to half-spaces on the convex hull. (d) The power diagram cells of the poles, labeled inner and outer. (e) The Power Crust and the power shape of its interior solid.

$\mathcal{MA}$ [156]. A surface mesh is then constructed by a propagation scheme by selecting Delaunay faces meeting the r-regularity criterion. Kuo and Yau [121] improve the *crust* algorithm to preserve sharp features by a combined adaptive sculpting and region-growing scheme.

Cohen-Steiner and Da have proposed a greedy incremental algorithm that uses for its main heuristic in selecting interpolants the dihedral angle between Delaunay face pairs, which postpones difficult decisions on a queue, and uses additional heuristics to detect surface boundaries [50]. The greedy β-skeleton in [102] iteratively meshes suitable Delaunay faces satisfying certain topology constraints, but an overall error repairing heuristic is lacking.

* The volume-oriented methods for surface meshing and $\mathcal{MA}$ computation.

(II) Second, *volume-oriented methods* consider the restrictive problem of reconstructing the closed surface bounding some solid. An early "volume sculpting" method was proposed by Boissonnat where various tetrahedra faces can be removed one by one [33], which was later refined in particular by Attali with a notion of r-regular shape taken from mathematical morphology [13]. [5] Another variant, the *Power Crust*, is based on the power diagram (a weighted Voronoi diagram), with theoretical guarantees under a proper sampling assumption, and with polygonal interpolants rather than triangles [9]. The *tight cocone* is based on the original cocone method (see above), but specialized to produce a water-tight mesh for solids [60]. More recently, robustness to noise for this approach is based on a model where both the sampling density and noise level can vary locally [65]. A recent improvement of the Power Crust in [138] handles noisy dataset and allows arbitrary over-sampling densities.

2.3 Survey of the 3D Medial Axis Regularization

After computing the 3D $\mathcal{MA}$, the following step of *regularization* is usually necessary (due to the ubiquitous instability) and is in some cases embedded with the prior. Several $\mathcal{MA}$ regularization techniques have been independently developed but rather share common ideas and mainly vary in how the $\mathcal{MA}$ elements are selected to prune. For an in-depth surveys, see [12, 189, 67, 158, 200]. We briefly review two categories of approaches, namely, the *(i) Voronoi refinement* methods and *(ii)*

[5] A well-known variant of this type of methods is the α-shapes of Edelsbrunner *et al.* [71] and the recent conformal α-shape of Cazals *et al.* [42] and weighted α-shape of Park *et al.* [152].

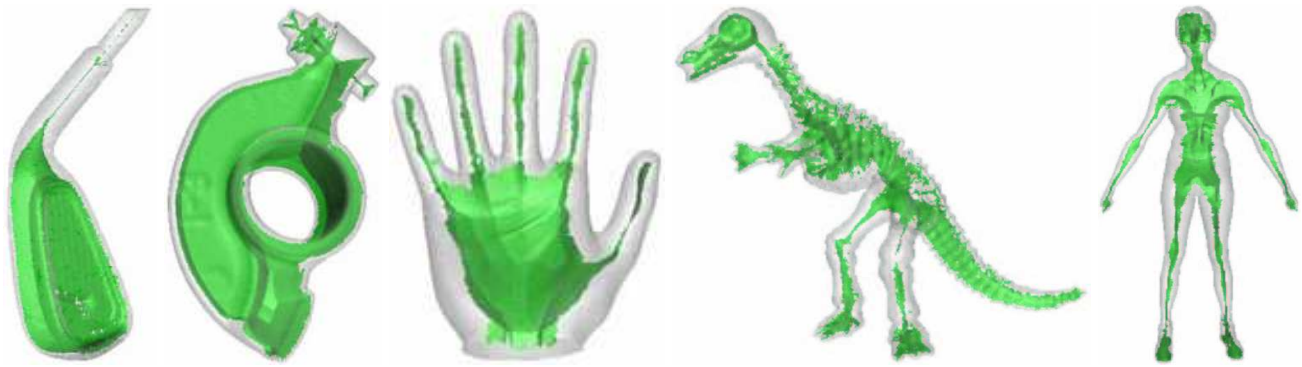


Figure 2.4: (Adapted from [67, Fig.13].) Result Dey and Zhao's 3D $\mathcal{MA}$ extraction based on filtering the 3D $\mathcal{VD}$ of several objects (the club, rocker, hand, dino, and female dataset) using the angle and ratio conditions (see text).

methods refining the $\mathcal{MA}$ of a polyhedron. We also mention a few other works pertinent to the $\mathcal{MA}$ regularization in § 2.3.1.

* $\mathcal{MA}$ regularization via refinement of the Voronoi-filtering results.

We continue to survey the Voronoi-filtering methods on how they refine the resulting $\mathcal{MA}$ in order to regularize it. In the early works of [143, 14], the interior Delaunay tetrahedra are deleted in layers while maintaining topological consistency. The main problem is that there exist Voronoi vertices near the object surface (centers of flat Delaunay tetrahedra called *'slivers'*) that restrain the regularization process. Such 'slivers' can be filtered out using the *poles* of Amenta *et al.* [7]. In the *PowerCrust* approach [7], the $\mathcal{MA}$ is computed using the *power diagram* (a weighted $\mathcal{VD}$) of the inner poles. The $\mathcal{MA}$ is then simplified by removing poles of small surface features or with contact balls overlapping significantly. Although this method has some theoretical support, it requires two passes of Voronoi computations and the resulting $\mathcal{MA}$ is not on the Voronoi complex of the input (and needs additional heuristics to clean up). A following work in [67] extracts the $\mathcal{MA}$ by directly filtering the $\mathcal{VD}$, Figure 2.4. A Voronoi face is removed if *(i)* the *angle* between the estimated surface normal and its dual Delaunay edge is not small, *(ii)* the *ratio* of the object feature size to its radius is small, which is related to the $\mathcal{MA}$ *significance* [6]. The resulting $\mathcal{MA}$ may contain unwanted internal 'holes' and the association between input samples and the $\mathcal{MA}$ can be lost.

* $\mathcal{MA}$ regularization methods retaining a *"homotopy"* of the $\mathcal{MA}$.

Recent development in $\mathcal{MA}$ simplification has focused on retaining the *homotopy* in the process. Informally, the purpose is to ensure that the topology between the shape and the $\mathcal{MA}$ are the same [7] Tam and Heidrich [194] uses the PowerCrust to compute the $\mathcal{MA}$ and *'peels'* off medial sheets according to *(i)* the sheet size (number of triangles) and *(ii)* its corresponding shape volume (estimated using the Delaunay tetrahedra), while maintaining the topological consistency, Figure 2.5.

[6] The *significance* of a $\mathcal{MA}$ branch is related to the corresponding *object angle* θ and the $\mathcal{MA}$ *formation speed* v. At any $\mathcal{MA}$ point p, v changes with an angle ϕ between the $\mathcal{MA}$ formation direction and the vector from p to the corresponding object point: $v = -1/\cos\phi$ [86].

[7] Lieutier shows that any solid shape are *homotopy equivalent* to its $\mathcal{MA}$ [132]: There exists a one-to-one correspondence between connected components, cycles, holes, tunnels, cavities, *etc.* and the way they are related. The homotopy avoids the case that the shape has holes or is connected while the $\mathcal{MA}$ does not, or vice versa [215].

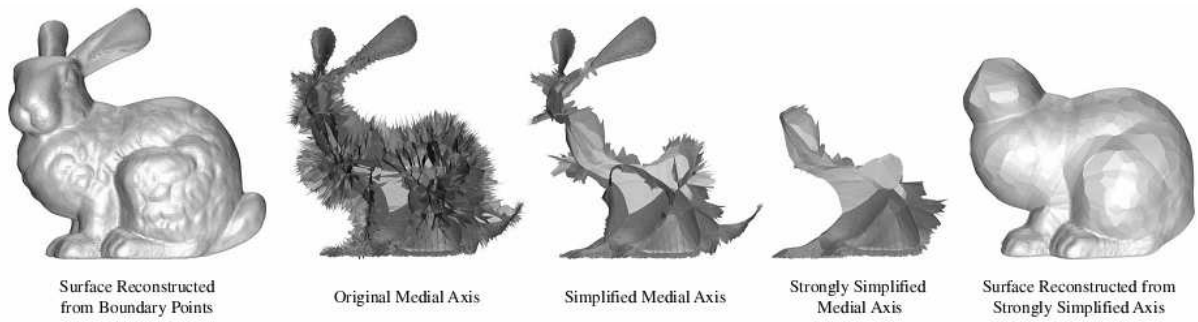


Figure 2.5: (Adapted from [194, Title Fig.].) R. Tam and W. Heidrich's Voronoi-based 3D $\mathcal{MA}$ computation using Power Crust and $\mathcal{MA}$ simplification by 'peeling' of medial sheets.

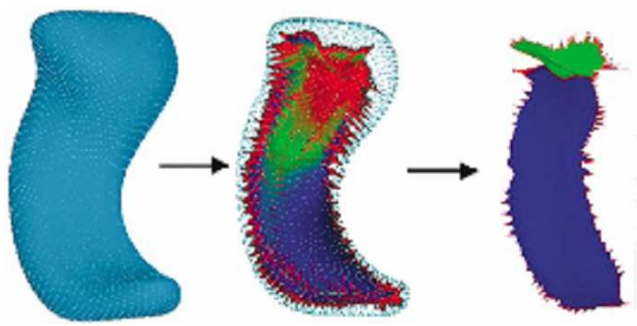


Figure 2.6: (Adapted from [188, Fig.3].) Voronoi-based $\mathcal{MA}$ computation and pruning of medial sheets toward the *m-Rep*, a medial axis representation by Styner, Gerig, Joshi, and S. Pizer *et al.* The branching topology of $\mathcal{MA}$ is represented by a set of sheets computed from pruning Voronoi skeletons. The (unpruned) 'raw' Voronoi skeleton is shown in the middle.

[8] In [188], the *m-Rep* is extracted from an inner Voronoi skeleton via the pruning and merging of medial sheets while minimizing the change of the underlying shape according to two criteria similar to [194], Figure 2.6: *(i)* the sheet area (number of vertices) and *(ii)* its corresponding shape volume.

* The "flow complex" in surface meshing from points and $\mathcal{MA}$ computation.

Another recent line of work, the *flow complex* ($\mathcal{FC}$) [92] relies on the *Morse* analysis on the distance flow on the dual Voronoi–Delaunay complex. The $\mathcal{FC}$ summarized in Figure 2.7 reconstructs a surface mesh from the input points and extracts the $\mathcal{MA}$ while retaining the homotopy and is closely related to our approach. The details is summarized as follows. The critical points of the (discrete) distance function, h_Σ, from the point samples are exactly given as the intersection of Voronoi k-faces and their dual Delaunay simplices [92, 64]. Specifically, Giessen *et al.* consider properties of the "flow" induced by h_Σ, *i.e.*, the unique direction of steepest (distance) ascent of h_Σ at any non-critical point, x. All critical points, c, are given a 0 vector, while every other point in $\mathbb{R}^3$ is associated to the unique unit vector of steepest ascent (of h_Σ); this defines a vector field on which discrete Morse theory can be applied. A *"stable manifold"* of a critical point c of h_Σ is defined as the set of points whose orbit ends in c (*i.e.*, flow into c). This creates *four* types (indices) of stable manifolds in 3D:

- an *index-0* critical point, *i.e.*, a local minimum of h_Σ — identically a sample point — has for stable manifold the sample point itself;

[8]The use of volume as a saliency measure without referring to its relative $\mathcal{MA}$ radius can cause to remove salient but smaller $\mathcal{MA}$ sheets prior to pruning of unstable but larger ones. See [200] for a 2D analysis. Also refer to Tam and Heidrich's work [193] on 2D $\mathcal{MA}$ noise removal.

- an *index-1* critical point is a 1-saddle which sits at the intersection point of a Delaunay edge with its dual Voronoi facet, and has for associated stable manifold a Delaunay (Gabriel) edge;
- an *index-2* critical point is a 2-saddle which sits at the intersection point of a Delaunay facet (triangle) with its dual Voronoi edge, and has for associated stable manifold a piecewise linear surface patch; and
- an *index-3* critical point is a local maximum (of h_Σ), a Voronoi vertex.

For surface meshing, the stable manifolds of index-2 are used to reconstruct a surface in this scheme. Under high density sampling conditions and for smooth surfaces, Giessen *et al.* show that the critical points can be separated into two classes: one class made of critical points remaining nearby the original surface, and the other made of critical points near the (theoretical) $\mathcal{MA}$ of that (assumed) smooth surface. *For the computation of the* $\mathcal{MA}$, a possibly extendable *"core"* (set of 'unstable manifolds') is computed from the flow analysis on the $\mathcal{VD}$ to approximate the $\mathcal{MA}$.

The classification of critical points is highly related to the way we classify the shock points in Chapter 3. We point out that our approach is based on a *singularity* theory [87, 125] and produces a richer superset of critical points then the $\mathcal{FC}$. That is, all the "relay" type of *shock* vertices (A_1^2-3, A_1^3-3) [125] (see Chapter 3) are *not* identified as *critical* points in the Morse analysis in the flow complex framework. Omitting these singular points makes their subsequent analysis and algorithms different from ours. [9]

* Refining the $\mathcal{MA}$ of a polyhedron.

Another branch of research extracts 3D $\mathcal{MA}$ from a (solid) polyhedral mesh and regularizes it. In [189], $\mathcal{MA}$ branches are pruned using a *separation angle* related to the $\mathcal{MA}$ significance. Similar approach has been implemented using graphics hardware in [190], Figure 2.8. The homotopy of the $\mathcal{MA}$ is preserved by only removing non-interior medial sheets. In [187, 180] the $\mathcal{MA}$ is computed and pruned simultaneously by measuring the *Average Outward Flux* (AOF) of the gradient of the distance field in each voxel, Figure 2.9. [10] However, the extracted $\mathcal{MA}$ is rough in accuracy (due to voxelization) and the topology of the inter-connectivity between medial sheets are still not handled well. In [56], an exact $\mathcal{MA}$ computation of a polyhedron is proposed. The exact $\mathcal{MA}$ computation of polygonal meshes involves complicated intersection between high-order polynomials [128, Ch.7], thus only results on models of relatively low-resolution are demonstrated, which is summarized in Figure 2.13.

2.3.1 Remarks on related $\mathcal{MA}$ regularization works

We point out a few related works in $\mathcal{MA}$ regularization not mentioned so far.

* The *pair-mesh* [172] to extract a "coupled" $\mathcal{MA}$-shape structure.

It is well-known that in 2D the medial branches can be tightly coupled with portions of the shape (as shown in Figure 1.1(e)), which is a desirable property one want achieve in 3D. Our $\mathcal{MS}$ provides one such solution which will be detailed in § 7.3. In comparison, Shamir and Shaham's *'pair-mesh'* [172] also provides such a coupled $\mathcal{MA}$-shape structure, and it works in a different manner, Figure 2.10.

[9] Also see a recent work in [164] for an extension of Edelsbrunner's WRAP algorithm for surface reconstruction, based on ideas from the Morse theory applied to the flow map induced by the distance function: the "separation of critical points" resembles our "segregation of shock points" in § 6.

[10] The discontinuity of the gradient flux is large at significant $\mathcal{MA}$ branches and is again related to the speed v and object angle θ.

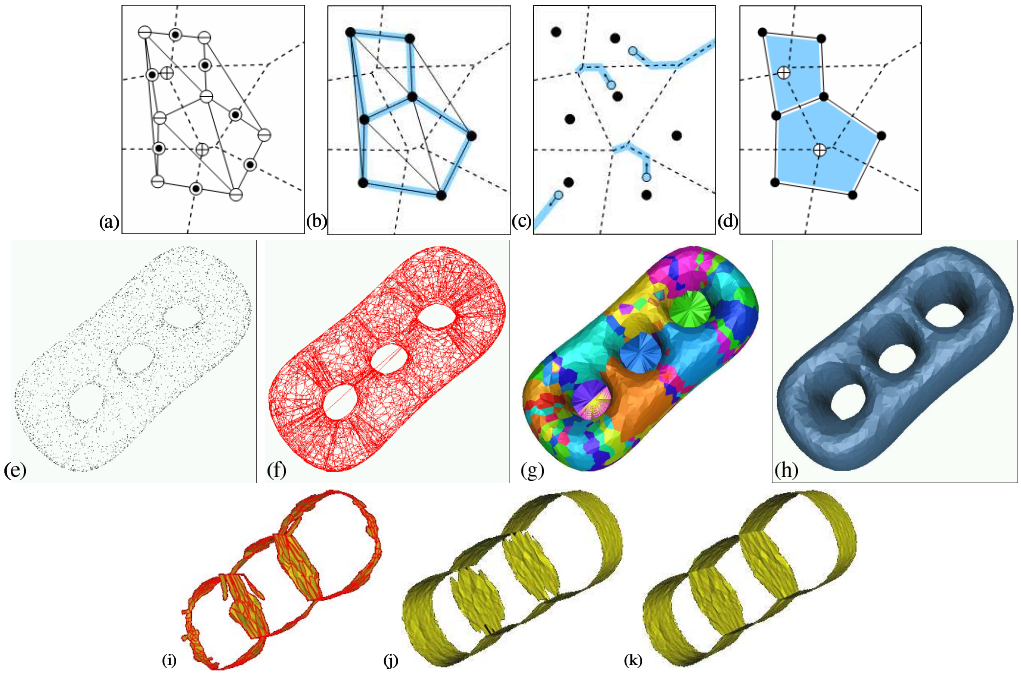


Figure 2.7: (Adapted from [91, (a-b): Fig.2, (c-d): Fig.3, (e-h): Fig.8] and [92, (i-k): Fig.7].) A summary of J. Giessen, T. Dey, E. Ramos, and B. Sadri's *Flow Complex* ($\mathcal{FC}$) on surface reconstruction and $\mathcal{MA}$ computation in 2D (a-d) and in 3D (e-k). (a) The minimum ('-' in the circles), saddles $\odot$, and maxima $\oplus$ of the distance function associated with the samples. By definition, the distance at the sample is 0, the minimum. (b) The *Gabriel graph* of the samples where the Gabriel edges are highlighted in blue; the Delaunay edges are in dark lines. (c) Some orbits of the 'flow' induced by the sample points. (d) The two regions where the 'flow' flows into the sink, the local maxima of the distance field. (e) The 3-hole model of $12,008$ points. (f) The Gabriel graph of (e). (g) The flow complex, where the boundaries between stable manifolds of different maxima are colored randomly. (h) The *reduced flow complex* which provides a reconstructed surface. (i) The *'core'* computed for the 3-holes model. The red lines are either unstable manifolds of index-2 saddle points or the 1D parts (hairs) of index-1 saddle points. (j) Filtered Voronoi facets based on a condition similar to the *angle condition* in [67]. (k) The *'extended core'*, the core plus the flow closures of the facets in (j), which approximates the $\mathcal{MA}$.

The pair-mesh is a tetrahedral complex representation of the solid based on its approximated $\mathcal{MA}$ skeleton, which preserves the topological relation between them. However, this approach only takes a solid mesh (water-tight without boundary) as input, due to that the inner $\mathcal{VD}$ and the PowerCrust is used in extract the $\mathcal{MA}$. A related work is [36], where a hierarchical union of spheres centered at the skeleton with varying radius are used to approximate the shape, in the application of collision handling.

* The λ-*MA* [47] and the regularization of the $\mathcal{MS}$ boundary (rib curves).

The recent work of Chazal and Lieutier *et al.*'s λ-*MA* [47] defines a subset of the $\mathcal{MA}$ by excluding $\mathcal{MA}$ points with radius less than the parameter λ, Figure 2.11. They provide a proof that if λ is smaller than a *weak feature size* (which can be informally interpreted as a condition that no $\mathcal{MA}$ junction points are pruned), the λ-MA is continuous under measure in the Hausdorff distance. This gives a theoretical support of our approach of a "min-radius" pruning (via element-wise splice transforms) in Chapter 7 in regularizing the rib curves by pruning out all $\mathcal{MA}$ elements closer to the

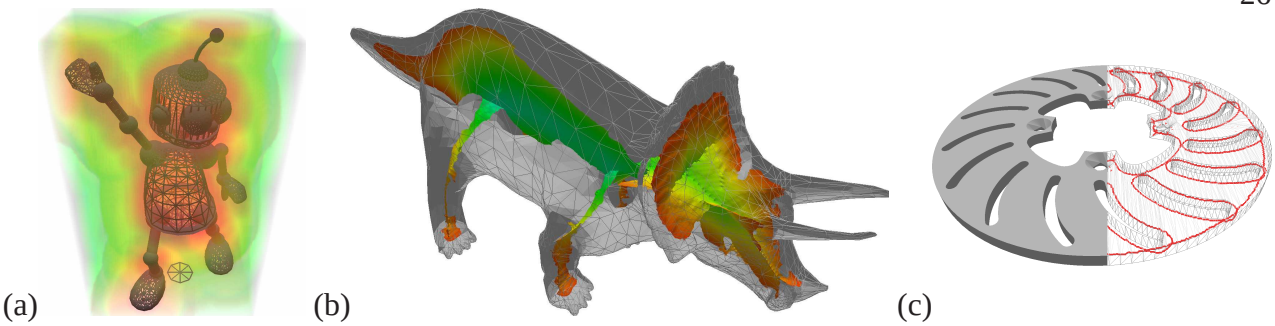


Figure 2.8: (Adapted from [190, (a): Fig.1, (b-c): Fig.6].) The $\mathcal{MA}$ computed by A. Sud, M. Otaduy and D. Manocha. (a) The 3D distance field of the Hugo model (17k polygons, grid size=$73 \times 45 \times 128$). Distance to the surface is colored increasingly from red to green. (b) The $\mathcal{MA}$ of the Triceratops model (5.6k polygons, grid size=$128 \times 56 \times 42$) is colored by the distance from the boundary. (c) The medial seam curves (the A_1^3 axial curves) of the Brake rotor model (4.7k polygons, grid size=$4 \times 128 \times 128$) are shown in red.

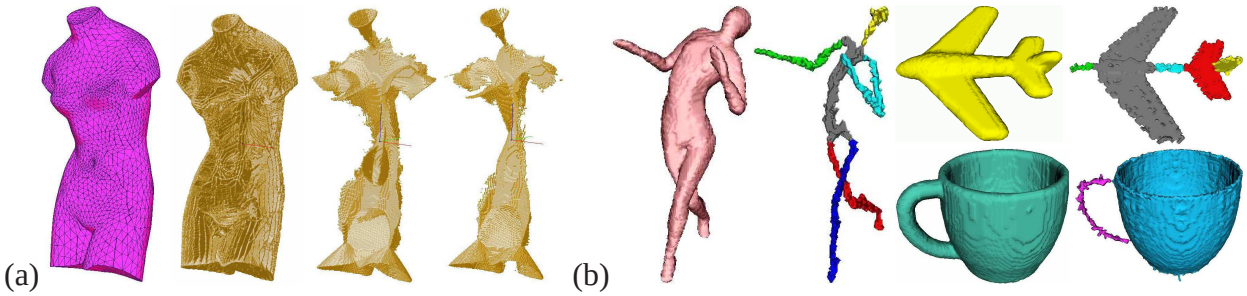


Figure 2.9: (Adapted from [187, (a): Fig.6] and [183, (b): Fig.1,2].) Medial surface computed by K. Siddiqi *et al.* based on an analysis of the *Average Outward Flux* (AOF) of the gradient of the distance function [180]. (a) The Venus mesh (5,672 triangles) and its medial surface at a resolution of $177 \times 129 \times 36$ voxels: the full medial surface, voxels with a (processed) AOF values below -0.15 and -0.25. In (b), the medial surfaces of voxelized models from the McGill shape database [183] is automatically partitioned into parts (in different colors).

shape boundary (within the threshold λ), detailed in § 7.2.2.

* Yoshizawa *et al*'s "two-sided" $\mathcal{MA}$ mesh to apply smoothing for regularization.

Notably, the approach of Yoshizawa *et al.* [219, 100] extracts smooth rib curves by representing the (Voronoi-refined) $\mathcal{MA}$ as a *two-sided* non-intersecting surface and apply standard Laplacian smoothing (and loop subdivision) on the $\mathcal{MA}$ manifold, Figure 2.12. However, this approach requires a dense and uniformly sampled mesh as a input, and all interior structural connectivity (junctions) of the $\mathcal{MA}$ is lost. [11]

* Limitation of existing $\mathcal{MA}$ regularization methods; comparison to ours.

To the best of our knowledge, all existing $\mathcal{MA}$ regularization approaches focus on *pruning* medial sheets (either by element, by component, in a sequence, or in a batch) but make no effort to simplify the structural inter-connectivity (topology) between the sheets, which we see as a crucial element in

[11] In comparison, we can handle a very sparse input (> 5 points) of unorganized points and provide both the meshing and $\mathcal{MA}$.

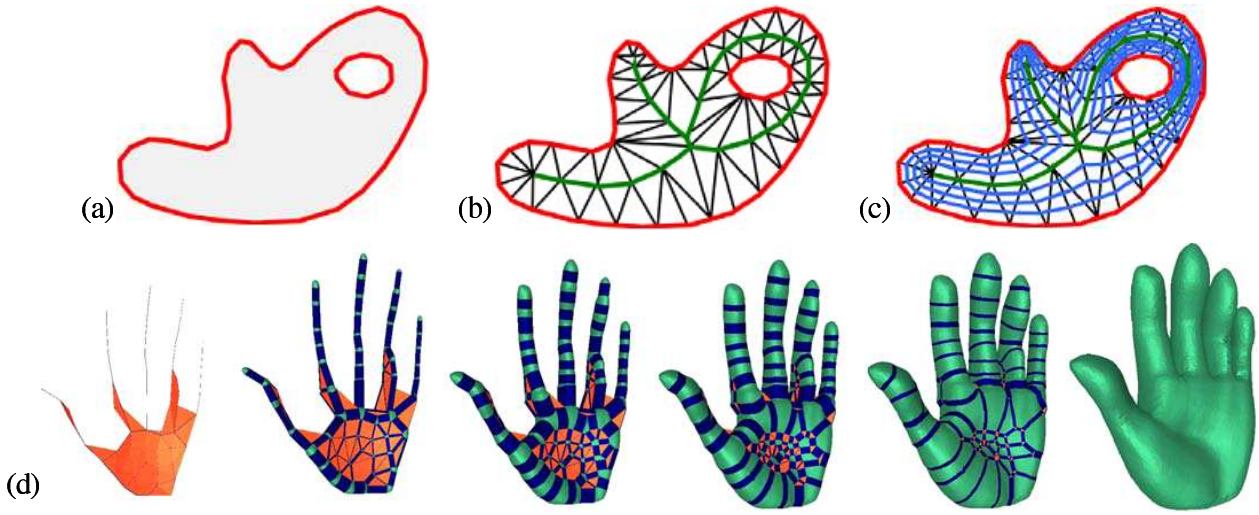


Figure 2.10: (Adapted from [172, (a-c): Fig.3, (d): Fig.14].) A illustration of A. Shamir and A. Shaham's *'pair-mesh'*, a tetrahedral complex representation based on approximated $\mathcal{MA}$ in 2D (a-c) and in 3D (d). The pair-mesh of a 2D model in (a) is a triangle complex in (b), where each triangle connects the inner Voronoi skeleton and the restricted Delaunay boundary. (c) depicts several iso-parametric poly-lines inside the object. (d) A family of parametric meshes interpolate between the $\mathcal{MA}$ and the hand object at several iso-parametric values t of: $0.01, 0.2, 0.4, 0.6, 0.8,$ and 1.0, where $t \in [0, 1]$; for $t \to 0$, the node resides on the inner Voronoi vertex; when $t = 1$, the node resides on the boundary.

the regularization process. We note that regularizing the $\mathcal{MA}$ topology may require complicated re-computation of the $\mathcal{MA}$ (locally or globally) and remains challenging in 3D. The major difference of our approach in comparing to others in regularizing the $\mathcal{MA}$ is that we propose a practical solution to avoid the above difficulty (by transforming the $\mathcal{MS}$). Also, our approach is based on an analysis of $\mathcal{MA}$ transitions (define the transforms) to explicit simplify the $\mathcal{MS}$ structure (Chapter 5.4.1). The second difference in comparison is on the smoothing of the $\mathcal{MA}$ rib curves. Note that only a few works in the literature address the problem of how to extract a smooth *boundary curve* of the $\mathcal{MA}$. These are the *rib* curves [87] where the $\mathcal{MA}$ starts to form in the grass-fire propagation (see § 3.1.1). Such rib curves are unstable (noisy in practice), due to its high order of symmetry (A_3), and need a proper regularization to retain its feature to capture sharp features of the shape. Our approach for a reliable rib curve extraction will be discussed in Chapter 7.

2.3.2 Survey of the 3D curve-like skeleton computation

In comparison to the above approaches which aim to extract a sheet-like medial structure out of a 3D shape, a different branch of research extracts a 1D *curve skeleton* ($\mathcal{CS}$) out of a 3D shape, instead of the 2D medial sheets. The $\mathcal{CS}$ is more simplified than the $\mathcal{MA}$, but its mathematical formulation is still an ongoing issue [54], *i.e.*, it is not *a priori* clear how to summarize the 2D $\mathcal{MA}$ sheets into a lower dimensional structure. Notably, Dey and Sun [66] have related the $\mathcal{CS}$ and the $\mathcal{MA}$ via a *geodesic* function on the medial sheets, Figure 2.15. We summarize recent works on the computing the $\mathcal{CS}$ for 3D mesh models [155, 210], voxelized data [55], and medical models (which often contain noise) [97, 35] in Figure 2.14. Also, we summarize Dickinson *et al.*'s use of the $\mathcal{CS}$ in matching in Figure 2.16.

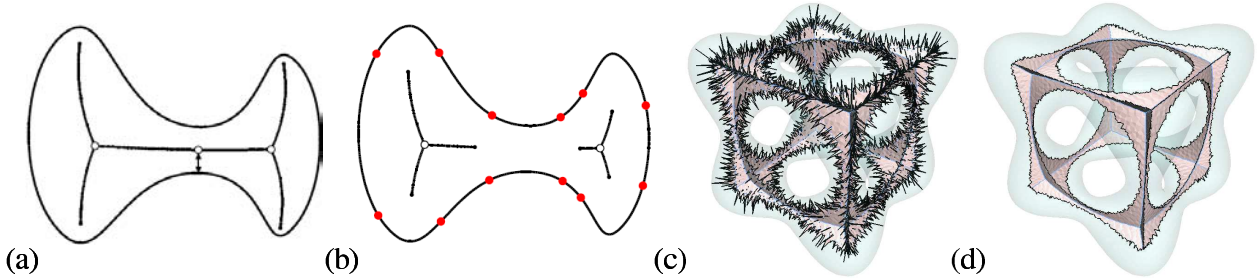


Figure 2.11: (Adapted from Attali *et al.*'s survey [12, (a-b): Fig.9, (c-d): Fig.11] of the λ-MA [47].) (a) A 2D shape with *weak feature size indicated* by the arrow. (b) The λ-MA of (a) for a value of λ greater than the weak feature size, which is broken into two parts. Each endpoint of the λ-MA has two closest points on the boundary, whose distance from each other is 2λ. In the 3D case, (c-d) shows two λ-MA of the same shape, with λ increasing from (c) to (d), constructed as a subset of the λ-Voronoi graph of a sample of the boundary. The $\mathcal{MA}$ is regularized by increasing λ to a proper value (*w.r.t.* the feature size and sampling of the shape.

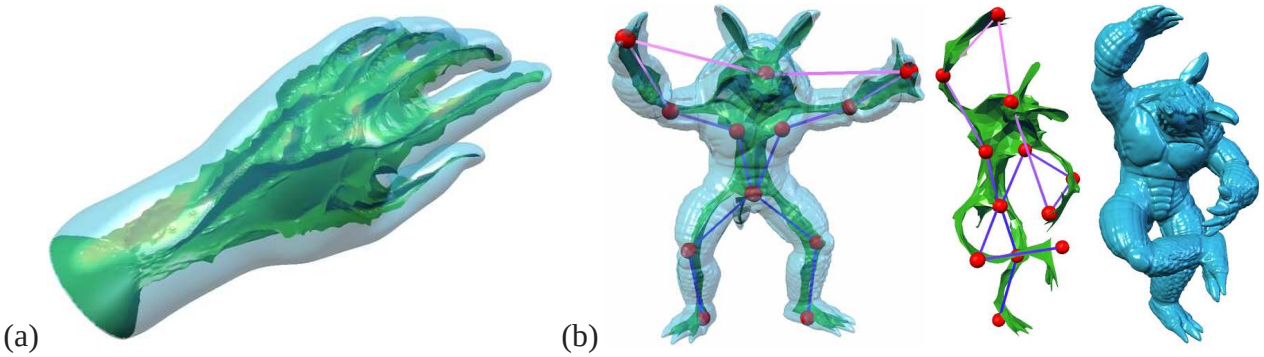


Figure 2.12: (Adapted from [222] and [219, Fig.1].) The 3D $\mathcal{MA}$ computed and regularized by S. Yoshizawa, A. Belyaev, and H.-P. Siedel. Instead of representing the 3D $\mathcal{MA}$ as a non-manifold sheets intersected at medial seams, the $\mathcal{MA}$ is represented a two-sided non-intersecting 2-manifold surface, thus standard mesh processing techniques can be applied to deform the $\mathcal{MA}$ to produce animation.

* Difficulties of extracting the curve-like skeletons for modeling and matching.

While the curve-like skeletons provide a 3D graph-like descriptions of shapes, they have several important limitations: *(i)* The resulting skeletons are often over-simplified and do not always capture essential geometric features such as surface ridges. Also the overall structure is not always captured, (*e.g.*, the center of the palm of the hand can not be intuitively described by a 'curve'). In addition, in relating the $\mathcal{CS}$ and the 3D shape, unlike the case of the $\mathcal{MA}$ that it can be closely coupled with the 'parts' of the shape, [172]), the association of 'parts' of the shape to the $\mathcal{CS}$ is non-trivial and depends on the definition of the $\mathcal{CS}$, *e.g.*, see results in [155, 55]. *(ii)* In many approaches, a well-segmented 3D object (with a closed surface mesh or a volumetric voxelization) is typically required. Some methods further require user-specified surface features which serve as end-points of the skeleton. *(iii)* The lack of a consistent mathematical definition increases the difficulties in matching that there is no clear understanding of how to deal with skeletal graphs having different local topologies yet representing perceptually similar shapes.

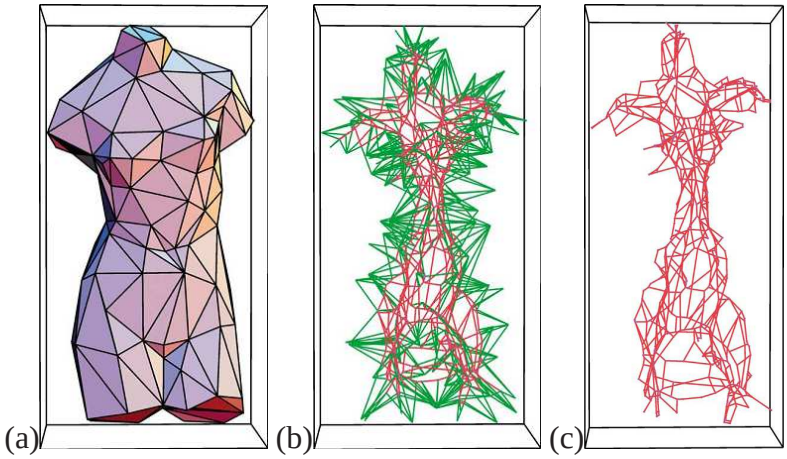


Figure 2.13: (Adapted from [56, Fig.13].) The exact $\mathcal{MA}$ of the Venus de Milo sculpture (in low resolution) computed by T. Culver, J. Keyser, and D. Manocha. The $\mathcal{MA}$ of this asymmetrical polyhedron in (a) contains only non-degenerate *seam* curves and sheets. (b) shows the seam curves as line segments. (c) shows the "central" seams, those that do not have an endpoint on the boundary.

2.4 Survey of Graph Matching for Recognition

* Overview on the use of graph matching for recognition.

Graphs are powerful data structures which describe the relationship among abstracted structure. Object representation through graph and hypergraph is very useful and popular for matching and recognition. For overview of literature in graph matching, refer to the survey in [51] and articles in [68, 81, 40]. Shapiro and Haralick [173] were among the pioneers in their use in structural description of objects in images via weighted graphs. Fu [77] used attributed relational graphs (ARG's) to describe parametric information as a basis of a general image understanding system based on extracting and matching hierarchical ARG's. Eshera and Fu [75, 76] found the best inexact match between two ARG's by minimizing the overall distance between the two graphs, defined as the incremental distance between corresponding nodes and links. Minimization of distance was converted into a shortest path problem over the directed acyclic branch-weighted lattice from the initial state to a state in the set of final states. This is solved by dynamic programming in a time linearly proportional to the number of lattice's states, which grows very rapidly with the number of nodes.

* Computational intractability and approximated methods.

The computational intractability of graph matching as an NP-complete problem has led to the development of several classes of algorithms. *(i) Search-Oriented* methods explore the shortest path in the state space, *e.g.*, via branch and bound methods [122]. These methods require heuristics to reduce exponential time complexity (worst case) to a low-order $(2, 3)$ polynomial in the number of nodes(l) and links(m). *(ii)* Another class of algorithms is based on *nonlinear optimization* which does not explicitly search the state space and its computational complexity is typically linear in the number of nodes/links, *e.g.*, *relaxation labeling* [104]. However, these methods only enforce *one-way* constraints. Eigenvalue decomposition works well when a pair of weighted graphs are nearly isomorphic, and while their combination with hill-climbing improves matching, poor local minima can often result [204]. *(iii)* Other types of techniques for graph matching include the use of neural networks [185], linear programming [4] Lagrangian Relaxation [165], and indexing the graph spectra [179].

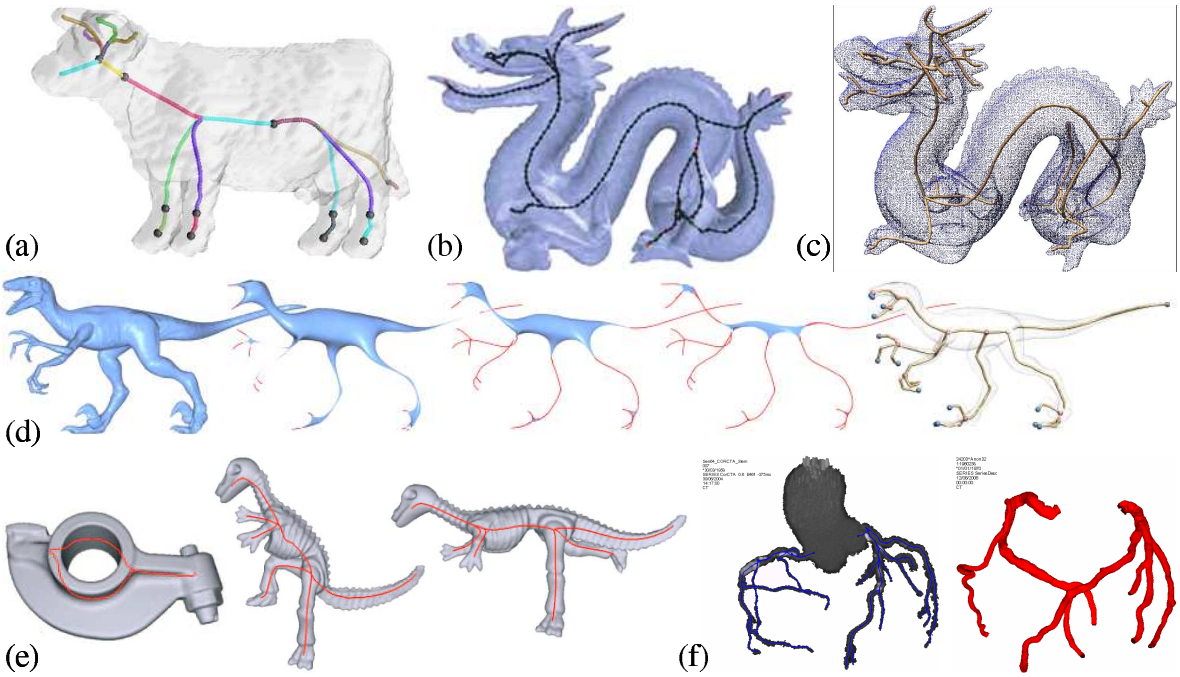


Figure 2.14: Recent results on the *curve skeleton* ($\mathcal{CS}$) computation. (a) (Adapted from [55, Fig.7].) The hierarchical $\mathcal{CS}$ computed by N. Cornea *et al.* based on a topological analysis on a *repulsive force field* over the object. (b) (Adapted from [216, Fig.19].) The *domain connected graph* computed by Ouhyoung *et al.* by combining three main ideas in skeleton extraction: the medial axis transform, generalized potential field, and decomposition-based methods. (c) (Adapted from [174, Fig.5].) The filtered $\mathcal{CS}$ computed by A. Sharf *et al.* based on evolution of a deformable model. (d) (Adapted from [155, Fig.1].) The $\mathcal{CS}$ extracted by Au and Lee *et al.* via geometry contraction and connectivity surgery. (e) (Adapted from [210, Fig.9,10].) The $\mathcal{CS}$ computed by Wang and Lee *et al.* by shrinking and thinning a volumetric model followed by pruning noisy branches. (f) (Adapted from [97, Fig.11].) The centerline and surface model of a coronary artery tree computed by H. Tek *et al.* based on a minimization of medialness measurements in a graph-based optimization framework.

* Rangarajan's graduated assignment algorithm.

The *graduated assignment* ($\mathcal{GA}$) [95] is a relaxation-based energy-minimizing graph matching algorithm based on a technique called "softassign" [186] in solving the assignment problem. It has been used in matching 2D shock graphs in [175]. On recent improvements of the $\mathcal{GA}$ is the work of Zass and Shashua [224], who improve the $\mathcal{GA}$ by applying a statistical global optimization via an iterative successive projection algorithm to match hypergraphs. We will elaborate more on the $\mathcal{GA}$ in Chapter 9 in extending it to match the $\mathcal{MS}$ hypergraphs.

Table 2.1 summarize main abbreviated symbols used in this thesis.

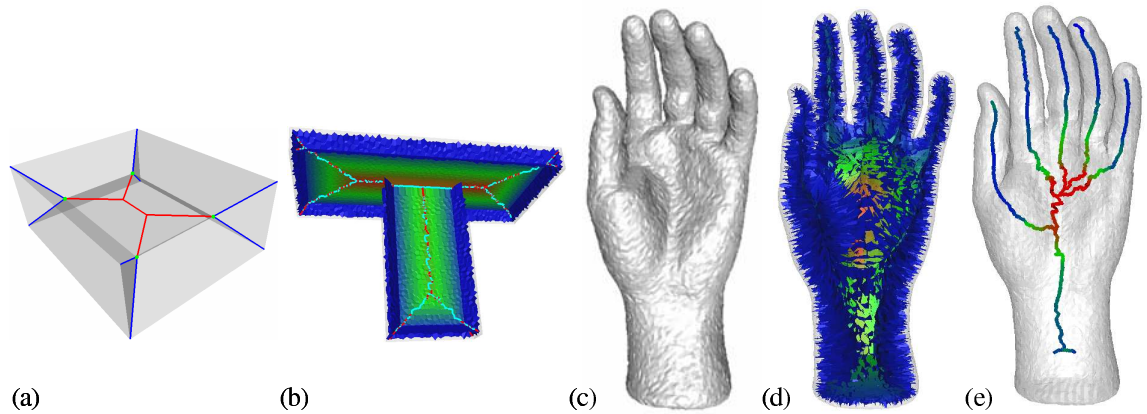


Figure 2.15: (Adapted from [66, (a): Fig.2, (b): Fig.9, (c-e): Fig.6].) Relating the *curve skeleton* ($\mathcal{CS}$) and the 3D $\mathcal{MA}$. The $\mathcal{CS}$ computed by T. Dey and J. Sun [66] by using a *medial geodesic function* (MGF) defined on the 3D $\mathcal{MA}$. (a) The 3D $\mathcal{MA}$ of a rectangular block, where the singularities of the MGF on the middle sheet (in red) are used to define the $\mathcal{CS}$. (b) The $\mathcal{MA}$ of a T-shape rendered with the MGF values in color, where the skeleton edges (cyan) are collected during the erosion. (c-e) A noisy hand model, the $\mathcal{MA}$ rendered with the MGF values, and the extracted $\mathcal{CS}$.

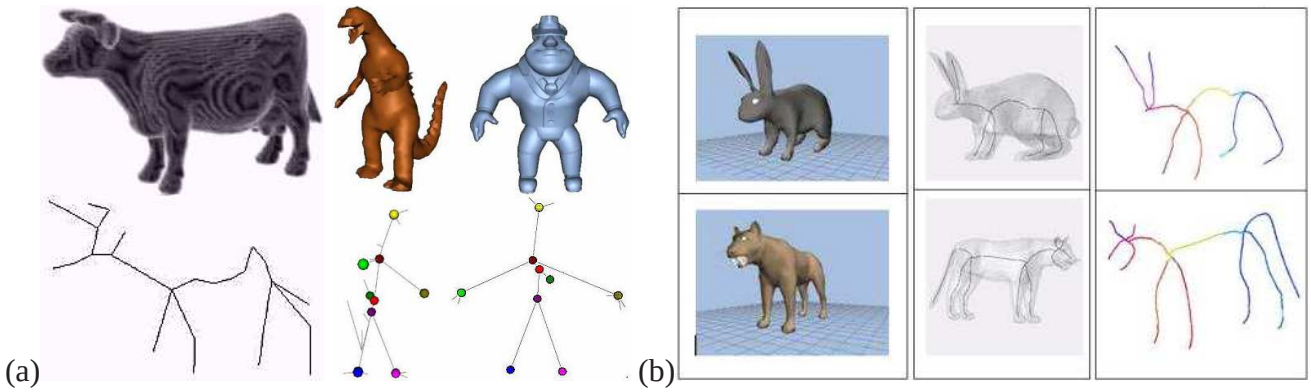


Figure 2.16: Matching *curve skeletons* ($\mathcal{CS}$) for 3D shape matching. (a) (Adapted from [192, Fig.3,7].) Skeletal stick graphs computed by H. Sunder, D. Silver and S. Dickinson *et al.* for matching. (b) (Adapted from [53, Fig.2].) Curve skeletons computed by D. Cornea, D. Silver and S. Dickinson *et al.* for similarity matching. T he correspondence is shown in color code.

Table 2.1: Main abbreviated symbols used in this thesis.

Symbol	Description
$\mathcal{G}$	Input generators (points) $\{G_i\}$
$\mathcal{SS}$	Symmetry Set
$\mathcal{MA}$	Medial Axis
$\mathcal{SG}$	Shock Graph (in 2D)
$\mathcal{MS}$	Medial Scaffold
$\mathcal{SC}$	Shock Scaffold

Symbol	Description
$\mathcal{VD}$	Voronoi Diagram
$\mathcal{FC}$	Flow Complex
$\mathcal{CS}$	Curve Skeleton
m-Rep	The *medial atom* representation
$\mathcal{RG}$	Reeb Graph

Chapter 3

A Review of the Medial Scaffold

* Overview of chapter: reviewing the A_k^n notation and the $\mathcal{MS}$ / $\mathcal{SC}$ hierarchies.

This chapter reviews the **Medial Scaffold** ($\mathcal{MS}$) and the **Shock Scaffold** ($\mathcal{SC}$), the hierarchical graph representations of the 3D $\mathcal{MA}$ originally proposed by Leymarie and Kimia [125, 128] based on a theoretical analysis of Giblin and Kimia [87]. We review key ideas and terminologies such as the contact typology (A_k^n) and shock typology (A_k^n-l). We also revise the organization of the $\mathcal{MS}$/$\mathcal{SC}$ hierarchy to reflect a practical implementation of the $\mathcal{MS}$ hierarchy. In addition, we focus on an important case when the shape is specified as an unorganized point set and classify the $\mathcal{MS}$/$\mathcal{SC}$ sheets, curves, and nodes based on a shock flow analysis. We show that the shock flow field of the medial sheets can be recovered and thus recover the (more informative) $\mathcal{SC}$ from a (less informative) $\mathcal{MS}$ for a point-sampled shape. Finally, we point out a future direction on deriving a complete shock flow analysis of the $\mathcal{MS}$ to lead toward an implementation of the $\mathcal{SC}$ hierarchy.

* Organization of chapter.

This chapter is organized as follows. Section 3.1 reviews the mathematical definition of the 3D $\mathcal{MA}$ (by Giblin and Kimia [87]) by contacting a sphere and defining the contact typology of the MA points using the A_k^n notation. This leads to a complete analysis of the 3D $\mathcal{MA}$ local form and an organization toward the $\mathcal{MS}$ (§ 3.1.1). § 3.1.2 further reviews the shock flow analysis of the 3D $\mathcal{MA}$ and define the flow typology (A_k^n-l) to classify the 3D shock points, leading to an organization toward a $\mathcal{SC}$. Based on the above context, Section 3.2 reviews Leymarie's organization of the $\mathcal{MS}$/$\mathcal{SC}$ into a hierarchy based on reducing the information and keeping only the essence structure [128, Ch.3], [125]. We revise this organization to reflect our practical experience in implementation of the $\mathcal{MS}$ hierarchy. Section 3.3 reviews a special case of the $\mathcal{MS}$ and $\mathcal{SC}$ when the input shape boundary consists of only sample points. This special case turns out to be significant that the $\mathcal{MS}$/$\mathcal{SC}$ can be regularized (filtered) to approximate the *true* $\mathcal{MS}$/$\mathcal{SC}$ as the sampling is dense enough, and in practice this leads to a practical computational approach of the $\mathcal{MS}$ detailed in Chapters 6 and 7. § 3.3.1 classifies the $\mathcal{MS}$ sheets, curves, and nodes of the point case based on the shock flow and presents a practical algorithm to implement such classification in each case. This enables to an approach detailed in § 3.3.2 to compute the $\mathcal{MS}$/$\mathcal{SC}$ from the Voronoi-based approaches (such as the QHull [15]) in recovering the missing flow information, in supplement to Leymarie's original flow-based computation of the $\mathcal{SC}$. Finally, Section 3.4 describes future directions to recover the shock flow of the $\mathcal{MS}$ in the general cases toward the $\mathcal{SC}$.

3.1 Local Form of Shock Points: Contact with Spheres and Shock Flow Analysis

* Overview of the symmetry set ($\mathcal{SS}$), $\mathcal{MA}$ and the shocks.

We briefly review Giblin and Kimia's discussion of the *symmetry set* ($\mathcal{SS}$) [38] , $\mathcal{MA}$ [31] and the *shocks* [89]. The $\mathcal{SS}$ is the closure of the loci of centers of spheres tangent to the surface at two or more points; such bi-tangent spheres are called "contact spheres." The $\mathcal{MA}$ is the subset of the $\mathcal{SS}$ for which all such spheres are *maximal*, *i.e.*, they do not contain any points of the surface other than the contact points. The *shock structure* arises from a "dynamic" interpretation of the $\mathcal{MA}$, as in Blum's grassfire [31], where the locus of singularities, or shocks, formed in the course of wave propagation from boundaries have an associated direction and speed of flow.

The above notion is consistent in both the 2D and 3D cases. For the simpler 2D case, shock segments are those segments of the $\mathcal{MA}$ which have monotonic flow. This grouping of shocks into segments is a more *refined* partition of the $\mathcal{MA}$ than by grouping between $\mathcal{MA}$ junctions and end-points [116]. Shocks are obtained in 2D either by detecting the singularities of the evolving boundary in a curve evolution (PDE) approach [116, 180] or in a mixed Eulerian–Lagrangian propagation which combines wave propagation and computational geometry concepts [202, 201]. The computation of the $\mathcal{MA}$ / shocks in 3D is more complex and together with the goal of making it accessible to applications is the main topic of this thesis. See Chapter 2 for a survey and Chapter 7 for our approach.

3.1.1 Contact typology of a shock point: the A_k^n notation

The notion of the $\mathcal{MS}$/$\mathcal{SC}$ relies on an understanding the 3D $\mathcal{MA}$ local form, *i.e.*, a classification of 3D $\mathcal{MA}$ points by Giblin and Kimia in 2D [89], and in 3D [87], based on osculating a tangent sphere with the shape boundary and studying the order of contact, Figure 3.1.

Definition 1 The A_k^n notation: *Let A_k^n denote the contact of a circle (in 2D) or a sphere (in 3D) osculating a shape boundary at n distinct points, each with $k{+}1$ degree of contact:* [1] *$k{=}1$ denotes a regular tangency at a contact point* p*; $k{=}2$ denotes a sphere of curvature (*i.e.*, the radius of contact sphere is equal to the radius of curvature, but* p *is not part of a ridge)* [2]*; $k{=}3$ denotes a sphere of curvature at a* ridge *point (*i.e.*, when the sphere of curvature is locally maximal, and in 3D,* p *is a curvature extremum along a principal curve); $k{=}4$ denotes a sphere of curvature at a* turning *point of a ridge (*i.e.*, where the ridge becomes tangent to the line of curvature at* p*) [98, Ch.6],[87]* [3]*; $k{=}5$ denotes a degenerate contact where two turning points merges (such that the degree 5 terms of the* Monge *form are involved, [98, 87, 39]* [4], etc.

Refer to Figure 3.2 for an example of *generic* contacts (*e.g.*, A_1^2, A_1^3, A_3, A_1^4, A_1A_3) in 3D. Among the A_k^n contact points, only odd orders of contact (*i.e.*, $k = 1, 3, 5, ...$) can contribute to the

[1] For the case of $n = 1$, we do not write the superscript and use the reduced notation A_k.

[2] The A_2 contact can be also viewed as the transition where two A_1 contact merges at a single point.

[3] From [87]: "At a turning point of a ridge, the ridge is itself tangent to the line of curvature, and the principal curvature has a degenerate extremum, in transition between maximum and minimum. For such as sphere, the distance from the center to the shape surface never has a minimum at the point of contact, the sphere can therefore never be a maximum sphere, thus never be a part of the $\mathcal{MA}$."

[4] The A_5 transition involves (from [98, §7.2.8]): two turning points are created or destroyed by a process in which a ridge remains non-singular but momentarily makes an inflection with the corresponding line of curvature.

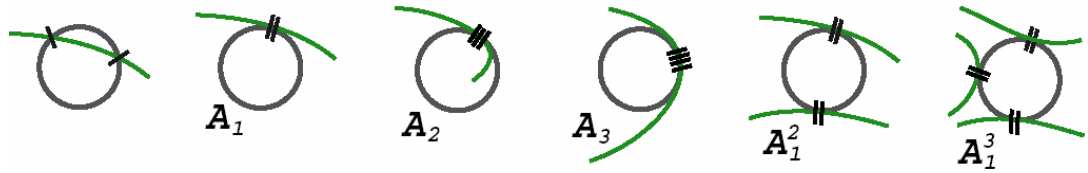


Figure 3.1: (From [87, Fig.3].) Illustration of the notation A_k^n based on contact of curves with a circle in 2D [87]; $k+1$ counts order of contact (indicated by straight short dark segments); *e.g.*, A_1 is regular tangent contact, A_2 is regular "curvature" contact, A_3 is a curvature maximum contact. The superscript n counts the number of contact points, so that A_1^2 means two A_1 contacts. A similar definition holds for the contact of surfaces with spheres in 3D.

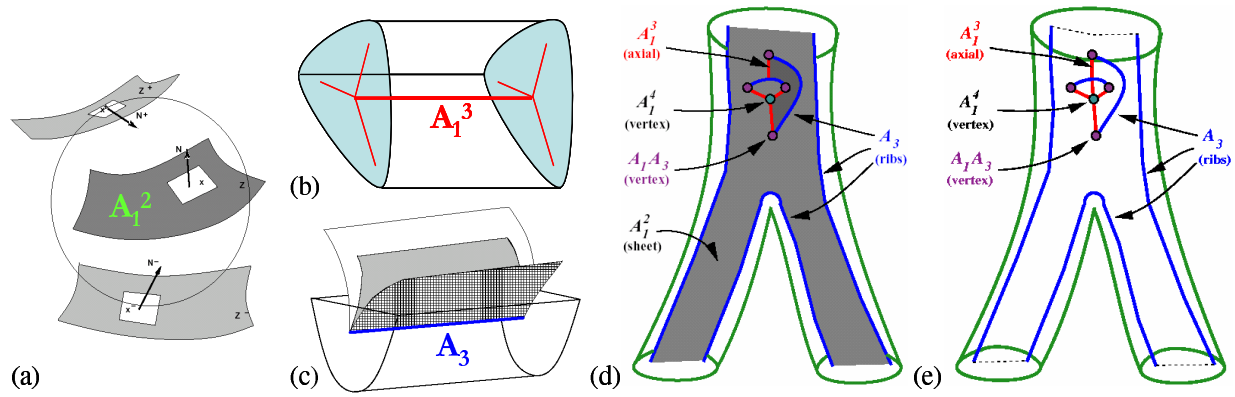


Figure 3.2: (From [125, Fig.5].) Summary of the *five* generic types of 3D $\mathcal{MA}$ points [87, 125]: (a) Each point on an A_1^2 sheet is the center of a sphere with two ordinary A_1 contacts. (b) Each point on an A_1^3 axial curve is the center of a sphere with three ordinary A_1 contacts; this is where three sheets come together. (c) Each point on an A_3 rib curve is a limiting case of two A_1^2 points coming in unison; it corresponds to the 'boundary' of the $\mathcal{MA}$ sheet which is associated to the *ridges* on the shape boundary. The *five* generic types are illustrated on a $\mathcal{MS}$ hypergraph in (d) and in (e) with the medial sheets implicit to emphasize its graph structure.

$\mathcal{MA}$, that is, being the center of a *maximal* sphere (without intersecting the shape surface). Other *non-generic* higher-order contacts (such as A_5) require further elaboration and become significant in man-made and biological objects which have spherical areas, for example. These become apparent after we apply the shock transforms, refer to Chapter 5 for further analysis.

* Summary of Giblin and Kimia's organization of 3D $\mathcal{MA}$ points [87].

Giblin and Kimia organized all *generic* 3D $\mathcal{MA}$ points, based on their order of contact A_k^n, into **five** principal types of shock points, Figure 3.2:

1. A_1^2 *contact* (**sheet**), where at each point, the contact sphere has two ordinary A_1 contacts. By adjusting the radius and rolling this sphere between two surface patches, we encounter other A_1^2 points such that the local form of A_1^2 is a medial *sheet* which is locally smooth.
2. A_3 *contact* (**rib curve**), where at each point, the contact is the limiting case when two A_1^2 points come together. The set of A_3 points form a space curve spanning centers of one of the principal curvatures. This space curve is a *rib* curve bordering an A_1^2 sheet, and the associated contact points correspond to surface *ridges*. [5]

[5] We will present a ridge detection approach based on this notion in § 10.2.

3. A_1^3 *contact* (**axial curve**), where at each point, the contact sphere has three ordinary A_1 contacts. The local form is a space curve where 3 A_1^2 sheets meet together.
4. A_1A_3 *contact* (**rib end**): This is where the sphere has A_1 contact (*i.e.*, ordinary tangency) at one point and A_3 contact at another. The local form is an isolated point where a pair of A_1^3 and A_3 curves meet and terminate.
5. A_1^4 *contact* (**axial end**): This is where the contact sphere has four ordinary contacts. The local form is also an isolated point which is at the intersection of *six* smooth A_1^2 *sheets* of the $\mathcal{MA}$, or alternatively, at the intersection of *four* A_1^3 *curves*, *i.e.*, either six distinct pairs ($C_2^4 = 6$) or four distinct triplets ($C_3^4 = 4$) from the four contact points.

In summary, the $\mathcal{MA}$ points naturally organize into sheets, curves, and isolated points. A_1^2 points are interior points of a medial sheet. Each sheet is bounded by a collection of A_1^3 and A_3 curves. Each A_1^3 curve ends at either an A_1A_3 or A_1^4 point. Each A_3 curve must end at an A_1A_3 point. Note that **throughout the thesis,** A_3 **and** A_1^3 **curves are shown in blue and red**, respectively.

* Degenerate configurations observed in practice.

While the above five types of $\mathcal{MA}$ points are *generic* in 3D, in practice we observe more *degenerate* types of $\mathcal{MA}$ points. In particular as we transform the $\mathcal{MS}$ by moving it toward higher order of degeneracy, we produce more degenerate $\mathcal{MA}$ points (Chapter 5). We list three types of degeneracies we observed in practice. Further analysis requires to locally perturb these degenerate configurations to decompose them into general configurations and will be elaborated in § 5.5. [6]

1. A_1^n *axial curve*, $n > 3$. This occurs when more than three boundary points are co-circular to make their central axis degenerate. An A_1^4 axial curve can be perturbed to decompose into two A_1^3 curves, see Figure 5.21(e). This degeneracy is in fact necessary if we are to deform a shape towards a circular cylinder as in many tubular shapes in applications, *e.g.*, vessels, bronchial airway, colon, *etc*.
2. A_1^n *point* **(axial end)**, $n > 4$. This occurs when more than four boundary points are co-sphere to make their center degenerate. An A_1^5 point can be perturbed to decompose into two A_1^4 points, see Figure 5.21(j). Also, the end point of the above A_1^n degenerate axial curve is an A_1^{n+1} degenerate point. This degeneracy is in fact necessary if we are to deform a portion of a shape towards a spherical shape.
3. $A_1^mA_3$ *point* **(rib end)**, $m \geq 2$. This occurs when more than one A_1 contact happens at an ordinary A_1A_3 point. An $A_1^2A_3$ point can be perturbed and decomposed into an A_1^4 and an A_1A_3 point, Figure 5.21(k). This degeneracy is in fact necessary if we are to deform a portion of a shape towards a spherical shape on one side and with a bump-like perturbation on the other side.

We conclude the degenerate $\mathcal{MA}$ analysis by two remarks. *(i)* First, while the degenerate A_1^n (n>3) axial curve is possible, we do not observe any degenerate (A_5 or more) rib curve in practice, since by definition the A_5 should be an isolated point [88]. Another reason A_5 does not appear is that we work on point-sampled shapes, thus the A_3 is sufficient (and does not require further degeneracy other than isolated singular points). *(ii)* Second, a degenerate configuration not yet mentioned is at the corner of a shape, see Figure 5.20, where in a perfect case three A_3 rib curves and one A_1^3 axial curve meet. This is an interesting degeneracy whose local form is not understood so far (an ongoing work with Peter Giblin *et al.*). Refer to § 5.4.1 and § 10.3.1 for further exploration.

[6] The shock transforms defined in Chapter 5 will be used to decompose the $\mathcal{MA}$ degeneracy.

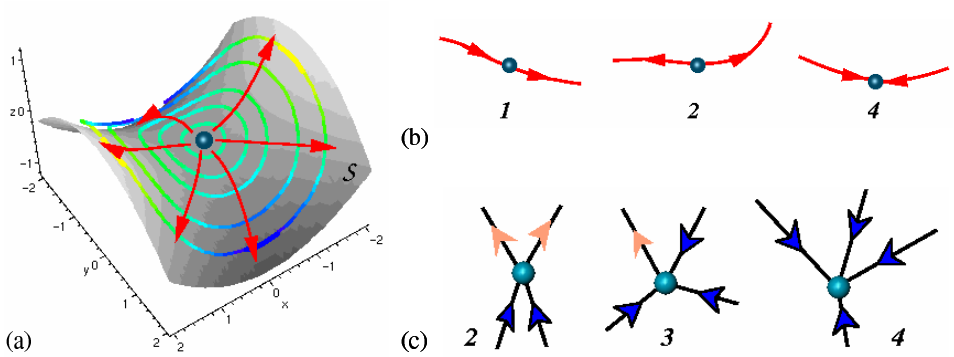


Figure 3.3: (From [125, Fig.4].) (a) Generic situation for an A_1^2 shock sheet: from the A_1^2-2 point a 3D surface radially grows-out; arrows indicate some directions of increasing radius values. (b) Flows at A_1^3 and A_3 shock curves. (c) Flows at A_1^4 shock points, where the number of *inward* flows is indicated.

3.1.2 Flow typology of a shock point: the A_k^n-l notation

* Summary of Leymarie and Kimia's 3D shock flow analysis [125].

In addition to the above classification of 3D $\mathcal{MA}$ points based on the order of contact and the local form, Leymarie and Kimia further explored the notion of **shock flow** for each $\mathcal{MA}$ point in the direction of increasing radius, r, of its associated maximal contact spheres and lead to a *finer* classification of the $\mathcal{MA}$ in the form of a shock structure both in 2D [116, 89], and in 3D [125, 87]. The shock flow is a *vector field* defined on the $\mathcal{MA}$ whose typology we denote via l, Figure 3.3. Specifically, the shock points can be (older classification [125] is in parentheses:

1. **regular** (first-order: l=1), monotonically flowing along a path,
2. acting as **sources** and initiating flow (second-order: l=2),
3. acting as **relays** or **saddles**, where shocks simultaneously flow in and out (third-order: l=3), or
4. acting as **sinks** and terminating flow (forth-order: l=4).

For A_1A_3 vertices where a pair of A_3 and A_1^3 curves meet, three of four flow configurations are possible [87]: *(i)* both curves can flow outward (*i.e.*, a source); *(ii)* the A_1^3 can flow outward and the A_3 flow inward (*i.e.*, a relay); *(iii)* both curves flow inward (*i.e.*, a sink). The configuration where A_1^3 is flowing in and A_3 is flowing out is not possible.

For A_1^4 vertices, the classification is based on the *number* of *inward flows* as dictated by the four intersecting A_1^3 curves, where either two, three or four curves can flow inward, Figure 3.3c, ruling out other configurations.

We summarize Leymarie and Kimia's classification of 3D shock points in Table 3.1. On the left is the table classifying the **18** *generic shock points* in the notation A_k^n-l (contact typology A_k^n and shock flow type l). On the right is the organization of the 18 types into *(i)* 3 *regular* types of shock points which compose of the shock sheets (A_1^2) and shock curves (A_1^3 and A_3) and *(ii)* 15 *singular* types of shock nodes (A_1^2, A_1^3, A_3, A_1A_3, A_1^4).

* Revising the A_k^n-l notation for the *axial-end relay* shocks.

We revise the original notation of the two types of the *axial-end relay* shocks in [125]. Specifically, the A_1^4-3a (was A_1^4-3) and A_1^4-3b (was A_1^4-2) are both of the "relay" type and l should be 3. (Refer to Figure 3.8 for an illustration of their local configuration.)

Shocks	Regular	Source	Relay	Sink
Sheet	A_1^2-1	A_1^2-2	A_1^2-3*	A_1^2-4
Rib	A_3-1	A_3-2	A_3-3*	A_3-4
Axial	A_1^3-1	A_1^3-2	A_1^3-3*	A_1^3-4
Rib end	-	A_1A_3-2	A_1A_3-3	A_1A_3-4
Axial end	-	-	A_1^4-3a, A_1^4-3b	A_1^4-4

* Degeneracies, *i.e.*, part of A_1^2-3, A_3-3 and A_1^3-3, are special cases of relay where shocks flow simultaneously in and out.

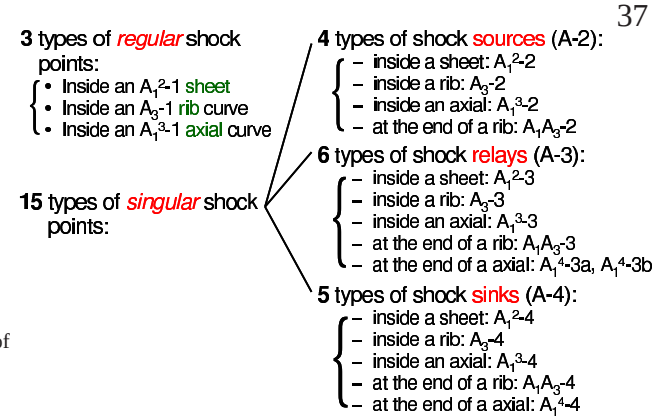


Table 3.1: (Adapted from [125, Table.2].) Summary of Leymarie and Kimia's classification of **18** types of shock points based on their contact typology (A_k^n) and the shock flow typology. There are 3 *regular* shock types (with monotonic flow), and 15 *singular* shocks (which are the *sources*, *relays* and *sinks* for the flow).

* Comparison of terminology with the Morse analysis and the "Flow Complex".

Our classification of the 3D $\mathcal{MA}$ and shock types is closely related to Giesen *et al.*'s Flow Complex ($\mathcal{FC}$) [92], which is based on a *Morse analysis* of the distance function. The main difference is that ours is based on a singularity analysis and results in a richer set of classification. Specifically, some "relay" types of shocks are not *critical* points in the Morse analysis, which makes the subsequent analysis (classification and algorithm) different. Refer to Chapter 2 for a detailed comparison.

The terminology we use to describe the shock flow in the $\mathcal{MS}$/$\mathcal{SC}$ is also different, which we explicitly address here to remove confusion. In our notation, all source/relay/sink are with respect to the corresponding shock entity (*i.e.*, the A_k^n object). For example, the A_1^3-2 is a 'source' point on the A_1^3 shock curve, despite the fact that the shock flow at an A_1^3 point is originated from a nearly A_1^2 shock sheet (which is further from the A_1^2 source on the sheet). In the $\mathcal{FC}$, the A_1^2 is called a 'saddle' point in the Morse analysis (in comparing to our 'source' point). This difference in terminology also applies to the A_3-2 and A_1A_3-2 shock points in Figure 3.1, where our A_3-2 and A_1A_3-2 points are called 'saddle' points in the $\mathcal{FC}$. We do not observe any other major differences in considering the $\mathcal{FC}$ analysis in our framework as a sub-structure of the $\mathcal{SC}$.

3.2 The Medial Scaffold ($\mathcal{MS}$) and Shock Scaffold ($\mathcal{SC}$) Hierarchies

* Review of Leymarie's $\mathcal{MS}$ and $\mathcal{SC}$ hierarchies [125].

We now review Leymarie and Kimia's organization of the 3D $\mathcal{MA}$ into a hierarchy of the $\mathcal{MS}$ and $\mathcal{SC}$ representations. The idea is to first organize all information (topology, geometry, shock dynamics) of the $\mathcal{MA}$ together, and then drop the detailed information from the complete representation toward a reduced one, while retaining the qualitative structure of the shape.

In the most detailed representation, the **shock scaffold** ($\mathcal{SC}$) is a directed hypergraph representation of the shape. The key insight is that slight deformations of the shape do not generally affect the topological relations among the special shock points A_1^4, A_1A_3, A_1^3-2, A_3-2, as indicated by the shock curves A_1^3 and A_3 connecting them, while the geometry and flow dynamics of the medial sheets and curves can change. By dropping all shock flow information from the $\mathcal{SC}$, the **medial**

scaffold ($\mathcal{MS}$) is then an un-directed hypergraph. [7] The term "scaffold" is used in analogy to building constructions where a set of metallic beams support relatively weaker building materials, so as to indicate the relative significance of curves over sheets, and points over curves, in describing the *qualitative shape* of an object [125].

* Defining the shock scaffold ($\mathcal{SC}$) hierarchy.

(Revised from Leymarie [125].) The typology described in the previous section is required for a construction of a *hypergraph* [20] in analogy to the notion of a *shock graph* in 2D [182]. We identify *flow singularities* and organize these in three classes: (1) ***shock sources***: *(a)* A_1^2-2 along a sheet, *(b)* A_1^3-2 and A_3-2 along a curve, and *(c)* A_1A_3-2 at a vertex; (2) ***shock relays***: *(a)* A_1^2-3 along a sheet, *(b)* A_1^3-3 and A_3-3 along a curve, and *(c)* A_1^4-3a, A_1^4-3b and A_1A_3-3 at a vertex; and (3) ***shock sinks***: *(a)* A_1^2-4 along a sheet, *(b)* A_1^3-4 and A_3-4 along a curve, and *(c)* A_1^4-4 and A_1A_3-4 at a vertex. The remaining shocks act as linking structures consisting of (1) A_1^2-1 along a sheet and (2) A_1^3-1 and A_3-1 along a curve. The structure which embeds this hierarchical notion is referred to as the **shock scaffold hypergraph** ($\mathcal{SC}^{\mathcal{H}}$), where the above shock sources, relays, and sinks serve as its **nodes**, the shock curves act as its **links** between nodes, and shock sheets act as **hyperlinks** bringing together several links and nodes.

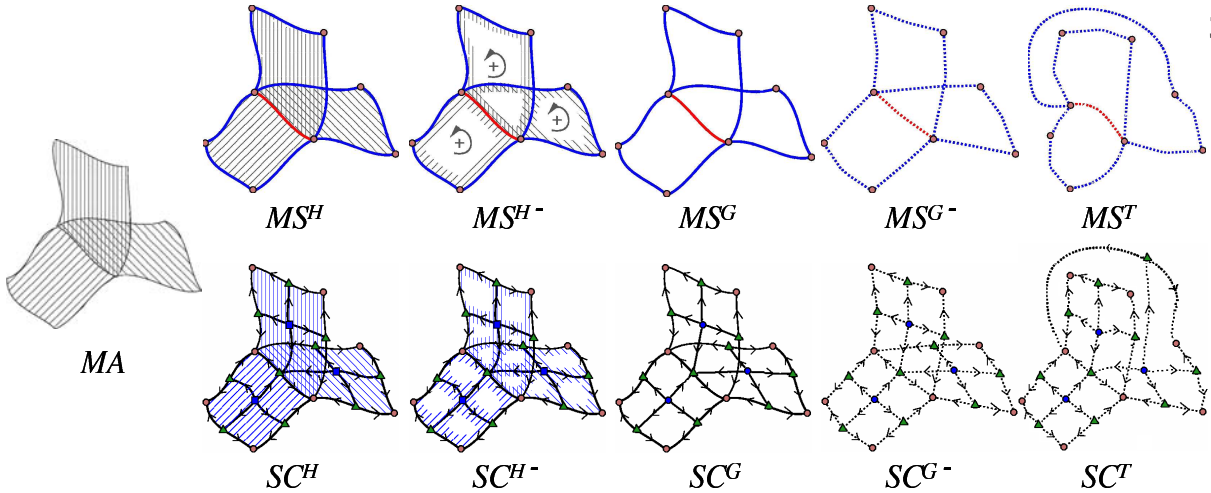
(Revised from Leymarie [125].) We now define the *"reduced"* representations of the $\mathcal{SC}^{\mathcal{H}}$ to form a hierarchy as follows, Figure 3.4.

1. The dropping of all shock sheet attributes (geometry and dynamics), except the topology, of the $\mathcal{SC}^{\mathcal{H}}$ is named the **reduced shock scaffold hypergraph** ($\mathcal{SC}^{\mathcal{H}-}$). The idea in keeping the sheet topology (which can be completely stored in the shock curves) is significant that the topology captures the global structure of a sheet, where the detailed geometry/dynamics of the sheet are not salient nor robust and can be approximated from the shock curves bordering the shock sheet.
2. The dropping of all sheet topology of the $\mathcal{SC}^{\mathcal{H}-}$ gives the **shock scaffold graph** ($\mathcal{SC}^{\mathcal{G}}$), which is a (1D) graph structure, which summarizes the $\mathcal{MA}$ in the next level.
3. The dropping of all shock curve attributes (geometry and dynamics), except the graph topological connectivity, yields the **reduced shock scaffold graph** ($\mathcal{SC}^{\mathcal{G}-}$), which is a yet simplier graph. Similarly, the missing shock curve geometry can be approximated from their ending shock nodes.
4. Finally, the dropping of all geometry info (stored in the graph nodes) of the $\mathcal{SC}^{\mathcal{G}-}$ gives the **topological shock scaffold** ($\mathcal{SC}^{\mathcal{T}}$), where only the topological graph structure is preserved.

The significant idea of Leymarie and Kimia [125] in extracting a graph/hypergraph structure over the "classical" trace of the $\mathcal{MA}$ is that *"the $\mathcal{MA}$ information is organized into groups and the connectivity between them is specified. It is precisely the connectivity among these groups which contains the qualitative information"*, while the remaining information allows for an exact reconstruction or an approximation of the shape from the shock structure [86, 87]. It is this qualitative structure that the applications require.

* Defining the medial scaffold ($\mathcal{MS}$) hierarchy.

[7] We make a distinction from a "shock point" to a "medial point". A "shock point" is a $\mathcal{MA}$ point equipped with the topology, geometry, and shock dynamics, *etc.* in the $\mathcal{SC}$, while a "medial point" is a superset of the shock points, which is equipped with topology and geometry, but dynamics.



Level	$\mathcal{MS}$ Symbol	$\mathcal{SC}$ Symbol	Distinctive features
I	$\mathcal{MS}^{\mathcal{H}}$	$\mathcal{SC}^{\mathcal{H}}$	Medial nodes, curves & sheets.
II	$\mathcal{MS}^{\mathcal{H}-}$	$\mathcal{SC}^{\mathcal{H}-}$	Sheets stripped of geometry, *etc.*
III	$\mathcal{MS}^{\mathcal{G}}$	$\mathcal{SC}^{\mathcal{G}}$	Graph: medial nodes & curves.
IV	$\mathcal{MS}^{\mathcal{G}-}$	$\mathcal{SC}^{\mathcal{G}-}$	Curves stripped of geometry, *etc.*
V	$\mathcal{MS}^{\mathcal{T}}$	$\mathcal{SC}^{\mathcal{T}}$	Nodes & curves stripped of geometry, *etc.*

Figure 3.4: (Adapted and augmented from [125, Fig.6] and [128, Figs.3.8,3.9].) [Upper] From the "classical" $\mathcal{MA}$ static representation to the hierarchies of the *medial scaffold* ($\mathcal{MS}$) and *shock scaffold* ($\mathcal{SC}$) representations. The $\mathcal{MS}$ hierarchy captures the topology (inter-connectivity between sheets, curves, and nodes), while the $\mathcal{SC}$ hierarchy is further augmented (classified) with complete shock flow information and is a refined structure of the $\mathcal{MS}$ hierarchy in each corresponding level. Red dots correspond to $\mathcal{MA}$ vertices, *i.e.*, A_1^4 or A_1A_3. Green triangles correspond to shock sources of curves, *e.g.*, A_1^3-2 points, which are needed for capturing the $\mathcal{SC}$. A_1^3 and A_3 curves are shown in red and blue, respectively. [Lower] Summary of the $\mathcal{MS}$ and $\mathcal{SC}$ hierarchies comprising of five levels. The superscript $^{\mathcal{H}}$ stands for a hypergraph, $^{\mathcal{G}}$ stands for a graph, $^{\mathcal{T}}$ stands for a topological graph, and the minus sign "$^-$" stands for a *reduced* representation.

(Reviewed from Giblin and Kimia [125].) A parallel hierarchy of the above shock scaffolds can be similarly defined for the medial scaffolds, by taking out the shock flow classification of the $\mathcal{SC}$, Figure 3.4. Specifically, the **medial scaffold hypergraph** ($\mathcal{MS}^{\mathcal{H}}$) takes the isolated medial points (A_1A_3 and A_1^4) as its **nodes**, the medial curves (A_3 and A_1^3) as its **links** between nodes, and the medial sheet (A_1^2) as its **hyperlinks** bringing together several links and nodes.

(Revised from Leymarie [125].) The *"reduced"* representations of the $\mathcal{MS}^{\mathcal{H}}$ is defined similarly to form a hierarchy as follows, Figure 3.4.

1. The dropping of all medial sheet attributes (geometry), except the topology, of the $\mathcal{MS}^{\mathcal{H}}$ yields the **reduced medial scaffold hypergraph** ($\mathcal{MS}^{\mathcal{H}-}$). Similarly, the sheet topology can be completely stored in the bordering medial curves and the detailed geometry of the sheet are not significant and can be reasonably approximated.
2. The dropping of all sheet topology of the $\mathcal{MS}^{\mathcal{H}-}$ gives the **medial scaffold graph** ($\mathcal{MS}^{\mathcal{G}}$), which is a (1D) graph structure, which summarizes the $\mathcal{MA}$ in the next level.
3. The dropping of all medial curve attributes (geometry), except the graph topological connectivity, gives the **reduced medial scaffold graph** ($\mathcal{MS}^{\mathcal{G}-}$), which is a yet simplier graph. Again, the missing curve geometry can be approximated from their ending nodes.

4. Finally, the dropping of all geometry info (stored in the graph nodes) of the $\mathcal{MS}^{\mathcal{G}-}$ yields the **topological medial scaffold** ($\mathcal{MS}^{\mathcal{T}}$), where only the topological graph structure is preserved. We will elaborate how to efficiently handle the topology of the $\mathcal{MS}^{\mathcal{T}}$ in Chapter 4.

We note that the $\mathcal{MS}^{\mathcal{H}-}/\mathcal{MS}^{\mathcal{G}}$ is a geometric graph, *i.e.*, without self intersections, and is not a *tree* in general, *i.e.*, it contains *circuits* (chains of links forming closed loops) bounding $\mathcal{MA}$ sheets. Despite the lack of an explicit representation of sheets, the $\mathcal{MS}^{\mathcal{H}-}/\mathcal{MS}^{\mathcal{G}}$ alone gives a fairly good idea of the shape of the object due to the remaining connectivity. The $\mathcal{MA}$ can be approximated by interpolating the missing $\mathcal{MA}$ sheet points, by stretching smooth elastic surfaces over the links, much as is done when a "tent" is constructed over a scaffold.

* Remarks on the re-organization of the $\mathcal{MS}$ and $\mathcal{SC}$ hierarchies.

A major reason to re-organize the $\mathcal{SC}$ and $\mathcal{MS}$ into two parallel hierarchies, in comparing the Leymarie and Kimia's original single-track hierarchy [125], is two fold: *(i)* First, a formal characterization of the $\mathcal{SC}^{\mathcal{H}}$ is not yet accomplished, which is an ongoing work requiring a full shock flow analysis on the medial sheets (detailed in § 3.4). *(ii)* Second, the important notion of $\mathcal{MA}$ transitions and transforms (refer to Chapter 1) should be first studied on the $\mathcal{MS}$, allowing a better focus on the topological changes of the structure. The study of the $\mathcal{SC}$ transitions and transforms can be extended smoothly later on. Thus we only focus on the $\mathcal{MS}$ transforms (Chapter 5.4.1) in this thesis.

3.3 $\mathcal{MS}$ and $\mathcal{SC}$ for a Point Set

In this section, we consider a restricted case where the surface dataset is represented by clouds of *unorganized* points, where no *a priori* neighborhood relationship is known between the points. We call an input sample point a *"generator"* in analogy to sources when simulating wavefront propagation from points to create the $\mathcal{MA}$ [201, 128]. In this context, the sample point can be viewed as an infinitesimal spheres which can be viewed as a degenerate kind of surface.

* Review: 3 types of shock elements (A_1^2 sheets, A_1^3 curves, A_1^4 nodes) arising from a point set.

(Revised from Leymarie [125].) For this restricted class of surfaces, *i.e.*, infinitesimal spheres, certain types of shocks do not occur. There are no A_3 curves (ribs) since there are no ridges on spheres, nor any A_1A_3 vertices (rib ends) by extension. Also, the geometry implies that there are no shock sinks and relays in the interior of sheets, nor shock sinks in the interior of curves, so no A_1^2-4, A_1^2-3, and A_1^3-4 occur. Hence, when considering point generators, out of the **18** shock points in Figure 3.1, we are left with only **eight** possible ones, Table 3.2, *six* of which are singularities for the flow, *i.e.*, shock nodes. This simplified case is closely related to the Voronoi diagram ($\mathcal{VD}$) [145] of a point set, *i.e.*, the remain medial components in the $\mathcal{MS}$ hypergraph are of three types, namely, *(i)* the A_1^2 sheet which corresponds to a Voronoi face (dual of a Delaunay edge), *(ii)* the A_1^3 curve which corresponds to a Voronoi edge (dual of a Delaunay triangle), and *(iii)* the A_1^4 node which corresponds to a Voronoi vertex (dual of a Delaunay tetrahedron), Figure 3.5.

3.3.1 Classification of the $\mathcal{MS}$ sheets, curves, and vertices based on a radius flow analysis

* Overview: classification of the $\mathcal{MS}$ sheets/curves/nodes (of a point set) based on their shock flow.

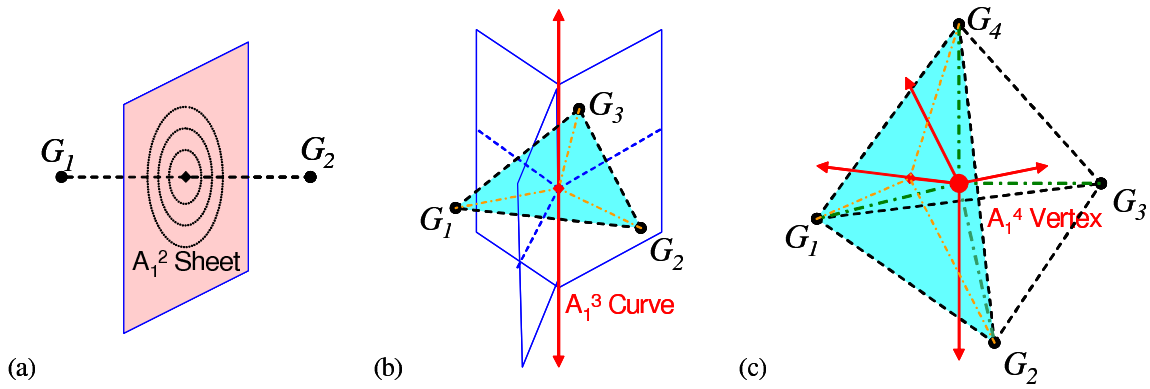


Figure 3.5: (Adapted from [125, Fig.7].) For the input of only points, the $\mathcal{MS}$ consists of only three types: (a) A_1^2 sheets, (b) A_1^3 curves, and (c) A_1^4 vertices.

Shocks	Regular	Source	Relay	Sink
Sheet	A_1^2-1	A_1^2-2	-	-
Axial	A_1^3-1	A_1^3-2	A_1^3-3	-
Axial end	-	-	A_1^4-3a A_1^4-3b	A_1^4-4

2 types of *regular* shock points:
- Inside an A_1^2-1 sheet
- Inside an A_1^3-1 axial curve

6 types of *singular* shock points:
- **2** types of shock sources (A-2):
  - inside a sheet: A_1^2-2
  - inside an axial: A_1^3-2
- **3** types of shock relays (A-3):
  - inside an axial: A_1^3-3
  - at the end of an axial: A_1^4-3a, A_1^4-3b
- **1** type of shock sinks (A-4):
  - at the end of an axial: A_1^4-4

Table 3.2: (Adapted from [125, Table.4].) Summary of Leymarie and Kimia's classification of 8 types of shock points in a restrictive case of unorganized point inputs.

hen the input set is restricted to a point set, the flow information of a $\mathcal{MS}$ can be completely reconstructed to obtain a $\mathcal{SC}$ out of it. This section achieves it by classifying the shock flow types of the $\mathcal{MS}$ hypergraph elements, namely, the A_1^2 sheets, A_1^3 curves, and A_1^4 nodes based on their local configuration. An algorithm to implement each classification is also presented. The classification is aimed to be general to cover all degenerate cases, while considering the numerical issued frequently occur in practical computation.

* Three types of A_1^2 shock sheets (planar polygons).

The A_1^2 shock sheets (planar polygons) arising from two generators ($\mathsf{G_i}$,$\mathsf{G_j}$) can be classified into *three* types, including the degenerate cases, based on the local shock flow configuration, Figure 3.6.

I. The sheet polygon S contains the mid point M of the two generators (G_i, G_j) in the interior. In this case, M is the A_1^2-2 source, which is also the smallest radius point of S. The degenerates when $\mathsf{M}(\mathsf{G_i}, \mathsf{G_j})$ is on the polygon boundary edge or boundary vertex is not of this type; they are classified into type II and III , respectively (detailed below).

II. The point R of smallest radius of the sheet S is on S's boundary edge. The case of $\mathsf{M}(\mathsf{G_i}, \mathsf{G_j}) = \mathsf{R}$ belongs to this category, since R can be viewed as an A_1^3-2 source. Another degeneracy when R is on S's boundary vertex is classified into type III (see below).

III. The point R of smallest radius of the sheet S is at S's boundary vertex. The case of $\mathsf{M}(\mathsf{G_i}, \mathsf{G_j}) = \mathsf{R}$: belongs to this category, since R can be viewed as an A_1^4-3b relay.

* Algorithm to determine the (flow) type of an A_1^2 sheet (arising from points).

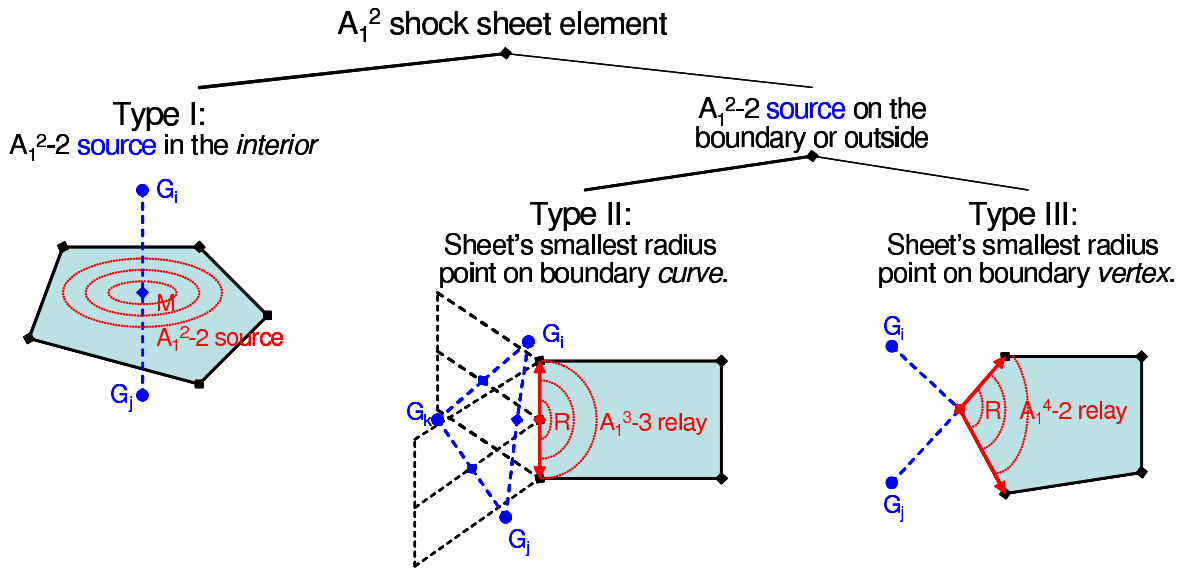


Figure 3.6: Three types of A_1^2 shock sheet elements (planar polygons) arising from two generators $(\mathsf{G_i}, \mathsf{G_j})$ via shock flow analysis: If sheet polygon contains the A_1^2-2 source in the interior, it is Type I. Otherwise, if the sheet's smallest radius point is at the boundary curve (not including end points), it is Type II; else the smallest radius point must be at the boundary vertex, Type III.

The algorithm to determine the (flow) type of an A_1^2 sheet S (arising from points) is described below in two steps, Figure 3.6: *(i)* The first step is to check if S contains the A_1^2-2 source point in the interior of S or not. Since the geometry of an A_1^2 shock sheet in this case is a (convex) planar polygon, this can be done by first project the mid point $\mathsf{M}(\mathsf{G_i}, \mathsf{G_j})$ to the plane of S and check if M is included in the interior of the sheet polygon or not to determine S's type. (Refer to *e.g.*, [146] for a robust "point inclusion in polygon" test). (ii) The second step is to check the point R of smallest radius of S is on S's boundary edge or vertex. This can be dome by traversing through all S's boundary edges and find the smallest radius point R to determine S's type.

* Three types of A_1^3 (or degenerate A_1^n) shock curves (line segments).

The A_1^3 shock curves (line segments) arising from three generators $(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k})$ can be classified into *three* types, based on the local shock flow configuration, Figure 3.7. This classification also applies to the degenerate cases of A_1^n $(n > 3)$ shock curves arising from n generators.

I. Both the shock curve L and its dual Delaunay triangle T contain the circumcenter $\mathsf{C}(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k})$ in the interior. In this case, the circumcenter C has the following properties: *(i)* C is the largest radius point on T; *(ii)* C is the smallest radius point on the shock curve L; *(iii)* C is the A_1^3-2 source of L; and *(iv)* T is acute. The degenerate case that C is on the triangle boundary happens when T is a right triangle, which can be viewed as a transition between this case (type I) and the following case (type II). We classify the above degenerate case as type I.

II. The shock curve L contains the circumcenter C, but C is not contained in the interior of L's dual Delaunay triangle T. In this case, the circumcenter C has the following properties: *(i)* C is the smallest radius point on the shock curve L; *(ii)* C is the A_1^3-3 source of L; and *(iii)* T is obtuse.

III. The shock curve L does not contain the A_1^3-3 circumcenter C at all. This case occurs when there exists a close-by generator $\mathsf{G_l}$ which invalidates C, and the shock curve L is created from either a A_1^4-3a or A_1^4-3b shock node.

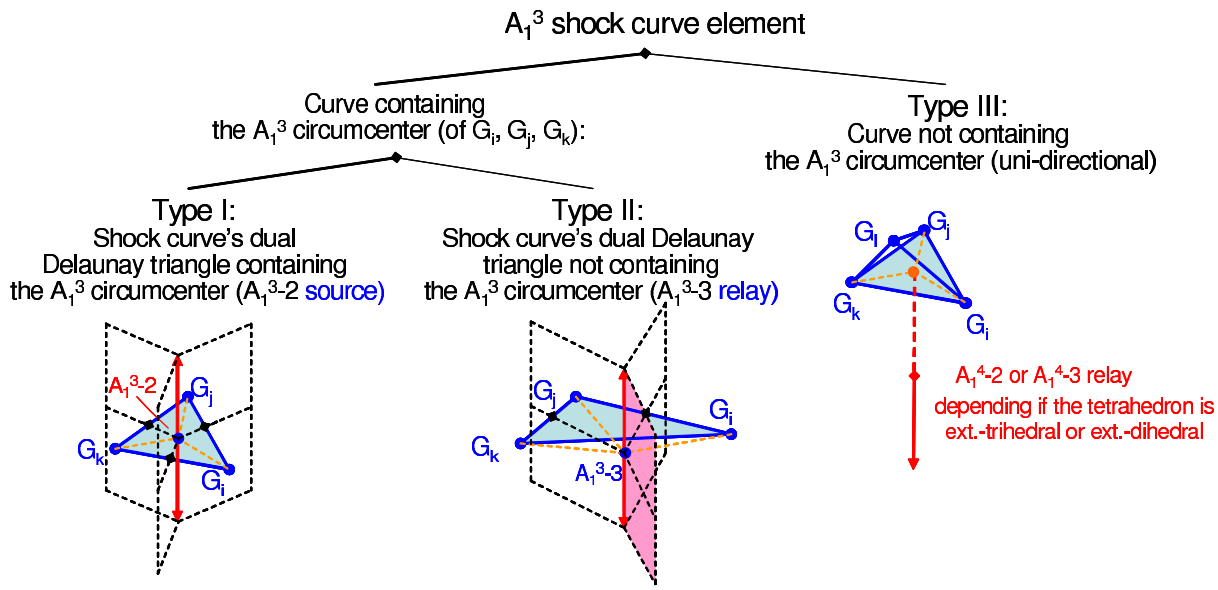


Figure 3.7: Three types of A_1^3 shock curve elements (line segments) arising from three generators $(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k})$ via shock flow analysis: If shock curve does not contain the A_1^3 circumcenter, it is Type III and the shock curve must be uni-directional in flow. Otherwise, if the shock curve's dual Delaunay triangle contains the A_1^3 circumcenter (which is an A_1^3-2 *source*), it is type I; else it is type II (and the A_1^3 circumcenter is an A_1^3-3 *relay*).

* Algorithm to determine the (flow) type of an A_1^3 (or degenerate A_1^n) curve (arising from points).

The algorithm to determine the (flow) type of an A_1^3 (or *degenerate* A_1^n) shock curve is described below in two steps, Figure 3.7:

(1). Check if the A_1^3 circumcenter C is on the shock curve L or not, which is equivalent to check if shock curve is uni-directional or bi-directional. This can be done by computing the position of the circumcenter C and compare the distance of $(\overline{\mathsf{CS}}, \overline{\mathsf{SE}})$ and $(\overline{\mathsf{CE}}, \overline{\mathsf{SE}})$, where S and E are the two end nodes of L.

(2). Check if the shock curve's dual Delaunay triangle T containing the A_1^3 circumcenter C or not (to distinguish type I or II). For a regular A_1^3 shock curve, this can be done by testing if triangle $\mathsf{G_i G_j G_k}$ is acute or not. In the degenerate case of an A_1^n shock curve $(n > 3)$, this can be done by testing if shock curve intersects with the Delaunay polygon or not. Again we use the property that the Delaunay polygon is guaranteed to be *convex* and use the above "point inclusion in polygon" test to determine the flow type.

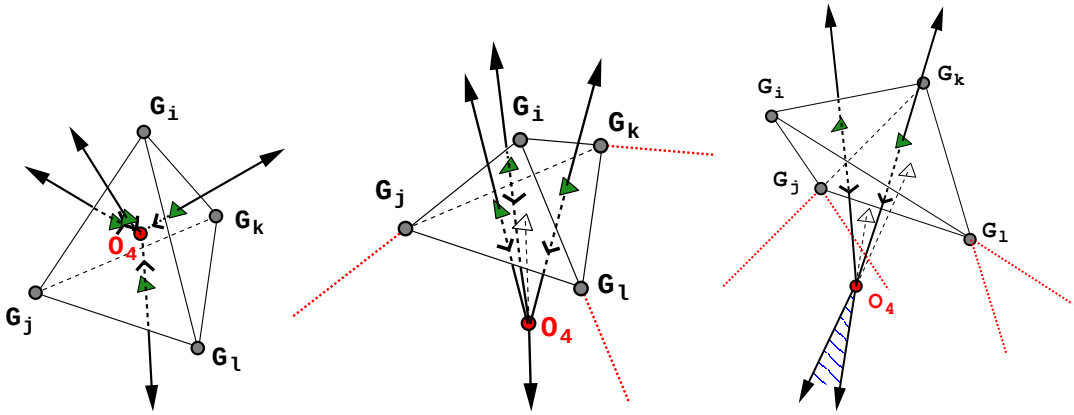
* Three types of A_1^4 shock nodes (singular points).

(Adapted from Leymarie [125] for completeness.) The A_1^4 shock nodes (singular points) arising from four generators $(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k}, \mathsf{G_l})$ can be classified into *four* types based on the shock flow analysis, Figure 3.8. The classification of the degenerate cases of A_1^n $(n{>}4)$ shock vertices arising from n generators is considered similarly.

I. A_1^4-4 sink: with four A_1^3 shock curves flowing in.

II. A_1^4-3a relay: with three shock curves flowing in and one shock curve flowing out.

III. A_1^4-3b relay: with two shock curves flowing in and two shock curves flowing out.



(a) A_1^4-4 (Interior) (b) A_1^4-3a Ext.-Trihedral (c) A_1^3-3b Ext.-Dihedral

Figure 3.8: (From [125, Fig.9].) Three types of A_1^4 shock node elements via the flow analysis. Possible configurations of quadruplets of generators creating (a) interior, (b) exterior–trihedral, or (c) exterior–dihedral tetrahedra. Generators are shown as grey dots, valid A_1^3-2 sources of shock curves for triplets of generators are shown as small green triangles, and the A_1^4 shock vertex, sitting at the circumcenter, O_4, of the tetrahedral configuration, is shown as a small red disk. Directions of flow along curves are indicated by arrows.

* Algorithm to determine the (flow) type of an A_1^4 node (arising from points).

While Leymarie and Kimia [125] proposed to determine the A_1^4 type by using the barycentric coordinate, their method is not extensible to the degenerate A_1^n node, $n > 4$. We propose to directly count the number of "in-flow" incident shock curves to determine the type. Our approach is extensible to handles the degenerate $A_1^n, n > 4$ nodes and can be implemented by adopting the previous algorithm to determine whether an incident curve is uni-directional or bi-directional.

* Additional radius flow analysis on the A_1^4 shock nodes.

The above classification of the A_1^4 shock nodes can be further refined based on the local configuration of the tetrahedron of the four generators $(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k}, \mathsf{G_l})$. A closely related work is Siersma *et al.*'s classification of the generic tetrahedra [184] into **nine** types based on a Morse analysis, summarized in Figure 3.9, which provides a refined classification of the A_1^4 shock nodes, if they are considered in *isolation*, namely, no shock flow comes from generators other than $(\mathsf{G_i}, \mathsf{G_j}, \mathsf{G_k}, \mathsf{G_l})$. However, Siersma's classification of tetrahedra is not directly applicable in our case, since in general the shock flow of more than 4 generators does come from outside the tetrahedron in many cases; refer to the Flow Complex papers [92, 64] reviewed in Chapter 2. We leave the further analysis and the extension to classify the degenerate $A_1^n(n > 4)$ shock nodes as future works.

3.3.2 Construction of the $\mathcal{MS}$ from a Voronoi-based computation

While the previous section describes an approach to construct a $\mathcal{SC}$ from the $\mathcal{MS}$ for a point set, by recovering the missing shock flow information, this section further describes a continuing approach to construct the $\mathcal{MS}$ from a Voronoi Diagram ($\mathcal{VD}$) of the point set. This provides an alternative approach to compute the $\mathcal{MS}$ / $\mathcal{SC}$ from a Voronoi or Delaunay Triangulation ($\mathcal{DT}$) based computations (which have been extensively studied in the computational geometry [145, 162, 15]), in supplement to Leymarie's original flow-based computation [125]. This also bridges the gaps between two different classes of approaches, namely *(i)* the *geometry*-based computation of the $\mathcal{VD}$ /

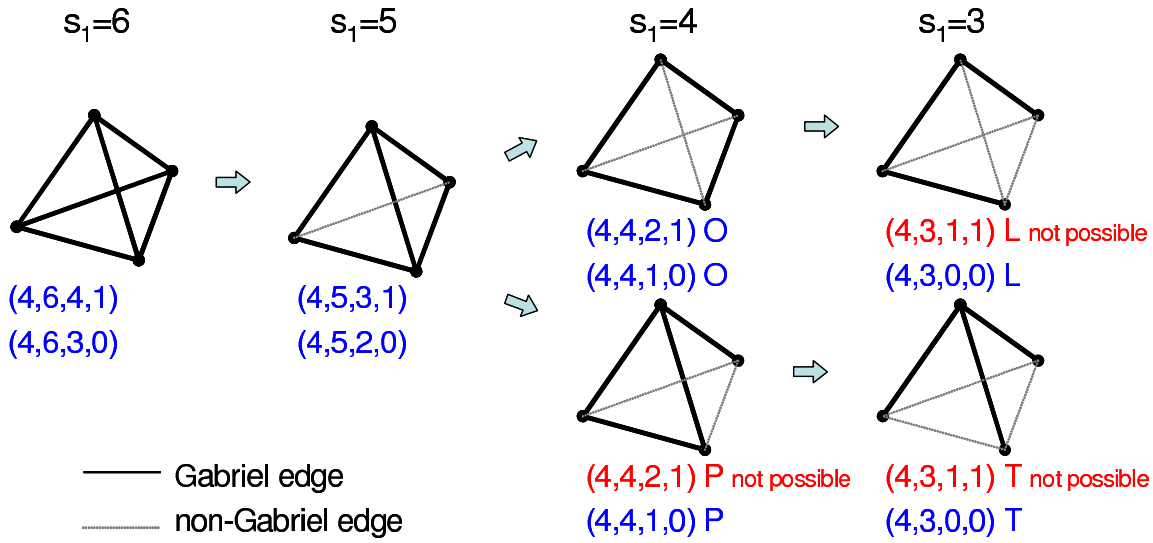


Figure 3.9: (Adapted from Siersma *et al.*[184, Fig.1].) The classification of the tetrahedra of isolated A_1^4 shock nodes into *nine* generic types based on counting the *Gabriel edges* of the tetrahedra. The notation (m, s_1, s_2, M) is defined in [184] as follows: m refers to the number of the minimum of radius (*i.e.*, the generators), s_1 refers to the number of index-1 saddle (*i.e.*, the A_1^2-2 sources), s_2 refers to index-2 saddle (*i.e.*, A_1^3-2 sources), and M refers to the number of the maximum of radius (*i.e.*, the A_1^4-4 sinks). Siersma *et al.* show that up to the combinatorial equivalence of the Morse posets, there are *nine* generic tetrahedra, which are uniquely described by the nine configurations: $(4,6,4,1)$, $(4,6,3,0)$, $(4,5,3,1)$, $(4,5,2,0)$, $(4,4,2,1)$ O, $(4,4,1,0)$ O, $(4,4,1,0)$ P, $(4,3,0,0)$ L, and $(4,3,0,0)$ T. The other three configurations of $(4,4,2,1)$ P, $(4,3,1,1)$ L, and $(4,3,1,1)$ T do not occur.

$\mathcal{DT}$ by intersecting and trimming bisectors [153, Ch.11], and *(ii)* the *propagation*-based computation of the $\mathcal{MS}$ / $\mathcal{SC}$ using thinning, distance transform, and surface evolution based on the nature of Blum's grassfire [31, 30, 202].

We first summarize Leymarie's flow-based computation of the $\mathcal{SC}$ (and essentially the $\mathcal{MS}$ and $\mathcal{VD}$) as follows. We then describe an approach to recover the $\mathcal{MS}$ from the $\mathcal{VD}$.

* Summary of Leymarie's flow-based computation of the $\mathcal{SC}$ of a point set [125].

Leymarie and Kimia present a Lagrangian computation of the $\mathcal{SC}$ from an unorganized point set using a propagation-based framework [125] [128, §4.5]. This method comprises two main steps: *(i)* an **initialization** step, where all A_1^2-2 shock sources are computed using an efficient bucketing scheme, followed by *(ii)* a **propagation** step, where the pairing of A_1^2 sheets are used to find all A_1^3 curves, and the A_1^3 curves are again paired to find all A_1^4 vertices. Specifically, the A_1^3 curves are computed from a sequence of critical points of the radius flow (the radius of maximal balls projected on the $\mathcal{MA}$): sources for $\mathcal{MA}$ sheets (A_1^2-2) are paired to find sources for A_1^2 curves (A_1^3-2), which in turn are paired to find A_1^4 endpoints; details in [125].

Leymarie's $\mathcal{SC}$ computation has three main advantages in comparing to other approaches mainly developed in computational geometry. First, it computes the $\mathcal{SC}$ directly, a richer structure other than a $\mathcal{VD}$. Second, it supports adding/deleting of generators and allows to "update" the $\mathcal{SC}$ structure locally and dynamically. Third, it can be generalized to handle polygonal generators. However, its *capacity* (of number of generators to process) and the computational *speed* is not optimal, due to the additional overheads (such as the shock flow) other than the geometry itself. Below we describe

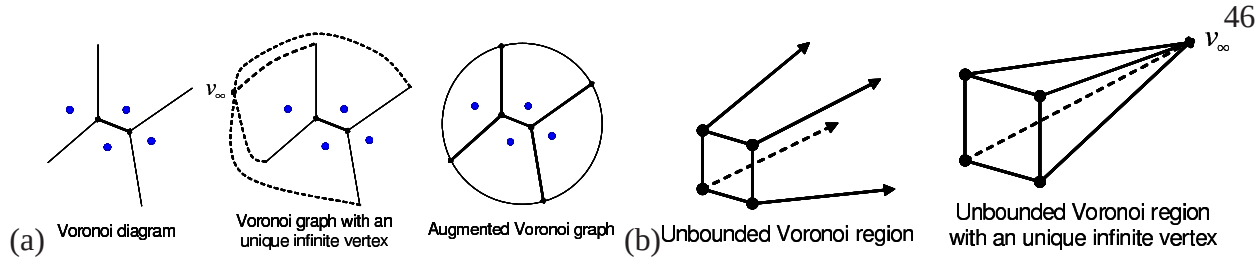


Figure 3.10: (Adapted from Okabe [145, Fig.4.2.1].) (a) A 2D Voronoi diagram and its graph structure with an unique *"infinity" vertex* v_∞ [145, p.236]. (b) The analogy in 3D,where the unbounded Voronoi region is represented similarly using an unique "infinity" vertex v_∞.

an approach to recover the $\mathcal{MS}$ from a pre-computed $\mathcal{VD}$. Specifically, we use the QHull [15] (http://www.qhull.org/), a popular package which computes the Voronoi/Delaunay complex in a higher dimensional space (detailed in [15]). Our recovery step is efficient and can be done in $O(n)$, where n is the number of input generators.

* Recover the $\mathcal{MS}$ from a $\mathcal{VD}$ computed from QHull.

While the $\mathcal{MS}$ and the $\mathcal{VD}$ of a point set are highly related in structure (a $\mathcal{MS}$ sheet corresponds to a $\mathcal{VD}$ face, a $\mathcal{MS}$ curve corresponds to a $\mathcal{VD}$ edge, and a $\mathcal{MS}$ node corresponds to a $\mathcal{VD}$ vertex), thus enables a straight-forward conversion, the recovery of the $\mathcal{MS}$ from $\mathcal{VD}$ is not trivial in degenerate cases. We list two main difficulties as follows.

- It is well-known that in 2D the $\mathcal{VD}$ can be organized into a graph form, namely the *Voronoi Graph* ($\mathcal{VG}$) [145, p.236], Figure 3.10(a); while in 3D the analogy of a hypergraph form (*i.e.* the "Voronoi hypergraph") is not extensively studied.
- In 2D, the unbounded Voronoi region can be wisely represented by introducing an *infinity* vertex v_∞ to represent the outmost Voronoi region [145, p.236] and thus make consistent the dual of the $\mathcal{VD}$ and the $\mathcal{DT}$ (in the number of vertices and edges). We note that this "infinity issue" results in a major difference between the $\mathcal{VD}$ and our notion of the $\mathcal{MS}$ in 3D. We do not consider the unbounded $\mathcal{MS}$ curves to intersect at an infinity vertex, since their direction is different as they form in the propagation framework, Figure 3.10(b). Furthermore, it make the recovery of the $\mathcal{MS}$ from $\mathcal{VD}$ difficult for degenerate cases where numerical precision breaks down. Specifically, it is ambiguous to reconstruct all unbounded A_1^3 curves from a single infinity vertex in degenerate cases, since the numerical decision (perturbation) performed in the $\mathcal{VD}$ computation is not known. [8]

* Solution to the numerical degeneracy of the $\mathcal{VD}$: introduce a bounding sphere.

Our solution to recover the full $\mathcal{MS}$ hypergraph from the QHull result is to introduce a bounding sphere $\mathcal{B}$ large enough to enclose the input shape and then "patch" the resulting $\mathcal{VD}$ to recover the $\mathcal{MS}$ for its bounded and unbounded components. Figure 3.11 illustrates an example in 2D and in 3D. Specifically, $\mathcal{B}$ should be large enough to enclose all original finite $\mathcal{VD}$ vertices, such that the recovered $\mathcal{MS}$ remains complete in its exterior (unbounded) component. [9]

[8] Since our approach is to recover the $\mathcal{MS}$ from a pre-computed $\mathcal{VD}$ via QHull, we assume the numerical consistency is always valid and should not rely on interpreting how the numerical decision is made in the QHull computation.

[9] A naive way to determine a safe size of the bounding sphere $\mathcal{B}$ is to run QHull of the input points and set $\mathcal{B}$ larger enough such that all Voronoi vertices remain.

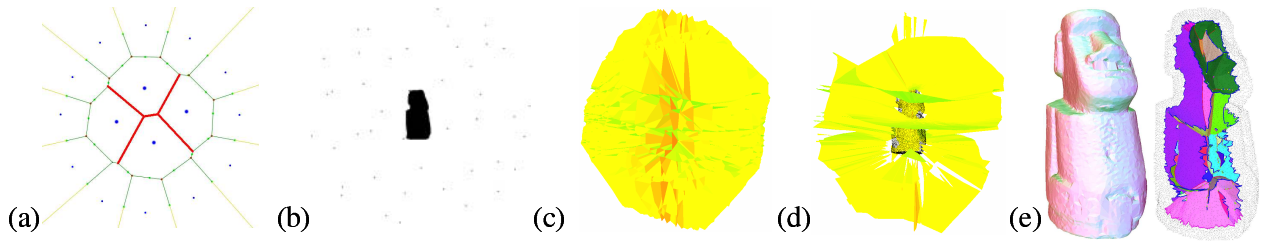


Figure 3.11: This figure illustrates the introducing of a bounding sphere $\mathcal{B}$ to recover the $\mathcal{MS}$ from the $\mathcal{VD}$. (a) The 2D analogy of adding a bounding circle $\mathcal{B}$ of the input of 4 points (blue) to recover the $\mathcal{MA}$ (red). The green and yellow shock branches arisen from $\mathcal{B}$ and thus should be removed. In 3D, (b) shows the adding of 28 points on $\mathcal{B}$ to the input shape (the Moai of 10k points). (c) is the full $\mathcal{MS}$ recovered from the $\mathcal{VD}$ which corresponds to the red and green shock branches in (a), with the unbounded (yellow) branches in (a) not shown. Observe that the interior $\mathcal{MA}$ of the shape is completely invisible in (c). We show the result of the interior $\mathcal{MA}$ in (d,e) which is computed using the approach in Chapters 6 and 7. In (d) the interior and exterior $\mathcal{MS}$ is separated using a segregation algorithm (detailed in § 6.2), where the interior $\mathcal{MS}$ is barely visible. (e) shows the re-meshed shape surface and the regularized interior $\mathcal{MS}$. See also Figure 7.5 for more details of how the $\mathcal{MS}$ of the input shape is extracted and regularized in our approach.

The recovery of the $\mathcal{MS}$ from the $\mathcal{VD}$ comprises two steps. We first label all shocks arisen from $\mathcal{B}$ as "at infinity", *i.e.* to be *infinity nodes, curves, and sheets*, in mimicking the infinity vertex v_∞ in the 2D case. We then label all medial elements incident to the infinity nodes/curves/sheets to be *unbounded*. The rest medial elements are then the *bounded* ones, which remains unaltered as if the case without the introducing of the bounding sphere $\mathcal{B}$.

We observe in practice a very sparse sampling of the bounding sphere $\mathcal{B}$ is sufficient thus does not affect the overall computation time. In our implementation, we sample $\mathcal{B}$ by parameterizing it in spherical coordinates. Typically the introducing of 20 to 30 sample points is sufficient to extract a visually plausible $\mathcal{MS}$.

3.4 Future Work: a Complete Flow Analysis of the $\mathcal{MS}$ toward the $\mathcal{SC}$

* Future work: further classification of the shock flow inside $\mathcal{MS}$ sheets toward a coarse-scale $\mathcal{SC}$.

While the local form of the $\mathcal{SC}$ is studied and classified into 18 types of shocks, a global analysis on the shock flow in the interior of the medial sheet is yet to be explored. Figure 3.12 shows an example of such flow analysis on the interior of shock sheets, where a more accurate simulation of the shock flows need to be investigated. Note that we have not explicitly defined any shock curve to connect an A_1^2-2 source to other shock nodes in the $\mathcal{SC}$ hypergraph. We expect in the future to describe such shock curves to further divide a medial sheet into *districts* of monotonic flows. One of our ongoing works is to study the topological *surface network* of the shock radius r as a height function on the $\mathcal{MA}$ sheets [142]. This is related to the ridge extraction of a height function (a 2D scalar function). [10] We expect this coarse-scale shock scaffold to be useful in robust matching and compression of shapes. Figure 3.12

[10] Note that this *height ridge* is different than the *shape ridge* we propose to detect in § 10.2.

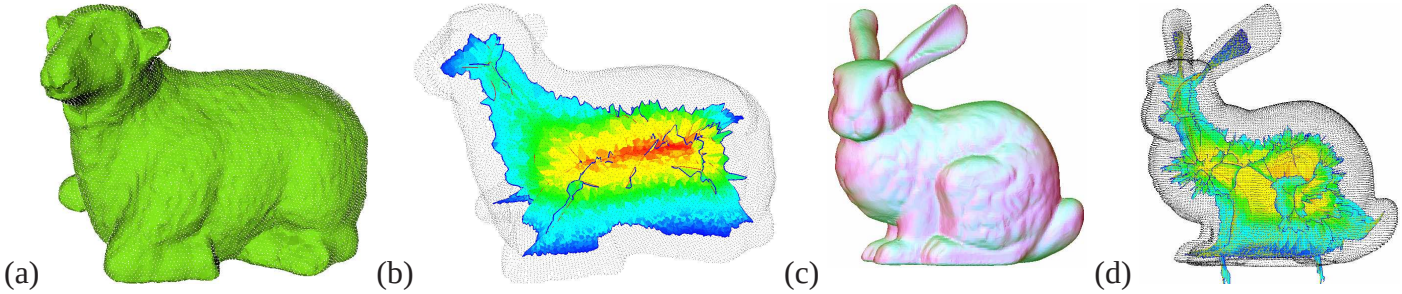


Figure 3.12: This figure illustrates the shock radius field on the shock sheets by coloring each A_1^2 sheet element from red (largest radius) to blue (smallest radius on the A_3 rib curves) for two shapes: (a-b) A sheep toy object ($22,619$ points). (c-d) The Stanford bunny ($35,947$ points).

Chapter 4

Separating the Topology and Fine-Scale Geometry of the Medial Scaffold

* Overview of chapter: representing the $\mathcal{MS}$ topology, geometry, and other attributes.

This chapter addresses the *representation* of the medial scaffold ($\mathcal{MS}$) with respect to its *topology*, *geometry*, *dynamics* (of the shock flow), and other attributes. A key idea is to separate the topology (which is pertinent to the qualitative structure) from its fine-scale geometry and other attributes and organize the $\mathcal{MS}$ as a *dual-scale* structure, Figure 4.1: *(i)* The *coarse-scale* structure is a (topological) **hypergraph** describing the global structural inter-connectivity between the medial sheets, curves, and nodes, which essentially implements the topological medial scaffold ($\mathcal{MS}^{\mathcal{T}}$) in § 3.2. *(ii)* The *fine-scale* structure is a **polygonal mesh** describing the local metric attributes such as the geometry and dynamics of the $\mathcal{MS}$. In the dual-scale $\mathcal{MS}$ representation, the coarse-scale sheets/curves explicitly comprise its fine-scale mesh face/edge elements (detailed in § 4.1). We develop a **novel *Extended Half-Edge* ($\mathcal{EHE}$)** data structure by adopting the popular *half-edge* [114, 113, 69] to represent both the coarse-scale hypergraph and the fine-scale mesh. We emphasize its capability to handle *non-manifold* junctions, which is essential in describing the ubiquitous intersections of the medial sheets at the medial curves.

* Main difficulty in representing the $\mathcal{MS}$: represent the medial sheet topology.

The main difficulty in representing the qualitative structure of the $\mathcal{MS}$ lies on representing its topology, *i.e.*, the **global inter-connectivity** of a medial sheet with respect to other sheets. [1] More precisely, the goal is to understand all topological configurations a medial sheet could possibly contain. We classify *three* such cases and represent the topological incidence between the medial sheets and curves using several *chains* of half-edges in the $\mathcal{EHE}$ representation, detailed in § 4.4. A recent related work is James Damon's study [58] on the "irreducible" medial sheet component that is topologically equivalent to a 2D disk.

Upon our solution described above, the second difficulty is on the topological *degeneracies* of a medial sheet, which cause more than one valid forms of a sheet in the $\mathcal{EHE}$ representation. Thus a *canonicalization* of the $\mathcal{EHE}$ representation is required. We analysis these cases and define a **canonical form** of the medial sheet topology, which will be useful in editing the $\mathcal{MS}$ topology in applying the $\mathcal{MS}$ transforms in Chapter 5.

* Organization of chapter.

[1] Possibly a medial sheet could intersect itself, see below.

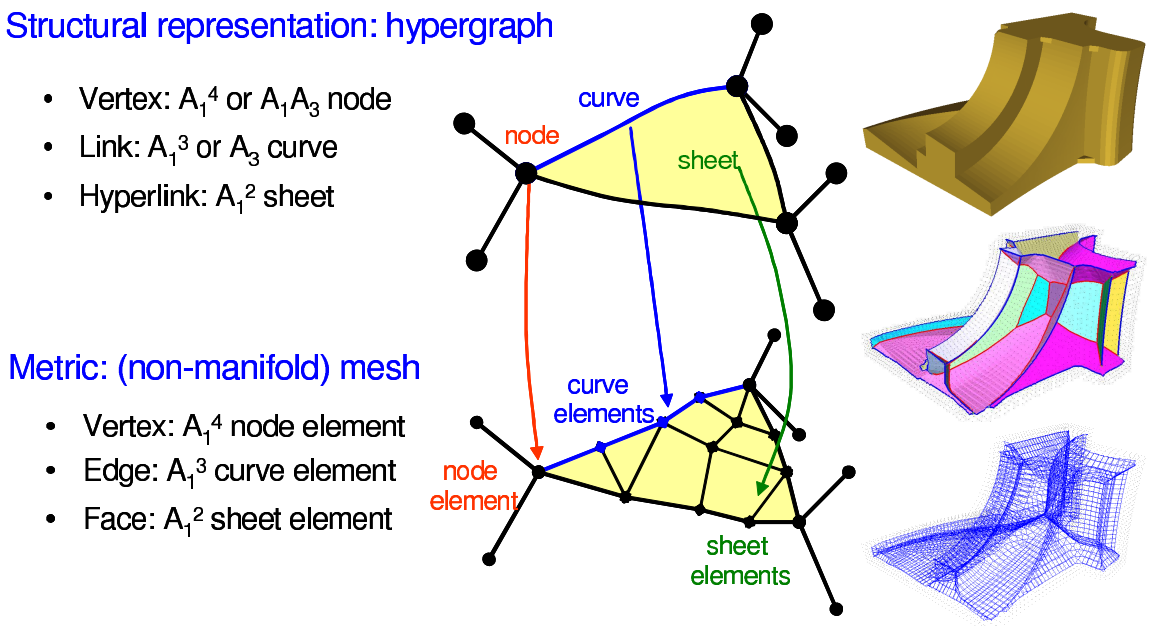


Figure 4.1: The *dual-scale* $\mathcal{MS}$ structure [43]: The coarse-scale is a *hypergraph* capturing the global *topology* and the fine-scale is a (typically non-manifold) *mesh* capturing the metric information such as the *geometry* and other attributes. The figures on the right depicts an industrial part (fandisk) and its $\mathcal{MS}$ in both scales.

This chapter is organized as follows. Section 4.1 discusses how the $\mathcal{MS}$ representation can be separated into two parts: a coarse-scale structure representing its global topology and a fine-scale structure capturing its detailed metric properties. The remaining of this chapter describes how the coarse-scale $\mathcal{MS}$ topology can be represented by extending the half-edge ($\mathcal{HE}$) data structure (originally designed to represent *2-manifold* meshes) to efficiently handle *non-manifold* meshes. Specifically, Section 4.2 reviews the $\mathcal{HE}$ data structure to represent a polyhedral surface. Section 4.3 extends the $\mathcal{HE}$ data structure to represent a *non-manifold* polygonal mesh. Section 4.4 further extends the above approach to represent the topology of a *geometric hypergraph*, which can be viewed as a generalization of the mesh to a hypergraph consisting of sheets and curves by deforming the mesh faces and edges.

4.1 The Dual-Scale Representation of the $\mathcal{MS}$

* Motivation: separating the $\mathcal{MS}$ topology from its fine-scale attributes.

The following advantages motivate the separation of the $\mathcal{MS}$ topology from its fine-scale metrics:

- *Enable an effective representation of the $\mathcal{MS}$ attributes.* Among the important attributes of the $\mathcal{MS}$ such as the topology, geometry, dynamics, *etc.*, the topology is a global attribute describing the structure, while the others are local properties. [2] We explicit construct a coarse-scale structure to handle the global topology of the $\mathcal{MS}$, namely the coarse-scale *hypergraph* composing of medial sheets/curves/nodes as hyperlinks/links/vertices, respectively. All other attributes is stored in a fine-scale structure, namely the fine-scale polygonal *mesh* composing

[2] The division of the shock flow dynamics interior to each shock sheet into *districts* of monotonic flows results in a *surface network*, which is another global attribute. This belongs to the future work of the shock scaffold ($\mathcal{SC}$) in § 3.4.

of face/edge/vertex elements. The coarse-scale components explicitly compose of their fine-scale elements (*i.e.*, contain pointers pointing to the mesh elements), as shown in Figure 4.1.

- *Enable an explicit implementation of the $\mathcal{MS}$ hierarchy.* The hierarchy of $\mathcal{MS}$ representations from the most complete $\mathcal{MS}^{\mathcal{H}}$ to the reduced ones of $\mathcal{MS}^{\mathcal{H}-}$, $\mathcal{MS}^{\mathcal{G}}$, $\mathcal{MS}^{\mathcal{G}-}$, and finally the $\mathcal{MS}^{\mathcal{T}}$ in Chapter 3 can be explicitly implemented via the dual-scale structure by dropping pertinent information.
- *Represent the qualitative $\mathcal{MA}$.* The explicit separation of two-level of structures enables representation of the qualitative $\mathcal{MA}$ where both the simplified global topology and detailed metrics are available. This is related to an idea in scientific computation (numerical analysis) that instead of aiming to compute an exact $\mathcal{MA}$ of the input shape, which is virtually no possible. We aim to approximate the exact $\mathcal{MA}$ of a *close-by* (perturbed) shape, whose $\mathcal{MA}$ is succinct and represents the qualitative structure. This can be done by apply a set of transforms operating on the two-level structure of the $\mathcal{MS}$, which will be elaborated in Chapter 5.
- *Enable an explicit simulation of $\mathcal{MS}$ transforms.* The explicit separation of a global structure to handle topology also enables to explicitly "edit" the hypergraph in applying the $\mathcal{MS}$ transforms. Note that the exact re-computation of the $\mathcal{MS}$ is costly. Instead, we *simulate* a $\mathcal{MS}$ transform in the coarse-scale hypergraph, while keeping the fine-scale $\mathcal{MS}$ elements intact (Chapter 5). In addition, the fine-scale geometry of the $\mathcal{MS}$ and its associated boundary generators are useful in estimating the transform costs.

* Coarse-scale medial sheets/curves/nodes to represent the global structure of the $\mathcal{MS}$.

We explicitly construct the coarse-scale medial sheets/curves/nodes of the hypergraph. They not only represent the global structure of the $\mathcal{MS}$ but also organize their corresponding fine-scale mesh elements. Specifically, the coarse-scale medial sheet composes of a set of fine-scale faces (which can be viewed as also a "meshing" of the faces), the coarse-scale medial curve composes a set of fine-scale mesh edges as a *poly-line*, and the coarse-scale medial node simply contain a fine-scale vertex. In our implementation a coarse-scale component only stores two items to reduce unnecessary redundancy: *(i)* its global (topological) connectivity via the $\mathcal{EHE}$ detailed below and *(ii)* its pointers to all fine-scale elements.

* Fine-scale polygonal mesh elements to handle the local metric attributes of the $\mathcal{MS}$.

We explicit construct a fine-scale polygonal mesh to organize and store local attributes as follows:

- *Geometry:* We store the position $p(x, y, z)$ of each fine-scale vertex v at it. Note that this is the only *external* information we store in the dual-scale structure. The geometry of the $\mathcal{MS}$ such as the length and area, are then available by linear approximation via the polygonal mesh.
- *Associated boundary points:* The boundary sample points (generators) of the $\mathcal{MS}$ are explicitly associated with the fine-scale A_1^2 mesh faces. The generators of an A_1^3 edge and A_1^4 vertex elements are available through the fine-scale mesh connectivity. The issue of maintaining a consistent boundary-$\mathcal{MS}$ association in the $\mathcal{MS}$ transforms will be handled in § 7.3.
- *Dynamics of shock radius and derivatives:* The **dynamics** of the shock *radius* r and its derivatives (*velocity* $v = \frac{dr}{ds}$, where s is the arc-length along the shock, and *acceleration* $a = \frac{dv}{ds}$) can be computed from the associated generators in the local configuration. We thus do not explicitly store the initial dynamics, but for applications where they are explicitly used such as to reconstruct the shape in § 8.2, we could store them in the A_1^2 face elements.

Table 4.1: A summary of the coarse-scale hypergraph components and the fine-scale mesh elements.

	Topological representation
fine-scale $\mathcal{MS}$ vertex v	incident $\{e_i \| i = 1, ..., n_{ve}\}$
fine-scale $\mathcal{MS}$ edge e	(v_s, v_e), $\{he_i \| i = 1, ..., n_{ef}\}$ circular list
fine-scale $\mathcal{MS}$ face f	$\{he_i \| i = 1, ..., n_{fe}\}$ circular list
coarse-scale $\mathcal{MS}$ node N	incident $\{C_i\} \| i = 1, ..., n_{NC}$
coarse-scale $\mathcal{MS}$ curve C	(N_s, N_e), $\{he_i \| i = 1, ..., n_{CS}\}$ circular list
coarse-scale $\mathcal{MS}$ sheet S	boundary $\{he_i \| i = 1, ..., n_{SC_b}\}$ circular list, interior $\{he_j \| j = 1, ..., n_{SC_i}\}$ circular lists

Table 4.1 summarizes the components comprising the dual-scale $\mathcal{MS}$ representation. Observe that the main topological relationship between the components is the **incidence relationship** between a 2D sheet (or face) to an 1D curve (or edge). Such topological incidence can be efficiently described by a "half-edge", a key feature of the $\mathcal{HE}$ data structure. Below we describe a general definition of the polygonal mesh and review the $\mathcal{HE}$ data structure in details.

4.2 A Review of the Half-Edge ($\mathcal{HE}$) Data Structure to Represent Polyhedral Meshes

We first define a general polygonal mesh as follows. Note that our definition is generalized from the popular view of a (surface-like) *2-manifold* mesh commonly used in computer graphics [82]. Specifically, we consider the *non-manifold* cases and the *degeneracies* (*e.g.* of a face degenerates into an edge, *etc.*), which is useful in modeling the 3D $\mathcal{MA}$ and other applications.

Definition 2 *A* **general polygonal mesh** M *is a collection of* vertices $V = \{v_i | v_i = (x_i, y_i, z_i), i = 1, ..., n_v\}$, edges $E = \{e_j | j = 1, ..., n_e\}$, *and* faces $F = \{f_k | k = 1, ..., n_f\}$, *such that each vertex is a point in* $\mathbb{R}^3$, *each edge is a line segment with two ending vertices in* V, *each face is a polygon with* m *edges in* E, *i.e.:*

$$\forall e \in E, \exists v_1, v_2 \in V, e = \{v_1, v_2\}, \tag{4.1}$$

$$\forall f \in F, \exists e_1, e_2, ..., e_m \in E, f = \{e_1, e_2, ..., e_m\}, \tag{4.2}$$

and no two edges intersect except at the ending vertices, no two faces intersect except at the boundary edges.

A **complete** *mesh is a mesh with no isolated edges nor vertices, i.e.,*

$$\forall e \in E, \exists f \in F, e \cap f \neq \emptyset, \;\; \forall v \in V, \exists e \in E, v \cap e \neq \emptyset, \tag{4.3}$$

A **2-manifold** *mesh is a* complete *mesh where each edge is shared by at most two faces, where* (i) *the set of all edges with only one incident face is the* boundary *of the manifold, and* (ii) *the neighborhood of any interior point on the mesh (except the boundary) is homeomorphic to a small disk in* $\mathbb{R}^2$. *A* **triangular** *mesh is a* complete *mesh with triangles as faces. A* **regular** *2-manifold mesh does not contain any* degenerate *vertex, which if removed, disconnects the mesh (i.e., a "vertex-only" connectivity).*

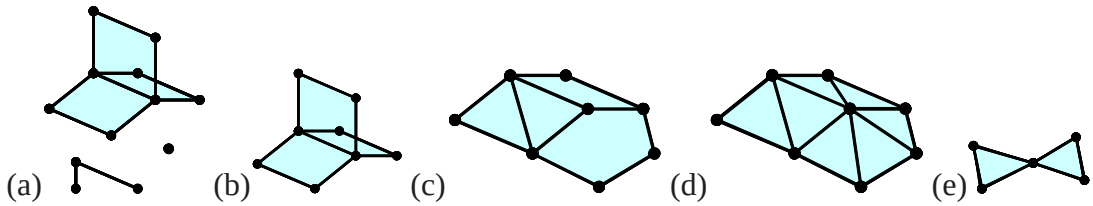


Figure 4.2: This figure illustrates several example meshes defined in the text. (a) A *general polygonal mesh* containing several degeneracies of isolated edges and vertices. (b) A complete *non-manifold* polygonal mesh. (c) A complete *2-manifold* polygonal mesh. (d) A complete 2-manifold *triangular* mesh. (e) An *irregular* triangular mesh with a *"vertex-only"* connectivity.

Figure 4.2 illustrates several example meshes defined above. Again we note that the surface mesh commonly used in the computer graphics community is the *2-manifold* polygonal mesh such as the one in Figure 4.2(c), and the $\mathcal{HE}$ data structure we review below is originally designed for this category of meshes.

* A review of the data structures to represent a polygonal mesh.

The developing of an efficient data structure to store and represent a polygonal mesh has been an important topic in solid modeling. A suitable data structure provides efficient connectivity query between the mesh elements (faces, edges, vertices) and the modification of them; it as well avoid duplications in the representation. We survey a few basic mesh data structure in the footnote. [3]

* A review of the half-edge data structure.

The **Half-Edge** ($\mathcal{HE}$) data structure [114, 113, 69]) is a popular *edge-centered* data structure originally proposed to describe a *2-manifold* mesh. Its basic element is a *half-edge*, which describes the *incidence relationship* between a mesh face and a mesh edge. Since originally only the 2-manifold mesh is considered, such *edge-face* incidence corresponds to only "half" of an edge, in that, an edge can only share at most two faces (two faces at the interior and one face at the boundary), Figure 4.3(b). Specifically, we include a C++ implementation of the $\mathcal{HE}$ data structure in Figure 4.3(c). The "next" pointers of several half-edges form a circular list describing all boundary edges of a mesh face. The "pair" pointer of an half-edge enables to "navigate" across the other half-edge to fetch the two ending vertices of the current edge. The representation is decently optimized for a 2-manifold mesh.

The original $\mathcal{HE}$ only handles a *regular 2-manifold* mesh with several limitations: It can not represent a mesh with *(i)* a *degenerate* vertex as shown in Figure 4.2(e), which if removed, the mesh is disconnected, or *(ii)* a *degenerate* edge which is with no incident face(s).

4.3 Extend the Half-Edge Data Structure to Represent General Non-Manifold Polygonal Meshes

We extend the $\mathcal{HE}$ data structure with the following *two* capabilities to represent a *general polygonal mesh* defined in Definition 2: [4]

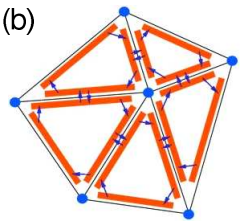
[3]The **Indexed Face Set** (IFS) [82, p.473] is a common data structure for mesh storage and visualization (used in OpenGL and VRML), but it is not suitable for mesh connectivity queries. The **Winged-Edge** ($\mathcal{WE}$) data structure [16] can be viewed as the precursor of the half-edge ($\mathcal{HE}$) data structure. For recent development of the mesh data structures in *non-manifold* object modeling, see *e.g.*, [79, 80, 17].

[4]The author gratefully thanks J. Mundy for helpful discussions in developing this data structure.

(a)

	V	E	F
V		-	
E	x		-
F		x	

(b)



(c)

```
struct HE_edge{
        HE_vert* vert;   // vertex at the end of the half-edge
        HE_edge* pair;   // oppositely oriented adjacent half-edge
        HE_face* face;   // face the half-edge borders
        HE_edge* next;   // next half-edge around the face
};
struct HE_vert{
    float x;
    float y;
    float z;

    HE_edge* edge;  // one of the half-edges emantating from the vertex
};
struct HE_face{
    HE_edge* edge;  // one of the half-edges bordering the face
};
```

Figure 4.3: The *half-edge* ($\mathcal{HE}$) data structure adapted from the *flipcode* web site (http://www.flipcode.com/archives/The_Half-Edge_Data_Structure.shtml). (a) shows a table describing the topological relationship between mesh faces, edges, and vertices. A topologically valid representation allows to access the edges of a face and the ending vertices of an edge, but not *vice versa*. (b) is an example mesh represented in the $\mathcal{HE}$ data structure. (c) shows the C++ code implementing the $\mathcal{HE}$ data structure.

- *Handle non-manifold junctions.* By extending the couple of half-edges in the $\mathcal{HE}$ in representing an edge, we make the "pair" pointer of a half-edge to form an ordered circular loop around a non-manifold junction edge, Figure 4.4(a). [5]
- *Handle degenerate mesh configurations.* We explicitly represent the mesh *edge* in our data structure, Figure 4.4(a), such that a degenerate edge (with no incident faces) can be represented. In addition, we store the *vertex-edge* incidence at each vertex with a dynamic array. [6]

Our new data structure is named the **Extended Half-Edge** ($\mathcal{EHE}$) data structure. It is *effective* in storing only the minimum set of information to represent the general polygonal mesh, including the non-manifold cases and all degenerate configurations. It is also *efficient* in providing instant access to the incidence relationship between all mesh faces, edges, and vertices in traversal.

4.4 The Extended Half-Edge ($\mathcal{EHE}$) Data Structure to Represent the $\mathcal{MS}$ Hypergraph

This section describes a further extension of the $\mathcal{EHE}$ to represent the topology of the $\mathcal{MS}$ hypergraph. We first define the notion of two types of hypergraphs and then describe the data structure to representation them as follows.

[5] The half-edge in this circular list can be called a *partial-edge* instead of a "half" edge.

[6] This is the only dynamic array necessary in our mesh data structure, which can be efficiently implemented using the C++ vector of the Standard Template Library (STL).

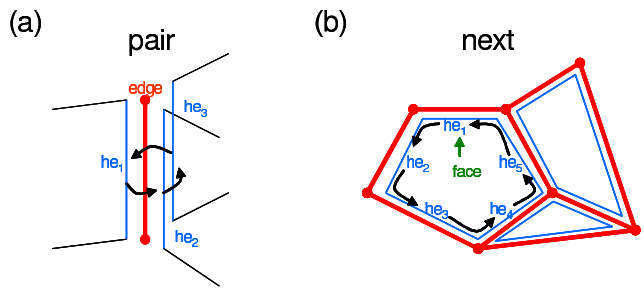


Figure 4.4: The *extended half-edge* ($\mathcal{EHE}$) data structure to represent the *non-manifold* polygonal meshes. (a) The half-edge's pair pointers form a circular loop to represent the *edge-face* incidence around a junction. (b) The half-edge's next pointers form a circular loop to represent all *edge-face* incidence of a face at its boundary.

Definition 3 *An* **ordered topological hypergraph** *is a collection of vertices V, edges E, and hyperedges H such that:*

$$V = \{v_i \mid i = 1, ..., n_v\} \tag{4.4}$$

$$E = \{e_j = \{v_a^j, v_b^j\} \mid v_a^j, v_b^j \in V, j = 1, ..., n_e\} \tag{4.5}$$

$$H = \{h_k = (e_1^k, e_2^k, ..., e_m^k) \mid e_1^k, e_2^k, ..., e_m^k \in E, e_i^k \cap e_{(i+1)\ mod\ m}^k = v_i^k, \quad i = 1, ..., m, m > 2, k = 1, ..., n_h\} \tag{4.6}$$

Definition 4 *A* **geometric hypergraph** *is an* ordered topological hypergraph *where each vertex is a point in $\mathbb{R}^3$, each edge is a curve in $\mathbb{R}^3$ with two ending vertices in V, each hyperedge is a surface in $\mathbb{R}^3$ bounded by the edges in E. No two edges intersect except at the ending vertices, and no two hyperedges intersect except at the* boundary *edges. The* internal boundary *on a hyperedge surface is called an* **internal *anchor curve*** *of it. A hypergraph is* **complete** *if it has no isolated edges nor vertices.*

The geometric hypergraph defined above can be viewed as a generalized case of a polygonal mesh defined in Definition 2, in that a mesh edge is now allowed to deform into a curve, and a mesh face is deformed into a surface.

Observe in Figure 4.1 for the dual-scale representation of the $\mathcal{MS}$, that the *incidence relationships* between the fine-scale mesh face/edge/verex elements and the coarse-scale hypergraph sheets/curves/vertices are **identical** in a local sense. The major extension in the $\mathcal{MS}$ hypergraph is that non-planar variations of the sheets (in comparison to the mesh face as a planar polygon) may cause additional topological variations. We describe the new cases below.

* Three *general* types of topological connectivity of the medial sheets.

A coarse-scale medial sheet S in general may contain the following *three* types of boundary topology, based on Giblin and Kimia's analysis of the generic $\mathcal{MA}$ transitions [88], Figure 4.5:

- (One-incident) *boundary curve* including the A_3 *ribs* and A_1^3 *axials* bordering with other sheets. The boundary curve of any internal *'void'* of the sheet S, which can be either a hole or part of other sheets, also belongs to this category.
- (Two-incident) *anchor curve* internal to S where another medial sheet (such as a *tab*) intersects with this sheet S.

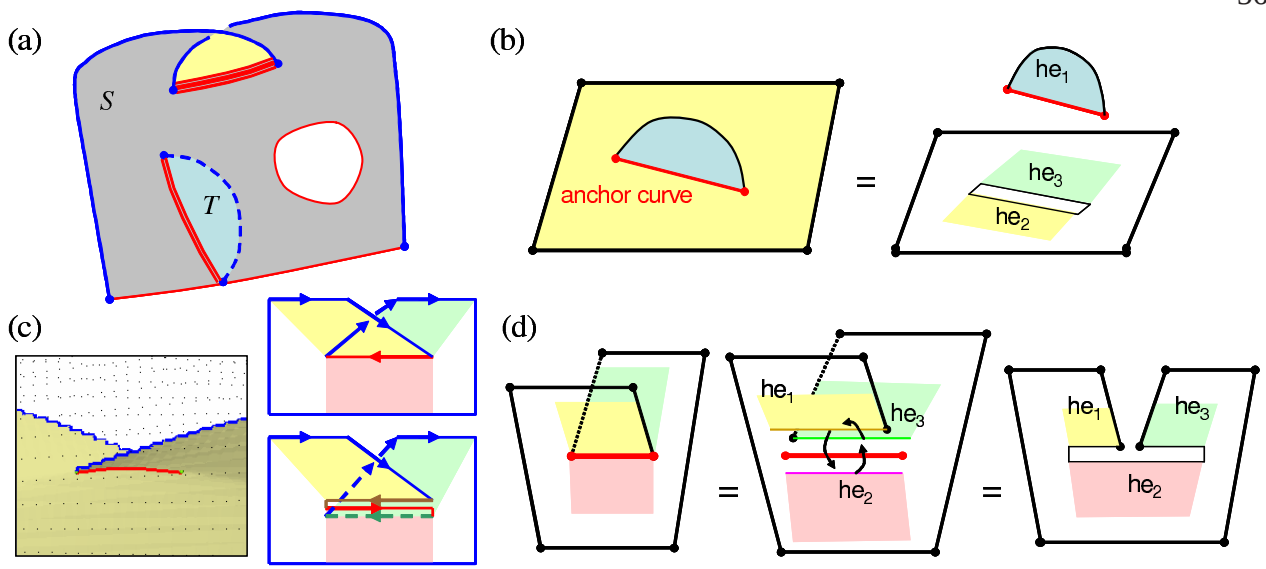


Figure 4.5: The general topology of an A_1^2 shock sheet $\mathcal{S}$ in the $\mathcal{MS}$ hypergraph and its representation using the *extended half-edge* ($\mathcal{EHE}$) data structure [43]. (a) Three possible types of sheet-curve incidence of $\mathcal{S}$: the boundary curve, anchor curve (with a tab $\mathcal{T}$), and swallow-tailed self-intersection. The topology of the 2-incident anchor curve (double red curve in (a)) can be represented by two half-edges in (b). (c) The A_1^3 curve is triply incident to $\mathcal{S}$, where the 3 incidence can be ordered to create a loop in $\mathcal{S}$'s boundary chain. (d) In the $\mathcal{EHE}$ representation, the 3-incident swallow-tailed self-intersection (triple red curve in (a)) can be represented by 3 half-edges both in a *loop* (at the junction) and in a *chain* in (d).

- (Three-incident) *swallow-tailed* self-intersection of S, which occurs near an A_5 transition (detailed in [88] and Chapter 5). This configuration can be viewed as that the A_1^3 curve is triply incident to S in a loop, and the three incidences are ordered as a single boundary chain (bordering the boundary of S), Figure 4.5c.

The three types of medial sheet topology coincide with James Damon's study in the global $\mathcal{MA}$ topology [58, p.2390] (in decomposing the $\mathcal{MA}$ into *irreducible* medial components). The above three types cover all cases we observed in practice; no further case is encountered in all our experiments.

We are now ready to present a proper data structure to describe the $\mathcal{MS}$ hypergraph topology. We have described the topology of the medial *sheets* above, and the topology of medial curves and nodes are relatively simple: *(i)* an A_1^n medial *curve* is the intersection of n medial sheets ($n \geq 3$), and *(ii)* a medial *node* is the intersection of several medial curves and sheets, where only the incidence relationship between the node-curve needs to be explicitly stored.

* Representing the topology of a medial sheet: the *boundary curve chain* and *internal curve chains*.

We further apply the $\mathcal{EHE}$ data structure described above to handle the $\mathcal{MS}$ hypergraph as a *geometric hypergraph*. [7] Specifically, the mesh edge (line segment) is now a medial curve; and the mesh face (polygon) is now the medial sheet. The half-edge now represents each *sheet-curve* incidence, in that each boundary of the medial sheet can be organized into a ***chain* of half-edges** in three cases:

[7] We can view the $\mathcal{MS}$ hypergraph is a special case of a *geometric* hypergraph: Any representation to describe a geometric hypergraph are capable to describe the $\mathcal{MS}$ hypergraph.

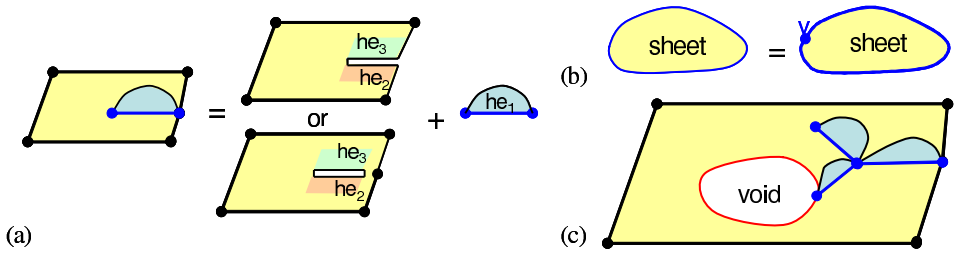


Figure 4.6: Degenerate medial sheet topology. (a) An anchor curve intersecting at the sheet boundary can be viewed as two way in the $\mathcal{EHE}$ representation. *(i)* a sheet with a single boundary chain given from an Euler tour, which contains a (degenerate) overlap, or *(ii)* a boundary chain and a separate internal *anchor curve* chain. (b) A sheet with a closed curve (with no ending node) as boundary can be represented in $\mathcal{EHE}$ by introducing a "dummy" node. (c) An example that the combination of the three types of general sheet topology could create complex configuration this requires a *canonicalization*.

- *A boundary curve chain* to describe the outer boundary of a medial sheet. [8]
- *An internal curve chain* to describe the inner boundary of a medial sheet. [9]
- *An internal curve pairs* to describe the internal *anchor* curves (2-incidence), which can be viewed as a degenerate case of an internal curve chain of two half-edges.

* Handle *degenerate* cases in the medial sheet topology.

While the three general types of $\mathcal{MS}$ sheet topology is explicitly handled above, the combination of them could create *degenerate* configurations which cause complication in handling the $\mathcal{EHE}$ data structure. Specifically, the degenerate configuration could cause more than one valid representations in the $\mathcal{EHE}$. Figure 4.6(a) depicts one example, where two interpretations using the $\mathcal{EHE}$ are both valid. We list more cases observed in practice below:

- Medial sheet with a closed curve (loop with no ending node) as boundary, Figure 4.6(b).
- Medial sheet with internal voids, where the internal boundary has no distinction from the outer boundary.
- Other combinations of the above cases could produce more complication, such as the one in Figure 4.6(c).

* Canonicalization of $\mathcal{MS}$ sheet topology in the $\mathcal{EHE}$ representation.

The above complications of the multiple representations of the degenerate $\mathcal{MS}$ topology can be handled by a *canonicalization* of the $\mathcal{EHE}$ representation. Specifically, we convert the $\mathcal{EHE}$ into a **canonical form**, which effectively remove the non-standard but valid representations. This can be done by comparing different representations for equivalence and standardize their sorting order. This canonicalization of medial sheet topology will be useful in two ways in our implementation: *(i)* in editing the $\mathcal{MS}$ topology in the $\mathcal{MS}$ transforms in Chapter 5 and *(ii)* in constructing of the coarse-scale $\mathcal{MS}$ sheets in Chapter 7.

[8] The A_5 swallowtail (self-intersection) is a special case of a boundary curve, Figure 4.5(c), where the half-edges can also be ordered in a sequential chain.

[9] The internal curve chains can be treated identical to a boundary chain, since topologically they are equivalent. We explicit distinguish the (unique) boundary chain for a major reason, that it provides a better compatibility in designing the C++ class hierarchy: a sheet (surface) is a sub-class (an extension) of a mesh face, where only a boundary chain is defined.

Chapter 5

Medial Scaffold Transforms

* Overview of chapter: $\mathcal{MS}$ transforms to make equivalent the $\mathcal{MA}$ across transitions.

This chapter describes the **medial scaffold** ($\mathcal{MS}$) **transforms**, a *symmetry transform* that perform the edit operations of the $\mathcal{MS}$ to make equivalent the $\mathcal{MS}$ across *transitions* (sudden topological changes) as well as jointly deform the shape surface. As described in Chapter 1, that the omnipresent *instabilities* of the $\mathcal{MA}$ are analyzed as transitions, and the generic case of $\mathcal{MA}$ across transitions can be making equivalent by editing the $\mathcal{MA}$ toward the transition point. In this chapter, we define such edit operations corresponding to each transition case as *transforms*. Specifically, the set of $\mathcal{MS}$ transforms explicitly modify the *dual-scale* structure of the $\mathcal{MS}$ (hypergraph/mesh) described in Chapter 3 to implement the edit operations. More over, the result of the $\mathcal{MS}$ transforms is thus a $\mathcal{MS}$ hypergraph of higher and higher order of symmetry (more degenerate), where its instabilities are removed and being regularized (recall the idea in Chapter 1 that degenerate $\mathcal{MA}$ of higher symmetry is simplier and more significant). The system of $\mathcal{MS}$ transforms is our main tool to regularize our shape representation (Chapter 7) as well as the core platform to match shapes (Chapters 8 and 9).

The development of the $\mathcal{MS}$ transforms is motivated from two aspects:

- *Remove the $\mathcal{MS}$ instabilities for regularization.* In order to use the $\mathcal{MS}$ as a shape representation to model and match shapes, the $\mathcal{MS}$ structure must be robustly regularized (simplified) in removing the instabilities. Also, this regularization should not simply yield a simplified skeleton, the corresponding shape must be updated consistently to couple with the skeleton, which essentially *symmetrize* the shape.
- *Edit the $\mathcal{MS}$ across transitions to match shapes.* Recall in § 1.1 for the notion that $\mathcal{MA}$ transitions are used to characterize a deformation path between two arbitrary shapes (Figure 1.3(b)), the $\mathcal{MS}$ transforms thus can be used as tool (of edit operations) to search for a *theoretically* optimal deformation path between shapes in an edit-distance metric (Figure 1.4). This theoretical setup will be explored more in Chapter 8.

* The *five* general steps of a $\mathcal{MS}$ transform.

We organize each $\mathcal{MS}$ transform into *five* elementary operations/steps, which shall be defined for all transforms. We will describe each of them in details in the remaining of this chapter.

1. *Detection* (condition) of a transform: identify whether any part of a $\mathcal{MS}$ hypergraph is close to any valid transition point.

2. *Cost* (saliency measure) of a transform: further refine the selection of all candidate $\mathcal{MS}$ transforms and rank order them. The cost is used to prioritize the transforms in reflecting their effect in regularizing the $\mathcal{MS}$, in that transforms with slight modification of the shape *w.r.t.* their scale should be prefer in prior to others when considered in regularization.
3. *Operation* on editing the $\mathcal{MS}$ hypergraph: edit operation of the $\mathcal{MS}$ in the dual-scale structure, including the topology changes in the coarse-scale hypergraph and the underlying fine-scale mesh changes. [1]
4. *Update* of the shape boundary: identify which part of the shape boundary need to be modified to reflect the change in the skeleton and update it. This can be done by referring to the corresponding boundary sample points of the fine-scale $\mathcal{MS}$ mesh elements and determine how to organize them after the transform (Refer to § 7.3 and the details of each transform below).
5. *Tracking* of all modified $\mathcal{MS}$ components to consider other future transforms. This can be done by tracking all $\mathcal{MS}$ sheets/curves/nodes modified in the above edit operation and re-evaluate them in the detection step and obtain their update costs. [2]

* Result of the $\mathcal{MS}$ transforms: $\mathcal{MS}$ nodes of higher-order degeneracy.

As the transforms are applied to regularize the $\mathcal{MS}$, it essentially moves the resulting $\mathcal{MS}$ toward higher-order of degeneracy. Specifically, in the $\mathcal{MS}$ hypergraph, the $\mathcal{MS}$ sheets are gradually removed (by pruning-like operations) and the $\mathcal{MS}$ curves are also removed (by contact-type operations). What is remaining is the $\mathcal{MS}$ nodes of higher and higher orders of contact. (Recall the degenerate contact typology of the $\mathcal{MS}$ nodes described in § 3.1.1.) We will discuss the analysis of these nodes, with a result to decompose an arbitrary high-order $\mathcal{MS}$ node into equivalent *generic* low-order nodes (§ 5.5).

* Organization of chapter.

The remaining of this chapter is organized as follows. Section 5.1 reviews the generic $\mathcal{MA}$ transitions in 2D and 3D, which motivates the development of the $\mathcal{MS}$ transforms. Section 5.2 defines the three main types of *generic* transforms (splice, contract, merge) for a simple closed shape, Section 5.3 continues to describe additional transforms (gap and loop) to handle non-closed shapes, and Section 5.4 further describes additional transforms to handle *non-generic* transitions, which we found useful in practice. Finally, Section 5.5 analyzes the resulting high-order $\mathcal{MS}$ nodes producing by the $\mathcal{MS}$ transforms.

5.1 A Review of Generic $\mathcal{MA}$ Transitions and Corresponding $\mathcal{MS}$ Transforms

We first review the $\mathcal{MA}$ transitions and transforms defined in 2D, which are easier to understand and serve as a background to review the 3D cases.

* Review of 2D generic $\mathcal{MA}$ transitions and *shock graph* transforms.

[1] The resulting $\mathcal{MS}$ hypergraph should be converted into the canonical form as described in § 4.4.

[2] This step can be viewed as a post-processing after a transform is done in an greedy iterative scheme for regularization in Chapter 7. We skip the implementation details of this as we elaborate each transform below.

We summarize Giblin and Kimia's study on the 2D $\mathcal{MA}$ transitions [89] as follows. First, the 2D $\mathcal{MA}$ can be organized into a *directed* graph, namely the *shock graph* ($\mathcal{SG}$). For a shape of a *simple closed* boundary curve, the $\mathcal{MA}$ / $\mathcal{SG}$ transitions are formally classified into a complete set of **six** generic transitions, Figure 5.1(a), by slightly perturb the shape and observe how the $\mathcal{MA}$ topology changes by *e.g.*, growing of an axis, or swapping of $\mathcal{MA}$ branches. Accordingly, **three** transforms ($\mathcal{MA}$ / $\mathcal{SG}$ edits) are defined to make the $\mathcal{MA}$ / $\mathcal{SG}$ graph structure equivalent across these transitions, Figure 5.1(b), namely:

- The *splice* transform deletes a (leaf) shock branch and merges the remaining two.
- The *contract* transforms (two types) delete a shock branch between the degree-three nodes.
- The *merge* transforms (three types) combine two branches at a degree-two node.

The transforms effectively simplify the $\mathcal{MA}$ / $\mathcal{SG}$ topology to remove instabilities. For example, the *splice* transform in Figure 5.1(b) upper illustrates how a splice transform removes a $\mathcal{MA}$ branch (leaf) corresponding to a small perturbation of the shape (the A_1A_3 transition in Figure 5.1(a)).

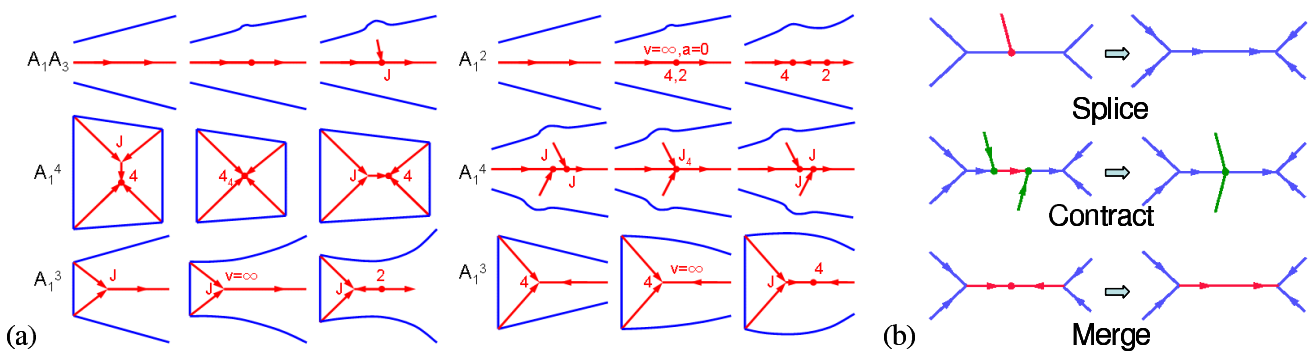


Figure 5.1: (a) (From [88, Fig.1].) *Six* types of $\mathcal{MA}$ / $\mathcal{SG}$ **transitions** for closed shapes in the 2D case [89]. The central shape in each group represents the transition point in the deformation from left to right, or right to left columns. (b) (From [169, Fig.15].) *Three* types of $\mathcal{SG}$ **transforms** (edit operations) which make equivalent the $\mathcal{MA}$ graph topology across transitions [169].

* Review of the 3D generic $\mathcal{MA}$ transitions for *simple closed* shapes.

The extension of the $\mathcal{MA}$ transition analysis from 2D to 3D is analogical, however non-trivial due to the additional dimensionality. In addition to the 2D work, Giblin and Kimia study the 3D $\mathcal{MA}$ local form [87] and classify the generic 3D $\mathcal{MA}$ transitions for *single closed* shapes only recently [88]. Their result is summarized in Figure 5.2 and 5.3, where a set of mathematical parameterized shapes are deformed [3] to simulate the $\mathcal{MA}$ across the seven *generic* types of transitions. We illustrate two examples as follows: *(i)* For an A_1A_3-I transition, refer to the shape of two parallel planes where one of them is perturbed by a Gaussian "bump". This bump corresponds to a shock sheet ("tab") anchoring on another shock sheet. [4] As we gradually smooth out the bump, the shock tab eventually disappears, which is precisely the transition point. This transition is named A_1A_3-I since it is locally an A_1A_3 contact [5] and this is one of the two cases of transitions that occurs in the A_1A_3 point.

[3] in a one-parameter family of deformations [88], refer to footnote 7 in § 1.2.

[4] Terminology: The shock sheet with no other sheets anchored on it is called a shock **"tab"**, which is the *leave* of a $\mathcal{MS}$ hypergraph, corresponding to a bump on the surface.

[5] Terminology: The A_1A_3 point is called a **fin** point of a shock *tab*.

(A_1A_3-II is the other transition which merges two A_1A_3 points.) *(ii)* For an A_1^5 transition, refer to the triangular prism-like shape in Figure 5.2. As we squish or elongate the prism, the center $\mathcal{MA}$ curve/sheet eventually shrink to a point, where the local form is an A_1^5 point (regular tangency at five distinct points). Other transitions can be similarly interpreted. We will address the A_5 transition as we present the A_5 transform below. For more detail, refer to [88].

* Three origins of the $\mathcal{MS}$ transforms *w.r.t.* the $\mathcal{MA}$ transition analysis.

Note that the above generic seven transitions are analyzed on the set of simple closed shapes. As mentioned before we handle point-sampled shapes in practice, thus additional transitions/transforms shall be introduced. Also, in practice we observe transforms that are *non-generic* (not belonging to the above seven cases) that are significant. These results in the development of $\mathcal{MS}$ transforms in each of the above three cases, summarized as follows: This chapter formally defines the operations of transforms to edit the $\mathcal{MA}$ toward the transition points.

1. $\mathcal{MS}$ *transforms for* **simple closed** *shapes*: A total of 11 of splice-type (pruning a medial tab and "splice" together remaining sheets), contract-type (shrinking of a medial sheet or curve), and merge-type (bringing together medial nodes or curves) transforms are defined in a case-by-case analysis of the seven generic transitions. (Detailed in § 5.2.)
2. $\mathcal{MS}$ *transforms for* **non-closed** *or sampled shapes*: Two sets of transforms, namely, the gap-type (removing a medial branch and close a gap in the boundary shape) and the loop-type (removing a boundary point thus removing the medial curves/sheets bounding the boundary point) are defined for handling non-closed shapes or sampled shapes, *i.e.*, unorganized points. (Detailed in § 5.3.)
3. $\mathcal{MS}$ *transforms for additional* **non-generic** *transitions*: Additional merge-type transforms only occur in the degenerate cases such as the man-made objects of high regularity are defined to handle these cases, which we found useful in practice. (Detailed in § 5.4.)

Table 5.1 overviews the system of five types of transforms—namely, the *splice*, *contract*, *merge*, *gap*, and *loop*, totally 16 transforms (11 generic, 3 for non-closed shapes, 2 degenerate). We found them sufficient to handle all shape deformations we observed in practice. Below we describe each of them in details.

Table 5.1: Summary of the system of $\mathcal{MS}$ transforms.

Category	Operation on	Transforms
Splice	sheet	A_1A_3-I sheet splice, $A_1^2A_3$-II sheet splice.
Contract	curve	A_5 curve contract, $A_1^2A_3$ curve contract, A_1^5 curve contract.
	sheet	A_1^5 sheet contract, A_1^4 sheet contract.
Merge	node-node	A_1A_3-II node-node merge, $A_1^3A_3$-I node-node merge*.
	node-curve	$A_1^2A_3$-I node-curve merge, corner node-curve merge*.
	curve-curve	A_1^4 curve-curve merge, A_1A_3-II curve-curve merge.
Gap	sheet	A_1^2 sheet gap transform.
	curve	A_1^3 curve gap transform.
Loop	generator	generator-loop transform.

* Non-generic (higher-order) transforms.

5.2 Generic $\mathcal{MS}$ Transforms for Simple Closed Shapes

* The *eleven* $\mathcal{MS}$ transforms for the *seven* generic $\mathcal{MA}$ transitions.

This section presents a case-by-case analysis of $\mathcal{MS}$ transforms for the seven generic transitions in Figure 5.2. Refer to Figure 5.3 for another illustration with the medial sheets implicit to better visualize the structure.

Among the seven transitions, some involve only a single way of deformation toward simplification of shape and the $\mathcal{MA}$, while the others can be approached from both sides (as the red arrows indicates), thus there are **11** cases of transforms. Depending on whether a $\mathcal{MS}$ node, curve, or sheet is the main component under modification, the transforms can be organized into *three* main types:

- The **splice-type** transforms prune a medial sheet branch and "splice" together the remaining two sheets.
- The **contract-type** transforms contract a medial sheet or a medial curve.
- The **merge-type** transforms merge together two medial components, *i.e.*, bringing two medial nodes or two medial curves together, or join together a medial node to a curve. [6]

These transforms can be further classified depending on whether the transform operates on a medial sheet, curve, or node, into 6 sub-categories: *sheet-splice*, *curve-contract*, *sheet-contract*, *node-node merge*, *node-curve merge*, and *curve-curve merge*, the second column in Table 5.1.

5.2.1 Splice Type Transforms

The **splice-type transforms** remove a medial sheet "tab" (sheet without an interior anchor curve, where another sheet anchored on it) T and merge the remaining sheets at the remaining A_1^3 axial curve C, as illustrated in Figure 5.4. The traditional *pruning* of the $\mathcal{MA}$ can be recognized as the *splice* transform, but where our transform in addition to removing a medial sheet branch, also merges the remaining sheets.

* Estimate *splice* cost as the shape deformation cost weighted by the local *scale*.

Estimating Cost: We approximate the splice cost by the volume (V) of the corresponding 'bump' on the boundary, weighted by the scale, *i.e.*, local (average) shock radius; refer to Figure 5.5 for a 2D case and Figure 5.6 for a 3D simulation. Specifically, in 2D, the splice cost is the difference in area divided by the radius ($\Delta A/r$), while in 3D, the splice cost is $\Delta V/r$. [7] The computation of ΔV in 3D is closely related to how the boundary shape is organized to coupled with the medial tab under pruning, a topic will be explored in § 7.3. Here we assume such coupled boundary is available, as shown in an example in Figure 5.6.

A naive approach is to estimate ΔV directly. Specifically in Figure 5.6(e), the two blue curves are the *"boundary trace"* curves which correspond to the two contacting trace on the shape boundary as we traverse along the A_3 curve from one end to the other. Similarly, the two boundary trace

[6] The organizing of the 3D $\mathcal{MS}$ transforms is an analogy of the 2D case of splice, contract, and merge transforms. However, the "merge" transform has a different meaning. Note that in 2D the merge transform requires the notion of *shock flow* in the $\mathcal{SG}$, while the merge-type transforms in 3D work on the $\mathcal{MS}$ and do not involve the notion of shock flow.

[7] We also note that in 2D, $\Delta A/r \approx \Delta s$, which is the boundary length difference, Figure 5.5(c-d). In 3D, $\Delta V/r \approx \Delta A$, which is the boundary area difference. Also, we can further introduce more terms in additional to the volume and scale terms, such as the "stretching" term in two directions of the protrusion.

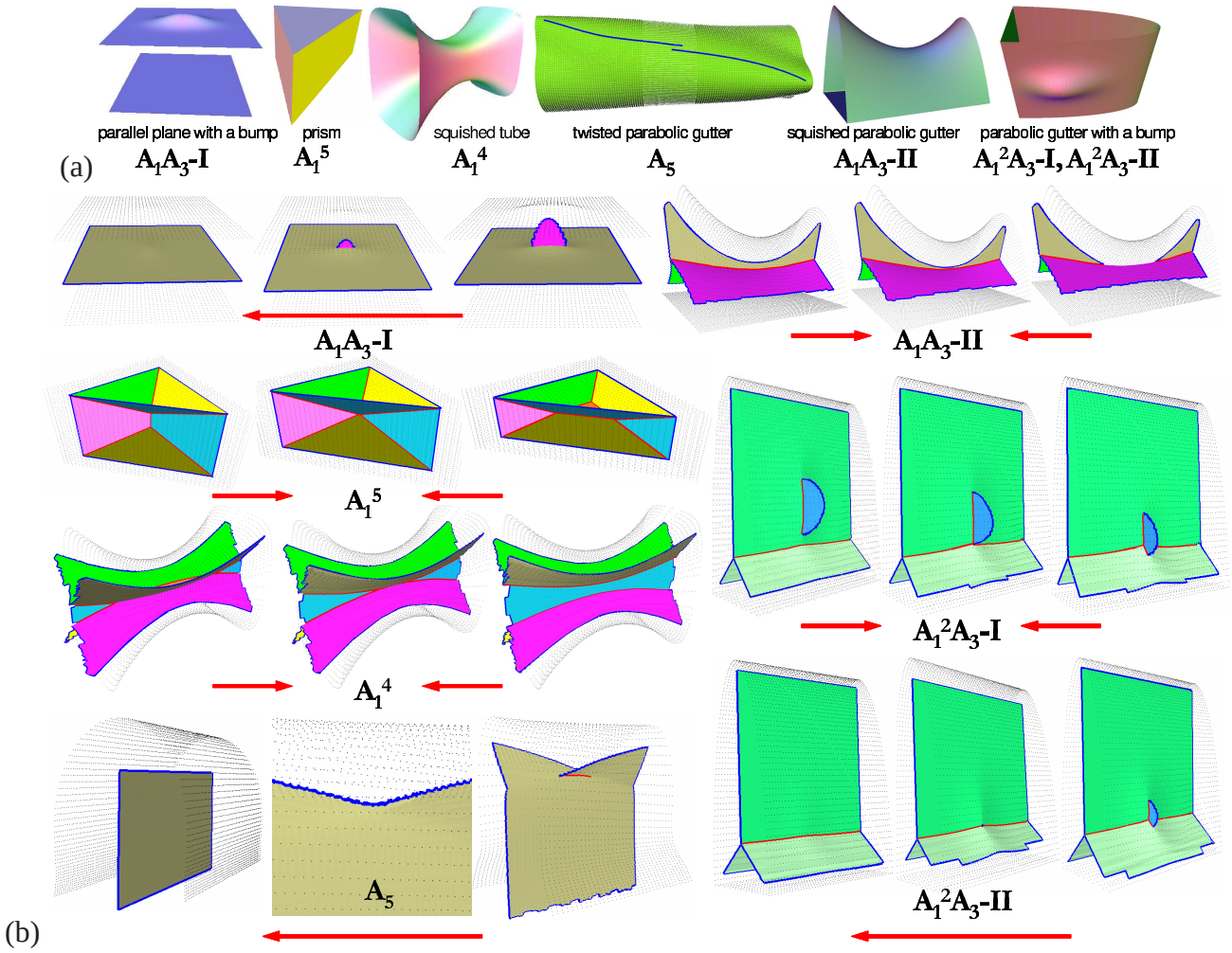


Figure 5.2: (From [43, Fig.2].) Illustration of some $\mathcal{MA}$ *transitions* in 3D [88]. (a) The example shapes near the transition points. (b) Simulation of the **seven** $\mathcal{MA}$ transitions case by case. Thee (red arrows indicates the **eleven** $\mathcal{MS}$ transforms corresponding to the seven transitions [43].

curves of the A_1^3 (yellowish brown) can be similarly defined. ΔV corresponds to the difference of the shape volume in between the blue and yellowish brown curves. While this estimation of ΔV is accurate for large shock tabs, it is not suitable for small tabs and is problematic in practice due to low sampling. The above boundary trace curves can contain topological or geometrical problem such as self-intersection or not enclosing a surface patch/volume, if the sampling density is not enough.

While the above accurate estimation of the splice cost can not be robustly achieved when sampling density is low, it does not create a serious problem. In fact, we found several approximations of ΔV works well in practice. For example, the convex hull of all associated boundary points estimates the approximation volume well. We further simplify the approximation in estimating the cost of the shock tab T as the number of associated boundary points. This is intuitive that, by assuming an roughly uniform sampling, as large shock sheet should have more coupled sample points, while the small tab has less. Another remark is that, initially there are typically many $\mathcal{MS}$ sheets to handle prior to the regularization (see Figure 7.11 for an example), the computational efficiency is more

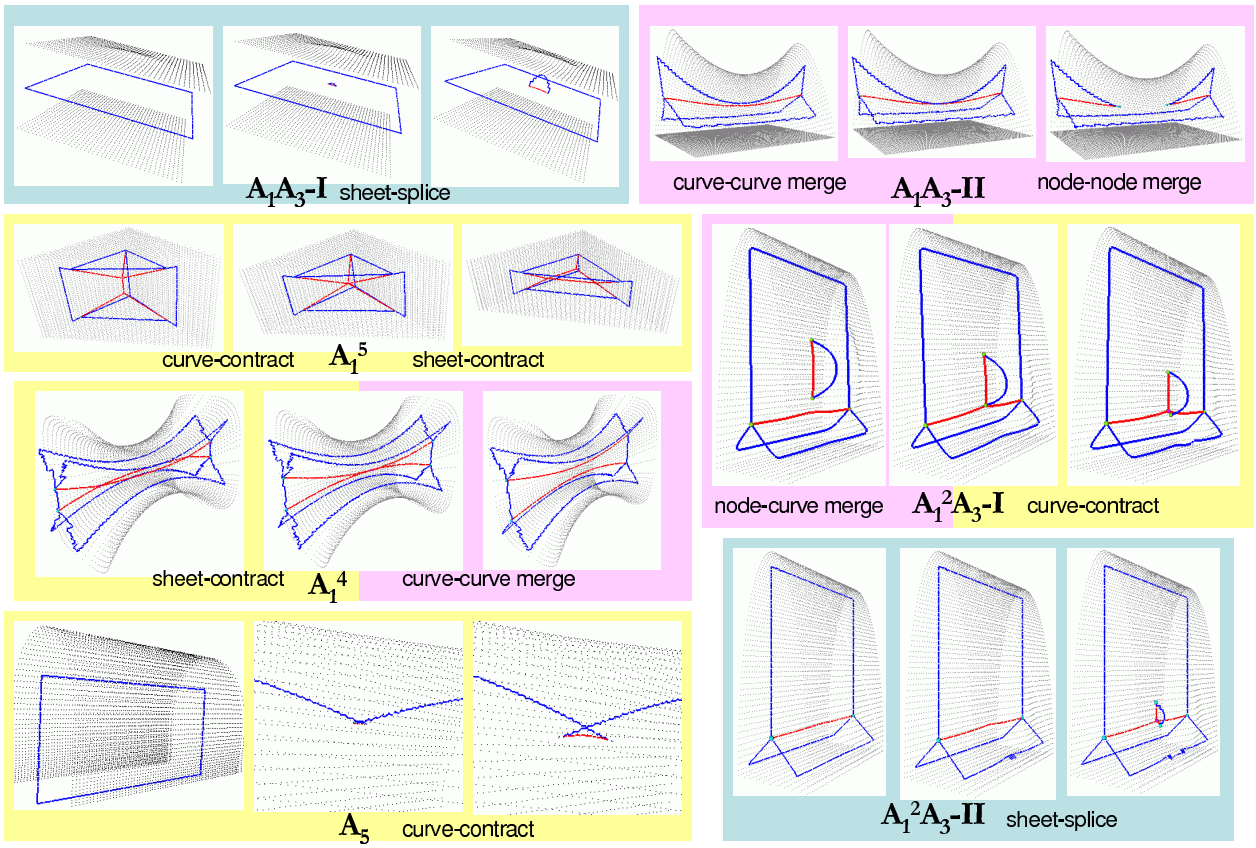


Figure 5.3: (From [43, Fig.2].) Continue from Figure 5.2, all medial sheets are hidden to better visualize their structures. The 11 $\mathcal{MS}$ transforms related to the 7 transitions are also labeled.

important than the accuracy in estimating pruning cost in practice. [8]

* Modify the shape boundary after the splice operation.

Updating shape: The approach to update the shape boundary after the splice transform is closely related to the "boundary trace" curves described above. In 2D, the boundary after a splice transform is approximated by a circular arc, Figure 5.5(d). In Figure 5.6, the removing of the shock tab essentially remove the surface patch in Figure 5.6)(a) and the task is to fill the "hole" on the boundary in Figure 5.6(b). This can be done by generating a new surface (a "cut-off" patch [126] corresponding to the protrusion) in a zig-zag fashion, along the two boundary trace curves of the A_1^3 curve C. This approach essentially approximate a "spherical" patch with a polygonal mesh, Figure 5.7. We note that this boundary updating step can be ignored for pruning of small tabs (since the difference in change is tiny in discrete approximation). [9]

There are two types of splice-type transitions (A_1A_3-I and $A_1^2A_3$-II), thus we shall have two types of splice-type transforms as follows.

A_1A_3-I sheet splice transform

Refer to Figure 5.4(a) for the edit operations of the $\mathcal{MS}$. We observe in practice in § 7.4).

[8] Comparing to [194, 188], where the pruning cost is estimated from the number of $\mathcal{MA}$ sheet elements, our splice cost estimation is better in refereing to the underlying shape.

[9] The splice transform also provides a good chance to fix the topological errors (if any) of the corresponding surface patch. The meshing of an initial shape mesh from input points will be detailed in Chapter 6.

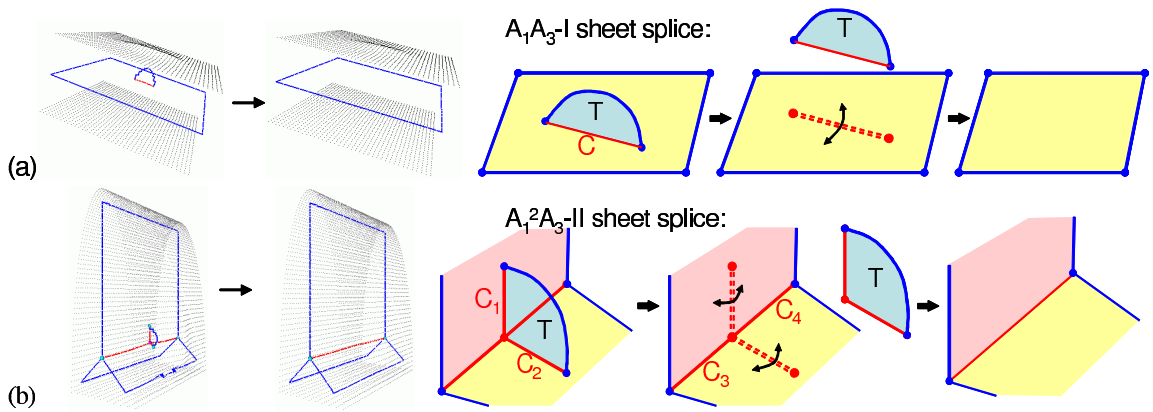


Figure 5.4: The *splice* transform removes a $\mathcal{MS}$ sheet (tab) corresponding to a bump on the surface and 'splice' together the remaining $\mathcal{MS}$ sheets. (a) the A_1A_3-I sheet splice and (b) the $A_1^2A_3$-II sheet splice transforms.

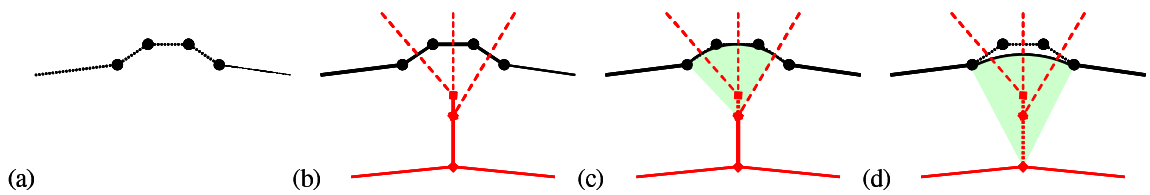


Figure 5.5: The application of *gap* and *splice* transforms and the update of shape contour in 2D. (a) A few boundary sample points and the putative boundary (in dots). (b) The boundary contour can be reconstructed by removing shock branches between samples (black dots) and connect them. The remaining shocks are shown in solid lines. (c) A splice transform prunes a shock branch and replace the boundary with a circular arc centered at the remaining shock node. (d) Further application of the splice transform prunes the whole shock branch out and smoothes the boundary into a larger arc.

$A_1^2A_3$-II sheet splice transform

Refer to Figure 5.4(b) for the edit operations of the $\mathcal{MS}$. In addition, this transform also merges the remaining $\mathcal{MS}$ curves C_3 and C_4 in Figure 5.4(b).

We observe in practice that the splice transforms are very effective in simplifying the coarse-scale $\mathcal{MS}$ topology and reducing the fine-scale mesh elements; refer to Figures 7.4 and 7.11 for examples.

5.2.2 Contract Type Transforms

The **contract-type transforms** contract a medial curve/sheet into a node. Depending on whether the change is near the *boundary* (A_3 rib) or at the *interior* (A_1^3 axial) of the $\mathcal{MS}$, they are organized into two sub-categories: *(i)* the A_5 and $A_1^2A_3$-I curve-contract transforms operate at the boundary, and *(ii)* the A_1^5 curve-contract and A_1^5/A_1^4 sheet-contract transforms operate at the interior of the $\mathcal{MS}$.

A_5 curve contract transform

The A_5 curve contract transform removes the *twisted swallow-tail* like self-intersection of a $\mathcal{MS}$ sheet S near an A_5 transition, as illustrated in Figure 5.8.

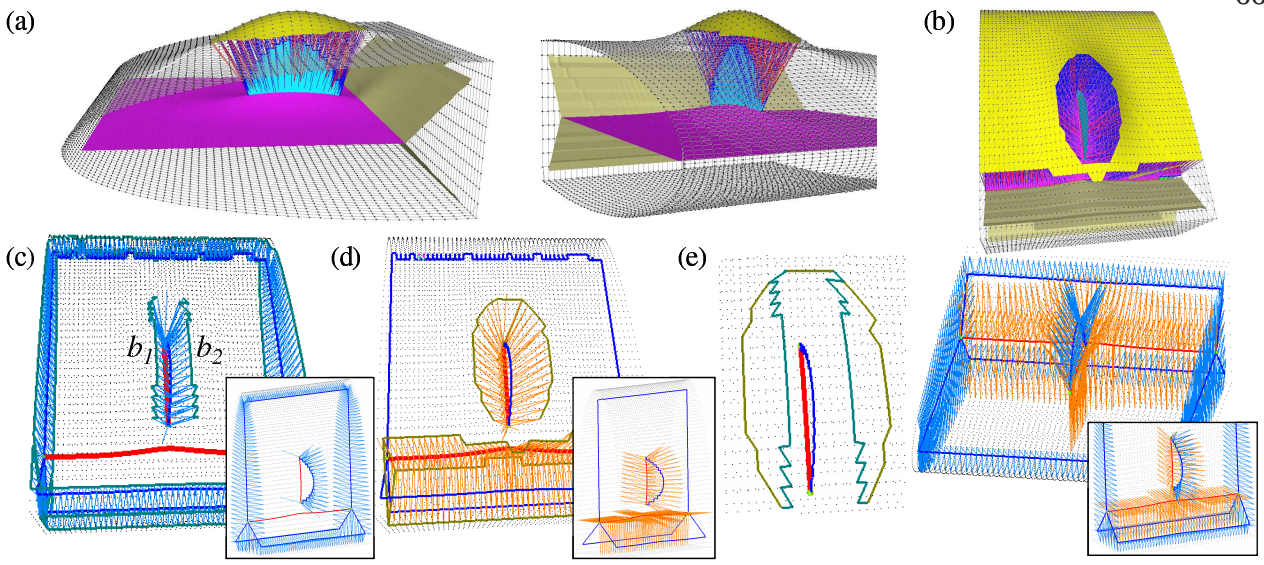


Figure 5.6: Illustration of the association of the boundary shape (input sample points in black) to the $\mathcal{MS}$ on a parabolic gutter shape perturbed with a bump, which induces an A_1^2 *tab*. (a-b) Two views of the association of the boundary points to the A_3 curves (blue) in cyan lines. (c-d) Two views of the association of the boundary points to the A_1^3 curves (red) in orange lines. (e) Zoom in the to bump surface with both *"boundary trace curves"* of the A_1^3 (yellowish brown) and A_3 (blue), where the two curves are superimposed to emphasize the region in between. (f) A side view of the association of the A_1^2 tab (light blue) in colored lines.

Details on the A_5 transition: The local form near an A_5 transition is analyzed by Giblin, Kimia, and Pollit [88] and summarized below, Figure 5.8. First, the A_5 is entirely local in nature (*i.e.*, there is no interference between parts of the surface which are far from each other). Second, the A_5 transition is closely related to the transitions of the two close-by *ridges* on the surface (corresponding the two A_3 ribs of the $\mathcal{MA}$) and the change of *turning points* of the ridge (from 'hyperbolic' to 'elliptic'); we omit details beyond the scope, see [88, Fig.22] for more details). Third, at the transition point and after it, there is no A_1^3 curve present (*i.e.*, only one A_3 rib curve remains as the generic case).

Operation of Transform: This transform can be done by identifying a middle point m on the A_1^3 curve C and *'trim'* out the two intersecting *wings* of swallow-tail (by applying splice transforms element-by-element), detailed as follows. First, in Figure 5.8(c), two trimming points u / v on the two A_3 rib curves R_1 / R_2 (respectively) are estimated with a proper scale. Secondly, two *trimming paths* $\overline{\mathsf{um}}$ / $\overline{\mathsf{vm}}$ are computed as a *geodesic* shortest path on S. The shortest path can be efficiently approximated by applying *Dijkstra's* algorithm [52], by treating the fine-scale mesh edges/vertices as a graph, so that no cutting through the fine-scale sheet elements is involved. [10]

Detection / Cost of Transform: This transform is applicable to any A_1^3 shock curve C ending at two A_1A_3 shock nodes (fin points) D and E, and C is triply incident to a *single* shock sheet S. Also the transform cost should be low. The transform cost is estimated as the local shape change weighted by its local scale, which in practice can be approximated by the length of the A_1^3 shock curve C.

[10]The complexity of running the Dijkstra's shortest path algorithm on the fine-scale $\mathcal{MS}$ elements of sheet S is $O(e + v \log v)$ amortized time, where e and v are the number of edges and vertices in S, respectively, if the Fibonacci heap is used to implement the priority queue, with a reasonable assumption that the graph of the fine-scale mesh is sparse [52].

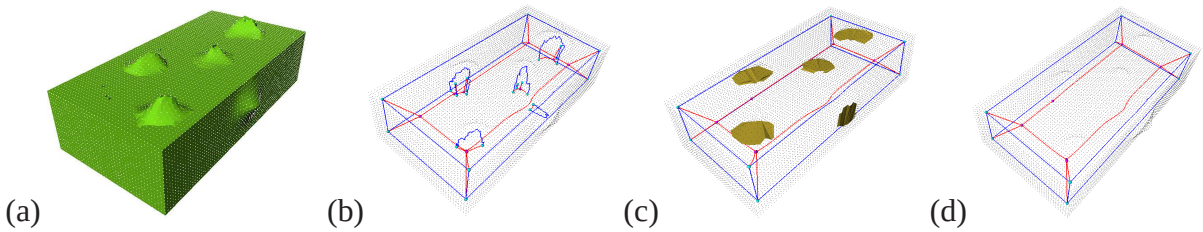


Figure 5.7: (From [126, Fig.6].) (a) Point samples from a rectangular box, deformed by 5 protrusions (4 on top, 1 on a side). (b) The $\mathcal{MS}$ shows five transitions which when removed both smooth the surface (c) as well as regularize the scaffold (d).

Detailed Operations: To ensure that the swallow-tail topology is completely removed (otherwise this transform is not valid), one needs to constrain the above *trimming path* finding in applying the shortest path searching, such that the trimming paths are *(i)* as short as possible and *(ii)* topologically valid. Specifically, in Figure 5.8(d), the two trimming paths $\overline{\mathsf{um}}$ and $\overline{\mathsf{vm}}$ together with the A_1^3 curve C must fully delineate the set of fine-scale A_1^2 face elements $\{\mathsf{F_i}\}$ to prune, such that the swallow-tail topology is completely removed.

We describe the detailed algorithm in finding the valid trimming paths $\overline{\mathsf{um}}$ and $\overline{\mathsf{vm}}$ to identify $\{\mathsf{F_i}\}$ as follows. *(i)* First, we determine the middle point as a fine-scale vertex element m on the A_1^3 axial curve C. [11] *(ii)* Secondly, the two trimming points u and v on the two A_3 rib curves $\mathsf{R_1}$ and $\mathsf{R_2}$ (respectively) are estimated such that the curve length of $\overline{\mathsf{Ev}}$ and $\overline{\mathsf{Du}}$ are similar to the length of C. *(iii)* Third, if the valid shortest paths of $\overline{\mathsf{mu}}$ and $\overline{\mathsf{mv}}$ are successfully found, the set of faces $\{\mathsf{F_i}\}$ to pruning can be successfully labeled. *(iv)* Otherwise, the steps of (i) to (iii) are to be repeated, by first trying a nearby candidates of u and v in a *zig-zag* fashion [12], and trying a different m, until a valid pair of trimming paths are found. [13] *(v)* Once the trimming paths $\overline{\mathsf{mu}}$ and $\overline{\mathsf{mv}}$ are found, the set of faces $\{\mathsf{F_i}\}$ to prune can be determined by a connected component labeling, by setting all A_1^2 face elements incident to the A_3 rib curves $\mathsf{R_1}$ and $\mathsf{R_2}$ as seeds and propagate until $\overline{\mathsf{mu}}$ or $\overline{\mathsf{mv}}$ is reached, or the A_1^3 axial curve C itself is reached. [14]

Once the set of trimming faces $\{\mathsf{F_i}\}$ is identified, they can be pruned by performing splice transforms element-by-element in sequence, from the bordering ones to the interior ones with (ordered by their maximum radius). As we perform the element-wise splice transform on $\{\mathsf{F_i}\}$, we also pass their associated generators to the remaining (neighboring) A_1^2 face elements. One important reason to ensure such pruning order is to maintain a consistent boundary-to-$\mathcal{MS}$ association. (See § 7.3 for

[11] We handle the degenerate case when C contains only one fine-scale A_1^3 edge element, such that m coincides with D or E.

[12] Namely, start with an initial position p and try the immediate positions, and then near-by positions, in the sequence of $p-1$, $p+1$, $p-2$, $p+2$, ..., until all positions are explored, or a pre-defined stopping criterion is reached.

[13] In this strategy, the trimming paths can deviate from the theoretical shortest path, when the number of fine-scale faces of S are limited due to insufficient samples. Even worse, in some (rare) case, such valid trimming path does not exist, unless the paths cutting through the fine-scale mesh faces are considered (which is not currently implemented). Such transforms are considered *degenerate*. In other words, if after all trials there is still no valid trimming paths found, a **brute-force** search can be performed, or the transform should be delayed or skipped, which occurs very rarely in practice.

[14] Here more complicated scenario can arise in two rare cases. *(i)* The trimming paths $\overline{\mathsf{mu}}$ and $\overline{\mathsf{mv}}$ can possibly overlap with the A_1^3 axial curve C. We avoid the two cases that the path $\overline{\mathsf{mu}}$ to overlap with $\overline{\mathsf{mD}}$ and the path $\overline{\mathsf{mv}}$ to overlap with $\overline{\mathsf{mE}}$, since both of them result in incomplete removal of the swallow-tail topology. *(ii)* The set of trimming faces $\{\mathsf{F_i}\}$ determined by this approach can include other unrelated shock elements (due to a non-local search on the $\mathcal{MS}$ sheet S) that invalidates this current transform. Although this is highly unlikely, a step to ensure that $\{\mathsf{F_i}\}$ is local to the A_5 transform is necessarily (details omitted).

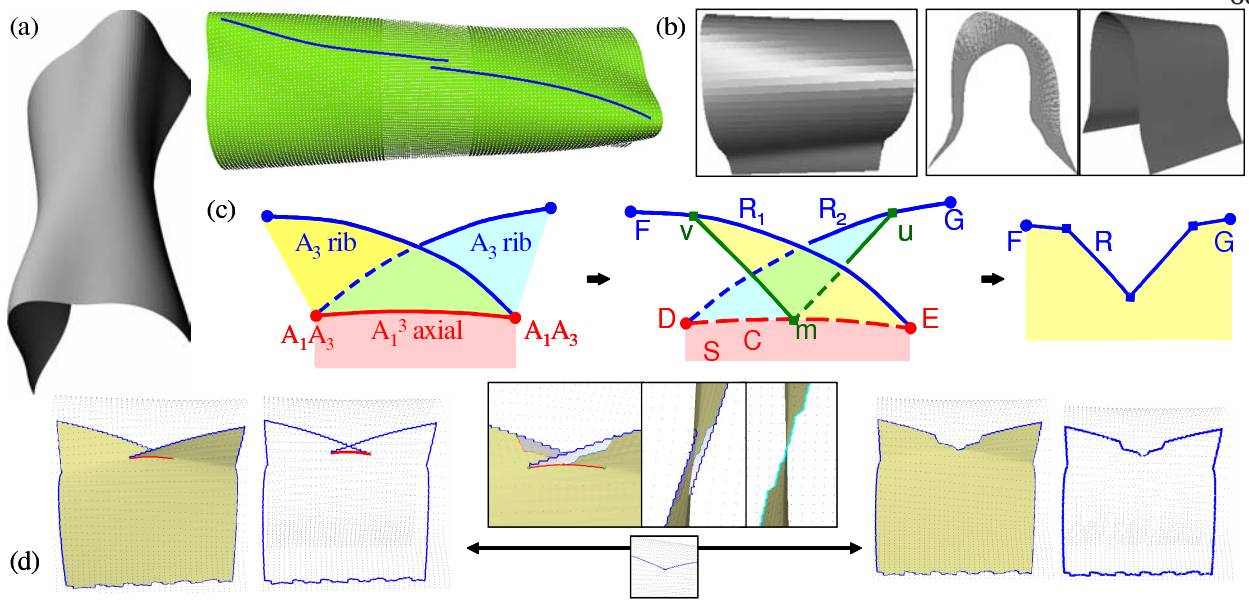


Figure 5.8: The A_5 curve contract transform. (a) A 'twisted parabolic gutter' shape produced from [88, Eq.2], a gutter with two long bumps corresponding to two ridges (blue) which gradually merge at the center. (b) show three views of the central portion whose $\mathcal{MS}$ is computed in (d). (c) In simulating the A_5 transition, where the swallow-tail structure gradually disappears, a trimming path u-m-v is identified (via approximated geodesic shortest path on the $\mathcal{MS}$ sheet) and used to remove the swallow-tail topology. The three colors (yellow, blue, red) depict three regions of a *single* $\mathcal{MS}$ sheet S around the swallow-tail. (d) The transform result, where a simple A_3 rib curve is recovered by removing all fine-scale $\mathcal{MS}$ elements $\{\mathsf{F_i}\}$ of the swallow-tail portion (in gray).

more on this issue.)

The last step of this transform is to delete the A_1^3 axial curve C and its ending A_1A_3 nodes D and E. Finally, the new A_3 rib from F to G can be traced element-by-element following the sequence F-v-m-u-G, as shown in Figure 5.8(c,d).

The corresponding shape changes of these two transforms are typically small and can be safely ignored. We observe in practice that the A_5 transitions do occur frequently, and the A_5 transforms simplify the topology near the $\mathcal{MS}$ boundary significantly; refer to Figure 7.11(f,g) in § 7.4.

$A_1^2A_3$-I curve contract transform

This transform contracts an A_1^3 curve to move an A_1A_3 fin point toward an A_1^3 axial, resulting in an $A_1^2A_3$ point, Figure 5.9. This transform can be done by finding a trimming path similar to the above A_5 transform, where only trimming is performed at the end of the tab T.

Detection / Cost of Transform: This transform is applicable to any A_1^3 axial curve C ending at an A_1A_3 fin point on one end and an A_1^n nodes $(n \geq 4)$ on the other end of C, where C is both incident to an A_1^2 shock tab T and doubly-incident to another A_1^2 shock sheet S at its internal anchor curve, Figure 5.9(b). Also the transform cost should be low. The transform cost is again estimated s the curve length of C. The corresponding shock transition is with low deformation cost: The length of A_1^3 shock curve C is shorter than a given threshold.

Operation of Transform: Refer to Figure 5.9, this transform can be done by first identifying a trimming point v on the A_3 rib curve (of the shock tab). *e.g.*, make the length of the curve $\overline{\mathsf{Ev}}$ along

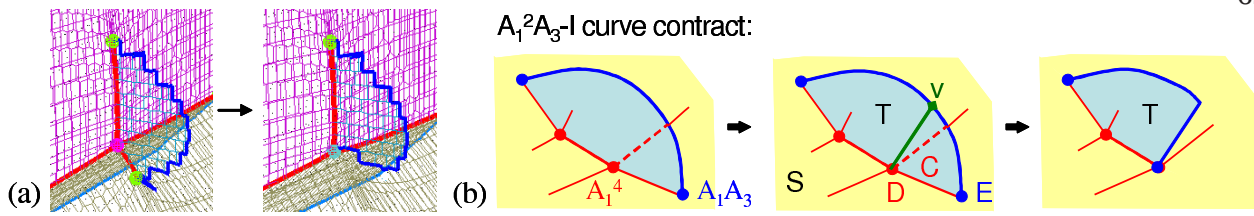


Figure 5.9: The $A_1^2A_3$-I curve contract transform: (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

the rib curve roughly equal to $\overline{\mathsf{DE}}$ (to fit the scale of local changes). Second, we use geodesic path to define a "trimming path" $\overline{\mathsf{Dv}}$ on the shock tab T, via Dijkstra's shortest path algorithm (to determine $\overline{\mathsf{Dv}}$ on the fine-scale mesh). Finally, we trim out the $\overline{\mathsf{DEv}}$ region of T and re-trace the A_3 rib as done previously in Figure 5.9. The corresponding shape change are typically small and can be safely ignored. [15]

* Summary of the above four "boundary" type $\mathcal{MS}$ transforms; comparison to remaining ones.

The *four* transforms introduced so far (namely, the two splice transforms, the A_5 and the $A_1^2A_3$-I contract) can be categorized as the **"boundary"** type of $\mathcal{MS}$ transforms, which simplify the *boundary* of the $\mathcal{MS}$ by removing medial sheets (by elements or by component) and can be performed *explicitly* on both scales of the $\mathcal{MS}$. In contrast, the remaining transforms are the **"interior"** type of transforms, which edit the *interior* of the $\mathcal{MS}$ and require a simulation of the edits on the coarse-scale hypergraph, while *keeping the fine-scale elements intact*. Their costs can be roughly estimated using the length of the curve (or area of the sheet) under contraction or merging. The modification of the corresponding shape changes for these transforms requires a further investigation and is omitted in this thesis.

We continue to describe the remaining transforms as follows.

A_1^5 curve contract transform

In the A_1^5 curve-contract transform, Figure 5.10, an A_1^3 axial C is contracted into an A_1^5 node N by keeping one of C's end point as N and repeatedly merging C with other A_1^3 curves on the three incident sheets (S_1, S_2, S_3).

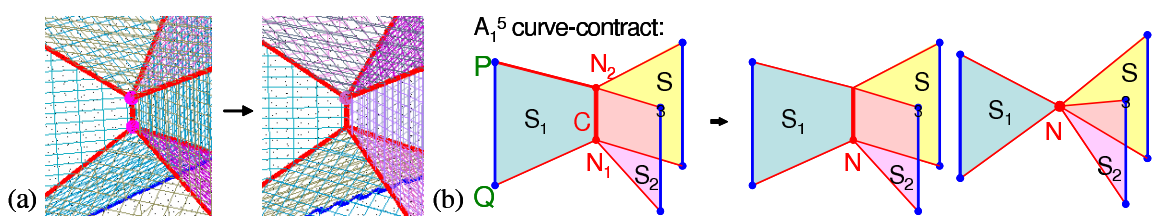


Figure 5.10: The A_1^5 curve contract transform: (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

[15] Degenerate Cases: A possible degenerate case is that the shortest trimming path $\overline{\mathsf{Dv}}$ intersects with the A_1^3 shock curve $\overline{\mathsf{DE}}$, which results in an incomplete transform. Our solution is to avoid such invalid scenario, by not allowing fine-scale edges and vertices of $\overline{\mathsf{DE}}$ included in the $\overline{\mathsf{Dv}}$ shortest path searching — an approach similar to that is used in the previous A_5 transform.

Detection / Cost of Transform: This transform is applicable to any A_1^3 shock curve C ending at two A_1^4 shock nodes, where C is incident to three distinct shock sheets S_1, S_2, S_3, where the merging node N is not an end point of any other A_1^3 axial curve C_i, which is not incident to S_1, S_2, or S_3. Also, the transform cost should be low. The transform cost is estimated as the length of the A_1^3 shock curve C.

Operation of Transform: Refer to Figure 5.10, for each incident shock sheet S_i of C, remove one A_1^4 shock node (say N_2) and merge the axial curve C with $\overline{PN_2}$ (to become $\overline{PN_1}$, and delete the shock node N_2. [16]

A_1^5 sheet contract transform

This transform is performed similarly as the previous one, that the sheet S is separately merged with two sheets S_1 and S_2 so that N is the final A_1^5 node, Figure 5.11. The A_1^3 axial curve $\overline{PQ}$ previously bordering S, S_1, S_2 is virtually removed (in the coarse-scale) [17]. The above 'merging' of sheets (namely, merging S to S_1 and merging S to S_2) involves *cloning* of fine-scale $\mathcal{MS}$ elements, which can be done by simply adding a pointer of the fine-scale object to the coarse-scale $\mathcal{MS}$ component. The two scales (coarse- and fine- scale)of $\mathcal{MS}$ representations may thus contain different (but each self-consistent) topology.

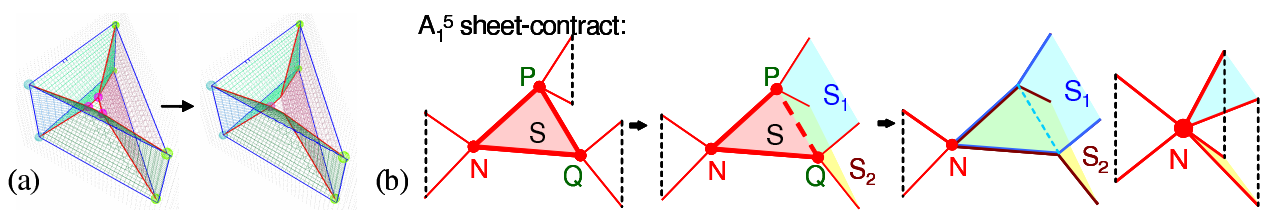


Figure 5.11: The A_1^5 sheet contract transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

Detection / Cost of Transform: This transform is applicable to any "interior" A_1^2 shock sheet element S (, *i.e.*, with no A_3 shock rib curve). Also, the transform cost should be low. The transform cost is estimated as the number of A_1^2 face elements.

Operation of Transform: Refer to Figure 5.11, pick one A_1^4 node N of the S to be the final A_1^5 node, and merge N across all shock curves of S not incident to N (in this example, only one such curve, $\overline{PQ}$) with the other shock sheets (in this example S_1 and S_2). Finally, merging the A_1^3 curves passing through P and Q to make consistent the $\mathcal{MS}$ hypergraph topology. At last, N should be an A_1^5 node with 6 incident curves and 9 sheets. [18]

[16] As mentioned before, no modification of the fine-scale shock element is required, since there is no easy modification other than re-computing the $\mathcal{MA}$. Our solution is to explicitly keep the difference between the coarse-scale and the fine-scale representations, where each of them is consistent on their own. In this example, N_2 is a point on the coarse-scale A_1^3 shock curve, and N_2's fine-scale element is an A_1^4, and we consider this to be valid.

[17] The A_1^5 node should have 6 incident curves and 9 sheets, the A_1^4 node should have 4 curves and 6 sheets, which are all correctly simulated.

[18] No modification of the fine-scale shock element is required. No easy modification or pruning can be done without re-computing shocks. Make both the coarse-scale and the fine-scale representation consistent at their own. Both P and Q are a fine-scale vertex on the coarse-scale A_1^3 shock curve. But their fine-scale elements are A_1^4's (which is considered valid).

A_1^4 sheet contract transform

This transform is performed similarly as the previous one, by merging S and S_2, Figure 5.12.

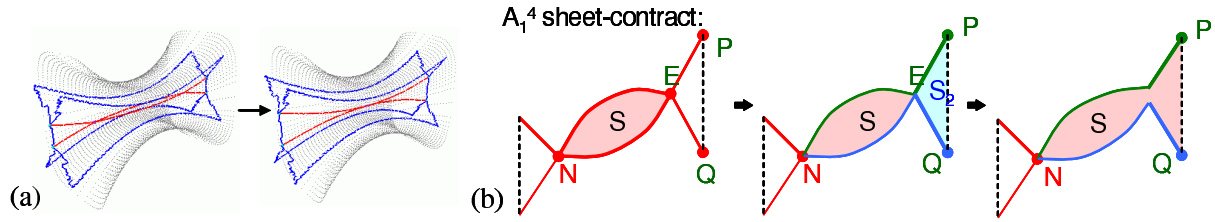


Figure 5.12: The A_1^4 sheet contract transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

Detection / Cost of Transform: This transform is applicable to any "interior" A_1^2 shock sheet S (, *i.e.*, with no incident A_3 shock rib curve), Figure 5.12. Also, the transform cost should be low. The transform cost is estimated as the number of A_1^2 face elements.

Operation of Transform: Refer to Figure 5.12, pick one A_1^4 shock node N of S to be the final A_1^4 node, and merge S across the opposite shock node E. Merge the A_1^3 curves passing to E to make consistent the $\mathcal{MS}$ hypergraph topology. N should be still of A_1^4 symmetry with 4 curves and 6 shock sheets.

Remark: The updating of boundary shape of the above three contract type of transforms (namely, A_1^5 curve contract, A_1^5 sheet contract, and A_1^4 sheet contract) is skipped, since this is an unsolved issue in 3D and is part of future work.

5.2.3 Merge Type Transforms

The **merge type transforms** merge medial curves or nodes together; they and are also performed in a simulated fashion on the coarse-scale $\mathcal{MS}$ hypergraph. Comparing to the splice-type and contract-type transforms, which *reduce* the $\mathcal{MS}$ hypergraph components, the merge-type transforms regularize the $\mathcal{MS}$ by *"adding"* hypergraph components (by splitting them). In addition, a *geodesic* distance field is required to detect candidate positions in the $\mathcal{MS}$ hypergraph for these transforms. Both the Dijkstra's algorithm [52] or the popular *fast marching method* (FMM) [117] can be applied to approximate such a geodesic distance field. Figure 5.13 show two examples in taking all vertex elements on the medial curves as *sources* and computing the distance field on all medial sheets using FMM. [19]

A_1A_3-II node-node merge transform

In this transform, Figure 5.14, two A_1^2 shock tabs T_1 and T_2 are merged at an A_1A_3 fin point N. This transform can be done by merging any two close-by T_1 and T_2 which are on a common "ground" sheet S, by joining two A_1A_3 fin points N_1 and N_2 (to become N) and make the two shock tabs connected. The cost of this transform is defined as the geodesic distance d between N_1 and N_2.

[19] We note that the ideal approach to search for candidate nodes/curves for the merge transform is to consider more factors other than simply the shortest path. For example, the tangent direction at the end of the $\mathcal{MS}$ curve should be consistent with a candidate merging path. Refer to Figure 5.18 for an example. We leave this as future works.

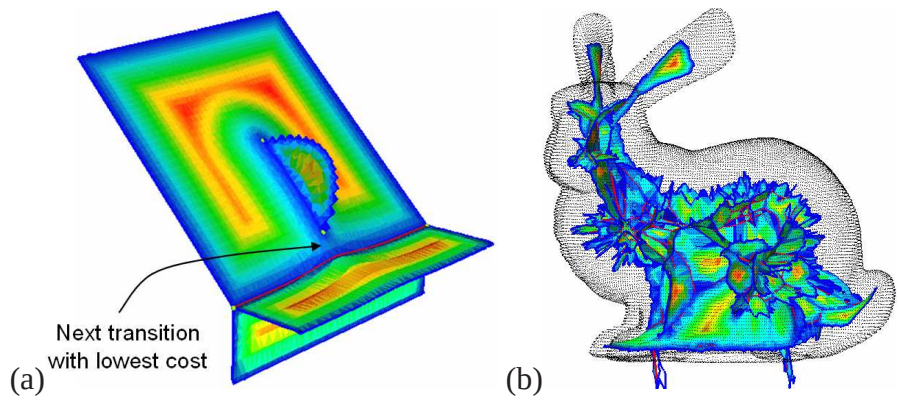


Figure 5.13: The use of geodesic distance transform on the medial sheets to detect the next possible transition with the lowest cost [43]: (a) The distance map of the $\mathcal{MS}$ near an $A_1^2A_3$-I transition taking all vertex elements on the sheet boundary as *sources* (blue). (b) Result of the distance map on the $\mathcal{MS}$ of the Stanford bunny.

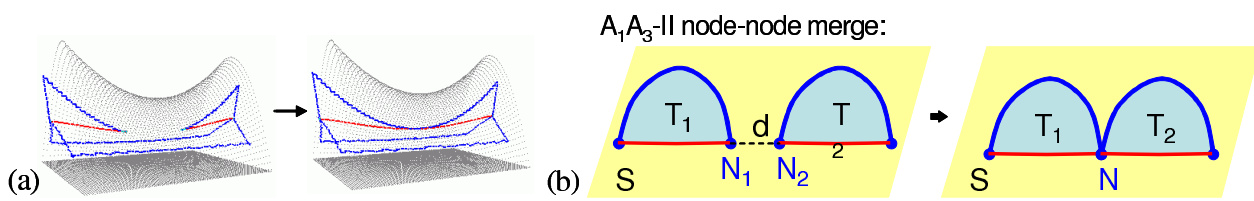


Figure 5.14: The A_1A_3-II node-node merge transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

$A_1^2A_3$-I node-curve merge transform

In this transform, Figure 5.15, an A_1A_3 fin point N of an A_1^2 tab T is merged with an A_1^3 curve C to create the $A_1^2A_3$ local form and split C into two curves C_1 and C_2. This transform can be done by merging N of shock tab T on a "ground" sheet S with a close-by A_1^3 shock curve C on S, where C can be either the *boundary* or *interior* curve of S (Recall the notion of a generic medial sheet topology in Chapter 4). This transform join N to interior of C and break C into two curves C_1 and C_2. The cost of this transform is defined as the geodesic distance d between N and C.

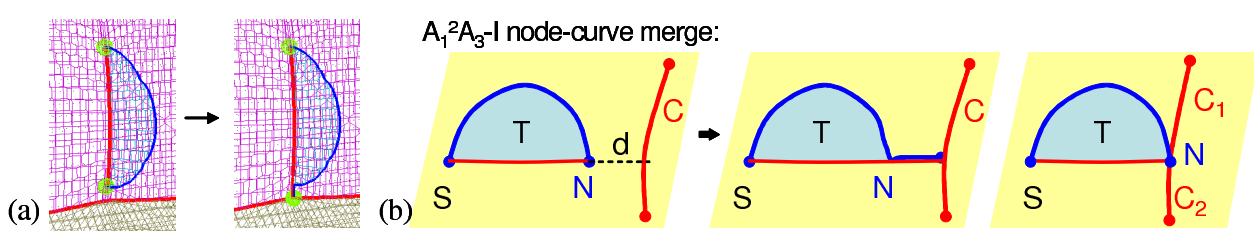


Figure 5.15: The $A_1^2A_3$-I node-curve merge transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

A_1^4 curve-curve merge transform

In this transform, Figure 5.16, two close-by A_1^3 axial curves C_1 and C_2 sharing a single A_1^2 sheet S are brought together to create a new A_1^4 node N. The merging nodes N_1 on C_1 and N_2 on C_2 are

detected by finding the closest points on C_1 and C_2, again using the geodesic distance field on S. Note that N_1 and N_2 can not be then end point of C_1 and C_2, otherwise the local form is not A_1^4 and will result in another type of transform. [20] The cost of this transform is estimated as the geodesic distance d between the N_1 and N_2.

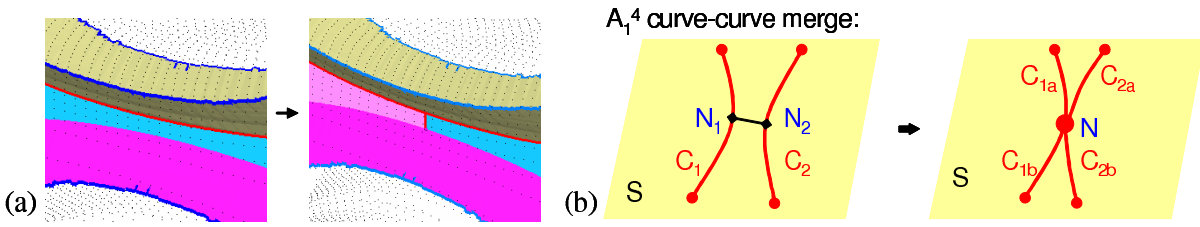


Figure 5.16: The A_1^4 curve-curve merge transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

A_1A_3-II curve-curve merge transform

This transform is very similar to the above one in structure, while their local form are different. In this transform, Figure 5.17, an A_3 rib curve R approaches an A_1^3 axial curve C and merge to it to create an A_1A_3 fin point N. The merging nodes N_1 in R and N_2 in C are again detected via the geodesic distance field on T. Similarly, N_1 and N_2 can not be the end points of R and C, otherwise the local form is not A_1A_3 and results in another type of transform. The cost of this transform is estimated as the geodesic distance d between R and C.

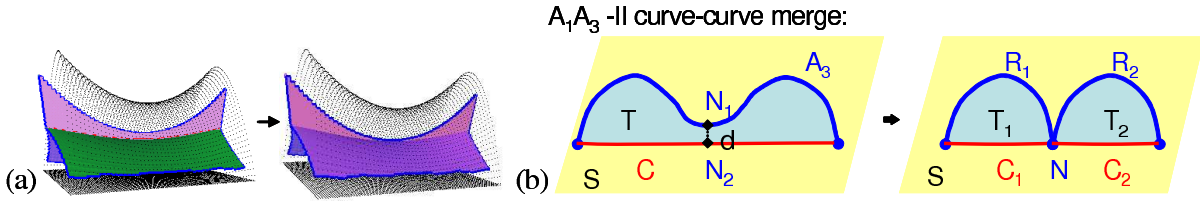


Figure 5.17: The A_1A_3-II curve-curve merge transform. (a) a simulated example. (b) edit operations on the $\mathcal{MS}$ hypergraph.

Figure 5.18 shows some example merge transform candidate paths detected by our implementation of the (discrete) geodesic path searching on the $\mathcal{MS}$ hypergraph.

5.3 $\mathcal{MS}$ Transforms for Non-Closed Shapes

The above 11 transforms completely cover all 7 generic transitions in [88] for *simple closed* shapes. Our framework is augmented with two additional types of transforms, namely, the **gap** and **loop** transforms to handle *non-closed* shapes (in particular the *sampled* shapes. The *gap* transforms have been define in [45] close gaps on the shape surface, while the loop transform has been used in 2D [109] to remove noisy sample points.

[20] In practice we prevent N_1 and N_2 to be close to the end points of C_1 and C_2 by enforcing a threshold parameter to avoid detecting N_1/N_2 as the immediate neighbors of the end points of C_1/C_2.

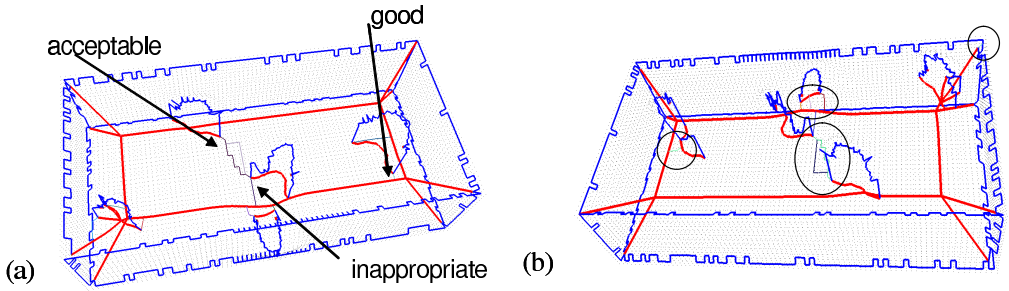


Figure 5.18: Some example merge transform candidates detected using the geodesic path searching on a box with several artificial bumps (which create shock tabs). Candidate merge paths are shown in random colors. In (a), three example of a good, an acceptable, and inappropriate merge paths are labeled. In (b), more examples are highlighted in circles.

5.3.1 Gap Type Transforms

Recall that our approach handles shapes given as unorganized sample points, *i.e.*, the "only" available information is that these points are close to a possible shape surface. An intuitive idea is to group together closely sample points and start to form curves and surfaces to reconstruct the underlying shape. The $\mathcal{MA}$ branches between point samples in this case provide a good heuristic on choosing a good close-by sample points, and this is precisely the motivation of the gap type transforms.

The **gap type transforms** remove a set of medial branches and fill the "gaps" between the (boundary) shape surfaces. We define two types of gap transforms based on whether a medial sheet or curve is removed:

- An A_1^2**-sheet gap transform** closes the gap between a pair of boundary sample points by inserting a line segment and remove the A_1^2 shock sheet between the pair of points.
- AN A_1^3**-curve gap transform** closes the gap between a triplet of boundary sample points by inserting a triangle and remove the A_1^3 shock curve between the triplet of points. (Refer to Figure 6.4 for an illustration.)

The gap transforms defined on the point-based shapes are closely related to the *Voronoi-filtering* methods used in the domain of surface reconstruction surveyed in Chapter 2. The idea is that the geometry of the $\mathcal{MA}$ coincides with the Voronoi diagram ($\mathcal{VD}$) in this case: the A_1^2 medial sheets are essentially the Voronoi faces, and the A_1^3 medial curves are the Voronoi edges (§ 3.3). It's well-known that the $\mathcal{VD}$ and the Delaunay triangulation ($\mathcal{DT}$) are dual structures [52]. The above A_1^3-curve gap transform essentially replace an Voronoi edge with a Delaunay triangle.

We will elaborate the use of the gap transforms to mesh a surface from input points in Chapter 6. This meshing process is in fact part of our $\mathcal{MS}$ computational scheme, which will be detailed in Chapter 7.

On final remark is in addition to define the gap transform on the fine-scale $\mathcal{MS}$ to create an initial mesh (as described above), we can also perform the gap transforms on the coarse-scale $\mathcal{MS}$ hypergraph, to handle a larger (true) holes of the object (with larger scale *w.r.t.* the sampling), once the regularized $\mathcal{MS}$ as a qualitative structure of the shape is available. For example, refer to Figure 6.17(b) for such an example.

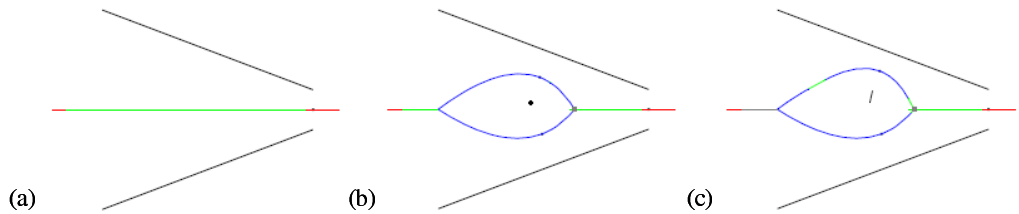


Figure 5.19: (Adapted from [109]) The introducing of a sample point between two boundary curves creates a loop of the $\mathcal{MA}$, which can be removed by a gap transform.

5.3.2 Loop Transform

A **loop transform** remove a boundary sample point (or a shape patch) and remove its corresponding $\mathcal{MA}$/shocks that form a loop enclosing the point. Figure 5.19 [109] illustrates that the effect of adding an improper sample point to the interior of the shape *drastically* changes its $\mathcal{MA}$. The introducing of such interior point can also be viewed as a transition (in that the shapes before/after adding this point is along a one-parameter family of deformation) illustrated in Figure 5.19(a-b), and the loop transform is defined to remove such instability.

In comparison to other transforms which can be *approximated* or *simulated* on the $\mathcal{MS}$, the loop transform is not. In other words, the only way to perform a loop transform is to re-compute the $\mathcal{MS}$ locally or globally. For this reason, we do not integrate the loop transform into our computational framework and we only perform the loop transform in the pre-processing stage to remove noisy sample points far form the shape boundary.

5.4 Non-Generic Higher-Order $\mathcal{MS}$ Transforms

This section describes two additional transforms we found useful which corresponds to *non-generic* transitions frequently observed in practice.

* Non-generic 3D $\mathcal{MA}$ transitions actually observed in practice.

We list *five* types of non-generic $\mathcal{MA}$ transitions observed in practice as we apply the set of transforms (described so far) in iterations: [21]

1. *Collision of an A_1^4 to an A_1^5 (or A_1^n, $n > 5$).* Refer to Figure 5.21(j). This is essentially the A_1^5 *curve contract transform* in the high order configuration. [22]
2. *Collision of an A_1^4 to an $A_1^2 A_3$ (or $A_1^m A_3$, $m > 1$).* Refer to Figure 5.21(k). This is similar to the above case and can be handled by the A_1^5 *curve contract transform* in the high order configuration near an A_3 rib.
3. *Collision of three (or more) $A_1 A_3$ fin points.* Refer to Figure 4.6. This is a rarely observed but possible in practice, which is a special case of the $A_1 A_3$-*II node-node merge transform*.
4. *Collision of an $A_1 A_3$ fin point and an A_3 rib, where the local form is not the aforementioned A_5 transitional collision.* This is an interesting transition only observed around a corner, Figure 5.20(a), detailed in § 5.4.1.

[21] The author gratefully acknowledges helpful conversations with Prof. Peter J. Giblin, Dept. of Math. Science, Univ. of Liverpool, UK.

[22] We found the A_1^5 curve contract transform particularly important in decomposing the high order $\mathcal{MS}$ nodes, detailed in § 5.5.

5. *Collision of an A_1A_3 fin point to an A_1^4 (or A_1^n, $n > 4$).* This transition is similar to the $A_1^2A_3$-I node-curve merge, but instead of merging a fin point N with an A_1^3 curve C, N is merged to the end point of the curve C.

Remarks: Out of the above *five* degenerate transitions, three of them can be handled with the existing $\mathcal{MS}$ transform (generalized to handle additional hypergraph topology in a higher order). We will need to define *two* additional transforms, described below. An theoretical question not answered so far is that *"Is the above cases cover the complete set of degenerate configurations?"*. In particular, from the theoretical work [88], we expect to see many degeneracies, which is not observed in our point-based $\mathcal{MS}$ computation system. This is an ongoing work we are currently investigating (a joint work with Peter Giblin).

5.4.1 Corner-merge transform

This transform is defined to handle the collision of an $A_1A - 3$ fin point N with an A_3 rib on an A_1^2 sheet S, which we found useful in regularizing the $\mathcal{MS}$ around a corner, Figure 5.20(a-b). This regularization of $\mathcal{MS}$ around a corner is motivated in Figure 5.20(c), which is important in modeling and matching highly regularized shapes (such as man-made objects and models often used in computer graphics).

Transition of $\mathcal{MA}$ around a corner. Despite it usefulness, this non-generic **"corner"** transition is not fully understood. Specifically, the question is what is the generic configuration as we start to perturb a perfect corner as shown in Figure 5.20(a), where one A_1^3 axial meets at three A_3 ribs (the intersection of three ridges at the surface)? We suspect that a fin point N can *only* approach an A_3 rib C when C is curved and essentially forming a corner. This is an intricate problem we are currently investigating with Peter Giblin. See § 10.3.1 for a simulation of such analysis by deforming a perfect smooth corner.

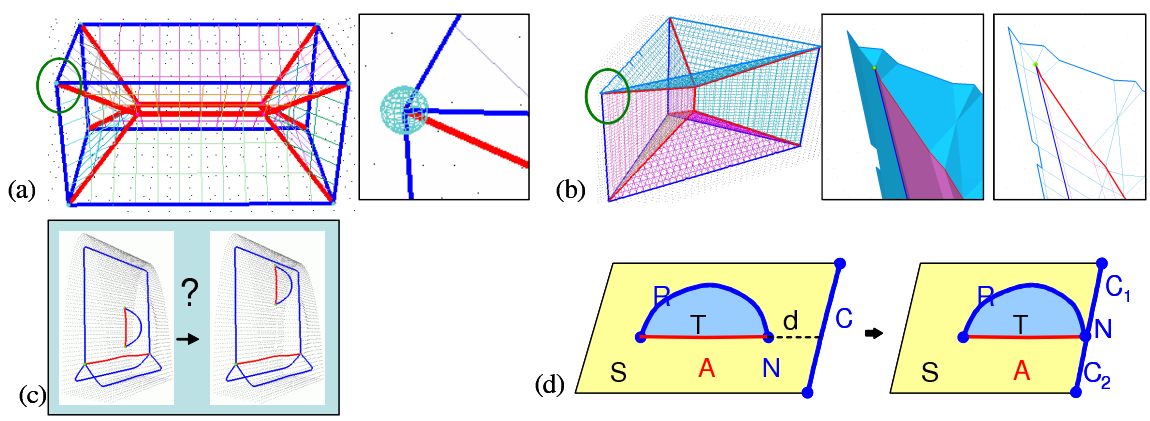


Figure 5.20: The corner node-curve merge transform. (a) The $\mathcal{MS}$ of a perfect box and a zoom-in at the corner, where three A_3 ribs intersects with one A_1^3 axial at the corner. (b) Zoom in to the corner of $\mathcal{MA}$ of a prism shape. Due to slight perturbation and sampling variations, two of the ribs group into one and the other rib and axial intersects at an A_1A_3 (fin) point. (c) Another situation is to move a shock tab toward a rib, while moving toward an axial (down) is an $A_1^2A_3$-I transition. (d) The corner node-curve merge transform regularizes by merging the A_1A_3 fin point toward the A_3 rib.

The above difficulty does not hamper the regularization of the $\mathcal{MA}$ around a corner. We define the **corner-merge transform** as follows. This transform can be detected by using the geodesic field

on S searching for the shortest merging path between N and C, Figure 5.20(a-b). The operation is simply joining N with C in the interior and break C into two curves C_1 and C_2. The transform coast is estimated as the geodesic distance d between the N and C.

5.4.2 Higher-order merge transforms

We define the only transform we found necessary to handle the transition of collision of an A_1A_3 fin point N to an A_1^4 node.

$A_1^3A_3$-I node-node merge transform

This transform is similar to the $A_1^2A_3$ node-curve merge transform and can be performed in mimicking the node-node merge transforms described before.

5.5 Analysis of High-Order $\mathcal{MS}$ Nodes and Curves

We propose to make equivalent a degenerate high-order $\mathcal{MS}$ node to a group of generic low-order nodes by decomposing them analytically. This analysis is useful in two aspects: *(i)* First, it is useful in providing a consistent compatibility analysis in matching the $\mathcal{MS}$, a topic to be explored more in Chapter 9. *(ii)* Second, this organization of high-order $\mathcal{MS}$ nodes also complements Giblin and Kimia's original work on the *generic* cases of $\mathcal{MS}$ representation and extends it to cover all cases (include the degeneracies).

* The general form of high-order $\mathcal{MS}$ nodes: $A_k^m...A_l^n$.

In general a high-order $\mathcal{MS}$ node is in the form of $A_k^m...A_l^n$, where $k, ..., l$ are odd integers > 0, $m, ..., n$ are integers > 0. The numbers $(k, m, ..., l, n)$ can be completely determined from the local configuration, (*i.e.*, via counting incident curves and sheets, Figure 5.21.

* Simplifying the general $A_k^m...A_l^n$ nodes into two types: the $A_1^mA_3$ and A_1^n.

In order to simplify the above generally, we make an important *assumption* that all $\mathcal{MS}$ ribs are of type A_3. The reason is that *(i)* for point-sampled shapes, rib curves higher than A_5 is unnecessary (the A_5 is by definition isolated, thus does not form a curve). Also, *(ii)* there is no significant modification of the $\mathcal{MS}$ in reducing the degeneracy this way when the shape is given in point samples. As a result, the above assumption avoids all A_k ($k{\geq}5$) degenerate ribs, which leads to significant *simplification* to decompose a $A_k^m...A_l^n$ node into two general forms:

- the *boundary* $A_1^mA_3$ node with one or more incident A_3 rib(s), $m \geq 2$, and
- the *interior* A_1^n node with no incident A_3 rib, $n{\geq}5$. [23]

The two general forms only involves axial type changes and thus can be decomposed by expanding A_1^3 axials via an *inverse* A_1^5 **curve contract transform** mentioned before. Figure 5.21 depicts some typical high-order nodes and their decomposition. Table 5.2 summarizes the equivalent incident generic curves/sheets.

* Representing the local configuration of a $\mathcal{MS}$ node.

[23] For $m = 1$, the $A_1^mA_3$ = A_1A_3 is a generic $\mathcal{MS}$ node; for $n = 4$, A_1^n = A_1^4 is a generic $\mathcal{MS}$ node.

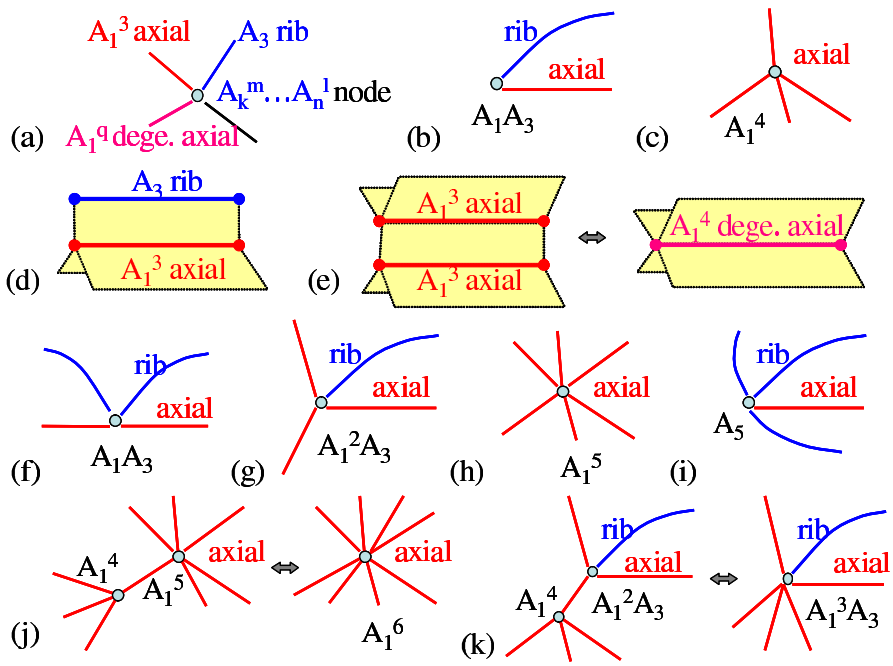


Figure 5.21: Local configurations of the $\mathcal{MS}$ nodes and their incident A_3 ribs and A_1^3 axials. (a) The general form of an $A_k^m ... A_n^l$ node. (b-c) The generic A_1A_3 and A_1^4 nodes. (d) The incident sheet(s) of a generic A_3 and A_1^3 curves. (e) Merging two A_1^3 axials into an equivalent degenerate A_1^4 axial. (f) The merging of two A_1A_3 nodes at an A_1A_3-II transition. (g-h) The high-order $A_1^2A_3$ and A_1^5 nodes arising immediately from the $\mathcal{MS}$ transforms. (i) The A_1A_5 node arising from a degenerate A_1A_5 curve-merge transform at a corner. (j-k) The two types of A_1^n and $A_1^mA_3$ high-order nodes.

We define the local configuration of a high-order $\mathcal{MS}$ node by counting its incident $\mathcal{MS}$ curves as follows. The configuration of a $\mathcal{MS}$ node can be described as a tuple $(r, a, \{d^q\})$, where r and a is the number of incident A_3 rib and A_1^3 axial curves, d^q is interpreted that the node contains d incident degenerate A_1^q curves, $q > 3$.

* Determine the order of $A_1^mA_3$ or A_1^n nodes by analyzing the local incident curves.

Below we describe an analysis to determine the type of an high-order $\mathcal{MS}$ node (*i.e.*, determining m or n), by counting its incident $\mathcal{MS}$ curves (*i.e.*, from the given tuple (r, a, d^q)). Before we start, we note that we have not mentioned an additional degeneracy of collision of multiple A_1A_3 fin points (see § 5.4) that should be integrated into the analysis. We denote the number of collision of fin points as the number f for an $A_1^mA_3$ node. Below is our approach to find the tuple (m, n, f) from the local incident curves (r, a, d^q).

First, we note that an A_1^q degenerate axial curve ($q > 3$ can be decomposed into equivalent A_1^3 axials, see Figure 5.21(e). We convert d^q and denote the number of equivalent A_1^3 axials as a', which is determined as: [24]

$$a' = a + d(q - 2). \tag{5.1}$$

We essentially converted the input tuple (r, a, dq) into $(r, a', 0)$ in the above step. The next is to determine (m, n, f) from r and a', which can be done by induction on the two local configurations in Figure 5.21(j) for $A_1^mA_3$ and Figure 5.21(k) for A_1^n. We summarize the results for $\mathcal{MS}$ nodes

[24] For multiple degeneracy's, $a' = \sum_i q_i(d_i - 2)$, for $\sum_i d_i = d$.

Table 5.2: Canonical from of local configuration of generic high-order $\mathcal{MS}$ nodes in terms of its incident $\mathcal{MS}$ curves and sheets. Integer numbers r, a', s denote the number of incident A_3 rib curves, A_1^3 axial curves, and A_1^2 sheets, respectively. (Left): the *'boundary'* $A_1^m A_3$ nodes. (Right): the *'interior'* A_1^n $\mathcal{MS}$ nodes.

Type	r	a'	s
A_1A_3	1	1	2
$A_1^2A_3$	1	3	5
$A_1^3A_3$	1	5	8
$A_1^4A_3$	1	7	11
$A_1^5A_3$	1	9	14
...	...	...	...
$A_1^mA_3$	1	$2m$-1	$3m$-1

Type	r	a'	s
A_1^4	0	4	6
A_1^5	1	6	9
A_1^6	1	8	12
A_1^7	1	10	15
A_1^8	1	12	18
...	...	...	...
A_1^n	1	$2(n$-2)	$3(n$-2)

frequently observed in our experiments in Table 5.2. The tuple (m, n, f) is then derived as follows:

$$\begin{cases} \text{- if } r = 0 : m = 0, n = a'/2 + 2, f = 0. \\ \text{- else if } r = 1 : m = (a' + 1)/2, n = 0, f = 1. \\ \text{- else if } r = 3 \text{ and } a' = 1 : \text{degenerate case of } A_5, m = 0, n = 0, f = 2. \\ \text{- else : special case of } (A_1A_3)_f, \text{ where } f = r. \end{cases} \tag{5.2}$$

Finally, the degeneracies of collision of multiple A_1A_3 fin points is organized in Table 5.3.

Table 5.3: Additional configuration of the *degenerate* high-order $\mathcal{MS}$ nodes.

Type	r	a'	s
A_1A_3	1	1	2
$(A_1A_3)_2$	2	2	4
$(A_1A_3)_3$	3	3	6
...	...	...	...
$(A_1A_3)_f$	f	f	$2f$

Type	r	a'	s
A_5	3	1	3
...	...	...	...

Chapter 6

From Point Clouds to Surfaces: On Using the Medial Scaffold for Surface Meshing

* Overview of chapter: segregate the $\mathcal{MS}$ of point clouds to mesh the shape surface.

This chapter address the computational aspect of the $\mathcal{MS}$ from a sample dataset: a *segregation* process of the $\mathcal{MS}$ is elaborated in interpolating point samples, *i.e.*, surface mesh reconstruction together with retrieving an initial $\mathcal{MS}$ which approximates the true $\mathcal{MA}$ of the sampled shape. This chapter is an updated view and extension of the work initiated by Leymarie [128, Ch.6], a version of which was published at 3DIM'07 [45].

Specifically, the *gap* transform described in the precious chapter is exploited here to create surface interpolants. The meshing of surface from sample points is an essential step in the $\mathcal{MS}$ computation and regularization process, since working directly on the *unorganized* points is generally difficult, due to the lack of a notion of neighborhood around the shape. The goal of the $\mathcal{MS}$ *segregation* is two fold. *(i)* First, the explicit recovery of surface interpolants (meshing) provides the *connectivity* between the sample points to reconstruct the (unknown) shape. *(ii)* Second, an initial $\mathcal{MS}$ of the input shape, which is dual to this reconstructed surface, is also constructed. The computational scheme to further process such dual $\mathcal{MS}$ and shape will be elaborated in Chapter 7.

* Organization of chapter.

This chapter focuses on the theory and computational strategy on how to consider gap transforms to segregate the $\mathcal{MS}$ in order to obtain a reconstruct surface mesh and the initial $\mathcal{MS}$. It is organized as follows. Section 6.1 overviews the philosophy behind our surface reconstruction approach and motivates the use of the gap transform. Section 6.2 describes the proposed surface reconstruction algorithm composed of greedy iterative gap transforms. Section 6.3 validates our method with experimental results. An important point not raised thus far is that since we have not imposed any strong restriction on the input surface sampling and topology, we can handle very large datasets by simply dividing the space into 3D buckets and meshing surface within each bucket and fuse them together. This is detailed in Section 6.4.

6.1 Philosophy of the Proposed Surface Reconstruction Approach

* Recover surface mesh from a "minimum" set of assumptions: point samples are nearby a shape.

We consider the problem of constructing surface meshes from a sampling in the form of an *unorganized cloud of points*; that is, no assumptions (on sampling, density, normals) are needed to process

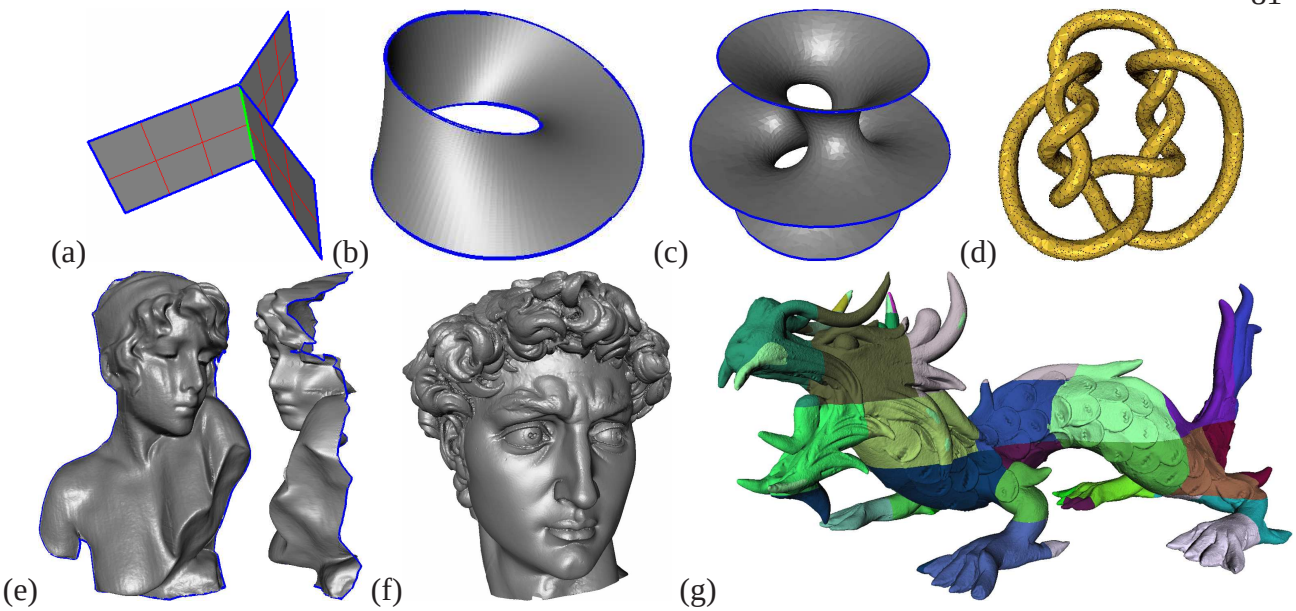


Figure 6.1: (From [45].) Surface reconstruction from *unorganized* point clouds using $\mathcal{MS}$ transforms of several topology types [45]: (a) non-manifold surface with self-intersections, (b) Möbius strip (non-orientable), (c) Costa's minimal surface with the topology of a torus thrice punctured, (d) connected water-tight knot in highly non-uniform sampling of 10K points, (e) a complex surface with recovered boundary (in blue): the Sapho raw scan dataset (121K points, Stony Brook), (f) Michelangelo's David head (250K points, Stanford), (g) the combination of several reconstruction results (in various colors) from a 3D bucketing and stitching of the Stanford Asian Dragon dataset ($\approx 3.6 \times 10^6$ points).

a raw data. The recovery of this connectivity amongst points is typically based on assuming *(i)* some surface continuity, possible smoothness, and *(ii)* sufficient sampling density to capture all surface features. The reason for assuming the significant case of the *unorganized* points is that: *(i)* it is the native type of free-form data acquired with modern scanning devices, such as laser or computerized topography (CT) scanners; *(ii)* in many cases, the local connectivity between sample points is not available, or it can be unreliable when the sampling condition is not known; and furthermore, *(iii)* it allows to process data "on-the-fly" as new sample points can be immediately processed by updating the local neighboring structure.

* Handle *"generic"* surface topologies.

Our approach is meant to be general and deal with generic topological configurations. It is applicable to surface of various topologies (Figure 6.2); in particular it does not assume that the surfaces are smooth, nor closed (enclosing a volume), nor orientable, it can handle *non-manifold* surfaces (a case not considered in most main-stream methods), it does not require uniform sampling, it can handle varying noise levels, and it is scalable to arbitrarily large input datasets (Figure 6.1), as discussed below.

* Approach: shape deformation and gap transforms to *recover* the sampling process.

Our approach is based on the notion of representing shape via deformations and their local successive topological variations. It is directly inspired from previous work by Kimia *et al.* in 2D and 3D [116, 169, 87, 88], summarized in Chapter 1 to represent shape as a member of an equivalence class,

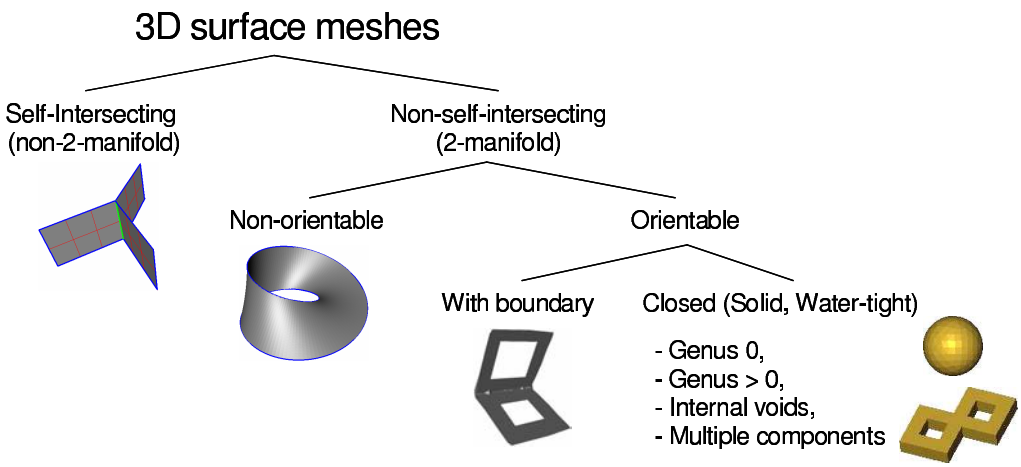


Figure 6.2: (From [45].) Classification of surfaces in 3D based on the topology.

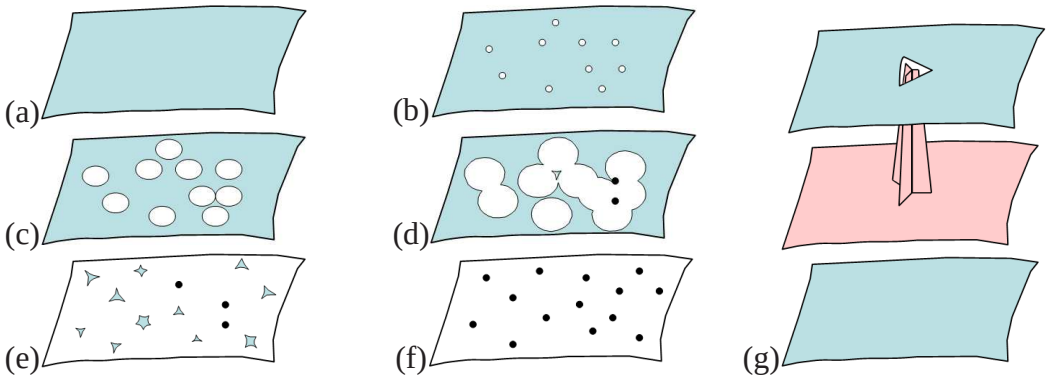


Figure 6.3: (From [45].) This figure illustrates that "sampling" a surface to produce an unorganized cloud of points can be viewed as a *deformation process*: (a) the original surface; (b) some points are taken away from the surface, creating holes; (c) these holes grow thus reducing the area of the remaining surface patch, and (d) eventually start joining to form larger gaps and isolated patches; (e) the remaining surface patches each tend to shrink to a point; (f) the resulting point cloud represents samples of the surface. (g) An infinitesimal but generic hole creates three A_1^2 sheets intersecting at A_1^3 curve.

each defined as the set of shapes sharing a common $\mathcal{MS}$ topology, and the notion of $\mathcal{MS}$ transitions and transforms defined in Chapter 5.

The philosophy behind our surface meshing approach can be fully embedded in the above context: *complete surfaces and sampled surfaces are simply two points in the shape space that are closely related*. In an approach to 2D perceptual grouping [200], a greedy approach finds more regular shapes from unorganized data. Similarly, in our case in 3D, a sampled shape is simply a deformation of the original surface, as illustrated in Figure 6.3. In this process, the $\mathcal{MS}$ undergoes abrupt *transitions* (topological events).

Ideally, all sequences of transitions should be considered, *i.e.*, the ones described in Chapter 5. In this chapter, however, we restrict ourselves only to the transition arising from removing a patch from the surface (the first step in our view of the sampling process) and consider only the symmetry transform pertaining to that, *i.e.*, the **gap transform** defined in § 5.3.1. Specifically, removing a surface patch generically results in adding to the $\mathcal{MS}$ a group of three $\mathcal{MA}$ sheets and one $\mathcal{MA}$ curve at their intersection. Thus, the inverse process of reconstructing a surface patch corresponds

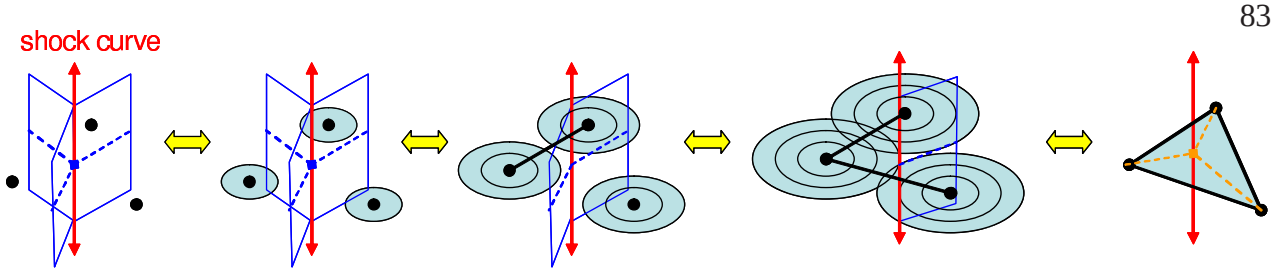


Figure 6.4: This figure illustrates how a $\mathcal{MS}$ curve (with three sheets) is transformed into a surface triangle in a deformation process simulated as a gap transform.

to consider the removal of such curves and their attached sheets (delimited by loops) from the $\mathcal{MS}$, Figure 6.4.

As in perceptual grouping in 2D [200], the gap transform in 3D is considered in a *greedy* sequence, proceeding first with the least cost transform. This cost is based on the likelihood that a $\mathcal{MS}$ curve represents a 'missing' surface patch, as well as on the local context (*i.e.*, other reconstructed surface patches which may already exist in the vicinity). Figure 6.5 illustrates this process, which can be viewed as a **segregation process** (Figure 6.5(a)) of the 3D shocks on a pair of planar sheets, one of which is somewhat protruding (Figure 6.5(b)). The $\mathcal{MS}$ (Figure 6.5(c)), after a sequence of gap transforms, then resembles that of the original two complete un-sampled surfaces (Figure 6.5(d)). This in turn enforces a connectivity of the surface samples thus creating a surface mesh together with an approximated $\mathcal{MA}$ (Figure 6.5(e)).

Since points are meant to represent a surface patch whose *orientation* is not *a priori* clear, we represent all possible such patches by using an infinitesimal small sphere around each point and call it a point *generator*, denoted by $G_i(x_i, y_i, z_i)$, where x_i, y_i and z_i are the coordinates of the given point also representing the center of the tiny sphere.

6.2 The Shock Segregation (Meshing) Algorithm in Applying the Gap Transforms

We now present the details of our method, once the $\mathcal{MS}$ is computed by the method of Leymarie and Kimia [125].

* Overview of algorithm: greedy iteration with strategies to present "local minimum".

We propose to reconstruct a surface mesh from unorganized point clouds with minimalistic assumptions, by considering the *gap* transforms on all A_1^3 shock curves in a rank-ordered greedy approach.The greedy nature of the algorithm, in contrast to the ideal but impractical algorithm considering all sequences of gap transforms on the $\mathcal{MS}$, implies that local decisions in ambiguous cases may lead to erroneous results. Since the ranking is partially decided on the basis of a local surface neighborhood, these errors can then potentially propagate. To prevent the negative effects of such "local minima", we adopt a three-fold strategy. *(i)* The set of A_1^3 curves is divided into two distinct categories, based on whether the gap transform decision is categorically easy or difficult, leading to two passes of greedy iterations. The first pass aims at constructing valid surface triangles (*i.e.*, without ambiguity about their shape or local topology) in a greedy iterative sequence. Given a

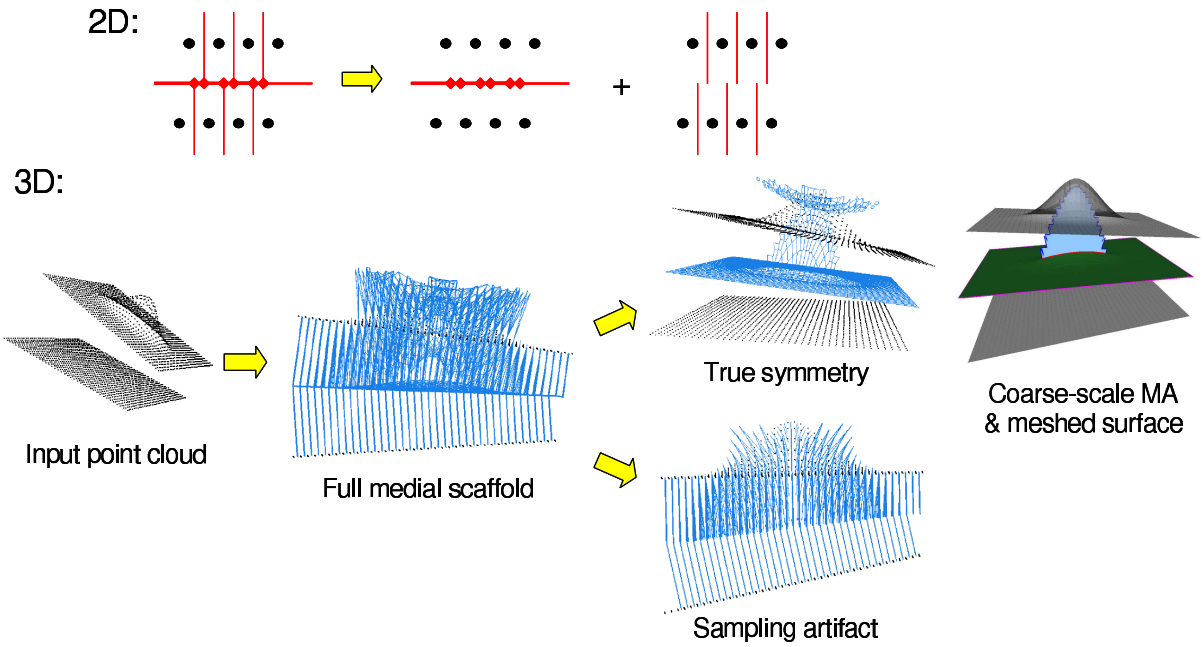


Figure 6.5: (Adapted from [128, Fig.6.6].) Illustration of the *shock segregation* process in 2D (upper) and in 3D. The input is a set of 3,200 points uniformly sampling a pair of planes, one of which is deformed by an elongated Gaussian kernel. At a first step, the full $\mathcal{MS}$ is computed and the side of is shown. The remaining $\mathcal{MS}$ after undergoing the segregation into a set pertinent to the "true symmetry" and the other set belongs to "sampling artifacts" via a series of *gap transforms*. The results are two-fold: *(i)* the reconstructed surface and *(ii)* its corresponding $\mathcal{MS}$ organized into a hypergraph form.

well-sampled input (*i.e.*, the *intra*-sampling distance is smaller than the *inter*-sampling distance, Figure 6.6), [1] most surface triangles can be successfully built in this step. All remaining cases are then handled in the second pass, where the surface built in the first pass can be used to help resolve the initially ambiguous situations as discussed below. *(ii)* Since applying a gap transform can invalidate another candidate gap transform, in situations when a pair of gap transforms are close in ranking, *i.e.*, the difference in costs $\delta\rho$ is smaller than a threshold Δ_ρ (where "ρ" denotes a cost function), their ranking is reduced by increasing their cost by a factor ζ, so as to delay a decision until additional local context is available.[2] *(iii)* An *error recovery* option is available: after each gap transform, costs associated to neighbor surface interpolants (including already selected and candidate triangles) are re-evaluated: if the newly computed cost of a previously accomplished gap transform exceeds our top candidate's cost, we "undo" it by putting back the associated A_1^3 curve into a (priority) queue. This essentially increases the robustness of the algorithm in recovering from errors, which will be addressed in more details later.

* Initialization of the shock segregation algorithm.

Initialization: The surface reconstruction is initialized by sorting all A_1^3 curves into the two priority queues, $\mathbf{Q_1}$ and $\mathbf{Q_2}$ (for the 1st and 2nd greedy iterations). Each A_1^3 curve identifies three sample

[1] We consider the theoretical sampling issues for our method elsewhere; the focus here is on introducing the algorithm for surface reconstruction and main philosophy behind the use of $\mathcal{MS}$ transitions/transforms.

[2] Experimentally, we found that values of $\Delta_\rho = 5\%$ and $\zeta = 4\Delta_\rho$ lead to robust results.

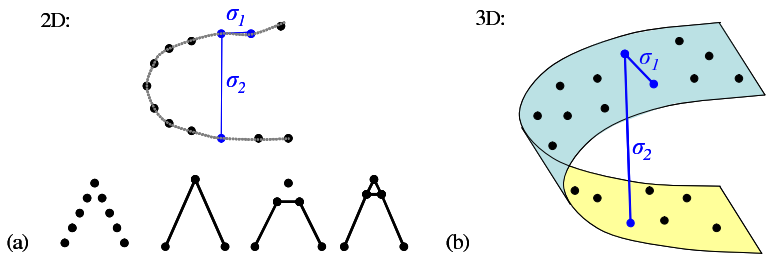


Figure 6.6: The sampling condition: intra-sampling distance $\sigma_1 \ll$ inter-sampling distance σ_2. Observe that this condition is not satisfied near sharp features.

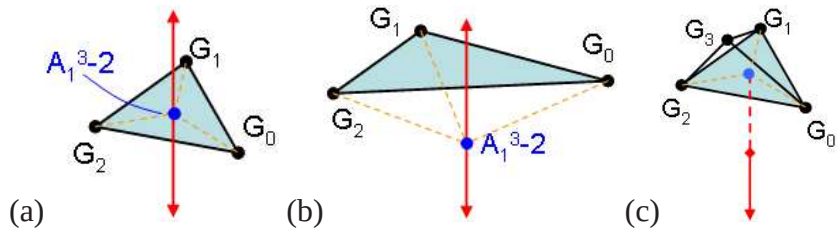


Figure 6.7: (From [45].) The three types of A_1^3 shock curves (red) and their corresponding Delaunay triangles (cyan) used in our method: (a) an A_1^3 curve with an *acute* triangle (type I), (b) an A_1^3 curve with an *obtuse* triangle (type II), (c) an A_1^3 curve not containing the A_1^3-2 critical point (type III).

points, its *generators*, which can potentially be meshed with neighbors on the original surface. Figure 6.7 shows the *three possible types* of candidate surface triangular interpolant. When there are no other sample points nearby the three generators, the minimum-radius $\mathcal{MA}$ point for these three generators is always on the A_1^3 curve: it is the critical point of the radius flow, A_1^3-2, which sits at the circumcenter for the three generators and which is used to build the $\mathcal{MS}$ [125]. This critical point is then either inside (type I) or outside (type II) the interpolating (Delaunay) triangle, Figure 6.7(a,b). Cases corresponding to types I and II are straightforward and used to create queue $\mathbf{Q_1}$. The remaining possibility of having the A_1^3-2 critical point not belonging to part of the A_1^3 curve indicates that a nearby forth generator is preventing its "formation" (*i.e.*, as being part of the coarse-scale $\mathcal{MA}$) and that the local connectivity is ambiguous (type III), Figure 6.7(c). Cases corresponding to type III are more likely to lead to meshing ambiguities and are therefore used to create a separate queue $\mathbf{Q_2}$.

Rank ordering A_1^3 curves: The rank of each A_1^3 curve is based on *(i)* the likelihood that the corresponding triangle could have arisen from the original surface given the $\mathcal{MS}$ curve length,[3] and *(ii)* the consistency of the gap transform in reconstructing a surface given already neighboring meshed sample points. We discuss each case in turn.

* Ranking A_1^3 curves ***without*** **local context.**

Ranking A_1^3 curves *without* local context: Such a ranking can be decided on the basis of the shape and size of the candidate triangle with respect to the length of the $\mathcal{MS}$ curve. Let the sides of the triangle through three generators be d_1, d_2 and d_3, and the shortest length (among the two sides) of

[3]Intuitively, the longer an A_1^3 curve is, the likelier it can approximate well the local normal field.

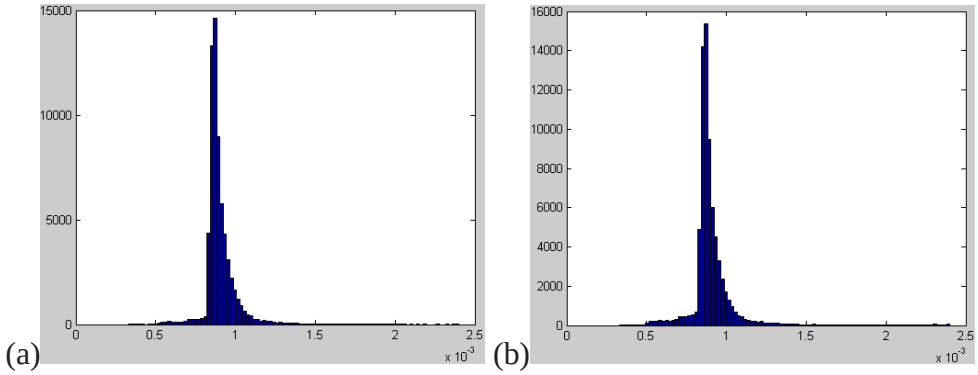


Figure 6.8: (From [45].) (a) The distribution of the A_1^3-2 radii of all shock curves corresponding to triangles in the original mesh of the Stanford Bunny model (Figure 6.17). (b) The distribution of the A_1^3-2 radii of all shock curves of type I and II in the $\mathcal{MS}$ of the point cloud. Observe how the shape of the distributions resemble each other. The median of the A_1^3-2 distribution (d_{med}) is used to set $d_{\max}$ in Eq.(6.1). [45]

the A_1^3 curve with respect to its A_1^3-2 critical point be R, and define:

$$\begin{aligned} d &= \max(d_1, d_2, d_3) \\ P &= d_1 + d_2 + d_3 \\ m &= (d_1 + d_2 - d_3)(d_3 + d_1 - d_2)(d_2 + d_3 - d_1) \\ A &= \sqrt{(P \cdot m)/16} \\ C &= 4\sqrt{3} \cdot A/(d_1^2 + d_2^2 + d_3^2)\,, \end{aligned}$$

where P is the triangle's perimeter, A is its area (Heron's formula), and C measures its compactness (Gueziec's formula). Then, the cost:

$$\rho_1 = \begin{cases} \frac{P}{R} \cdot \frac{1}{C^2}\,, & \text{if } d < d_{\max} \\ \infty\,, & \text{if } d \geq d_{\max} \end{cases}, \tag{6.1}$$

favors compact triangles rather than elongated ones, and triangles with smaller size (*w.r.t.* the shortest length R of their associated A_1^3 curve); $d_{\max}$ represents the maximal length of an expected triangle side and is set in practice as: $d_{\max} = \eta \cdot d_{med}$, where d_{med} is the median of the histogram of A_1^3-2 radii of all type I and II shock curves (Figure 6.8).[4] Observe that this cost will delay the completion of triangles which are near corners and ridges in favor of those in flat regions, away from such "close encounters" (Figure 6.9), a direct influence of the factor $1/R$ in Eq.(6.1).

* Ranking A_1^3 curves ***with* local context**: triangles sharing an edge.

We now consider the relationship between an A_1^3 curve's putative surface triangle which can either share an *edge* or a *vertex* with its neighboring already reconstructed surface triangles, Figure 6.10. First, a candidate triangle is more likely to interpolate the surface if it is oriented similarly to its neighboring triangles, sharing an *edge* as determined by the dihedral angle θ between the two. When θ is small ($< 45°$) we expect the observed continuity to offset the cost ρ_1. When θ is large ($\geq 45°$)

[4] d_{med} is used as the estimate of the expected *unit distance between samples*, while $5 < \eta < 15$ is a parameter which experimentally varies according to the sampling uniformity.

Figure 6.9: (From [45].) Steps in reconstructing a "corner shape" from 800 sample points which illustrate how our method favors meshing triangles in the region distant to the corner prior to meshing sharp features, where difficult decisions (due to under-sampling) have to be made. From Left to Right: the full $\mathcal{MS}$ (in red, with unbounded elements not shown); reconstructing the first 10%, 50%, and 90% of candidate triangles; final surface reconstruction result. [45]

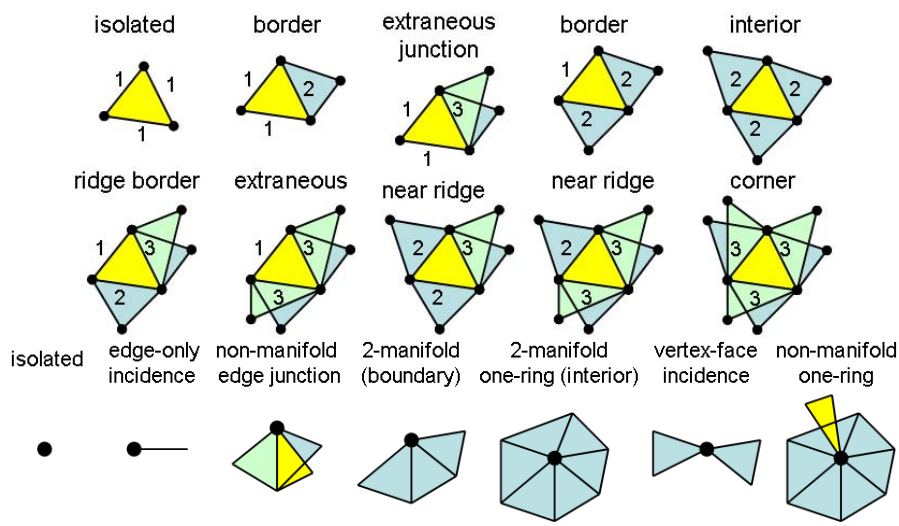


Figure 6.10: (Upper) Typology of mesh triangle topology based on simplicial adjacency at edges. (Bottom) Typology of mesh vertex topology based on its incident edges and faces. [45]

we expect the lack of continuity to rather augment the cost ρ_1. The function $f(\theta) = [\exp^{\theta} - 1]^2 - 1$ captures this notion well: at $\theta = 0$, $f(\theta) = -1$; at $\theta = 40°$, $f(\theta) \simeq 0$; at $\theta = 80°$, $f(\theta) \simeq 8.24$, giving us:

$$\rho_2 = \frac{d_i}{R} \cdot \frac{1}{C^2} \cdot f(\theta)\,, \tag{6.2}$$

where d_i is the length of the shared edge. Thus, for a triangle with its three edges part of existing smooth surface patches, the contributions for ρ_2 add up to completely cancel ρ_1 in Eq.(6.1).

* Handling the second form of local context: triangles sharing a vertex.

The second form of local context for a triangle is when it shares a *vertex* with an existing triangle. This is a locally ambiguous situation that can potentially lead to (non-manifold) topological errors

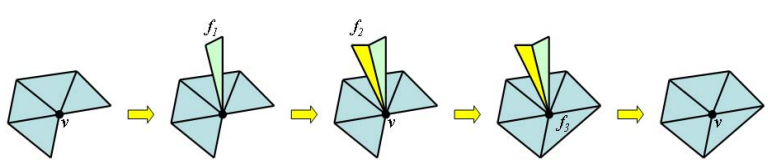


Figure 6.11: Schematic steps of meshing and fixing vertex topology dynamically in our meshing process to retain a "one-ring" around a vertex. [45]

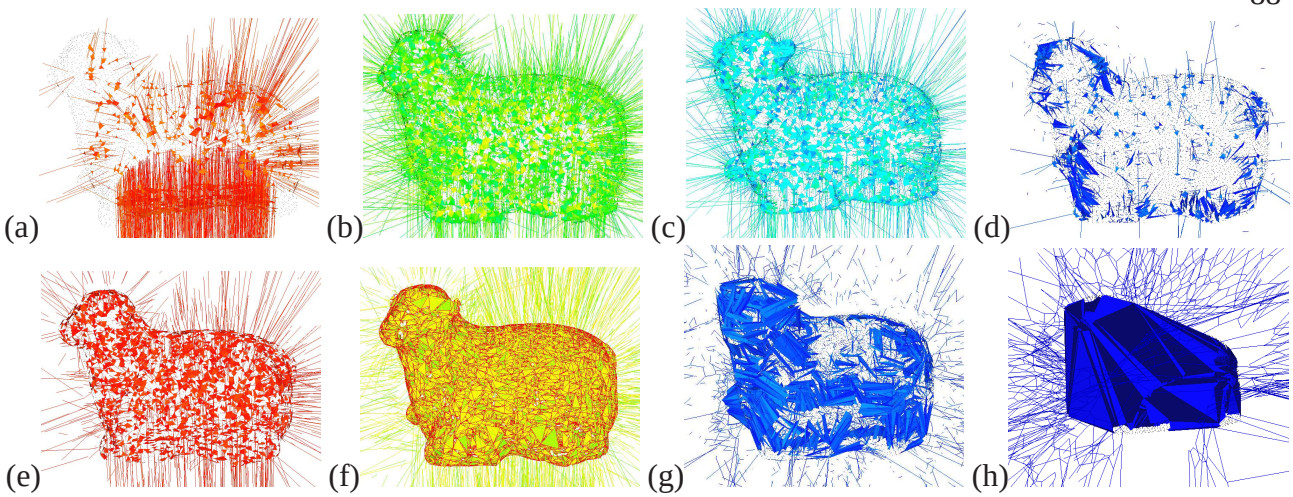


Figure 6.12: This figure illustrates how the surface interpolants (of A_1^3 shock curves) are sorted according their suitability in the classified queues for a toy sheep shape. (a-d) The most relevant 10%, 30%-60%, 60%-90%, and last 10% of type I and II shock curves in $\mathbf{Q_1}$. (e-g) The most relevant 5%, 20%-40%, and last 10% of type III shock curves in $\mathbf{Q_2}$. (h) The oversize ($d > d_{max}$) surface interpolants, which are not considered in the algorithm.

(Figure 6.10 bottom right) and must thus be delayed, avoided, or undone, to maintain a "one-ring" vertex topology, as described below in three possible cases. *(i)* If the shared vertex topology is already a *one-ring*, the new triangle should be rejected (Figure 6.10 bottom rightmost), since further meshing is unlikely to yield a better 2-manifold mesh with lower cost. *(ii)* If the shared vertex is a *vertex-face incidence*, (Figure 6.10 bottom second from right), the triangle should be delayed by increasing its cost and reinserted into its queue to be considered later. *(iii)* If the shared vertex is a *non-manifold one-ring* (the second last of Figure 6.11), the *one-ring* topology will be recovered because the gap transforms will be undone as described earlier.

* Summary of the shock segregation algorithm.

In summary, we mesh surfaces in a *best-first* manner considering the suitability of each candidate triangle: it's shape, corresponding shock curve type and length, continuity from neighbors, and the local mesh topology. Our method is implemented as a multiple-pass greedy iterative scheme. The first pass builds all confident surface triangles (with costs estimated via ρ_1 and ρ_2) from $\mathbf{Q_1}$; the second pass uses candidates from $\mathbf{Q_2}$ and resolves difficult cases using the local supports built from the first pass. Figure 6.12 depicts an example of $\mathbf{Q_1}$ and $\mathbf{Q_2}$ where the input is a set of 5,728 points sampling a toy sheep model. The meshing process is depicted in Figure 6.13. Our method can be viewed as a *multiple seeded propagation scheme*, where each A_1^3 curve without local context serves as an initial seed, whose selection is optimal (in a greedy sense) and is integrated with the meshing and error-recovery process. We note that the ability to *retract* improper meshing from errors is a major advantage over other greedy/propagation-based methods. Figure 6.14 shows an example of error recovery often observed in practice. [5]

* Extensions: exploit surface normal when available.

[5]The proposed algorithm is *greedy* with error recovery, however *not* optimal in considering candidate triangles in a best sequence and thus could produce meshes with artifacts such as the *twisted surface* in Figure 6.14, depending on the sampling condition. One solution is to use the remaining $\mathcal{MS}$ to guide the recovery process: the surface artifacts can be detected and fixed as other $\mathcal{MS}$ transforms are applied [43].

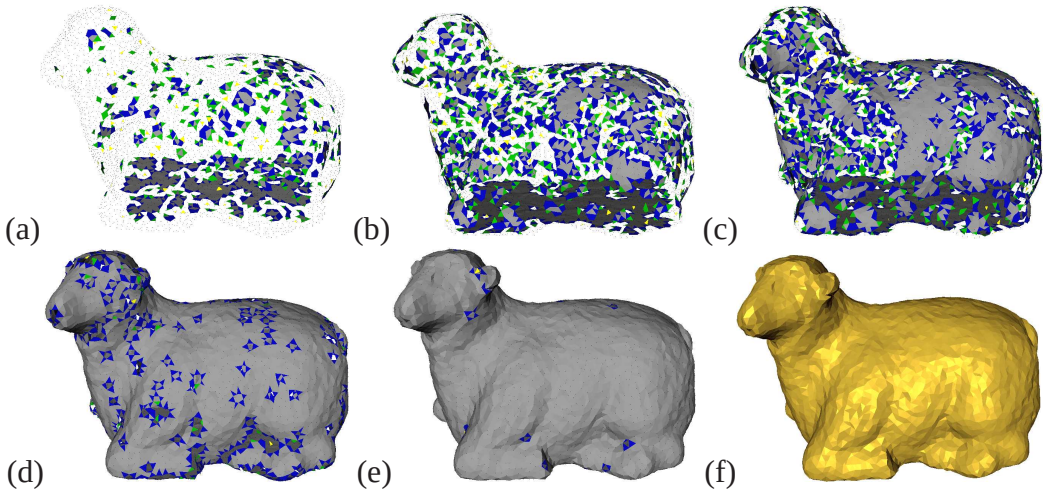


Figure 6.13: (From [45].) Intermediate result of meshing the set of $22,619$ points sampling a toy sheep model at (a) 10%, (b) 30%, (c) 50%, (d) 75%, (e) 90% completeness of the queue $\mathbf{Q_1}$, which simulates an "inverse sampling" process as described in Section 6.1. Color scheme: in gray are shown "interior" triangles (Figure 6.10), in blue, triangles with 1 boundary side, and in green those with 2 boundary sides. (f) In this case, a water-tight surface is obtained after completion of meshing the A_1^3 curves in $\mathbf{Q_1}$. [45]

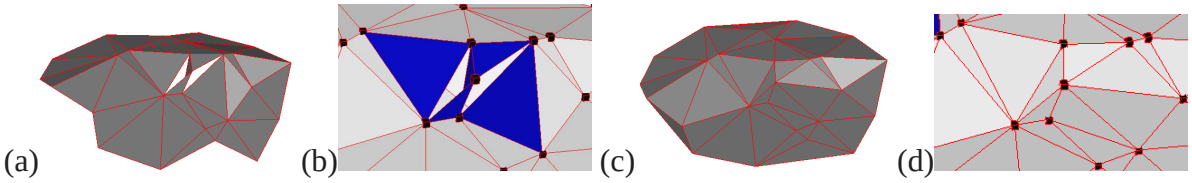


Figure 6.14: This figure shows a "local minima" typically observed in greedy meshing algorithms and a solution to recover from it. (a) After our first greedy iteration we are left with a "twisted surface" with two holes on both sides. (b) Detailed view of the twisted surface. The introduction of the half-occluded triangle in the middle prevents the meshing to be 2-manifold and no further meshing is possible unless a retraction is allowed. (c,d) A retraction and re-meshing of all local triangles (blue) give a slightly different (but valid) manifold mesh.

Our method can be further fine-tuned when additional information is available. For example, each A_1^3 curve can be shown to robustly estimate a surface normal, a result which can be derived from Amenta *et al.*'s analysis of *poles* [5]; thus, when normals are available in conjunction with a point cloud (as many main-stream methods assume), the candidate A_1^3 curve in the normal direction should be prioritized. When some existing mesh connectivity is already available (*i.e.*, a partial mesh to be re-meshed), the local structure of the $\mathcal{MS}$ sharing such connectivity can be used to correctly select A_1^3 curves with associated surface triangles (detailed below).

* Handle non-manifold surfaces.

A second extension of our method is to reconstruct surfaces with coarse-scale non-manifold *seams* (path of intersecting junctions) [227] typically observed in CAD models and other multi-media objects, Figure 6.15. This problem is further ill-posed and rarely considered in the literature [209], since without the manifold constraint, any sample point in the vicinity could potentially be meshed. We propose to add a third queue $\mathbf{Q_3}$ (after $\mathbf{Q_1}$ and $\mathbf{Q_2}$) to handle the re-meshed surfaces closer

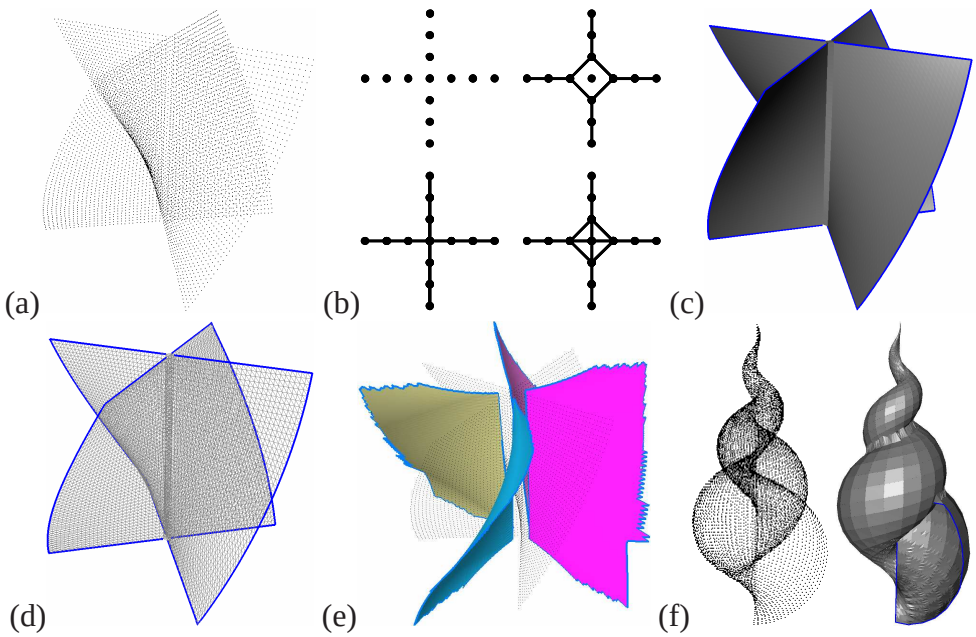


Figure 6.15: Reconstructing surface with coarse-scale self-intersecting topology: (a) an artificial non-manifold surface with '+' like cross-sections rotated along an axis. (b) The sample points in the cross section and possible reconstructions in 2D. (c) The resulting surface where the mesh structure is depicted in (d) and the corresponding $\mathcal{MA}$ in (e). (f) Another example of meshing a seashell model with non-manifold seam junctions.

to such (coarse-scale) seams and surfaces with (true) boundary and large holes. This extends our approach to handle all surface topologies in Figure 6.2. [6] The coupled $\mathcal{MA}$ (Figure 6.15(e)) can be used to organize the resulting surface: an A_3 *rib* curve corresponds to a *ridge* or seam on the surface.

6.3 Experimental Results

We have implemented our method and extensively tested it in reconstructing surfaces with *various topologies*: closed (Figures 6.1(d,g), 6.13, 6.16(a,b,d), 7.14, 6.20, 6.23(e)), with multiple components (Figure 6.5, 6.18), non-orientable (Figure 6.1(b)), multiply punctured (Figure 6.1(c)), with multiple holes (genus) (Figures 6.16(b), 7.14(e), 6.20), closely knotted (Figures 6.1(d), 6.16(a), 6.23(e)), with boundaries (Figures 6.1(a-c,e-f), 6.5, 6.9, 6.15, 6.16(c), 6.17, 6.18, 6.19, 6.24), with sharp ridges (discontinuous in curvature) (Figures 6.1(e-f), 6.9, 6.20, 6.19, 6.24), non-manifold surfaces intersecting at seams (Figures 6.1(a), 6.15(c,f)), and with (relatively) low sampling (Figure 6.1(d), 6.14, 6.16(b-c), 6.19 (in each bucket), 6.23(e)). [7]

We have also tested our method for inputs made of *non-uniform* sampling as Figure 6.1.(c), Figure 6.16.(b,c) and Figure 6.23 demonstrate. The performance of our method degrades reasonably as *perturbation* increases, Figure 6.24. In addition, our method is capable to handle complex surfaces of relatively low (but more uniform) sampling, such as in Figure 6.1(d), 6.16(b), and 6.23(e).

* Three other useful applications of the surface meshing approach.

[6] Additional heuristics can be used *a posteriori* to fill remaining (large) holes if needed.

[7] Color scheme: all surfaces in "gold" are closed (water-tight) by our method; otherwise they are left in gray, with boundaries in blue.

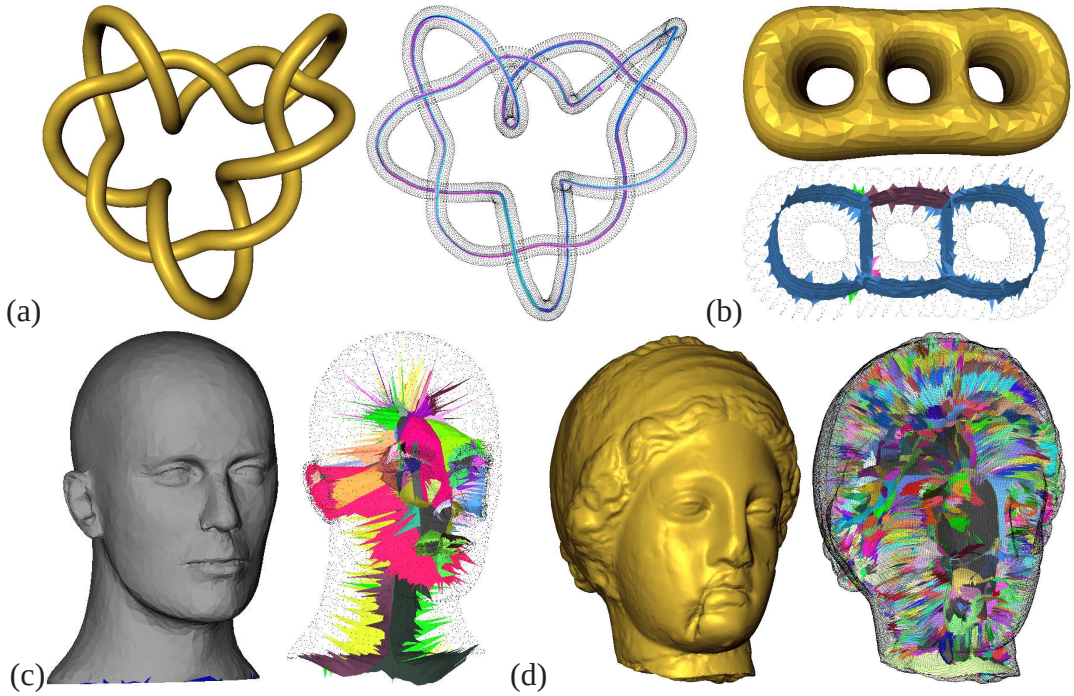
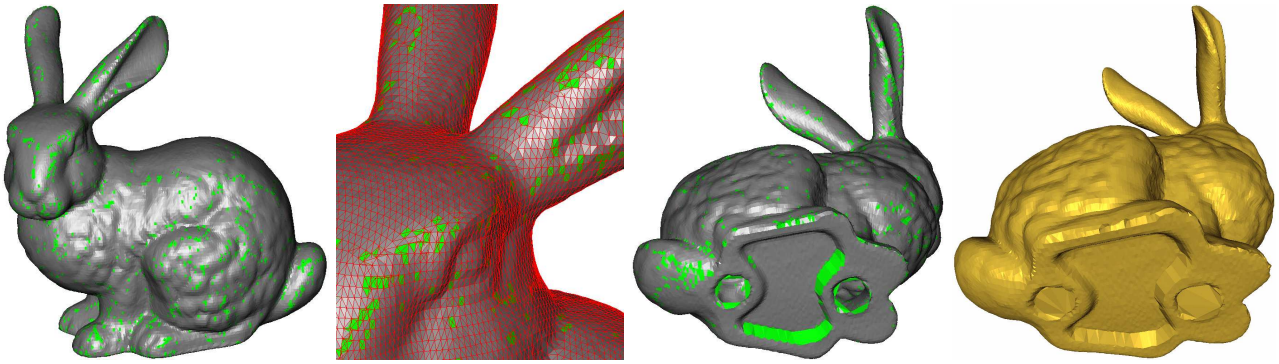


Figure 6.16: (From [45].) Results of our reconstructed surfaces and corresponding $\mathcal{MS}$'s. (a) Uniformly sampled knotted figure ($28,653$ points, $57,306$ faces, data from MPII). (b) Non-uniformly and low- sampled triple donut shape ($1,996$ points, $3,999$ faces). (c) Cyberware Mannequin ($6,386$ points, $12,727$ faces); NB: not a solid, bottom is left open. (d) Cyberware Igea ($134,345$ points, $268,686$ faces). The initial $\mathcal{MS}$ after the segregation typically contains abundant details about the shape. Further regularization of the $\mathcal{MS}$ into a succinct form such as the ones in Figure 7.14 is detailed in [43].



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

Figure 6.17: (From [45].) Validation: Superimposing our resulting mesh on the Stanford Bunny shows that most of the original mesh is recovered. (a,b) The minor differences (green) are the result of different triangulations of geometrically similar surface patches. (c) The narrow strips of holes at the bottom of the model are filled and the two larger circular holes are left unfilled. (d) Increasing η to be 30 (and thus increasing $d_{\max}$) allows larger triangles to close these holes, producing a water-tight model. [45]

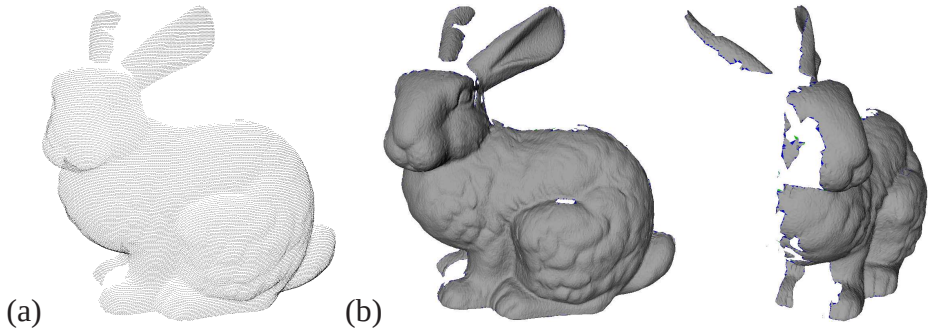


Figure 6.18: (From [45].) Surface reconstruction from (a) a single scan of the Bunny (40,256 points) from the Stanford scanning repository. (b) Two views of our meshing result. [45]

In addition to meshing unorganized points, we point out *four* other useful applications below: *(1)* re-meshing/repairing a partial mesh, *(2)* accurate fusing of row scans data, *(2)* computing a tightly-coupled $\mathcal{MA}$ with the meshed shape, which is useful in further modeling and matching applications [43], and *(4)* handling large datasets, which will be elaborated in § 6.4.

Re-meshing partial meshes: This can be done in two ways, either *(i)* by keeping all existing triangles known *a priori* to be correct and letting other candidate triangles compete to grow surface patches from these, or *(ii)* by assigning high priority to existing triangles (in a "*polygon soup*") over other new candidates and letting the algorithm re-mesh them to a final surface. Figure 6.17 validates our reconstruction result against the original model.

Application to accurately fuse row scans. In the domain of digital scanning of 3D objects, the iterative closest point (ICP) [22] is frequently used in registering two or more scans to fuse them together. The basic idea of ICP is to move toward the closest point of a target model and iterate on all sample points until they converge to a final position. It is well known that if a meshing between sample points is available, the point-to-mesh ICP performs better than a point-to-point ICP [166]. Our ability to mesh unorganized points suggests a potential to accurately fuse raw scans by first meshing individual raw scans into initial meshes and fuse them using a point-to-mesh ICP to better align them. Figure 6.18 shows an example of meshing a raw scan data. We can further pre-smooth the initial raw scans using the prior knowledge of the characteristics from the scanners (if available) to remove outliers, in prior to the meshing/fusing steps to further increase the fusion accuracy. [8]

Extension to robust $\mathcal{MA}$ computation: The proposed method leads to a successful $\mathcal{MA}/\mathcal{MS}$ computation scheme in producing a coupled shape-skeleton representation [43], which is useful in a range of shape modeling and matching applications. The other $\mathcal{MS}$ transforms (in Chapter 5) are involved to further regularize the remaining $\mathcal{MS}$ into a simplified form, while in the process, the $\mathcal{MS}$ is modified together with the tightly-coupled surface mesh [43]. This will be further investigated in Chapter 7.

6.4 A Bucketing Technique for Handle Large Inputs

* Survey on methods handling large datasets.

Recent needs in handling huge collections of samples, *e.g.* from laser scanned points, often demand out-of-core techniques which are scalable to such large datasets. Typically, the computation is first performed locally and then iteratively refined to produce a globally consistent result. For example,

[8] Refer to our technical report for more details on this approach.

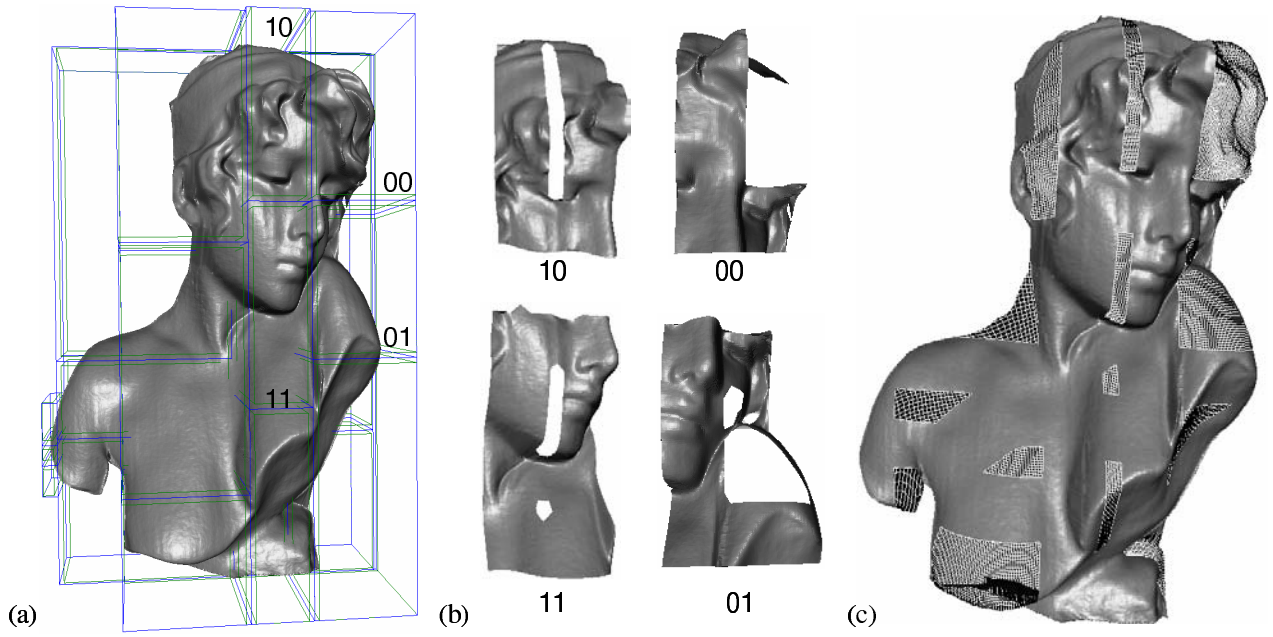


Figure 6.19: This figure illustrates the bucketing/stitching approach on the Sapho dataset, where the bucket size is made small (10 K points each) to exaggerate the stitching regions. (a) The buckets (blue) and the final surface after stitching. (b) The *stitching meshes* of the 4 indicated buckets. (c) The final stitching surface is the union of all confident triangles in the stitching meshes.

the *super-cocone* in [63] uses an octree subdivision and runs the *cocone* algorithm multiple times towards a final surface. Allegre *et al.*'s (out-of-core) streaming surface reconstruction to handle large point sets [2, 3]. Notably, a recent multi-level streaming Poisson-based approach [34] handles up to 400 million oriented points. In our novel method discussed below, we rely on an adaptable and scalable partition of space function of local sampling densities to also produce a multi-pass version of our reconstruction scheme, allowing us to handle arbitrarily large datasets.

* Approach: bucketing and stitching to handle large datasets.

An important point not raised thus far is that since we have not imposed any strong restriction on the surface topology, we can handle *very large datasets* by simply dividing the space into 3D buckets and meshing surface within each bucket, Figure 6.22, Figures 6.1(g), 6.20, 6.19. [9] We can then stitch the surface pieces together to get a final model using the same algorithm *again*. Prior to stitching, we exclude the un-reliable triangles near bucket boundaries (*i.e.*, those which are built without support of nearby points from adjacent buckets). Then, the stitching of surfaces in adjacent buckets can be viewed as completing or repairing a partial mesh (by taking already meshed triangles near bucket boundaries as an initial solution).

* Details on stitching the surfaces in buckets.

[9] We make use of an efficient way of constructing 3D buckets in the course of computing the $\mathcal{MS}$ as described by Leymarie & Kimia [125]. The two common ways to create buckets are as follows: *(i)* **grid-cell** bucketing: divide the space into $N \times N \times N$ buckets, where the number of sample points in buckets can be different, Figure 6.20. *(ii)* **adaptive** (variable-size) bucketing: each bucket contains roughly the same number of points and can be different in size, Figures 6.22, 6.1(f) and 6.19.

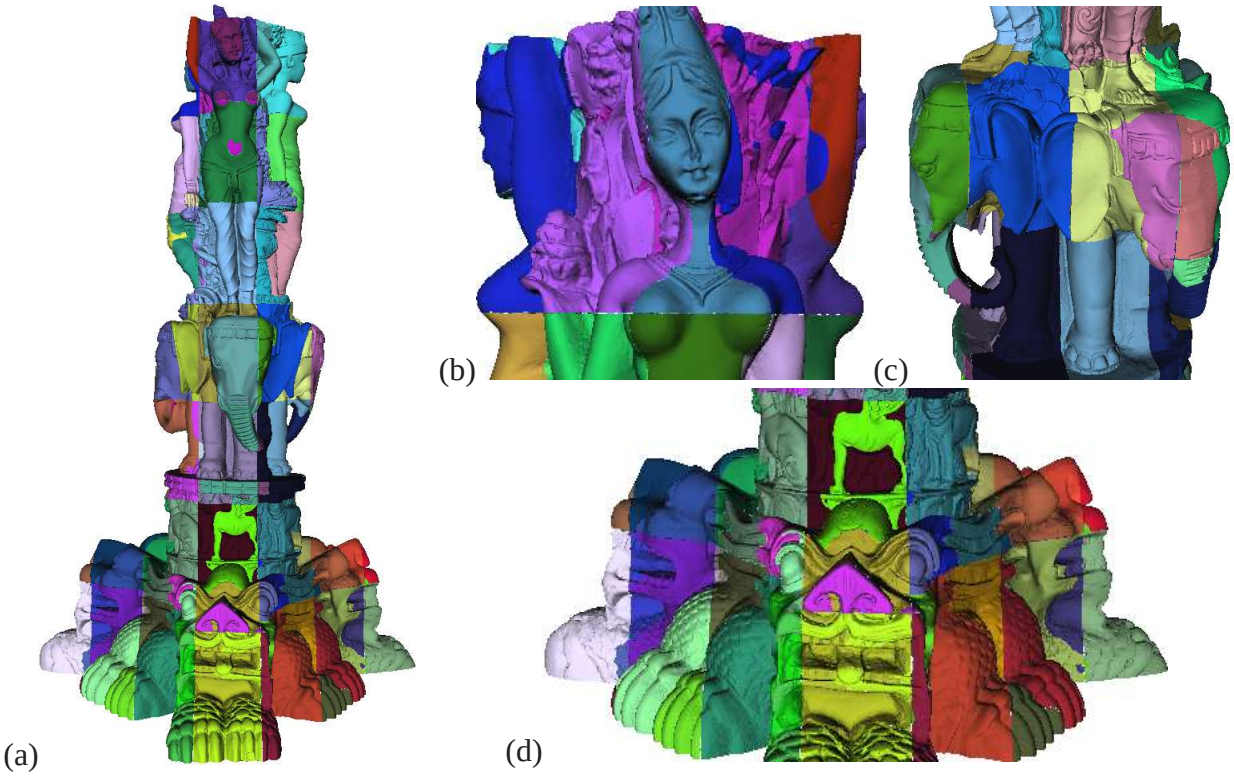


Figure 6.20: (From [45].) Views of the individual reconstructed surface of the Stanford Thai Statue dataset ($\approx 5 \times 10^6$ points) inside the $7 \times 7 \times 7$ 3D buckets. Observe that the surface is fully meshed except at the boundary of the buckets. [45]

The stitching algorithm is briefly described as follows. *(i)* First, after the triangles inside each bucket are meshed, the un-reliable triangles near bucket boundary are excluded and all remaining triangles are considered *confident* and finalized. *(ii)* Second, the triangles to stitch each bucket near its boundary (in the *stitching sausage*) are meshed, by finding all finalized triangles in the enlarged vicinity of nearby buckets as initial triangles and run the meshing algorithm to seamlessly stitch them. Again the un-reliable triangles are excluded from this *stitching mesh* of each bucket. *(iii)* Finally, the union of all confident triangles (inside all buckets and all stitching meshes) gives the final surface, Figure 6.19. The operation inside each bucket/stitching sausage is *local*, so it is scalable to arbitrary large datasets. The only assumption is that the sampling density is limited to guarantee d_{max}, thus constraints the confident triangles. As an extension, a *meta-bucket* (multiple layers of buckets) can be exploited to handle huge datasets containing billions of points. Also, the meshing of surfaces inside each bucket can be run in parallel, *e.g.*, on multi-core architectures. Figure 6.19 illustrates our implementation. Figure 6.21 shows an application of our approach in meshing unorganized scanned points in urban modeling.

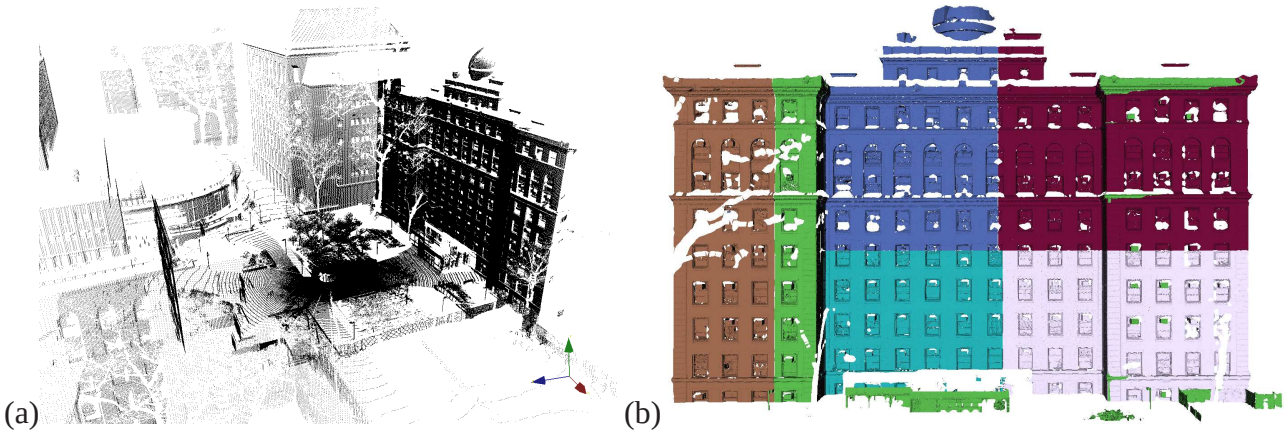


Figure 6.21: (a) Unorganized $2,113,141$ points scanned from Columbia Pupin plaza. (b) Reconstructed surface: totally $950,710$ faces on $781,026$ points.

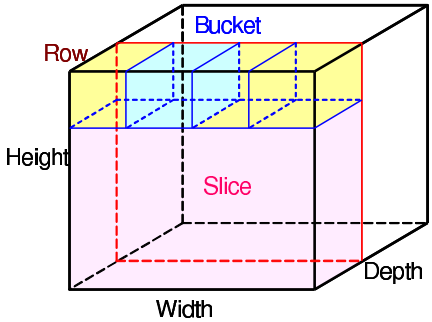


Figure 6.22: The sub-division of the 3D space into splices, rows, and buckets.

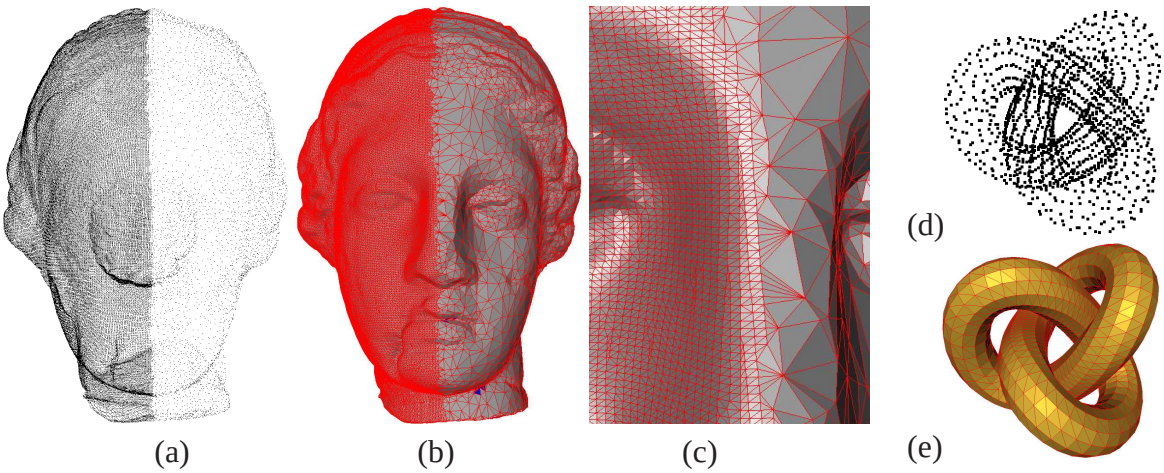


Figure 6.23: (From [45].) (a) Igea dataset non-uniformly sampled with $75,545$ points (from MPII). (b) Our result made of $75,545$ vertices and $145,082$ faces. (c) Observe how our method preserve fine details on the densely sampled surface. (d-e) The meshing of a low-sampling ($1,440$ points) knotting surface (Data come with MeshLab, an open source software); refer to Figure 7.13 for its $\mathcal{MS}$). Our algorithm recovers the surface by propagating from the (outside) non-ambiguous regions which helps to solve the ambiguity in the tangled regions. [45]

Initialization:

1. Compute the full medial scaffold ($\mathcal{MS}$) of the input points.
2. Determine d_{med} and d_{max} from the A_1^3 shock curve distribution analysis.
3. Insert all type I and II shock curves with $d < d_{max}$ to $\mathbf{Q_1}$.
4. Insert all type III shock curves with $d < d_{max}$ to $\mathbf{Q_2}$.

First greedy iteration:

Repeat

1. Remove the next shock curve γ with least cost from $\mathbf{Q_1}$:
2. If meshing of γ disallow any local candidate γ_2 with similar cost, delay both γ and γ_2 by increasing their costs and re-insert to $\mathbf{Q_1}$.
3. If the meshing of triangle T_γ does not violate the 2-manifold topology, perform *gap transform* on γ, modify the costs of neighboring shock curves $\{\gamma_n\}$. Otherwise add all local shock curves violating the 2-manifold topology to $\mathbf{Q_2}$.
4. If any non-manifold one-ring surface topology is detected, undo gap transform(s) to recover an one-ring.

Until $\mathbf{Q_1}$ is empty.

Second greedy iteration:

Repeat

1. Remove the next shock curve γ with least cost from $\mathbf{Q_2}$:
2. If the meshing of triangle T_γ does not violate the 2-manifold topology, perform gap transform on γ, modify the costs of neighboring triangles $\{\gamma_n\}$.
3. If any conflict of mesh topology is detected, undo gap transform(s) to recover an one-ring.

Until $\mathbf{Q_2}$ is empty.

Third greedy iteration:

Table 6.1: Pseudo code of our surface reconstruction algorithm.

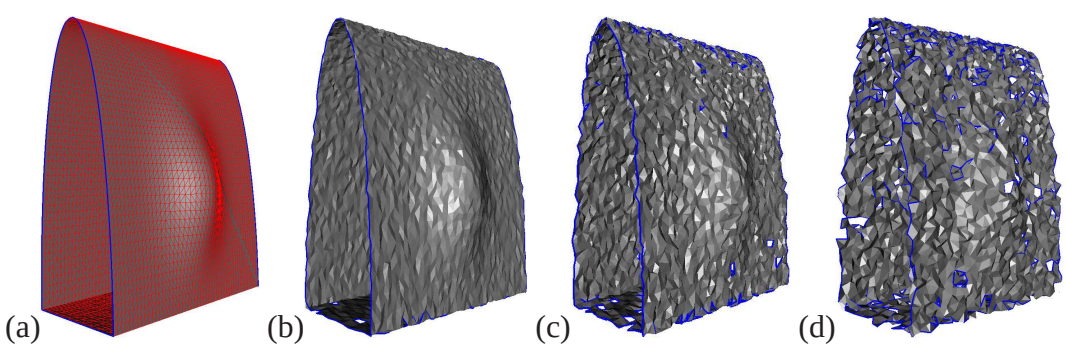


Figure 6.24: (From [45].) This figure shows how our method performs on inputs ($5,701$ points) under perturbation *w.r.t.* d_{med}; Max. displacement $d_n = n \times d_{med}$, where n is a percentage; each point coordinate, x, y, z, is perturbed by a factor $h = rand \times d_n$, where $rand$ is a random number between $[-0.5, 0.5]$. Left to Right: original dataset ($8,014$ faces); 50% of noise ($11,196$ faces); 100% ($11,101$ faces); 150% ($11,018$ faces). Topological quality is initially reasonable and then degrades as the noise extent increases. [45]

Models	# of points	Time (sec)
sheep	22,619	6.22
Stanford bunny	35,947	10.97
bunny scan 1	40,256	10.297
igea (non-uniform)	75,545	26.56
sapho	120,965	33.6
alison (whole body)	121,723	40.954
igea	134,345	50.296
ball joint	137,062	52.734
john (whole body)	146,614	50.906
elephant	206,618	72.953
hip	265,081	105.906
boat	291,117	102.844
john	321,239	132.531

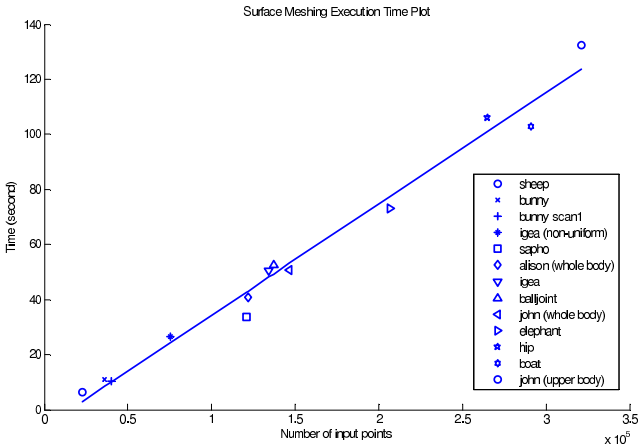


Figure 6.25: Computation time plot of the proposed surface meshing implementation on a PC with a Pentium 4 3G MHz CPU and 2GB RAM.

Chapter 7

Medial Scaffold Regularization

* Overview of chapter: computational aspects in $\mathcal{MS}$ regularization.

This chapter describes the computational aspects of how the $\mathcal{MS}$ transforms defined in Chapter 5 is performed to **regularize** the $\mathcal{MS}$ in the dual-scale (hypergraph/mesh) representation. A computational pipeline is developed — starting with unorganized points as input to compute the $\mathcal{MS}$ and regularize it with full automation. The computation continues from the *shock segregation* process (Chapter 6) and further regularizes the remaining $\mathcal{MS}$. As a next step, the coarse-scale structure of the $\mathcal{MS}$ is built to perform the subsequent $\mathcal{MS}$ transforms. This chapter elaborates two aspects of the computational scheme: *(i)* the strategy to choose a suitable computational scheme (algorithm) and representation (data structure) to overcome several computational bottlenecks, and *(ii)* the maintaining of a coupled shape boundary with the $\mathcal{MS}$ during the regularization process.

* Organization of chapter.

This chapter is organized as follows. Section 7.1 motivates the computational strategy in applying the $\mathcal{MS}$ transforms for regularization. Section 7.2 elaborates the algorithm in detailed steps. Section 7.3 is on how to associate the boundary (points) with the $\mathcal{MS}$ elements in the regularization process. Finally, Section 7.4 presents experimental results. The applications of the regularized $\mathcal{MS}$ will be shown in Chapters 10.

7.1 Overview of the Computational Strategy for $\mathcal{MS}$ Regularization

* Overview of the computational pipeline.

The proposed $\mathcal{MS}$ computational scheme starts with *unorganized* point samples as input. [1] The goal is to regularize the $\mathcal{MS}$ by applying the set of transforms defined in Chapter 5 in a suitable order, such that the resulting $\mathcal{MS}$ approximates the true $\mathcal{MA}$ of the shape. Specifically, the regularized $\mathcal{MS}$ must be *(i) topologically* correct, *(ii) geometrically* accurate (feature preserving), and *(iii)* consistently associated with its boundary shape.

We have discussed the first part of the computational pipeline in Chapter 6, *i.e.*, the initial computation of the full $\mathcal{MS}$ and the *shock segregation* process (gap transforms, § 6.2). The next step is to further regularize the remaining $\mathcal{MS}$ by applying the full set of transforms (splices, contracts, merges, *etc.*).

[1] Refer to § 2.2.1 and § 6 for justification on considering the *unorganized* points as the general form of input for 3D shapes.

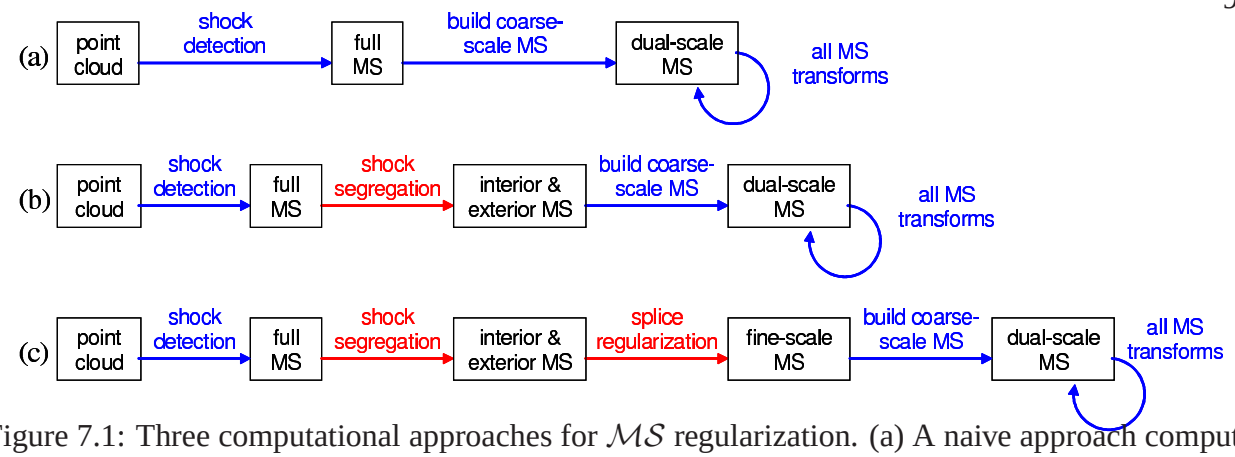


Figure 7.1: Three computational approaches for $\mathcal{MS}$ regularization. (a) A naive approach compute the (full) $\mathcal{MS}$ directly from the input and build a coarse-scale representation to apply all $\mathcal{MS}$ transforms, which consumes the most computational resource. (b) A better approach first perform the segregation (gap transforms) to obtain a surface mesh and then apply all $\mathcal{MS}$ transforms. (c) The proposed approach is more efficient in grouping together gap transforms and splice transforms into individual steps and perform all $\mathcal{MS}$ transforms only on the necessary parts.

* Need of a (coarse-scale) structural representation.

An immediate computational issue in applying the $\mathcal{MS}$ transforms is that a *dual-scale* representation is required to perform the $\mathcal{MS}$ hypergraph edits (Chapter 3). To illustrate, a splice transform needs to 'splice' together two sheets, a contract transform needs to combine together two elements, and a merge transform can split a medial sheet/curve into two. An intuitive idea is to view the *segregated* $\mathcal{MS}$ as the *fine-scale* structure (since it is an immediate result from input points). The construction of a higher-level structure is necessary.

* Computational bottlenecks of a brute-force approach.

Ideally all the $\mathcal{MS}$ transforms covered in Chapter 5 should be considered and rank-ordered in the regularization process. A naive approach is to construct the coarse-scale sheets/curves directly from the fine-scale mesh by grouping together elements with the same topology, Figure 7.1(b). [2] While this approach is capable in considering all $\mathcal{MS}$ transforms in a 'best-first' fashion (ranked-ordering by their *deform costs*), the problem in practice is of three-folds:

- *Limitation of computer memory*: Observe that the immediate segregated $\mathcal{MS}$ is extremely noisy in containing numerous "spiky" medial *tabs* due to sampling artifacts, Figure 7.3(c). This approach requires to represent all $\mathcal{MS}$ sheets, curves, and nodes in both scales, which is a huge burden in computer memory. (Clearly, this redundancy can be avoided by performing low-cost simplification in priority, detailed below.)
- *Redundant computation*: Since the $\mathcal{MS}$ is initially noisy, most computation in this approach will be redundant, in that many low cost transforms are performed and later the resulting $\mathcal{MS}$ elements are removed by subsequent transforms. (Refer to Figures 7.3, 7.4, and 7.5 to see how much this redundancy could be by comparing the number of sheets in the initial and resulting

[2] A further generalization in computation is to combine the gap transforms (in shock *segregation*) as well into an unified iteration of all transforms in Figure 7.1(a). However, such scheme requires more computational memory and time, since the full $\mathcal{MS}$ is more complex than the segregated $\mathcal{MS}$ (see Figure 7.3 for an example). To give a practical sense, if the full $\mathcal{MS}$ of a set of random points is put in this dual-scale structure, we have a serious limitation in capacity of $5K$ to $20K$ points on a modern PC with 2GB of memory.

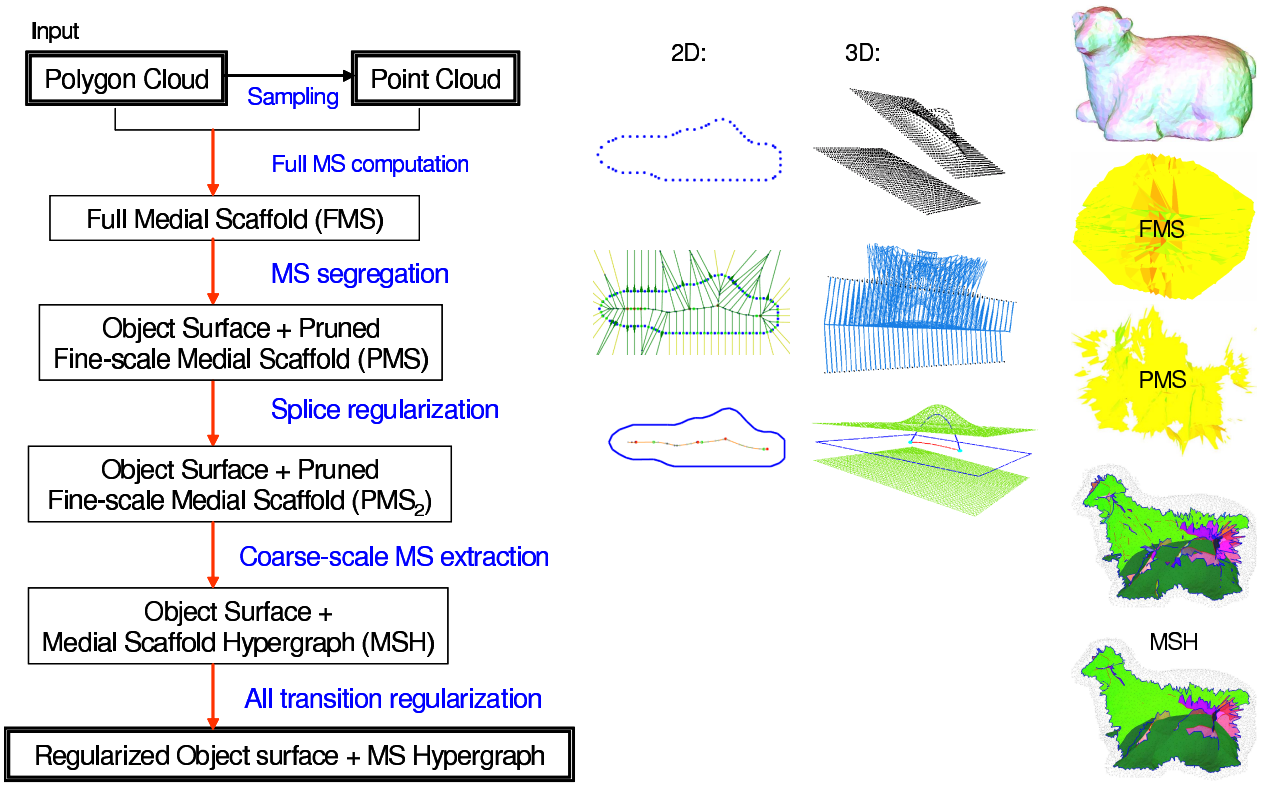


Figure 7.2: The proposed $\mathcal{MS}$ computation and regularization pipeline. On the right are the snapshot of the $\mathcal{MS}$ under processing in each step.

$\mathcal{MS}$.) A solution to avoid this redundancy is to perform simplification transforms such as the splices in prior to the others.

- *Difficult in performing transforms*: In each $\mathcal{MS}$ transform, a consistent boundary shape must be maintained in order to estimate the costs of subsequent transforms. At this stage where only a fine-scale $\mathcal{MS}$ mesh is available, it is difficult to estimate costs and update the boundary, since only very primal discrete approximation is available. [3]

The above thoughts motivate the following strategies in our approach:

- *Most effective first* to order transforms: The most effective transform (with smaller cost) in simplifying the $\mathcal{MS}$, such as the *splice transforms*, should be performed in prior to others.
- *Group similar transforms*: The transforms of the same type and with similar costs should be grouped together and applied in a batch. This suggests a *'multi-pass'* iterations of transforms, *i.e.*, continuing from Chapter 6 of low-cost *gap* transforms, one should apply low-lost *splice* transforms (§ 7.2.1).
- *Build a coupled boundary surface*, in prior to a full treatment of all transforms: Since our computation starts with points and the $\mathcal{MS}$ transforms operate tightly with the boundary, it is necessary to first produce a surface meshing of the points, to better estimate the transform cost and subsequent changes. This also motivates the shock segregation in Chapter 6 as a

[3] For example, a "spiky" shock tab with a single A_1^2 element in Figure **??** are associated with only two generators from the corresponding boundary tetrahedron. (This issue is more involved and will be covered in § 7.3.)

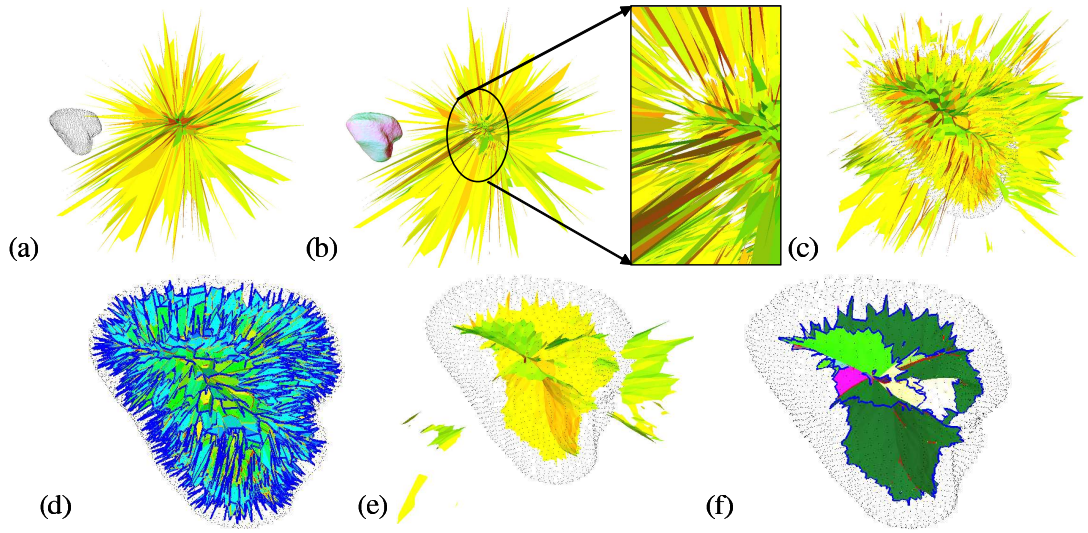


Figure 7.3: The regularization of the $\mathcal{MS}$ of a bone in low sampling. (a) The full $\mathcal{MS}$ recovered from QHull (the unbounded sheets to infinity is not shown). (b) The $\mathcal{MS}$ after shock segregation. Zoom in to see the inside and outside $\mathcal{MS}$ are separated. Many *"spiky"* tabs (spurious artifacts) due to sampling. (c) $\mathcal{MS}$ inside an enlarged bounding box of the input points. (d) The coarse-scale $\mathcal{MS}$ of (c) is very noisy. (e) After the r-min splice regularization. the $\mathcal{MS}$ in (f) contains much less complex graph structure.

standalone process. However, the resulting $\mathcal{MS}$ contains many "spiky" shock tabs (artifacts). Since a *coarse-scale* $\mathcal{MS}$ is not available at this stage, we propose to use the following intermediate representation to apply subsequent splice transforms to regularize them.

- *Use of a* **reduced** *representation* in the initial stage: The idea is to migrate the (noisy) fine-scale $\mathcal{MS}$ mesh toward the (regularized) coarse-scale $\mathcal{MS}$ hypergraph by perform the *gap* and *splice* transforms on an intermediate *minimum* (but sufficient) representation, which minimizes the computational burden. We propose to use a *reduced* representation (steps marked 'red' in Figure 7.1) for such need, where the coarse-scale medial sheets only contain pointers to their A_1^2 elements (and all other coarse-scale curves and nodes are kept *implicit*). The advantages are two-folds: *(i)* It avoids to explicitly maintain a coarse-scale hypergraph structure, which effectively relieves the memory bottleneck, *i.e*, allow to handle larger dataset. [4] *(ii)* It ensures no information is lost in recovering the *full* (original) dual-scale representation in the later stage (*i.e.*, all topology and geometry can be completely recovered, detailed in § 7.2.4). There is only a few relatively minor penalties in this approach: *(i)* limited $\mathcal{MS}$ transforms in support (only the gap and splice transforms are straight-forward to perform), *(ii)* inaccuracy in estimating the transform costs, *(iii)* incapable to modify boundary (at a detailed geometric level), and *(iv)* a few extra steps to recover the full representation.

7.2 The Computational Pipeline for $\mathcal{MS}$ Regularization

This section elaborates the proposed $\mathcal{MS}$ regularization pipeline (Refer to Figure 7.2 for a summary). A few optional steps are introduced to improve and fine-tune the resulting $\mathcal{MS}$ (detailed below). Refer to Figures 7.3, 7.5, and 7.11 for results of the intermediate steps.

[4] The current implementation takes up to 300K points as input, see § 7.4.

1. Computation of the full (fine-scale) $\mathcal{MS}$ from the input point cloud.
2. A **segregation** step on the full $\mathcal{MS}$ (surface meshing via *gap* transforms), which segregates the *sampling artifact* scaffold from the remaining $\mathcal{MS}$, which reflects the true symmetry. The main procedure is a greedy sequence of A_1^3-curve gap transforms on the fine-scale $\mathcal{MS}$. (§ 6.2)
3. An early **splice regularization** step (*splice* transforms on low-cost fine-scale A_1^2 sheets), which prunes out most noisy spike-like shock tabs and greatly simplify the $\mathcal{MS}$ topology. The main procedure is a greedy sequence of A_1A_3-I and $A_1^2A_3$-II sheet splice transforms on the *reduced* representation. (§ 7.2.1)
4. The **regularization of** A_3 **ribs**. The A_3 rib curves are important which corresponds to the 'boundary' of the $\mathcal{MS}$ hypergraph (*i.e.*, the initial $\mathcal{MS}$ boundary formed in the 'grass-fire' propagation [87]). It can be noisy due to the sampling effect (of points as input), Figure 7.3(e). This step regularizes the A_3 rib curves by applying the splice transforms element-by-element to 'trim' them in a way that the ribs of salient features are kept. (§ 7.2.2)
5. An (optional) **component selection** step. Our system works with point clouds thus does not impose assumptions on the input shape topology. For a well-sampled *simple closed* surface, the $\mathcal{MA}$ is separated into two parts, *i.e.*, the *interior* and *exterior*. Typical applications focus on the interior $\mathcal{MA}$, while some others interest in the exterior $\mathcal{MA}$ (see § 10.3). This step allows to specify the $\mathcal{MS}$ component of interest. [5] (§ 7.2.3)
6. The extraction of the (full) **dual-scale** representation. Construction of the coarse-scale $\mathcal{MS}$ hypergraph from the fine-scale meshes, which consists of two steps: *(i)* connected component labeling of $\mathcal{MS}$ sheets and curves, *i.e.*, grouping the connected elements with the same topology into components; and *(ii)* a second step to recover the topology of medial sheets into the *canonical* forms (as described in Chapter 3, in the *enhanced half-edge* ($\mathcal{EHE}$) data structure detailed in Chapter 4). (§ 7.2.4)
7. An **all-transform regularization** step, which considers all transforms defined in Chapter 5 to regularize the $\mathcal{MS}$ in the (full) dual-scale representation. The main procedure is a greedy iteration of all $\mathcal{MS}$ transforms, rank-ordered by their cost metric reflecting the amount of corresponding shape changes. (§ 7.2.5)
8. A (optional) **smoothing of** A_3 **ribs**. In comparing to step 4 which regularizes the A_3 rib curves in the (fine-scale) *element-wise* level, this step further smoothes the ribs in a *sub-element* level. Specifically, curve smoothing techniques such as the *discrete curve shortening* (DCS) or Gaussian smoothing are applied to smooth the A_3 rib curves as well as its incident A_1^2 sheet elements. This step effectively removes remaining artifacts of the rib curves due to low-samplings. (§ 7.2.6)

The remaining of this section covers the detailed algorithm of each step. In an overview, § 7.2.1 describes the splice regularization, and § 7.2.2 regularizes the A_3 rib curves by applying element-wise splice transforms. § 7.2.4 constructs the coarse-scale $\mathcal{MS}$ from the selected fine-scale component in § 7.2.3. § 7.2.5 is the main step to rank-order and perform all $\mathcal{MS}$ transforms for regularization. Finally, § 7.2.6 further smoothes the A_3 rib curves.

[5] The selection of $\mathcal{MS}$ component can be manual or fully automatic, or simply keeping all components in case the notion of interior/exterior does not exist.

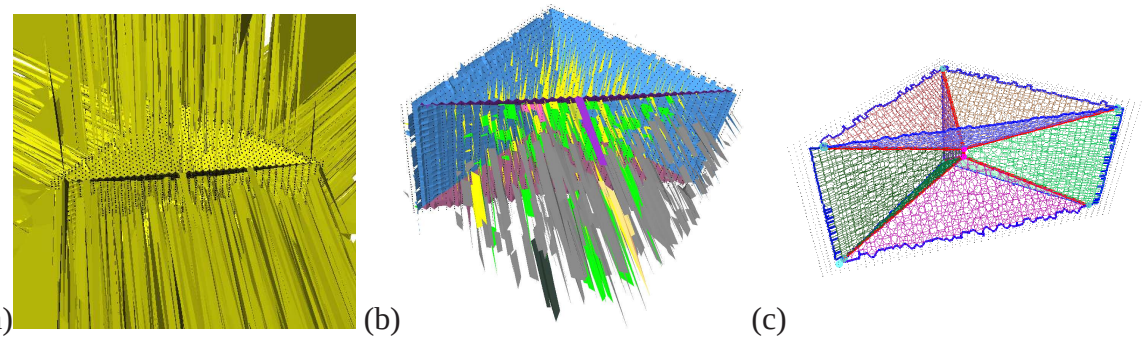


Figure 7.4: Regularizing the $\mathcal{MS}$ of a prism shape. (a) The immediate $\mathcal{MS}$ after the *shock segregation* (meshing) process [45] contains numerous noisy “spiky" medial tabs. (b) Grouping the medial sheets inside a bounding box into components (yellow: a single sheet element, green: component with 2 elements, gray: sheets connecting to the exterior $\mathcal{MS}$, other sheets are in random colors). (d) The splice transforms remove the spiky tabs; the remaining $\mathcal{MS}$ hypergraph is shown in wire frame to depict its structure.

7.2.1 Splice regularization

This step takes the remaining $\mathcal{MS}$ from the *segregation* process (§ 6.2) and regularize it by removing the *‘spike’*-like shock *tabs* (sheets with no interior *anchor* curves in the $\mathcal{MS}$ hypergraph), which are the spurious artifacts created from the sampling process. Figure 7.4 illustrates an example on a prism shape.

The main procedure is an iteration of *splice* transforms (A_1A_3-I and $A_1^2A_3$-II sheet splices) performed on the *reduced* representation, where the update of shape boundary is neglected in the transforms. The candidate shock sheets are rank-ordered by the cost of *total number of associated generators* of each sheet $\mathsf{S_i}$, which approximates the amount of shape changes corresponding to pruning $\mathsf{S_i}$. [6] The iteration stops when a pre-set cost threshold T_s is reached. We found $T_s = 10$ works well in practice.

The *reduced* representation suits well here in minimizing the computational burden. Since the coarse-scale $\mathcal{MS}$ sheets only store the minimum information (*i.e.*, pointers to the set of fine-scale A_1^2 elements), the splice operation only involves deleting and merging of sets of pointers.

7.2.2 Regularizing A_3 rib curves

* Motivation: importance of the A_3 ribs and its regularization.

The splice regularization removes the spiky tabs (first artifact of sampling) while leaving the zig-zag A_3 rib curves (second artifact of sampling) intact. This step regularize the A_3 rib curves by pruning out noisy sheet elements, Figure 7.3(d-e). Further smoothing of the rib curves involves modification within the mesh elements is detailed in § 7.2.6.

Recall from the theory in Chapter 3 (Figure 3.2) that the A_3 rib curve for a smooth shape surface should be smooth, while the rib for a sharp corner should be right at the ridge (a degenerate case).

[6] The assumptions are that *(i)* the input points are uniformly sampled, and *(ii)* the shock sheet is small enough to neglect other cost terms (such as the *shock flow* and angle ϕ) in § 5.2.1. While other complex cost definitions are possible, *i.e.*, by considering the boundary geometry (area, curvature, volume) and the shock attributes (radius, flow, angle), practical results indicate that this approximation works well.

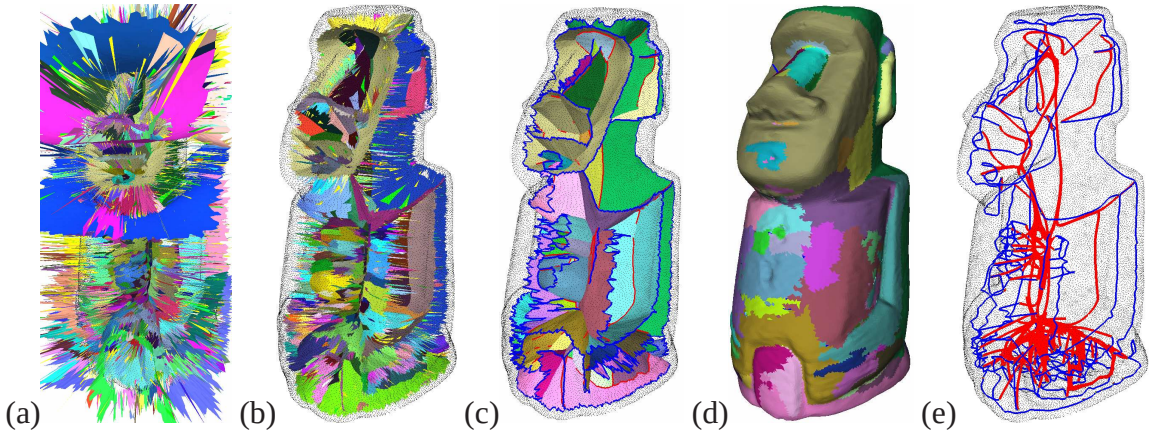


Figure 7.5: (a) Both the interior and exterior $\mathcal{MS}$ components of an Easter island statue (from MPII, 40k points) after *shock segregation* are extremely noisy. The interior $\mathcal{MS}$ ($6,084$ sheets) in (b) is regularized into 191 sheets (c), where the boundary region of each sheet is shown in (d) in the same color. The $\mathcal{MS}$ graph with all sheets implicit better visualizes its internal structure (541 curves, 354 nodes) in (e).

However, in practice of point-sampled shapes, the A_3 curves are highly unstable due to the high order of singularity of A_3 (instead of A_1 regular tangency contact). [7]

The main advantage in extracting a (robust) regularized A_3 rib curve is to capture salient features of the shape. Since the ribs are the boundary of the $\mathcal{MS}$ hypergraph, the close it is to the shape, the more detailed is captured. The rib curves map to the *ridge* curves on the surface (§ 10.2). On the other hand, the robustness of the rib curves are closely affected by the sampling density *w.r.t.* the *local feature size* (§ 2.2.1). Our strategy is to find a balance between the robustness and the accuracy of the extracted rib curves in computation. [8]

* Approach to regularize A_3 ribs: a *"min-radius"* trimming of bordering mesh faces.

We adopt a *'close-to-surface' (minimum-radius) trimming* of the A_1^2 sheets incident to the A_3 ribs, which is essentially an iteration of *splice* transforms *element-by-element*, which prunes out noisy A_1^2 sheet elements close to the boundary (within a shock radius threshold R_t and with low costs). [9] The adopting the radius threshold R_t imposes an important assumption that the object boundary is smooth to some extent (so the local radius of curvature r is larger than R_t), an idea is related to Chazal's λ-MA [47] reviewed in § 2.2.1, in that the radius threshold R_t is essentially the parameter λ.

* Algorithm for the A_3 rib curve trimming.

In implementation, we iterate the splice transforms on the fine-scale A_1^2 shock element near the boundary by ordering candidate A_1^2 shock elements by their 'maximum radius' (of all vertices of

[7] We observe that the A_3 rib curves for the point-sampled shapes are not smooth in general (sheet polygon are typically elongated along the radius increasing direction). This instability exists even if the surface is smooth and densely sampled without noise. An interesting observation is that the denser the sampling, the stronger these artifacts (sharper spikes) are observed, which requires further study.

[8] Refer to Figure 7.14 to see how the $\mathcal{MS}$ captures the loosely sampled surface features (and how close they are)in our approach, when compared to other methods reviewed in Chapter 2.

[9] The *element-wise* splice transform can be viewed as a *degenerate* splice transform, which prunes a sheet element without the 'splicing' step: There is only one shock sheet element remaining (instead of two) on the seam curve.

the polygon), until the radius threshold R_t is reached. [10] The parameter R_t is related to the input sampling condition and can be *(i)* estimated automatically from the average density between samples (as d_{med} in § 6.2) or *(ii)* specified manually. A larger R_t results in more details of the $\mathcal{MA}$ to be pruned out, while a smaller R_t might result in large zigzags on the A_3 ribs.

In additional to the above fix-radius trimming, two ideas can be exploited to further regularize the A_3 rib curves. (I) First, candidate A_1^2 sheets can be ordered by their local *saliency* for removal, which is reflected by the shock angle ϕ and speed of formation v (in § 2.3). Specifically, the *elongated* A_1^2 sheet elements are typically with larger shock angle ϕ (and lower speed v) and thus should be removed. This is implemented by ordering candidate A_1^2 shock elements by their *'compactness'*, similar to the compactness C measure of a triangle in § 6.2. [11] (II) Second, the *geodesic* distance (shortest path on the shock sheets) can be used to outline a smooth A_3 curve for trimming. The computation of geodesic field on shock sheets has been discussed in Chapter 5. This idea can be combined with the flow analysis inside the medial sheets (§ 3.4) and is referred as future works.

7.2.3 (Optional) $\mathcal{MS}$ component selection

This optional step selects a specific connected component of the $\mathcal{MS}$ hypergraph. In the case of a closed shape, the $\mathcal{MS}$ is often separated into two components, the *interior* and *exterior*. This step allow to choose the specific component of interest for further processing, or simply retain all components.

In addition, this step also provides a chance to fix topological error of the $\mathcal{MS}$ when a *prior* knowledge of the interior/exterior $\mathcal{MS}$ is available. Specifically, the greedy algorithm of the shock segregation process (§ 6.2) is not perfect that in some rare case for a close shape, a gap transform may not performed (which it should), leaving an artifact shock to connect the interior and exterior $\mathcal{MS}$ elements together. If the input shape is known *a priori* to be solid, one can force to close the boundary surface holes and remove all shocks arising from the holes (by locally enforcing gap transforms).

The system default is to keep the *largest interior* $\mathcal{MS}$ component by first cropping the $\mathcal{MS}$ elements inside a enlarged bounding box of the input, Figure 7.3(c). The largest component is typically the *interior* $\mathcal{MS}$ component. Otherwise, the second and third largest components could be manually selected. A few remarks regarding the practical implementation is detailed in the footnote. [12]

7.2.4 Construct coarse-scale $\mathcal{MS}$

* Goal: construct coarse $\mathcal{MS}$ nodes, curves, sheets and recover their inter-connectivity.

[10] In performing the splice transforms, we pass the generators associated with the pruned sheets to the remaining adjacent shock curves. The issue of maintaining a consistent generator to shock association will be covered in § 7.3.

[11] Since the A_1^2 sheet element is a *convex* polygon (corresponding to a Voronoi face), its 'compactness' can be roughly estimated by dividing it into triangles and computed from the weighted sum of the compactness of these triangles.

[12] **Remarks:** *(i)* If any portion of the $\mathcal{MS}$ is removed in this optional step by a non-standard (transform-like) procedure, one should repeat the previous *splice regularization* again to remove all low-cost shock sheets. *(ii)* (See § 7.3) The generators associated with the exterior $\mathcal{MS}$ might be *"lost"* after its removal. One can *re-assign* them to relevant shocks, *e.g.*, by assigning to: *(a)* the closest valid neighboring shocks, which can be found by a *octree* search in the remaining $\mathcal{MS}$ hypergraph. *(b)* the closest valid shocks, which is found by a search the original (unpruned) full $\mathcal{MS}$. *(iii)* For the case the interior and exterior $\mathcal{MS}$ is separated except a few errors, one can identify the *'weakest'* link on the $\mathcal{MS}$ hypergraph which connects them and break the hypergraph. This is an extension of finding the *bridge* on a graph into the hypergraph, which is related to the k-connected (bi-connected, tri-connected, ...) graph connectivity problem and the 'maximum flow and minimum cut' on a hypergraph [52].

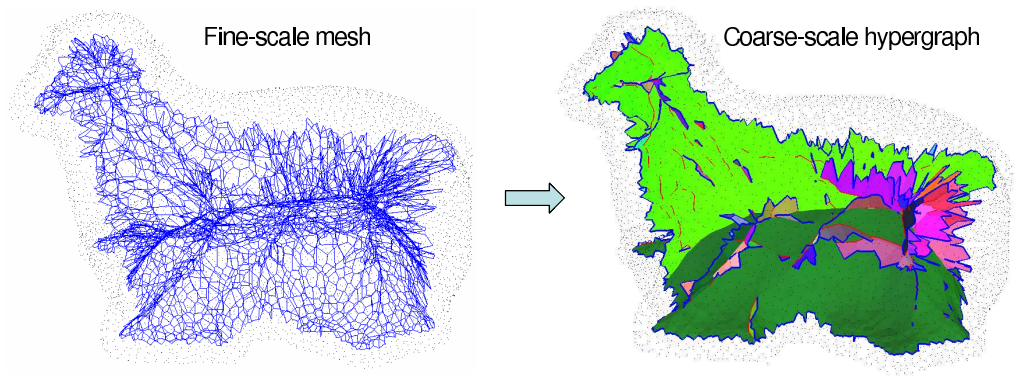


Figure 7.6: Construct the coarse-scale $\mathcal{MS}$ components from the fine-scale mesh and recover their inter-connectivity into a hypergraph. The sheet components are randomly colored.

This session recovers the coarse-scale $\mathcal{MS}$ hypergraph from the fine-scale polygonal mesh, Figure 7.6, which can be decomposed into two main steps: *(i)* construct the coarse-scale $\mathcal{MS}$ nodes (A_1A_3, A_1^4), curves (A_3, A_1^3) by grouping together curve elements, and sheets (A_1^2) by grouping together sheet elements, and *(ii)* recover the *inter-connectivity* (topological incidence) between the nodes/curves/sheets toward a hypergraph.

The topology of a $\mathcal{MS}$ curve is simple (1D *poly-line*, which can be either *closed* or *open*). In contrast, the topology of a $\mathcal{MS}$ sheet is more complex. As discussed in § 4.4 (Figure 4.1), there are three cases at a sheet boundary: *(i)* the interior/exterior boundary curves, *(ii)* the internal *anchor* curves, and *(iii)* the self-intersections (*e.g.*, 'triple junction' near an A_5 swallow-tail).

* Algorithm to recover the structural $\mathcal{MS}$ hypergraph.

The algorithm to recover the coarse-scale $\mathcal{MS}$ hypergraph is as follows.

1. *Construct coarse-scale $\mathcal{MS}$ nodes* from the fine-scale shock vertex elements of types A_1A_3, A_1^4 and higher-orders.
2. *Construct coarse-scale $\mathcal{MS}$ curves* from the fine-scale curve elements of A_3, A_1^3 and higher-orders. Specifically, traverse all coarse-scale $\mathcal{MS}$ nodes and construct each incident $\mathcal{MS}$ curve by tracing along the curve elements one-by-one (as a polyline) until another $\mathcal{MS}$ node is reached. As each $\mathcal{MS}$ curve is built, all its fine-scale edge elements are set to be visited, and the curve-to-node and node-to-curve connectivity is created properly. In case a closed curve is extracted, it is handled as a *loop* curve with identical ending nodes; refer to Figure 4.6(b).
3. *Construct coarse-scale $\mathcal{MS}$ sheets* by grouping together sheet elements. A propagation-based connected component labeling scheme is exploited, by visiting (and labeling) fine-scale sheet elements and propagate to its neighbors, until the bordering A_1^3 or A_3 curve of the current sheet is reached. The sheet-to-curve and curve-to-sheet connectivity is represented by the extended half-edge ($\mathcal{EHE}$) data structure described in § 4.4. A last step is to convert the set of half-edges of each coarse-scale sheet toward a *canonical form* detailed in § 4.4.

7.2.5 Iterative $\mathcal{MS}$ regularization using all shock transforms

This is the main step of $\mathcal{MS}$ regularization in considering all $\mathcal{MS}$ transforms defined in Chapter 5. All candidate transforms (gaps, splices, contracts, merges, loops) are detected and rank-ordered in a priority queue $\mathbf{Q_t}$ according to their *cost* (saliency reflecting the amount of local shape change)

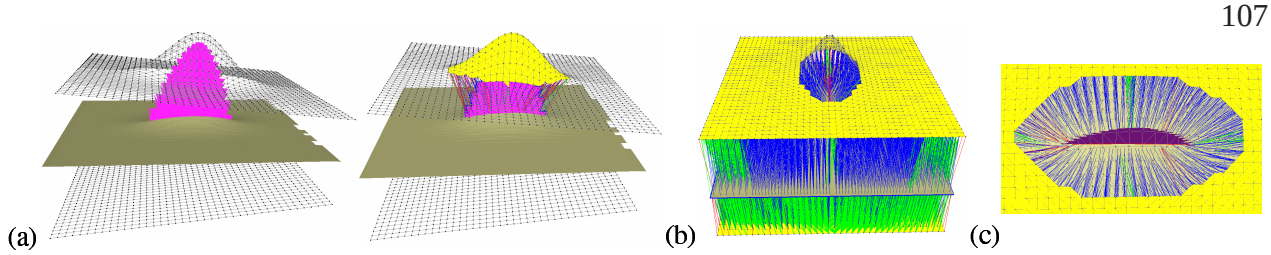


Figure 7.7: (a) Result of association of boundary points $\{G_i\}$ to a medial sheet corresponding to the bump of the 'deformed pair-of plane' shape in Figure 6.5. Color lines between the boundary points and the fine-scale $\mathcal{MS}$ elements depict such association: blue, green, red lines for assignment with A_1^2 sheet, A_1^3 curve, and A_1^4 vertex elements, respectively. (b) All other generators (except the bump) are associated with the other sheet. (c) Zoom in to the bump area to depict how the medial sheet in (a) occupies a volume and the complemental cavity of the other sheet.

and are performed in a 'best-first' greedy iterations. [13] After each transform is performed, all locally modified $\mathcal{MS}$ sheets, curves and nodes are re-evaluated and new candidate transforms are re-insert into $\mathbf{Q_t}$. The iteration stops when a pre-defined stopping condition (cost threshold T_c) is reached.

7.2.6 (Optional) smoothing A_3 rib curves

This step continues from § 7.2.2 to further smooth the A_3 rib curves. In contract to the element-wise pruning of A_1^2 sheet elements near the ribs, this section smooth the rib curves by modifying the geometry of each fine-scale element. An immediate consequence is that the "medial" property along the rib curves is *locally* invalid, *e.g.*, a rib curve element after modification is no long equal-distant to its three generators. However, globally the resulting rib curves are smoother and in fact more close to the true scenario. We discuss two smoothing techniques of different advantages, which can be used in combination.

(I) **Discrete Curve Shortening (DCS)**. Observe that the geometry of all A_1^2 sheet elements are *convex* polygons (Voronoi faces). The idea is to contract the elongated polygon by pulling the rib vertices inward. The *discrete curve shortening* (DCS) technique can be applied on the A_3 rib curve of each fine-scale face element on the side. The advantage is that *(a)* the smoothing will not violates the mesh topology (since the polygons are convex) and *(b)* the resulting A_1^2 sheet elements remain to be still *planar* polygons.

(II) **Gaussian smoothing**. Specifically, convolving a 3D Gaussian kernel to smooth the A_3 rib as a space curve. This approach can violate the local topology and the resulting A_1^2 sheet elements are no longer planar.

7.3 Maintaining a Consistent Boundary and $\mathcal{MS}$ Association

This section describes how to maintain a consistent association of the boundary (sample points) with the $\mathcal{MS}$ during the regularization process in § 7.2. A consistent association between the boundary shape and skeleton is important in and useful in various aspects: *(1)* estimating the $\mathcal{MS}$ transform

[13] The priority queue $\mathbf{Q_t}$ is implemented as a C++ multi-map to sort all candidate transforms according to their cost. Note that the entry of a *merge* transform requires storing two components (*e.g.*, a node to a curve), while other transforms (splice, contract, ...) requires only one component (*e.g.*, either a sheet or a curve). Refer to Table 8.1 for a summary of the main components involved in the $\mathcal{MS}$ transforms.

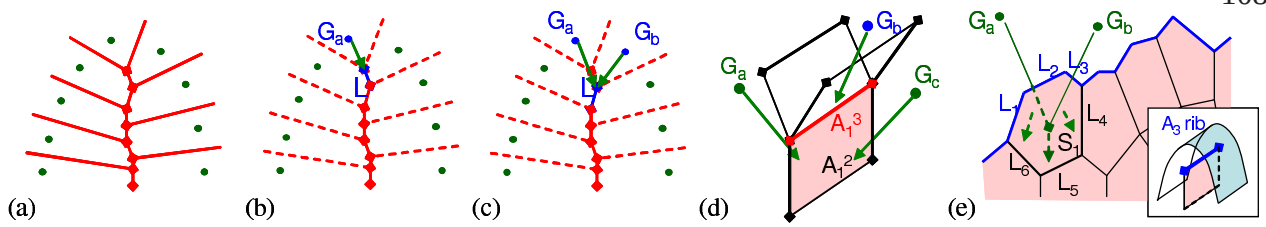


Figure 7.8: (a-c) The 2D case of maintaining a consistent generator-to-shock association during a *splice* transform (pruning of a shock branch [200]). (d-e) The 3D case near an A_3 rib curve.

cost, *(2)* estimating the corresponding shape portion for matching, *(3)* detecting salient surface features such as the ridge, *(4)* reconstruct shapes from the $\mathcal{MS}$ for morphing and animation.

Observe that initially the input boundary points are perfectly *dual* to the full $\mathcal{MS}$ (§ 3.3), similar to the duality of the Delaunay triangles ($\mathcal{DT}$) to the Voronoi diagram ($\mathcal{VD}$). Thus, a boundary mesh face is intuitively dual to an A_1^3 shock curve element, while a boundary mesh edge is dual to an A_1^2 shock sheet element. As we start to "transform" this dual structure, the duality is broken and need to be maintained consistently. For example, as the gap transform is performed in Figure 6.4 in § 6, the three generators of the triangle should be assigned to the two ending A_1^4 shock vertices. Similarly, as the splice transform is performed in Figure 7.8, the generators of the pruned shock should be *'passed'* to the relevant neighboring shocks.

* Avoid redundancy in associating the boundary elements to the $\mathcal{MS}$.

In order to couple a boundary shape to its $\mathcal{MS}$ tightly, the boundary mesh elements (faces, edges, and vertices) should be associated with the fine-scale $\mathcal{MS}$ elements (A_1^2 sheets, A_1^3 curves, and A_1^4 vertices) consistently throughout the regularization process. However, maintaining such multiple-to-multiple element association is redundant and should be avoided. The following thoughts motivates our approach:

1. *One way assignment.* Instead of maintaining a *two-way* mapping between the boundary elements and shock elements, use a *one-way assignment* to assign a boundary element to shock element(s).
2. *Assign only points:* We assign the boundary samples (generators) to the $\mathcal{MS}$ and explicit maintain such assignment in each transform. The association of boundary edges and faces to the shocks can be thus inferred from the point assignments.
3. *Avoid redundant assignment*: Observe that the association of a generator G to a shock elements E_0 implies associating G to E_0's incident higher-order shocks E_1, (since the wave from G reaches both E_0 and E_1 [125]). For example, if G is assigned to an A_1^2 sheet S, it should be automatically associated with S's incident A_1^3 curves and A_1^4 vertices. We thus *explicitly* assign *each generator to relevant* A_1^2, A_1^3, *and* A_1^4 *elements with the* **lowest** *singularity and without redundancy*. For example, in the case of associating G to an A_1^3 shock curve element L, if G is already assigned to any of L's incident A_1^2 sheets, we skip the explicit assignment of G to L, since G can always be accessible from L's incident shock sheets.

Table 7.1 summarize the possible boundary to shock associations. We only maintain the assignment of the generators (points) to the shocks (A_1^2 faces, A_1^3 edges, and A_1^4 vertices), which is implemented as follows. Each shocks element stores a set of points (pointing to the assigned generators). The *association* of the boundary to any particular shock face/edge/vertex can be obtained by querying all incident lower-order shocks for their assigned generators.

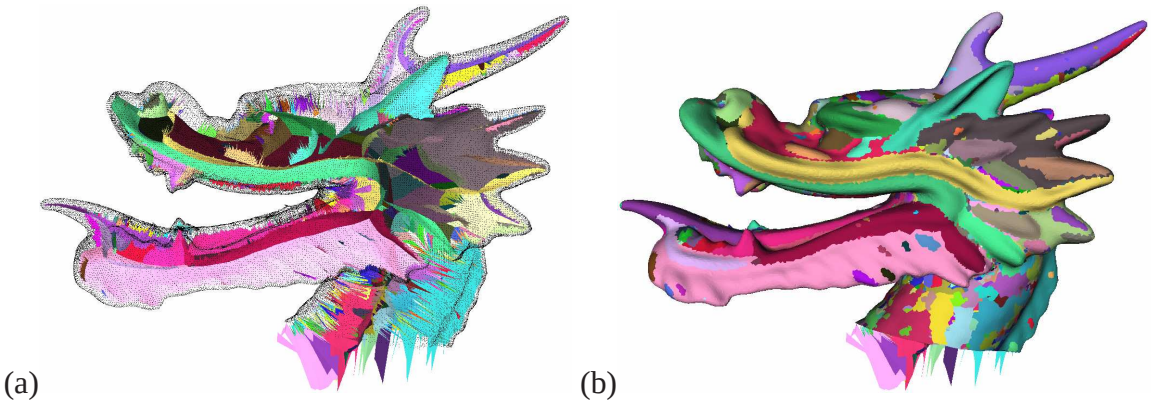


Figure 7.9: (a) The $\mathcal{MS}$ computed from $127,200$ points sampling the Stanford Dragon head ($254,291$ triangles). (b) The tightly-coupled surface regions (in colors) corresponding to the $\mathcal{MS}$ give an initial segmentation of the shape, which identifies salient features such as the *ridges* and flat regions and is suitable for further modeling use.

Figure 7.9 shows a result of our approach in coupling a shape boundary with the $\mathcal{MS}$ throughout the regularization process. Observe how well the fine shape details (ridges and high curvature regions) are captured with the medial branches. Refer to Figures 7.10, 7.12, 7.14 and others in § 7.4 for more results.

We address some remarks and future developments in the footnote. [14]

Table 7.1: Possible type of boundary-shock associations.

type	dual	possible association
boundary point		A_1^2 *face*, A_1^3 *edge*, A_1^4 *vertex*
boundary edge	A_1^2 face	A_1^3 edge, A_1^4 vertex
boundary face	A_1^3 edge	A_1^4 vertex

7.4 Implementation and Experimental Results

This section describes the implementation of the proposed $\mathcal{MS}$ computation and regularization scheme and shows a few results.

* Features of our implementation and its parameters.

The implementation of the proposed system dates back to Leymarie's initial work [128] and migrates to reflect the advance of modern computing technology in several fronts. The current implementation contains two key features: *(i)* First, we exploit the open-source, cross-platform computer vision library, the **VXL** (http://vxl.sourceforge.net/) in the C++ programming language to build our system;

[14]Remarks: *(I)* In the splice transforms, we pass the generators to *all* incident shocks. A better solution is to pass the generators according to the shock *flow* on the A_1^2 sheets (future work). *(II)* Instead of passing generators in the transforms, another approach is to ignore all generator passing and *recover* the generators by *backtracking*. This is possible in 2D (the shock is a directed 'graph') but is problematic in 3D with the $\mathcal{MS}$ (future work of the $\mathcal{SC}$ 'hypergraph'), Figures 7.8. *(III)* Refer to the related work of Shamir and Shaham's *pair-mesh* [172] reviewed in Chapter 2, where a different way of associating the boundary mesh to its Voronoi skeleton is proposed.

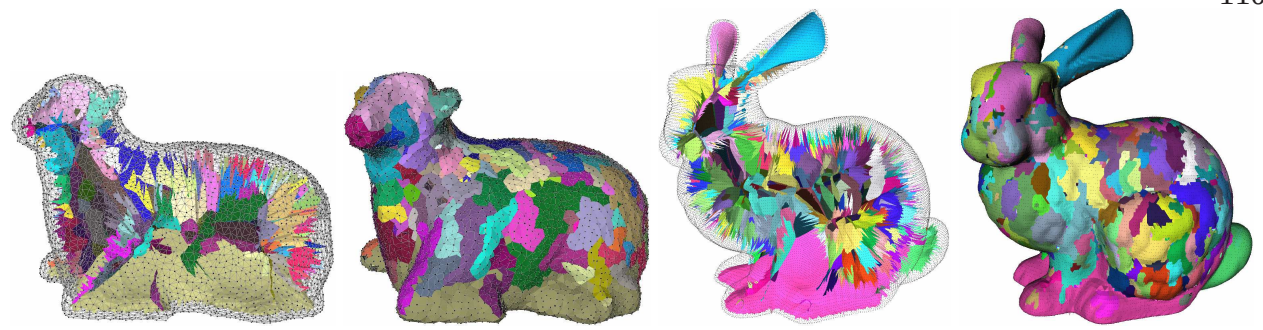


Figure 7.10: The association of the boundary mesh (patches) to the $\mathcal{MS}$ of a toy sheep object and the Stanford bunny after the *splice regularization* of their $\mathcal{MS}$. The surface patches and their corresponding sheets are randomly-colored. Note that the $\mathcal{MS}$ are still noisy at this stage, thus the colors resemble a 'camouflage' coloring of the shapes.)

and we follow the generic programming paradigm, *i.e.* using the C++ Standard Template Library (STL) to design each C++ class in coding. We exploit the **Coin3D** (http://www.coin3d.org/) 3D graphics library, which is also open-source and cross-platform, for the visualization and handing the graphical user interface (GUI) of the system. *(ii)* Second, we completely separate the algorithmic computation from the visualization, such that the code can be executed in two modes: *(a)* in an optimized (command-line) application allowing a batch process, or *(b)* in a GUI window-based application allowing a visual interaction or debugging.

Our system takes unorganized point-sampled shapes as input and compute the regularized $\mathcal{MS}$ automatically, where the default system parameters are pre-determined from field tests and can be manually adjusted. We list two important parameters, which can be fine-tuned manually to produce a better resulting $\mathcal{MS}$:

- The ratio η which controls the maximum size of surface interpolants (triangle sides produced by *gap* transforms) described in § 6.2 (default value 10).
- The threshold R_t of minimum radius which controls how close an A_3 rib curve can approach the shape surface (in applying *splice* transforms), described in § 7.2.2 (default value 1).

Both parameters are determined automatically (if not specified), which work fairly well for many datasets. Since the program has no control on the unorganized input points and do not impose other assumptions, setting improper values could cause poor surface meshing results and noisy resulting $\mathcal{MS}$ hypergraphs. [15]

We have extensively tested the proposed computational scheme on a large variety of 3D dataset, including artificial shapes to simulate all transitions (Figures 5.2 and 7.4), general shapes with salient structure and local details (bunny (Figure 5.13), dragon (Figure 7.11), dinosaur (Figure 7.14(a-c)), pot fragments (Figures 10.6 and 10.7)), shapes for industrial applications (fan disk (Figure 4.1) and rocker arm (Figure 7.14(e-f)) from Cyberware), and medical applications (hand (Figure 1.5), carpal bones (Figure 9.9)). Our system handles up to $300k$ points of input (the bottleneck is the large initial *full* fine-scale $\mathcal{MS}$ limited by 2GB of computer memory). The regularized $\mathcal{MS}$ is computed from a few seconds to a few minutes.

[15] General guidelines in setting η and R_t: Small η could cause the reconstructed surface mesh to contain (unwanted) holes and artifacts and could cause the interior and exterior portions of the $\mathcal{MS}$ to connect together. Large η could cause (unwanted) large surface interpolants to form. Small R_t could produce (unwanted) noisy $\mathcal{MS}$ branches, which also could cause the interior and exterior portions of the $\mathcal{MS}$ to connect together. Large R_t could over-prune the $\mathcal{MS}$, which could drop salient surface features (ridges, corners) captured with the A_3 ribs of the $\mathcal{MS}$.

* Experiments on degenerate cases and on degrading the sampling condition.

We have also tested our system on a most degenerate case—the perfect sphere, which in theory should yield a single A_∞ (or A_1^∞) point as the $\mathcal{MA}$, Figure 7.15, where the $\mathcal{MS}$ is regularized to the ultimate case of a single medial sheet topologically equivalent to a disk. (We can further shrink this disk into a single point at the center of the sphere.)

On testing our system on low-sampling case, Figure 7.13 shows three examples where the sampling conditions are extremely low (only around 1.5K sample points for a dog, a head, and a twisted knot shape). Our system successfully extracts the interior $\mathcal{MS}$ in the above cases, demonstrates its superior capability in processing most practical dataset. For example, see Figure 10.3 for further use in the medical domain.

Finally, we experiment on the robustness of our $\mathcal{MS}$ computational scheme *w.r.t.* the sampling condition. Specifically, we reduce the sample points of a well-sampled model (the scan of Michelangelo's David from Stanford) step-by-step, and observe how its $\mathcal{MS}$ degrades. This experiment also simulates the reverse process of dynamically adding of sample points. The result in Figure 7.16 shows that the $\mathcal{MS}$ degrades slowly as we remove 50%, 75%, 87.5% sample points, and it only starts to break down only until very insufficient 3% to less than 1% of sample points remain in the last few tests.

* Comparison of results with other approaches.

In comparing to existing Voronoi refinement results, ours exhibits three advantages: *(i)* The $\mathcal{MA}$ is better regularized on both the boundary and the interior topology, *(ii)* The $\mathcal{MS}$ (rib curves) are closer to the object surface, indicating that our regularization is better in capturing finer details, even in the case of sparsely-sampled or ill-sampled inputs (for example see the dinosaur's hands in Figure 7.14(a)). *(iii)* All our results are obtained from the sole assumption of unorganized points with reasonable sampling (and nothing further). We can handle shapes with boundary where there is no distinguishing between the interior/exterior $\mathcal{MA}$.

* Summary of our advantages (contributions) in modeling the $\mathcal{MA}$.

The proposed framework of $\mathcal{MS}$ transforms models generic shape perturbations and deformations in terms of: (i) ensuring the *robustness* in simplifying the hypergraph topology against small perturbations. (ii) ensuring *generality*: the $\mathcal{MA}$ is extracted in difficult cases (comparing to other main stream methods surveyed in § 2.3).

- *Capturing fine shape details.* Regularized A_3 ribs close to surface to capture fine shape details, even in relatively low samplings Specifically, the cases of very low samples and non-solid surfaces (with boundary) are handles. In addition, both scanned and degenerate (man-made graphics) objects are handled.

- *Simplified $\mathcal{MA}$/$\mathcal{MS}$ topology.* The inter-connectivity between medial sheets is greatly simplified. Once a medial scaffold has been regularized, and thus greatly simplified by removing the adverse effects of smaller features and perturbations, we can use it for registration by matching the scaffold structure as described next.

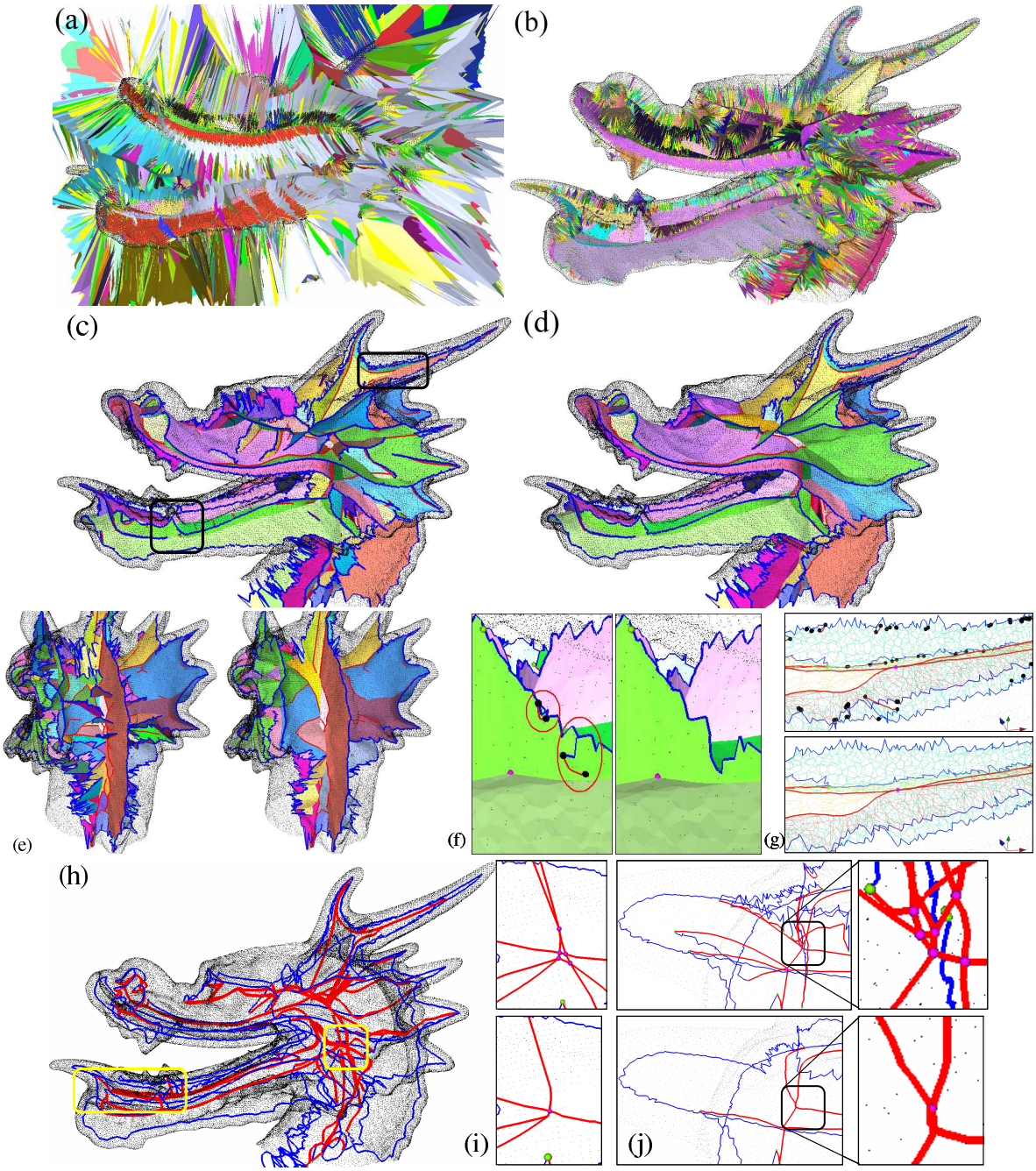


Figure 7.11: Regularizing the $\mathcal{MS}$ of a complex shape—the Stanford dragon head (127k points). (a) Both the interior and exterior $\mathcal{MS}$ (74k sheets) after *shock segregation* are extremely noisy. The interior $\mathcal{MS}$ (35k sheets) in (b) is simplified in a first stage, *splice regularization*, into 285 sheets, and the coarse-scale hypergraph ($1,695$ curves, $1,446$ nodes) is built as in (c). The $\mathcal{MS}$ hypergraph is then further regularized by a second stage involving all transforms, ending up with only 76 **sheets,** 262 **curves,** 219 **nodes** in (d). (e) shows the effect of *splice* transforms to remove and merge several sheets. (f-g) shows numerous A_5 swallow-tails removed by the *contract* transforms: near the tooth (f) and the horn (g). (h) shows the $\mathcal{MS}$ graph of (d) with sheets implicit to better visualize its interior structure. (i-j) depicts the effect of *contract* and *merge* transforms in simplifying the interior hypergraph topology: near the neck (i) and tongue (j). [43]

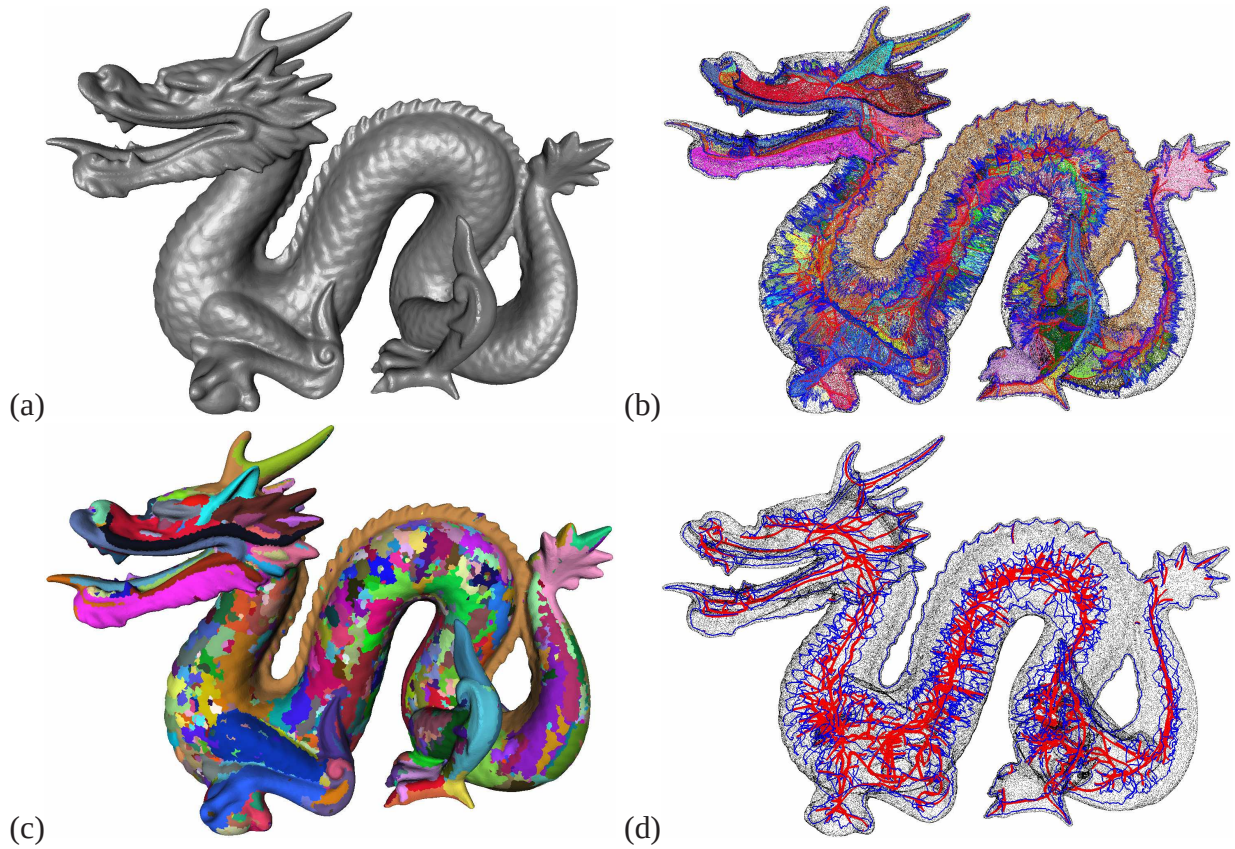


Figure 7.12: Regularizing the $\mathcal{MS}$ of the Stanford dragon dataset (250K points). (a) shows the reconstructed surface mesh. (b) is the $\mathcal{MS}$ hypergraph. Notice how the A_3 rib curves of the $\mathcal{MS}$ approaching the shape to capture salient features at the sharp regions. (c) is the coupled surface regions of the $\mathcal{MS}$, where corresponding sheets are shown in the same colors. (d) shows the $\mathcal{MS}$ graph to better illustrates its structure.

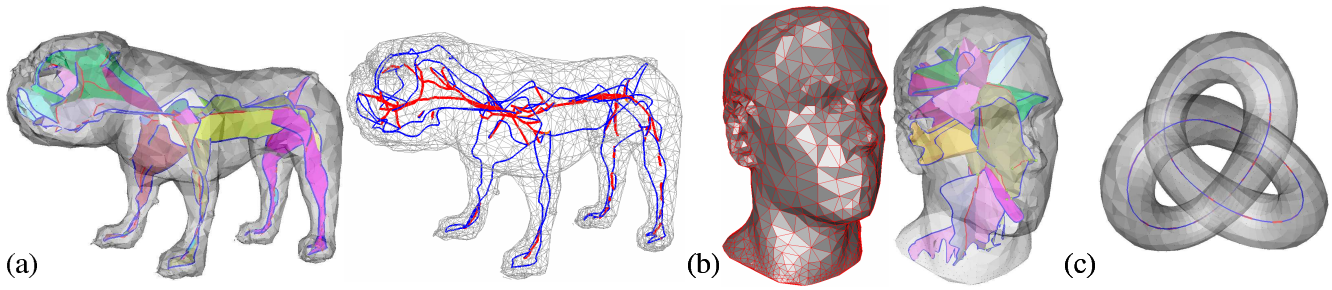


Figure 7.13: $\mathcal{MS}$ of dataset with low sampling. (a) The Bulldog dataset (from ETH Zurich) with $1,783$ points, re-meshed surface with $3.526 faces$; the regularized $\mathcal{MS}$ contains 76 sheets. (b) A head model sampled with $1,368$ points, re-meshed surface with $2,695$ faces; the regularized $\mathcal{MS}$ with 35 sheets still captures many features (*e.g.*, eyes, nose, ears, hair protrusions). (c) The $\mathcal{MS}$ of the low-sampling knot of $1,440$ points of Figure 6.23(d).

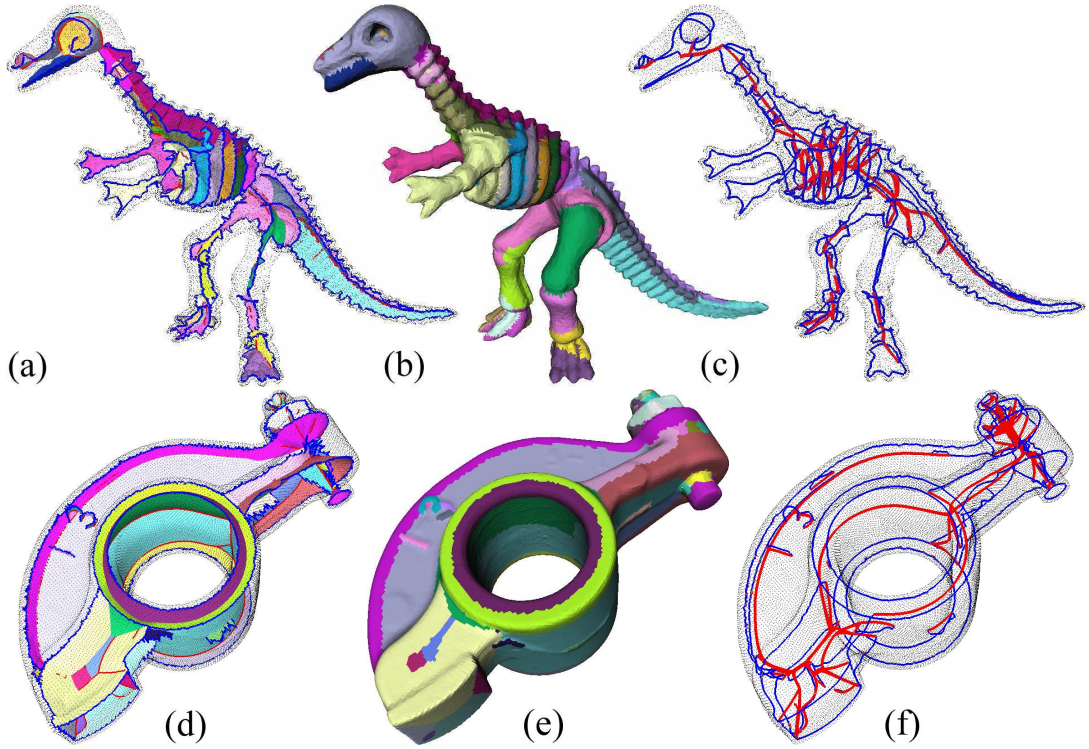


Figure 7.14: The regularized $\mathcal{MS}$ of (a-c) a toy dinosaur model ($14,050$ points) and (d-f) a rocker arm ($40,177$ points), data from Cyberware. Both $\mathcal{MS}$ after regularization appear to be neat in structure and are close to the object surface to capture fine details. The tightly-coupled surface regions of the $\mathcal{MS}$ in (b,e) are suitable for further shape modeling and segmentation use. [43]

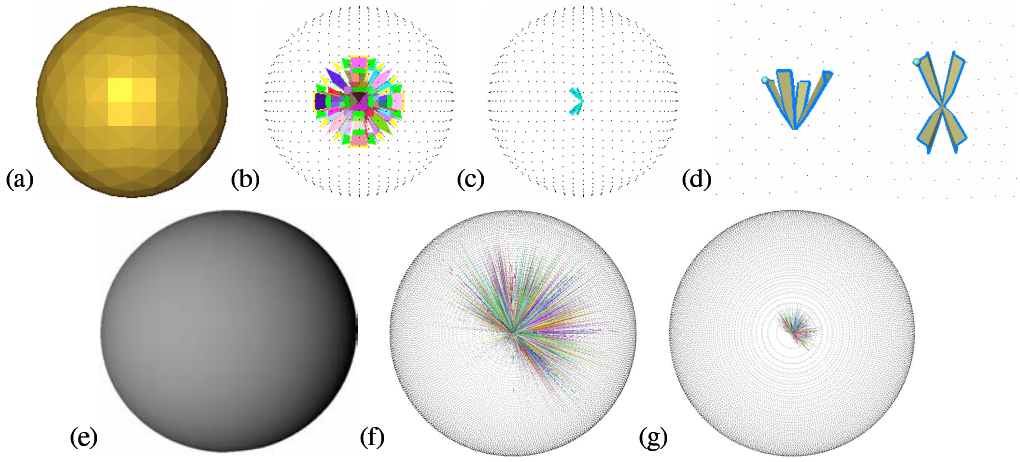


Figure 7.15: Degeneracy test of the $\mathcal{MS}$ computation on the perfect spheres. (a) The water-tight surface mesh ($2,048$ faces) built from $1,026$ sampling points on a sphere from the *shock segregation* process. The remaining $4,656$ medial sheets are shown in (b). (c) An ultimate regularization simplifies the $\mathcal{MS}$ to become a single medial sheet. (d) Zoom in to the remaining sheet in (c) in two views to show its degenerate single 'loop' boundary $\mathcal{MS}$ curve enclosing this sheet. (e) Sphere surface of $53,827$ faces reconstructed from $26,927$ points sample points. (f) The remaining $135,425$ sharp "spike-like" medial tabs are due to the high degeneracy and dense sampling. (g) is the $\mathcal{MS}$ of $1,679$ sheets after regularization. In this experiment. Further applying the transforms should yield a result similar to (d).

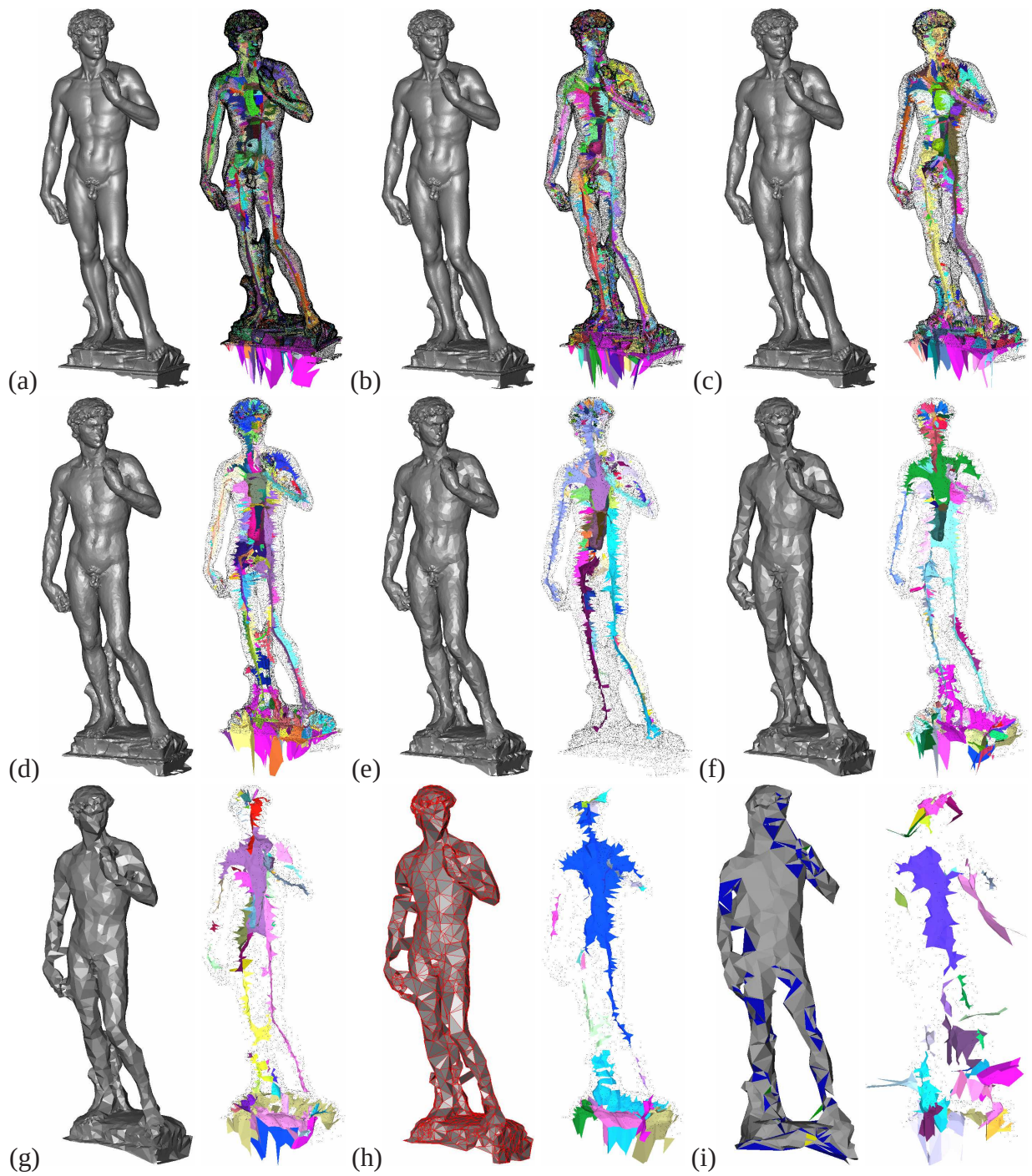


Figure 7.16: The Michelangelo's David dataset (from Stanford) in (a) is repeatedly sub-sampled by randomly remove sample points to the half, *i.e.*, into (b) 50%, (c) 25%, (d) 12.5%, (e) 6.25%, (f) 3.125%, (g) 1.56%, (h) 0.78%, and finally (i) 0.39%, and their $\mathcal{MS}$ hypergraphs are computed for each case. The $\mathcal{MS}$ degrades slowly and starts to break down only until very insufficient sample points are available in the last few tests.

Chapter 8

On Edit-Distance Matching of the Medial Scaffolds

* Overview of chapter: on edit-distance matching of the $\mathcal{MS}$ to measure 3D shape similarities.

The measuring of similarity between shapes has been a central task toward the ultimate goal of recognition in computer vision. In this chapter we discuss a theoretical framework extended from our successful experience in matching 2D shapes [169], to measure 3D shape *"dissimilarity"* (or distance) as the *optimal* deformation between shapes represented by the medial scaffold ($\mathcal{MS}$). This framework is a direct extension of Sebastian and Kimia *et al.*'s approach [169] who *edit* the 2D *shock graphs*($\mathcal{SG}$) to match 2D shapes. This is done by exploiting the $\mathcal{MA}$ *transitions* to characterize the shape deformation path, a key idea which has been reviewed in §1.1. In 3D, the $\mathcal{MS}$ hypergraph is the analogy and extension of the 2D shock graph ($\mathcal{SG}$). While this optimal similarity is viable in theory, several difficulties arise from the additional dimensionality in 3D (which will be elaborated in § 8.1), we thus look for a *sub-optimal* solution that is computationally practical, which leads to the development of the graph matching scheme in Chapter 9 in matching the $\mathcal{MS}$ hypergraphs.

A second contribution of this chapter is on deriving a theoretical support on a better definition of the cost (saliency measure) of the set of $\mathcal{MS}$ transforms described in Chapter 5. A key idea is that while the aforementioned optimal deformation between shapes is *global* (thus is more difficult to compute), the deformation between pairs of medial sheets or curves is *local* thus can be computed optimally. We derive the optimal **deformation cost** to match *three* corresponding pair of the $\mathcal{MS}$ hypergraph components: *(i)* matching a pair of A_1^2 sheets (§ 8.2), *(ii)* matching a pair of A_1^3 axial curves (§ 8.3), and *(iii)* matching a pair of A_3 rib curves (§ 8.4). The dissimilarity (distance or cost) in each case is then defined to reflect the *elastic* deformation to measure the optimal similarity between the corresponding portions of shapes. We finally show that these pairwise similarities can be used to define the costs of all $\mathcal{MS}$ transforms in reflecting the amount of shape change in a more consistent way.

* Organization of chapter.

This chapter is organized as follows. Section 8.1 discusses the extension of the 2D shock graph edit-distance matching approach [169] to match the $\mathcal{MS}$ hypergraph in 3D and addresses main difficulties to motivate a sub-optimal solution. Specifically, the sub-optimal matching is to replace the (global) structural matching of the hypergraph with a graph-matching solution and meanwhile keep the (local) pairwise curve-to-curve and sheet-to-sheet matching optimal. Sections 8.2, 8.3, and 8.4 then continue to define the pairwise deformation cost for the A_1^2 sheets, A_1^3 curves, and A_3 curves, respectively, in terms of a *joint boundary-shock elastic deformation*. Finally, Section 8.5

applies the above deformation costs to define the $\mathcal{MS}$ transform costs to reflect the corresponding shape changes consistently.

8.1 Main Difficulties of the $\mathcal{MS}$ Hypergraphs Edit-Distance Matching and a Sub-optimal Solution

* Extend the 2D edit-distance matching to measure similarities as optimal deformation.

We first recall the main ideas in using the $\mathcal{MA}$ as shape representation to match shapes in § 1.1 from Sebastian and Kimia *et al.* [169] to relate shapes with the minimum-cost path in the shape space. The goal is to extend the idea to 3D shapes to measure shape *dissimilarity* (distance) as the *optimal* deformation between them. Recall that the $\mathcal{MS}$ is a *complete* representation [86, 161] allowing a complete reconstruction of the shape, thus is qualified to be used as a shape representation here. [1]

The space of all possible deformations between two arbitrary shapes is huge, therefore several steps are adopted to make the search (of the optimal path) practical, in mimicking the idea of Sebastian and Kimia *et al.* [169]: *(i)* First, the $\mathcal{MA}$ instabilities are modeled as transitions, and we explicitly use the transitions to define an equivalence class (shape cell) for shapes with the same $\mathcal{MS}$ hypergraph topology. The collection of shape cells essentially partitions the shape space (Figure 1.3(a)). *(ii)* Second, the deformation path between two shapes could be arbitrary and there exists infinite many such paths. We define an equivalent class for the deformation paths by grouping the path passing through the same sequence of shape cells into a "deformation path bundle" (Figure 1.3(b)). *(iii)* Third, we avoid complexity increasing deformations by decomposing the optimal deformation into a pair of simplifying deformation paths (Figure 1.3(c)).

The above steps tremendously reduce the search requirement, in that the deformations are now *discretely* characterized by the $\mathcal{MS}$ topology, and the "edits" between topologies (across *transitions*) can be explicitly handled by the $\mathcal{MS}$ *transforms* described in Chapter 5. It is thus possible to explore all $\mathcal{MS}$ hypergraph edits to find the optimal sequence which guides the deformation. Observe in Figure 8.1 for a schematic edit-distance exploration in matching two 2D shapes: the space to explore in edit-distance matching of the $\mathcal{MS}$ hypergraphs is still *huge*.

* Difficulties of the optimal edit-distance matching of the $\mathcal{MS}$ hypergraphs.

While the above optimal scheme is theoretically viable (although complex), several barriers hamper its practical implementation:

1. *Additional dimensionality in matching the hypergraphs.* As described in Chapter 4, the $\mathcal{MS}$ hypergraph contains far more complex topology than the 2D case of a shock graph, which

[1] **Review of shape reconstruction from the $\mathcal{MA}$.** In [86], Giblin & Kimia show how a 2D shape can be reconstructed from the shock graph. Let $\gamma(s)$ denote a first-order A_1^2 shock curve, with arclength s, radius r (or time of formation) and instantaneous velocity v. The pair $(\kappa(s), a(s))$ and the length of the branch L are sufficient to intrinsically reconstruct the corresponding pair of boundaries $\gamma^{\pm}(s)$ of the shape and their differential properties. Specifically, the boundary points $\gamma^{\pm}(s)$ are given by $\gamma^{\pm}(s) = \gamma(s) - \frac{r}{v}\vec{T} \pm \frac{r}{v}\sqrt{v^2-1}\vec{N}$. where $\vec{T}$ and $\vec{N}$ are the tangent and normal to the curve γ, respectively, where v and r are obtained by integrating a, and γ is obtained by integrating κ. This local intrinsic reconstruction is essential in computing the similarity between two shock segments. The same analogy expands to 3D, in [161], points on the two sides of 3D shape (boundary) can be reconstructed by $\gamma^{\pm} = \gamma - r\vec{N}^{\pm}$, where $\vec{N}^{\pm}$ is now the unit normal to the boundary surface, oriented towards the center of the bi-tangent sphere, and $\vec{N}^{\pm} = -\cos\phi\vec{T} \mp \sin\phi\vec{N}$, and $\cos\phi = -\frac{1}{v}$. In summary, the $\mathcal{MS}$ with a radius field on the A_1^2 sheets is sufficient to reconstruct the 3D shape. Additional consistency condition at other types of $\mathcal{MS}$ points, namely A_3, A_1^3, A_1A_3, A_1^4 is further addressed in [161].

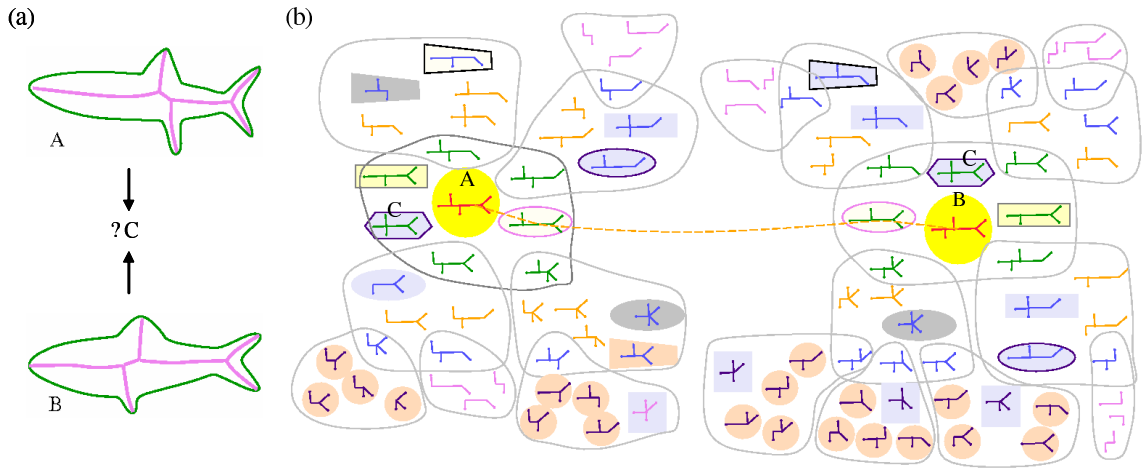


Figure 8.1: (From [169, Fig.14].) (a) The optimal deformation between two shapes is obtained by searching all pairs of simplifying deformation paths leading to a common shape C. (b) A hand-drawn sketch of how the space of one-parameter family of deformations for each shape can be discretized. The optimal path goes through the shape in the grey-hashed hexagon. [169]

makes the matching of them difficult. For a 2D closed contour, the shock graph of the interior is a planar *acyclic tree* (*i.e.*, with no loops); while in 3D, even for a closed surface boundary, the reduced form of a $\mathcal{MS}$ graph ($\mathcal{MS}^{\mathcal{G}}$) is typically non-planar and contains loops (which are the boundaries of the $\mathcal{MA}$ sheets). Furthermore, it is often the case in 3D that no closed surface boundary is available (Chapter 6), hence without a notion of the interior/exterior $\mathcal{MA}$. The matching of acyclic trees in 2D is a crucial assumption in [169] to exploit a *dynamic programming* algorithm to efficiently explore the edit-distance between shapes. The extension to 3D to match a "tree-like" organization of the $\mathcal{MS}$ hypergraphs is non-trivial and remains unexplored.

2. *Additional dimensionality in matching the medial sheet surfaces.* Even without the above issue, the additional dimensionality in matching the medial sheets (2D surfaces) is still challenging. (However this can be approximated, detailed in § 8.2).
3. *Additional dimensionality in the $\mathcal{MA}$ transitions and transforms.* In compared to the 2D approach [169] where only *two* $\mathcal{MA}$ transforms is implemented (splice and contract) on the shock graph, in 3D there are *seven* generic transitions thus requires *eleven* $\mathcal{MS}$ transforms plus others (Chapter 5) with a more complex scenario in the case of hypergraphs.
4. *The 3D shock flow analysis is incomplete.* The generalization of a (directed) shock graph in 3D with a complete radius *flow* analysis of the $\mathcal{MA}$ is a *shock scaffold* ($\mathcal{SC}$), which is a further refined hypergraph of the $\mathcal{MS}$ that needs further investigations (§ 3.4). Also, the additional *transitions* pertinent to the $\mathcal{SC}$ flow changes remain to be explored.

With the above difficulties in mind, we thus try to simplify the problem and seek for approximated solutions.

* Sub-optimal solution by adopting a graph matching approach.

Several ideas motivate our sub-optimal solution in matching the $\mathcal{MS}$ hypergraphs:

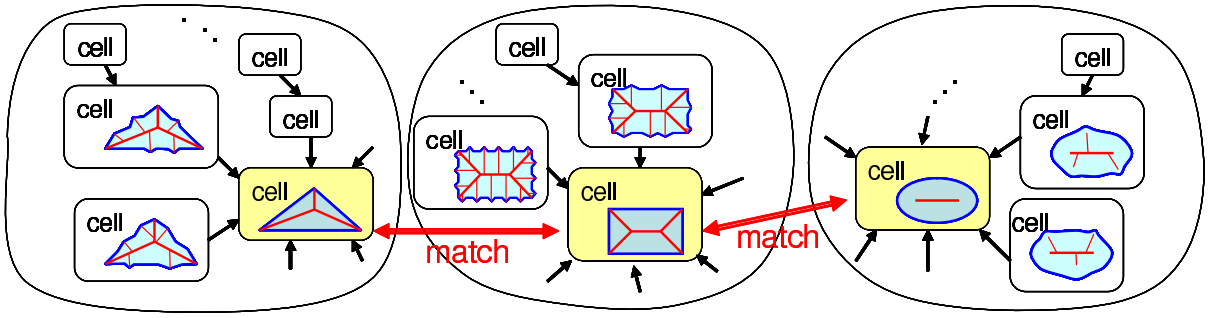


Figure 8.2: Recall Figure 1.9 for our shape regularization scheme by simplifying shapes toward higher-order of symmetry. This Figure is a re-interpretation of it to illustrate our sub-optimal matching scheme that shapes in each shape cells are essentially first simplified toward the *representative* shapes (yellow) in each category and then matched by a graph matching approach.

1. Since the hierarchical $\mathcal{MS}$ representation allows us to reduce the hypergraph into a graph structure (the $\mathcal{MS}^{\mathcal{H}-}$ and $\mathcal{MS}^{\mathcal{G}}$ in Chapter 3), we can adopt the popular graph-matching approaches as the main matching scheme (Refer to § 2.4 for a survey).
2. In the above context of matching the $\mathcal{MS}$ by matching their hypergraph components (sheets, curves, and nodes), one important idea is that such component-wise matching can still be computed optimally, similar to what has been done in the 2D case [169] with a few extensions. This will be the main topic to elaborate in § 8.2, § 8.3, and § 8.4.
3. *Simplification in the search (of optimal matching).* The second idea pertinent to the adoption of a graph matching scheme is that instead of exploring the full edit-space in matching the $\mathcal{MS}$ hypergraphs, we can first regularize (simplify) the $\mathcal{MS}$ (as in Chapter 7) in prior to the graph matching, which effectively reduces the $\mathcal{MS}$ instability and complexity. This essentially decomposes an optimal deformation search into the two paths of sub-optimal deformations, namely, the *suboptimal graph matching* and the *(greedy) regularization*, depicted in Figure 8.2. This *sub-optimal* approach is essentially a simplified and approximated version of the more complex optimal approach. Specifically in Figure 8.3, while the optimal edit-distance explores all possible deformation paths, the sub-optimal approach simply chooses the lowest-cost deformation at each regularization step, and relies on the graph-matching to produce the final similarity.
4. *Intractability of graph matching—look for a practical solution.* With the above approximation/simplification in mind, the matching of two graph structures is still an **NP**-hard computational problem [95], which has lead to the development of several practical solutions (§ 2.4). We will elaborate our solution in Chapter 9 to extend a robust graph matching scheme to match the $\mathcal{MS}$ hypergraphs.

Below we relate the sub-optimal (graph matching) approach with the optimal edit-distance approach in terms of their resulting distance metrics.

* Triangular inequality to bound the sub-optimal distance metric.

In [169], the edit-distance between shapes reflecting their optimal deformation is shown to be a **metric** satisfying the *triangular inequality*, which can be used to bound the above sub-optimal distance measure. Specifically, in Figure 8.3, the error between matching original shapes $(\gamma_0, \bar{\gamma}_0)$ and

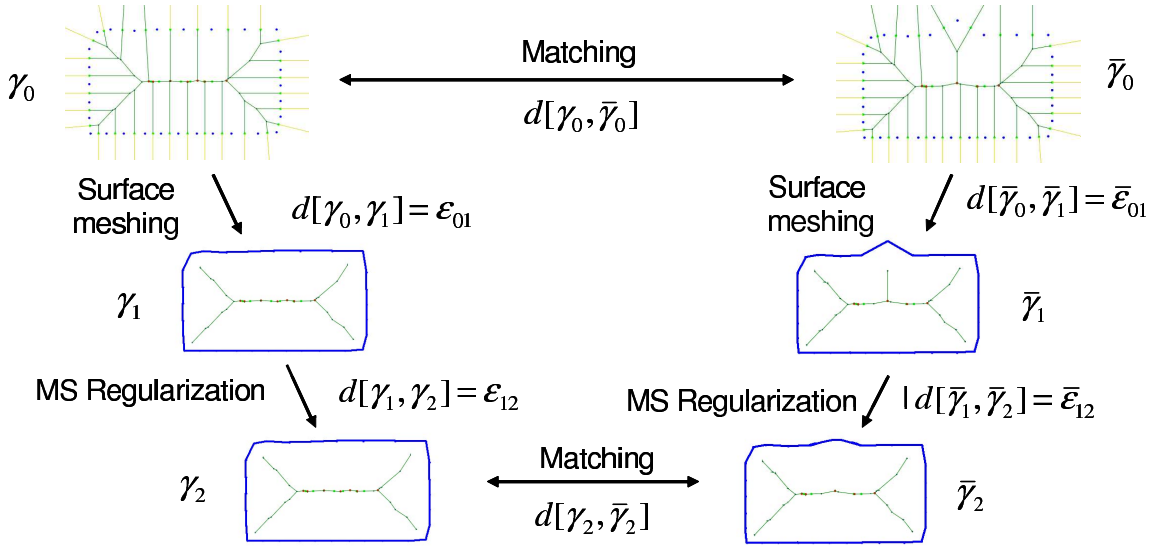


Figure 8.3: The error between matching the original shapes $(\gamma_0, \bar{\gamma}_0)$ and the regularized shapes $(\gamma_2, \bar{\gamma}_2)$ is bounded by Eq.(8.1) and approximated by Eq.(8.2).

regularized shapes $(\gamma_2, \bar{\gamma}_2)$ is bounded by:

$$d[\gamma_0, \bar{\gamma}_0] - d[\gamma_2, \bar{\gamma}_2] \leq (\varepsilon_{01} + \bar{\varepsilon}_{01} + \varepsilon_{12} + \bar{\varepsilon}_{12}), \tag{8.1}$$

where ε_{01}, $\bar{\varepsilon}_{01}$, ε_{12}, and $\bar{\varepsilon}_{12}$ are the distances between shapes before/after regularization. [2]

If we assume that *(i)* the greedy regularization is optimal toward simplification (*i.e.*, ε_{01}, $\bar{\varepsilon}_{01}$, ε_{12}, and $\bar{\varepsilon}_{12}$ belongs to the optimal deformation path between γ_0 and $\bar{\gamma}_0$), and *(ii)* the matching of $(\gamma_2, \bar{\gamma}_2)$ is also optimal, the (unknown) distance between the original shapes $d[\gamma_0, \bar{\gamma}_0]$ can be approximated by

$$d[\gamma_0, \bar{\gamma}_0] \cong d[\gamma_2, \bar{\gamma}_2] + (\varepsilon_{01} + \bar{\varepsilon}_{01} + \varepsilon_{12} + \bar{\varepsilon}_{12}), \tag{8.2}$$

* Sub-optimal matching: decomposing the shape deformation path into two types of deformations.

The above sub-optimal matching solution (regularization + graph matching) in Figure 8.3 essentially decomposes the optimal deformation path between two shapes $(\gamma_0, \bar{\gamma}_0)$ into two types of deformations, namely: *(i)* the $\mathcal{MS}$ transform edits in the regularization steps (the vertical arrows in Figure 8.3), and *(ii)* the fixed $\mathcal{MS}$ topology deformation explored by the graph matching step (the horizontal arrow of $d[\gamma_2, \bar{\gamma}_2]$ in the last row in Figure 8.3). To ensures the above two types of measures are *consistent* under the same metric, we adopt a similar approach to [169] to first derive the *deformation cost* between any arbitrary pair of medial sheets or curves, by integrating the elemental shape changes in an intrinsic manner. Specifically, the pairwise similarity between the A_1^3 andA_3 curves can be obtained by matching the *optimal* elastic deformation similar as done in 2D [169], while the pairwise similarity between the A_1^2 medial sheets is more difficult (due to the additional dimensionality), detailed below. We then define the $\mathcal{MS}$ transform costs as the *limit* cases of the above results in § 8.5.

[2] A proof is straight-forward by repeatedly applying the triangular inequality to bound the difference of distances. $d(\gamma_2, \bar{\gamma}_2) + \varepsilon_{12} \geq d(\gamma_1, \bar{\gamma}_2)$, $d(\gamma_1, \bar{\gamma}_2) + \bar{\varepsilon}_{12} \geq d(\gamma_1, \bar{\gamma}_1)$, $d(\gamma_1, \bar{\gamma}_1) + \varepsilon_{01} \geq d(\gamma_0, \bar{\gamma}_1)$, $d(\gamma_0, \bar{\gamma}_1) + \bar{\varepsilon}_{01} \geq d(\gamma_0, \bar{\gamma}_0)$. Putting them together gives $d(\gamma_2, \bar{\gamma}_2) + (\varepsilon_{12} + \bar{\varepsilon}_{12} + \varepsilon_{01} + \bar{\varepsilon}_{01}) \geq d(\gamma_0, \bar{\gamma}_0)$.

8.2 Deformation Cost to Match a Pair of A_1^2 Medial Sheets

In this section, we define the deformation cost to match the shapes of a pair of A_1^2 sheets by summing up the elastic deformation between the shapes and the A_1^2 sheets in an optimal sense. This can be viewed as a direct extension of matching a pair of shock branches in matching the shock graph in [169, Eq.5].

* Theoretical formulation: integration of infinitesimal shape changes to define the A_1^2 deformation cost.

The local configuration in matching infinitesimal A_1^2 sheets $da \in S$ to $d\bar{a} \in \bar{S}$ is depicted in Figure 8.4a, where S and $\bar{S}$ are parameterized by (ξ, η) and $(\bar{\xi}, \bar{\eta})$ respectively. [3] Let the boundary surfaces of da to be da^+ and da^-, parameterized by (ξ^+, η^+) and (ξ^-, η^-) respectively; the boundaries of $d\bar{a}$ are similarly defined. In order to compare the similarity between S and $\bar{S}$, a notion of "assignment" to match the infinitesimal elements da and $d\bar{a}$ is required. We extend the notion of an *alignment curve* α in [168] to match two curves in 2D to an **alignment surface** α_s to specify such assignment between da and $d\bar{a}$. The **dissimilarity** $d[S, \bar{S}]$ is defined as the *optimal* deformation cost between S and $\bar{S}$ in optimizing the alignment $\alpha_s(da, d\bar{a})$. We parameterize α_s by infinitesimal elements $d\tilde{a}$ and formulate $d[S, \bar{S}]$ as an optimization problem of a *joint boundary-shock deformation and matching* in optimizing the deformation cost $\mu[S, \bar{S}, \alpha_s]$. Specifically, given an alignment $\alpha_s(d\tilde{a})$ mapping each da to $d\bar{a}$, $\forall da \in S$, the overall **deformation cost** $\mu[S, \bar{S}, \alpha_s]$ is the integration of the element-wise cost over $d\tilde{a}\,(da, d\bar{a})$. [4] The (optimal) dissimilarity $d[S, \bar{S}]$ is the minimum deformation cost when α_s is optimal:

$$d[S, \bar{S}] = \min_{\alpha_s} \quad \mu[S, \bar{S}, \alpha_s]. \tag{8.3}$$

The optimization of α_s in the above cost definition is similar to the integral of cost in, *e.g.* a simulation of continuum mechanics in [78]. The optimization can be done by an extension of a dynamic programming technique over the alignment curve α in matching 2D curves in [168] to the α_s with additional dimensionality.

An intuitive way to define the deformation cost μ is to compare the chunks of shape between the triplet (da^+, da, da^-) to $(d\bar{a}^+, d\bar{a}, d\bar{a}^-)$ in terms of their **elastic** similarity. With the intrinsic formulation of the $\mathcal{MS}$, this can be expressed in terms of bending/stretching and relative radius/orientation of the boundaries. Let the orientation of the boundary to be $(\theta_\xi^+, \theta_\eta^+)$ and $(\theta_\xi^-, \theta_\eta^-)$, and let the A_1^2 radius to be r and relative angle to be ϕ, Figure 8.4(a). Since the deformation is two-dimensional, in addition to the one-dimensional stretching in ξ and η respectively, let the cost of *shear* deformation (deforming a square toward a parallelogram) to be denoted by u and let the *trapezoidal* deformation (deforming a square toward a trapezoid) to be denoted by w, respectively. We define μ in terms of **elastic similarity** as: [5]

[3] One way to choose (ξ, η) is to use the intrinsic radius field r, *i.e.*, let ξ to be in the direction of ∇r and η to be in r = const. The detail in parameterizing (ξ^+, η^+), *etc.* needs extensive elaboration and is omitted.

[4] The alignment $\alpha_s(d\tilde{a})$ is a 2D manifold in a 4D space, which can be *monotonically* parameterized by $(\tilde{\xi}, \tilde{\eta})$. In comparison, the analogy of an alignment curve α in [169] is an 1D curve in a 2D space. The exact formulation of α_s in matching surfaces requires additional elaboration and is left as a future work.

[5] We only include significant elastic terms and omit the others. $R = 10$ is a scale-depending parameter of the weight between measures of angle and length.

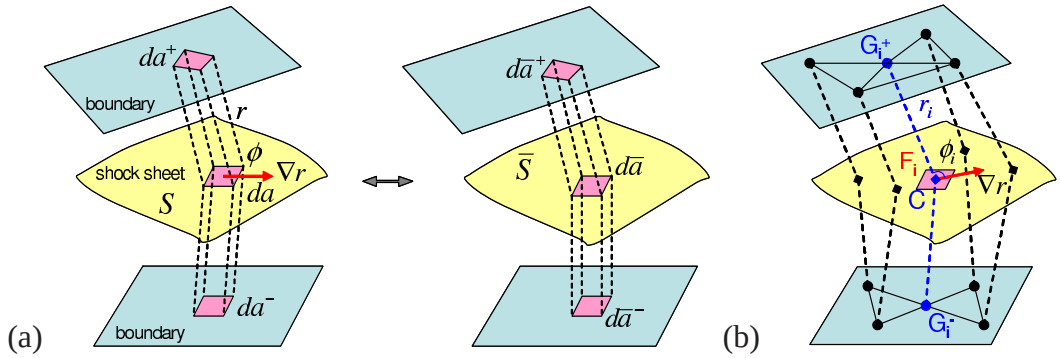


Figure 8.4: (a) Local configuration in matching two infinitesimal A_1^2 sheet elements da and $d\bar{a}$ by referring to their coupled boundary elements. (b) In practice the shape is sampled as points, and each A_1^2 face element $\mathsf{F_i}$ is coupled with two boundary points $(\mathsf{G_i^+}, \mathsf{G_i^-})$.

$$
\begin{aligned}
\mu[S,\bar{S},\alpha_s] = & \iint_{\alpha_s} \left[\left| \frac{d\bar{\xi}^+}{d\tilde{a}} - \frac{d\xi^+}{d\tilde{a}} \right| + \left| \frac{d\bar{\xi}^-}{d\tilde{a}} - \frac{d\xi^-}{d\tilde{a}} \right| \right] d\tilde{a} \; + \\
& \iint_{\alpha_s} \left[\left| \frac{d\bar{\eta}^+}{d\tilde{a}} - \frac{d\eta^+}{d\tilde{a}} \right| + \left| \frac{d\bar{\eta}^-}{d\tilde{a}} - \frac{d\eta^-}{d\tilde{a}} \right| \right] d\tilde{a} + R \iint_{\alpha_s} \left[\left| \frac{d\bar{\theta}_s{}^+}{d\tilde{a}} - \frac{d\theta_s^+}{d\tilde{a}} \right| \right. \\
& \left. + \left| \frac{d\bar{\theta}_s{}^-}{d\tilde{a}} - \frac{d\theta_s^-}{d\tilde{a}} \right| + \left| \frac{d\bar{\theta}_t{}^+}{d\tilde{a}} - \frac{d\theta_t^+}{d\tilde{a}} \right| + \left| \frac{d\bar{\theta}_t{}^-}{d\tilde{a}} - \frac{d\theta_t^-}{d\tilde{a}} \right| \right] d\tilde{a} \; + \\
& \iint_{\alpha_s} \left[\left| \frac{d\bar{u}^+}{d\tilde{a}} - \frac{du^+}{d\tilde{a}} \right| + \left| \frac{d\bar{u}^-}{d\tilde{a}} - \frac{du^-}{d\tilde{a}} \right| \right] d\tilde{a} + \iint_{\alpha_s} \left[\left| \frac{d\bar{w}^+}{d\tilde{a}} - \frac{dw^+}{d\tilde{a}} \right| \right. \\
& \left. + \left| \frac{d\bar{w}^-}{d\tilde{a}} - \frac{dw^-}{d\tilde{a}} \right| \right] d\tilde{a} + 2\left|\bar{r}^0 - r^0\right| + 2 \iint_{\alpha_s} \left| \frac{d\bar{r}}{d\tilde{a}} - \frac{dr}{d\tilde{a}} \right| d\tilde{a} \\
& + 2R\left|\bar{\phi}^0 - \phi^0\right| + 2R \iint_{\alpha_s} \left| \frac{d\bar{\phi}}{d\tilde{a}} - \frac{d\phi}{d\tilde{a}} \right| d\tilde{a},
\end{aligned}
\tag{8.4}
$$

which can be viewed as an extension of Eq.(5) in [169] in the 3D case, where $d\tilde{a}$ denotes an infinitesimal area in the integration terms.

* Approximation in practice: summing up differences of the pairs of A_1^2 sheet elements.

In practice the shapes are often sampled in points, making the differential terms of μ (such as stretching, bending, shear, and trapezoidal deformations) difficult to estimate. Let the A_1^2 sheets compose of elements, $S = \{\mathsf{F}_i \,|\, i = 1, ..., n\}$, $\bar{S} = \{\bar{\mathsf{F}}_i \,|\, i = 1, ..., \bar{n}\}$, where each A_1^2 mesh face element $\mathsf{F_i}$ is associated with two *generator* points $(\mathsf{G_i^+}, \mathsf{G_i^-})$, Figure 8.4(b), and approximate α_s by an element-wise mapping of $\{\mathsf{F_i}\}$ to $\{\bar{\mathsf{F}}_\mathsf{i}\}$ [6]. We approximate the deformation cost μ by summing only the essential *volume* (radius) and orientation terms:

$$
\mu[S,\bar{S},\alpha_s] \simeq \sum_{i=1}^{N} \left|\bar{r}_i - r_i\right| + R \sum_{i=1}^{N} \left|\bar{\phi}_i - \phi_i\right| . \tag{8.5}
$$

[6] $\mathsf{F_i}$ approximates $d\tilde{a}$ in Eq.(8.4) if the sampling are dense enough.

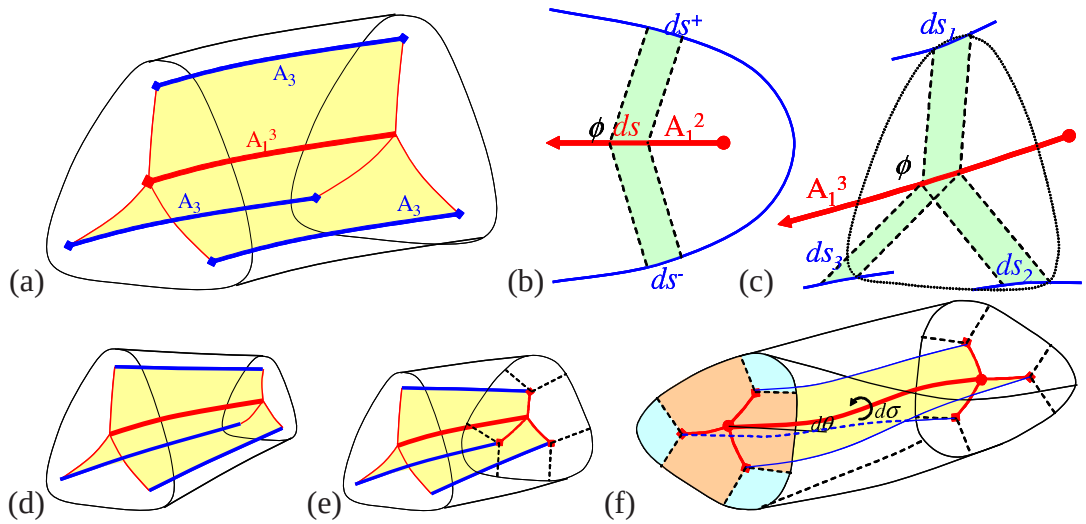


Figure 8.5: (a) Measuring A_1^3 axial curve similarity as a generalized cylinder. (b-c) The sectional slices corresponding to infinitesimal A_1^2 and A_1^3 segments in 2D and 3D. (d-e) The radius $r(s)$ along the A_1^3 curve controls the size of the corresponding shape slice, where ρ_1, ρ_2, and ρ_3 control how the sectional shape resembles a triangular blub or a circle. (f) Shape change corresponding to the bending and twisting of the A_1^3 curve.

where R is the weighting constant between the angle and length measures depending on the object scale.

Despite such a great simplification of μ, determining the discrete alignment $\alpha_s(\mathsf{F}_i \leftrightarrow \bar{\mathsf{F}}_i)$ is still another barrier [7]. However, this turns out not causing a problem in our sub-optimal matching approach described in § 8.1. Specifically, we can avoid an explicit determination of α_s throughout the matching process, since *(i)* α_s is trivial in estimating the sheet splice/contract transform cost, where $\bar{S}$ shrinks to a point (detailed in § 8.5) and *(ii)* the graph-matching of $\mathcal{MS}$ hypergraph in Chapter 9 does not explicitly involve matching of $\mathcal{MS}$ sheets. We will address this more in § 8.5 to show that no explicit determination of α_s is necessary in defining the $\mathcal{MS}$ transform cost, except the above trivial cases.

In additional to comparing the A_1^2 sheets, the comparing of A_1^3 and A_3 curves is explicitly used in the graph matching algorithm and is discussed as follows.

8.3 Deformation Cost to Match a Pair of A_1^3 Medial Curves

* Theoretical formulation: integration of infinitesimal shape changes to define the A_1^3 deformation cost.

The A_1^3 axial curve is equally-distinct from three boundary points and is also the intersection of three medial sheets, which captures salient information pertinent to the shape. We define the pairwise similarity of A_1^3 axials as the *optimal deformation cost of joint boundary elastic matching* similar to the A_1^2 case. Let the A_1^3 curve C be parameterized by arc-length s; each infinitesimal length ds corresponds to a volumetric *"sectional slice"* of a shape similar to a triangular blob (mimicking a *generalized cylinder*), Figure 8.5(c). Note that such slice is two dimensional in shape and may not be planar (and can be as well like a pyramid), see Figure 8.5(b) for a sectional slice of an A_1^2 shock

[7] Unlike the 2D case in [169], determination of the optimal α_s via dynamic programming as an extension is not trivial, since the shapes of $\{\mathsf{F}_i\}$ and $\{\bar{\mathsf{F}}_i\}$ can be irregular and involve an additional dimensionality.

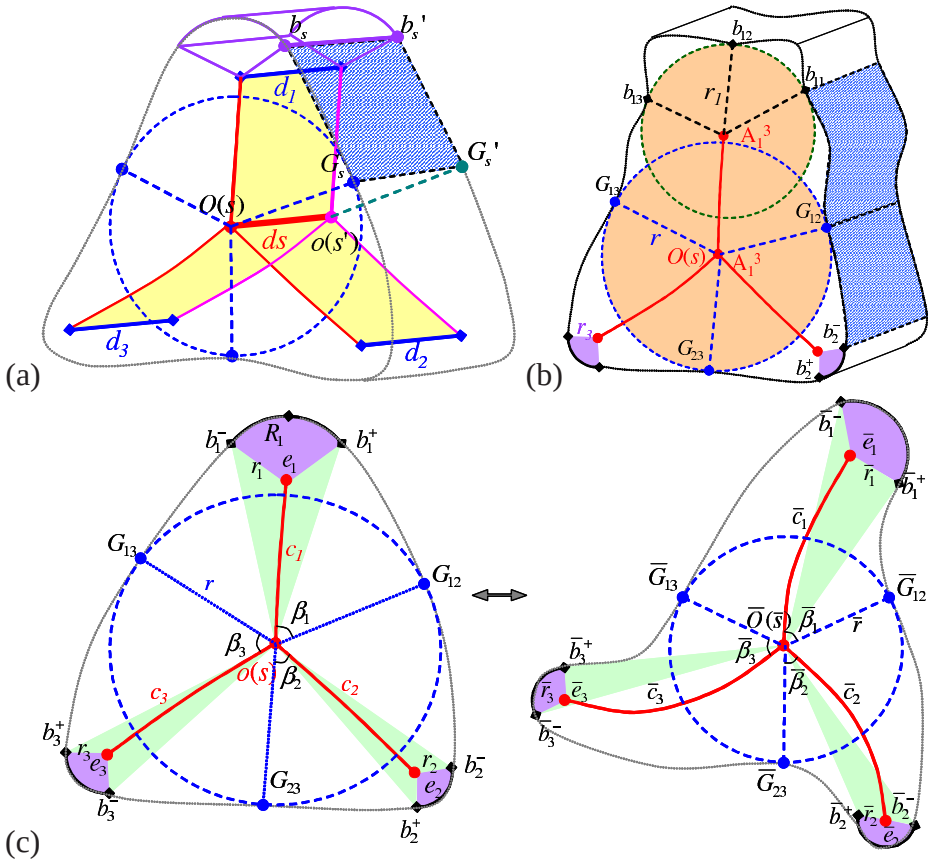


Figure 8.6: (a) A sectional slice along the A_1^3 curve. Given the A_1^3 axial and three A_3 ribs and that the $\mathcal{MS}$ sheets are implicit, the rectangular boundary block $G_sG'_sb'_sb_s$ (blue shaded region) is to be reconstructed to approximate the missing boundary region for similarity comparison. (b) A_1^3 with sheets ending in another A_1^3 instead of a rib. (c) Matching A_1^3 triangular blobs in different shapes, Eq.(8.6).

curve in 2D. [8]. Let the *sectional slice* of an A_1^3 curve C in 3D intersects its three incident sheets (S_1, S_2, S_3) at three *slicing curves* (c_1, c_2, c_3); see Figure 8.6(c) for an illustration. Let (ρ_1, ρ_2, ρ_3) to be the lengths of (c_1, c_2, c_3) and β_{12}, β_{23}, β_{13} to be the angles between (c_1, c_2), (c_2, c_3), and (c_1, c_3), respectively. By assuming curve c_j to be the bisector of angle $(\mathsf{G}_{ij}\text{-}\mathsf{O}\text{-}\mathsf{G}_{jk})$, Figure 8.6(c), we use $(\beta_1 = \angle \mathsf{G}_{12}\mathsf{O}\mathsf{e}_1, \beta_2 = \angle \mathsf{G}_{23}\mathsf{O}\mathsf{e}_2, \beta_3 = \angle \mathsf{G}_{13}\mathsf{O}\mathsf{e}_3)$ to characterize the sectional angles instead of $(\beta_{12}, \beta_{23}, \beta_{13})$. These parameters control how the triangular blob resembles a triangle or a circle [9].

Consider the $\mathcal{MS}^{\mathcal{H}-}$ with geometry of the sheets implicit, the boundary shape pertinent to the A_1^3 curve C is two-fold: *(i)* the three generators $(\mathsf{G}_{12}, \mathsf{G}_{23}, \mathsf{G}_{13})$ equally distant at radius r and *(ii)* the three end points $(\mathsf{e}_1, \mathsf{e}_2, \mathsf{e}_3)$ of (c_1, c_2, c_3), which are either on the A_3 ribs or on other A_1^3 axials. Figure 8.6(a,b) depicts two cases of the triangular blob where in Figure 8.6(a) all $\{\mathsf{e}_i\}$ are on the A_3 ribs and in Figure 8.6(b) the case one of them is A_1^3. In either case, the radii (r_1, r_2, r_3) of $(\mathsf{e}_1, \mathsf{e}_2, \mathsf{e}_3)$ spans three *"fans"* in some angles which give a significant hint of the overall shape. Essentially, we are interpolating the missing shape (the shaded blue region) between the three fans and the three

[8]The sectional slice degenerates to become planar if $\phi = \pi/2$, when the A_1^3 point is an A_1^3-2 source or A_1^3-4 sink. For ease of illustration, the sectional slices in the Figures might look planar, but keep in mind it is no-planar in general.

[9]The radii (r_1, r_2, r_3) and lengths (ρ_1, ρ_2, ρ_3) are important observe how Figure 8.5(d,e) are similar in the shape of the $\mathcal{MS}$ hypergraphs, while how Figure 8.5(e,a) are similar in the overall boundary shape. In practice ρ_j can be approximated by a more robust distance from the rib curve to $\mathsf{O}(s)$, *i.e.*, $\rho_j = r(\mathsf{O}) - r(\mathsf{e}_j)$, where $r(\cdot)$ is the radius.

generators and compare them using elastic deformation.

One important observation in practice is that the A_1^3 axials are generally more reliable than the A_3 ribs [10], suggesting an explicit usage of the geometry of the A_1^3. We propose to replace (ρ_1, ρ_2, ρ_3) by the a more robust measure on the boundary: $\mathsf{b_j} = (\mathsf{Ob_j^-} + \mathsf{Ob_j^+})/2$. The bending of (c_1, c_2, c_3) is modeled as the difference of distance of the 6 end points of the fan to the center o in pairs. Let $\psi_j^- = \angle(\mathsf{OG_{jk}}, \mathsf{Ob_j^-})$ and $\psi_j^+ = \angle(\mathsf{OG_{jk}}, \mathsf{Ob_j^+})$. Define the bending term δ in two cases: *(i)* for c_j ending in an A_3, $\delta_j = |\psi_j^- - \phi_j| + |\psi_j^+ - \phi_j|$; *(ii)* for c_j ending in an A_1^3, $\delta_j = |\angle(\mathsf{OG_{jk}}, \mathsf{Oe_j})|$. Summing up all three bending terms for curves (c_1, c_2, c_3), $\delta = \delta_1 + \delta_2 + \delta_3$. The deformation cost to match A_1^3 curves $(C, \bar{C})$, given an alignment curve α between $(C, \bar{C})$ is:

$$
\begin{aligned}
\mu[C, \bar{C}, \alpha] = {} & 3\left|\bar{r}^0 - r^0\right| + 3\int_0^{\tilde{L}} \left|\frac{d\bar{r}}{d\xi} - \frac{dr}{d\xi}\right| d\xi + \sum_{j=1}^{3} \left|\bar{\rho_j}^0 - \rho_j^0\right| + \\
& \sum_{j=1}^{3} \int_0^{\tilde{L}} \left|\frac{d\bar{\rho}_j}{d\xi} - \frac{d\rho_j}{d\xi}\right| d\xi + \sum_{j=1}^{3} \left|\bar{r_j}^0 - r_j^0\right| + \sum_{j=1}^{3} \int_0^{\tilde{L}} \left|\frac{d\bar{r}_j}{d\xi} - \frac{dr_j}{d\xi}\right| d\xi \\
& + R\sum_{j=1}^{3} \left|\bar{\beta_j}^0 - \beta_j^0\right| + R\sum_{j=1}^{3} \int_0^{\tilde{L}} \left|\frac{d\bar{\beta}_j}{d\xi} - \frac{d\beta_j}{d\xi}\right| d\xi + R\sum_{j=1}^{3} \left|\bar{\delta_j}^0 - \delta_j^0\right| \\
& + R\sum_{j=1}^{3} \int_0^{\tilde{L}} \left|\frac{d\bar{\delta}_j}{d\xi} - \frac{d\delta_j}{d\xi}\right| d\xi + \sum_{j=1}^{3} \int_0^{\tilde{L}} \left|\frac{\bar{d}_j(\xi)}{d\xi} - \frac{d_j(\xi)}{d\xi}\right| d\xi \\
& + 3R\int_0^{\tilde{L}} \left|\frac{d\bar{\theta}(\xi)}{d\xi} - \frac{d\theta(\xi)}{d\xi}\right| d\xi + R\int_0^{\tilde{L}} \left|\frac{d\bar{\sigma}(\xi)}{d\xi} - \frac{d\sigma(\xi)}{d\xi}\right| d\xi. \qquad (8.6)
\end{aligned}
$$

where the r_j and d_j terms reflect the boundary bending and stretching differences; the θ and σ terms are the shock bending and twisting differences; $\tilde{L}$ is the length of the alignment curve α.

* Approximation in practice: summing up differences of the pairs of A_1^3 curve elements.

In practice the A_1^3 curve C is modeled by a poly-line. Let $C = \{\mathsf{L}_i \,|\, i = 1, ..., n\}$, $\bar{C} = \{\bar{\mathsf{L}}_i \,|\, i = 1, ..., \bar{n}\}$, where each l_i is associated with 3 generators and 3 incident A_1^2 sheet elements. The alignment curve α is approximated by an assignment of $\{\mathsf{L}_i\} \leftrightarrow \{\bar{\mathsf{L}}_i\}$. For the case where the sheet S_j of C ending in an A_3, the slicing curve length terms ρ_j are replaced by the $\mathsf{b_j}$ terms for better robustness. The deformation cost is then:

$$
\begin{aligned}
\tilde{\mu}[C, \bar{C}, \alpha] \simeq {} & 3\sum_{i=1}^{N} |\bar{r}_i - r_i| + \sum_{i=1}^{N}\sum_{j=1}^{3} \left|\bar{b}_{ij} - b_{ij}\right| + \\
& \sum_{i=1}^{N}\sum_{j=1}^{3} |\bar{r}_{ij} - r_{ij}| + R\sum_{i=1}^{N}\sum_{j=1}^{3} \left|\bar{\beta}_{ij} - \beta_{ij}\right| + R\sum_{i=1}^{N}\sum_{j=1}^{3} \left|\bar{\delta}_{ij} - \delta_{ij}\right| \\
& + \sum_{i=1}^{N}\sum_{j=1}^{3} \left|\bar{d}_{ij} - d_{ij}\right| + 3R\sum_{i=1}^{N} \left|d\bar{\theta}_i - d\theta_i\right|, \qquad (8.7)
\end{aligned}
$$

where the optimal pairing between the three branches (S_1, S_2, S_3) to $(\bar{S}_1, \bar{S}_2, \bar{S}_3)$ can be obtained by testing all permutations and use the optimal one.

[10] This is expectable that the high symmetry of the A_1^3 makes it robust, and the high 'order' of symmetry of A_3 makes it sensitive to perturbations and unstable/noisy even after regularization).

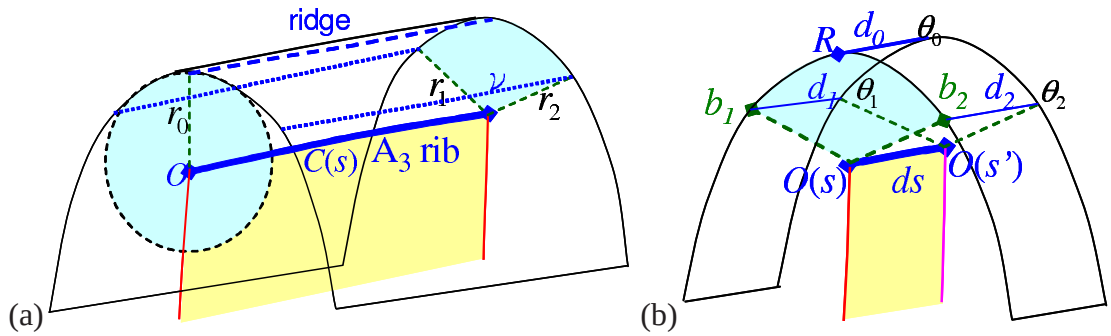


Figure 8.7: The shape of an A_3 rib corresponds to a parabolic gutter, where the distance to the ridge is controlled by the A_3 radius r. (a) Parameters of the A_3 rib curve for matching. (b) A sectional slice along the A_3 curve.

The similarity $d[C, \bar{C}]$ between the pairs of A_1^3 curves are computed by finding the optimal alignment α similarly as in Eq.(8.3) via dynamic programming [169].

8.4 Deformation Cost to Match a Pair of A_3 Medial Curves

* Theoretical formulation: integration of infinitesimal shape changes to define the A_3 deformation cost.

The A_3 rib curve is the formation of the A_1^2 sheet in the *grass-fire* propagation scheme and its boundary. The matching between A_3 ribs are similarly defined as the elastic deformation between the ridge regions of the parabola-like gutters, Figure 8.7. We again parameterize the A_3 curve C by arc-length s. At each infinitesimal segment ds, the spanning *fan* of the rib curve can be characterized by the following terms: *(i)* the radius r_0 from the center o to the ridge point R, *(ii)* the radii r_1, r_2 of the immediate A_1^2 point (on the border of a sheet) to its two distinct boundary points $(\mathsf{b}_1, \mathsf{b}_2)$, *(iii)* the boundary stretching terms (d_0, d_1, d_2), and *(iv)* the spanning angle ν of the fan $\mathsf{O}\mathsf{b}_1\mathsf{b}_2$, Figure 8.7.

The deformation cost to match A_3 curves $(C, \bar{C})$, given an alignment α between them is:

$$\begin{aligned}\mu[C, \bar{C}, \alpha] &= 3\left|\bar{r}^0 - r^0\right| + 3\int_0^{\tilde{L}}\left|\frac{d\bar{r}}{d\xi} - \frac{dr}{d\xi}\right| d\xi + R\left|\bar{\nu}^0 - \nu^0\right| \\ &+ R\int_0^{\tilde{L}}\left|\frac{d\bar{\nu}}{d\xi} - \frac{d\nu}{d\xi}\right| d\xi + \sum_{j=0}^{2}\int_0^{\tilde{L}}\left|\frac{\bar{d}_j(\xi)}{d\xi} - \frac{d_j(\xi)}{d\xi}\right| d\xi \\ &+ R\sum_{j=0}^{2}\int_0^{\tilde{L}}\left|\frac{d\bar{\theta}_j(\xi)}{d\xi} - \frac{d\theta_j(\xi)}{d\xi}\right| d\xi.\end{aligned} \tag{8.8}$$

where the r terms are the A_3 shock radius difference, the ν terms are the spanning angle difference, the d_j and $d\theta_j$ terms are the boundary stretching and bending differences, the shock twisting term is omitted (due to its unreliability).

* Approximation in practice: summing up differences of the pairs of A_3 curve elements.

In practice, the A_3 curve is modeled by a poly-line and can be noisy depending on the sampling condition. Each A_3 line segment L_i is associated with 3 generators and one incident A_1^2 sheet elements F_i. Let the alignment curve α is approximated by the assignment of $\{\mathsf{F}_\mathsf{i}\} \leftrightarrow \{\bar{\mathsf{F}}_\mathsf{i}\}$, the radius term is

estimated on an average fashion, the deformation cost is then:

$$\tilde{\mu}[C,\bar{C},\alpha] \simeq \sum_{j=0}^{2}\left(\sum_{i=1}^{\bar{N}}\bar{r}_{ij}\cdot\frac{N}{\bar{N}}-\sum_{i=1}^{N}r_{ij}\right)+R\sum_{i=1}^{N}|\bar{\nu}_i-\nu_i|+ \\ \sum_{j=0}^{2}\sum_{i=1}^{N}\left|\bar{d}_{ij}-d_{ij}\right|+R\sum_{j=0}^{2}\sum_{i=1}^{N}\left|d\bar{\theta}_{ij}-d\theta_{ij}\right|. \tag{8.9}$$

The similarity $d[C,\bar{C}]$ between the pairs of A_3 curves are computed by finding the optimal alignment α similarly as in Eq.(8.3) via dynamic programming [169].

8.5 Defining Deformation Costs for the $\mathcal{MS}$ Transforms

We have described the deformation costs in optimally match arbitrary A_1^2 sheet-sheet, A_1^3 curve-curve, and A_3 curve-curve pairs in the previous section. In this section we show how to use these results to define the set of $\mathcal{MS}$ transforms described in Chapter 5 as a **limit** case, where the second sheet $\bar{S}$ or curve $\bar{C}$ shrinks to a point.

Specifically, the sheet-splice transform is modeled as a deformation sequence where a sheet shrinks to a point. The contract transform is modeled as either a sheet or a curve shrinks to a point. The merge transform along a path on some sheet is modeled as a deformation sequence that shrinks this path (as a virtual curve) to a point. The deformation cost of $\mathcal{MS}$ transforms is summarized in Table 8.1, whose the implementation is left as a future work.

Table 8.1: Summary of the proposed system of $\mathcal{MS}$ transforms and the deformation (operation) of them.

$\mathcal{MS}$ transform	Deformation of the $\mathcal{MS}$
A_1A_3-I sheet splice	a sheet $\rightarrow$ a point
$A_1^2A_3$-II sheet splice	a sheet $\rightarrow$ a point
A_5 curve contract	a curve with 3 sheets $\rightarrow$ a point
$A_1^2A_3$-I curve contract	a curve with a sheet $\rightarrow$ a point
A_1^5 curve contract	a curve with 3 sheets $\rightarrow$ a point
A_1^5 sheet contract	a sheet $\rightarrow$ a point
A_1^4 sheet contract	a sheet $\rightarrow$ a point
A_1A_3-II node-node merge	merging a path (curve) with a sheet $\rightarrow$ a point
$A_1^2A_3$-I node-curve merge	merging a path (curve) with a sheet $\rightarrow$ a point
A_1^4 curve-curve merge	merging a path (curve) with a sheet $\rightarrow$ a point
A_1A_3-II curve-curve merge	merging a path (curve) with a sheet $\rightarrow$ a point
Corner node-curve merge	merging a path (curve) with a sheet $\rightarrow$ a point
$A_1^3A_3$-I node-node merge	merging a path (curve) with a sheet $\rightarrow$ a point
A_1^3 curve gap transform	1 curves with 3 sheets $\rightarrow$ 2 points
A_1^2 sheet gap transform	1 sheet $\rightarrow$ 0
Loop transform	$\epsilon \rightarrow 0$ *

* Summing of deformation cost of (all shocks modifications), which should be tiny.

Chapter 9

3D Object Recognition by Graph-Based Matching of the Medial Scaffolds

* Overview of chapter: adopt a graph-matching scheme to match the regularized $\mathcal{MS}$ hypergraphs.

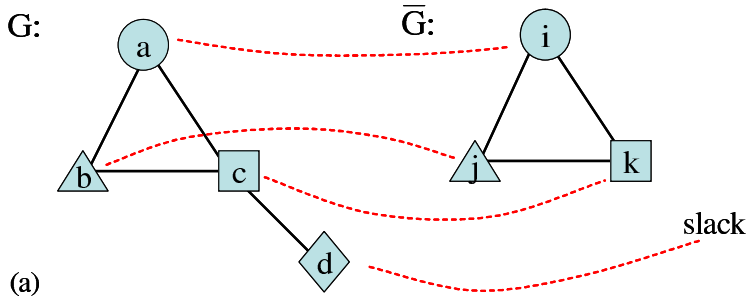
This chapter describe a graph-based matching approach to match the regularized medial scaffold ($\mathcal{MS}$). We adopt an *relaxation*-based *energy-minimization* approach—the **graduated assignment** ($\mathcal{GA}$) graph matching algorithm [95] to match the $\mathcal{MS}$ hypergraphs. Our approach in matching the $\mathcal{MS}$ of 3D shapes can be viewed as an extension of an earlier work [44] based on a 2D experience [175]. It is as well an implementation of the *sub-optimal* matching approach discussed in Chapter 8.

* Motivation: why choose the *graduated-assignment* approach?

Graph matching is a fundamental topic in recognition with extensive studies (Refer to § 2.4 for a survey). We choose to extend the *Graduated Assignment* ($\mathcal{GA}$) graph matching algorithm [95] to match the $\mathcal{MS}$ hypergraphs. Several factors motivate the use of this algorithm: First, it enforces *two-way* assignment via *softassign* [186, 120] in contrast to *relaxation labeling* type algorithms which enforce one-way assignment. It is possible to further extend the two-way assignment to a *three-way* assignment to match the hypergraphs (detailed below), which makes this technique a strong fit in our case. Second, it avoids poor local minimum by the use of *graduated convexity* [29, 213] continuation technique. Third, this algorithm is efficient in comparison to current techniques (an order of magnitude better than relaxation labeling), partly due to an explicit encoding of sparsity. Fourth, the algorithm handles missing/extra nodes/links, which is important in matching shapes, and is superior in this regard to other existing techniques [95]. Fifth, the algorithm is stable under noisy conditions [95]. Finally, the formulation can be adapted to also take into account continuous variables representing similarity transformations and shape deformations. In addition, the $\mathcal{GA}$ was successfully used in *(i)* matching 2D shock graphs [175] (where 2D shock graphs are matched to index a 25-shape database) as well as in *(ii)* registering 3D shapes [44], both with promising results. The goal of this chapter is to extend this line of approach to match the node/link/hyperlink attributes of the $\mathcal{MS}$ hypergraph based on their structure and their geometry and shock dynamic features.

* Organization of chapter.

This chapter is organized as follows. We first review Rangarajan's original $\mathcal{GA}$ graph matching approach (§ 9.1) and then discusses several issues in adopting the $\mathcal{GA}$ into our framework to match the $\mathcal{MS}$ hypergraphs (§ 9.2). Our main extension to the $\mathcal{GA}$ is on designing a set of proper *compatibility* measures in matching between the $\mathcal{MS}$ nodes, curves, and sheets, which will be elaborated in § 9.3. In addition, recall that we first regularize the $\mathcal{MS}$ and then match them. The regularization



M:

	i	j	k	slack
a	1	0	0	0
b	0	1	0	0
c	0	0	1	0
d	0	0	0	1
slack	0	0	0	0

(b)

Figure 9.1: (a) The matching of two synthetic graphs G and $\bar{G}$, where the extra node is matched into the 'slack' column. (b) shows the matching matric $\mathbf{M}$.

in Chapter 7 does not remove all transitions (and should not), thus the remaining $\mathcal{MS}$ might still contain unhandled $\mathcal{MS}$ transitions which can degrade the graph-matching performance. Section 9.4 describes a way to handle the $\mathcal{MS}$ across transitions by adding *virtual links*, which improves the graph matching robustness. Moreover, after the matching is performed (*i.e.*, yielding node assignments), Section 9.5 computes a final *similarity* measure between the two shapes by summing up the compatibility between their medial nodes, curves and sheets. Finally, Section 9.6 summarizes the main steps of the matching algorithm and shows experimental results.

9.1 A Reviews of the Graduated Assignment ($\mathcal{GA}$) Graph Matching Algorithm

* Setting up the basics of the *graduated assignment* algorithm.

We briefly review Rangarajan *et al.*'s graduated assignment algorithm [95] as follows. The basic idea underlying this graph matching approach is to associate nodes in two graphs as represented by a *match matrix* $\mathbf{M}$ (a permutation matrix if the numbers of nodes in two graphs are equal) where 1 represents association of two nodes and 0 represents no association, Figure 9.1. A **slack** *row* and *column* are added to $\mathbf{M}$ to represent missing/extra nodes. In order to allow for differential movement for one permutation to another, *graduated non-convexity* is used to turn these discrete (binary) variables into continuous ones between $(0, 1)$. To avoid poor local minima, a control parameter is used to slowly move the matrix towards a 0 or 1 discretization. At each stage, the best match matrix is estimated and normalized to ensure it remains the continuous analogue of a discrete assignment, using a technique discovered by Sinkhorn [186].

Formally, consider two graphs G and $\overline{G}$. Refer to nodes of G and $\overline{G}$ by G_a and $\overline{G}_i$, respectively, and links of G and $\overline{G}$ by G_{ab} and $\overline{G}_{ij}$, respectively, where $a, b = 1, \ldots, A$, and $i, j = 1, \ldots, I$. The *match matrix* $\mathbf{M}$ associates nodes in two graphs:

$$\mathbf{M}_{ai} = \begin{cases} 1 & \text{if the node } a \in G \text{ corresponds to node } i \in \overline{G} \\ 0 & \text{otherwise.} \end{cases} \tag{9.1}$$

In graduated assignment [95], an objective *energy function* $E(\mathbf{M})$ is defined for each possible assignment $\mathbf{M}$. Let the energy E of a *quadratic* matching problem be

$$E(\mathbf{M}) = \sum_{i=1}^{I} \sum_{a=1}^{A} \sum_{b=1}^{A} \sum_{j=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot L_{aibj} \quad , \tag{9.2}$$

which we want to maximize, where L_{aibj} represents the similarity between links G_{ab} and $\overline{G}_{ij}$.

* The *"gradual"* assigning process.

A significant idea in [95] is to extend the *discrete* assignment problem to a *continuous* one by embedding it into a large space, where *gradient descent* can be performed to iteratively move from one assignment to another. A continuous analogy to the *discrete* matching matrix is a *continuous* assignment matrix (also denoted $\mathbf{M}$), which takes values between 0 and 1 with the constraint that $\mathbf{M}$ has to be a *doubly stochastic* matrix, *i.e.*, $\sum_a \mathbf{M}_{ai} = 1$ and $\sum_i \mathbf{M}_{ai} = 1$ [186]. The graduated assignment then *differentially* moves from one assignment $\mathbf{M}$ to another, retaining it to take values between 0 and 1 as a doubly stochastic matrix (including the *slack* rows and columns). The moving relies on a *gradient descent* like iterations on refining the matching energy $E(\mathbf{M})$, in a *graduated non-convexity* setting. The *Taylor expansion* of the energy function *w.r.t.* $\mathbf{M}$ is:

$$E(\mathbf{M}) = E(\mathbf{M}^0) + \sum_{a=1}^{A}\sum_{i=1}^{I} Q_{ai} \cdot (\mathbf{M}_{ai} - \mathbf{M}^0_{ai}), \tag{9.3}$$

where the derivative matrix Q is:

$$Q = \left.\frac{\partial E}{\partial \mathbf{M}_{ai}}\right|_{\mathbf{M}=\mathbf{M}^0} = \sum_{b=1}^{A}\sum_{j=1}^{I} \mathbf{M}^0_{bj} \cdot L_{aibj}, \tag{9.4}$$

which turns the maximizing E into maximizing

$$\sum_{a=1}^{A}\sum_{i=1}^{I} Q_{ai} \cdot \mathbf{M}_{ai}, \tag{9.5}$$

which is (again) an *assignment problem* [95]. This assignment can be solved by *softassign* [186], where an initial matrix is moved toward a solution by decreasing a parameter T ('temperature' in annealing) which controls the convexity of the energy landscape (to avoid poor local minima). In each iteration of graduated assignment, the continuous match matrix $\mathbf{M}$ is best estimated and normalized such that it gradually moves toward a 0 or 1 discretization, which gives its final assignment.

* The iterative assignment algorithm.

The $\mathcal{GA}$ algorithm follows three nested iterations summarized in Table 9.1. Start with an initial $\mathbf{M}$: *(i)* For each temperature T and $\mathbf{M}$, find $Q(M)$ by optimizing the compatibility measures (1 for a perfect match, 0 for no match) between graph nodes and links using the current $\mathbf{M}$. *(ii)* Updating $\mathbf{M}$ through softassign, by repeatedly normalizing rows and columns to make converge into a doubly stochastic matrix. *(iii)* Repeat the procedure by decreasing T until $\mathbf{M}$ converges or enough iterations are performed.

* Extend the energy E to match the general attributed relational graphs.

Having outlined the algorithm, it remains to define E in a meaningful manner. Gold and Rangarajan [95] give a generic definition of $E(\mathbf{M})$ for *attributed relational graphs* (ARGs) as

$$E(\mathbf{M}) = \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot L_{aibj} + \alpha \sum_{a=1}^{A}\sum_{i=1}^{I} \mathbf{M}_{ai} \cdot N_{ai} \quad , \tag{9.6}$$

Begin A: (Do A until $T < T_f$ or # $iters. > I_A$)
  Begin B: (until $\mathbf{M}$ converges or # $iters. > I_B$)
    $Q_{ai} \leftarrow -\frac{\partial E}{\partial \mathbf{M}_{ai}}$ estimated from graph compatibilities.
    $\mathbf{M}^0_{ai} \leftarrow \exp(1/T \cdot Q_{ai})$
    Begin C: (until $\hat{\mathbf{M}}$ converges or # $iters. > I_C$)
      Update $\hat{M}$ by normalizing across all rows:
      $\hat{\mathbf{M}}^1_{ai} \leftarrow \frac{\hat{\mathbf{M}}^0_{ai}}{\sum_{i=1}^{I+1} \hat{\mathbf{M}}^0_{ai}}$
      Update $\hat{M}$ by normalizing across all columns:
      $\hat{\mathbf{M}}^0_{ai} \leftarrow \frac{\hat{\mathbf{M}}^1_{ai}}{\sum_{a=1}^{A+1} \hat{\mathbf{M}}^1_{ai}}$
    End C
  End B
  $T \leftarrow T_r \cdot T$
End A

Parameter	Description
T_0 = 100	The initial temperature
T_f = 0.01	The termination temperature
T_r = 0.95	Rate in decreasing temperature
I_A = 100	Maximum iterations in *softassign*
I_B = 4	Maximum iterations in loop B
I_C = 30	Maximum iterations in loop C
ϵ_B = 0.005	Criterion of convergence of $\mathbf{M}$
ϵ_C = 0.0005	Criterion of convergence of $\hat{\mathbf{M}}$
ϵ_S = 0.00005	Criterion of convergence in *softassign*

Table 9.1: (Left, adapted from [95]) A summary of the graduated assignment algorithm in [95]. (Right) The $\mathcal{GA}$ parameters with our suggested values.

where L_{aibj} represents the total compatibility between links G_{ab} and $\overline{G}_{ij}$, and N_{ai} represents total compatibility between nodes G_a and $\overline{G}_i$:

$$L_{aibj} = \begin{cases} 1 & \text{if links } G_{ab} \text{ and } \bar{G}_{ij} \text{ both exist} \\ 0 & \text{otherwise} \end{cases} \tag{9.7}$$

$$N_{ai} = \begin{cases} 1 & \text{if nodes } a \text{ and } i \text{ match} \\ 0 & \text{otherwise} \end{cases} \tag{9.8}$$

This leads to,

$$Q_{ai}(M) = 2 \sum_{b=1}^{A} \sum_{j=1}^{I} M_{bj} L_{aibj} + \alpha N_{ai} \ , \tag{9.9}$$

which allows to robust match any two arbitrary attributed graphs. In our case of hypergraph matching, we continue to extend the energy E to include compatibility to match hyperlinks (medial sheets), detailed as follows.

9.2 On Extending the $\mathcal{GA}$ to Match $\mathcal{MS}$ Hypergraphs

We have exploited several ideas in extending the original $\mathcal{GA}$ to match the $\mathcal{MS}$ hypergraphs. *(i)* First, the similarity should reflect the (optimal) deformation cost in matching two shapes (a sub-optimal approximation of the edit-distance matching described in Chapter 8). *(ii)* Second, in order to match the additional dimension of the hypergraphs, a **third-order assignment** is introduced to capture the matching of surface segments, in additional to the first-order and second-order energy to capture the matching of nodes and curves, respectively. *(iii)* Third, to accurately filter out dissimilar matches, we match both the (graph) structure and the parametric attributes (along the curves and sheets) of the $\mathcal{MS}$ hypergraphs. *(iv)* Forth, we use a square-root distance metric, which outperforms other metrics [175] (detailed below). *(v)* Finally, we introduce a set of "virtual links" as in [175] to deal with unhandled $\mathcal{MS}$ transforms across transitions to improve the matching robustness (§ 9.4). We elaborate each of these ideas below.

* Similarity via deformation.

We define the similarity between two shapes to be inversely proportional to the minimum amount of deformation required to bring one shape in matching with another. Recall in Chapter 9 that the optimal solution of exploring the edit-distance of $\mathcal{MS}$ hypergraph matching is difficult, our goal is to approximate it via a computationally practical graph matching algorithm.

* Introduce a third-order assignment.

The matching between graph nodes in the $\mathcal{GA}$ can be viewed as the first-order assignment and the matching between curves can be viewed as the second-order assignment. We employ a *third-order* energy term to match the sheets (hyperlinks) in the $\mathcal{MS}$ hypergraph. While the medial sheets could have complex topology and thus is difficult to match directly, we indirectly model their matching by matching individual corners of the sheet (which involves two curves thus is third-order), whose overall effects accumulate to match the sheets. Note that the introduction of the third-order energy does not affect the computation time much due to its sparsity.

* Match the *structure* of the $\mathcal{MS}$ hypergraph.

The $\mathcal{MS}$ can be matched *structurally*, *i.e.*, given any assignment, simply consider whether or not a link or a hyperlink exists or checking if their shock type is consistent. We discuss the matching of medial nodes, curves, and sheets case-by-case as follows. *(i)* For matching medial nodes, recall that the $\mathcal{MS}$ regularization process (Chapter 7) moves the $\mathcal{MS}$ toward a high-order degeneracy, which essentially produces high-order $\mathcal{MS}$ nodes, and we have analyzed such high-order $\mathcal{MS}$ nodes in § 5.5. This analysis of high-order $\mathcal{MS}$ nodes is useful in defining their structural compatibility. *(ii)* For matching medial curves, in additional to checking their existence and shock type compatibility, the *orientation* and the ending node types are useful in matching. *(iii)* For matching medial sheets, we compare the structural information of each sheet *corner* (such as shock curve type and orientation) and let them sum up to match the whole sheet.

* Match the *parametric* attributes of the $\mathcal{MS}$ hypergraph.

In additional to the above structural matching, we can further refine the compatibility definition to consider more variational aspects of the $\mathcal{MS}$ hypergraphs, by *parameterizing* the local properties of the $\mathcal{MS}$ nodes, curves, and sheet corners and define a metric to compare them. Consider that a shape can be deformed to another via a sequence of (canonical) transformations. The dissimilarity between two shapes is the length of the *minimum length path* in the space of transformations (Chapter 9). We consider the following canonical deformations: *(i) stretching* or *compressing* where similarity is effectively translated into length comparison between link length; *(ii) fattening* or *thinning*, where similarity is measured by comparing the shock acceleration functions [86]; *(iii) bending* which affects the curvature in comparison. We summarize the parametric matching of nodes/curves/sheets case-by-case as follows. *(i)* Node similarity compares the size (shock radius r), shock velocities (related to the derivative of r), and other attributes such as the angle between incident curves. *(ii)* Link similarity is the functions of the aforementioned canonical deformations considered as separable dimensions of deformations. *(iii)* Corner similarity compares the corner angle, shock radius, and shock velocity, detailed in § 9.3.3.

* Use the *square-root* distance norm to compare similarity measures.

Table 9.2: Possible distance metrics. We choose to use the normalized square-root distance metric between two measurements m_1 and $m_2 \in R$, where M is the maximum value of all measurements used to normalize their values into $[0, 1]$.

	Weighted	Pre-normalized
Absolute distance	$d^w_{abs}(m_1, m_2) = \frac{\lvert m_1 - m_2 \rvert}{\max(m_1, m_2)}$	$d^n_{abs}(m_1, m_2; M) = \frac{1}{\lvert M \rvert} \lvert m_1 - m_2 \rvert$
Square-root distance	$d^w_{sqr}(m_1, m_2) = \sqrt{\frac{\lvert m_1 - m_2 \rvert}{\max(m_1, m_2)}}$	$d^n_{sqr}(m_1, m_2; M) = \frac{1}{\sqrt{\lvert M \rvert}} \sqrt{\lvert m_1 - m_2 \rvert}$
Square distance	$d^w_{sq}(m_1, m_2) = \left(\frac{\lvert m_1 - m_2 \rvert}{\max(m_1, m_2)}\right)^2$	$d^n_{sq}(m_1, m_2; M) = \frac{1}{M^2} (m_1 - m_2)^2$

Table 9.3: Comparing distance norms: Example measurement values m_1, m_2 and their absolute, square-root, and square distance norms.

	m_1	m_2	M	d^w_{abs}	d^n_{abs}	d^w_{sqr}	d^n_{sqr}	d^w_{sq}	d^n_{sq}
Tiny difference of small values	1.1	1.2	10	0.083	0.01	0.289	0.1	0.0069	0.0001
Tiny difference of middle values	5.5	5.6	10	0.018	0.01	0.134	0.1	0.0003	0.0001
Tiny difference of large values	8.1	8.2	10	0.012	0.01	0.11	0.1	0.0001	0.0001
Small difference	1.0	2.0	10	0.5	0.1	0.707	0.316	0.25	0.01
Mid. difference	3.0	5.0	10	0.4	0.2	0.632	0.447	0.16	0.04
Large difference	1.0	8.0	10	0.875	0.7	0.935	0.837	0.7656	0.49

We use a *square-root distance* $\sqrt{(m_1 - m_2)}$ for comparison between two measures m_1 and m_2, motivated by the following reasons [175]: *(i)* a re-interpretation of Weber's law, *(ii)* to maintain sensitivity to variations when two items are close, and *(iii)* to reduce sensitivity when two items are very distant. We have found this norm to perform better for our application than others, *e.g.* the weighted distance $\frac{\|m_1 - m_2\|}{max(m_1, m_2)}$ [1]. A second issue of the distance norms is whether the two measurements are *scale*-dependent, *i.e.* a pre-normalization is required or not. We summarize these distance norms in Table 9.2 and list a few example values to illustrate their characteristics in Table 9.3. [2]

9.3 Designing the $\mathcal{MS}$ Hypergraph Compatibility Measures

This section defines the node-to-node (§ 9.3.1), curve-to-curve (§ 9.3.2), and sheet-to-sheet (§ 9.3.3) **compatibility** measures in matching the $\mathcal{MS}$ by explicitly combining both the *structural* and *parametric* matching described above. Table 9.4 overviews the main terms used in each case.

9.3.1 The first-order *node* compatibility measure ($\mathcal{N}$)

The compatibility between two shock nodes $N_a \in \mathcal{MS}$ and $\bar{N}_i \in \overline{\mathcal{MS}}$ is considered *first-order* in our hypergraph matching framework and designed *intrinsically* to compare the two given nodes. The compatibility $\mathcal{N}$ should take value between 0 and 1, where $\mathcal{N} = 1$ represents a perfect match

[1] The problem of the weighted distance $\frac{\|m_1 - m_2\|}{max(m_1, m_2)}$ is that if both measures m_1 and m_2 are low, the distance is dominated by the small values and can 'flip' the distance measure. For example, observe in Table 9.3, the weighted square-root distance d^w_{sqr} is 0.707 for the measurements of a small difference but only 0.632 for those with a mid. difference.

[2] In comparison, Rangarajan [95, §2.2] suggests to use $1 - 3|m_1 - m_2|$ to yield an expected value of zero when m_1 and m_2 are randomly selected from a uniform distribution in the interval [0, 1]. However, this norm creates negative similarity values, thus not suitable here.

Table 9.4: The compatibility design in matching the $\mathcal{MS}$ hypergraphs.

	Structural similarity	Parametric similarity
1st order **node** compatibility	node type, incident curve types	radius r, ∇r along incident curves, angle between curves at sheet corners
2nd order **curve** compatibility	existence, curve type, ending node types, orientation	$\sum r_i$ (to approximate volume), edit-distance in matching elastic deformations, Euclidean-distance between the alignment.
3rd order **sheet (corner)** compatibility	existence, incident curve types, corner node type	corner angle, radius r, ∇r at the corner, $\sum r_i$ (to approximate volume)

and $\mathcal{N} = 0$ represents the least match. We consider both *(i)* the *structural* compatibility and *(ii)* the *parametric* compatibility in computing a final measure between two nodes.

Structural node compatibility ($\mathcal{N}_s$)

The structural compatibility $\mathcal{N}_s$ compares the shock node types and incident shock curve types for their similarity. An intuitive idea is to avoid matching an interior A_1^4 node to an A_1A_3 node near the boundary.

* Compare high-order $A_1^m A_3$ and A_1^n $\mathcal{MS}$ nodes.

Consider the matching of higher-order shock nodes ($A_1^m A_3$ and A_1^n) in general (§ 5.5), we propose to penalize the shock node type difference by their difference in their *shock contact order*, and normalized it (to be within 0 and 1) by the maximum shock singularity between the two, (*e.g.*, mimicking the weighted distance d^w_{abs} in Table 9.2). To illustrate, the unit difference of an A_1 is 1, and the difference of an A_3 is 3 (three units); so the difference between an A_1A_3 and A_1^4 nodes cost three units and it is normalized by a total contact order of four, leading to $3/4$. This analysis can be generalize to compare all high-order $\mathcal{MS}$ nodes $A_1^m A_3$ and A_1^n discussed in § 5.5 as follows, $m, \bar{m} \geq 1$, $n, \bar{n} \geq 4$.

$$d[A_1^m A_3, A_1^n] = \frac{max(|n-m|, 3)}{max(n, m+3)}, \tag{9.10}$$

$$d[A_1^n, A_1^{\bar{n}}] = \frac{|n-\bar{n}|}{max(n, \bar{n})}, \tag{9.11}$$

$$d[A_1^m A_3, A_1^{\bar{m}} A_3] = \frac{|m-\bar{m}|}{max(m, \bar{m}) + 3}, \tag{9.12}$$

where $d[\cdot,\cdot]$ denotes the *shock type difference* between two $\mathcal{MS}$ nodes. Table 9.5 list its value between a few high-order $\mathcal{MS}$ nodes frequently observed in practice.

In § 5.5, we have also discussed the special shock nodes such as the *corner shock node* N_{cor} (intersection of one A_1^3 axial and three A_3 curves) and the *double-fin* node $(A_1A_3)_2$ and the *triple-fin* node $(A_1A_3)_3$. The distance between them is defined similarly as above and elaborated into three cases as follows. *(i)* The difference between the special shock nodes {N_{cor}, $(A_1A_3)_2$, $(A_1A_3)_3$} and

	A_1A_3	A_1^4	$A_1^2A_3$	A_1^5	$A_1^3A_3$	A_1^6	$A_1^4A_3$	A_1^7	$A_1^5A_3$	A_1^8	...	$A_1^mA_3$	A_1^n
A_1A_3	0	3/4	1/5	4/5	2/6	5/6	3/7	6/7	4/8	7/8	...	$\frac{m-1}{m+3}$	$\frac{n-1}{n}$
A_1^4		0	3/5	1/5	3/6	2/6	3/7	3/7	4/8	4/8	...	$\frac{m-1}{m+3}$	$\frac{n-4}{n}$
$A_1^2A_3$			0	3/5	1/6	4/6	2/7	5/7	3/8	6/8	...	$\frac{m-2}{m+3}$	$\frac{n-2}{n}$
A_1^5				0	3/6	1/6	3/7	2/7	3/8	3/8	...	$\frac{m-2}{m+3}$	$\frac{n-5}{n}$
$A_1^3A_3$					0	3/6	1/7	4/7	2/8	5/8	...	$\frac{m-3}{m+3}$	$\frac{n-3}{n}$
A_1^6						0	3/7	1/7	3/8	2/8	...	$\frac{m-3}{m+3}$	$\frac{n-6}{n}$
$A_1^4A_3$							0	3/7	1/8	4/8	...	$\frac{m-4}{m+3}$	$\frac{n-4}{n}$
A_1^7								0	3/8	1/8	...	$\frac{m-4}{m+3}$	$\frac{n-7}{n}$
$A_1^5A_3$									0	3/8	...	$\frac{m-5}{m+3}$	$\frac{n-5}{n}$
A_1^8										0	...	$\frac{m-5}{m+3}$	$\frac{n-8}{n}$
...											0	...	...
$A_1^mA_3$												0	$\frac{max(\lvert n-m\rvert,3)}{max(n,m+3)}$
A_1^n													0

Table 9.5: Difference d between general shock nodes reflecting the lack of structural (type) compatibility between them.

a general $A_1^mA_3$ shock nodes is defined as:

$$d[N_{cor}, A_1^mA_3] = \frac{m}{max(m+3,5)}, \tag{9.13}$$

$$d[(A_1A_3)_2, A_1^mA_3] = \frac{3+|m-2|}{max(m+3,8)}, \tag{9.14}$$

$$d[(A_1A_3)_3, A_1^mA_3] = \frac{6+|m-3|}{max(m+3,12)}. \tag{9.15}$$

(ii) The case of $\{N_{cor}, (A_1A_3)_2, (A_1A_3)_3\}$ to an A_1^n node is special that an A_1^n node has no A_3 rib curve incident to it. To simplify the analysis, their distance is defined to take the maximum value:

$$d[N_{cor}, A_1^n] = d[(A_1A_3)_2, A_1^n] = d[(A_1A_3)_3, A_1^n] = 1. \tag{9.16}$$

(iii) Finally, the distances between the special shock nodes are also simplified as :

$$d[N_{cor}, (A_1A_3)_2] = 2/8, \quad d[N_{cor}, (A_1A_3)_3] = 6/12, \quad d[(A_1A_3)_2, (A_1A_3)_3] = 4/12. \tag{9.17}$$

Table 9.6 summarizes the above results. Finally, the structural node compatibility $(\mathcal{N}_s)$ is defined as the compliment of the node type difference d, *i.e.*,

$$\mathcal{N}_s[N_a, \bar{N}_i] = 1 - d[N_a, \bar{N}_i], \tag{9.18}$$

such that $\mathcal{N}_s = 1$ if the shock types are identical, and $\mathcal{N}_s = 0$ if the shock types are completely different.

Parametric node compatibility ($\mathcal{N}_p$)

We consider *three* main terms between two medial nodes $N_a \in \mathcal{MS}$ and $\bar{N}_i \in \overline{\mathcal{MS}}$ in the *parametric* node compatibility measure: *(i)* the shock node *radius* r, *(ii)* the *gradient* of radius (∇r) along the incident shock curves, and *(iii)* the *angles* a of incident sheet corners between incident shock curves.

	A_1A_3	A_1^4	$A_1^2A_3$	A_1^5	...	$A_1^mA_3$	A_1^n	N_{cor}	$(A_1A_3)_2$	$(A_1A_3)_3$
A_1A_3	0	3/4	1/5	4/5	...	$\frac{m-1}{m+3}$	$\frac{n-1}{n}$	1/5	4/8	8/12
A_1^4		0	3/5	1/5	...	$\frac{m-1}{m+3}$	$\frac{n-4}{n}$	5/5	1	1
$A_1^2A_3$			0	3/5	...	$\frac{m-2}{m+3}$	$\frac{n-2}{n}$	2/5	3/8	7/12
A_1^5				0	...	$\frac{m-3}{m+3}$	$\frac{n-5}{n}$	5/5	1	1
...					0	...	...	...	...	...
$A_1^mA_3$						0	$\frac{max(\lvert n-m\rvert,3)}{max(n,m+3)}$	$\frac{m}{max(m+3,5)}$	$\frac{\lvert m-2\rvert+3}{max(m+3,8)}$	$\frac{\lvert m-3\rvert+6}{max(m+3,12)}$
A_1^n							0	1	1	1
N_{cor}								0	2/8	6/12
$(A_1A_3)_2$									0	4/12
$(A_1A_3)_3$										0

Table 9.6: Difference d between *special* and general shock nodes reflecting the structural (type) compatibility between them. Portion of Table 9.5 is replicated here for convenience.

1. *Radius* difference between N_a and $\bar{N}_i$.
2. *Gradient* of radius along incident shock curves: Compare the radius gradient ∇r along each incident curve of N_a and $\bar{N}_i$. Since there exist numerous incident curves at each medial node (for example, there are four curves at an A_1^4 node), we simply take the maximum and minimum measures $(\nabla r_{max}, \nabla r_{min})$. There are two additional reasons motivating this: *(i)* First, the measurements in between the maximum and minimum are less salient and less robust to compare. *(ii)* Second, the A_1A_3 node has only two incident curves while other nodes has more curves; thus we can at most select two distinct curves in a general setup.
3. *Angle* between incident shock curves: For the node N_a, compute the angle of the sheet corner S_{bac} between two incident curves C_b and C_c. Similarly for $\bar{N}_i$, compute the angle of sheet corner $\bar{S}_{jik}$ between $\bar{C}_j$. Since there exists numerous sheet corners at each medial node (for example, there are six sheets intersecting at an A_1^4 node), again we only take the maximum and minimum measures (a_{max}, a_{min}).

The parametric node compatibility is defined as:

$$\begin{aligned}\mathcal{N}_p[N_a, \bar{N}_i] = 1 &- w_r^n \cdot d_{sqr}^n[r(N_a), r(\bar{N}_i)] \\ &- \frac{w_g^n}{2} \cdot d_{sqr}^n[\nabla r_{max}(N_a), \nabla r_{max}(\bar{N}_i)] - \frac{w_g^n}{2} \cdot d_{sqr}^n[\nabla r_{min}(N_a), \nabla r_{min}(\bar{N}_i)] \\ &- \frac{w_a^n}{2} \cdot d_{sqr}^n[a_{max}(N_a), a_{max}(\bar{N}_i)] - \frac{w_a^n}{2} \cdot d_{sqr}^n[a_{min}(N_a), a_{min}(\bar{N}_i)]. \end{aligned} \tag{9.19}$$

where the weighting constants $w_r^n = 0.5$ (radius), $w_g^n = 0.25$ (gradient of radii), $w_a^n = 0.25$ (incident curves) specify the relative importance between different measurements.

Finally, the first-order node compatibility is defined as the multiplication of the *structure* compatibility and the *parametric* compatibility:

$$\mathcal{N}[N_a, \bar{N}_i] = \mathcal{N}_s \cdot \mathcal{N}_p. \tag{9.20}$$

To illustrate the effect of the above compatibility design, we investigate an example in matching two bone shapes in Figure 9.2 to examine the compatibility values throughout this chapter. A set of manually selected pairs of matching nodes are used as ground truth. The resulting node compatibility is shown in Table 9.7. Note that the node compatibility produces many correct matches but also some

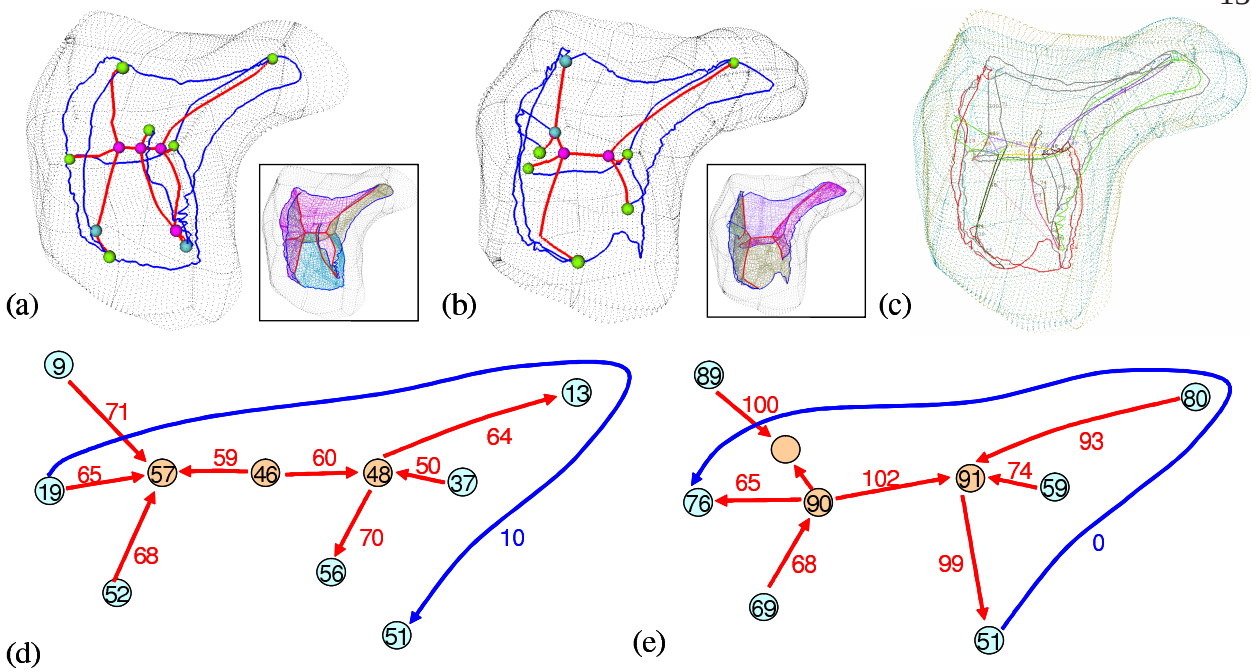


Figure 9.2: The matching of the $\mathcal{MS}$ hypergraphs of two carpal bones (female hamate, Data set from Dr. Crisco, RI Hospital). (a) is the left hand and (b) is the right hand. (c) gives an example matching. (d) and (e) shows a manual matching, which serves as the ground truth to validate our approach.

errors, due to its locality (lack of enough information), *e.g.*, the matching of $N52$ to $n69$ has low compatibility.

9.3.2 The second-order *link (curve)* compatibility measure ($\mathcal{L}$)

The compatibility between two shock curves $C_{ab} \in \mathcal{MS}$ and $\bar{C}_{ij} \in \overline{\mathcal{MS}}$ is considered *second* order in our hypergraph matching framework and is designed *intrinsically* to compare the two given curves. Again we consider both *(i)* the **structural** similarity and *(ii)* the **parametric** similarity in designing the final compatibility between the two curves. Note that the *orientation* of the curves should be explicitly compared, *i.e.*, matching $(C_{ab}, \bar{C}_{ij})$ is different than matching $(C_{ab}, \bar{C}_{ji})$, to better filter out unsuitable matches.

Structural link (curve) compatibility ($\mathcal{L}_s$)

The structural compatibility $\mathcal{L}_s$ checks the following *three* terms in matching two medial curves $C_{ab} \in \mathcal{MS}$ and $\bar{C}_{ij} \in \overline{\mathcal{MS}}$: *(i)* the existence of the shock curve, *(ii)* the shock curve type, and *(iii)* the two shock end node types.

1. *Existence* of shock curves: Avoid the matching if any of the shock curves C_{ab} or $\bar{C}_{ij}$ does not exist.
2. *Shock curve type*. Avoid the matching of an A_3 curve to an A_1^3 (or a degenerate A_1^n) curve.
3. *Ending shock node type*. This also considers the *orientation* of the shock curves into the matching: If $\mathcal{N}_s[N_a, \bar{N}_i]$ or $\mathcal{N}_s[N_b, \bar{N}_j]$ is low, $\mathcal{L}_s[C_{ab}, \bar{C}_{ij}]$ should be low.

Table 9.7: An example *node compatibility* table ($\mathcal{N}_{ai}$) to match two $\mathcal{MS}$ nodes of the $\mathcal{MS}$ hypergraphs of the two carpal bones in Figure 9.2. Values of the ground truth matching pairs are highlighted in parentheses (boldface in blue), which should be with high compatibilities. Several *erroneous* matches with compatibility higher than the ground true are underlined (in red).

	n51	n59	n69	n76	n80	n83	n89	n90	n91	n92	n94
N00	0.44	0.41	0.41	0.55	0.49	0.53	0.24	0.14	0.16	0.15	0.35
N09	0.57	0.68	0.67	0.68	0.73	0.56	**(0.37)**	0.10	0.12	0.11	0.22
N13	0.68	0.68	0.77	0.54	**(0.72)**	0.57	0.40	0.11	0.12	0.12	0.28
N19	0.54	0.62	0.56	**(0.69)**	0.68	0.49	0.39	0.09	0.10	0.10	0.22
N25	0.59	0.67	0.73	0.45	0.63	0.50	0.34	0.10	0.11	0.10	0.23
N37	0.55	0.58	0.59	0.72	0.66	0.52	0.39	0.09	0.10	0.10	0.23
N46	0.12	0.10	0.13	0.11	0.12	0.12	0.00	0.75	0.69	0.71	0.15
N48	0.14	0.12	0.11	0.12	0.11	0.13	0.00	0.71	**(0.86)**	0.80	0.16
N51	0.20	0.19	0.18	0.20	0.20	0.20	0.37	0.00	0.00	0.00	0.22
N52	0.20	0.18	**(0.17)**	0.22	0.19	0.18	0.34	0.00	0.00	0.00	0.19
N56	0.19	0.14	0.15	0.16	0.14	0.18	0.00	0.44	0.53	0.55	0.14
N57	0.12	0.10	0.10	0.11	0.10	0.11	0.00	**(0.80)**	0.73	0.73	0.14

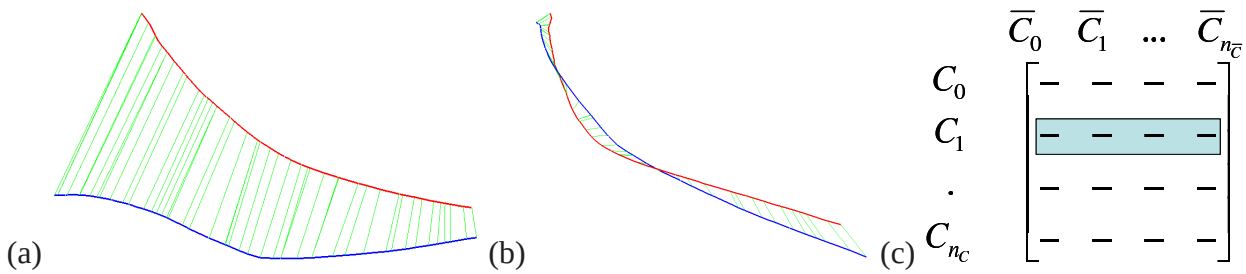


Figure 9.3: (a) Result of optimal alignment of D.P. matching of two shock curves (geometry as 3D space curves and other attributes). (b) Result of using the D.P. alignment to compute the best (R, T) to align the two curves and compute the Euclidean distance. In this example, $d^n_{ed}[C_{ab}, \bar{C}_{ij}] = 12.76$, $d^n_{Eu}[C_{ab}, \bar{C}_{ij}] = 17.83$. (c) Result stored in the D.P. result table, and the normalization of the compatibility is against the row (see text).

Specifically, the structural curve compatibility is defined as:

$$\mathcal{L}_s[C_{ab}, \bar{C}_{ij}] = \begin{cases} 0, \text{if } C_{ab} \text{ or } \bar{C}_{ij} \text{ is missing or the types are different,} \\ 1 - w^l_n \cdot d[N_a, \bar{N}_i] - w^l_n \cdot d[N_b, \bar{N}_j], \text{ otherwise,} \end{cases} \tag{9.21}$$

where the node type distance d is defined in the structural node compatibility (§ 9.3.1). The weighting parameter $w^l_n = 0.2$ penalizes the difference between the shock end node types.

Parametric link (curve) compatibility ($\mathcal{L}_p$)

We consider *three* main terms in the parametric link (curve) compatibility measure between the two medial curves C_{ab} and $\bar{C}_{ij}$: *(i) integration of shock radius* along the curves (to reflect the corresponding shape volumes), *(ii) edit distance* d_{ed} optimizing the elastic matching between the two curves, and *(iii) Euclidean distance* d_{Eu} between them using the correspondence from the above edit distance matching result, detailed as follows.

1. *Integration of shock radius* (to approximate corresponding shape volume),

$$V(C) = \int_{s \in C} r \cdot ds \approx \sum_{k=1}^{n_{sample}} r(C_k). \tag{9.22}$$

The normalization is done by dividing a maximum value in $\mathcal{MS}$ and $\overline{\mathcal{MS}}$, such that:

$$0 \leq V^n(C_{ab}), V^n(\bar{C}_{ij}) \leq 1 \tag{9.23}$$

2. *Edit distance* $d_{ed}[C_{ab}, \bar{C}_{ij}]$ by the *elastic* matching between the two curves C_{ab} and $\bar{C}_{ij}$ to get their *optimal* alignment minimizing the following energy terms, computed via *dynamic programming* (D.P.) [168, 169], Figure 9.3(a).

   - *stretching* difference $ds(C_p, C_{p\text{-}1}, \bar{C}_q, \bar{C}_{q\text{-}1})$, where p, q are indices of the fine-scale mesh vertices along the curves,
   - *bending* difference $dt(C_p, C_{p\text{-}1}, \bar{C}_q, \bar{C}_{q\text{-}1})$,
   - shock *radius* difference $dr(C_p, C_{p\text{-}1}, \bar{C}_q, \bar{C}_{q\text{-}1})$, and
   - the *angle* difference $da(C_p, C_{p\text{-}1}, \bar{C}_q, \bar{C}_{q\text{-}1})$ between incident sheets of two cases: *(i)* for an A_3 curve, we match the "side" of sheet, *i.e.*, the angle between the curve normal and the sheet tangent direction, *(ii)* for an A_1^3 (or the degenerate A_1^n) curve, we match the angles between the three or more incident sheets,

   Specifically, the two shock curves is matched both as space curves (in their elastic bending and stretching terms) and as a joined skeleton-boundary matching to approximate their similarity discussed in Chapter 8. [3]

$$\min \quad \left\{ [ds + dr] + w_a^l \cdot [dt + da] \right\}, \tag{9.24}$$

   where the weighting constant $w_a^l = 0.1$ specifies the ratio between the *length* and *angle* measurements (the value is relative to the *scale* of shapes as in [169]).

3. (Optional) *Euclidean distance* from the correspondence (alignment) between the two curves $d_{Eu}[C_{ab}, \bar{C}_{ij}]$ using the alignment from the edit-distance matching. Specifically, we compute the optimal rotation and translation (R, T) to align C_{ab} to $\bar{C}_{ij}$, and compute the average distance between the corresponding pairs, Figure 9.3(b). [4] This term enhances the matching of *rigid* objects or shapes with slight deformations. It should be avoided for matching *deformable* objects or shapes across large variations.

* Storing the curve-matching results: the curve orientation and normalization issues.

We store the above curve matching results into *four* tables (two tables for d_{ed} and d_{Eu}, and each in two orientations) of size $n_c \times \bar{n}_c$, where n_c and $\bar{n}_c$ are the number of shock curves in $\mathcal{MS}$ and $\overline{\mathcal{MS}}$, respectively, Figure 9.3(c). In matching, we normalize the values of these tables by normalization

[3] We use an *alignment curve* α to represent the result of the correspondence between C_{ab} and $\bar{C}_{ij}$ [169, 168]. We omit details on computing the differential terms of the shock curves as space curves, *i.e.*, stretching terms and their derivatives ϕ, ϕ_s, ϕ_{ss}, bending terms and their derivatives θ, θ_s, θ_{ss}, tangent T, normal N, binomal B, curvature κ, torsion τ.

[4] Given an alignment, the optimal (R, T) can be computed in a closed form solution.

Table 9.8: An example *curve compatibility* table ($\mathcal{L}_{aibj}$) to match two $\mathcal{MS}$ curves in their regular orientation, *i.e.*, C_{ab} to $\bar{C}_{ij}$ in matching the $\mathcal{MS}$ hypergraphs of the two carpal bones in Figure 9.2. Numbers of the ground truth matching pairs are highlighted in parentheses (boldface in blue), which should be with high compatibilities. Only four *erroneous* matches with compatibility higher than the ground true are underlined (in red), which shows that our curve compatibility measure is excellent in this example.

	c00	c25	c47	c72	c74	c84	c93	c94	c95	c99	c100	c101	c102	c103	c105	c107
C01	0.00	0.00	0.00	0.00	0.60	0.27	0.19	0.00	0.78	0.39	0.38	0.40	0.59	0.79	0.52	0.68
C02	0.11	0.68	0.04	0.84	0.00	0.00	0.00	0.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C10	0.81	0.24	0.39	0.19	0.00	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C14	0.10	0.61	0.03	0.80	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C26	0.15	0.80	0.09	0.61	0.00	0.00	0.00	0.27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C33	0.66	0.45	0.61	0.37	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C39	0.00	0.00	0.00	0.00	0.51	0.23	0.15	0.00	0.75	0.40	0.41	0.42	0.46	0.54	0.50	0.48
C50	0.00	0.00	0.00	0.00	**(0.91)**	0.32	0.19	0.00	0.75	0.44	0.50	0.44	0.69	0.51	0.35	0.43
C59	0.00	0.00	0.00	0.00	0.53	0.24	0.17	0.00	0.68	0.51	0.33	0.50	0.73	0.96	0.58	0.58
C60	0.00	0.00	0.00	0.00	0.48	0.23	0.15	0.00	0.61	0.48	0.31	0.49	0.71	0.92	0.63	0.61
C64	0.00	0.00	0.00	0.00	0.23	0.53	0.65	0.00	0.20	0.57	0.33	0.55	0.35	0.25	0.18	0.13
C65	0.00	0.00	0.00	0.00	0.73	0.39	0.21	0.00	0.51	0.44	0.74	0.42	0.45	0.36	0.24	0.26
C66	0.00	0.00	0.00	0.00	0.51	0.23	0.15	0.00	0.67	0.35	0.36	0.36	0.53	0.73	0.48	0.69
C67	0.00	0.00	0.00	0.00	0.51	0.23	0.15	0.00	0.67	0.35	0.36	0.36	0.53	0.73	0.48	0.69
C68	0.00	0.00	0.00	0.00	0.54	**(0.39)**	0.23	0.00	0.40	0.38	0.72	0.36	0.35	0.29	0.19	0.20
C69	0.00	0.00	0.00	0.00	0.40	0.37	0.24	0.00	0.32	0.63	0.47	0.58	0.51	0.39	0.18	0.20
C70	0.00	0.00	0.00	0.00	0.43	0.43	0.24	0.00	0.31	0.65	0.48	0.56	0.50	0.39	0.18	0.19
C71	0.00	0.00	0.00	0.00	0.59	0.72	0.41	0.00	0.41	0.36	**(0.69)**	0.32	0.34	0.26	0.14	0.18

against the minimum/maximum values of each row (Figure 9.3(c)), denotes as d^n (to obtain values between $(0, 1)$). [5]

The parametric curve compatibility is defined as:

$$\mathcal{L}_p[C_{ab}, \bar{C}_{ij}] = 1 - w_d^l \cdot d_{ed}^n[C_{ab}, \bar{C}_{ij}] - w_e^l \cdot d_{Eu}^n[C_{ab}, \bar{C}_{ij}] - w_v^l \cdot \left|V^n(C_{ab}) - V^n(\bar{C}_{ij})\right|, \quad (9.25)$$

where the weighting constants $w_d^l = 0.33$ (edit distance), $w_e^l = 0.33$ (Euclidean distance), and $w_v^l = 0.34$ (integration of radius/volume).

Finally, the second-order link (curve) compatibility is defined as:

$$\mathcal{L}[C_{ab}, \bar{C}_{ij}] = \mathcal{L}_s \cdot \mathcal{L}_p. \quad (9.26)$$

Tables 9.8 and 9.9 demonstrates excellent performance of this compatibility design in matching medial curves. We found the curve compatibility $\mathcal{C}_L$ provides strong clue and improve the $\mathcal{GA}$ matching significantly.

9.3.3 The third-order *sheet (corner)* compatibility measure ($\mathcal{C}$)

The compatibility between two sheet corners $C_{aibjck}[S_{abc}, \bar{S}_{ijk}]$ takes two shock sheet $S \in \mathcal{MS}$ and $\bar{S} \in \overline{\mathcal{MS}}$ at their corners S_{abc} and $\bar{S}_{ijk}$, respectively, and estimate the compatibility between them by considering the **structural** similarity and the **parametric** compatibilities. Note that the *orientation* of the sheet corners should be explicitly compared, *i.e.*, matching $(C_{abc}, \bar{C}_{ijk})$ is different than matching $(C_{abc}, \bar{C}_{kji})$ in the (nested) $\mathcal{GA}$ iterations, to better filter out unsuitable matches.

[5] This normalization is to emphasize the difference of measurement. It is better than the normalization against a global maximum value.

Table 9.9: An example *'flipped'-curve compatibility* table ($\mathcal{L}_{ajbi}$) with all the second curves in $\overline{\mathcal{MS}}$ flipped in orientation during the matching, (*i.e.*, matching C_{ab} to $\bar{C}_{ji}$ in matching the $\mathcal{MS}$ hypergraphs of the two carpal bones in Figure 9.2. Numbers of the ground truth matching pairs are highlighted in parentheses (boldface in blue), which should be with high compatibilities. There is no erroneous match whose compatibility measure is higher than the ground true ones (100% correct).

	c00	c25	c47	c72	c74	c84	c93	c94	c95	c99	c100	c101	c102	c103	c105	c107
C01	0.00	0.00	0.00	0.00	0.42	0.19	0.13	0.00	0.51	0.59	0.34	0.59	0.60	0.79	0.68	0.52
C02	0.11	0.59	0.04	0.72	0.00	0.00	0.00	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C10	**(0.81)**	0.21	0.52	0.16	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C14	0.10	0.57	0.03	0.69	0.00	0.00	0.00	0.49	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C26	0.16	0.69	0.10	0.48	0.00	0.00	0.00	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C33	0.63	0.43	0.61	0.34	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C39	0.00	0.00	0.00	0.00	0.45	0.20	0.13	0.00	0.57	0.50	0.40	0.50	0.50	0.59	0.48	0.50
C50	0.00	0.00	0.00	0.00	0.49	0.20	0.49	0.00	0.47	0.78	0.39	0.93	0.65	0.58	0.43	0.35
C59	0.00	0.00	0.00	0.00	0.53	0.24	0.17	0.00	0.68	0.51	0.32	0.51	0.74	0.96	0.58	0.58
C60	0.00	0.00	0.00	0.00	0.50	0.23	0.15	0.00	0.66	0.48	0.31	0.49	0.69	0.92	0.61	0.63
C64	0.00	0.00	0.00	0.00	0.40	0.82	**(0.92)**	0.00	0.32	0.26	0.37	0.23	0.27	0.23	0.13	0.18
C65	0.00	0.00	0.00	0.00	0.40	0.24	0.15	0.00	0.37	0.77	0.54	**(0.87)**	0.51	0.44	0.26	0.24
C66	0.00	0.00	0.00	0.00	0.36	0.18	0.12	0.00	0.45	0.48	0.31	0.48	0.54	0.73	0.69	0.48
C67	0.00	0.00	0.00	0.00	0.36	0.18	0.12	0.00	0.45	0.48	0.31	0.48	0.54	0.73	0.69	0.48
C68	0.00	0.00	0.00	0.00	0.31	0.23	0.14	0.00	0.28	0.66	0.52	0.70	0.43	0.34	0.20	0.19
C69	0.00	0.00	0.00	0.00	0.46	0.47	0.26	0.00	0.36	0.44	0.60	0.41	0.46	0.32	0.20	0.18
C70	0.00	0.00	0.00	0.00	0.50	0.58	0.29	0.00	0.35	0.47	0.61	0.48	0.46	0.36	0.19	0.18
C71	0.00	0.00	0.00	0.00	0.28	0.38	0.22	0.00	0.25	0.65	0.48	0.74	0.40	0.29	0.18	0.14

Structural sheet (corner) compatibility ($\mathcal{C}_s$)

The structural sheet (corner) compatibility estimates *(i)* the *existence* of the corners, *(ii)* the boundary shock *curve type*, *(iii)* the ending shock *node types*, detailed as follows.

1. *Existence* of sheet corners.
2. *Boundary shock curve type*: Avoid the cases if either $(C_{ab}, \bar{C}_{ij})$ or $(C_{bc}, \bar{C}_{jk})$ are of different types, which also help to match the 'orientation' (permutation) of the indices at the corner.
3. *Ending shock node type*: Avoid the cases if either $(N_a, \bar{N}_i)$, $(N_b, \bar{N}_j)$, or $(N_c, \bar{N}_k)$ are of different types, which also help to match the 'orientation' (permutation of the indices) at the corner.

The structural sheet (corner) compatibility is defined as:

$$\mathcal{C}_s(S_{abc}, \bar{S}_{ijk}) = \begin{cases} 0, \text{ if } S_{abc} \text{ or } \bar{S}_{ijk} \text{ is missing, or } (C_{ab}, \bar{C}_{ij}) \text{ or } (C_{bc}, \bar{C}_{jk}) \text{ are of diff. types,} \\ 1 - w^c_{n1} \cdot d[N_b, \bar{N}_j] - w^c_{n2} \cdot d[N_a, \bar{N}_i] - w^c_{n2} \cdot d[N_c, \bar{N}_k], \text{ otherwise,} \end{cases} \tag{9.27}$$

where the weighting constants $w^c_{n1} = 0.8$ (corner angle) and $w^c_{n2} = 0.1$ (volume). We make $w^c_{n1} + 2w^c_{n2} = 1$ to completely determine the structural compatibility around a sheet corner, since there is no structural information from the medial sheets (they are all of A^2_1 type).

Parametric sheet (corner) compatibility ($\mathcal{C}_p$)

The **parametric** sheet corner compatibility considers: *(i)* the shock radius r of corner vertex, *(ii)* the gradient of shock radius (∇r) along the sheet (bisector curve of the corner) *(iii)* the corner angle, *(iv)* the corresponding shape volume of the sheet, detailed below.

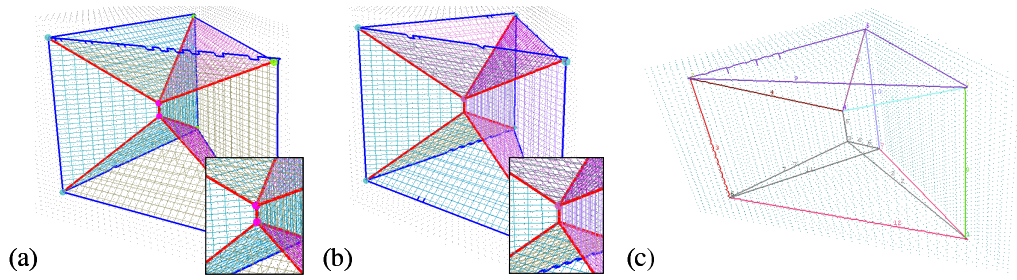


Figure 9.4: Match two prisms before (a) and after (b) applying an A_1^5 curve contract transform at the center. The matching result is in (c). This figures illustrates that adding virtual links could successfully simulate the transform.

1. *Shock radius* of the corner vertices (N_b and $\bar{N}_j$).
2. *Radius gradient* along the sheet bisector curves, which can be approximated by the average of the radius gradients along the two incident curves of each corner, *i.e.*, $(C_{ba}, \bar{C}_{bc})$ and $(\bar{C}_{ji}, \bar{C}_{jk})$, computed similar as in § 9.3.1.
3. *Corner angles* between $\angle(S^n_{abc})$ and $\angle(\bar{S}^n_{ijk})$.
4. *Integration of shock radius* (to approximate corresponding shape volume). This can be approximated by summing up the radius of all sheet elements:

$$V(S) = \iint_{s \in S} r \cdot ds \approx \sum_{k=1}^{n_{sample}} r(S_k). \tag{9.28}$$

The parametric sheet (corner) compatibility is defined as:

$$\begin{aligned} \mathcal{C}_p(S_{abc}, \bar{S}_{ijk}) = 1 - w^c_r \cdot d^n_{sqr}[r(N_b), r(\bar{N}_j)] - w^c_g \cdot d^n_{sqr}[\nabla r(S_{abc}), \nabla r(\bar{S}_{ijk})] \\ - w^c_a \cdot d^n_{sqr}[\angle^n_{abc}, \angle^n_{ijk}] - w^c_v \cdot d^n_{sqr}[V^n(S), V^n(\bar{S})]. \end{aligned} \tag{9.29}$$

where the weighting parameters $w^c_r = 0.2$ (radius), $w^c_g = 0.3$ (radius gradient), $w^c_a = 0.3$ (angle), and $w^c_v = 0.2$ (volume).

Finally, the third-order sheet (corner) compatibility is defined as:

$$\mathcal{C}[S_{abc}, \bar{S}_{ijk}] = \mathcal{C}_s \cdot \mathcal{C}_p. \tag{9.30}$$

Table 9.10 examines the performance of the proposed sheet (corner) compatibility design. We found it provides strong clues in selecting correct matches during the hypergraph matching.

9.4 Virtual Links to Handle Remaining $\mathcal{MS}$ Transitions

* Use of "virtual links" to match graph structure changes across transitions.

In the previous section we have improved the $\mathcal{GA}$ hypergraph matching significantly by improving its component-to-component matching. However, there is an important factor not yet addressed in the graph matching, that is the **structural coherence** between two graphs. This is precisely the (optimal) edit-distance matching in Chapter 8 should achieve, to explore the space of the possible edits across

Table 9.10: An example *(sheet) corner compatibility* table ($\mathcal{C}_{aibjck}$) in matching the $\mathcal{MS}$ hypergraphs of the two carpal bones in Figure 9.2 (values in $1/100$). Numbers of the ground truth matching pairs are highlighted in parentheses (boldface in blue), which should be with high compatibilities. Several *erroneous* matches with compatibility higher than the ground true are underlined (in red).

	c76	c91	c91	c91	c89	c91	c91	c89	c90	c90	c90	c51	c90	c90	c89	c91	c92	c92	c94	c59	c69	c80	c94	c92	c90	c83
C[00-46-48]	0	48	53	59	0	52	49	0	47	55	53	0	54	56	0	52	45	53	0	0	0	0	15	57	60	0
C[00-46-56]	0	40	47	46	0	37	49	0	39	42	46	0	51	48	0	52	44	56	0	0	0	0	15	41	53	0
C[00-46-57]	0	52	63	67	0	48	53	0	50	63	56	0	59	62	0	61	40	55	0	0	0	0	18	58	70	0
C[00-51-56]	21	0	0	0	13	0	0	17	0	0	0	22	0	0	20	0	0	0	27	16	19	18	0	0	0	0
C[09-52-57]	28	0	0	0	25	0	0	31	0	0	0	20	0	0	34	0	0	0	33	21	23	28	0	0	0	0
C[09-57-19]	0	68	74	63	0	70	63	0	73	66	62	0	62	68	0	62	41	52	0	0	0	0	16	46	46	0
C[09-57-46]	0	54	62	66	0	57	64	0	58	69	67	0	64	64	0	70	38	49	0	0	0	0	22	43	57	0
C[09-57-52]	0	54	62	60	0	59	68	0	60	63	69	0	64	59	0	67	36	49	0	0	0	0	16	43	46	0
C[13-25-52]	41	0	0	0	26	0	0	29	0	0	0	55	0	0	26	0	0	0	21	48	53	43	0	0	0	0
C[13-48-37]	0	68	69	57	0	**(74)**	58	0	75	61	58	0	55	60	0	58	46	52	0	0	0	0	14	43	41	0
C[13-48-46]	0	53	60	65	0	57	66	0	59	71	69	0	64	63	0	**(73)**	40	52	0	0	0	0	22	43	60	0
C[13-48-56]	0	56	**65**	63	0	61	71	0	63	68	71	0	69	65	0	69	44	47	0	0	0	0	20	39	50	0
C[19-51-56]	27	0	0	0	25	0	0	25	0	0	0	26	0	0	29	0	0	0	23	26	25	26	0	0	0	0
C[19-57-46]	0	53	60	64	0	56	73	0	58	68	74	0	**(70)**	64	0	73	42	49	0	0	0	0	22	41	50	0
C[19-57-52]	0	63	64	55	0	74	55	0	**(66)**	58	55	0	54	57	0	56	46	53	0	0	0	0	13	46	40	0
C[25-13-48]	50	0	0	0	30	0	0	34	0	0	0	73	0	0	29	0	0	0	28	59	67	52	0	0	0	0
C[25-52-37]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
C[25-52-57]	0	7	7	10	0	8	12	0	8	12	12	0	11	10	0	13	8	5	0	0	0	0	29	2	5	0
C[37-48-46]	0	54	63	61	0	57	**(70)**	0	58	62	67	0	67	64	0	71	44	46	0	0	0	0	22	38	47	0
C[37-48-56]	0	**(61)**	58	57	0	69	55	0	59	58	55	0	56	60	0	56	50	52	0	0	0	0	19	43	41	0
C[37-52-57]	28	0	0	0	30	0	0	34	0	0	0	20	0	0	36	0	0	0	24	21	21	25	0	0	0	0
C[46-00-51]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56
C[46-48-56]	0	43	49	**54**	0	40	52	0	41	51	56	0	59	52	0	51	36	44	0	0	0	0	14	61	68	0
C[46-56-48]	0	33	37	40	0	35	33	0	32	37	35	0	37	39	0	38	43	41	0	0	0	0	11	57	56	0
C[46-56-51]	0	38	36	30	0	32	31	0	31	28	30	0	33	31	0	34	35	45	0	0	0	0	8	45	42	0
C[46-57-52]	0	43	51	50	0	40	41	0	42	**(47)**	43	0	45	46	0	46	33	51	0	0	0	0	9	68	63	0
C[48-37-52]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47
C[48-46-56]	0	42	44	48	0	40	42	0	42	45	42	0	48	51	0	39	42	60	0	0	0	0	11	56	60	0
C[48-46-57]	0	33	39	42	0	32	39	0	31	39	40	0	42	40	0	48	31	38	0	0	0	0	15	49	64	0
C[48-56-51]	0	43	50	44	0	41	47	0	42	41	44	0	48	46	0	49	34	42	0	0	0	0	9	54	59	0
C[51-19-57]	**(62)**	0	0	0	27	0	0	28	0	0	0	47	0	0	30	0	0	0	28	48	45	56	0	0	0	0
C[52-09-57]	58	0	0	0	22	0	0	26	0	0	0	52	0	0	30	0	0	0	36	53	57	75	0	0	0	0
C[56-46-57]	0	52	56	64	0	55	50	0	48	57	53	0	57	60	0	53	49	61	0	0	0	0	15	57	54	0

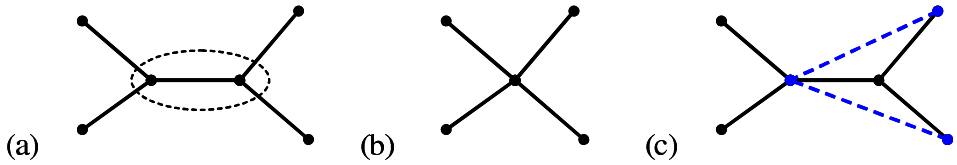


Figure 9.5: (a) Group the two nodes into a single node. (b) After contract transform. (c) Add two virtual links to one of the two nodes.

transitions to reach a common topology for matching. In this section, we provide an approximation to "simulate" such capability (to match across structural changes), under the limitation of the graph-matching methods such as the $\mathcal{GA}$.

The introducing of the virtual links provides a way to match the $\mathcal{MS}$ hypergraph robustly across transitions, *e.g.*, allow to match a high-order node into a group of generic low-order nodes (such as the 2 sides across the A_1^5 transition, Refer to Figure 9.4 for an example). Since the $\mathcal{GA}$ handles missing and extra nodes (by the slack variables), we can enforce the matching of possible transitions not yet handled by "simulating" the transition by connecting the nodes that will be brought together in the transitions (should it perform) using the virtual links. We only add virtual links to the local configurations that warrant high possibility of transitions, *i.e.*, configurations that are with low $\mathcal{MS}$ transform costs. In implementation, we can simply use the procedure for the *detection* of transforms detailed in Chapter 5 to look for such candidate positions to add virtual links (by allowing a larger transform cost threshold). The adding of virtual links essentially complements the one-to-one assignment nature of the graph-matching approach in handling the graph structural changes.

* Virtual links to simulate the set of $\mathcal{MS}$ transforms in Chapter 5.

The idea of virtual links is applicable to all $\mathcal{MS}$ transforms we defined in Chapter 5. *(i)* For the *contract* type of transforms: Figure 9.5 shows an example of adding virtual links for an contract type transform in matching 2D shock graphs. The contract type transforms in 3D can be processed similarly. *(ii)* For the *splice* transforms, there is no need to handle the A_1A_3-I splice transform, since the extra nodes will be matched into the slack during graph matching and not causing a problem. The case of the $A_1^2A_3$-II splice can be handled by adding a virtual link connecting the remaining A_1^3 curve. *(iii)* For the *merge* type of transforms: all node-node and node-curve types of merge transforms can be handled with the virtual links (detailed omitted), while the two curve-curve merge transforms needs an introducing of a **virtual node** together with four virtual links to simulate the change in structure. *(iv)* Finally, we can safely skip handling the gap and loop transforms, since they are not significant at this stage.

* Matching the virtual links (curves) in the $\mathcal{GA}$ matching.

We now describe how we define the compatibility measure in matching the virtual links (curves). *(i)* For the parametric compatibility (such as the bending and stretching cost), we refer to its underlying (actual) curves, *i.e.*, simply treat the virtual curve as a combination of its underlying curves. For the third-order corner compatibility, we also refer to the proper underlying actual corner for measurements. *(ii)* For the structural compatibility, we let the rib type curves dominate over the axial types, based on an observation that the contract transform near a rib curve *always* keeps the rib and removes the axial. In other words, for a virtual curve C^v, if any of its underlying (actual) curve C is an A_3 rib, it is treated as a rib; otherwise it is treated as an axial. Thus, all virtual curves can be treated equally as actual $\mathcal{MS}$ curves (and not distinguishable) in the $\mathcal{GA}$ matching.

We found the above notion of virtual links/nodes provides noticeable increase in matching rates in our experiments.

9.5 Computing the (Post-Matching) Similarity Measure

We have described our hypergraph matching algorithm that shall produce an assignment between graph nodes as an output. This section aim to derive a **similarity measure** computed from the output matching results. The idea is to use the final assignment matrix $\mathbf{M}$ together with the compatibility equations defined in § 9.3 to compute the final similarity, which indicates how similar the two shapes are in the resulting assignment.

* Compute the *total* similarity (energy E) of a matching (assignment).

We estimate the **total similarity energy** between two hypergraphs $\mathcal{MS}$, $\overline{\mathcal{MS}}$ (the larger, the similar) given an assignment $\mathbf{M}$ as follows:

$$E[\mathcal{MS}, \overline{\mathcal{MS}}, \mathbf{M}] = w^N \cdot E_N + w^L \cdot E_L + w^C \cdot E_C. \tag{9.31}$$

where $w^N = 0.3$, $w^L = 0.5$, and $w^C = 0.4$ are the weighting constants for the respective hypergraph components, and the node (E_N), curve (E_L), and sheet corner (E_C) energies are defined as:

$$\begin{cases} E_N = \sum_{a=1}^{A}\sum_{i=1}^{I} \mathbf{M}_{ai} \cdot \mathcal{N}_{ai}[N_a, \bar{N}_i], \\ E_L = \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathcal{L}_{aibj}[C_{ab}, \bar{C}_{ij}], \\ E_C = \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathbf{M}_{ck} \cdot \mathcal{C}_{aibjck}[S_{abc}, \bar{S}_{ijk}], \end{cases} \tag{9.32}$$

where $\mathcal{N}_{ai}$, $\mathcal{L}_{aibj}$, $\mathcal{C}_{aibjck}$ are defined in Equations 9.20, 9.26, and 9.30, respectively.

* Compute the *normalized* similarity (energy E^n).

Our initial implementation of Eq. (9.31) revealed that the energy defined in this way varies with the number of nodes and links in each hypergraph, such that the similarity between pairs of shapes could not be universally compared. This motivates a *normalization* of each component of the energy functional by the maximum possible value for each case, so that $0 \leq E^n \leq 1$, We define the **normalized similarity energy** E^n as motivated from a similar approach in [175] as *normalization by the maximal structural similarity energy*:

$$\begin{aligned} E^n[\mathcal{MS}, \overline{\mathcal{MS}}, \mathbf{M}] &= w^N \cdot E_N^n + w^L \cdot E_L^n + w^C \cdot E_C^n \\ &= w^N \cdot \frac{E_N}{E_N^{max}} + w^L \cdot \frac{E_L}{E_L^{max}} + w^C \cdot \frac{E_C}{E_C^{max}}. \end{aligned} \tag{9.33}$$

where the **maximal** node/curve/sheet energies ($E_N^{max}/E_L^{max}/E_C^{max}$) respectively are defined as:

$$\begin{cases} E_N^{max} = \sum_{a=1}^{A}\sum_{i=1}^{I} \mathbf{M}_{ai} \cdot \mathcal{N}_{ai}^{max}[N_a, \bar{N}_i] = \sum_{a=1}^{A}\sum_{i=1}^{I} \mathbf{M}_{ai}, \\ E_L^{max} = \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathcal{L}_{aibj}^{max}[C_{ab}, \bar{C}_{ij}], \\ E_C^{max} = \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} \mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathbf{M}_{ck} \cdot \mathcal{C}_{aibjck}^{max}[S_{abc}, \bar{S}_{ijk}]. \end{cases} \tag{9.34}$$

and where the **maximal structural similarity** terms between nodes/curves/corners are defined as:

$$\begin{cases} \mathcal{N}_{ai}^{max}(N_a, \bar{N}_i) = & 1. \\ \mathcal{L}_{aibj}^{max}(C_{ab}, \bar{C}_{ij}) = & \begin{cases} 0, \text{ if } C_{ab} \text{ or } \bar{C}_{ij} \text{ is missing,} \\ 1, \text{ otherwise.} \end{cases} \\ \mathcal{C}_{aibjck}^{max}(S_{abc}, \bar{S}_{ijk}) = & \begin{cases} 0, \text{ if } C_{ab}, C_{bc}, \bar{C}_{ij}, \bar{C}_{jk}, S_{abc}, \bar{S}_{ijk} \text{ is missing,} \\ 1, \text{ otherwise.} \end{cases} \end{cases} \tag{9.35}$$

In summary, we use the following energy to measure the **normalized similarity** between shapes:

$$E^n[\mathcal{MS}, \overline{\mathcal{MS}}, \mathbf{M}] = w^N \cdot \sum_{a=1}^{A}\sum_{i=1}^{I} \frac{\mathbf{M}_{ai} \cdot \mathcal{N}_{ai}}{\mathbf{M}_{ai}} + w^L \cdot \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I} \frac{\mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathcal{L}_{aibj}}{\mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathcal{L}^{max}_{aibj}}$$
$$+ w^C \cdot \sum_{a=1}^{A}\sum_{i=1}^{I}\sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} \frac{\mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathbf{M}_{ck} \cdot \mathcal{C}_{aibjck}}{\mathbf{M}_{ai} \cdot \mathbf{M}_{bj} \cdot \mathbf{M}_{ck} \cdot \mathcal{C}^{max}_{aibjck}}, \tag{9.36}$$

to ensure that $0 \leq E^n_{\mathcal{N}}, E^n_{\mathcal{L}}, E^n_{\mathcal{C}}, E^n \leq 1$.

While the above normalized energy E^n is suitable for a final similarity measure, there are two technical issues to solve: *(i)* First, the derivative matrix Q in the $\mathcal{GA}$ becomes complex as we use E^n. *(ii)* Second, the missing match in the *slack* rows and columns is not counted in estimating the final similarity. We describe both issues below and discuss a suitable solution for each case.

* Estimate the derivative matrix Q of the energy E.

The $\mathcal{GA}$ derivative matrix Q of the energy E in Eq. (9.31) is derived as:

$$Q_{ai}(\mathbf{M}) = \left.\frac{\partial E}{\mathbf{M}_{ai}}\right|_{\mathbf{M}=\mathbf{M}^0} = w^N \cdot \mathcal{N}_{ai} + 2w^L \cdot \sum_{b=1}^{A}\sum_{j=1}^{I} M_{bj} \cdot \mathcal{L}_{aibj} + 3w^C \cdot \sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} M_{bj} \cdot M_{ck} \cdot \mathcal{C}_{aibjck} \tag{9.37}$$

For the normalized energy E^n, the derivative matrix Q is more complicated (as follows):

$$Q_{ai}(\mathbf{M}) = \left.\frac{\partial E^n}{\mathbf{M}_{ai}}\right|_{\mathbf{M}=\mathbf{M}^0} = \frac{w^N}{E^n_N(\mathbf{M})} \cdot \mathcal{N}_{ai} + 2w^L \cdot \left\{ \frac{1}{E^n_L(\mathbf{M})} \sum_{b=1}^{A}\sum_{j=1}^{I} M_{bj} \cdot \mathcal{L}_{aibj} - \frac{E^L(\mathbf{M})}{[E^n_L(\mathbf{M})]^2} \sum_{b=1}^{A}\sum_{j=1}^{I} M_{bj} \cdot \mathcal{L}^{max}_{aibj} \right\} +$$
$$3w^C \cdot \left\{ \frac{1}{E^n_C(\mathbf{M})} \sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} M_{bj} \cdot M_{ck} \cdot \mathcal{L}_{aibjck} - \frac{E^C(\mathbf{M})}{[E^n_C(\mathbf{M})]^2} \sum_{b=1}^{A}\sum_{j=1}^{I}\sum_{c=1}^{A}\sum_{k=1}^{I} M_{bj} \cdot M_{ck} \cdot \mathcal{C}^{max}_{aibjck} \right\}, \tag{9.38}$$

which is difficult to compute in practice. Our solution is to use E in the $\mathcal{GA}$ matching algorithm (iterations), and after obtaining the matching result, use E^n to compute the final similarity.

* Penalize non-matches in the slack rows and columns.

Note that the above energy definition is based on an assumption that al nodes in $\mathcal{MS}$ and $\overline{\mathcal{MS}}$ match, *i.e.*, the *slack* row and column are *empty*. However, this is not the case in practice, in particular when matching a large graph to a small one, resulting in many mismatches in the slacks. We propose to penalize the slack row and column with compatibility **zero**. As a result, for the perfect case where both the slack row/column are empty, the final similarity is unaltered; and for the worst case if both the slack row/column are with value 1, the final similarity is 0 (a full mismatch).

9.6 Summary and Experimental Results

* The graduated assignment $\mathcal{MS}$ hypergraph matching algorithm.

The main steps of the the proposed $\mathcal{MS}$ matching approach, the **graduated assignment hypergraph matching** (GAHM) algorithm, is summarized as follows.

Table 9.11: Suggested weighting constants in matching the $\mathcal{MS}$ hypergraphs.

Category	Def. value	Description
$\mathcal{GA}$ parameters	w^N = 0.3	weight of $\mathcal{N}_{ai}$ in Q_{ai} and $E(\mathbf{M})$
	w^L = 0.5	weight of $\mathcal{L}_{aibj}$ in Q_{ai} and $E(\mathbf{M})$
	w^C = 0.4	weight of $\mathcal{C}_{aibjck}$ in Q_{ai} and $E(\mathbf{M})$
Node compatibility parameters	w^n_r = 0.5	shock radius r
	w^n_g = 0.25	gradient of shock radius
	w^n_a =0.25	sheet corner angle
Curve compatibility parameters	w^l_d = 0.33	edit distance
	w^l_e = 0.33	Euclidean distance
	w^l_v = 0.34	$\sum r$ to approx. volume
	w^l_n = 0.2	ending node of the curve
	w^l_a = 0.1	angle *v.s.* length measurement
Sheet compatibility parameters	w^c_v = 0.2	$\sum r$ to approx. volume
	w^c_a = 0.3	sheet corner angle
	w^c_r = 0.3	shock radius r
	w^c_g = 0.2	gradient of shock radius
	w^c_{n1} = 0.8	main corner vertex type
	w^c_{n2} = 0.1	other corner vertex type

1. Compute the dynamic programming (D.P.) cost for the *four* tables of size $n \times m$, namely, two tables for the edit-distance d_{ed}, Euclidean distance d_{Eu} in matching all pairs of medial curves of $\mathcal{MS}$ and $\overline{\mathcal{MS}}$, and the additional two tables for the same combination of curve pairs with the second curve flipped in orientation, *i.e.*, d^f_{ed} for the edit-distance and d^f_{Eu} for the Euclidean distance.
2. Perform the $\mathcal{GA}$ matching as described in the pseudo code in Table 9.1 using the energy E, derivative Q in Eq. (9.37), the compatibility defined in § 9.3 and the weighting parameters summarized in Table 9.11.
3. Compute the final similarity E^n in Eq. (9.36).

In handling the scale difference between shapes, we scale all shapes properly (to be with the same variance) in the pre-processing step prior to the matching. [6] After the matching, a Euclidean transform is performed to align the two objects together. [7] To provide a visual result of the matching, we color the matching nodes and curves and draw the unmatched ones in gray.

* Discussion of results.

We perform an initial experiment of this matching approach on a human wrist bone dataset (data is courtesy of Dr. Crisco, RI Hospital [140]) to see if the $\mathcal{MS}$ of the carpal bones match well to capture slight shape variations between the bones across ten patients. Figure 9.10 shows the matching result, where the carpal bone *hamate* of the left hand of the ten female patients are matched

[6] We can adopt the scaling factor into the compatibility design to make our method scale invariant. However, we note that the 'scale' matters in comparing shapes thus should be taken into account in general.

[7] The optimal rigid alignment between corresponding points can be computed by a closed form solution using *singular value decomposition* (SVD) detailed in [44].

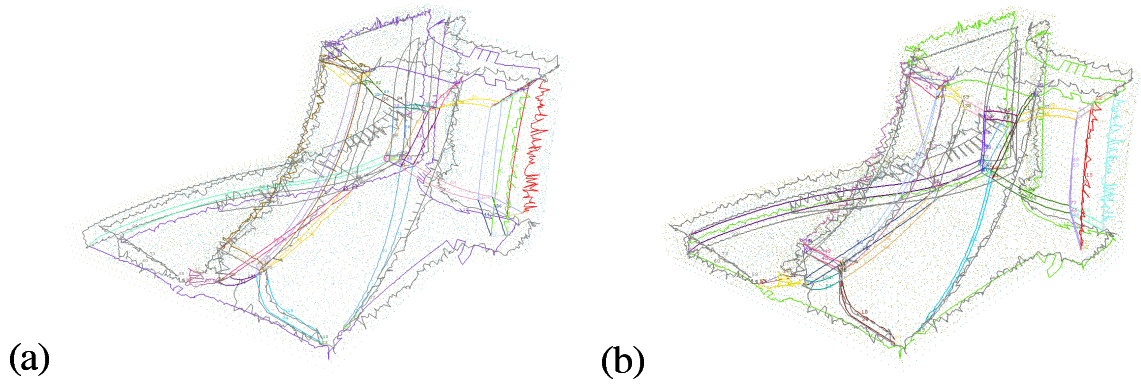


Figure 9.6: This figure shows an example of the asymmetric (but similar) matching results. Matching two different meshing A and B of the fandisk model (Data from xx), in matching A to B and B to A. Pairs of corresponding medial curves are shown in random colors.

against each other. Figure 9.9 shows five examples of the extracted $\mathcal{MS}$ hypergraph of the above hamate bones, showing that the regularized $\mathcal{MS}$ does extract the qualitative structure of the bones, which are similar across several patients. But note that even so, the matching of the bone's graph structure is still challenging. We note that the $\mathcal{MS}$ hypergraph topology from bone to bone still varies in large ranges: Refer to Figure 9.10, some bone contains only 8 node/10 curves/3 sheets in the $\mathcal{MS}$, while another contains 80 nodes/105 curves/26 sheets (an order in difference), making the hypergraph matching challenging. The $\mathcal{GA}$ works well here that the mismatches shall go into the slacks. Observe the result Figure 9.10 is promising that many of the matchings are correct, while the failed cases are mainly due to large unrecoverable structure variations.

We have also further test our matching approach on a 3D shape database (Table 9.11) of 19 shapes in 5 categories in our database, where the shapes are collected online (Cyberware, Polhemus, Stanford scanning repository, *etc.*). We aim to enlarge our database by adding shapes from the standard 3D shape databases such as the "aim@shape" repository, Princeton shape benchmark, *etc.* [8] The similarity measures of category shapes are much higher than non-category shapes, indicating the potential in applications such as in shape retrieval in registration and recognition. Note that we observe *asymmetry* matching results in the table, due to the asymmetry the way assignment is represented in the $\mathcal{GA}$ (see also Figure 9.6). [9]

* Two sources of matching errors.

We discuss two sources of errors in our matching approach. Since the method is based on a graph-based matching of regularized $\mathcal{MS}$, two types of errors can occur. *(i)* First, the match itself can be erroneous as shown in Figure 9.7, but given a sufficient number of correct correspondences, the overall matching is not typically affected much. *(ii)* Second, in the process of regularizing the scaffold, differences can still remain to degrade the matching result. If a sufficient number of correct correspondences exist, the overall registration is not affected, Figure 9.8.

[8] We use a centralize database and a system of a network of multi-core computers (VOX) to automate the $\mathcal{MS}$ computation and matching process.

[9] After the final $\mathbf{M}$ is converged (remember $\mathbf{M}$ is a doubly stochastic matrix), a clean up step is need to extract the *discrete* assignment matrix as the result of the matching [95]. We simply pick up the maximum in each row. This guarantees that each G_a will match to some $\bar{G}_i$. This is related to an assumption that we should make G a larger graph (with more nodes) and $\bar{G}$ the smaller one, so that more correspondence is available as a result in general.

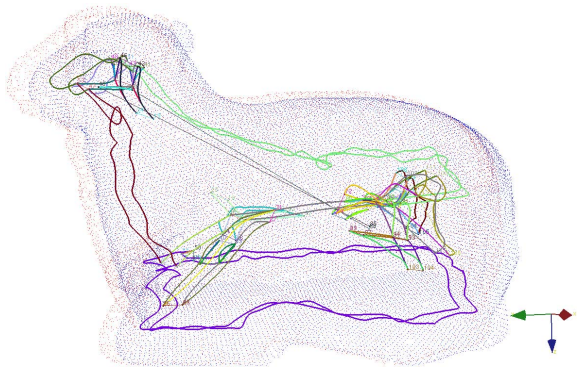


Figure 9.7: This figure illustrates an incorrect assignment as a result of erroneous matching of the $\mathcal{MS}$ for an application in registration. The assignment between corresponding nodes in $\mathcal{MS}$ and $\overline{\mathcal{MS}}$ are depicted by straight lines. Two of such lines (in gray) connects nodes of the head to the rear portion of the toy sheet model, indicating an error of matching, which creates a shift in the final alignment. Typically this does not affect the overall registration if a sufficient number of nodes are correctly assigned.

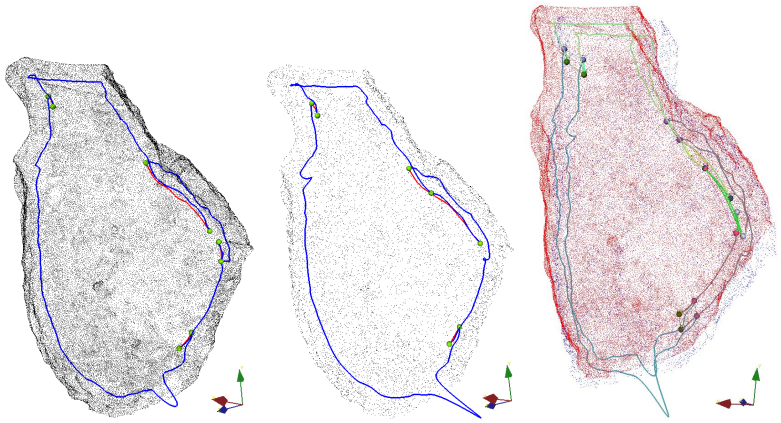


Figure 9.8: *Effect of unhandled transitions:* One pot sherd (50,473 points) is matched against its sub-sampled version (10,000 points). Both scaffolds contain relatively few (only 8) vertices, and even worse, some transitions are not fully handled. However, the proposed method gives an alignment that is sufficient close for an ICP type algorithm to obtain the optimal matching.

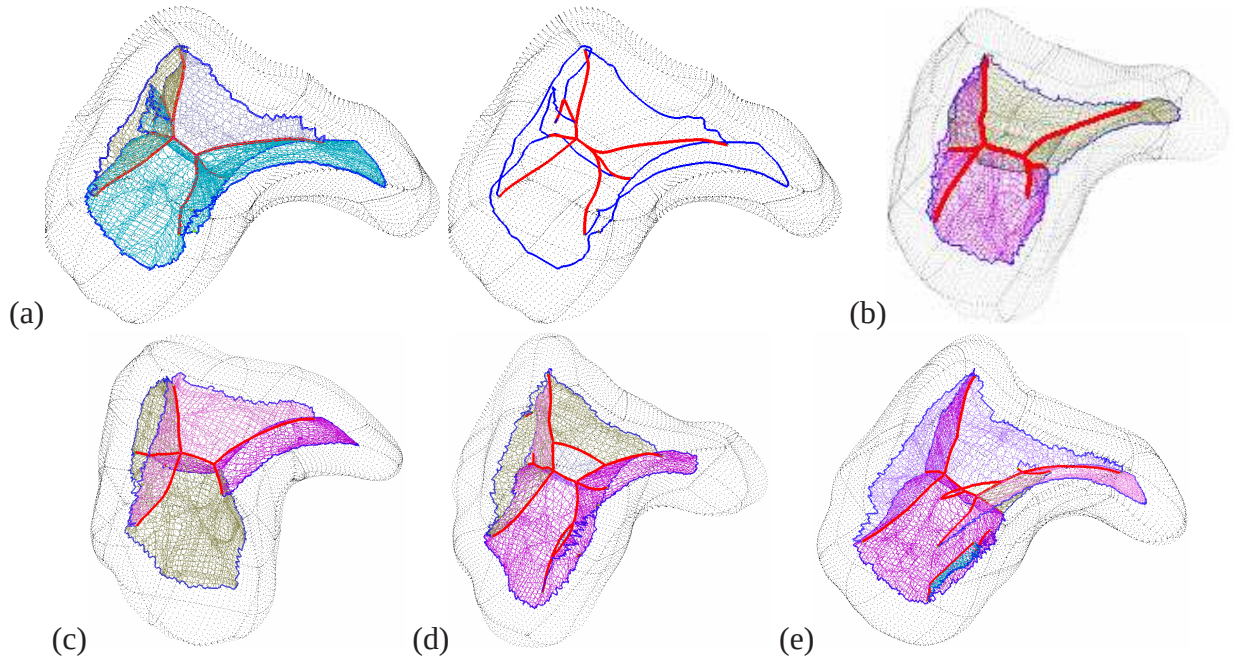


Figure 9.9: The regularized $\mathcal{MS}$ of the carpal bones (hamates) from several patients are similar in the structure, suggesting the application in shape matching and shape-based diagnosis (data is courtesy of Dr. Crisco, RI Hospital [140]).

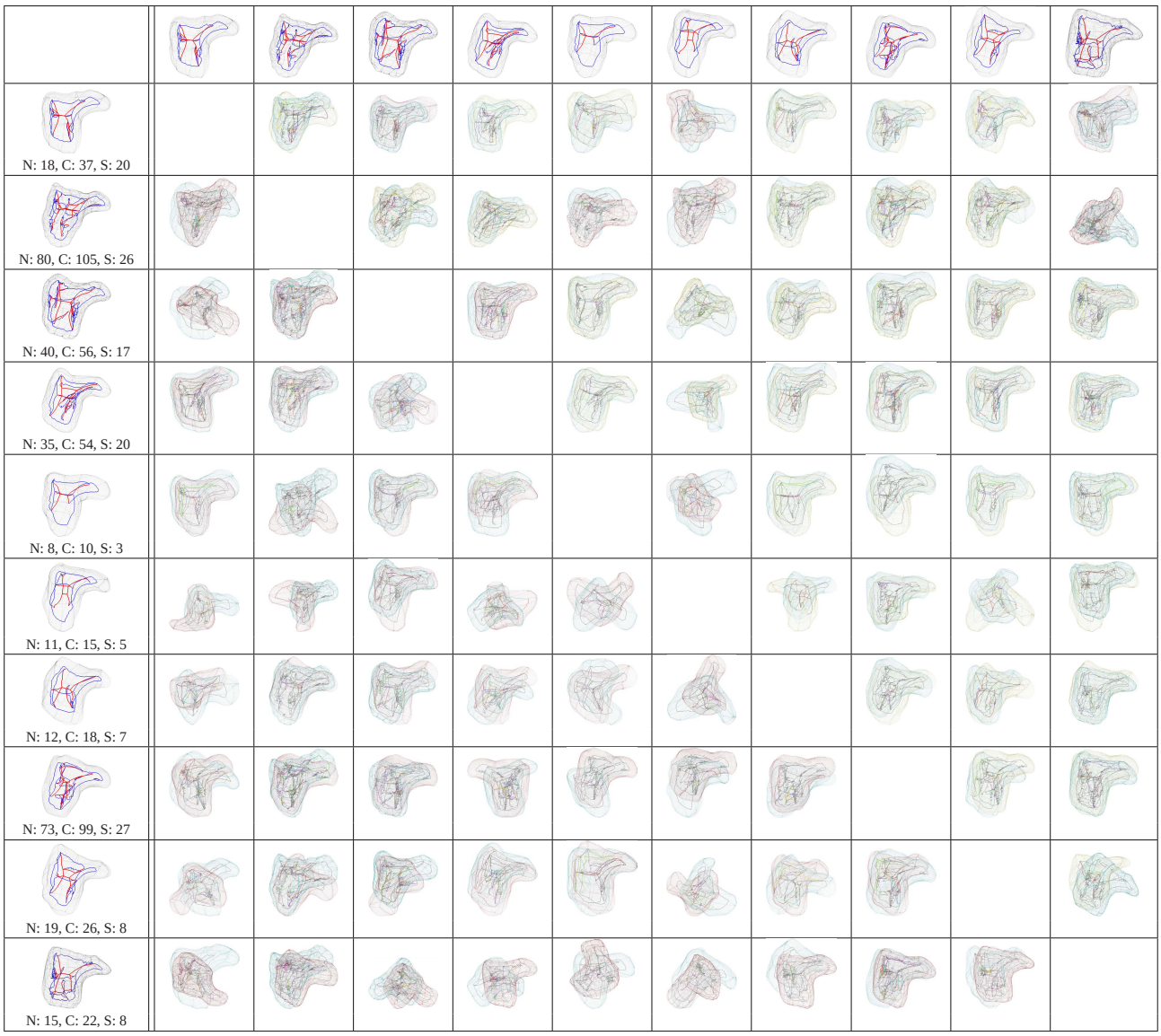
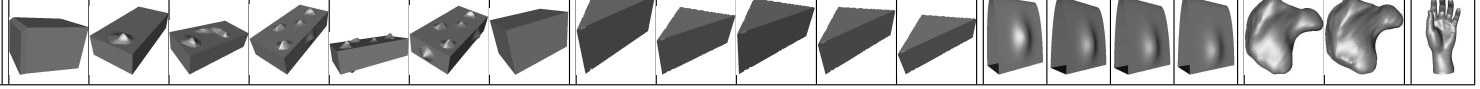


Figure 9.10: Result in matching the $\mathcal{MS}$ hypergraphs of the (hamate) carpal bones across 10 patients (data is courtesy of Dr. Crisco, RI Hospital [140]). The number of medial nodes (N), curves (C), and sheets (S) of the regularized $\mathcal{MS}$ are labelled below each bone on the left.



1.000	0.562	0.563	0.445	0.662	0.446	**0.659**	0.640	0.590	0.587	0.619	0.508	0.178	0.170	0.178	0.326	0.364	0.410	0.120
0.550	**1.000**	**0.870**	**0.710**	**0.692**	0.638	0.519	0.535	0.539	0.530	0.479	0.523	0.223	0.202	0.251	0.282	0.385	0.336	0.163
0.546	**0.790**	**1.000**	**0.700**	**0.660**	**0.647**	0.527	0.545	0.559	0.529	0.519	0.547	0.202	0.206	0.208	0.279	0.460	0.363	0.234
0.495	**0.701**	**0.660**	**1.000**	**0.869**	**0.955**	0.650	0.612	0.553	0.598	0.555	0.600	0.238	0.082	0.256	0.279	0.444	0.350	0.217
0.512	**0.719**	**0.699**	**0.843**	**1.000**	**0.877**	**0.672**	0.484	0.475	0.478	0.589	0.531	0.223	0.225	0.230	0.264	0.348	0.422	0.188
0.501	**0.646**	**0.661**	**0.953**	**0.893**	**1.000**	**0.741**	0.591	0.604	0.608	0.565	0.594	0.243	0.138	0.250	0.275	0.421	0.365	0.197
0.607	0.512	0.546	**0.615**	**0.712**	0.650	**1.000**	0.600	0.569	0.556	0.552	0.478	0.132	0.148	0.135	0.203	0.279	0.362	0.141
0.672	0.578	0.577	0.615	0.671	0.646	0.650	**1.000**	**0.917**	**0.934**	**0.862**	**0.730**	0.148	0.108	0.140	0.265	0.247	0.244	0.147
0.670	0.583	0.577	0.624	0.689	0.657	0.664	**0.908**	**1.000**	**0.924**	**0.899**	**0.823**	0.138	0.199	0.180	0.238	0.263	0.262	0.147
0.683	0.575	0.562	0.619	0.681	0.645	0.653	**0.925**	**0.918**	**1.000**	**0.885**	**0.729**	0.146	0.178	0.153	0.238	0.256	0.251	0.141
0.555	0.556	0.538	0.594	0.673	0.610	0.664	**0.867**	**0.907**	**0.896**	**1.000**	**0.854**	0.337	0.200	0.300	0.351	0.343	0.362	0.176
0.582	0.585	0.597	0.645	0.668	0.659	0.626	**0.734**	**0.832**	**0.733**	**0.818**	**1.000**	0.311	0.261	0.290	0.324	0.404	0.383	0.260
0.140	0.157	0.131	0.169	0.165	0.172	0.155	0.133	0.140	0.132	0.327	0.267	**1.000**	**0.613**	**0.903**	**0.505**	0.220	0.311	0.012
0.186	0.209	0.562	0.085	0.187	0.136	0.058	0.277	0.287	0.279	0.160	0.340	**0.662**	**1.000**	**0.597**	**0.496**	0.179	0.282	0.038
0.168	0.159	0.173	0.188	0.192	0.181	0.159	0.146	0.142	0.138	0.334	0.232	**0.931**	**0.596**	**1.000**	**0.527**	0.208	0.338	0.013
0.341	0.278	0.246	0.266	0.277	0.274	0.267	0.278	0.303	0.319	0.370	0.299	**0.492**	0.192	**0.493**	**1.000**	0.236	0.292	0.036
0.325	0.427	0.444	0.420	0.392	0.360	0.221	0.243	0.263	0.234	0.292	0.328	0.204	0.016	0.205	0.218	**1.000**	**0.483**	0.118
0.367	0.399	0.413	0.245	0.437	0.235	0.292	0.306	0.306	0.304	0.370	0.370	0.423	0.283	0.401	0.226	**0.513**	**1.000**	0.125
0.099	0.191	0.194	0.196	0.183	0.182	0.124	0.065	0.130	0.127	0.142	0.231	0.016	0.046	0.016	0.052	0.100	0.123	**1.000**

Figure 9.11: The similarity metric for the database. Correct matches are highlighted in boldface in blue, while erroneous ones are in red. Observe that most of the shapes are categorized well.

Chapter 10

Applications of the Regularized $\mathcal{MS}$ Medial Scaffolds

* Overview of chapter: applications in shape modeling and matching.

This chapter investigates the practical applications of the medial scaffold ($\mathcal{MS}$) as a shape representation. We first list the key features of our method and discuss its potential in several fields including shape feature extraction, manipulation, and matching. We then provide a comprehensive survey of applications (related to 3D $\mathcal{MA}$) in § 10.1 and point out our potential in these perspectives. Finally, we investigate a few applications and present experimental results in ridge detection (§ 10.2), analysis of mathematical shapes (§ 10.3), and shape registration (§ 10.4).

* Key features of our approach in practical applications.

We list *three* key features of our method in representing shapes using the regularized $\mathcal{MS}$ to unlocks a wide range of shape modeling and matching applications:

- *Handle general datasets.* Our system takes the primitive form of unorganized point-sampled shapes as input and deals with generic shape topology (Chapter 6) without imposing strict restrictions on the shape topology or the sampling condition. This allows our approach to handle "all" practical forms of shape datasets.
- *Extract qualitative structures of shapes* in retaining two key features. First, the regularized $\mathcal{MS}$ addresses the main bottleneck of the $\mathcal{MA}$ instabilities, thus enables to exploit many desirable properties of the $\mathcal{MA}$ (Chapter 1) such as its hierarchical organization of shape and the completeness of such information. Second, the shape is tightly coupled with the $\mathcal{MS}$ during the regularization, such that corresponding surface regions are maintained consistently in capturing the whole volume of the shape (Chapter 7), such that salient shape features such as high curvature regions and *parts* of the shape are made explicit and organized together with the $\mathcal{MS}$.
- *Match the shape structure in a general framework.* The shape structure is organized into a graph-like form to facilitate their comparison using a graph matching framework (Chapter 9).

10.1 A Survey of Shape Modeling and Matching Applications

In this section we survey the $\mathcal{MA}$-based shape modeling applications which we expect to investigate using the proposed $\mathcal{MS}$ framework. Recently, Leymarie and Kimia have presented a detailed survey

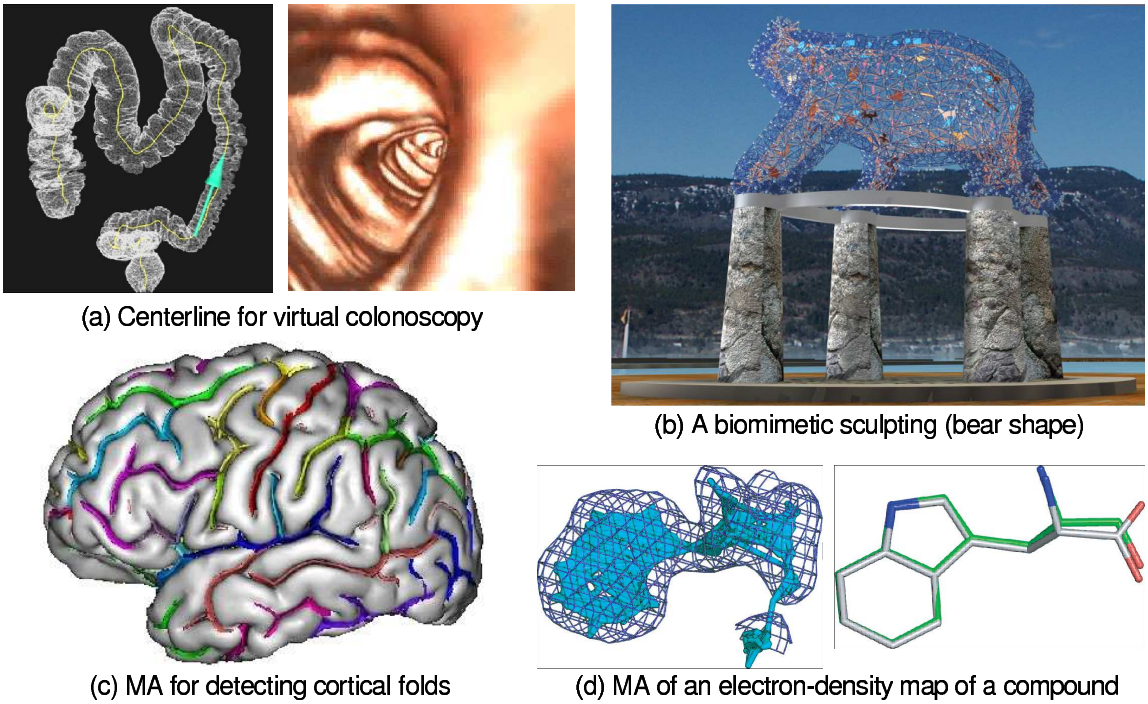


Figure 10.1: (Adapted from [127].) $\mathcal{MA}$ related applications. (a) (Adapted from [150]) The 3D $\mathcal{MA}$ is useful in computing a *curve skeleton* (centerline) for virtual colonoscopy. Left: colon and virtual camera pose along the computed path. Right: one shot of the virtual colonoscopy sequence produced for the colon. (b) Production of a biomimetic sculpting using the 3D $\mathcal{MA}$ (from Brower Hatcher and Karl Aspelund of Mid-Ocean Studio, Providence, RI, USA). (c) (Adapted from [135, Fig. 4]) Cortical folds are represented by computing a 3D $\mathcal{MA}$ using an erosion process acting on the deep crevasses. (d) (Adapted from [1, Fig. 4]) The $\mathcal{MA}$ of the electron-density map of a chemical compound (left) resembles its atomic structure and after some thinning and regularization (right), it can be used for identification via graph matching.

on many of these applications [127], where a few of them is highlighted in Figure 10.1. We organize the vast amount of applications into three categories: *(i)* shape feature detection and analysis (for a static shape), *(ii)* shape manipulation (modification of shapes), and *(iii)* matching (of two or more shapes).

(I) In **shape feature detection and analysis**, the $\mathcal{MS}$ and its tightly coupled shape boundary make explicitly several salient shape features as well as the structure of the shape, which is useful in:

1. *Ridge and corner detection.* The boundaries of the $\mathcal{MS}$ (at the A_3 rib curves) correspond to the *ridge* curves of the shape, which is an important surface feature where the surface bends sharply along them — a topic to be further explored in § 10.2. The *corner* is also a salient feature where several ridges meet and terminates. We will show initial results in studying the local form of the $\mathcal{MA}$ around a corner in § 10.3.1.
2. *Neck and limb detection. Necks* and *limbs* are important features useful in defining a hierarchical *parts* of a shape [181]. A neck is a narrowing part with local minimum in the distance between two boundaries of the shape, while a limb identifies a part of the shape from a pair of negative curvature minimum with co-circular tangents. In [181] the shock graph ($\mathcal{SG}$) is

used to detect these features in 2D, while we expect the same approach is extensible in 3D to access the radius fields of the medial sheets of the $\mathcal{MS}$ (see Figure 3.12 for an example). We further expect the shock scaffold ($\mathcal{SC}$) with a complete shock flow analysis (§ 3.4) to enhance this approach better.

3. *Detecting tubular and flat regions of shape.* In [96], Goswami and Dey *et al.* use the *"unstable manifold"* of the *flow complex* ($\mathcal{FC}$, surveyed in § 2.3) to identify the flat and tubular regions of a 3D shape. As discussed in § 2.3 that the $\mathcal{FC}$ and our $\mathcal{MS}$/$\mathcal{SC}$ share similar ideas in exploiting the distance field of the shape and obtaining the skeleton by filtering the Voronoi structure, we expect our approach to perform similar task in identifying such shape regions.
4. *Segmentation of shape* (partition or surface regions). In our approach, the boundary surface patches of the medial sheets are explicitly partitioned by the topological structure of the $\mathcal{MS}$ (see Figures 7.9, 7.12 and 7.14 for examples). This initial segmentation of shape identifies salient *parts* and can be further refined (by splitting and merging regions).
5. *Skeletonization* of the $\mathcal{MS}$ toward a *curve skeleton* ($\mathcal{CS}$). In our approach, the A_1^3 axial curves correspond to the cylindrical part of the shape (recall in Figure 8.5). As reviewed in § 2.3.2 that Dey and Sun [66] have related the $\mathcal{MA}$ and the $\mathcal{CS}$ by using a geodesic function on the medial sheets to shrink the sheets into an (1D) skeleton. We expect our approach to produce similar result, by using the geodesic distance computation developed for the contract and merge transforms in Chapter 5. We look forward to further exploit the shock flow field of the $\mathcal{SC}$ (instead of the geodesic field) to extract a better localization of the (1D) skeleton, since the shocks are directed related to the shape while the geodesic of the medial sheets is not.
6. *Analysis of mathematical shapes.* In addition to the problems in imaging and vision, the $\mathcal{MA}$ is also useful in exploring the properties of mathematical shapes, which often help to understand the fundamentals of the shape (such as the aforementioned ridges, corners, generalized cylinders, *etc.*). We will present several examples in § 10.3.
7. *Symmetry detection*. Recall that the $\mathcal{MA}$ is a subset of the symmetry set ($\mathcal{SS}$) in § 3.1. The regularized $\mathcal{MS}$ can be used to detect the symmetry of the shape [139], in finding the symmetric axis for planar-reflective symmetry [160] or their structural regularity [154].

(II) In **shape manipulation** (modification of shapes):

1. *Surface meshing from point clouds.* We have shown in Chapter 6 that the $\mathcal{MS}$ is useful in reconstructing surface meshes of various topologies from unorganized points with loose assumptions, including meshing large datasets (§ 6.4) and fusing raw scans (§ 6.3). In addition, the remaining $\mathcal{MS}$ after regularization is also useful in repairing remaining topological artifacts of the surface mesh [226]. [1]
2. *Smoothing* and *simplification* of shapes. In 2D, the shock graph ($\mathcal{SG}$) have been used to smooth a shape while preserving significant corners, by applying the splice transforms considering the local scale of the shape [200], as shown in Figure 10.2. In 3D, Tam and Heidrich have applied a related idea in pruning medial branches and update the 3D models (using the corresponding Delaunay tetrahedra) [194]. Our approach achieves a similar effect by applying the splice transforms (§ 5.2.1) and update the boundary mesh in a different way (see Figure 5.7 for an example). We expect to extend our $\mathcal{MS}$ regularization scheme to further simplify shapes, which

[1] The couple boundary-$\mathcal{MS}$ structure is useful to localize these surface artifacts, and the gap transforms can then be applied to close remaining surface holes. This also allows to enforcing a filling of 'true' surface holes larger than the scale of the sampling density (see Figure 6.17 for an example).

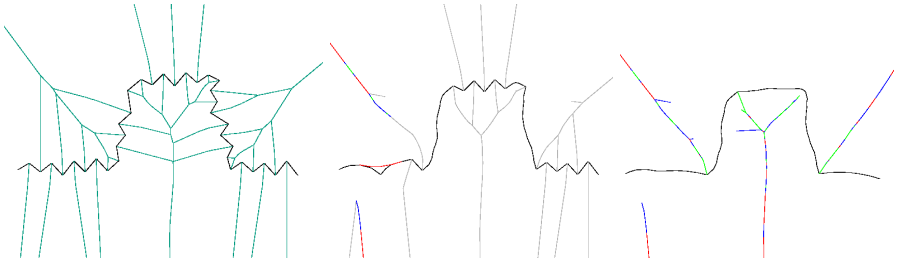


Figure 10.2: (From [200, Fig.1].) $\mathcal{MA}$ for smoothing shapes: The original square wave is perturbed at numerous places. Traditional smoothing methods smooth away both the noise and corners. Instead, the iterative removal of branches of the $\mathcal{MA}$ graph smoothes the shape while preserving its corner (only a subset of the discrete space is shown).

will provide an approach that is distinct to the traditional (surface) mesh-based smoothing and simplification approaches [199, 18, 72].

3. *Compression* of shapes. The $\mathcal{MS}/\mathcal{SC}$ hierarchies (Chapter 3) explicitly distinguish the qualitative shape structure from the local details, which suggests a way to compress shapes by reducing storage needs. It is also promising to transmit shapes in a *"progressive"* manner according to the order of the hierarchy: At first, the coarse-scale structure of the shape is obtained, which is gradually enriched with more details toward a complete reconstruction of the shape. We note that the future work of the $\mathcal{SC}$ hierarchy should provide a better qualitative abstraction of the shape, in that all shock sheets are further classified into monotonic flow districts and thus their detailed geometry can be safely discarded and are reconstructible from interpolating the *reduced shock scaffold hypergraph* ($\mathcal{SC}^{\mathcal{H}-}$) described in Figure 3.4.
4. *Animation / morphing* of shapes [191]. In [219, 222], Yoshizawa *et al.* have demonstrated a descent approach to animate 3D shapes by a skeleton-driven mesh deformation. As reviewed in § 2.3.1 that their $\mathcal{MA}$ is a two-way water-tight mesh allowing an easier parametrization and smoothing. In comparison, our explicit representation of the (dual-scale) $\mathcal{MS}$ hypergraph/mesh provides a richer structure, thus provides a better way to specify (automatically or interactively) any part of the $\mathcal{MS}$ to deform. We expect to explore more in the future.

(III) In shape **matching** (of two or more shapes), we have discussed in Chapter 9 for the matching of the $\mathcal{MS}$ in the application of shape recognition and retrieval [84] based on comparing shape similarities. We will discuss the registration of shapes in finding correspondence between shapes and compute an average shape using such correspondence in § 10.4.

* Example applications in several areas.

We further elaborate the above potentials of our approach in the following domain-specific applications:

- In the *CAD/CAM industry*, the $\mathcal{MS}$ can be used in three aspects: *(i)* the modeling mechanical parts/objects in prototyping (refer to the CAD shape modeler in [127, Fig.10]), *(ii)* the retrieval of models/parts from a database [24, 107, 157], and *(iii)* the *haptic* applications modeling the virtual "touching" of surfaces (in grabbing objects, *etc.*).
- In *digital archaeology* (heritage preservation), the $\mathcal{MS}$ can be used to identify break curves (ridges) of fragments, which is useful in re-assemble archaeological fragments [11, 137], see Figure 10.7 for an example. [2]

[2] Refer to related projects in the S.H.A.P.E. Laboratory at Brown University (http://www.lems.brown.edu/shape/).

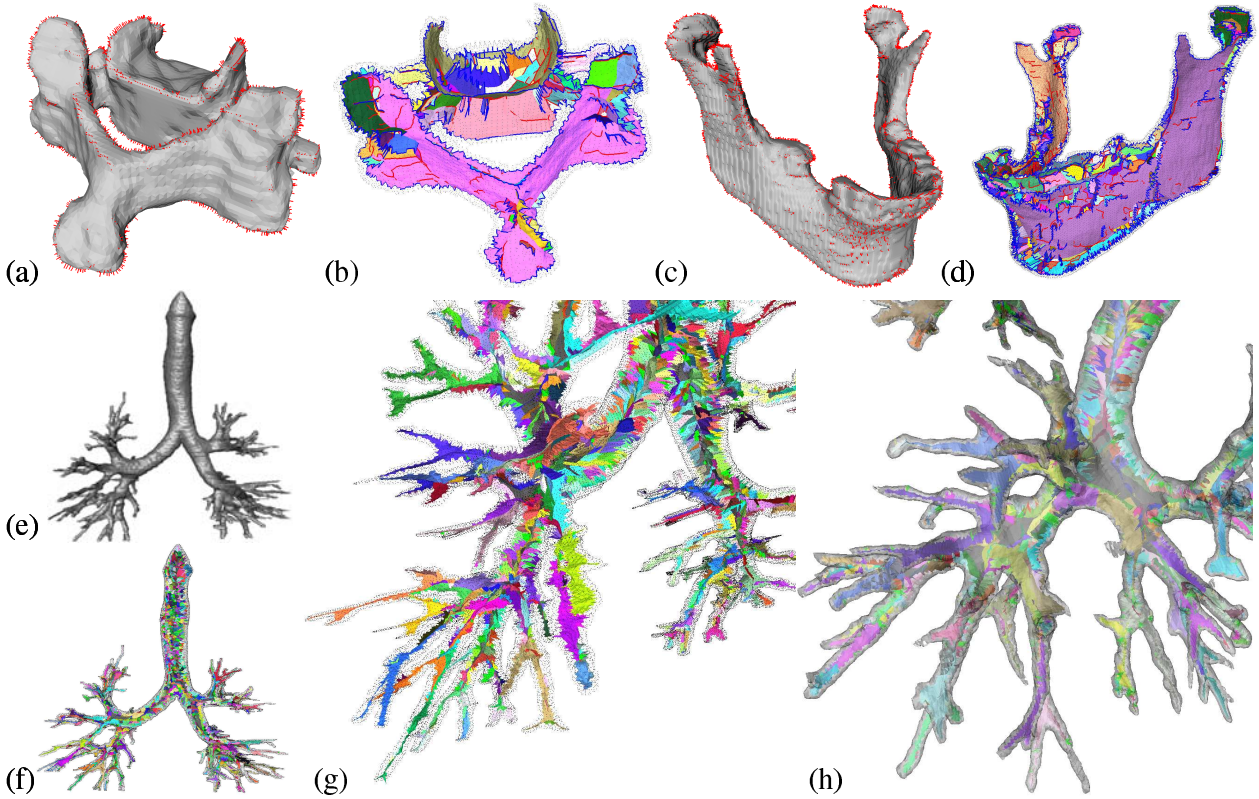


Figure 10.3: The $\mathcal{MS}$ extracted from a few medical datasets demonstrating potential applications (Data from Celina Imielinska, Columbia University). (a-b) the vertebra (red line segments indicate the detected *ridges*). (c-d) the jaw, and (e-h) the air tree. Observe how close the medial sheets approaching the object boundary to capture shape details in (g) and (h).

- In *medical imaging*, first, the aforementioned registration of shapes are useful in *e.g.*, studying the anatomical variability of the organs or bones, which will be explored more in § 10.4. Second, in virtual colonoscopy, the simplified $\mathcal{MS}$ into a centerline shall provide a centered path to "fly through" a virtual camera. Refer to Figures 1.5(a) and 10.3(e-h) for examples. We have shown the $\mathcal{MS}$ of many medical shapes in this thesis, *e.g.*, the brain (Figure 10.1(a)), the colon (Figure 10.1(a)), carpal bones (Figures 9.2, 9.9, 9.10), and the vetebral, the jaw, and the air tree (Figure 10.3) [3].
- In *anthropomorphic* [93, 94] or *biomimetic* applications, the extracting $\mathcal{MS}$ structure is also useful in *art* and sculpting, in that the $\mathcal{MS}$ features both the regularized ridges and regularized axes, which "spans" the shape interior as a scaffold (see Figure 10.1(b) for an example). We have shown the $\mathcal{MS}$ of many examples of anthropomorphic shapes *e.g.*, the human body (Figure 7.16), the face (Figure 10.8(a)), the head (Figures 10.12, 6.16, 7.13), and the hand (Figure 1.5) as well as many animal shapes such as the sheep (Figure 7.6), dog (Figure 7.13(a)), dinosaur (Figure 7.14(a-c)), and dragon (Figures 7.9, 7.11, 7.12 in this thesis.
- In *computational molecular chemistry*, the regularized $\mathcal{MS}$ is useful in modeling protein structures [133, 21] for their classification and retrieval, Figure 1.5(d).

[3]The author gratefully thinks Celina Imielinska Ph.D. and Vesalius^{TM} Project at Columbia as well as Frederic F. Leymarie for helpful discussions.

In the rest of this chapter, we describe the ridge detection (§ 10.2) and analyzing mathematical shapes (§ 10.3) in details and show further results in shape registration (§ 10.4).

10.2 Ridge Detection

* Background on $\mathcal{MA}$-based ridge detection.

Ridges are important surface features where the surface bends sharply. Mathematically, a ridge curve is the surface points where *the magnitude of the largest principal curvature attains a maximum along its corresponding lines of curvature*. The $\mathcal{MA}$ provides an explicit approach to detect ridges, in that the medial sheet boundary (A_3 ribs) directly corresponds to the ridges, Figure 10.4, which has been investigated in several works, *e.g.*, [100, 101, 144]. The $\mathcal{MA}$-based approach can be refined to combine with other ridge detection methods, *e.g.*, recent works in [221, 220, 218].

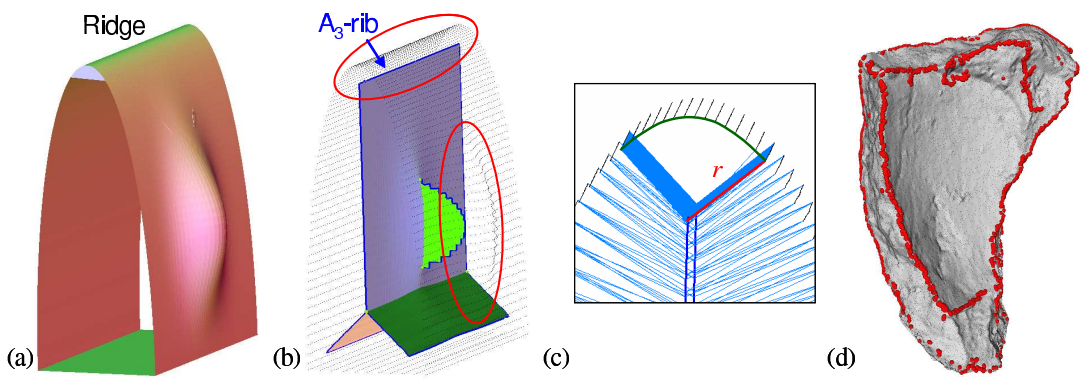


Figure 10.4: (a-b) The $\mathcal{MS}$ sheet boundary at the A_3 rib curves map to the ridge curves on the object surface. (c) The *radius of focus* of the parabolic gutter should correspond to the $\mathcal{MA}$ pruning radius r in the "min-radius" trimming process in §7.2.2. (d) Mapping the A_3 rib curves to the surface yields the *ridge* points (in red).

* Proposed approach: mapping A_3 rib curves to the ridge curves on the surface.

We detect ridges by explicitly *project* the A_3 ribs of the (regularized) $\mathcal{MS}$ to the corresponding surface to identify the ridge curves. Specifically, observe that the extension of an A_3 rib curve along the direction from the medial sheet perpendicular to the rib indicates the position of a ridge, Figure 10.4(b). We explicitly implement this idea by computing a vector pointing from each A_3 rib curve element and it's incident A_1^2 sheet element to their corresponding boundary point, which is essentially the ridge point. The details are as follows.

First, each A_1^2 sheet element S along the A_3 rib is associated with two boundary sample points, *i.e.*, its generators $(\mathsf{G_a}, \mathsf{G_b})$, Figure 10.5, which spans a fan-like region in Figure 10.4(c). The collection of this fan-like regions corresponds to the high curvature regions on the surface, where the ridges map to the curvature extrema along this region. We name it the **ridge region**, which is defined by each A_3 rib curve of the $\mathcal{MS}$. We can traverse each A_3 rib curve and use the traversing direction of the A_3 rib and the normal of the A_1^2 sheet element to orient the two boundary curves of the ridge region, Figure 10.5(a). For each sheet element S, its centroid C and the two boundary points $\mathsf{G_a}$ and $\mathsf{G_b}$ define a plane, the **sectional plane** π (yellow in Figure 10.5(a)), which is perpendicular to the ridge as well as perpendicular to the sheet § (red arrow in Figure 10.5)(a)). The **ridge vector** V is then defined to be in the direction of the intersection of the section plane π with S, which points to a

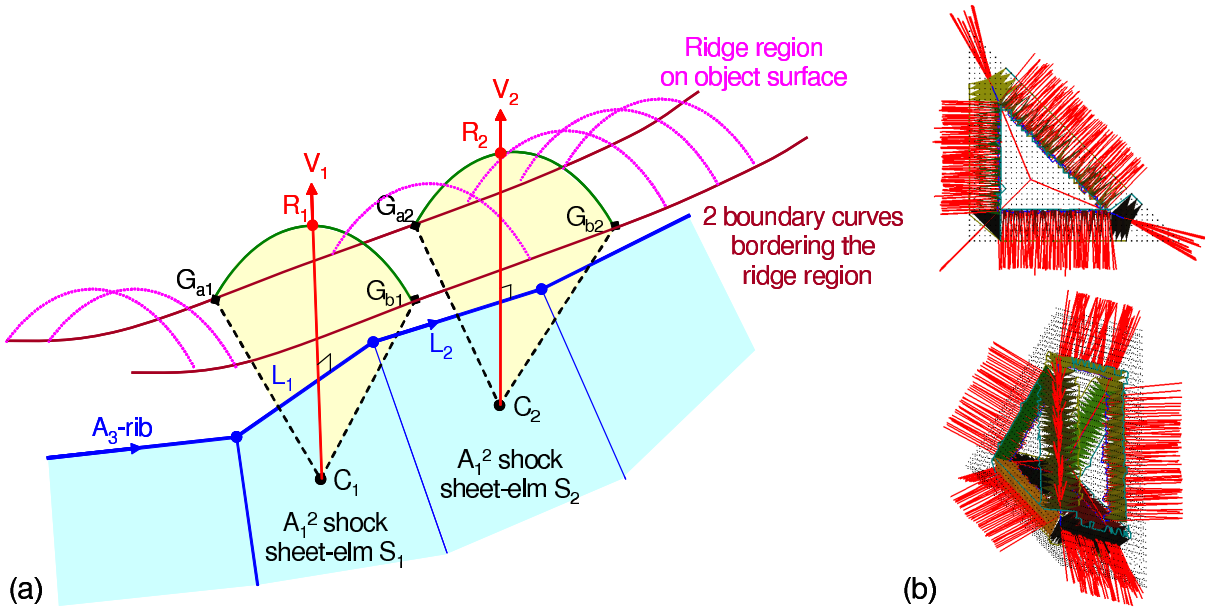


Figure 10.5: (a) The local configuration on using the "ridge vector" V_i defined on each A_1^2 sheet element S_i to detect a ridge point inside a (fan-like) surface region delineated by two boundary curves. (b) Two views of the ridge vectors (red lines) computed from the $\mathcal{MS}$ of a prism shape.

ridge point on the surface. Figure 10.5(b) shows two views of the ridge vectors (red lines) computed from the $\mathcal{MS}$ of a prims shape. The section planes π are also shown in yellowish triangles. Observe that the ridge vector is determined locally and can be noisy in pointing to the ridge points. We leave the accurate localization of the ridge points as a future work.

Our ridge detection approach then looks for all A_3 ribs and compute the ridge vectors and intersect them with the surface to obtain the ridge points. In computing the intersection point of each ridge vector on the surface, the searching space of candidate surface triangles can be narrowed down using the fact that each shock is associated with some particular surface region. Specifically, for each A_3 rib element, we only need to intersect V with L's associated boundary faces.

* Experimental results: detecting "coarse-scale" ridges and applications.

Figure 10.6 shows an example of our ridge detection on a broken pot fragments, where the ridges corresponds to the *break curves* with high curvatures. The detected ridge regions are depicted between blue points in Figure 10.7(b) and between the cyan and yellow curves in Figure 10.7(c). The detected ridge vectors are shown in red lines in Figure 10.7(c).

Figure 10.7 shows a potential application to re-assemble archaeological pot sherds by detecting the break curves and match them to find possible pairs of fragments to re-assemble them together. The structural information of the $\mathcal{MS}$ graph/hypergraph (*i.e.*, A_1^3 axial curves) is also useful in matching the break curves in assembling the pot sherds [11, 137, 212].

We have also applied the propose method to detect ridges on the human face model in Figure 10.8(a-b). Many salient face features such as the nose, chin, eyebrows, cheeks, and lips are successfully detected.

In comparing our results to other methods, we found that our approach responses to more *"coarse-scale"* ridges in Figure 10.8(c-d) that while to the refined ridge detection results from Yoshizawa *et al.* [218] in (d) detects two ridge curves (blue) on the nose, our method detects a single rib curve on the nose, indicating that our method can be used to highlight coarse-scale ridges in compensating the traditional local ridge detection methods.

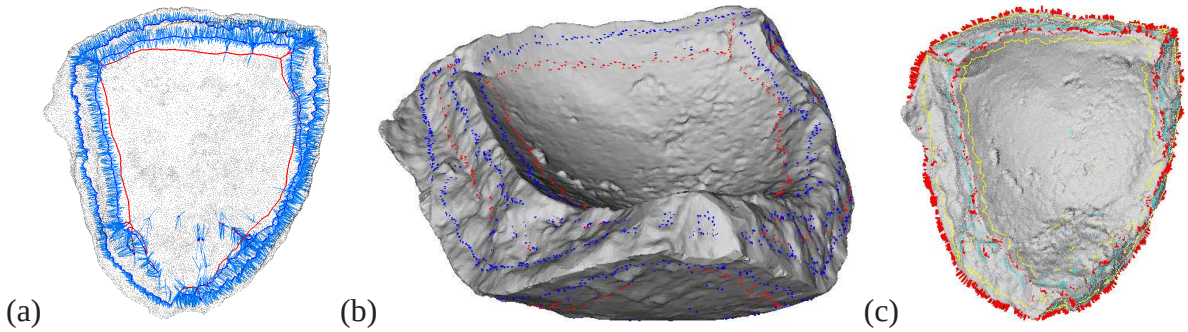


Figure 10.6: (a) The A_3 ribs and corresponding boundary (connecting via blue lines) of a pot sherd. (b) The two contact points of the A_1^2 sheet element along an A_3 rib curve are depicted in blue, while the three contact points along an A_1^3 axial curve is in red. (c) The detected *ridge region* (between the cyan and yellow curves) and the *ridge vectors* (spike-like, in red) which intersect the surface at the estimated ridge points.

10.3 Analysis of Mathematical Shapes

This section describes some results in analyzing a few $\mathcal{MA}$ of mathematical shapes. Specifically, the $\mathcal{MS}$ of three synthetic shapes are investigated to study the true $\mathcal{MA}$ of them: [4]

1. $\mathcal{MA}$ around a smooth corner with slight perturbation (Co-work with Peter J. Giblin).
2. The exterior $\mathcal{MA}$ of the "torus knots" (Co-work with Peter J. Giblin).
3. $\mathcal{MA}$ of the *Gomboc*, the world's first self-righting object (Data from Peter Varkonyi [205]).

10.3.1 $\mathcal{MA}$ around a smooth corner

The corner is an important shape feature in computer vision and in 3D scene reconstruction [206, 211]. The 3D $\mathcal{MA}$ around a smooth *corner* is an interesting problem not fully understood. Recall for the *non-generic* $\mathcal{MA}$ transitions around an corner in § 5.4 as well as the degenerate corner-merge transform of the $\mathcal{MS}$ in § 5.4.1. In order to understand the generic case of 3D $\mathcal{MA}$ around a corner, we perturb a 'perfect' (symmetric) smooth corner shape and see how the $\mathcal{MS}$ deforms, which gives a reasonable approximation to understand the behavior of the true $\mathcal{MA}$ around the corner. First, we generate a corner shape by stacking up 2D triangular blobs (similar to the Reuleaux triangles) with decreasing size as follows. Specifically, we exploit the *curve by support function* [49, 163, 19] to generate a closed convex curve in Figure 10.9(a) as follows, which is used to generate the smooth corner:

$$\begin{cases} h = \dfrac{a}{2}\cos(3t) + \dfrac{a}{2} + b + c\sin(3t) + c\sin(4t), \\ x = h\cos(t) - \dfrac{\partial h}{\partial t}\sin(t), \\ y = h\sin(t) - \dfrac{\partial h}{\partial t}\cos(t), \end{cases} \tag{10.1}$$

where the $c\sin(3t) + c\sin(4t)$ are additional terms to 'break the symmetry', *i.e.*, changing the sharpness around the three vertices of the triangular blob, to produce three *ridge* curves of different sharpness approaching the corner. The smooth corner shape in Figure 10.9(b) is produced by scaling the

[4] The author gratefully acknowledges helpful conversations with Prof. Peter J. Giblin, Dept. of Math. Science, Univ. of Liverpool, UK.

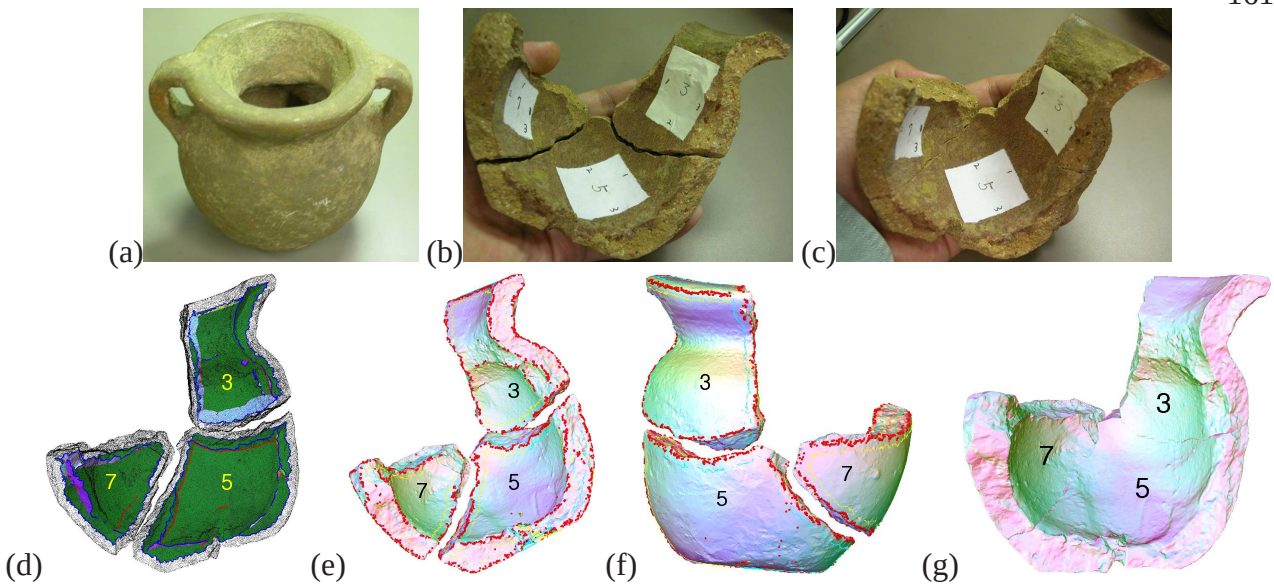


Figure 10.7: Application of ridge detection in identifying the "break curves" of sherds in digital archaeology. A pot in (a) is broken into pieces where three of them is shown in (b). Observe in (c) that the break curves are strong clues for re-assembly — they match tightly and seamlessly for the correct pairs. (d) shows the $\mathcal{MS}$ for the three sherds in (b). (e-f) shows tow views of the detected ridges, which are used to re-assemble the sherds perfectly in (g).

triangular blob by a factor $k = [0, 1]$ as increasing a height $z = d \cdot k^2$ with parameters $a = 2$, $b = 10.5$, $c = 0.3$, $d = 15$, and $t = [0, 2\pi)$. [5]

We observe in Figure 10.9(e) that the perturbed corner has the $\mathcal{MA}$ decomposed into an A_3 rib and an A_1^2 sheet (tab) with an A_1A_3 fin point, which is exactly the configuration in reversing the corner-merge transform we described in § 5.4.1. We observe the $\mathcal{MA}$ at the tip of the corner shrinks a bit, due to the local smoothness and the vanishing of the sharp ridges. We leave the formal investigation of the generic local form of the $\mathcal{MA}$ around a corner as a future work.

10.3.2 Exterior $\mathcal{MA}$ of a "torus knot"

The $\mathcal{MA}$ is useful in the mathematical study of the topology of shapes. In [58], the topology is used to limit possible properties of the $\mathcal{MA}$. Here we show an interesting example of the *exterior* $\mathcal{MA}$ of a "torus knot". [6]

We generate a *torus knot* shape in Figure 10.10 by wrapping a tube (with radius ρ) around a torus parameterized by:

$$\begin{cases} x = (R + r\cos(\phi))\cos(\theta), \\ y = (R + r\cos(\phi))\sin(\theta), \\ z = r\sin(\phi), \end{cases} \tag{10.2}$$

where $\theta = [0, 2\pi)$ and $\phi = [0, 2\pi)$ give a torus, and space curve of the tube is given by parameterizing $t = [0, 2\pi)$:

$$\begin{cases} \theta = m \cdot t, \\ \phi = n \cdot t. \end{cases} \tag{10.3}$$

[5] Letting $z = d \cdot k$ generates a 'sharp' corner (similar to the tip of a cone), whose $\mathcal{MA}$ shall start right at the tip. Refer to Figure 5.20(a) for a similar case.

[6] Acknowledgements also forward to helpful conversations with Prof. James Damon, Dept. of Math., Univ. of North Carolina, USA.

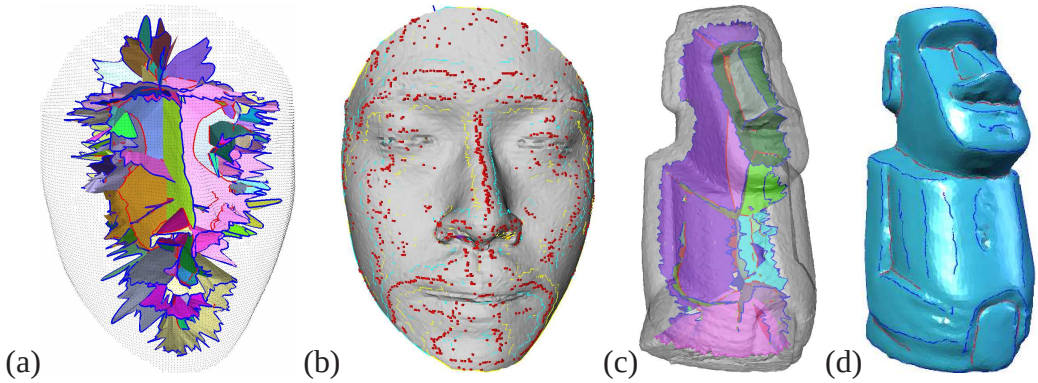


Figure 10.8: This figure illustrates that our method detects *"coarse-scale"* ridges. (a) The regularized $\mathcal{MS}$ of $11,748$ scan points of a human face (Data from MPII). (b) Our ridge detection result, where the ridge points are in red and the two ridge region curves are in cyan and yellow, respectively. Observe that many high curvature regions are detected as ridges such as the nose, while fine-scale sharp features such as the eyes are suppressed. (c) shows the regularized $\mathcal{MS}$ of the Moai dataset from MPII. (d) is the ridge detection result from Yoshizawa *et al.* [218]. Note that we identify a single ridge curve at the nose in (c), while in (d) there are two (fine-scale) ridge curves detected in [218].

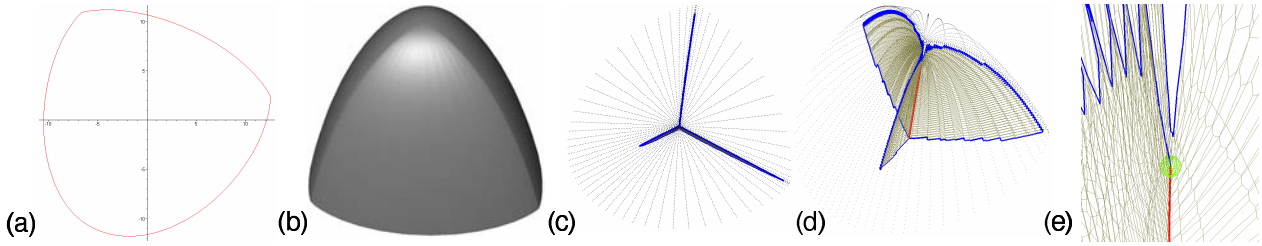


Figure 10.9: $\mathcal{MA}$ of a perturbed smooth corner. (Co-work with Peter Giblin). (a) A 2D triangular blob generated using Equation 10.1 to produce a corner shape in (b) with three *ridges* of different sharpness. (c,d) Two views of the $\mathcal{MS}$ which contains three medial sheet branches intersecting at the center (A_1^3 curve pointing to the corner), where the smaller sheet (on the left) corresponds to the more rounded ridge. (e) Zoom in to the tip of the corner in (d) to show the $\mathcal{MA}$.

where m and n must be co-prime integers to produce a closed knot. We use $R = 3$, $r = 1$, $\rho = 0.4$ for the example in Figure 10.10(a). One interesting fact about this family of torus knots is that their exterior $\mathcal{MA}$ of the tube intersects at a single *straight* line at the center of the torus [58]. Our $\mathcal{MS}$ validates this result as shown in Figure 10.10.

10.3.3 $\mathcal{MA}$ of the "Gomboc"

A *"Gomboc"* [205] (Figure 10.11) is the first found self-ridge object as a result of a long mathematical quest. To answer Vladimir Arnold's question on finding a *"mono-monostatic"* object [205], the Gomboc is discovered by Gabor Domokos and Peter Varkonyi: [7] it is a mathematical shape with one stable and one unstable point of equilibrium, enabling it to mimic the "self-righting" abilities of shelled animals such as turtles and beetles. [8]

[7] The author gratefully thank Peter Varkonyi to provide the Gomboc model and helpful discussions.
[8] Refer to http://en.wikipedia.org/wiki/Gomboc for more info.

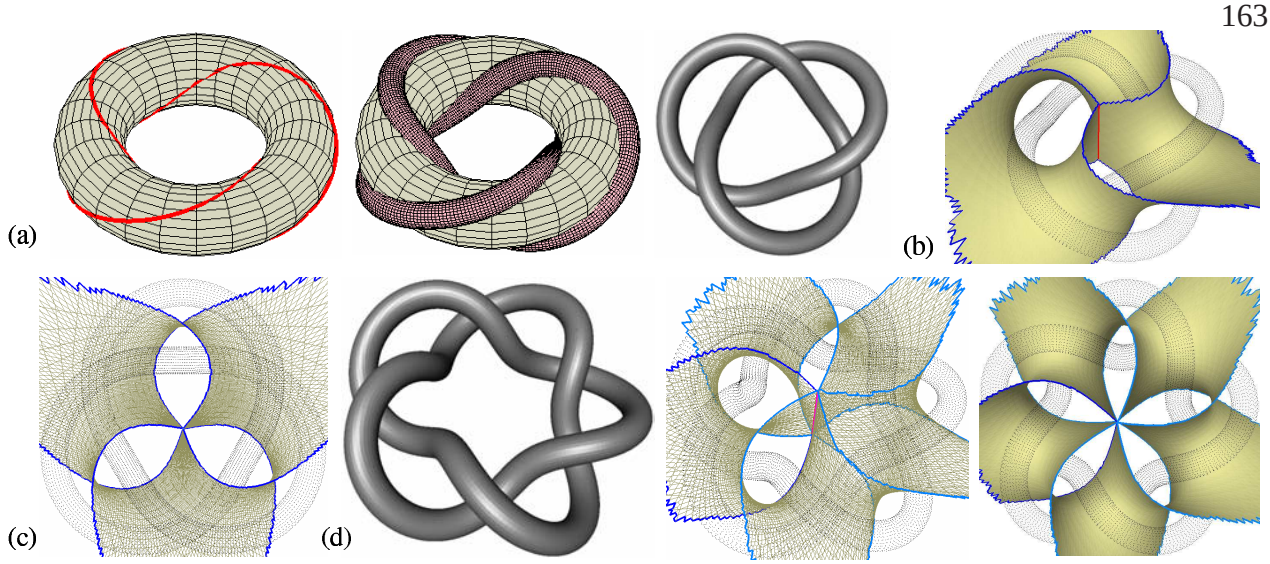


Figure 10.10: $\mathcal{MA}$ of the *"torus knot"* (Co-work with Peter J. Giblin). (a) The trefoil knot generated in Maple by constantly rotating a tube around a torus $(m = 2, n = 3)$ in Equation 10.3. (b) The exterior $\mathcal{MA}$ outside the tube wraps around it and intersects at the center at a *straight* line (red), which is confirmed in a top view in (c). (d) Another knot with 5 loops generated with $(m = 2, n = 5)$ has 5 $\mathcal{MA}$ sheets intersects at the center as a straight line.

We compute the $\mathcal{MA}$ of the Gomboc as shown in Figure 10.11. The A_1^3 axials and A_3 ribs of the $\mathcal{MS}$ provide a way to analyze the interior structure of the Gomboc shape (future work).

10.4 Global Registration by Matching the $\mathcal{MS}$

* Problem setup: global and local registration of 3D shapes.

Registration (*i.e.*, finding correspondence in matching) plays an important role in 3D data processing, which can be classified into two major types: *global* (*crude*) registration and *local* (*fine*) registration [217]. There are good methods available for *local* registration which often require that the initial pose is close to the optimal solution, such as the popular iterative closest point (ICP) [22]. On the other hand, *global* registration is considered to be more difficult. Although it can be done manually, this becomes a tedious job when the number of candidates to be matched is large, or the features are not perceptually obvious. In this section, we consider the global registration of surface datasets represented by unorganized point clouds.

* Register scans of objects with different scanning parameters.

We narrow the problem to derive a global alignment for registering scanned datasets obtained from the same object at different times, using different operators, settings, or equipments, using the $\mathcal{MS}$ matching framework described in Chapter 9. We first verify the accuracy of the results by aligning two distinct random sub-samples of a high resolution shape which is itself used as ground truth, Figure 10.12. We then test the real scan data as Figure 10.13(a-d) illustrates. We further "cut off" the a large portion of the model and show that our method is robust enough to tolerate missing chunk of data, Figure 10.13(e-f). Furthermore, the global alignment does not require closed surfaces. Figure 10.14 shows two scans of a pot's outer surface and the successful alignment of the two surfaces.

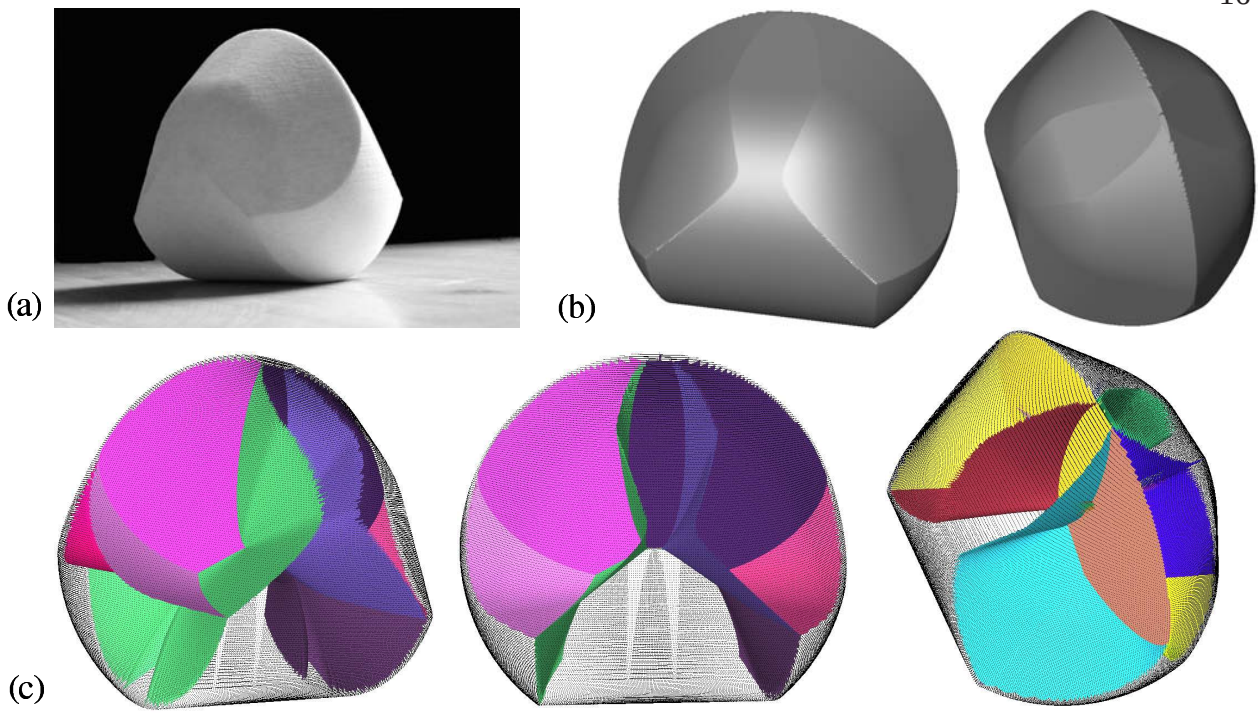


Figure 10.11: (a) The *"Gomboc"* (data from Varkonyi [205]) is a 3D shape that will stand upright which ever side it falls upon. (b) Two views of the re-meshed surface. (c) Three views of the $\mathcal{MA}$ of (b).

10.4.1 On computing an average shape (atlas)

* Future work: compute averaging shapes from the correspondence of the matching.

The above *non-rigid* registration framework can be extended naturally to compute an average shape out of a category of similar shapes toward a computational *atlas*. The matching correspondence between the regularized $\mathcal{MS}$ hypergraphs can be used to determine the point-to-point correspondence between shapes (from *e.g.*, surface patches of corresponding shock branches) and then to compute an average shape from such correspondence. This particularly useful in medial applications such as computing an atlas of the carpal bones shown in Chapter 9 to study their shape variations.

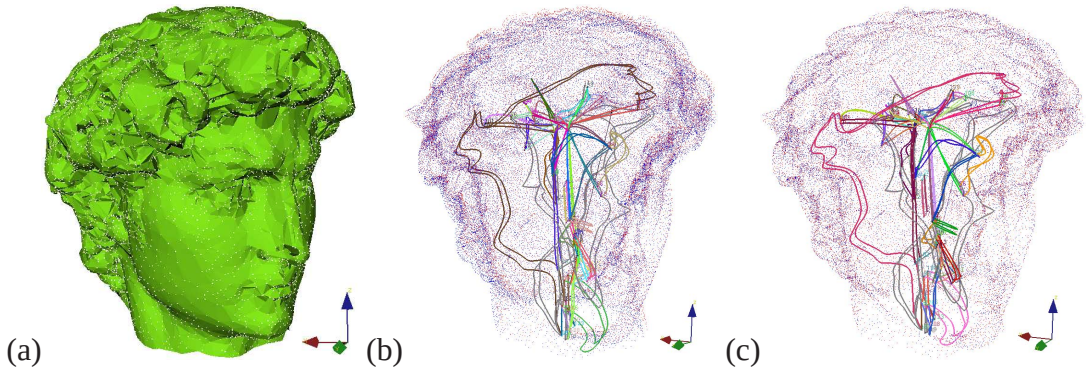


Figure 10.12: The ground truth validation on matching the $\mathcal{MS}$ of two *simulated* scans of David's head ($42,350$ points, from [124] shown in (a)), where 20K and 30K points are randomly selected from the ground truth file to test the global registration. (b) The matching $\mathcal{MS}$ curves are shown in identical colors; one dataset is shown with red dots, and the other with blue dots. Validation against the ground truth shows that the average square distance is 3.129372 (where the object bounding box is $69 \times 69 \times 76$). (c) shows the final registration result after 20 iterations of ICP, where the refined square distance *w.r.t.* to ground truth is only 0.000005.

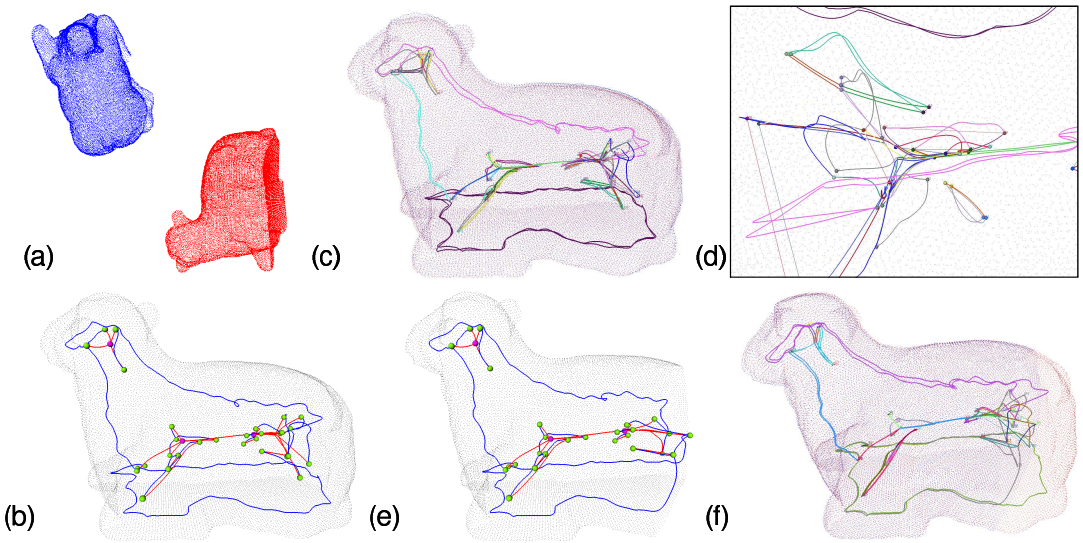


Figure 10.13: This figure illustrates the registration of two scans of a toy sheet model. (a) shows the two scans in different resolutions, parameters, and orientations. (b) is the $\mathcal{MS}$ of one of the scan (20K points). (c) depicts the $\mathcal{MS}$ matching results, and (d) zooms in to show the matching components colored in pairs. Although there are a few mismatches, globally the result is very close to the optimal solution. (e-f) illustrates the registration under chunks of missing data. In (e), the rear portion of one scan is cut, where the $\mathcal{MS}$ hypergraph topology is drastically changed. (f) shows that the $\mathcal{MS}$ still matches well in this case.

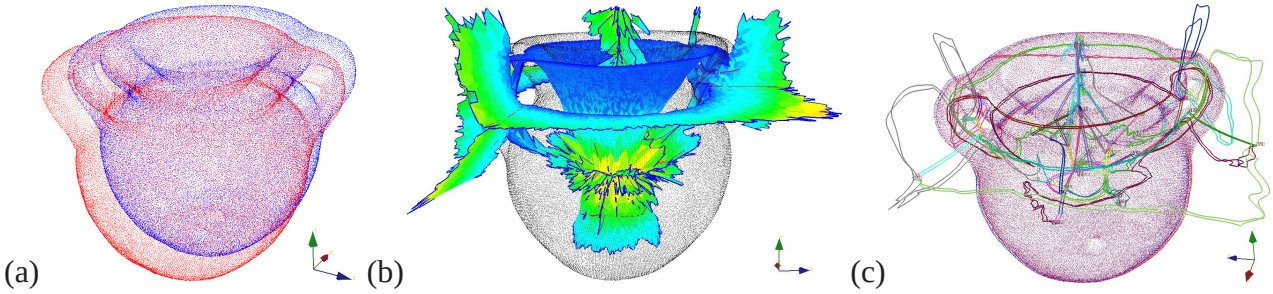


Figure 10.14: Our approach is capable to match both inside and outside medial structures. (a) An archaeological pot under two scans, which both contain un-scanned portions ("holes", *e.g.*, the interior of the handle is not reachable under scanning). (b) The $\mathcal{MS}$ of one of the scan, where the interior and exterior component connect together through the holes on the scanned surface. (c) The proposed method works well on matching the whole scaffold. This result is further fined by a few iterations of ICP to make converge perfectly (not shown).

Chapter 11

Conclusions

* A summary of main achievements of this thesis.

We have developed a general framework to represent 3D shapes by a hierarchical graph-like medial axis ($\mathcal{MA}$), the *Medial Scaffold* ($\mathcal{MS}$) [125, 87], a hypergraph containing 2D medial sheets, 1D medial curves, and medial vertices. We handle the $\mathcal{MA}$ instabilities as scaffold *transitions* [88], (*i.e.*, topological changes in the graph structure of this scaffold), which are regularized by a set of *transforms* [43], (*i.e.*, graph deformations towards higher symmetries and simplification) [45, 126]. We have also adopted a graph matching approach, the graduated assignment ($\mathcal{GA}$) algorithm to match the $\mathcal{MS}$ hypergraphs. The regularized $\mathcal{MS}$ as a shape representation tool has demonstrated its successfulness in shape recognition and other modeling applications.

* What is special about the medial scaffold?

We point out three significance of the proposed shape representation framework using the $\mathcal{MS}$:

1. *A qualitative $\mathcal{MA}$ structure with a consistent coupled shape.* The major difference of our approach when comparing to other existing 3D shape skeleton extraction methods is that we focus on regularizing medial structures while retaining its approximation to the true $\mathcal{MA}$. In comparison, other methods focus on *pruning* noisy branches and do not simplify their interconnectivity between medial sheets (§ 2.3), and on the other hand, the *curve skeleton* ($\mathcal{CS}$) based methods (§ 2.3.2) over-simplify the medial sheets. Our approach not only regularizes the medial structure but also maintain a consistent shape boundary which is tightly coupled with the $\mathcal{MS}$ (§ 7.3).
2. *An embedded theory of $\mathcal{MA}$ transitions allowing to effectively regularize the $\mathcal{MS}$ and match them with a shape similarity metric.* The structure of the $\mathcal{MS}$ (hypergraph topology, Chapter 4) can be effectively regularized by applying a system of $\mathcal{MS}$ *transforms* (Chapter 5) which essentially moves the $\mathcal{MS}$ toward nearby $\mathcal{MA}$ *transition* points. This approach is in fact embedded under a larger abstract framework to partition the *shape space* by *(i)* grouping similar shapes into a *shape cell* and discretize the *optimal* deformation path to match shapes (Chapter 8).
3. *Computationally practical.* Our approach takes the primitive form of unorganized sample points and re-mesh the surfaces (while additional information such as partial meshes or surface normals are also useful). An initial $\mathcal{MS}$ is obtained during this meshing process and continue to be regularized in our automatic computational pipeline. The regularized $\mathcal{MS}$ hypergraphs are then matched for their similarity using a graph matching scheme (Chapter 9). Our implementation of the $\mathcal{MS}$ are promising in various practical applications (Chapter 10).

* A overview of this thesis *w.r.t.* related line of works of Kimia *et al.*

This thesis continues the approach of Kimia and Leymarie *et al.* by applying the *shock transforms* based on a line of 2D and 3D works: *(i)* the hierarchical organization of the 3D $\mathcal{MA}$ sheets, curves, and isolated points into a *hypergraph* form [87] as well as in a reduced *graph* form [125] toward the notion of the *medial scaffold* ($\mathcal{MS}$), which serves as our representation of the 3D $\mathcal{MA}$; *(ii)* a theoretical study of the *transitions* (sudden topological changes) of the $\mathcal{MA}$ under shape deformations in 2D [89, 200] and in 3D [88]; *(iii)* a practical computational scheme to compute the (full) $\mathcal{MS}$ from unorganized points [125] and produce an initial surface mesh of the shape [45], and further regularize the remaining $\mathcal{MS}$ in an automatic computational system [43, 126]; *(iv)* a shock graph matching framework to match 2D shapes by estimating their optimal deformation guided by the transition of the shock graph ($\mathcal{SG}$) [169], and an approximated graph matching approach using the graduated assignment ($\mathcal{GA}$) to match the $\mathcal{SG}$ [175].

Table 11.1 overviews this thesis with respect to the related works of our group lead by Prof. Kimia *et al.* at LEMS, Brown University, USA.

Table 11.1: A overview of recent $\mathcal{MA}$ and "*shock*" related works Kimia *et al.* in 2D, the *shock graph* ($\mathcal{SG}$), and in 3D, the *medial scaffold* ($\mathcal{MS}$), in terms of theoretical investigations, practical implementations, and related applications.

	2D $\mathcal{MA}$/shocks	3D $\mathcal{MA}$/shocks
Theoretical investigations	$\mathcal{MA}$ local form and transitions [89]. Shape reconstruction from $\mathcal{MA}$ [86].	$\mathcal{MA}$ local form [87]. $\mathcal{MA}$ transitions [88]. $\mathcal{MA}$ consistency conditions [161].
Implementations in practice	$\mathcal{SG}$ computation [195, 202]. $\mathcal{SG}$ transition and regularization [169, 200].	$\mathcal{MS}$ formulation/computation [125]*. $\mathcal{MS}$ regularization [126, 43]*.
Applications	Boundary smoothing [200]. Matching/recognition [169, 175, 148]. Shape generation [203]. Appearance (visual fragments) [196, 149]. Segmentation [170].	Survey of applications [127, 90]. Surface meshing [45]*. Registration [44]*. Feature detection & modeling [43]*. Matching/recognition*.

* works accomplished/addressed in this thesis.

11.1 Conclusive Remarks on Main Topics Covered in this Thesis

* A summary of the $\mathcal{MA}$ and $\mathcal{MS}$, future work of the $\mathcal{SC}$.

The *medial axis* ($\mathcal{MA}$) is the closure of the loci of centers of maximal balls tangent to the object surface at two or more points. A classification of the local form of contact of the ball of tangency leads to ***five*** principal types of $\mathcal{MA}$ points: A_1^2, A_1^3, A_3, A_1^4 and A_1A_3 (Figure 3.2) [87]. The *medial scaffold* ($\mathcal{MS}$) [125] is a hierarchical structure based on this classification of the $\mathcal{MA}$: medial sheets are viewed as "hanging off" a scaffold made from medial curves A_1^3 and A_3 and medial points A_1A_3 and A_1^4). The $\mathcal{MS}$ is a hypergraph of isolated medial points as nodes, medial curves as links, and

medial sheets as hyperlinks. The coarse-scale structure of the $\mathcal{MS}$ is represented as a topological hypergraph, and its fine-scale metric is represented as a polygonal mesh.

Future works include a complete shock flow analysis of $\mathcal{MS}$ toward the coarse-scale shock scaffold ($\mathcal{SC}$). As described in § 3.4, this requires to study a topological *surface network* to partition the medial sheets into districts of *monotonic* flows. We believe the proposed dual-scale $\mathcal{MS}$ representation, *i.e.*, the notion of separating topology and fine-scale geometry/dynamics are directly extensible to construct a coarse-scale $\mathcal{SC}$, once a formal understanding of the shock flow on the medial sheets is accomplished.

* Remarks on the $\mathcal{MS}$ transitions and transforms.

Based on a formal study of all *generic* 3D $\mathcal{MA}$ transitions [88], we define the set of $\mathcal{MS}$ transforms covering all cases of the transitions, including the *(i)* generic transitions of simple closed shapes, *(ii)* transitions of non-closed shapes, and *(iii)* non-generic transitions observed in practice. This framework of $\mathcal{MS}$ transforms operates on the *dual-scale* $\mathcal{MS}$ representation (Chapter 4) and effectively regularizes the $\mathcal{MS}$ hypergraph in simplifying its topology and geometry while maintaining a consistent boundary shape. We have also analyzed the high-order degenerate medial nodes of the resulting $\mathcal{MS}$ (Chapter 5).

Future works include to derive more accurate transform cost estimations (§ 8.5) and develop a consistent way to update shape changes for the *interior* contract and merge transforms. We also expect to investigate the $\mathcal{MA}$ transition around the corner shape (§ 5.4.1). In addition, we expect to study the $\mathcal{SC}$ transitions and transforms, which involve additional transitions pertinent to the shock flow change, while the topology of the $\mathcal{MS}$ is keeping intact.

* Remarks on meshing unorganized sample points into surfaces.

We handle unorganized sampled shapes by developing a surface meshing method (Chapter 6) capable of dealing with generic surface topologies: whether they are closed or not, orientable or not, smooth or not, uniformly sampled or not, with non-manifold intersections or not. The input data consists of only 3D positions of (sample) points — no assumptions (on sampling density, normals) are needed to process a raw dataset, although additional knowledge, such as on the sampling density and the local connectivity as a partial mesh, can be used to refine our results. The current implementation is roughly as fast (and with pseudo-linear complexity in the number of samples) as other recent popular methods (see Figure 6.25) and the potential to handle vary large datasets is also very promising. This surface meshing process is also part of the framework of an automatic computation and regularization of the $\mathcal{MS}$ (Chapter 7).

* Remarks on the $\mathcal{MS}$ regularization.

We have developed an approach to stably regularize the $\mathcal{MS}$ by applying the set of transforms to simplify the $\mathcal{MS}$ towards close-by $\mathcal{MA}$ transitions. Our transform-based $\mathcal{MA}$ regularization is drastically different from the traditional approaches which focus on *pruning* medial sheets (see § 2.3). Our approach not only advocates a holistic component-based transformation in pruning spurious medial sheets, but also deal with many additional cases on simplifying the structural interconnectivity between medial sheets, so that the *qualitative* structure of the $\mathcal{MA}$ emerges. Our implementation composes of multiple stages of processes (Chapter 7) to best extract the details of the *"tips"* of the $\mathcal{MA}$ (A_3 ribs) in different cases such as low-sampling and non-solid surfaces. Our system is fully automatic and handles both real-life (scanned) objects, medical models, and degenerate man-made objects.

Future works include the further simplification of the $\mathcal{MS}$ toward an one-dimensional *curve skeleton* ($\mathcal{CS}$). We expect it to provide a solution to relate the $\mathcal{MA}$ to such reduced one-dimensional qualitative structure of the shape.

* Remarks on matching the $\mathcal{MS}$ for 3D shape recognition.

We have developed a 3D shape matching approach to measure 3D shape similarity by matching on their $\mathcal{MS}$ structures (Chapter 9). It is based on the *graduated assignment* algorithm to robustly matching the $\mathcal{MS}$ hypergraphs, while matching the hypergraph nodes/curves/sheets by a matching a set of compatibility functions to reflect both the graph structure and parametric variations of the components. Furthermore, this graph matching scheme can be viewed as an approximated solution embedded under a larger theoretical framework (Chapter 8), that is to view shape deformations as $\mathcal{MA}$ across *transitions* and exploit an optimal *"edit-distance"* algorithm to solve for the "minimal" deformation between two shapes (and thus build a metric in comparing them).

Future works include to exploring the above optimal hypergraph edit-distance matching to match the $\mathcal{MS}$ and the $\mathcal{SC}$, toward solving the general 3D shape recognition problem.

Bibliography

[1] Jun Aishima, Leo Guibas, Axel Brunger, and Paul Adams. Automated crystallographic building using the medial axis transform of an electron density isosurface. *Acta Crystallographica D — Biological Crystallography*, 61(Part 10):1354–1363, October 2005.

[2] Rémi Allègre, Raphaëlle Chaine, and Samir Akkouche. A flexible framework for surface reconstruction from large point sets. *Computer Graphics*, 31(2):190–204, 2007.

[3] Rémi Allègre, Raphaëlle Chaine, and Samir Akkouche. A streaming algorithm for surface reconstruction. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 79–88, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[4] H. A. Almohamad and S. O. Duffuaa. A linear programming approach for the weighted graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 15(5):522–525, May 1993.

[5] Nina Amenta and Marshall Bern. Surface reconstruction by Voronoi filtering. *Discrete and Computational Geometry*, 22:481–504, 1999.

[6] Nina Amenta, Sunghee Choi, Tamal Dey, and Naveen Leekha. A simple algorithm for homeomorphic surface reconstruction. *International Journal of Computational Geometry and its Applications*, 12(1–2):125–1141, 2002.

[7] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *Proceedings of the ACM Symposium on Solid Modeling and Applications*, pages 249–260, 2001.

[8] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *International Journal of Computational Geometry and its Applications*, 19(2-3):127–153, 2001.

[9] Nina Amenta et al. The power crust, unions of balls, and the medial axis transform. *International Journal of Computational Geometry and its Applications*, 19(2-3):127–153, 2001.

[10] Tarik Filali Ansary, Mohamed Daoudi, and Jean-Phillipe Vandeborre. A bayesian 3-d search engine using adaptive views clustering. *IEEE Transactions on Multimedia*, 9:78–88, 2008.

[11] H. Can Aras. Hindsite: A robust system for archaeological fragment re-assembly. Master's dissertation, Brown University, Providence, USA, 2007.

[12] D. Attali, J.-D. Boissonat, and H. Edelsbrunner. Stability and computation of the medial axis. *In Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, 2004.

[13] Dominique Attali. r-regular shape reconstruction from unorganized points. *Computational Geometry: Theory and Applications*, 10(4):239–247, July 1998.

[14] Dominique Attali and Annick Montanvert. Computing and simplifying 2D and 3D continuous skeletons. *Computer Vision and Image Understanding (CVIU)*, 67(3):261–273, September 1997.

[15] C.B. Barber, D.P. Dobkin, and H.T. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, December 1996.

[16] Bruce G. Baumgart. Winged-edge polyhedron representation. *Technical Report STAN-CS-320*, 1972.

[17] Mark W. Beall and Mark S. Shephard. A general topology-based mesh data structure. *International Journal for Numerical Methods in Engineering*, 40:1573–1596, 1997.

[18] Alexander Belyaev and Yutaka Ohtake. A comparison of mesh smoothing methods. In *In Proceedings of the Israel-Korea BiNational Conference on Geometric Modeling and Computer Graphics*, pages 83–87, 2003.

[19] Russell V. Benson. *Euclidean Geometry and Convexity*. McGraw-Hill, New York, 1966.

[20] Claude Berge. *Hypergraphs. Combinatorics of finite sets*, volume 45. North-Holland, 1989.

[21] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 1997.

[22] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2):239–256, Feb 1992.

[23] Dmitriy Bespalov, William C. Regli, and Ali Shokoufandeh. Reeb graph based shape retrieval for CAD. In *Proceedings of ASME Design Engineering Technical Conference (DETC)*, pages 1–10, 2003.

[24] Dmitriy Bespalov, William C. Regli, and Ali Shokoufandeh. Local feature extraction and matching partial objects. *Computer-Aided Design*, 38:1020–1037, 2006.

[25] Dmitriy Bespalov, Ali Shokoufandeh, William C. Regli, and Wei Sun. Scale-space representation and classification of 3D models. *Journal of Computing and Information Science in Engineering*, 3:315–324, 2003.

[26] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Reeb graphs for shape analysis and applications. *Theoretical Computer Science*, 392(1-3):5–22, 2008.

[27] Silvia Biasotti, Simone Marini, Michela Mortara, Giuseppe Patane, Michela Spagnuolo, and Bianca Falcidieno. 3D shape matching through topological structures. *Discrete Geometry for Computer Imagery, Lecture Notes in Computer Science*, 2886:194–203, 2003.

[28] Alberto Del Bimbo and Pietro Pala. Content-based retrieval of 3d models. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2(1):20–43, 2006.

[29] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, MA, 1987.

[30] H. Blum and R. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10(3):167–180, 1978.

[31] Harry Blum. Biological shape and visual science. *Journal of Theoretical Biology*, 38:205–287, 1973.

[32] Harry Blum and R. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10(3):167–180, 1978.

[33] Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3(4):266–286, October 1984.

[34] Matthew Bolitho, Michael Kazhdan, Randal Burns, and Hugues Hoppe. Multilevel streaming for out-of-core surface reconstruction. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 69–78, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[35] Sylvain Bouix, Kaleem Siddiqi, and Allen Tannenbaum. Flux driven automatic centerline extraction. *Medical Image Analysis*, 9:209–221, 2005.

[36] Gareth Bradshaw and Carol O'Sullivan. Adaptive medial-axis approximation for sphere-tree construction. *ACM Transactions on Graphics*, 23(1):1–26, 2004.

[37] Angela Brennecke and Tobias Isenberg. 3D shape matching using skeleton graphs. In Thomas Schulze, Stefan Schlechtweg, and Volkmar Hinz, editors, *Proceedings of Simulation and Visualization*, pages 299–310, March 2004.

[38] J. W. Bruce and P. J. Giblin. Growth, motion and 1-parameter families of symmetry sets. *Proceedings of the Royal Society of Edinburgh*, 104A:179–204, 1986.

[39] J.W. Bruce, P.J. Giblin, and F. Tari. Ridges, crests and sub-parabolic lines of evolving surfaces. *International Journal of Computer Vision (IJCV)*, 18(3):195–210, June 1996.

[40] H. Bunke and T. Caelli, editors. *International Journal of Pattern Recognition and Artificial Intelligence: Special issue on graph matching in pattern recognition and machine vision*, volume 18. World Scientific, 2004.

[41] Benjamin Bustos, Daniel A. Keim, Dietmar Saupe, Tobias Schreck, and Dejan V. Vranić. Feature-based similarity search in 3D object databases. *ACM Computing Survey*, 37(4):345–387, 2005.

[42] Frederic Cazals, Joachim Giesen, Mark Pauly, and Afra Zomorodian. Conformal alpha shapes. In *IEEE/Eurographics Symposium on Point-Based Graphics (PBG)*, pages 55–61, 2005.

[43] Ming-Ching Chang and B. Kimia. Regularizing 3D medial axis using medial scaffold transforms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[44] Ming-Ching Chang, F. Leymarie, and B. Kimia. 3D shape registration using regularized medial scaffolds. In *Proceedings of the IEEE Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 987–994, 2004.

[45] Ming-Ching Chang, F. Leymarie, and B. Kimia. Surface reconstruction from point clouds by transforming the medial scaffold. In *Proceedings of the IEEE 3-D Digital Imaging and Modeling (3DIM)*, pages 13–20, 2007.

[46] Ming-Ching Chang, Frederic F. Leymarie, and Benjamin B. Kimia. Surface reconstruction from point clouds by transforming the medial scaffold. In *submitted to Computer Vision and Image Understanding (CVIU)*. Elsevier, April 2008.

[47] Frédéric Chazal and André Lieutier. The "λ-medial axis". *Graphical Models*, 67(4):304–331, 2005.

[48] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3D model retrieval. In *Proceedings of the Eurographics*, pages 223–232, Granada, Spain, 2003.

[49] Ke Chen, Peter Giblin, and Alan Irving. *Mathematical exploration with MATLAB*. Cambridge University Press, Cambridge, UK, 1999.

[50] David Cohen-Steiner and Frank Da. A greedy Delaunay-based surface reconstruction algorithm. *The Visual Computer: International Journal of Computer Graphics*, 20:4–16, April 2004.

[51] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 2004.

[52] Thomas A. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition*. McGraw Hill, 2001.

[53] Nicu D. Cornea and et al. 3D object retrieval using many-to-many matching of curve skeletons. In *IEEE International Conference on Shape Modeling and Applications*, pages 366–371, 2005.

[54] Nicu D. Cornea and Patrick Min. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, 2006.

[55] Nicu D. Cornea, D. Silver, and R. Balasubramanian. Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer*, 21(11):945–955, 2005.

[56] Tim Culver, John Keyser, and Dinesh Manocha. Exact computation of the medial axis of a polyhedron. *Computer Aided Geometric Design*, 21:65–98, 2004.

[57] Christopher M. Cyr and Benjamin B. Kimia. 3D object recognition using shape similarity-based aspect graph. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 254–261, 2001.

[58] James Damon. The global medial structure of regions in $\mathbf{R}^3$. *Geometry and Topology*, 10:2385–2429, 2006.

[59] Petros Daras and et al. Efficient 3-D model search and retrieval using generalized 3-D radon transforms. *IEEE Transactions on Multimedia*, 8:101–114, 2006.

[60] T. K. Dey and S. Goswami. Tight cocone: A water-tight surface reconstructor. *Journal of Computing and Information Science in Engineering*, 3(4):302–307, December 2003.

[61] Tamal K. Dey and Joachim Giesen. Detecting undersampling in surface reconstruction. In *Discrete and Computational Geometry — The Goodman-Pollack Festschrift*, volume 25 of *Algorithms and Combinatorics Series*. Springer Verlag, 2003.

[62] Tamal K. Dey, Joachim Giesen, and Samrat Goswami. Shape segmentation and matching from noisy point clouds. In *IEEE/Eurographics Symposium on Point-Based Graphics (PBG)*, pages 193–199, 2004.

[63] Tamal K. Dey, Joachim Giesen, and James Hudson. Delaunay based shape reconstruction from large data. In *IEEE Symposium on Parallel and Large-Data Visualization and Graphics*, pages 19–27, 2001.

[64] Tamal K. Dey, Joachim Giesen, Edgar A. Ramos, and Bardia Sadri. Critical points of the distance to an epsilon-sampling of a surface and flow-complex-based surface reconstruction. In *ACM Proceedings of the Symposium on Computational Geometry*, pages 218–227, New York, NY, USA, 2005.

[65] Tamal K. Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. *Computational Geometry*, 35:124–141, 2006.

[66] Tamal K. Dey and Jian Sun. Defining and computing curve-skeletons with medial geodesic function. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 143–152. Eurographics Association, 2006.

[67] Tamal K. Dey and Wulue Zhao. Approximating the medial axis from the Voronoi diagram with a convergence guarantee. *Algorithmica*, 38:179–200, 2003.

[68] S.J. Dickinson, M. Pelillo, and R. Zabih, editors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI): Special Section on Graph Algorithms and Computer Vision*, volume 23. IEEE Computer Society, 2001.

[69] C. M. Eastman and S. F. Weiss. Tree structures for high dimensionality nearest neighbor searching. *Information Systems*, 7(2):115–122, 1982.

[70] Shimon Edelman. Representation is representation of similarities. *Behavioral and Brain Sciences*, 21:449–498, 1998.

[71] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shape. *ACM Transactions on Graphics*, 13(1):43–72, 1994.

[72] Michael Eigensatz, Robert W. Sumner, and Mark Pauly. Curvature-domain shape processing. *Computer Graphics Forum*, 27(2):241–250, 2008.

[73] Asi Elad and Ron Kimmel. On bending invariant signatures for surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25:1285–1295, 2003.

[74] Gershon Elber and Myung-Soo Kim. Computing rational bisectors. *IEEE Computer Graphics and Applications,*, 19(6), Nov./Dec. 1999.

[75] M. A. Eshera and K. S. Fu. A graph distance measure for image analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, 14:398–408, 1984.

[76] M. A. Eshera and K. S. Fu. A measure of similarity between attributed relational graphs for image analysis. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 75–77, 1984.

[77] M. A. Eshera and K. S. Fu. An image understanding system using Attributed Symbolic Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(5):604–618, Sept. 1986.

[78] Olaf Etzmuss, Joachim Gross, and Wolfgang Strasser. Deriving a particle system from continuum mechanics for the animation of deformable objects. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):538–550, 2003.

[79] Leila De Floriani and Annie Hui. A scalable data structure for three-dimensional non-manifold objects. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 72–82, Aire-la-Ville, Switzerland, Switzerland, 2003.

[80] Leila De Floriani, Paola Magillo, Enrico Puppo, and Davide Sobrero. A multi-resolution topological representation for non-manifold meshes. In *ACM Symposium on Solid Modeling and Applications*, pages 159–170, New York, NY, USA, 2002.

[81] Pasquale Foggia, Carlo Sansone, and Mario Vent, editors. *Pattern Recognition Letters: Special issue on graph based representations*, volume 24. Elsevier, 2003.

[82] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer graphics (2nd ed. in C): principles and practice*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.

[83] Thomas Funkhouser and Michael Kazhdan. Shape-based retrieval and analysis of 3D models. In *ACM SIGGRAPH Course Notes*, page 16, 2004.

[84] Thomas Funkhouser, Patrick Min, Michael Kazhdan, Joyce Chen, Alex Halderman, David Dobkin, and David Jacobs. A search engine for 3D models. *ACM Transactions on Graphics*, 22(1):83–105, 2003.

[85] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics*, 25(1):130–150, 2006.

[86] Peter Giblin and Benjamin Kimia. On the intrinsic reconstruction of shape from its symmetries. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(7):895–911, 2003.

[87] Peter Giblin and Benjamin Kimia. A formal classification of 3D medial axis points and their local geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(2):238–251, 2004.

[88] Peter Giblin and Benjamin Kimia. Transitions of the 3D medial axis under a one-parameter family of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, in press, September 2008.

[89] Peter J. Giblin and Benjamin. B. Kimia. On the local form and transitions of symmetry sets, medial axes, and shocks. *International Journal of Computer Vision (IJCV)*, 54(1-3):143–157, 2003.

[90] Peter J. Giblin and Benjamin B. Kimia. Local forms and transitions of the medial axis. In Kaleem Siddiqi and Stephen Pizer, editors, *Medial Representations: Mathematics, Algorithms and Applications*, page In Press. Kluwer Academic Publishers, 2007.

[91] J. Giesen and M. John. Surface reconstruction based on a dynamical system. *Computer Graphics Forum*, 21(3):363–371, September 2002.

[92] Joachim Giesen, Edgar A. Ramos, and Bardia Sadri. Medial axis approximation and unstable flow complex. In *Proceedings of the ACM Symposium on Computational Geometry*, pages 327–336, New York, NY, USA, 2006.

[93] Afzal Godil. Advanced human body and head shape representation and analysis. In *Digital Human Modeling, HCII*, pages 92–100. Springer Berlin Heidelberg, 2007.

[94] Afzal Godil and Sandy Ressler. Similarity based retrieval from a 3d human database. In *ACM SIGGRAPH 2005*, page 79, New York, NY, USA, 2005.

[95] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(4):377–388, 1996.

[96] Samrat Goswami, Tamal K. Dey, and Chandrajit L. Bajaj. Identifying flat and tubular regions of a shape by unstable manifolds. In *Proceedings of the ACM Symposium on Solid and Physical Modeling (SPM)*, pages 27–37, 2006.

[97] M. Akif Gulsun and Huseyin Tek. Geometric modeling of tubular structures. In *Proceedings of the IEEE Mathematical Methods in Biomedical Image Analysis (MMBIA)*, page To Appear, 2008.

[98] P. L. Halliman, G. G. Gordon, A. L. Yuille, Peter J. Giblin, and D. Mumford. *Two- and Three-Dimensional Patterns of the Face*. A. K. Peters, 1999.

[99] Masaki Hilaga, Yoshihisa Shinagawa, Taku Kohmura, and Tosiyasu L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. In *Proceedings of the ACM SIGGRAPH*, pages 203–212, 2001.

[100] M. Hisada, A. Belyaev, and T. Kunii. A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Computer Graphics Forum (Proc. Eurographics)*, 21:689–700, 2002.

[101] M. Hisada, A. G. Belyaev, and T. L. Kunii. Towards a singularity-based shape language: ridges, ravines, and skeletons for polygonal surfaces. *Soft Computing*, 7(1):45–52, 2002.

[102] Hisamoto Hiyoshi. Greedy beta-skeleton in three dimensions. In *Proceedings of the IEEE International Symposium on Voronoi Diagrams in Science and Engineering*, pages 101–109, Washington, DC, USA, 2007.

[103] C. M. Hoffmann. How to construct the skeleton of CSG objects. In *The Mathematics of Surfaces IV*, pages 421–438. Oxford University Press, 1994.

[104] Robert Hummel and Steven Warren Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6:267–287, 1983.

[105] Cheuk Yiu Ip and Satyandra K. Gupta. Retrieving matching CAD models by using partial 3D point clouds. *Computer-Aided Design and Applications*, 4:629–638, 2007.

[106] Natraj Iyer, Subramaniam Jayanti, Kuiyang Lou, Yagnanarayanan Kalyanaraman, and Karthik Ramani. A multi-scale hierarchical 3D shape representation for similar shape retrieval. In *Proceeding of the Tools and Methods of Competitive Engineering*, pages 1117–1118, 2004.

[107] Natraj Iyer, Subramaniam Jayanti, Kuiyang Lou, Yagnanarayanan Kalyanaraman, and Karthik Ramani. Shape-based searching for product lifecycle applications. *Computer-Aided Design*, 37:1435–1446, 2005.

[108] Natraj Iyer, Subramaniam Jayanti, Kuiyang Lou, Yagnanarayanan Kalyanaraman, and Karthik Ramani. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design*, 37:509–530, 2005.

[109] Marc Johannes, Thomas Sebastian, Huseyin Tek, and Benjamin Kimia. Perceptual organization as object recognition divided by two. In *IEEE Workshop on Perceptual Organization in Computer Vision (POCV)*, pages 41–46, 2001.

[110] Andrew Edie Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(5):433–449, 1999.

[111] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 156–164, 2003.

[112] D. G. Kendall. The diffusion of shape. *Advances in Applied Probability*, 9(3):428–430, 1977.

[113] Lutz Kettner. Using generic programming for designing a data structure for polyhedral surfaces. *Computational Geometry Theory and Applications*, 13:65–90, 1999.

[114] Lutz Kettner. Halfedge data structures. In *CGAL User and Reference Manual*. CGAL Editorial Board, 3.3 edition, 2007.

[115] Benjamin B. Kimia. On the role of medial geometry in human vision. *Journal of Physiology-Paris*, 97(2–3):155–190, 2003.

[116] Benjamin B. Kimia, Allen R. Tannenbaum, and Steven W. Zucker. Shapes, shocks, and deformations, I: The components of shape and the reaction-diffusion space. *International Journal of Computer Vision (IJCV)*, 15(3):189–224, 1995.

[117] Ron Kimmel and James A. Sethian. Fast Voronoi diagrams and offsets on triangulated surfaces. In *Computer Vision and Image Understanding (CVIU)*, 1999.

[118] Jan J. Koenderink. *Solid Shape*. MIT Press, Cambridge, Massachusetts, 1990.

[119] Marcel Kortgen, Gil-Joo Park, Marcin Novotni, and Reinhard Klein. 3D shape matching with 3D shape contexts. In *Central European Seminar on Computer Graphics*, 2003.

[120] J .J. Kosowsky and A. L. Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7:477–490, 1994.

[121] Chuan-Chu Kuo and Hong-Tzong Yau. A new conbinatorial approach to surface reconstruction with sharp features. *IEEE Transactions on Visualization and Computer Graphics*, 12(1):73–82, 2006.

[122] E. Lawler and D. Wood. Branch and bound methods: A survey. *Operations Reasearch*, 14:699–719, 1966.

[123] Soochahn Lee, Sehyuk Yoon, Il Dong Yun, Duck Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee. A new 3-D model retrieval system based on aspect-transition descriptor. In *European Conference on Computer Vision (ECCV)*, pages 543–554, 2006.

[124] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project: 3D scanning of large statues. In Kurt Akeley, editor, *Proceedings of the ACM SIGGRAPH*, pages 131–144, 2000.

[125] Frederic Leymarie and Benjamin Kimia. The medial scaffold of 3D unorganized point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(2):313–330, 2007.

[126] Frederic Leymarie, Benjamin Kimia, and Peter Giblin. Towards surface regularization via medial axis transitions. In *International Conference on Pattern Recognition (ICPR)*, pages 123–126, 2004.

[127] Frederic F. Leymarie and Benjamin B. Kimia. From the infinitely large to the infinitely small: Applications of medial symmetry representations of shape. In Kaleem Siddiqi and Stephen Pizer, editors, *Medial Representations: Mathematics, Algorithms and Applications*, page In Press. Kluwer Academic Publishers, 2007.

[128] Frederic Fol Leymarie. *3D Shape Representation via Shock Flows*. PhD thesis, Division of Engineering, Brown University, 2003.

[129] Frederic Fol Leymarie. Thoughts on shape. *Visual Thought: The Depictive Space of the Mind, L. Albertazzi, ed. of Advances in Consciousness Research series*, 67:303–350, 2006.

[130] M. Leyton. *Symmetry, Causality, Mind*. MIT press, Cambridge, MA, USA, 1992.

[131] Michael Leyton. *A Generative Theory of Shape*. Lecture Notes in Computer Science 2145. Springer-Verlag, 2001.

[132] André Lieutier. Any open bounded subset of $\mathbf{R}^n$ has the same homotopy type as its medial axis. *Computer-Aided Design*, 36:1029–1046, 2004.

[133] M. A. Lozano and F. Escolano. Protein classification by matching and clustering surface graphs. *Pattern Recognition*, 39(4):539–551, 2006.

[134] G. Malandain and S. Fernandez-Vidal. Euclidean skeletons. *Image and Vision Computing*, 16(5):317–327, 1998.

[135] Jean-François Mangin, D. Rivière, A. Cachia, D. Papadopoulos-Orfanos, D. L. Collins, A. C. Evans, and J. Régis. Object-based morphometry of the cerebral cortex. *IEEE Transactions on Medical Imaging*, 23(8):968–982, August 2004.

[136] David Marr. *Vision: a computational investigation into the human representation and processing of visual information*. W.H. Freeman, San Francisco, 1982.

[137] Jonah McBride. Archaeological fragment reassembly using curve matching. Master's dissertation, Brown University, Providence, USA, 2003.

[138] Boris Mederos, Nina Amenta, Luiz Velho, and Luiz Henrique de Figueiredo. Surface reconstruction from noisy point clouds. *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP)*, pages 53–62, 2005.

[139] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3D geometry. *ACM Transactions on Graphics*, 25(3):560–568, 2006.

[140] Douglas C Moore, Joseph J Crisco, Theodore G Trafton, and Evan L Leventhal. A digital database of wrist bone anatomy and carpal kinematics. *Journal of Biomechanics*, 40(11):2537–42, 2007.

[141] David B. Mumford. Mathematical theories of shape: Do they model perception? In *Proc. Geometric Methods in Computer Vision Conference*, volume SPIE–1570, pages 2–10, 1991.

[142] L. R. Nackman. Two–dimensional critical point configuration graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6(4):442–450, July 1984.

[143] M. Naf, G. Szekely, R. Kikinis, M.E. Shenton, and O. Kubler. 3D Voronoi skeletons and their usage for the characterization and recognition of 3D organ shape. *Computer Vision and Image Understanding (CVIU)*, 66(2):147–161, May 1997.

[144] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. In *Proceedings of the ACM SIGGRAPH*, pages 609–612, 2004.

[145] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: Concepts and applications of Voronoi diagrams*. Probability and Statistics. Wiley, NYC, 2nd edition, 2000. 671 pages.

[146] Joseph O'Rourke. *Computational Geometry in C*. Cambridge University Press, 2nd edition, 1998.

[147] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics*, 21(4):807–832, 2002.

[148] Ozge C. Ozcanli and Benjamin B. Kimia. Generic object recognition via shock patch fragments. In *British Machine Vision Conference*, pages 1030–1039, 2007.

[149] Ozge C. Ozcanli, Amir Tamrakar, Benjamin B. Kimia, and Joseph L. Mundy. Augmenting shape with appearance in vehicle category recognition. In *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 935–942, 2006.

[150] David S. Paik, Christopher F. Beaulieu, R. Brooke Jeffrey, Geoffrey D. Rubin, and Sandy Napel. Automated flight path planning for virtual endoscopy. *Medical Physics*, 25(5):629–637, May 1998.

[151] Panagiotis Papadakis, Ioannis Pratikakis, Stavros Perantonis, and Theoharis Theoharis. Efficient 3D shape matching and retrieval using a concrete radialized spherical projection representation. *Pattern Recognition*, 40:2437–2452, 2007.

[152] Si Hyung Park, Seoung Soo Lee, and Jong Hwa Kim. A surface reconstruction algorithm using weighted alpha shapes. In *Lecture Notes in Computer Science*, volume 3613, pages 1141–1150. Springer, 2005.

[153] N. M. Patrikalakis and T. Maekawa. *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer, 2002.

[154] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. Guibas. Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics*, 27(3), August 2008. Article No. 43.

[155] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. Guibas. Skeleton extraction by mesh contraction. *ACM Transactions on Graphics*, 27(3), August 2008. Article No. 44.

[156] S. Petitjean and E. Boyer. Regular and non-regular point sets: Properties and reconstruction. *Computational Geometry: Theory and Application*, 19(2-3):101–126, 2001.

[157] Chandan Pitta and Michael M. Marefat. Intelligent retrieval and reuse of CAD solid models. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1413–1418, 2005.

[158] Stephen Pizer, Kaleem Siddiqi, Gabor Székely, James Damon, and Steven Zucker. Multiscale medial loci and their properties. *International Journal of Computer Vision (IJCV)*, 55(2-3):155–179, 2003.

[159] Stephen M. Pizer, P. Thomas Fletcher, Sarang Joshi, Andrew Thall, James Z. Chen, Yonatan Fridman, Daniel S. Fritsch, A. Graham Gash, John M. Glotzer, Michael R. Jiroutek, Conglin Lu, Keith E. Muller, Gregg Tracton, Paul Yushkevich, and Edward L. Chaney. Deformable m-Reps for 3D medical image segmentation. *International Journal of Computer Vision (IJCV)*, 55(2-3):85–106, 2003.

[160] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas Funkhouser. A planar-reflective symmetry transform for 3D shapes. In *Proceedings of the ACM SIGGRAPH*, pages 549–559, 2006.

[161] Anthony Pollitt, Peter Giblin, and Benjamin Kimia. Consistency conditions on medial axis. In *European Conference on Computer Vision (ECCV)*, pages 530–541, 2004.

[162] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry — An Introduction*. Texts and Monographs in Computer Science. Springer-Verlag, 1985.

[163] Stanley Rabinowitz. A polynomial curve of constant width. *Missouri Journal of Mathematical Sciences*, 9:23–27, 1997.

[164] Edgar A. Ramos and Bardia Sadri. Geometric and topological guarantees for the wrap reconstruction algorithm. In *Proceedings of the ACM-SIAM Sympoisum on Discrete Algorithms (SODA)*, pages 1086–1095, 2007.

[165] A. Rangarajan and E. Mjolsness. A Lagrangian relaxation network for graph matching. In *IEEE International Conference on Neural Networks (ICNN)*, volume 7, pages 4629–4634. IEEE Press, 1994.

[166] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Proceedings of the IEEE 3-D Digital Imaging and Modeling (3DIM)*, pages 145–152, Quebec City, Canada, May 2001.

[167] R. Schnabel, R. Wahl, R. Wessel, and R. Klein. Shape recognition in 3D point clouds. *Technical Report, University of Bonn*, 2007.

[168] Thomas Sebastian, Philip Klein, and Benjamin Kimia. On aligning curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(1):116–125, January 2003.

[169] Thomas Sebastian, Philip Klein, and Benjamin Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26:551–571, 2004.

[170] Thomas B. Sebastian, Hüseyin Tek, Joseph J. Crisco, Scott W. Wolfe, and Benjamin B. Kimia. Segmentation of carpal bones from 3D CT images using skeletally coupled deformable models. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 1184–1194, London, UK, 1998. Springer-Verlag.

[171] Jean Serra. *Image Analysis and Mathematical Morphology*, volume 1. Academic Press, 1982.

[172] Ariel Shamir and Amir Shaham. Skeleton based solid representation with topology preservation. *Graphical Models*, 68(3):307–321, 2006.

[173] L. G. Shapiro and R. M. Haralick. Structual decriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 3:504–519, Sept. 1981.

[174] Andrei Sharf, Thomas Lewiner, Ariel Shamir, and Leif Kobbelt. On-the-fly curve-skeleton computation for 3D shapes. *Computer Graphics Forum*, 26(3):323–328, 2007.

[175] Daniel Sharvit, Jacky Chan, Hüseyin Tek, and Benjamin B. Kimia. Symmetry-based indexing of image databases. *Journal of Visual Communication and Image Representation*, 9(4):366–380, December 1998.

[176] E. Sherbrooke, N. M. Patrikalakis, and F.-E. Wolter. Differential and topological properties of medial axis transforms. *Graphical Models and Image Processing*, 58(6):574–592, November 1996.

[177] Yonggang Shi, Rongjie Lai, Sheila Krishna, Nancy Sicotte, Ivo Dinov, and Arthur W. Toga. Anisotropic Laplace-Beltrami eigenmaps: Bridging Reeb graphs and skeletons. In *Proceedings of the IEEE Mathematical Methods in Biomedical Image Analysis (MMBIA)*, 2008.

[178] Jau-Ling Shih, Chang-Hsing Lee, and Jian Tang Wang. A new 3D model retrieval approach based on the elevation descriptor. *Pattern Recognition*, 40(1):283–295, 2007.

[179] Ali Shokoufandeh, Diego Macrini, Sven Dickinson, Kaleem Siddiqi, and Steven W. Zucker. Indexing hierarchical structures using graph spectra. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(7):1125–1140, 2005.

[180] Kaleem Siddiqi, Sylvain Bouix, Allen Tannenbaum, and Steven W. Zucker. Hamilton-jacobi skeletons. *International Journal of Computer Vision (IJCV)*, 48(3):215–231, 2002.

[181] Kaleem Siddiqi and Benjamin B. Kimia. Parts of visual form: Computational aspects. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(3):239–251, March 1995.

[182] Kaleem Siddiqi and Benjamin B. Kimia. A shock grammar for recognition. In *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 507–513, 1996.

[183] Kaleem Siddiqi, Juan Zhang, Diego Macrini, Ali Shokoufandeh, Sylvain Bouix, and Sven Dickinson. Retrieving articulated 3-D models using medial surfaces. *Machine Vision and Applications*, pages 1432–1769, 2007.

[184] D. Siersma and M. van Manen. The nine morse generic tetrahedra. *Manuscript, Math.DG/0410251*, October 2004.

[185] P. D. Simic. Constrained nets for graph matching and other quadratic assignment problems. *Neural Computation*, 3:268–281, 1991.

[186] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Annals of Mathematical Statistics*, 35:876–879, 1964.

[187] Svetlana Stolpner and Kaleem Siddiqi. Revealing significant med. struct. in polyhedral meshes. In *Proceedings of the IEEE Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 365–372, 2006.

[188] Martin Styner, Guido Gerig, Sarang Joshi, and Stephen Pizer. Automatic and robust computation of 3d medial models incorporating object variability. *International Journal of Computer Vision (IJCV)*, 55(2-3):107–122, 2003.

[189] Avneesh Sud, Mark Foskey, and Dinesh Manocha. Homotopy-preserving medial axis simplification. In *Proceedings of the ACM Symposium on Solid and Physical Modeling (SPM)*, pages 39–50, 2005.

[190] Avneesh Sud, Miguel A. Otaduy, and Dinesh Manocha. Difi: Fast 3d distance field computation using graphics hardware. *Computer Graphics Forum (Proc. Eurographics)*, 23(3):557–566, 2004.

[191] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. In *Proceedings of the ACM SIGGRAPH*, page 80, 2007.

[192] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson. Skeleton based shape matching and retrieval. In *Proceedings of the Shape Modeling International*, pages 130–139, 2003.

[193] Roger Tam and Wolfgang Heidrich. Feature-preserving medial axis noise removal. In *European Conference on Computer Vision (ECCV)*, pages 672–686, London, UK, 2002. Springer-Verlag.

[194] Roger Tam and Wolfgang Heidrich. Shape simplification based on the medial axis transform. In *Proceedings of IEEE Visualization*, pages 481–488, 2003.

[195] Amir Tamrakar, Ming-Ching Chang, and Benjamin B. Kimia. Computation of exact 2D medial axis transform from piecewise circular inputs via dynamic wavefront and shock propagation. In *manuscript*, Providence, RI, USA, July, 2008.

[196] Amir Tamrakar and Benjamin B. Kimia. Medial visual fragments as an intermediate image representation for segmentation and perceptual grouping. In *Proceedings of the Workshop on Perceptual Organization in Computer Vision (POCV)*, page 47, 2004.

[197] Johan W. H. Tangelder and Remco C. Veltkamp. A survey of content based 3D shape retrieval methods. In *Shape Modeling International*, pages 145–156, 2004.

[198] M. J. Tarr, P. Williams, W. G. Hayward, and I. Gauthier. Three-dimensional object recognition is viewpoint dependent. *Nature Neuroscience*, 1(4):275–277, 1998.

[199] Gabriel Taubin. A signal processing approach to fair surface design. In *Proceedings of the ACM SIGGRAPH*, pages 351–358, 1995.

[200] Huseyin Tek and Benjamin Kimia. Boundary smoothing via symmetry transforms. *Journal of Mathematical Imaging and Vision*, 14(3):211–223, May 2001.

[201] Huseyin Tek and Benjamin B. Kimia. Curve evolution, wave propagation, and mathematical morphology. In Henk J.A.M. Heijmans and Jos B.T.M. Roerdink, editors, *Mathematical Morphology and its Applications to Image and Signal Processing*, volume 12 of *Computational Imaging and Vision*, pages 115–126. Kluwer Academic, Amsterdam, The Netherlands, June 1998.

[202] Huseyin Tek and Benjamin B. Kimia. Symmetry maps of free-form curve segments via wave propagation. *International Journal of Computer Vision (IJCV)*, 54(Issue 1-3):35–81, August 2003.

[203] Nhon Trinh and Benjamin Kimia. A symmetry-based generative model for shape. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

[204] S. Umeyama. An eigendecomposition aproach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 10:695–703, September 1988.

[205] Peter L. Varkonyi and G. Domokos. Mono-monostatic bodies: the answer to arnold's question. *The Mathematical Intelligencer*, 28(4):34–38, 2006.

[206] Annamária R. Várkonyi-Kóczy. Fuzzy logic supported corner detection. *Journal of Intelligent and Fuzzy Systems*, 19(1):41–50, 2008.

[207] A. Verroust and F. Lazarus. Extracting skeletal curves from 3D scattered data. *The Visual Computer: International Journal of Computer Graphics*, 16(1):15–25, 2000.

[208] Lawson Wade and Richard E. Parent. Automated generation of control skeletons for use in animation. *The Visual Computer: International Journal of Computer Graphics*, 18(2):97–110, March 2002.

[209] Jianning Wang, Manuel M. Oliveira, and Arie E. Kaufman. Reconstructing manifold and non-manifold surfaces from point clouds. In *Proceedings of IEEE Visualization*, pages 415–422, 2005.

[210] Yu-Shuen Wang and Tong-Yee Lee. Curve-skeleton extraction using iterative least squares optimization. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):926–936, 2008.

[211] Liu Wen-Yu, Li Hua, and Zhu Guang-Xi. A fast algorithm for corner detection using the morphologic skeleton. *Pattern Recognition Letters*, 22:891–900, 2001.

[212] A. Willis, S. Andrews, J. Baker, Y. Cao, D. Han, K. Kang, W. Kong, Frederic Fol Leymarie, X. Orriols, S. Velipasalar, E.L. Vote, D.B. Cooper, M.S. Joukowsky, B.B. Kimia, D.H. Laidlaw, and D. Mumford. Assembling virtual pots from 3D measurements of their fragments. In *Proceedings of the Conference on Virtual Reality, Archeology, and Cultural Heritage*, 2001.

[213] G. V. Wilson and G. S. Pawley. On the stablity of the traveling salesman problem algorithm of hopfield and tank. *Biological Cybernetics*, 58:63–70, 1988.

[214] F.-E. Wolter and K.-I. Friese. Local and global geometric methods for analysis interrogation, reconstruction, modification and design of shape. In *Proceedings of IEEE Computer Graphics International (CGI)*, pages 137–151, Geneva, Switzerland, June 2000.

[215] Franz-Erich Wolter. Cut locus and medial axis in global shape interrogation and represent. Technical Report, MIT, 1993.

[216] Fu-Che Wu, Wan-Chun Ma, Rung-Huei Liang, Bing-Yu Chen, and Ming Ouhyoung. Domain connected graph: the skeleton of a closed 3D shape for animation. *The Visual Computer: International Journal of Computer Graphics*, 22(2):117–135, 2006.

[217] Joris Vanden Wyngaerd and Luc Van Gool. Automatic crude patch registration: Toward automatic 3D model building. *Computer Vision and Image Understanding (CVIU)*, 87(1-3):8–26, July 2002.

[218] Shin Yoshizawa, Alexander Belyaev, and Hans-Peter Seidel. Fast and robust detection of crest lines on meshes. In *Proceedings of the ACM Symposium on Solid and Physical Modeling (SPM)*, pages 227–232, 2005.

[219] Shin Yoshizawa, Alexander Belyaev, and Hans-Peter Seidel. Skeleton-based variational mesh deformations. *Computer Graphics Forum (Proc. Eurographics)*, 26(3):255–264, 2007.

[220] Shin Yoshizawa, Alexander Belyaev, Hideo Yokota, and Hans-Peter Seidel. Fast and faithful geometric algorithm for detecting crest lines on meshes. In *Proceedings of the IEEE Pacific Conference on Computer Graphics and Applications*, pages 231–237, 2007.

[221] Shin Yoshizawa, Alexander Belyaev, Hideo Yokota, and Hans-Peter Seidel. Fast, robust, and faithful methods for detecting crest lines on meshes. *Computer Aided Geometric Design*, page in press, 2008.

[222] Shin Yoshizawa, Alexander G. Belyaev, and Hans-Peter Seidel. Free-form skeleton-driven mesh deformations. In *Proceedings of the ACM Symposium on Solid Modeling and Applications*, pages 247–253, 2003.

[223] Dimitrios Zarpalas, Petros Daras, Apostolos Axenopoulos, Dimitrios Tzovaras, and Michael G. Strintzis. 3D model search and retrieval using the spherical trace transform. *EURASIP Journal on Applied Signal Processing*, 2007(1):207–207, 2007.

[224] Ron Zass and Amnon Shashua. Probabilistic graph and hypergraph matching. In *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[225] Mourad Zerroug and Ramakant Nevatia. Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(3):237–253, 1996.

[226] Qian-Yi Zhou, Tao Ju, and Shi-Min Hu. Topology repair of solid models using skeletons. *IEEE Transactions on Visualization and Computer Graphics*, 13:675–685, 2007.

[227] Michael Zilske, Hans Lamecker, and Stefan Zachow. Adaptive remeshing of non-manifold surfaces. In *Proceedings of the Eurographics*, volume 27, pages 393–402, 2008.