國立臺灣大學電資學院資訊工程研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

布偶外觀縫線導向之網格分割方法

Appearance-aware seam generation for plush fabrication

饒亢

Kang Jao

指導教授：陳炳宇博士

Advisor: Bing-Yu Chen, Ph.D.

中華民國 104 年 7 月

July, 2015

國立臺灣大學碩士學位論文
口試委員會審定書

布偶外觀縫線導向之網格分割方法

Appearance-aware Mesh Segmentation for Plush Fabrication

本論文係饒　亢君（學號 R02922046）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 104 年 7 月 20 日承下列考試委員審查通過及口試及格，特此證明

口試委員：　[illegible]
（指導教授）

林文杰　　[illegible]

朱宏國

系 主 任　趙坤茂

致謝

感謝 Latias，讓我又多做了一隻布偶。另外我親愛的前室友楊淳文說他想出現在這裡。

中文摘要

絨毛布偶是很多人的童年回憶，布偶因為獨特的觸感和可愛的外觀而深受小朋友們的喜愛。然而，布偶的設計並不容易，設計師必須在布料上剪成特定的平面形狀才能將平面的布料縫成想要的立體形狀，而這些平面形狀我們稱之為「版型」。傳統上版型的設計必須經過多次的錯誤嘗試才能完成，整個過程既繁瑣又費時。目前雖然有自動的演算法可以從立體形狀逆推出版型，但這些版型卻沒有考慮到最後成品的外觀，使布偶表面充滿不規則、疤痕似的縫線，進而影響布偶的形象。本研究取出影響布偶外觀的因素作為權重，使用最短路徑最佳化縫線的路徑，再利用簡單的貪婪法消去不必要的縫線。最後的結果除了確保產生的版型可以縫成想要的立體形狀外，還減少了縫線的不自然感。文末將附上使用此方法縫製的實際成品。

關鍵字：布偶、縫線、實體化

Abstract

Stuffed toys, aka plushes, are children’ s memories in their childhood. These toys are adored by children because of their cute appearance and soft touches. In addition to the shape, the seams on the surface of plush also affect the appearance of plush. Poorly designed plush may result in unnatural seams, leaving scar-like feeling in the first impression. Previous works only focused on solving the developable problem while visual defects caused by boundary trenches remain unsolved. In this paper, we proposed a new segmentation method minimizing these defects by considering structural properties of the given shape. The resulting boundaries are less noticeable to human eyes while the overall segmentation remain quasi-developable so that it can be used for plush fabrication. We evaluate the proposed method on a variety of 3D models with physically fabricated results.

Key words: plush, seam, fabrication

Contents

List of Figures

List of Tables

Chapter 1

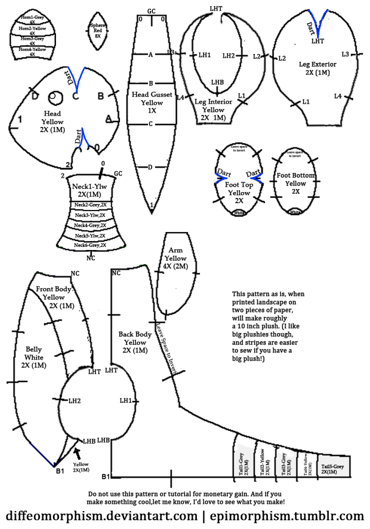
Introduction

1.1 Plush making

Figure 1.1: An example of different shapes of plush[1]

Stuffed toys, aka plushes, are children's memories in their childhood. These toys are adored by children because of their cute appearance and soft touches. Since everyone has his(her) own preference, there should be many different types and shapes of plush for people to choose. However, it's not an easy task to design plush for complex shapes.

(a) The target shape[2]

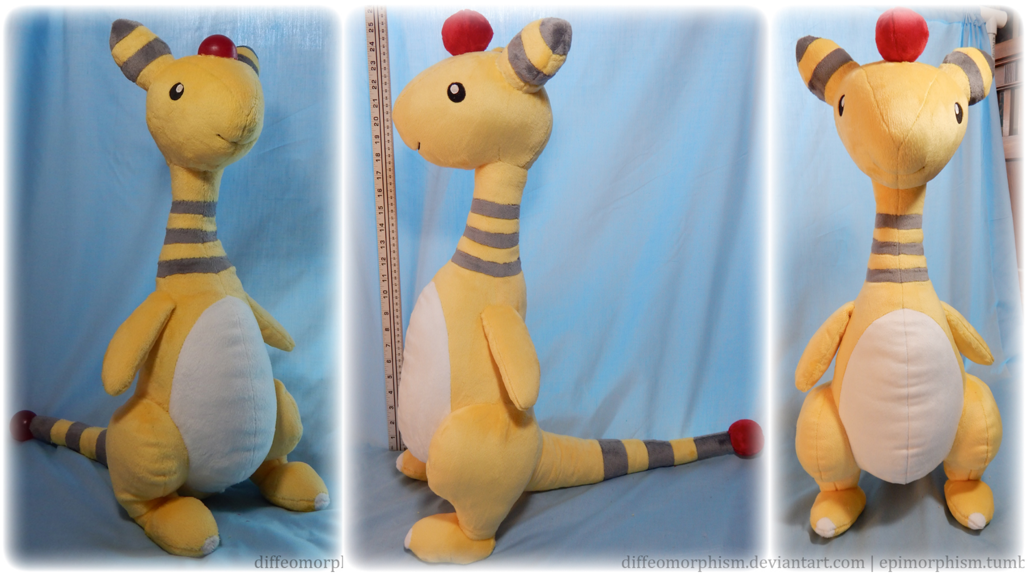


This pattern as is, when printed landscape on two pieces of paper, will make roughly a 10 inch plush. (I like big plushies though, and stripes are easier to sew if you have a big plush!)

Do not use this pattern or tutorial for monetary gain. And if you make something cool,let me know, I'd love to see what you make!

diffeomorphism.deviantart.com | epimorphism.tumblr.com

(b) Corresponding pattern[3]



(c) Fabrication result[3]

Figure 1.2: The workflow of fabrication from planar materials.

To make a plush like (Figure 1.2(a)), one must cut fabrics into a specific shape, *e.g.,* pattern (Figure 1.2(b)) and then assembled them back into 3D (Figure 1.2(c)). If the pattern is known, the only remaining works are cutting and sewing. The whole problem of plush making is essentially how to draw the patterns for different shapes.

[1]http://fishlover.deviantart.com/

[2]http://cdn.bulbagarden.net/

[3]http://diffeomorphism.deviantart.com/

1.2 Fabrication from planar materials

(a) Funiture[4]

(b) Garment[5]

(c) Leather products[6]

(d) Plush[7]

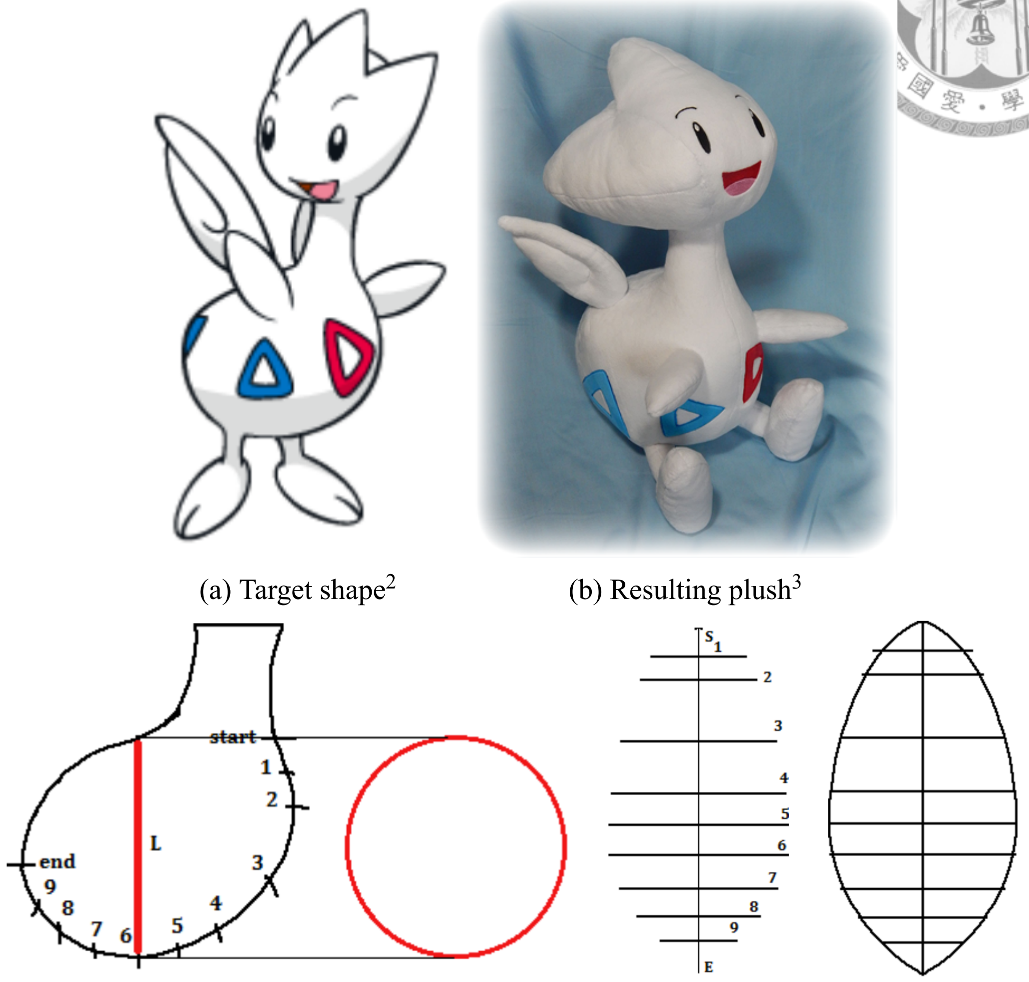
Figure 1.3: Examples of fabrications using planar materials.

In industrial fabrication, forming a 3D target shape from planar materials is an important topic. Such process can be found from furniture, garments, leather products, plushes (Figure 1.3(a)-(d)), etc. To achieve this goal, the target shape is split into smaller parts and flattened into planar charts. These flattened charts are called "pattern" (Figure 1.2(b)) and can be used for fabrication. Since the target shape (Figure 1.2(a)) is usually greatly different from its corresponding pattern (Figure 1.2(b)), making patterns requires much experience and imagination thus is far from a easy task.

Two methods have been developed to solve this problem. They are *estimation* and *drapping*.

[4]http://www.universalupholstering.com/img/top_center_pic.png
[5]http://clicktin.com/8421-13306-thickbox/garment-dyed-polo-indigo.jpg
[6]http://pimg.tradeindia.com/00078234/b/0/Scan-Leather-Bag.jpg
[7]http://foxkeh.jp/blog/2007/08/10/00.jpg

1.2.1 Shape estimation



(a) Target shape[2] (b) Resulting plush[3]

(c) Side view of the body[3] (d) Imaginary top view with lines[3] (e) Resulting top view[3]

Figure 1.4: The workflow using shape estimation

Designers sometimes use shape estimation to draw patterns if they have a rough sense of target shape. Cube, circles, and other primitives can be used for estimating the overall shape of different part of the target shape. In the case of Togetic Figure 1.4(a) plush design, because it's body is close to a non-developable sphere, it's impossible to use only two symmetric pieces of side view pattern like Figure 1.4(b). To solve this problem, more patterns are required to cover this part. An intuitive idea is to draw another top/bottom view pattern so that each pattern is much close to developable surface. We assume the body part is a sphere and the diameter are the same from any viewing angle. That is, the length of line L at 6 in Figure 1.4(c) should be the same length of line 6 in Figure 1.4(d). By this estimation, we can draw all of the lines in Figure 1.4(d) and finally draw the complete top view pattern Figure 1.4(e). Patterns of other parts are drawn using the same technique and the result is Figure 1.4(e).

This technique has the advantage of simplicity since it only requires some reference photo. However, it comes at the price of inaccuracy. Estimation actually assume all shape are the composite of simple primitives. This is obviously not true for all shapes. Also, designers should be aware of overlapping area between side view and top/bottom view. Some modification is needed to remove the overlapping area, but it's unclear how much should be removed for each pattern.

1.2.2 Drapping

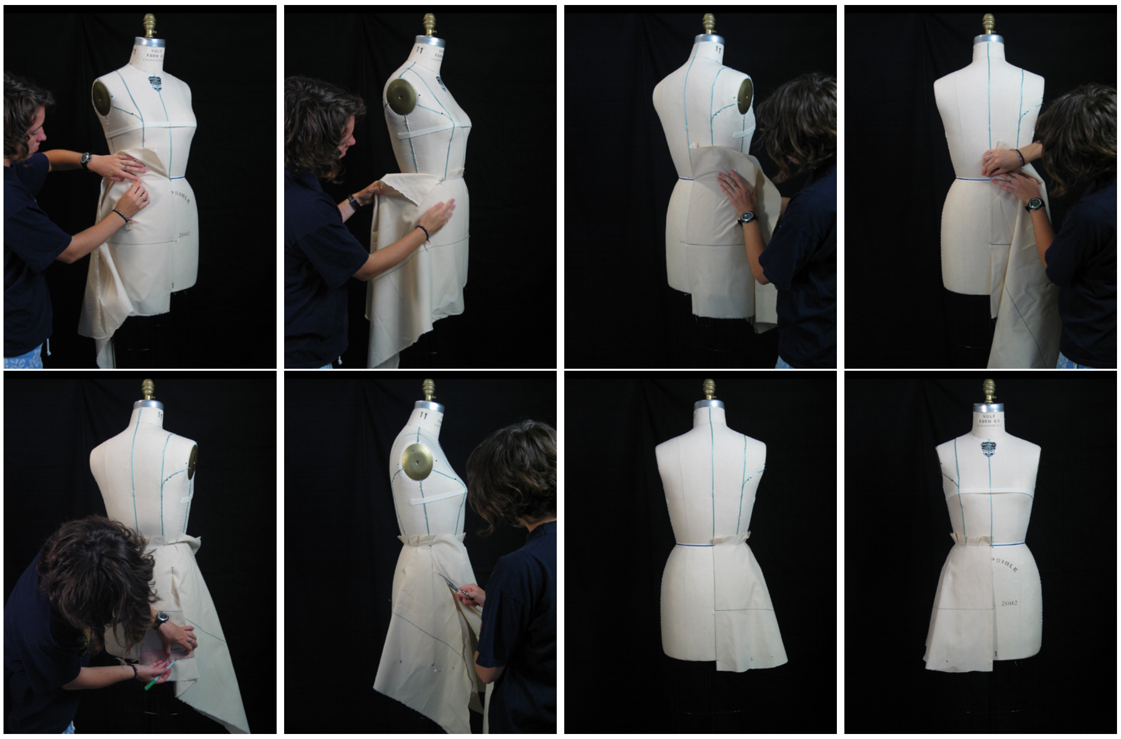


Figure 1.5: The process of pattern design using draping[8]

A technique called draping (Figure 1.5) in sewing is developed to solve this problem. Draping means putting the planar material onto the physical target shape model (*e.g.,*a human dummy) and try to fit it by cutting the material into appropriate shapes. In such way, one can instantly evaluate the result of intermediate pattern and modify if needed.

[8]https://fit.cit.cornell.edu/textiles/draping/drapes/flaredskir/steps/drapingpro/default.html

Draping significantly reduce the efforts of pattern design, but it has the disadvantage of requiring a physical model. Unlike garment design, plush design does not have a common shape, *e.g.,*human and can not reuse the physical model. Making a physical model every time before plush design is not desired since it's also time consuming.

1.3 Appearance-aware design

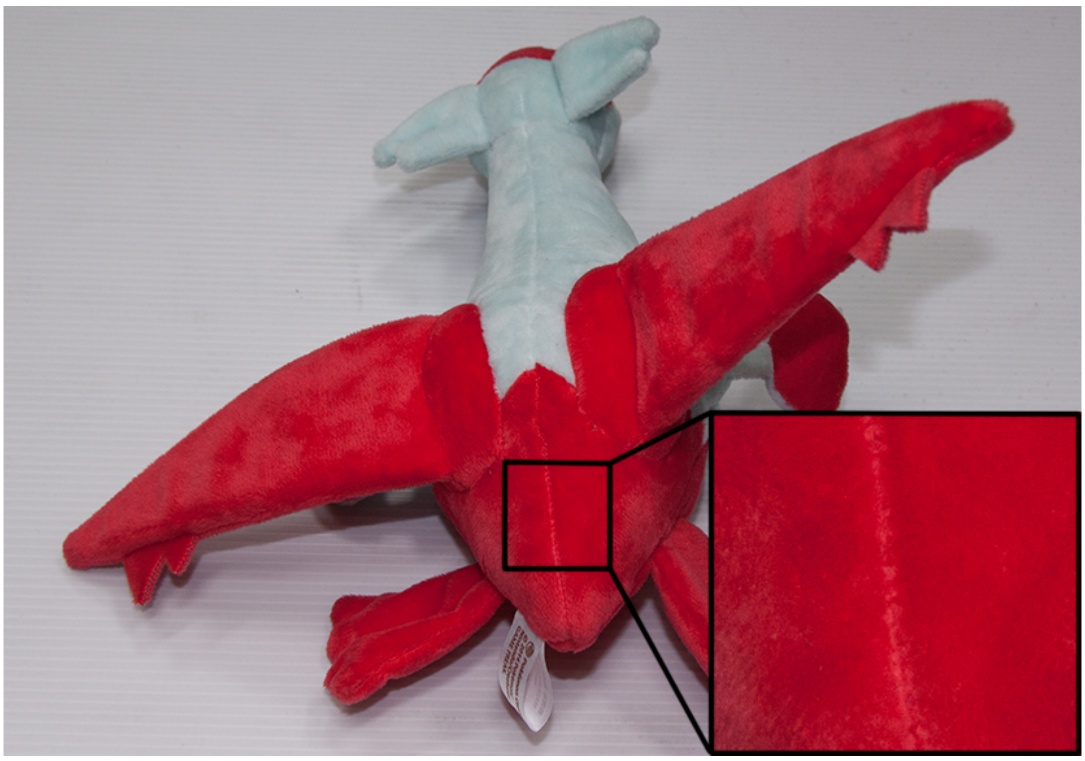


Figure 1.6: Trenches formed by seams on a stuffed toy.

Though these two techniques solve the problem of converting 3D shape into 2D pattern, they still requires user to determine how draw the cut lines, *i.e., seams* on the target shape. Care must be taken about of location of seams because the resulting patches cut by these seams may not be developable *i.e., flattenable*, which is the essential requirement for pattern design. Determine developability is quite a tricky task for untrained people due to the shape difference between 3D and 2D space. To solve this problem, some methods [4, 6] have been proposed to automatically generate developable patches that can be used for fabrication. Though these methods can successfully generate developable patches, the boundaries (*i.e., seams*) of these patches are usually not desired for fabrication due to their impact on appearance of fabricated result (Figure 2.5). When fabricating things, one must combine (*i.e.,* sewing, gluing) the adjacent boundaries and producing visually noticeable seams on the fabricated result (Figure 1.6). Well-designed seams should capture the salient structure of the target shape which conform with human perception so that it would not give negative impressions at first glance. On contrary, non-structural seams like the result of D-Charts (Figure 2.5) resembles scars on skins and degrades the appearance of fabrication result.

The salient structure, however, is difficult to be described by simple formulas. In this work, we take the strategy of *over-segmenting the target shape to obtain a candidate set of the final seams.* These candidates are then split into smaller seam segments and evaluated of their importance with respect to appearance. Finally, less important seam segments are removed by a greedy method similar with [7].

Chapter 2

Related works

In this section, we briefly review the most related research areas of the proposed method, including *fabrication-oriented design*, *mesh segmentation* and *surface flattening*.

2.1 Fabrication-oriented design

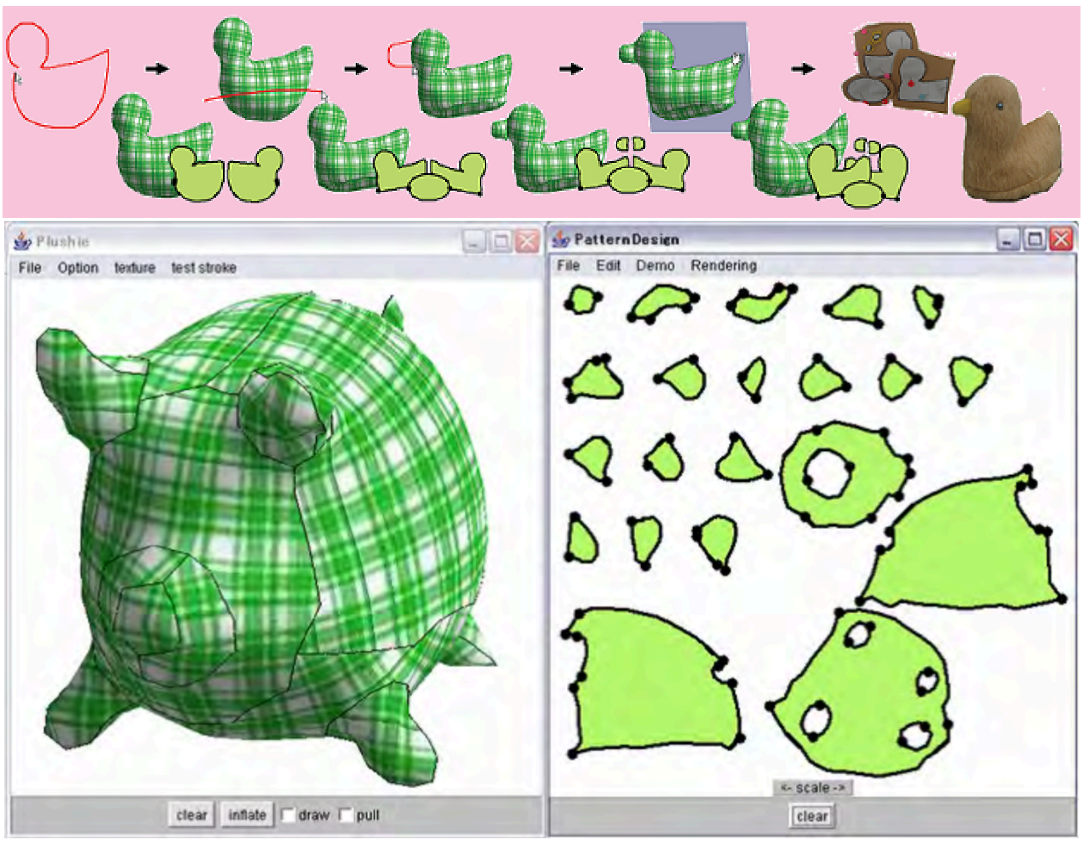
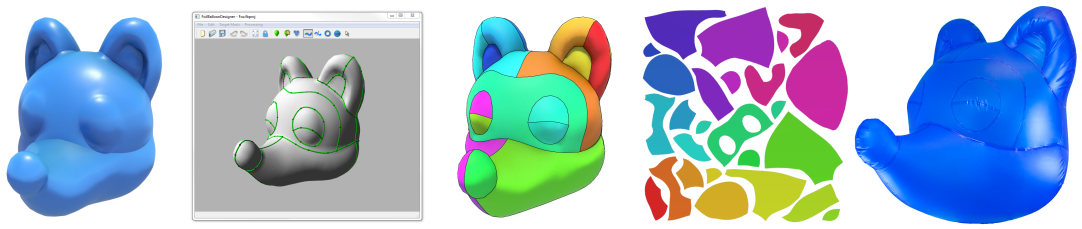
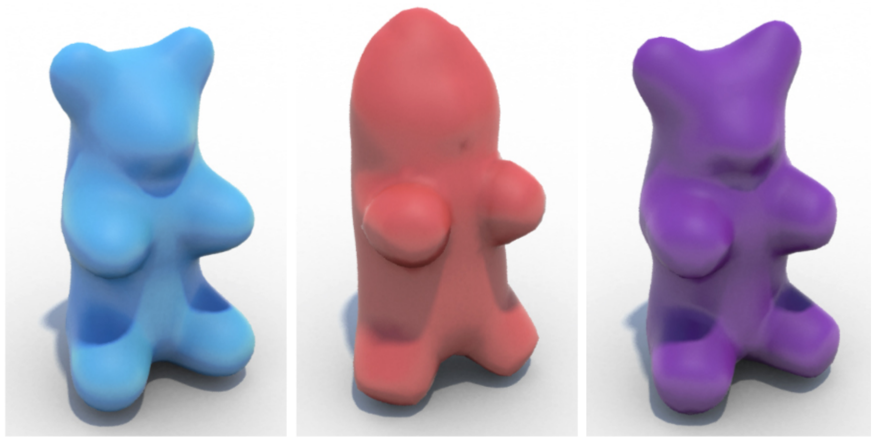


Figure 2.1: The workflow of Plushie [1] and its proposed UI

The traditional process in fabrication is accomplished on a trial-and-error basis. The loop of designing a prototype and validating its feasibility for physical fabrication is time-consuming and error-prone. To alleviate the efforts of designers, many well-known computer-aided-design tools [8–10] have been developed to facilitate model design and enable fast-prototyping. However, such tools are designed for expert users with domain knowledge and are not suitable for everyone. For plush fabrication, Mori *et al.* [1] proposed an interactive user interface combining modeling and pattern generation. In their work, user are asked to draw strokes to describe the shape they want. During the process, the 2D and 3D shape are synthesized simultaneously (Figure 2.1(b)). Users can quickly preview the result of their design and modify them if needed. Though their system is intuitive, the final result is greatly affected by the strokes given by user. Complex shapes that are not easy to be captured by strokes are not feasible for their system.



(a) The Workflow



(b) The input (c) Simulation result (d) Optimized result

Figure 2.2: Designing Inflatable Structures [2]

Mélina Skouras *et al.* proposed another interactive user interface for designing inflatables [2]. Unlike [1], they focusing on the correctness of the generated pattern. For some materials, it's not guaranteed that the patterns generated by target shape (Figure 2.2(b)) will give a physically correct result (Figure 2.2(c)). They utilize physically simulation to generate a physically correct result and then optimized it to resemble the target shape (Figure 2.2(d)) by modifying the generated patterns. However, since the seams have great influence on the generation of pattern, it's possible that patterns generated by inappropriate seams may not optimized to the desired shape. Though seams with respect to developability can be generated by algorithm such as D-Charts [4], for "aesthetic" reason, they still give a user interface for users to assign initial seams. The problem of determine seams for both appearance and developability remain unsolved.

2.2 Mesh Segmentation

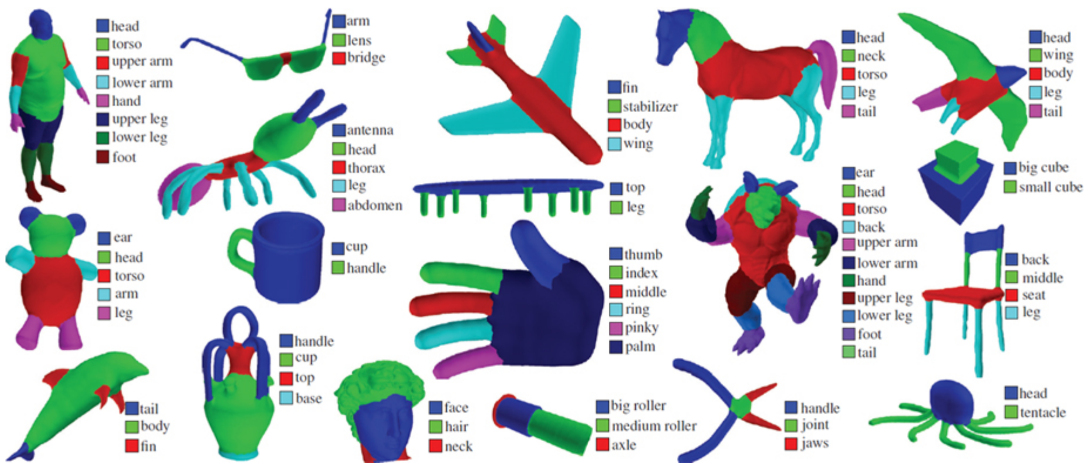


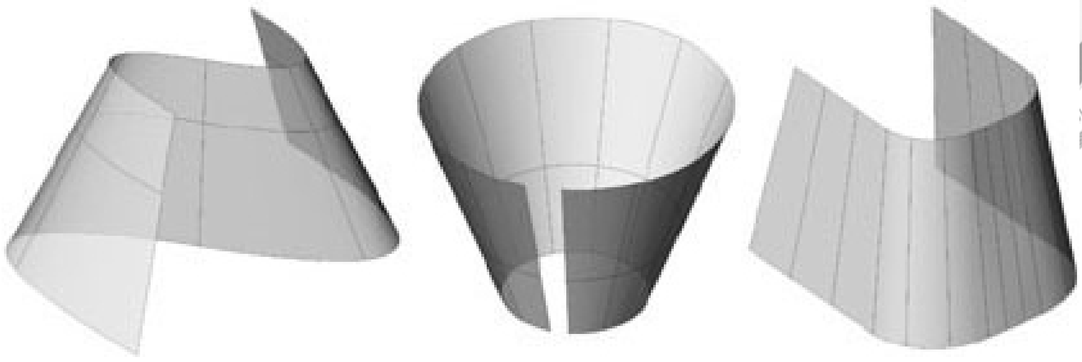
Figure 2.3: Examples of mesh segmentation [3]

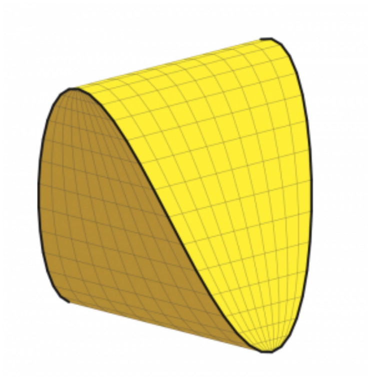
To generate planar pattern for fabrication, the target shape should be developable surface, *i.e.,* can be flattened into planar shape. This requirement is too strict since developable surface (Figure 2.4(a)) is only a small subset of all surfaces and the target shapes given by user are rarely developable surface. Fortunately this problem can be solved by split the target shape into piecewise developable surface (Figure 2.4(b)) using mesh segmentation.

Mesh segmentation (Figure 2.3) is a fundamental operation in geometric processing aiming to divide the original mesh into a collection of smaller sub-meshes or *patches*, which are coherent in terms of certain local properties, *e.g.,* flatness. A variety of segmen-

[9]http://complexitys.com/english/geometry/developable-surfaces/

[10]http://docmadhattan.fieldofscience.com/2014_06_01_archive.html


(a) Examples of developable surfaces[9]


(b) A piecewise developable surface with seam[10]

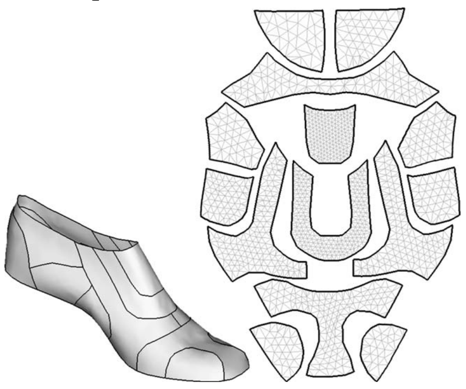

(c) A shoe and its corresponding piecewise developable surface [11]

Figure 2.4: Developable surfaces

tation techniques have been developed by taking advantage of different criterion functions, such as k-means [12], core extraction [13], fitting primitives [14], shape diameter function [15], machine learning [3].

Among these methods, we are particular interested in segmentation algorithms producing developable patches [4, 6]. These two works generate developable surfaces by approximating the target shape with conic surfaces. Julius *et al.* [4] aim to generate quasi-developable surface which is less strict than developable surface. They use a concept of "unions of uni-axial conics" to find the possible quasi-developable surface. Because the generated patch does not need to be fully developable, they can cluster each face into different patch without creating new meshes. Their work works well with materials with flexibility such as fabrics.

However, it's not the same to work with materials without flexibility such as paper. Shatz proposed another work *et al.* [6] generating fully developable surface for papercraft.

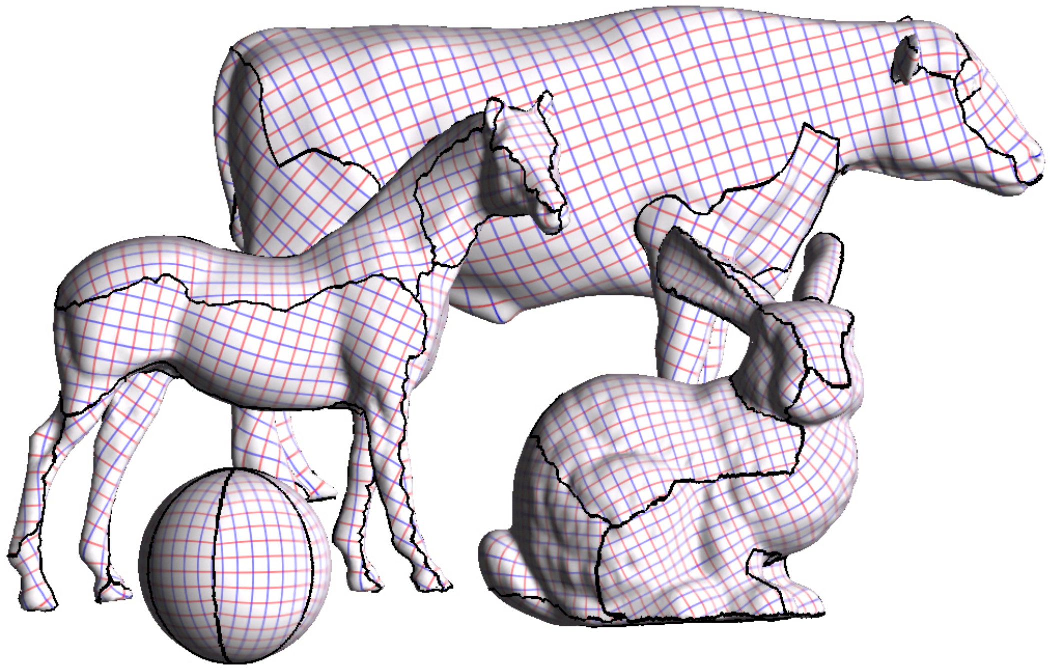


Figure 2.5: The result of D-Charts [4]

They use similar framework like [4] but solve additional problems as boundary inconsistency. Compared with [4], the number of generated patches (Figure 2.7(b)) is much more than the one of D-Charts (Figure 2.7(c)) due to the stricter requirement of developability.

Though these two works successfully segment the input mesh into quasi-developable/ developable charts, they do not take visual defects caused by seams into account. They focused on patches rather than boundaries, the resulting seams (boundaries) are merely the boundaries between each patch. These seams do not fit the salient structure of the given shape, resulting in a negative impression on the fabricated result shown in (Figure 2.5(b)).

In this work, we aim to generate seams with least visual defects by considering not only developability but also salient structure which has great influence on visual perception.

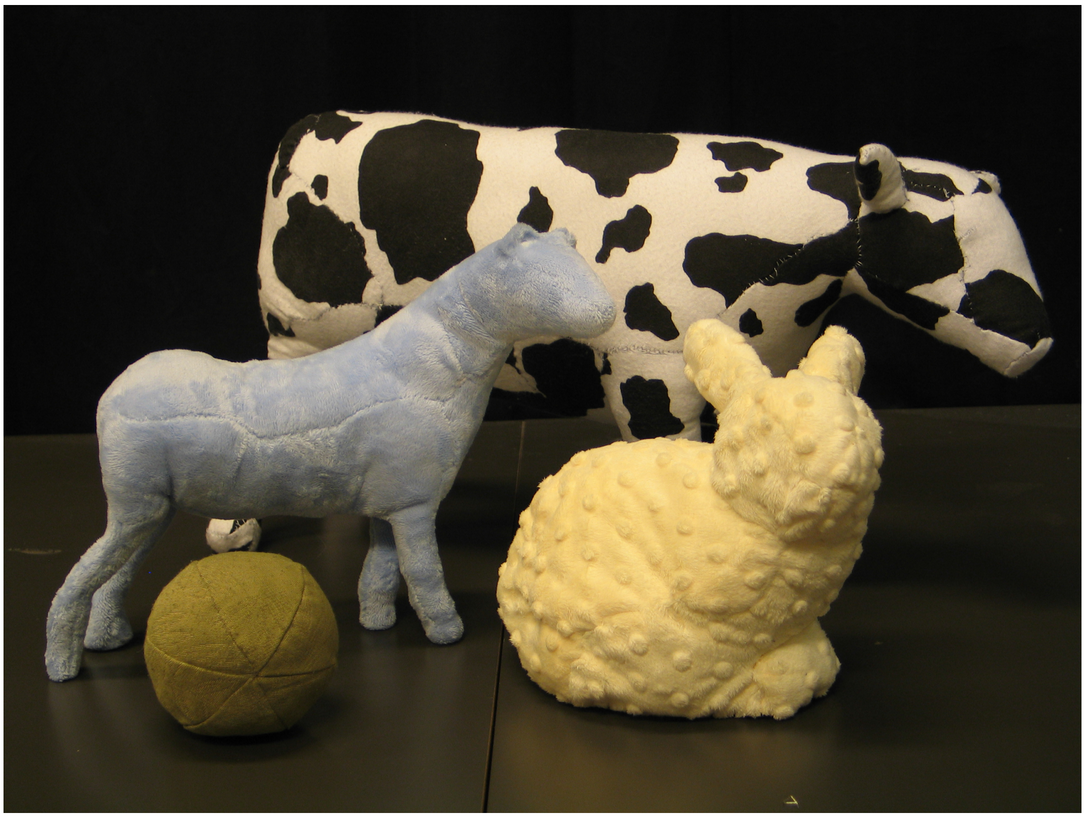
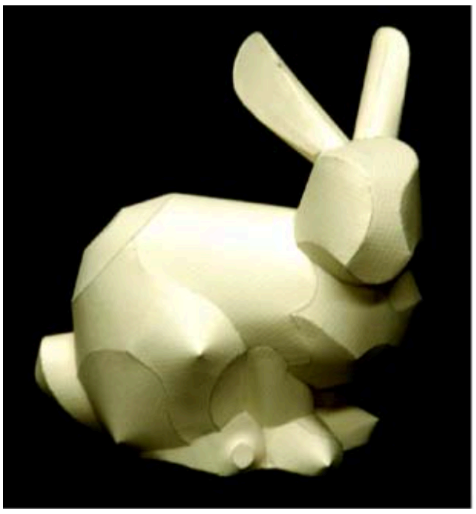
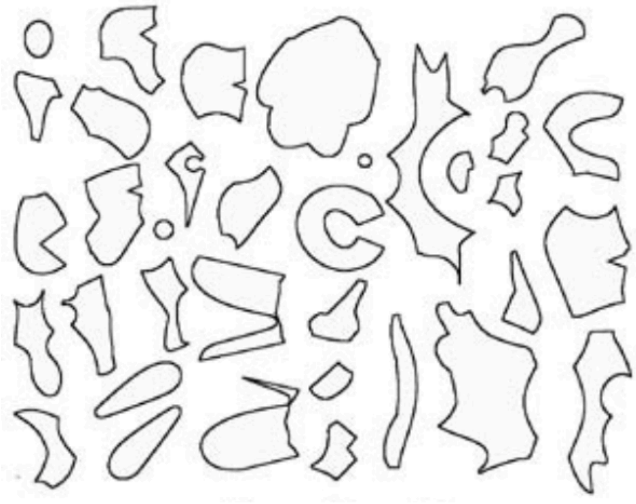


Figure 2.6: The fabricated plush using D-Charts


(a) The papercraft of Stanford bunny [6]


(b) The pattern of papercraft [6]

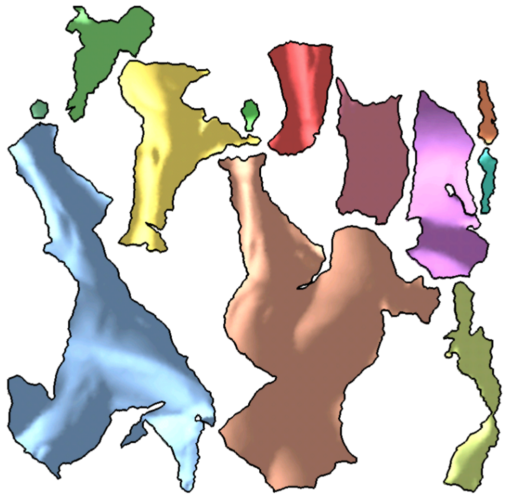

(c) The pattern of D-Charts [4]

Figure 2.7: Patterns generate for different application.

2.3 Surface Flattening

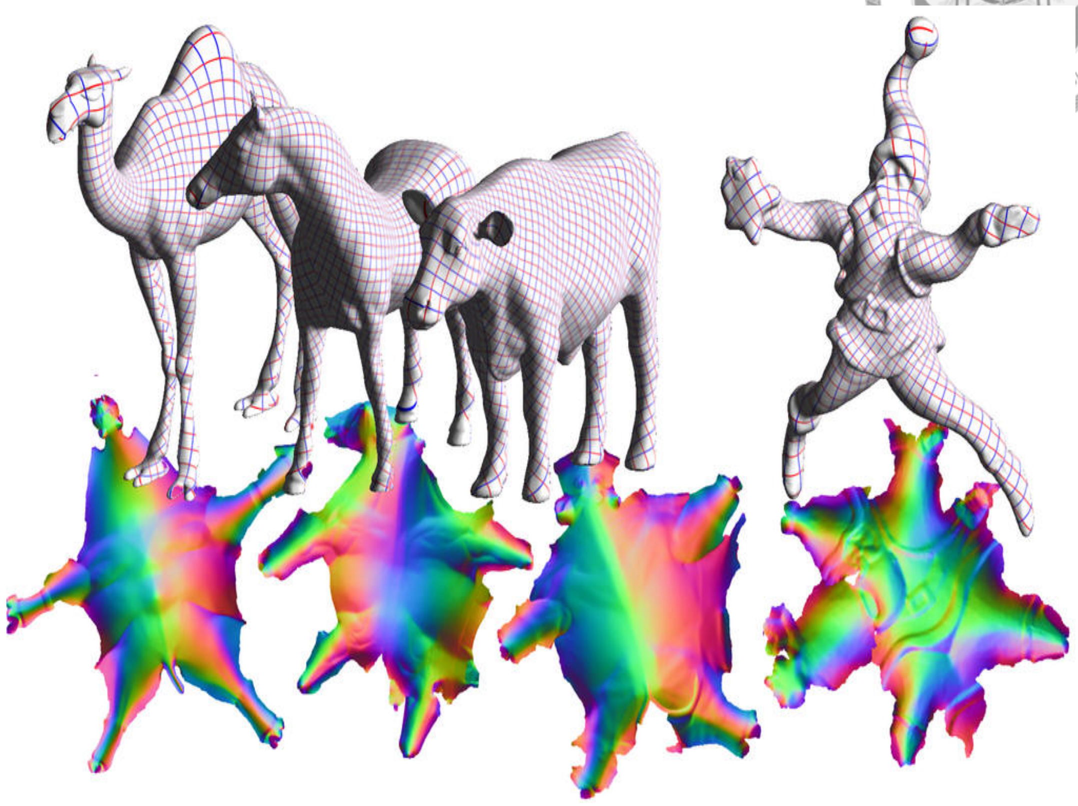
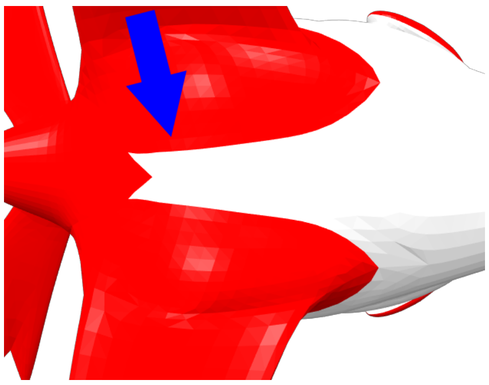
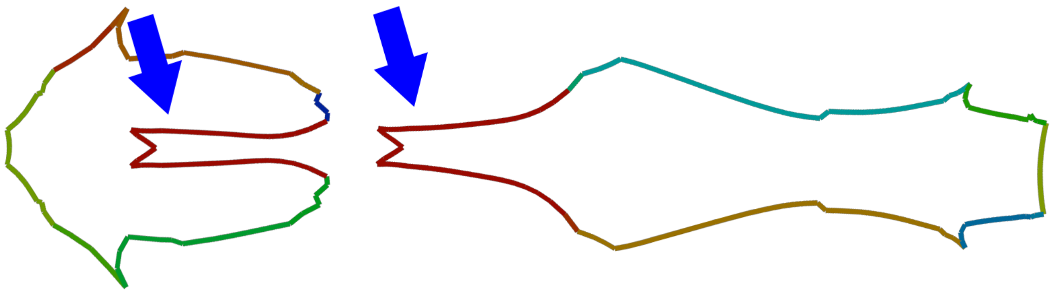


Figure 2.8: Examples of surface flattening [5]

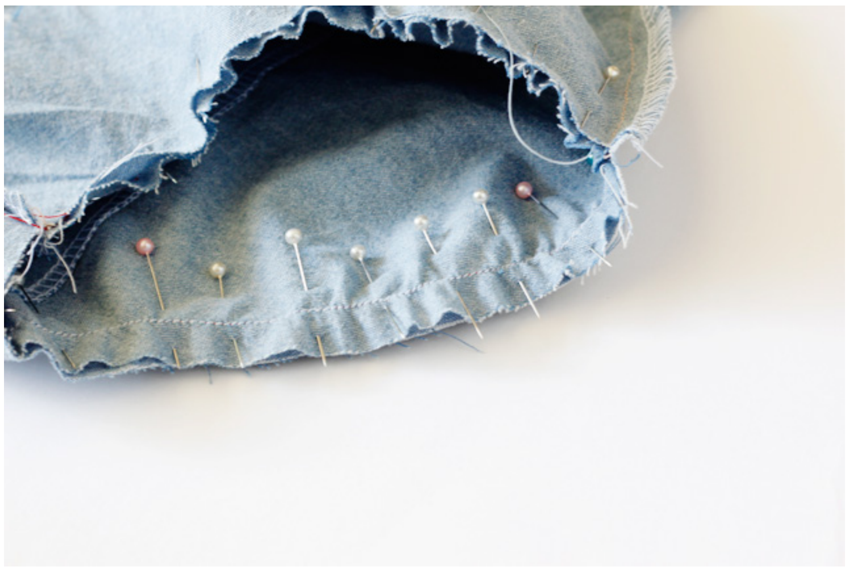
Surface Flattening is the final stage before generating patterns for fabrication. Many materials, such as fabric, leather and paper, are usually available as planar sheets, making it impossible to fabricate by ways such as casting. One have to print the designed patterns on these planar materials and then assembled them back to the target shape. To flatten a surface from 3D to 2D space, the typical solution is to perform mesh parameterization [5, 16–18], which is also widely used for texture mapping. Texture mapping computes a $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ mapping for each mesh vertex. It allows us to use the 2D texture coordinates as the flattened version of the original mesh. However, traditional texture mapping algorithms usually does not take physical constraints into account, which may result in infeasible results for physical fabrication. One essential consideration is to generate patches of the same seam length. For example, the two seams pointed by the arrows in Figure 2.9(b) are actually separated from the same position pointed by the arrow in Figure 2.9(a). During fabrication, the two seams in Figure 2.9(b) should be perfectly combined together, and therefore should have identical lengths. If the lengths differ, puckers like Figure 2.9(c) will appear on the side with longer seam since the additional fabrics are forced to be squeezed in a smaller space. Since the unexpected puckers are often seen during fabrication pattern design, to avoid this problem, we compute *length-preseving free boundaries* (LPFB) proposed by Wang [7] to flatten the sub-meshes generated by our algorithm.



(a) The original model



(b) The flattened patterns



(c) Example of puckers

Figure 2.9: Problems caused by boundary length

Chapter 3

Seam types

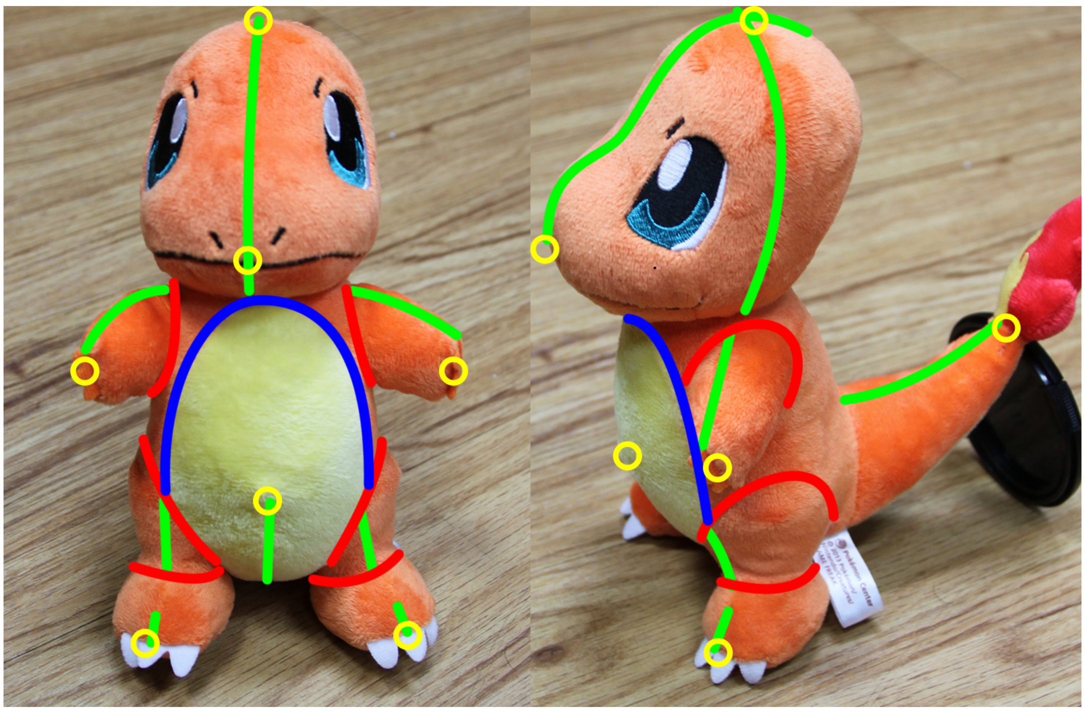


Figure 3.1: Red lines are segmentation seams. Green lines are structural seams and blue lines are texture seams. Yellow circles are extreme points.

Before introducing how to generate seams, we should first define what seams we need to generate. By observing a variety of plushes designed by experienced designer, we defined three types of seams as shown in Figure 3.1. They are *texture seams*, *segmentation seams* and *structural seams*.

3.1 texture seams

The *blue lines* in Figure 3.1 are the boundaries of different textures or colors. In plush making, we prefer using different fabrics for different color regions. The intersection lines between different fabrics will naturally become the seams. Texture seams can be considered as the hard constraints of seams.

3.2 segmentation seams

The *red lines* in Figure 3.1 are the boundaries between different body parts such as limbs and torso. Since it's the intersection region of different parts of body, there is a good chance that it will form a non-devlopable saddle surface. Thus, appropriate seams must be placed at such region to increase developability. Due to the properties mentioned above, we found mesh segmentation algorithms such as [13] are very effective to capture this type of seams.

3.3 structural seams

For the rest of the seams (*green lines*) that are necessary for plush making, we call them structural seams because they represent the geometric structure of the target shape. Since there are already clear answers of other two types of seams, we focus on this type of seams in this research.

This type of seams usually pass through high curvature areas such as extreme points showed as yellow circle in Figure 3.1. This can be explained as to solve developability problem near high curvature areas since they are non-developable. Other geometric properties such as skeleton direction or smoothness can be used to give a "natural" look for human perception. The details are described in Chapter 5.

Chapter 4

Overview

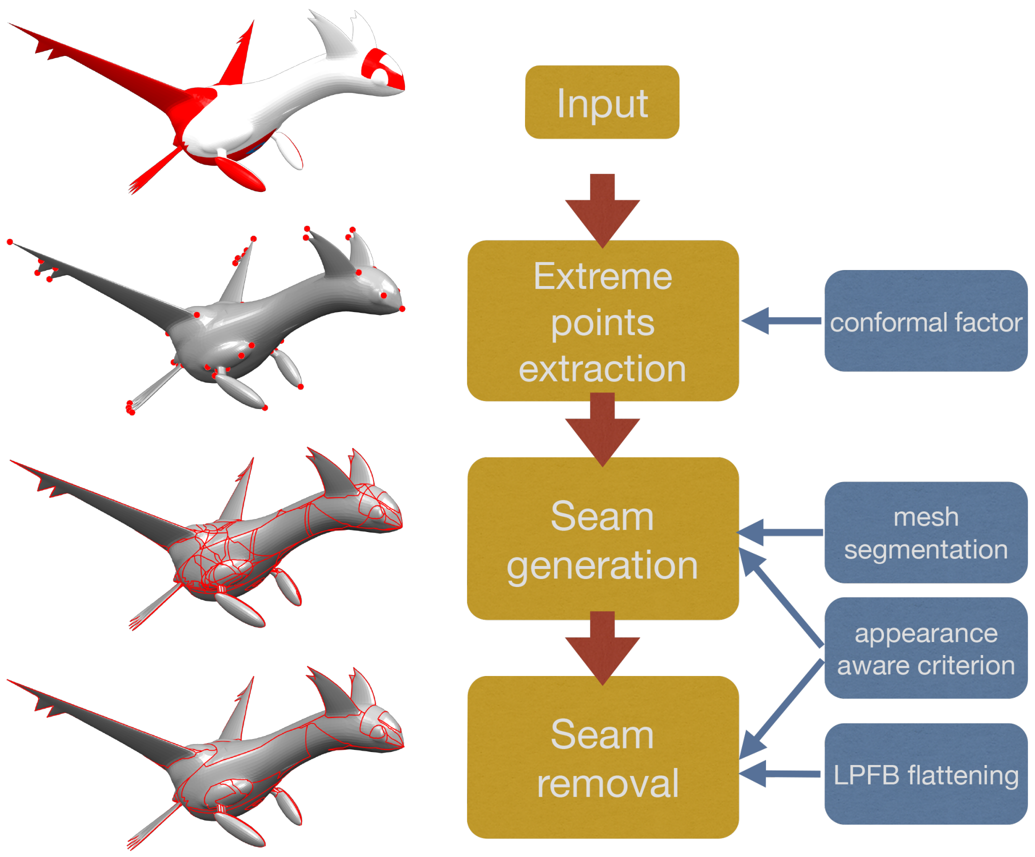


Figure 4.1: The system workflow.

In the following sections, we treat the input mesh as a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with its vertices and edges.

The proposed algorithm consists of three main stages, including *extreme points extraction*, *seam generation* and *seam removal*.

Starting from extreme points, which appears at high curvature location, it's natural to put seams on these place to increase developability. This observation can be found on commercial plush showed in Figure 3.1. To find extreme points, we compute conformal scaling factor [19] on $\mathcal{G}$ and treat local extrema points as $\mathbf{V}$ (Figure 5.1). Conformal scaling factor has the advantage of avoiding noise from local curvature (Figure 5.2(a)) by considering global curvature (Figure 5.2(b)).

For seam generation, due to the uncertainty of how to choose the right seams, the basic idea is to *generate all possible seams* and then filter out the unnecessary ones. We first calculate seam candidates $\mathbf{S}$ (Figure 5.10), which is the unions of the three types of seams mentioned previously. Structural seams are generated by applying all-pairs shortest path algorithm on $\mathcal{G}$ among $\mathbf{V}$ with the weight $\mathbf{W}$ defined in Chapter 5. Segmentation seams are generated by applying existing algorithm [13] and texture seams are just the boundary of different texture.

Though each of the seam candidates $\mathbf{S}$ generated above is the best candidate itself, it may contains segment that are not so good when comparing with other seam candidates. These segments should be removed since they will affect the final appearance. To analyze them, seam candidates $\mathbf{S}$ are broken down into seam segments $\mathbf{S}'$ (Figure 5.12(b)) using their intersection points $\mathbf{V}'$ (Figure 5.12(a)). By re-evaluate each seam segments with $\mathbf{W}$, we can discover the bad part (Figure 5.11) in a seam candidate and remove them. During removal, one should bewared that removing seam segments may cause a sub-mesh becoming non-developable (Figure 5.13(c)). User can give a threshold $\mathbf{h}$ measuring area changes after a sub-mesh is flattened to ensure the developablility of each sub-mesh split by seams. If removing a seam will cause the area change becomes greater than $\mathbf{h}$, this seam should not be removed.

Other than developability, one should also note the order of removal will effect the final result. For example, there are two possible seam segment to be removed in Figure 5.14(left). When one seam segment is removed (Figure 5.14(middle)), the other one can not be removed otherwise the mesh will become non-developable. As the result, the flattened patterns will eventually become different (Figure 5.14(right)). We use a greedy method like [15] to remove seam segment in the order of least important first. By doing so, we can reduce the chance of unable to remove undesired seam segment due to the requirement developability. The importance of seam segment can evaluated with **W** since each seam segment is a subset of the original seams.

Chapter 5

Proposed Method

The proposed algorithm consists of three main stages, including extreme points extraction, seam generation and seam removal.

For the three types of seams mentioned previously, the first two types are trivial. Texture seams are just the boundary of different texture. Segmentation seams can be generated by applying existing algorithm [13]. Now the only problem is the structural seams.

5.1 Extreme points extraction

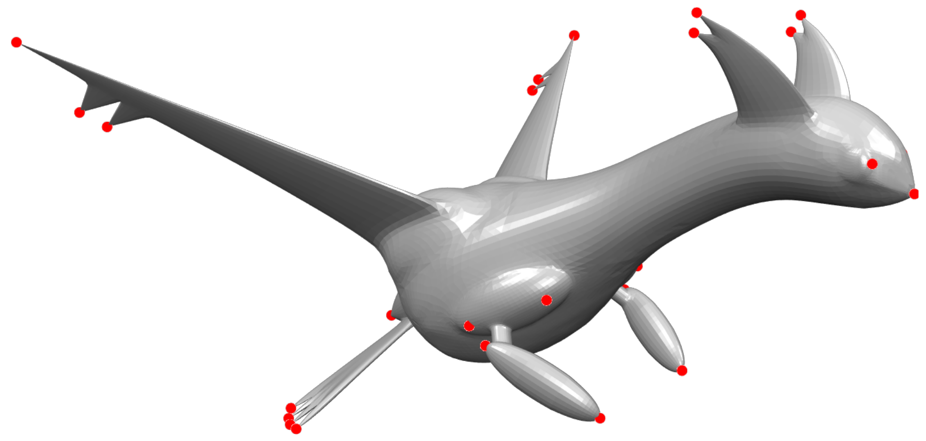


Figure 5.1: The extreme points $\mathbf{V}$

From Figure 3.1 we know that some structural seams should pass through non-developable regions such as extreme points (yellow circles) to solve developability problem. It's a wise idea to extract these points so that we can guide the seams to the correct location. Extreme

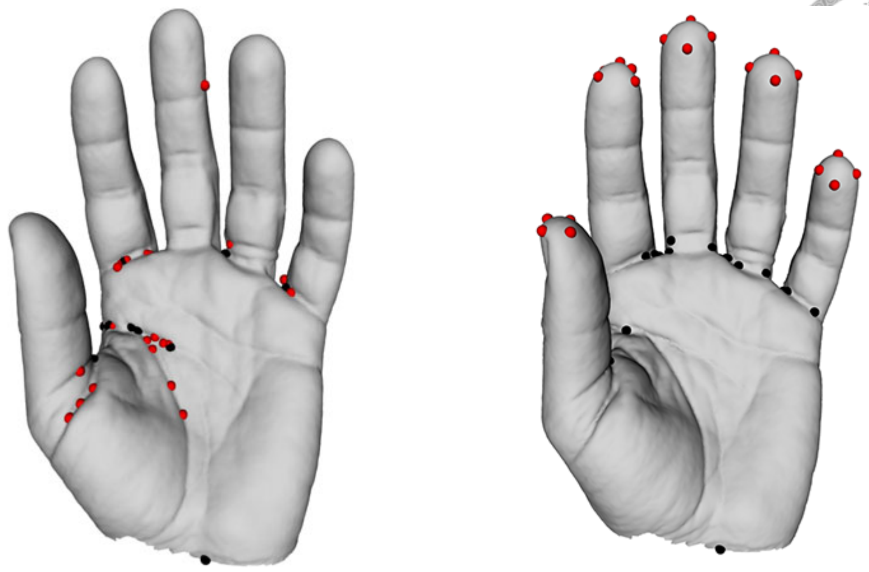

(a) Highest absolute curvature (b) Conformal scaling factor [19]

Figure 5.2: Extreme points extract by different method

points can be extract using Gaussian curvature directly, though it may not be reliable for complex shapes (Figure 5.2(a)). A better way is to use conformal scaling factor [19] which considering not local but global curvature information (Figure 5.2(b)). In general case, only a small number of points is needed for $\mathbf{V}$. We select at most $\sqrt{|\mathcal{V}|}$ points from $\mathcal{V}$ whose conformal factors form local extrema among their 2-rings neighbors as $\mathbf{V}$.

5.2 Seam Generation

Our goal is to generate seams that fit the nature feel of shape when people look at it. Inspired by [20], we formulate this problem as a shortest path problem with the extreme points in the last step. Unlike [20], we do not calculate Steiner tree because tree do not contains loops that may help splitting mesh into smaller developable patches. However, it's unclear of how to generate appropriate seams with loop, we thus seek the idea of *over-segmentation*. That is, first enumerate all possible candidates $\mathbf{S}$ and then discard the less likely ones.

To make sure the seams conform with the natural structure of the given shape, we propose several geometric criterion as below. These criterion measures how well the seam is close to the principle used by experienced designers. The result seams are generated by solving all pair shortest path problem with the criterion.

5.2.1 Geodesic distance:

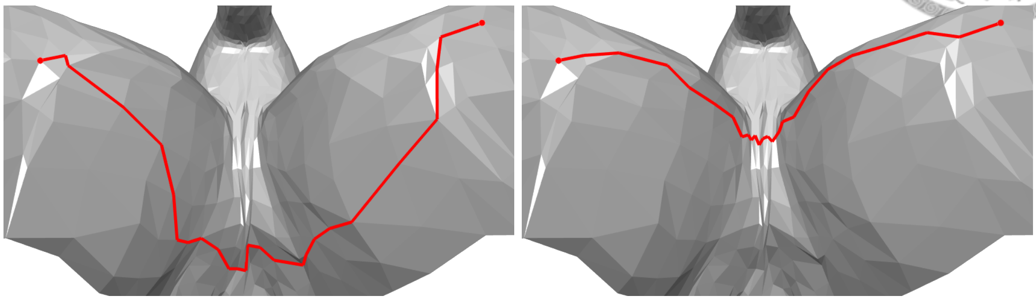

(a) Without geodesic distance (b) With geodesic distance

Figure 5.3: The effect of geodesic distance on seams

A simple yet elegant line along mesh surface without abrupt turning is preferred for seams. Such property can be achieved with the help geodesic distance (Figure 5.3. Short seams also have the advantage of reducing the time of fabrication. Edge weight of this property is defined as its length divided by the maximum edge length.

$$W_g(e) = \frac{\|\vec{e}\|}{l_{max}}, \tag{5.1}$$

$$l_{max} = \max_{e \in \mathcal{E}} \|\vec{e}\| \, . \tag{5.2}$$

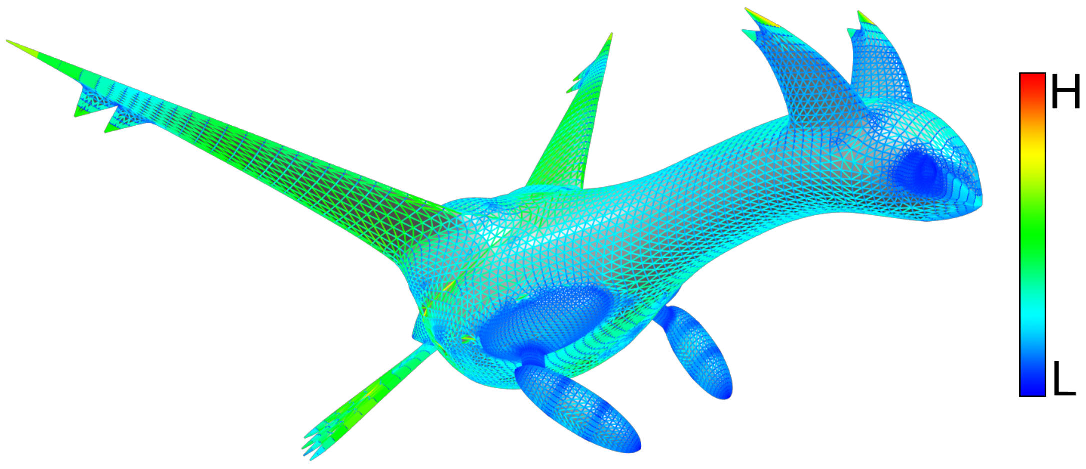


Figure 5.4: Visualization of geodesic distance weight.

5.2.2 Skeleton direction

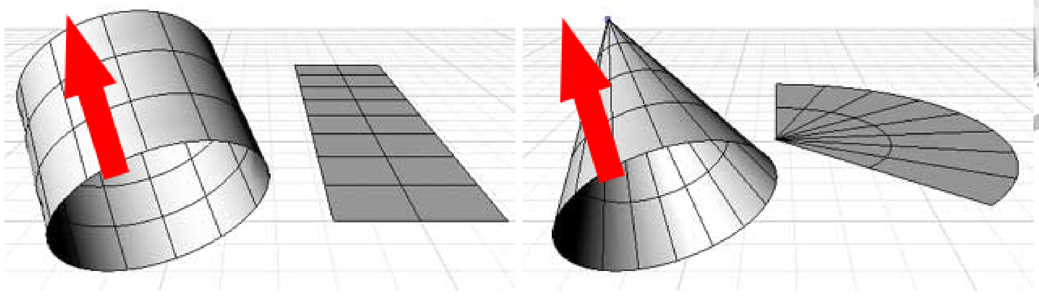


Figure 5.5: Cylinders and cones are well-known developable surfaces. The arrows are the optimal seam directions.[13]

In many case, limbs, horns and other parts are close to cylinders or cones. These two shapes are known to be developable surfaces and have an optimal seam parallel to axial direction shown in Figure 5.5. Laying seams parallel to the axial direction not only gives a nature look but also reduces the seams required for flattening.

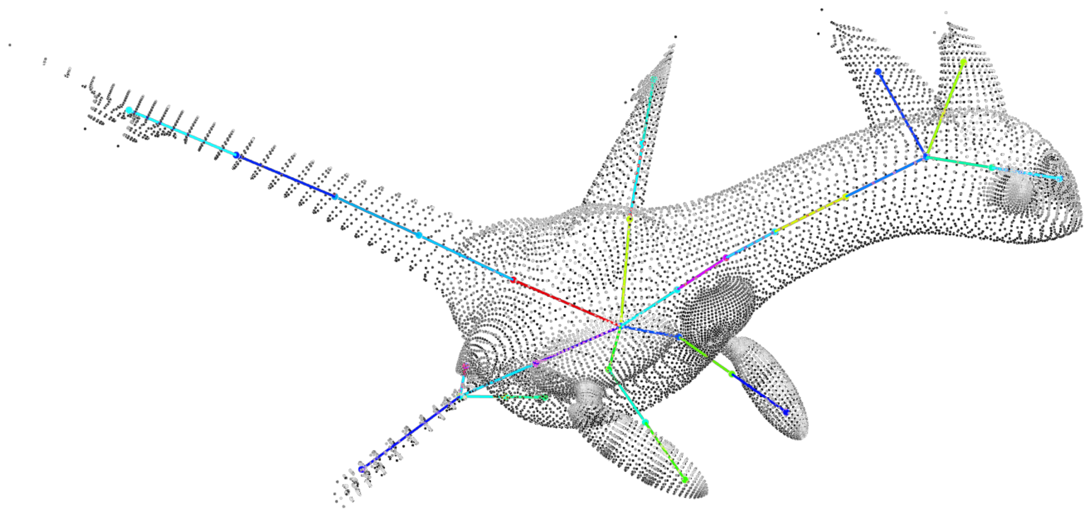


Figure 5.6: The extracted skeleton. Each bone is labeled with different color.

Since axial direction only exist for true cylinders and cones, we use bone direction for approximation for general shapes. With the help of automatic algorithm such as [21], the skeleton of mesh can be easily generated (Figure 5.6).

The following skeleton weight is defined to minimize the direction difference between

[13]http://www.rhino3.de/design/modeling/developable/

bone direction and edge direction.

$$W_s(e) = 1 - \alpha(e) \cdot |\cos\theta|\,, \tag{5.3}$$

where θ is the angle between $\vec{e}$ and $\vec{b_{avg}}$ is described below. $\cos\theta$ measures the consistency between $\vec{e}$ and $\vec{b_{avg}}$. Given n bones, b_{avg} is the averaged bone direction.

$$\vec{b_{avg}}(e) = \sum_{i=1}^{n} \vec{b_i} \cdot w_{i,e}, \tag{5.4}$$

$$w_{i,e} = \frac{w_{i,a} + w_{i,b}}{2}. \tag{5.5}$$

For i-th bone, $\vec{b_i}$ is the direction and $w_{i,a}$, $w_{i,b}$ are the influence of this bone to the two vertices connected by e respectively. There are many ways of determining the influence factor, for example, the vertex weight used for skinning. As long as it represents the spatial relationship from bone to vertex, it's suitable in our application. Here we assume the weights of all bones to sum up to 1. In our implementation, we use the weights produced by [22].

In some cases like chest where many bones gathered together, the averaged direction is not reliable since there are no dominant directions. To prevent from getting incorrect result, we introduce an influential factor α to weaken the effects in such case. This factor measures the direction deviation of effective bones.

$$\alpha(e) = \sum_{i=1}^{n} |\cos\phi| \cdot w_{i,e} \tag{5.6}$$

ϕ is the angle between $\vec{b_i}$ and $\vec{b_{avg}}$.

The visualization of skeleton weight is shown in Figure 5.7. Since this is a shortest path problem, lower weight (blue) is better. Note the color of body is close to red. This is because there are no dominant bone direction near body and causing alpha to be low. For wings, some edges are in blue since it corresponds to the averaged bone direction nearby. The other edges on wings are perpendicular to the averaged bone direction so their colors are in red.

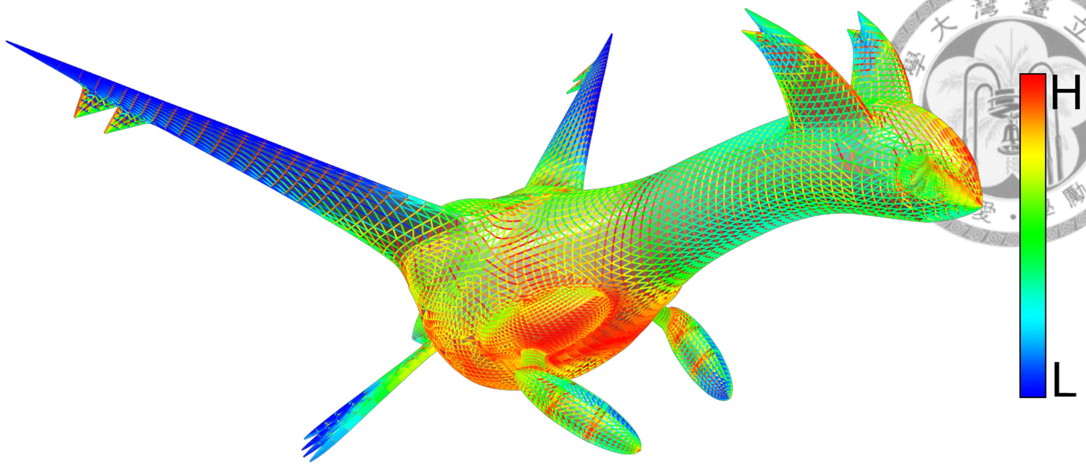


Figure 5.7: Visualization of skeleton weight.

5.2.3 Curvature

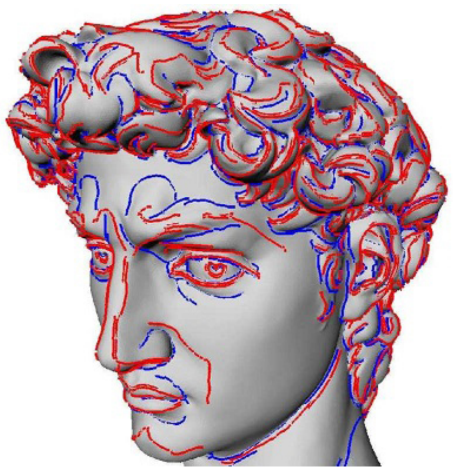


Figure 5.8: The crest lines can give visual hints on shapes.[14]

Curvature plays an essential role in both developability and appearance. By placing seams on high curvature region, the distortion of flattening can be redistributed along seams and greatly reduced. In the other hand, curvature riches a shape by features such as ridges and contours (Figure 5.8). Seams can be used as hints to tell people how this shapes is.

$$W_c(e) = \frac{1}{1 + |c_a| + |c_b|} \tag{5.7}$$

c_a and c_b are the mean curvature of the two vertice connected by e respectively.

[14]http://doc.cgal.org/latest/Ridges_3/index.html

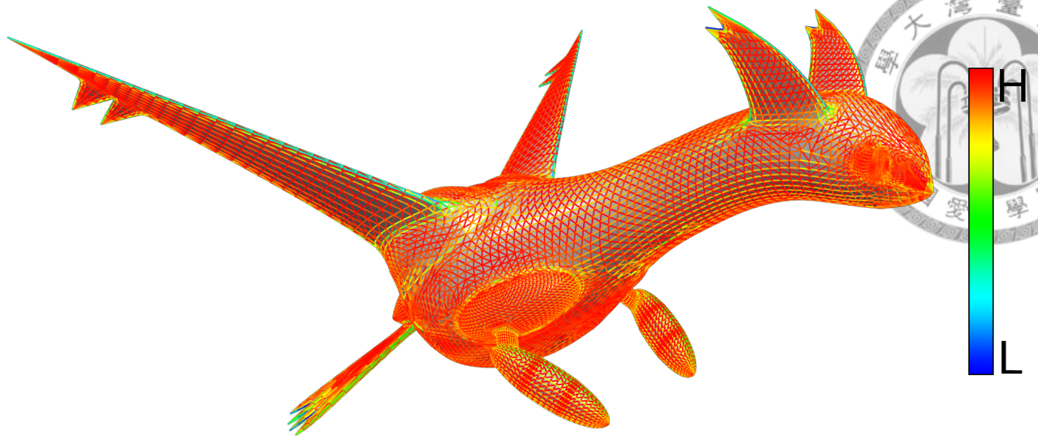


Figure 5.9: Visualization of curvature weight.

Finally the weight can be written as:

$$W = W_g^{\alpha} \cdot W_s^{\beta} \cdot W_c^{\gamma}. \tag{5.8}$$

α, β, γ are the parameters to control the effectiveness of each term. In our implementation, we use $\alpha = 0.3$, $\beta = 0.3$, $\gamma = 0.8$.

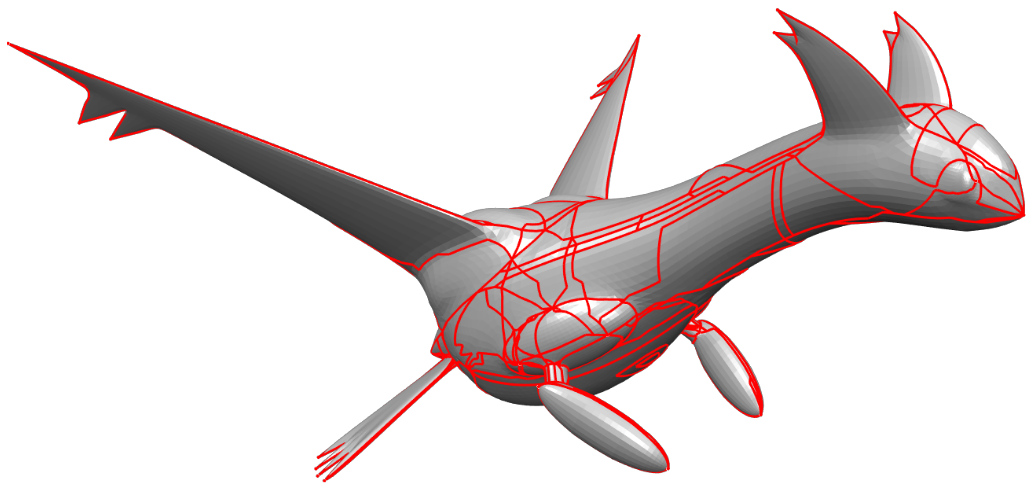


Figure 5.10: The seams generated in this step.

5.3 Seam Removal

Though each generated seam is the best candidate itself. When we give it a close look, it may contains both some good segments and bad segments. For example, in Figure 5.11, there is a path starting from black point to white point. The black-blue segment

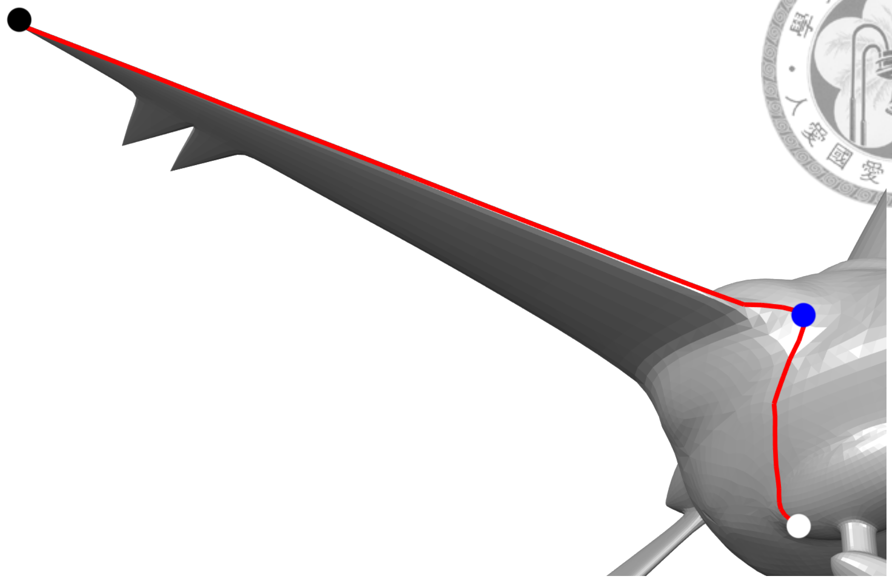
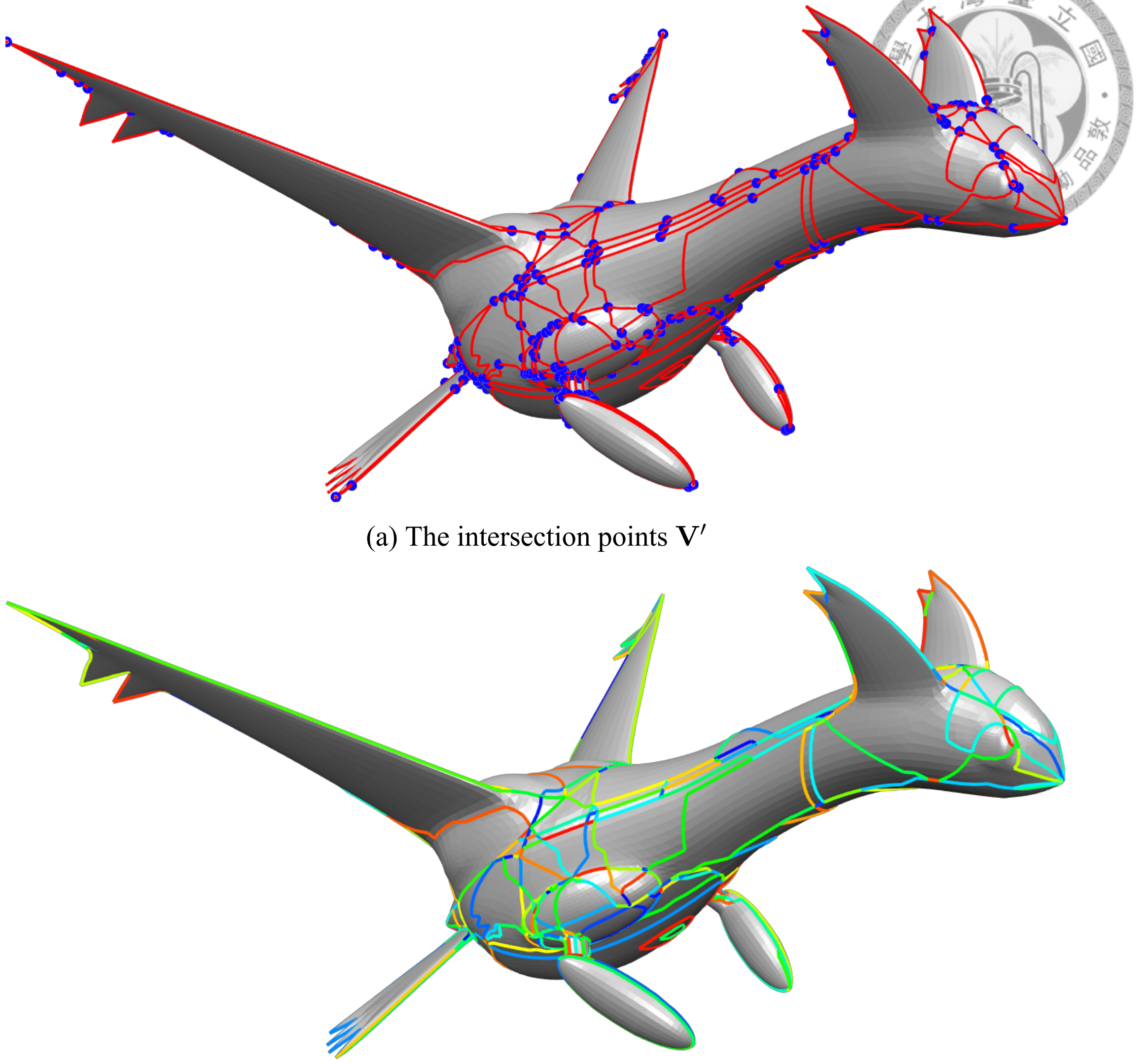


Figure 5.11: An example showing a seam may contains both good segment (black-blue line) which matches our goal and bad segment (blue-white line) which do not.

is considered good since it conform with the skeleton direction and curvature criteria. The blue-white segment, however, doesn't really conform to the criteria we defined. The reason why this segment still exist is because blue point has to connect to white point, the destination of shortest path. If we know the position of blue point, we can directly calculate the black-blue segment as seam without another extreme point (white point). Unfortunately, it's hard to find such blue points in advance since they do not have any characteristic. The alternative way we use here is to find the candidate seams first, then use the intersection points $\mathbf{V}'$ Figure 5.12(a) of seams as blue point. After intersection points are found, we can split seams into smaller seam segments $\mathbf{S}'$ Figure 5.12(b) and re-evaluate whether these segments are good or bad.

When removing seam segment, it important to keep developability of each sub-mesh split by seams. The process of seam segment removal may made sub-meshes nearby non-developable. For example, the mesh in Figure 5.13(a, left) is developale and Figure 5.13(a, right) is the flattened patterns. However, after some seam segments is removed, it becomes Figure 5.13(c, left) which is an open cuboid. Open cuboid is only developable if there is a seam cutting through the middle. For Figure 5.13(c, left), it's not developable and the flattening will fail (Figure 5.13(c, right)). We can use flattening algorithm such as [7] to


(a) The intersection points $\mathbf{V}'$

(b) The seam segments $\mathbf{S}'$

Figure 5.12: The intersection points $\mathbf{V}'$ of seam and seam segments $\mathbf{S}'$ generated by splitting seams with intersection points.

detect this case. Once detected, the last seam segment should not be removed and the system roll back to previous stage.

It's worth noting that the order of removal may leads to different result. The yellow and green line in Figure 5.14(left) are the possible removable seam segment. After one seam segment is removed (Figure 5.14(middle)), the other seam segment can not be removed. Otherwise the center point will become non boundary points and has non-zero Gaussian curvature, making the cuboid non-developable. As the result, these two set of seams will never generate the same result since yellow and green seam segments can not be removed at the same time. So it's better to choose the right order to remove those unwanted seam

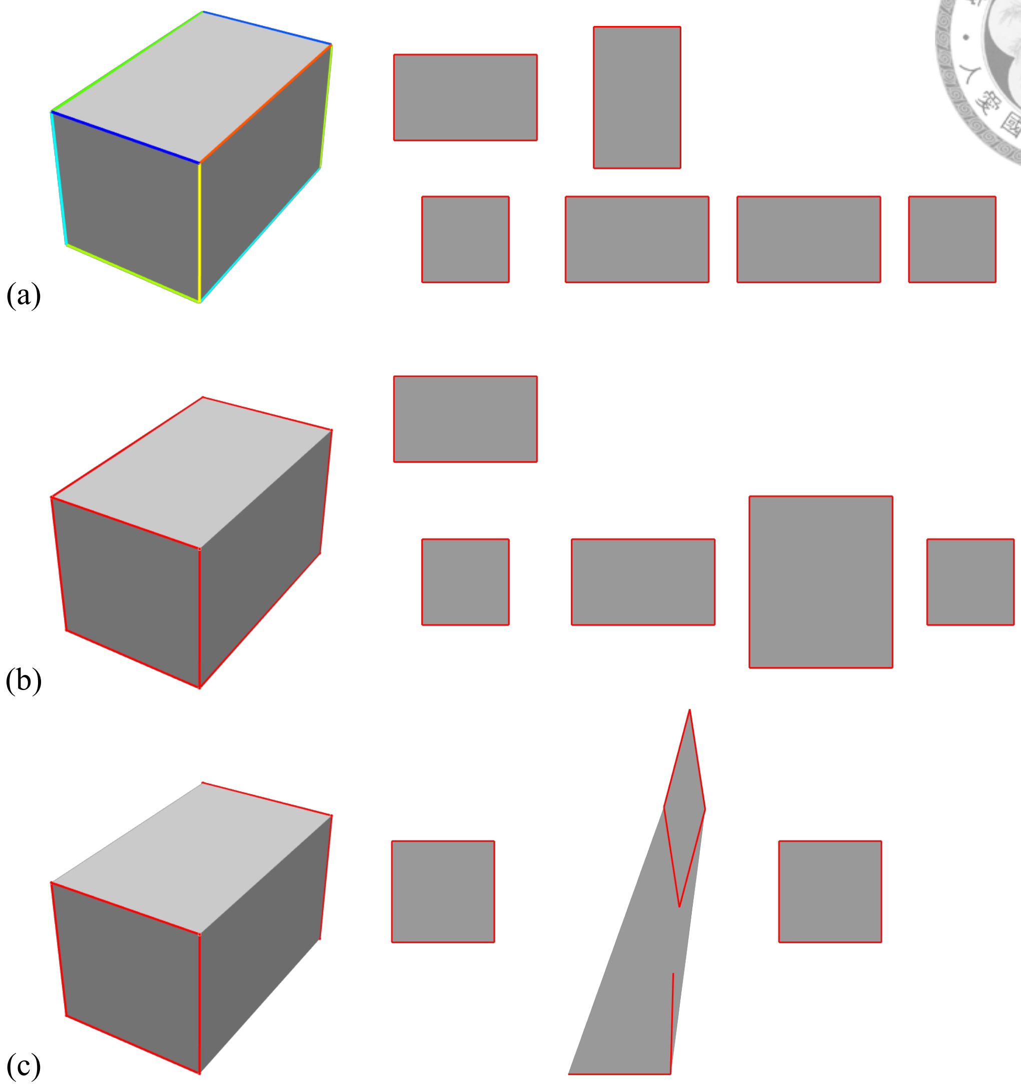


Figure 5.13: From left to right: target shape with seams, the corresponding flattened pattern. From upper to lower: (a) A cuboid with seam segments labeled in different colors (b) One seam segment is removed (c) Four seam segments are removed.

segment. Here we use a greedy method similar to [7] but with different priority to remove unnecessary seam segments.

It's an intuitive idea to start from the least important seam segment due to the decreasing developability. During seam removal, the sub-meshes split by seams will grow in size and become less developable. The chance a seam segment become not removable will also increase. In other words, the earlier candidates of seam removal have higher chance to be removable, and vice versa. Since we only want to preserve good seam segments, the bad seam segments should be removed first. This problem then become how to determine

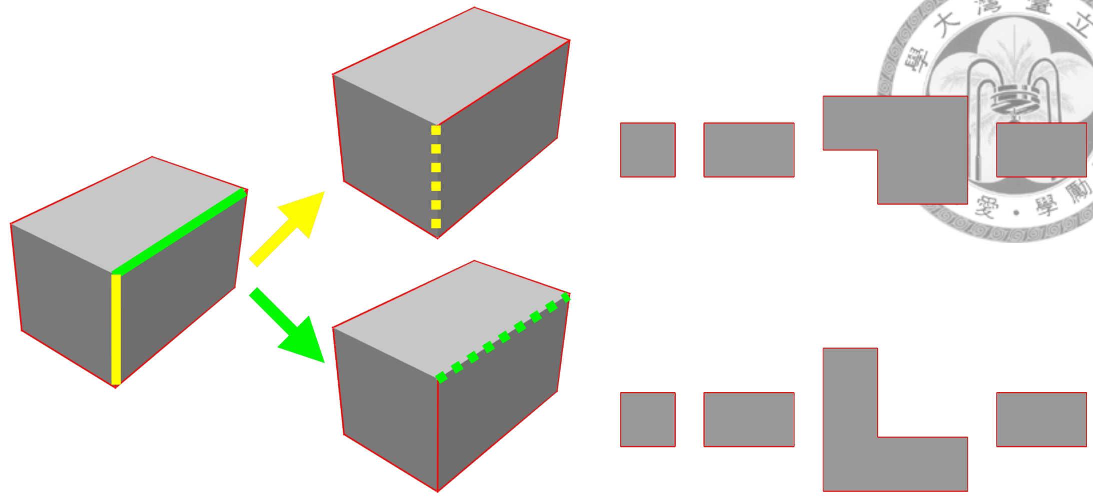


Figure 5.14: (Left) Yellow and green lines are removable seam segments (middle) After one seam segment is removed (showed in dots line) (right) The flattened patterns

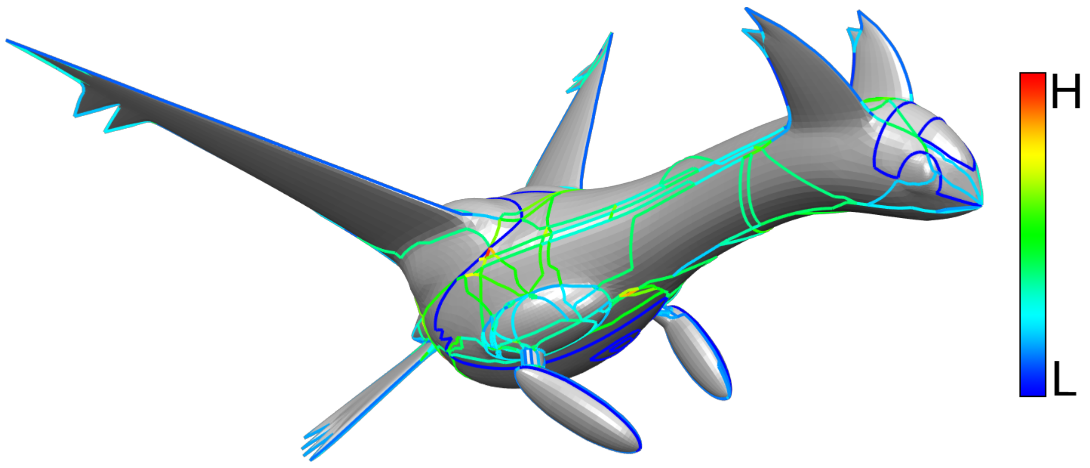
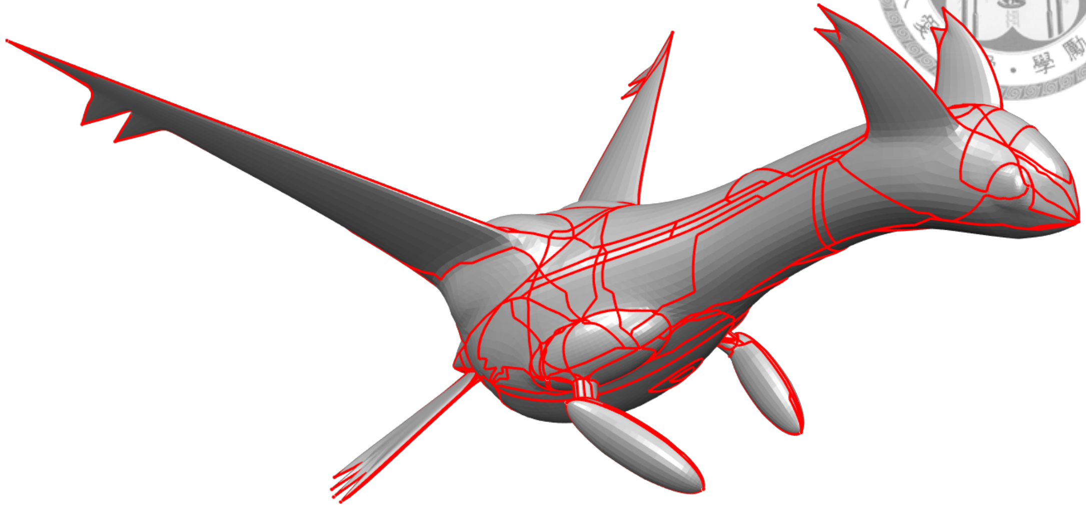
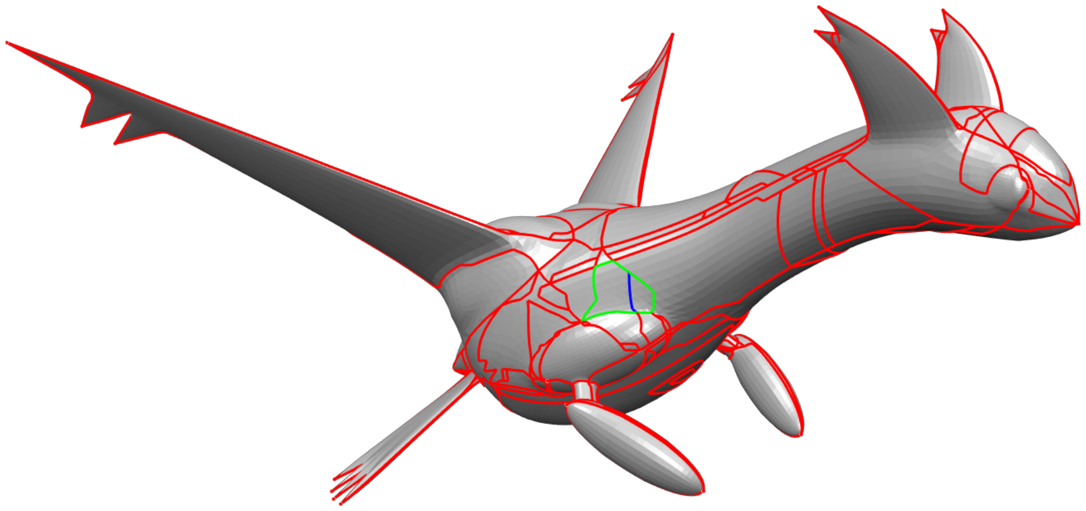


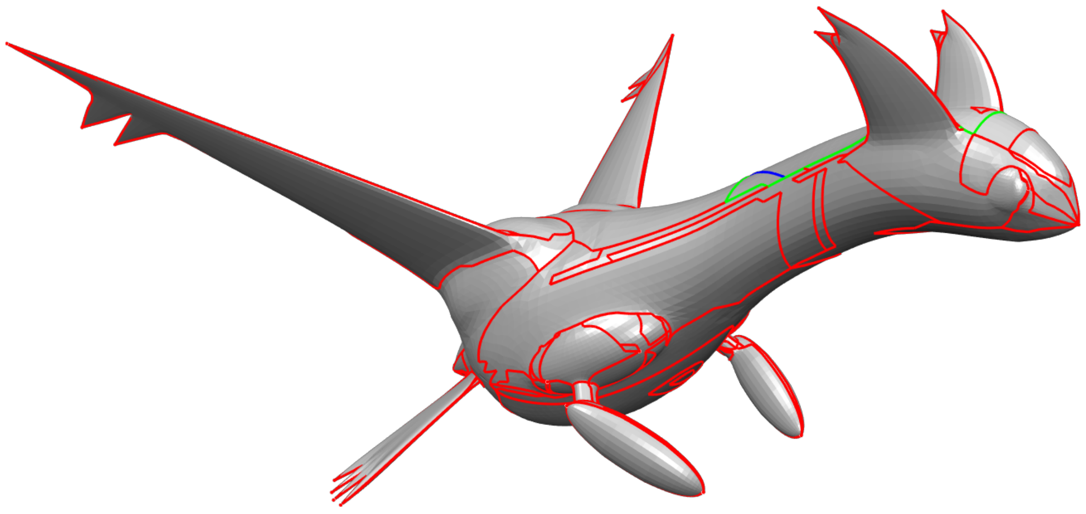
Figure 5.15: The visualization of importance of seam segments

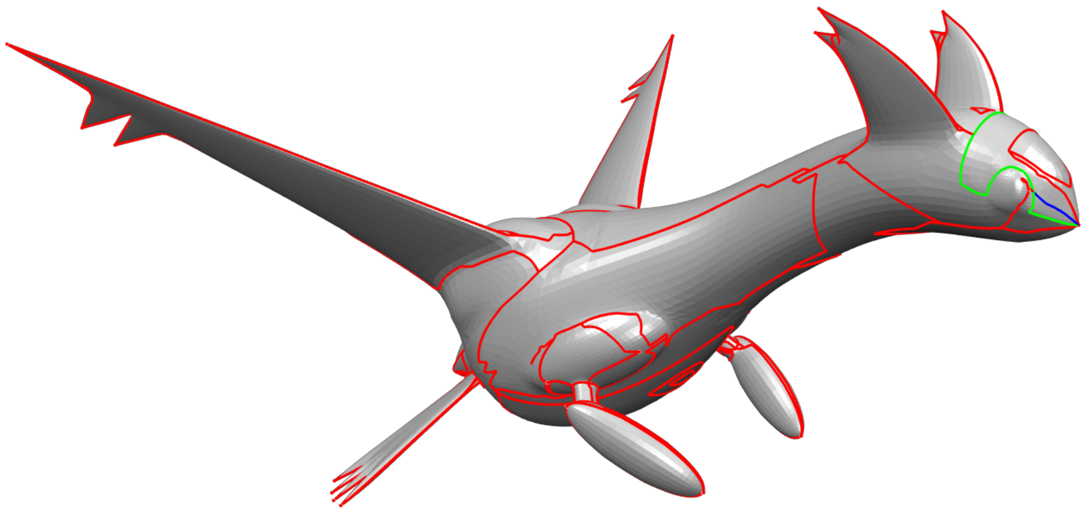
whether a seam segment is good or not. Fortunately we can reuse the metric **W** defined previously since each seam segment is a subset of the original seam. This process loops until no seam segments can be removed. Figure 5.16 illustrates the process of iterative seam removal.



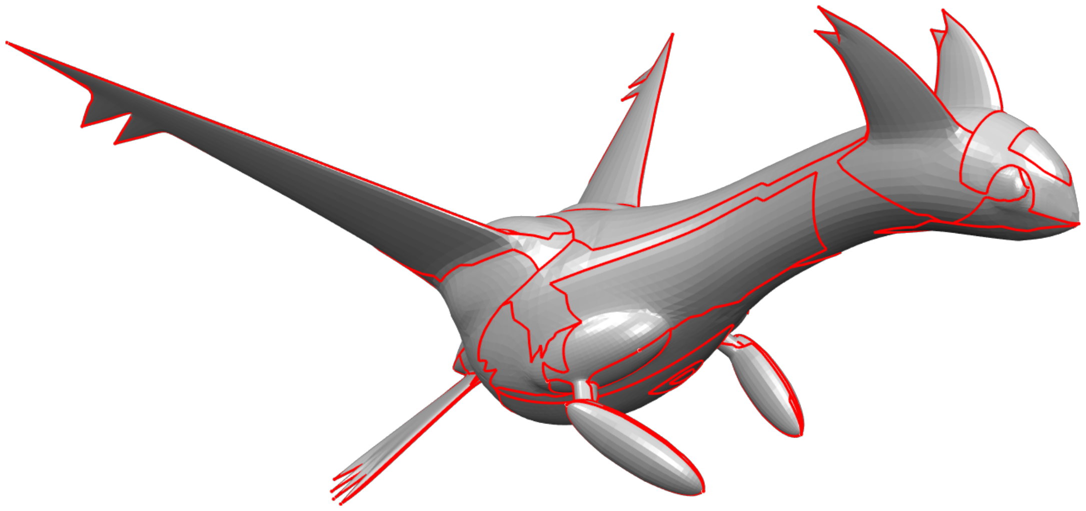
(a) Initial seams before seam removal







(b) Intermediate results of seam removal. The blue line indicates the seam to be removed and the green lines are the neighboring boundary used to estimate developability.



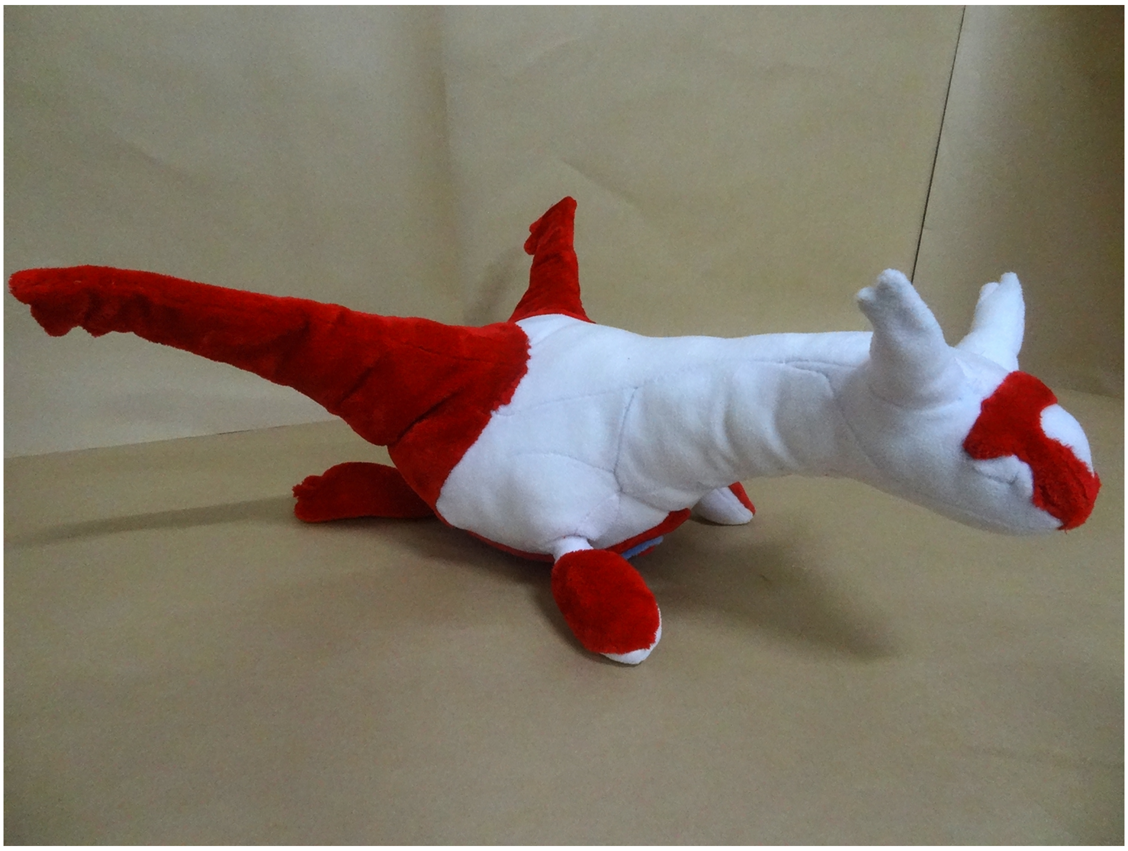
(c) Final result

Figure 5.16: The process of seam removal

Chapter 6

Experiments results

We test the proposed method using different types of models, showing the intermediate results and final results. We also fabricate a real plush using the result generated by the proposed method. There are few obvious overlapping errors in the generated patches caused by instability of LPFB [7]. In such case, we have to manually correct them during fabrication.



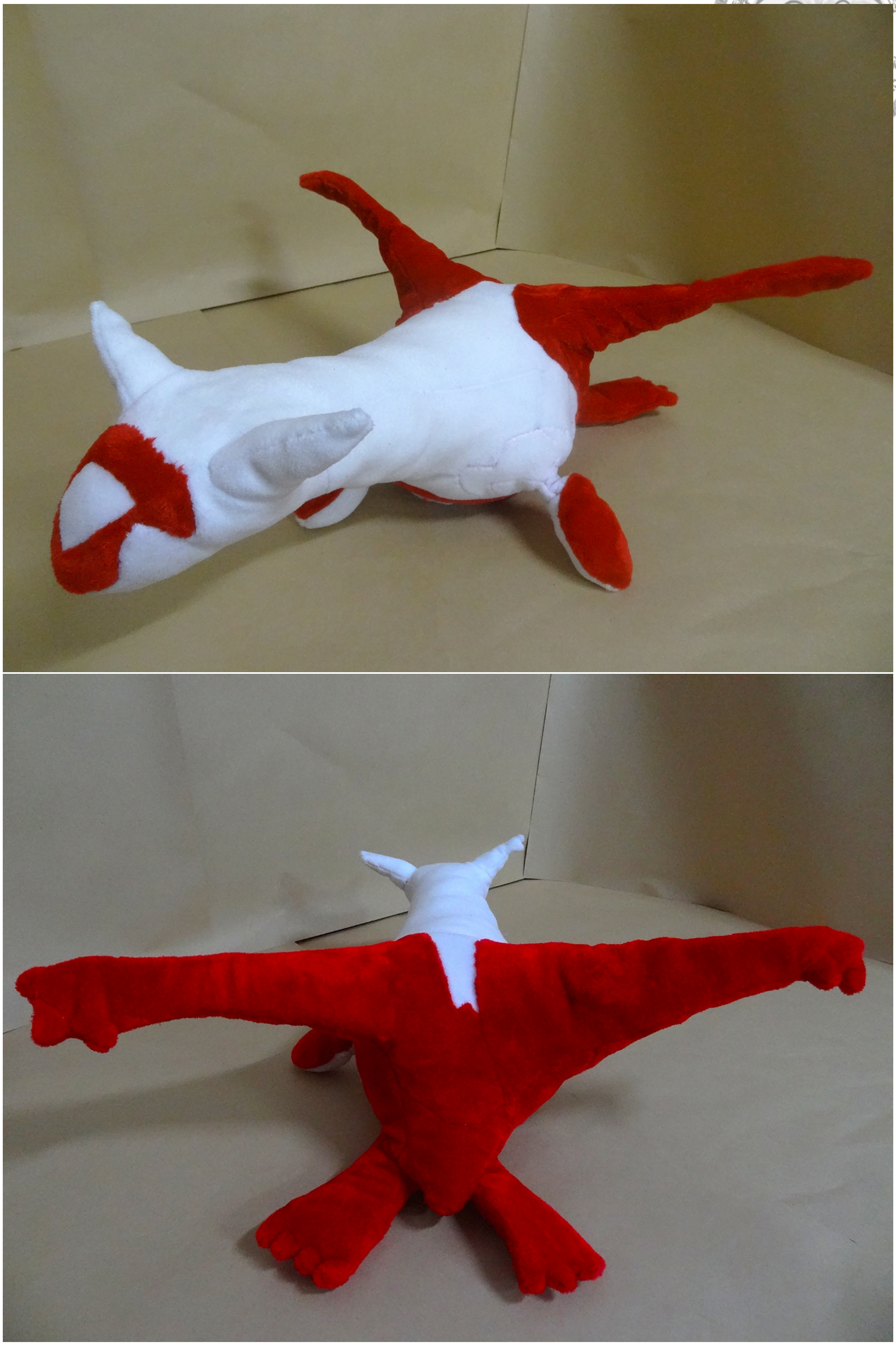
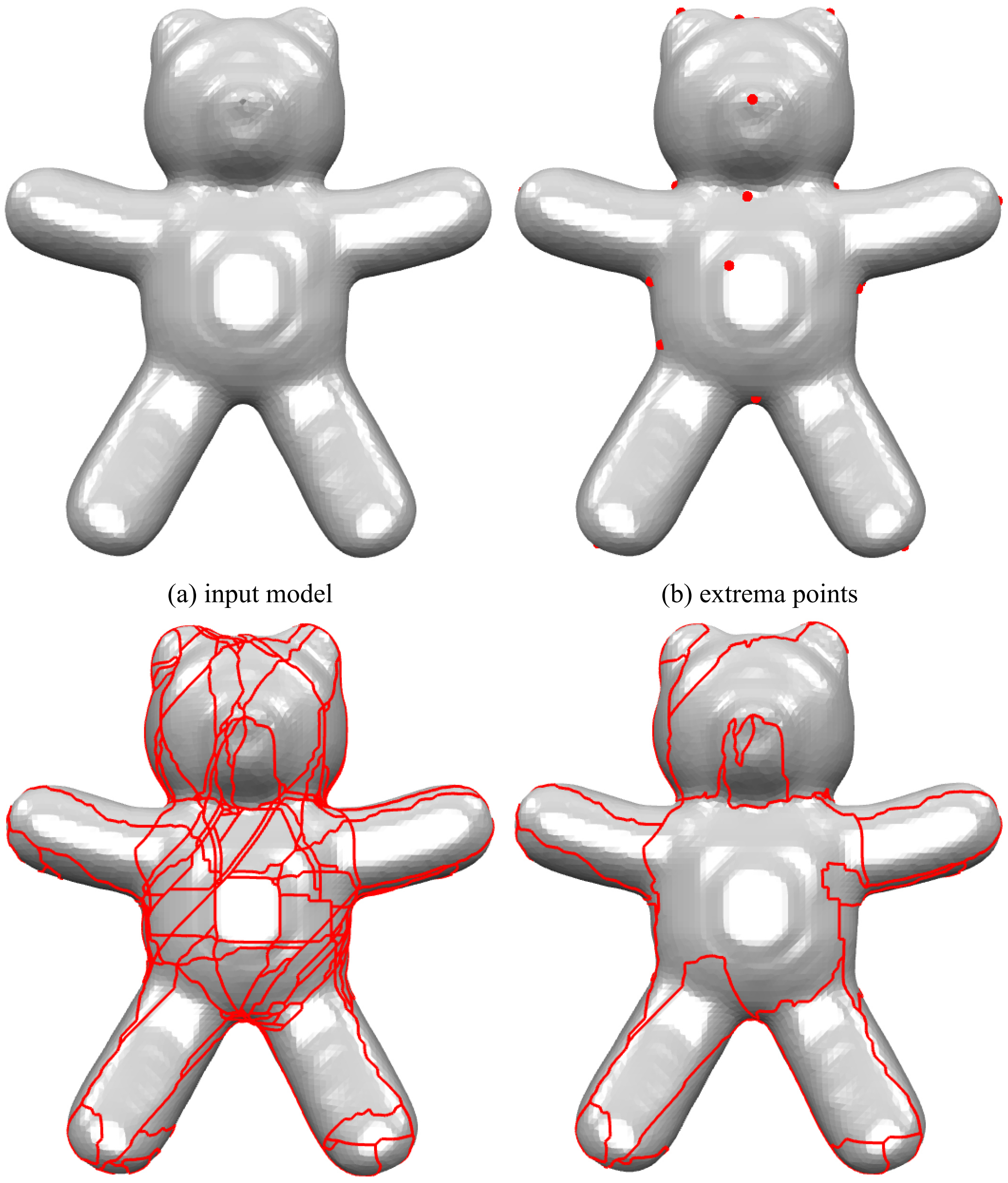


Figure 6.0: Fabricated plush from different views.

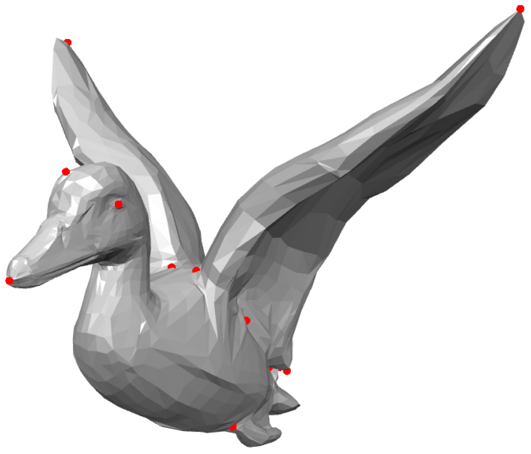


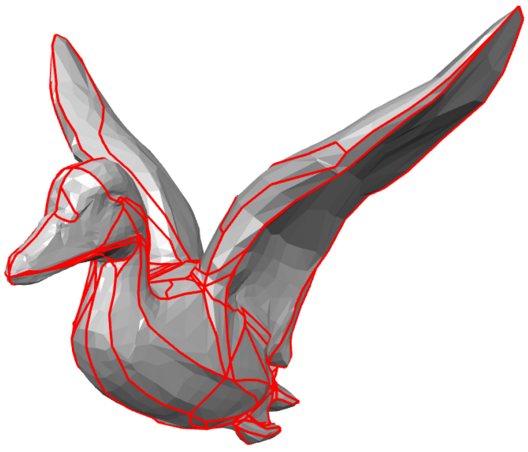
(a) input model (b) extrema points

(c) initial seams (d) after seam removal

Figure 6.1: Test case of teddy

(a) input model


(b) extrema points


(c) initial seams

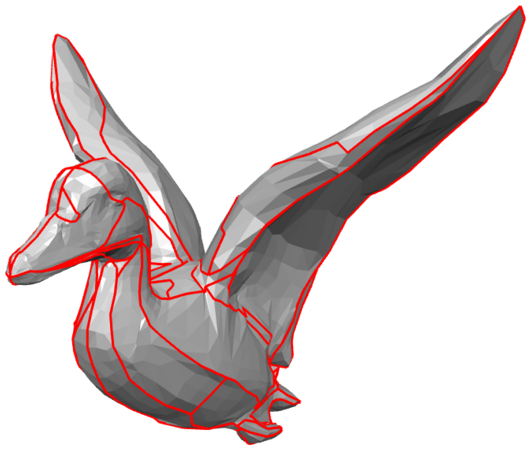
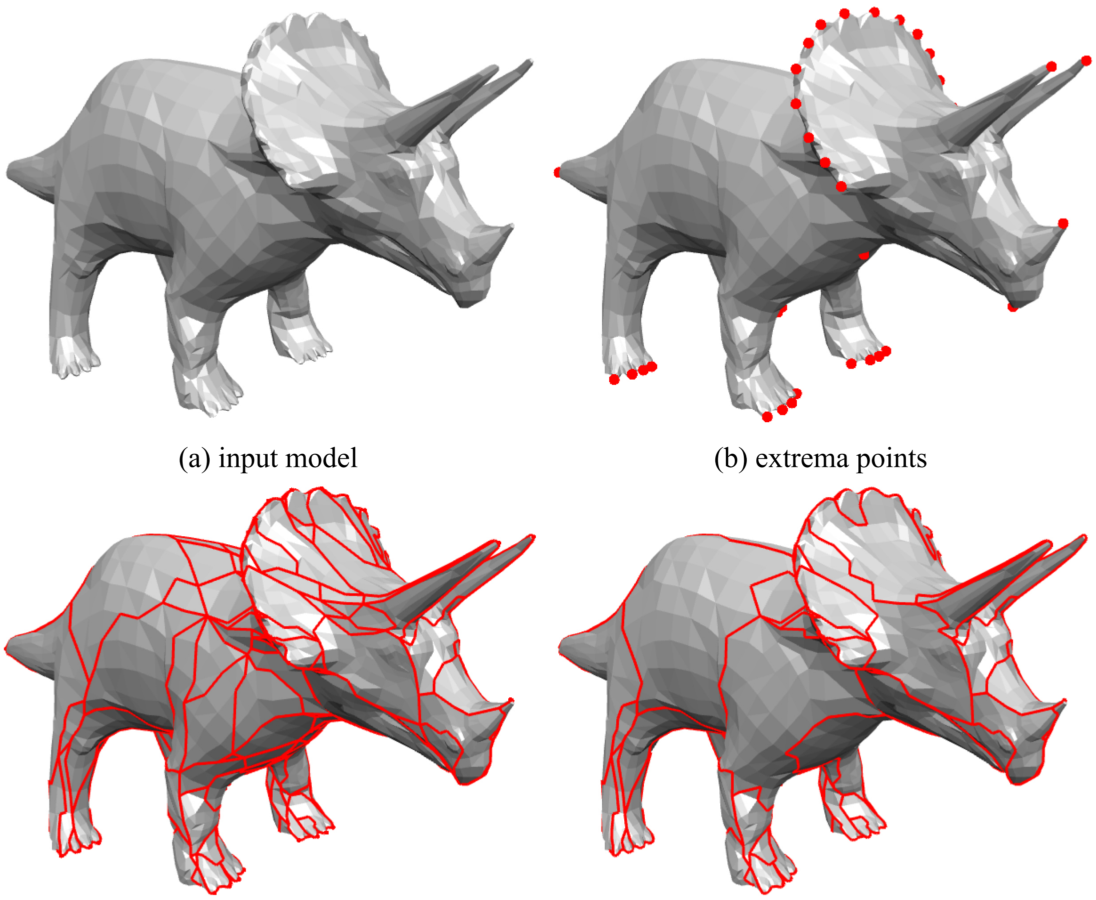

(d) after seam removal

Figure 6.2: Test case of bird

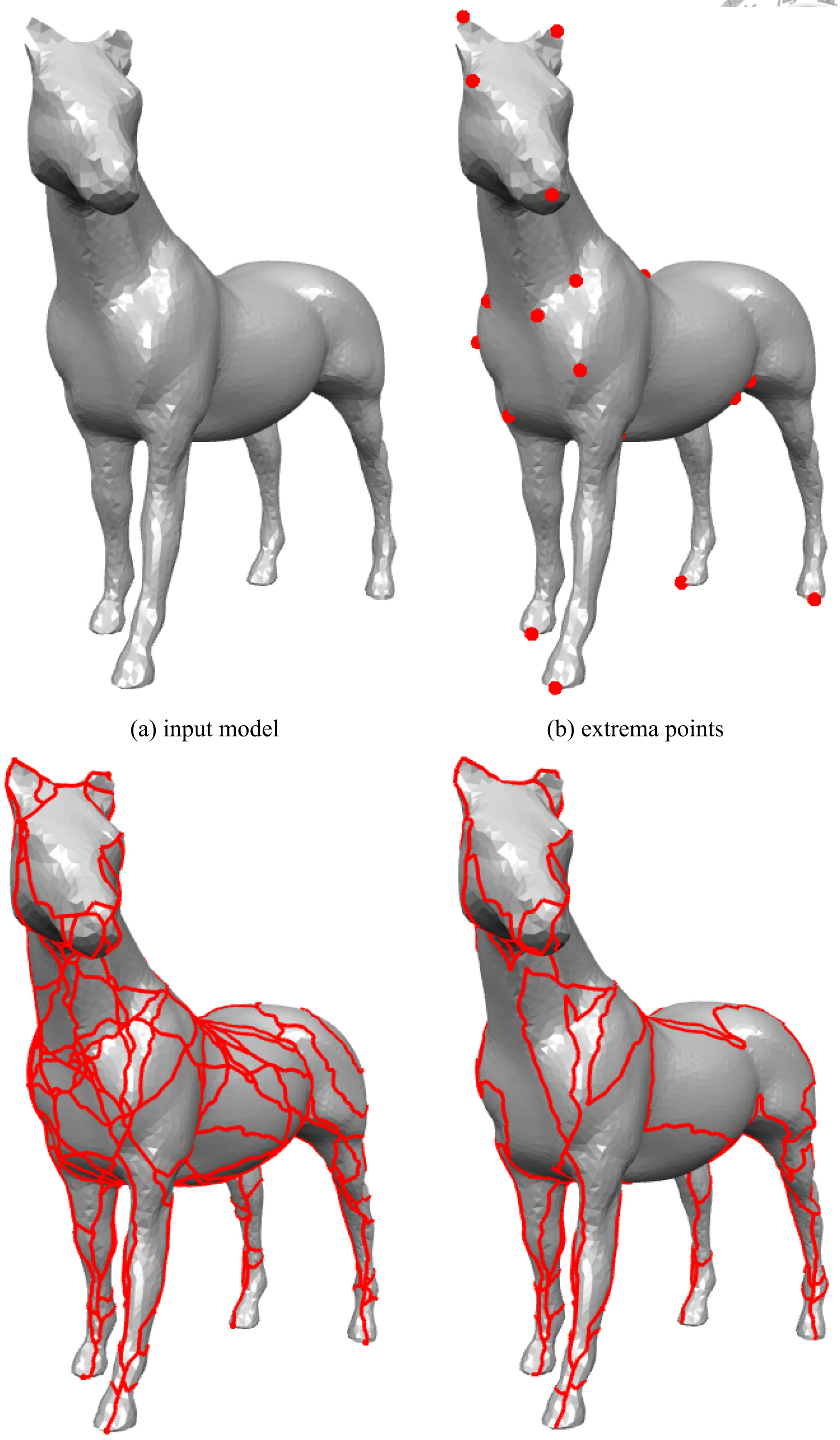

(a) input model

(b) extrema points

(c) initial seams

(d) after seam removal

Figure 6.3: Test case of triceratops



(a) input model (b) extrema points

(c) initial seams (d) after seam removal

Figure 6.4: Test case of horse

Chapter 7

Discussion

The weight function we proposed is composed with several terms which can be combined by summation or multiplication. In our experiments, we found multiplication yield a better result. This can be explained by the influence of each term. We design each term to be in range of $[0, 1]$. For a given edge, if one of these terms give a weight close to zero, then this term is a dominant strategy than others. By multiplication, other terms will not affect the output weight even if they are close to 1. If summation is used, though the dominant one still exist, the output weight will become larger due to other terms.

7.1 Performance

Models	#verts	#faces	Time of seam generation (sec)	Time of seam removal(sec)
Latias	13573	27142	47.77	272.2
Teddy	12561	25118	17.50	132.74
Bird	2497	4990	6.71	19.17
Triceratops	2832	5660	3.81	53.26
Horse	8078	16152	87.64	16.59

Table 7.1: The number of models and corresponding execution time.

We implemented the proposed method as a plugin of OpenFlipper [23] and runs on a MacBook Pro with a 2.7GHz dual-core Intel CPU. The test models and the corresponding execution time is listed in Table 7.1. Most of the processing time is spent on computing LPFB to test whether a sub-mesh is developable.

7.2 Limitation

We rely on LPFB to test whether a sub-mesh is developable. However, LPFB becomes unstable when the given boundary is not smooth. Such case is inevitable since the boundary is generated by union of seams. Different types of seams will form abrupt turns or spikes at intersection point.

The greedy algorithm we used for seam removal does not give an optimal result. The order of seams removal plays an essential role in the final result. If one seam is removed, the neighboring two patches are merged and become harder to merge with other patches. This is the reason why our result contains many small patches because they are surrounded by larger patches that can not be merged anymore.

The result of our method depends on the property of the given mesh. It will not perform well for heavily smoothed shape which does not contains useful curvature information. For shapes like sphere which skeleton information is not available, the resulting seams will become longer and curvy since it does not know the overall direction.

7.3 Future works

We observed that the seam of commercial plush, is a subset of the features (ridges, contours, etc) of the target shape. Though we use curvature information to guide seams to across such region, it result is not as well as expected. One possible solution is to first generate the feature lines regardless of developability. Developability problem should be dealt after because seams are always a subset of feature lines. After that, we can then choose the only necessary feature lines for seams and test developability.

7.4 Conclusion

In this paper, we proposed a segmentation method minimizing visual defects of plush seams by considering the natural structures that will affect the feeling of human perception. Due to the lack of canonical answer, the whole idea is to find all possible answers and then prune less important candidates. We formulated the whole problem into a shortest path problem and find seams that comply with the natural structure we defined. By union of different seams, we obtain a set of all possible seams containing the sub-seams we need. The unwanted seams is pruned by a greedy method starting from least important candidates. We maintain developability by testing developability of each new sub-mesh formed by removing a seam segment during pruning. If the new sub-mesh is non-developable, this seam segment will not be removed. Finally, we got the a set of seams that fit the human perception while still maintain developability required for plush fabrication.

Bibliography

[1] Yuki Mori and Takeo Igarashi. Plushie: An interactive design system for plush toys. *ACM Trans. Graph. (Proc. SIGGRAPH '07)*, 26(3), July 2007.

[2] Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. Designing inflatable structures. *ACM Trans. Graph. (Proc. SIGGRAPH '14)*, 33(4):63:1–63:10, 2014.

[3] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3D mesh segmentation and labeling. *ACM Trans. Graph. (Proc. SIGGRAPH 2010)*, 29(3), 2010.

[4] Dan Julius, Vladislav Kraevoy, and Alla Sheffer. D-Charts: Quasi-developable mesh segmentation. *Comput. Graph. Forum (Proc. EG '05)*, 24(3):581–590, 2005.

[5] Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogomyakov. ABF+ +: Fast and robust angle based flattening. *ACM Trans. Graph.*, 24(2):311–330, 2005.

[6] Idan Shatz, Ayellet Tal, and George Leifman. Paper craft models from meshes. *Vis. Comput.*, 22(9):825–834, 2006.

[7] Charlie Wang. Computing length-preserved free boundary for quasi-developable mesh segmentation. *IEEE Trans. Vis. Comput. Graph.*, 14(1):25–36, 2008.

[8] Autocad. http://www.autodesk.com/products/autocad/.

[9] Pro/engineer. http://www.ptc.com/product/creo/proengineer.

[10] 3ds max. http://www.autodesk.com/products/3ds-max/.

[11] Charlie Wang. Flattenable mesh surface fitting on boundary curves. *Journal of Computing and Information Science in Engineering*, 8(2):021006, 2008.

[12] Shymon Shlafman, Ayellet Tal, and Sagi Katz. Metamorphosis of polyhedral surfaces using decomposition. In *Comput. Graph. Forum*, pages 219–228, 2002.

[13] Sagi Katz, George Leifman, and Ayellet Tal. Mesh segmentation using feature point and core extraction. *Vis. Comput.*, 21(8–10):649–658, 2005.

[14] Marco Attene, Bianca Falcidieno, and Michela Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *Vis. Comput.*, 22(3):181–193, 2006.

[15] Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *Vis. Comput.*, 24(4):249–259, 2008.

[16] Mathieu Desbrun, Mark Meyer, and Pierre Alliez. Intrinsic parameterizations of surface meshes. *Comput. Graph. Forum*, 21(3):209–218, 2002.

[17] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3): 362–371, 2002.

[18] Michael S Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003.

[19] Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. Conformal flattening by curvature prescription and metric scaling. Citeseer.

[20] Alla Sheffer and John C. Hart. Seamster: Inconspicuous low-distortion texture seam layout. In *Proc. IEEE Vis '02*, pages 291–298, 2002.

[21] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM Trans. Graph. (Proc. SIGGRAPH '09)*, 28(3): 71:1–71:9, 2009.

[22] Liang-Tsen Shen, Sheng-Jie Luo, Chiun-Kai Huang, and Bing-Yu Chen. SD models: Super-deformed character models. *Comput. Graph. Forum (Proc. PG '12)*, 31(7): 2067–2075, 2012.

[23] Jan Möbius and Leif Kobbelt. Openflipper: An open source geometry processing and rendering framework. In *Proceedings of the 7th International Conference on Curves and Surfaces*, pages 488–500, 2012.