Efficient sketch-based character modelling with primitive deformer and shape generator

Anonymous cvm submission

Paper ID ****

Abstract

This paper proposes a new sketch-guided and ODE-driven character modelling technique. Our system consists of two main components: primitive deformer and detail generator. With such a technique, we first draw 2D silhouette contours of a character model. Then, we select proper primitives and align them with the corresponding silhouette contours. After that, we develop a sketch-guided and ODE-driven primitive deformer. It uses ODE-based deformations to deform the primitives to exactly match the generated 2D silhouette contours in one view plane and obtain a base mesh of a character model consisting of deformed primitives. In order to add various 3D details, we develop a local shape generator which uses sketches in different view planes to define a local shape and employs ODE-driven deformations to create a local surface passing through all the sketches. The experimental results demonstrate that our proposed approach can create 3D character models with 3D details from 2D sketches easily, quickly and precisely.

1. Introduction

- We develop an efficient sketch-guided and ODE-driven primitive deformer to create a base mesh. It can deform primitives to exactly match the generated silhouette contours. Compared to the existing methods, it automates shape manipulation, avoids tedious manual operations, can deform primitives to match the generated silhouette contours quickly, and is powerful in achieving different shapes of a same primitive.
- We develop a detail generator to add 3D details to the base mesh. Our proposed sketch-guided and ODE-driven local shape creator can create a new local shape to match user's drawn sketches in different views quickly. The image-based detail generator can automatically generate fine details from 2D images.
- Our character modelling system provide editing operations, support users to manipulate the control curves the draw in the creation stage and adding more control curves after the surface has been created. With our developed system, 3D character models with 3D details can be created easily and efficiently.

The rest of the paper is organised as follows. The previous related work is briefly reviewed in Section 2. The system overview of our proposed approach is presented in Section 3. Primitive deformer is examined in Section 4, and Detail generator is investigated in Section 5. Finally, the conclusions and future work are discussed in Section 6.

2. Related Work

2.1. Inflation Technology

Over the past two decades, sketch-based-modelling (SBM) has been widely studied in the computer graphic research community. Several research based systems have been proposed to generate organic models. The surface inflation technique extrudes the polygonal mesh from the skeleton outwards do a good job in modelling stuffed toys. One trend is to inflate freeform surfaces to create simple stuffed animals and other rotund objects in a sketch-based modelling fashion, such as [8, 9, 12]. The Teddy system [8] is the pioneer, it takes closed curves as inputs and find their cordial axes as spline, then wrap the splines with the polygonal mesh. Later, FiberMesh[12] enriched the editing operations for the inflating base mesh. FiberMesh also presents two types of the control curves: smooth and sharp. A smooth curve constrains the surface to be smooth across it, while a sharp curve only places positional constraints with C0 continuity. Sharp control curves appears when operations like cutting, extrusion and tunnel take place. Sharp control curves also serves the creation of creases on surface. However, FiberMesh [12] doesn't allow users to specify the ROI.Based on the study from William[16], the SmoothSketch system addressed the problem of T-junction and cusp, which Teddy fails to solve.

2.2. Primitive Technology

Unlike the inflating systems, primitives-based systems deconstruct the modelling task as a process of creating a certain set of geometry primitives and further editing on the primitives. The idea of assembling simple geometry primitives to form 3D models is very common in CSG(constructive solid geometry) modelling, related researches including [14, 4]. Shtof et.al [14] introduces a snapping method which helps determining the position and core parameters of several simple geometry primitives. In [4], the authors provides tools for generating a cylinder from only 3 strokes: the first two strokes define the 2D profile and the last stroke defines the axis along which the profile curve will sweep. Copies of the profile are not only perpendicularly aligned to the axis, but also resized to snap to the input outlines. However their work is only for man-made objects which simple sweeping surface can meet the quality requirements of the shapes. Structured Annotations for 2D-to-3D Modelling [7], on the other hand, focus on organic modelling. It is a system using two sets of the primitives, one is generalized cylinders, created by the input of a single open sketch stroke representing the spline, and then modified by using simple gestures such as tilt, scale local radius, rotate symmetrical plane, and change cap size; the other primitive is ellipsoid, generated according to the drawn closed ellipse sketch stroke. As the system's name indicates, there are a set pf annotation tools to further editing the surface shape using the annotations such as same-lengths, same angles, alignment and mirror symmetry.

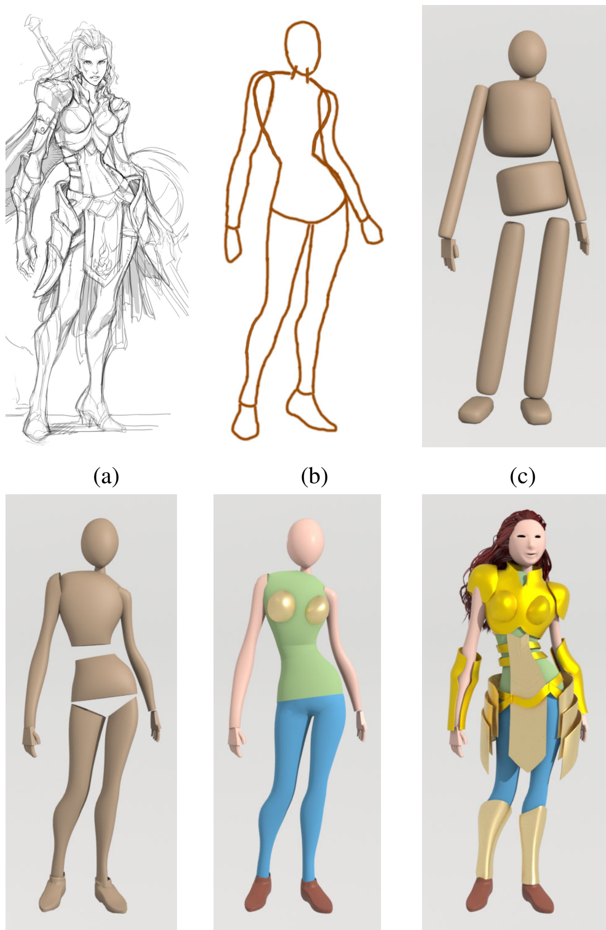
2.3. Mesh editing

A universal disadvantage of above systems is features provided are limited it was not suited to modelling complex and production ready models. This is partly due to their limitations in mesh editing operations, so several sketch-based editing methods have been development to help to improve the performance of sketch-based modelling system. One group of sketch-based editing methods [6, 10] treats individual sketch curve as the reference for deformations like bending. Take the [10] for example, user will draw a first curve as a reference curve to both set the ROI (region of interest) and be a controlling 'skeleton', and a new sketch curve indicating the desired deformation of the reference curve. Then mapping the reference curve to the second curve, the ROI will be deformed. These methods are very useful for simple keyframe animation, but for more small-scaled shape's editing task, studies like [11, 13, 17] are more suitable. These studies take mesh's contours (depend on the viewplan at the time of editing operation is happening) as reference curves, and the newly sketched curves indicating the new positions that the ROI of mesh should be reconstructed to. As they aiming at maintaining the local geometrical details, Laplacian coordinates and correspondingly laplacian surface editing technology are implemented to preserving the features.

3. System Overview

Based on our proposed approach, we have developed a modelling system which is composed of two main components: primitive deformer and detail generator. The primitive deformer is used to deform primitives to exactly match user's generated 2D silhouette contours and create a rough base mesh. The detail generator consists of local shape creator and image-based detail generator. The local shape creator creates a new local shape from four different algorithms. They are local shape creation from: 1) two open silhouette contours in two different view planes, 2) one open and one closed silhouette contour in two different view planes, 3) two open and one closed silhouette contours in three different view planes, and 4) two closed curves. The image-based detail generator generates fine 3D details from 2D images automatically through the SFS algorithm. The modeling process using our developed system is demonstrated in Figure 1 where a 3D female warrior model is created.

First, 2D character silhouette contours are generated. Users can draw their own silhouette contours directly or input their selected sketches into our developed system. If the selected sketches are input into our developed system, a further process may be required to extract the 2D silhouette contours from the input sketches. For the example demonstrated in Figure 1, a 2D female warrior sketch shown in Figure 1a is input. Then, users can extract 2D silhouette contours from the input sketch as shown in Figure 1b. After that, proper primitives are selected and placed to align with the corresponding silhouette contours through purely geometric transformations as shown in Figure 1c. Since the silhouette contours of the primitives do not match the generated 2D silhouette contours of the 2D female warrior, the primitive deformer developed from sketch-guided and ODE driven deformations described in Section 4 is applied to deform the primitives to exactly match the corresponding 2D silhouette contours and create a rough 3D base mesh depicted in Figure 1d. Once a 3D base mesh model is obtained, the detail generator described in Section 5 is employed to add 3D details to the 3D base mesh. First, the local shape creator developed in Subsection 5.2 is used to add local shapes and smoothly connect primitives together as demonstrated in Figure 1e. After that, we want to add fine details such as a dragon to the female warrior model. Since creating fine 3D details is not an easy task, we apply the image-based detail generator described in Subsection 5.3 to achieve the complicated 3D details shown in Figure 1f.
In the following two sections, we will introduce in detail the primitive deformer and the detail generator, respectively. Some examples will be presented to demonstrate their applications.



(a) (b) (c)

(d) (e) (f)

Figure 1: Quick creation of a 3D female warrior model: (a) 2D female warrior sketch, (b) 2D silhouette contours, (c) base mesh without primitive deformations, (d) base mesh by deforming primitives to match generated 2D silhouette contours or adding local shapes created from 2D silhouette contours, (e) base mesh by adding blending surfaces to smoothly connect deformed primitives, (f) detail generation through local shape creation and shape from shading (Sketch by EngKit Leong)

4. Primitive deformer

As shown in Figure 1c, the base mesh of the female warrior without primitive deformations cannot exactly match the generated 2D silhouette contours of the female warrior. In order to tackle this problem, in this section, we develop a primitive deformer. In the subsections below, we first introduce the interface of the primitive deformer in Subsection 4.1. Then, we discuss the algorithm of the primitive deformer in Subsection .

4.1. User interface of primitive deformer

The user interface of our developed primitive deformer uses four windows. The upper left window is used to display 3D base mesh without primitive deformations in the front view. The upper right window is used to display primitives and the user-drawn 2D silhouette contours for these primitives. If required, the user can edit the generated 2D silhouette contours in the upper right window. The deformed primitives are shown in the bottom windows where the left is from the front view and the right is from the side view. Taking the left leg of the female warrior shown in Figure 2 as an example, the primitive of the left leg is shown in the upper left window. It and the user-drawn 2D silhouette contours are depicted in the upper right window. Our proposed primitive deformer described in Subsection 4.2 deforms the primitives of the 3D base mesh to exactly match the generated 2D silhouette contours and create a deformed 3D base mesh with primitive deformations, and depicted the deformed 3D base mesh in bottom windows.

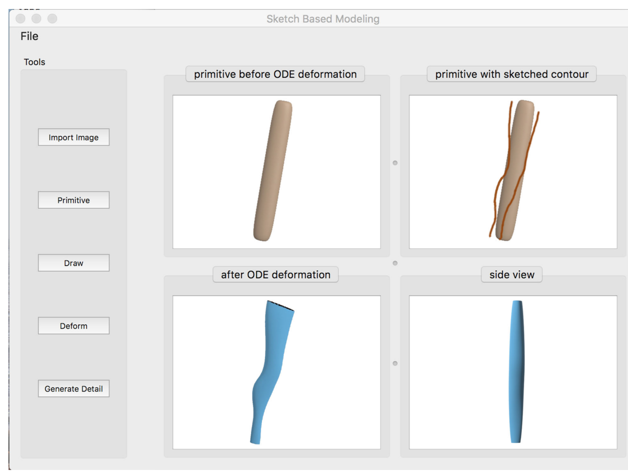
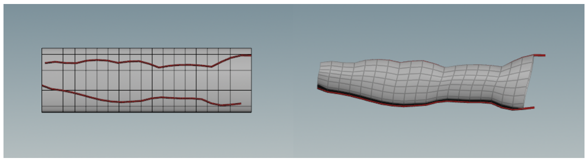


Figure 2: Interface for primitive deformer: (a) 3D base mesh of the left leg of the female warrior without primitive deformations, (b) 2D silhouette contours of the female warrior leg sketch and primitive, (c) and (d) front view and side view of the 3D left leg base mesh after primitive deformations

4.2. Algorithm of primitive deformer

After 3D primitives have been placed and aligned with the generated 2D silhouette contours, these 3D primitives should be deformed so that their 2D silhouette contours can match the generated 2D silhouette contours exactly. Here we use the example shown in Figure 3 to demonstrate the algorithm of our proposed primitive deformer and how it deforms a 3D primitive to match the 2D silhouette contours.



(a) (b)

Figure 3: Primitive deformer: a) female warrior arm represented with a cylinder and its 2D silhouette contour, b) deformed shape of the cylinder

Figure 3a depicts an arm model of the female warrior which is represented with a cylinder. The 2D silhouette contour to be matched is also shown in the image. Figure 3b shows how the cylinder is deformed with the algorithm developed below to match the 2D silhouette contour exactly. This is achieved by a sketch-guided and ODE-drive primitive deformer. It is developed from a simplified version of the Euler-Lagrange PDE (partial differential equation) which is widely used in physically-based surface deformations and briefly introduced below.

As discussed in [1], the main requirement for physically-based surface deformations is an elastic energy which considers locally stretching for solid objects plus bending for two-manifold surfaces called thin-shells. When a surface $\mathbb{S} \subset \mathbb{R}^3$ parameterized by a function $\mathbf{P}(\mathbf{u}, \mathbf{v}) : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{S} \subset \mathbb{R}^3$is deformed to a new shape $\mathbb{S}'$ through adding a displacement vector $\mathbf{d}(\mathbf{u}, \mathbf{v})$ to each point $\mathbf{P}(\mathbf{u}, \mathbf{v})$, the change of the first and second fundamental $I(u, v), \Pi(u, v) \in \mathbb{R}^{2\times2}$,forms in differential geometry [5] yields a measure of stretching and bending descried by [15]

$$E_{shell}(S') = \int_{\Omega} k_s \|I' - I\|_F^2 + k_b \|\Pi' - \Pi\|_F^2 dudv \quad (1)$$

Where ,I',Π' are the first and second fundamental forms of the surface $\mathbf{S}'$, $\|.\|$ indicates a (weighted) Frobenius norm, and the stiffness parameters k_s and k_b are used to control the resistance to stretching and bending.

Generating a new deformed surface requires the minimization of the above equation which is non-linear and computationally too expensive for interactive applications. In order to avoid the nonlinear minimization, the change of the first and second fundamental forms is replaced by the first and second order partial derivatives of the displacement function $\mathbf{d}(\mathbf{u}, \mathbf{v})$ (Celniker, 1991), (Welch, 1992) , i. e.,

$$\begin{aligned} \tilde{E}_{shell}(d) = \int_{\Omega} & k_s(\|d_u\|^2 + \|d_v\|^2) \\ & + k_b(\|d_{uu}\|^2 + 2\|d_{uv}\| + \|d_{vv}\|^2) dudv \end{aligned} \quad (2)$$

where $d_x = \frac{\partial}{\partial x}$ and $d_{xy} = \frac{\partial^2}{\partial x \partial y}$. The minimization of the above equation can be obtained by applying variational calculus which leads to the following Euler-Lagrange PDE

$$-k_s \triangle d + k_b \triangle^2 d = 0 \quad (3)$$

where $\triangle$ and $\triangle^2$ are the Laplacian and the bi-Laplacian operator, respectively.

$$\begin{aligned} \triangle d &= div \nabla d = d_{uu} + d_{vv} \\ \triangle^2 d &= \triangle(\triangle d) = d_{uuuu} + 2d_{uuvv} + d_{vvvv} \end{aligned} \quad (4)$$

Using the sketched 2D silhouette contours shown in Figure 3a to change the shape of the primitive can be transformed into generation of a sweeping surface which passes through the two sketched 2D silhouette contours. The generator creating the sweeping surface is a curve of the parametric variable u only, and the two silhouette contours are trajectories. If Equation (3) is used to describe the generator, the parametric variable in Equation (3) drops, and we have $d_{vv} = 0$ and $d_{vvvv} = 0$. Substituting and into Equation (3), we obtain the following simplified version of the Euler-Lagrange PDE (3)which is actually a vector-valued ordinary differential equation

$$k_b \frac{\partial^4 d}{\partial u^4} - k_s \frac{\partial^2 d}{\partial u^2} = 0 \quad (5)$$

As pointed out in [3] and [2] , the finite difference solution of ordinary differential equations is very efficient, we here investigate such a numerical solution of Equation (4).

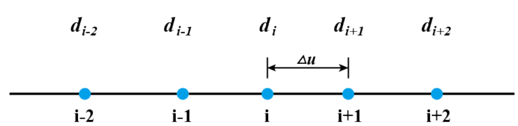


Figure 4: Typical node i for the finite difference approximations of derivatives

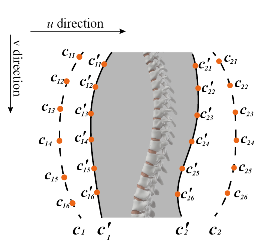
For a typical node shown in Figure 4, the central finite difference approximations of the second and fourth order derivatives can be written as [3]

$$\begin{aligned} \frac{\partial^2 d}{\partial u^2}|_i &= \frac{1}{\triangle u^2}(d_{i+1} - 2d_i + d_{i+1}) \\ \frac{\partial^4 d}{\partial u^4}|_i &= \frac{1}{\triangle u^4}[6d_i - 4(d_{i-1} + d_{i+1}) + d_{i-2} + d_{i+2}] \end{aligned} \quad (6)$$

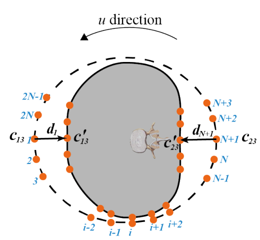
Introducing Equation (6) into Equation (5), the following finite difference equation at the representative node i can be written as:

$$\begin{aligned}(6k_b + 2k_s h^2)d_i + k_b d_{i-2} + k_b d_{i+2}\\ -(4k_b + k_s h^2)d_{i-1} - (4k_b + k_s h^2)d_{i+1} = 0\end{aligned} \tag{7}$$

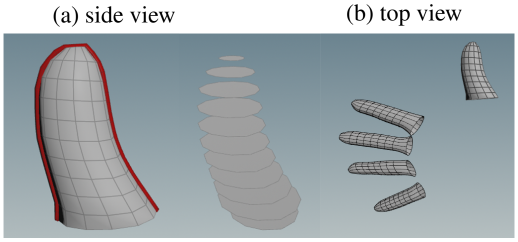
For character models, the 3D shape defined by two silhouette contours is closed in the parametric direction u as indicated in Figure 5b. Therefore, we can extract some closed curves each of which passes through the two corresponding points on the two silhouette contours. Taking the silhouette contours in Figure 5b as an example, we find two corresponding points $\mathbf{c_{13}}$ and $\mathbf{c_{23}}$ on the original silhouette contours $\mathbf{c_1}$ and $\mathbf{c_2}$, and two corresponding points $\mathbf{c'_{13}}$ and $\mathbf{c'_{23}}$ on the deformed silhouette contours $\mathbf{c'_1}$ and $\mathbf{c'_2}$ as shown in Figure 5b. Then we extract a closed curve $\mathbf{c}(\mathbf{u})$ passing through the two corresponding points $\mathbf{c_{13}}$ and $\mathbf{c_{23}}$ from the 3D model in Figure 5a and depicted it as a dashed curve in Figure 5(b). Assuming that the deformed shape of the closed curve $\mathbf{c}(\mathbf{u})$ is $\mathbf{c}'(\mathbf{u})$, the displacement difference between the original closed curve and deformed closed curve is $\mathbf{d}(\mathbf{u}) = \mathbf{c}'(\mathbf{u}) - \mathbf{c}(\mathbf{u})$. Our task is to find the displacement difference $\mathbf{d}(\mathbf{u})$ and generate the deformed curve $\mathbf{c}'(\mathbf{u}) = \mathbf{d}(\mathbf{u}) + \mathbf{c}(\mathbf{u})$.



(a) side view



(b) top view



(c) the making of the fingers

Figure 5: Finite difference nodes for local shape manipulation from sketches in different view planes and the deformed 3D finger models

In order to use the finite difference method to find the displacement difference $\mathbf{d}(\mathbf{u})$, we uniformly divide the closed curve into $2N$ equal interval as indicated in Figure 5. The displacement difference at node 1 and node N is known, i. e. $\mathbf{d_1} = \mathbf{c'_{13}} - \mathbf{c_{13}}$ and $\mathbf{d_{N+1}} = \mathbf{c'_{23}} - \mathbf{c_{23}}$.

When we write the finite difference equations for the nodes 2, 3, $2N-1$ and $2N$, the node 1 will be involved, and we have $\mathbf{d_1} = \mathbf{c'_{13}} - \mathbf{c_{13}}$. The finite difference equations at these points can be derived from Equation (7). Substituting $\mathbf{d_1} = \mathbf{c'_{13}} - \mathbf{c_{13}}$ into these equations, we obtain the finite difference equations for the nodes 2, 3, $2N-1$ and $2N$ When we write the finite difference equations for the nodes $N-1$,N, $N+2$ and $N+3$, the node $N+1$ will be involved, and we have $\mathbf{d_{N+1}} = \mathbf{c'_{23}} - \mathbf{c_{23}}$. Once again, the finite difference equations at these points can be derived from Equation (7). Substituting $\mathbf{d_{N+1}} = \mathbf{c'_{23}} - \mathbf{c_{23}}$ into these equations, we obtain the finite difference equations for the nodes N-1, N, N+2 and N+3

For all other nodes 3, 4, 5,..., $N-3$, $N-2$ and $N+4$,$N+5$,..., $2N-3$, $2N-2$, the finite difference equations are the same as Equation (7). For these nodes, there are $2N-5$ finite difference equations. Plus the 8 finite difference equations at node 2, 3, $N-1$, N , $N+2$, $N+3$, $2N-1$ and $2N$, we get $2N-2$linear algebra equations which can be solved to determine the unknown constants $\mathbf{d_2}$, $\mathbf{d_3}$, $\ldots$,$\mathbf{d_{N-1}}$, $\mathbf{d_N}$, $\mathbf{d_{N+2}}$, $\mathbf{d_{N+3}}$,..., $\mathbf{d_{2N-1}}$, and $\mathbf{d_{2N}}$. Adding the $\mathbf{d_i}$ ($i = 1, 2, \ldots, 2N-1, 2N$) to the original curve $\mathbf{c}(\mathbf{u})$, we obtain the deformed curve $\mathbf{c}'(\mathbf{u})$, and depict it as a solid curve in Figure 5(b). Repeating the above operations for all other points on the two silhouette contours, we obtain all deformed curves. These curves describe a new 3D deformed shape.

Taking a finger shown in Figure 5c as an example, the left image shows the silhouette contours. It is used deform a cylinder into the middle 3D model (the second image from the left). The cross-section shapes of the finger model are depicted in the third image from the left. The rightmost image shows the five finger models created with the above method.

With the primitive deformer developed above, we deform the primitive of the 3D base mesh shown in Figure 1c, obtain a deformed 3D base mesh, and depicted it in Figure 1d. It is clear that deformed primitives have matched the generated 2D silhouette contours exactly.

5. Detail generator

Compare the following:

`$conf_a$`	$conf_a$
`$\mathit{conf}_a$`	conf_a

See The TEXbook, p165.

The space after *e.g.*, meaning "for example", should not be a sentence-ending space. So *e.g.* is correct, *e.g.* is not. The provided `\eg` macro takes care of this.

When citing a multi-author paper, you may save space

Name	Performance
A	OK
B	Bad
Ours	Great

Table 1: An example for using tables.

by using "et alia", shortened to "*et al.*" (not "*et. al.*" as "*et*" is a complete word.) However, use it only when there are three or more authors. Thus, the following is correct: " Frobnication has been trendy lately. It was introduced by Alpher [**?**], and subsequently developed by Alpher and Fotheringham-Smythe [**?**], and Alpher *et al*. [**?**]."

This is incorrect: "... subsequently developed by Alpher *et al*. [**?**] ..." because reference [**?**] has just two authors. If you use the `\etal` macro provided, then you need not worry about double periods when used at the end of a sentence as in Alpher *et al*.

For this citation style, keep multiple citations in numerical (not chronological) order, so prefer [**?**, **?**, **?**] to [**?**, **?**, **?**].

6. Detail Generator

In order to add 3D details to 3D base mesh created from the primitive generator, we develop a detail generator consisting of a local shape creator and an image-based detail generator. In what follows, we first introduce the user interface of our developed detail generator in Subsection 5.1. Then we discuss the local shape creator and image-based detail generator in Subsections 5.2 and 5.3, respectively.

7. Conclusions and Future work

For now in the modelling stage of our system, all our control curves are creating smooth surfaces with C2 continuity, and the result template mesh looks pleasing as it is. In our future work, we will investigate the potentials of creases in more detailed character sculpting.

7.1. References

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [**?**]. Where appropriate, include the name(s) of editors of referenced books.

7.2. Illustrations, graphs, and photographs

All graphics should be centered. Please ensure that any point you wish to make is resolvable in a printed copy of the paper. Resize fonts in figures to match the font in the body text, and choose line widths which render effectively in print. Many readers (and reviewers), even of an electronic copy, will choose to print your paper in order to read it. You cannot insist that they do otherwise, and therefore must not assume that they can zoom in to see tiny details on a graphic.

When placing figures in LaTeX, it's almost always best to use `\includegraphics`, and to specify the figure width as a multiple of the line width as in the example below

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]
                {myfile.eps}
```

References

[1] M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE transactions on visualization and computer graphics*, 14(1):213–230, 2008. 4

[2] E. Chaudhry, S. Bian, H. Ugail, X. Jin, L. You, and J. J. Zhang. Dynamic skin deformation using finite difference solutions for character animation. *Computers & Graphics*, 46:294–305, 2015. 4

[3] E. Chaudhry, L. You, X. Jin, X. Yang, and J. J. Zhang. Shape modeling for animated characters using ordinary differential equations. *Computers & Graphics*, 37(6):638–644, 2013. 4

[4] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or. 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG)*, 32(6):195, 2013. 2

[5] M. P. Do Carmo, G. Fischer, U. Pinkall, and H. Reckziegel. Differential geometry. In *Mathematical Models*, pages 155–180. Springer, 2017. 4

[6] G. Draper and P. K. Egbert. A gestural interface to free-form deformation. In *graphics interface*, volume 2003, pages 113–120, 2003. 2

[7] Y. Gingold, T. Igarashi, and D. Zorin. Structured annotations for 2d-to-3d modeling. In *ACM Transactions on Graphics (TOG)*, volume 28, page 148. ACM, 2009. 2

[8] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 409–416. ACM Press/Addison-Wesley Publishing Co., 1999. 1

[9] O. A. Karpenko and J. F. Hughes. Smoothsketch: 3d free-form shapes from complex sketches. *ACM Trans. Graph.*, 25(3):589–598, July 2006. 1

[10] Y. Kho and M. Garland. Sketching mesh deformations. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pages 147–154. ACM, 2005. 2

[11] Y. Lipman, O. Sorkine, D. Cohen-Or, D. Levin, C. Rossi, and H.-P. Seidel. Differential coordinates for interactive mesh editing. In *Shape Modeling Applications, 2004. Proceedings*, pages 181–190. IEEE, 2004. 2

[12] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa. Fibermesh: designing freeform surfaces with 3d curves. *ACM transactions on graphics (TOG)*, 26(3):41, 2007. 1

[13] A. Nealen, O. Sorkine, M. Alexa, and D. Cohen-Or. A sketch-based interface for detail-preserving mesh editing. In *ACM SIGGRAPH 2007 courses*, page 42. ACM, 2007. 2

[14] A. Shtof, A. Agathos, Y. Gingold, A. Shamir, and D. Cohen-Or. Geosemantic snapping for sketch-based modeling. In

Computer graphics forum, volume 32, pages 245–253. Wiley Online Library, 2013. 2

[15] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *ACM Siggraph Computer Graphics*, volume 21, pages 205–214. ACM, 1987. 4

[16] L. R. Williams and D. W. Jacobs. Stochastic completion fields: A neural model of illusory contour shape and salience. *Neural computation*, 9(4):837–858, 1997. 1

[17] J. Zimmermann, A. Nealen, and M. Alexa. SilSketch: Automated Sketch-Based Editing of Surface Meshes. In M. van de Panne and E. Saund, editors, *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*. The Eurographics Association, 2007. 2