Axial Deformation with Controllable Local Coordinate Frames

CHOW, Yuk Pui

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Automation & Computer-Aided Engineering

The Chinese University of Hong Kong
July 2010


香港中文大學
大學
圖書館系統
07 MAR 2012
UNIVERSITY
LIBRARY SYSTEM
THE CHINESE UNIVERSITY OF HONG KONG

Declaration

I hereby declare that the Master of Philosophy thesis titled "Axial Deformation with Controllable Local Coordinate Frames" represents my own work. I also declare that the work reported in this thesis has not been previously included in a thesis, dissertation or report submitted to this University or to any other institution for a degree, diploma or other qualifications.

Chow Yuk Pui

January 2010

Acknowledgements

Firstly, I would like to express my acknowledge to the Chinese University of Hong Kong and the Mechanical and Automation Engineering for having provided an excellent academic atmosphere during the five years of my study here. There are so many people who helped and encouraged me during my study, that I cannot mention all those names here. However, I would like to take this opportunity to say thank you to those have provided me with helps and supports. Without their kindles assistance, I would not be able to complete the work in this thesis.

I would like to express my sincere gratitude to my supervisor, Professor Kin Chuen Hui, for his guidance and advice throughout my master of philosophy studying. He gives me a lot of encouragement and freedom in research works. In addition, he always gave constructive advice and provided supports to me. These supports ease the progress of my research work. I treasure a lot from him.

I would like to say thank you to my parents, who always support me and provided motivation for work. They have provided me with the best environment that allows me to concentrate on my work. Without their consideration and support, it would be impossible for me to take up the research work.

I am also grateful to Prof. C.L. Wang and Prof. C.K. Chung for giving some suggestions on my research topic after the yearly presentation. I am greatly indebted to them for help me in developing the new ideas.

Appreciation is also expressed to the graduate students and staffs in the Department of Mechanical and Automation Engineering for their help and concern. Without their helps in maintaining the computer facilities, implementation of the algorithms would not be possible.

Last but not least, I would like to thank the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong for the support and facilities provided. Thanks are also due to the Research Grant Council of SAR Government for providing financial support.

Abstract

Deforming freeform object is an important operation in computer-assisted geometric design and animation. Various approaches, e.g., point-based, lattice-based, and skeleton-based techniques are used for deforming freeform objects. The axial deformation technique adjusts an axial curve to deform the shape of a 3D object. This technique plays important role in computer graphics and computer-aided design. Recently, a curve pair based deformation technique has been proposed which allows the local coordinate frame along a curve to be controlled intuitively. Nevertheless, due to insufficient control on the local coordinate frame between control points of the curve-pair, unexpected twist may be obtained. An intuitive deformation tool is essential in computer-aided design applications where freeform control of an existing object is required.

In this thesis, we propose a technique for deforming 3D objects using an axial curve, which allows direct control on the local coordinate frames (LCFs) defined on the axial curves. Users can move and twist the LCFs intuitively, while maintaining the continuity and the smoothness of the shape. This avoids undesirable twists as a result of the lack of control on the LCFs. The continuity and smoothness of the axial curve, and hence the associated geometric shape is maintained by adopting a suitable interpolation scheme to determine the LCF between user defined LCFs.

In our investigation, the surfaces of the deformed object may overlap in an axial deformation. As a result, this causes self-intersection during the deformation. We introduced a technique to detect self-intersection using the axial curve. The system can use a set of controllable local coordinate frames to detect the self-intersection of the object which is deformed by axial curve, an algorithms is proposed to detect the self-intersection of the object surface by considering the planes constructed by the LCFs. Firstly, a set of normal planes are built according to the LCFs on an axial curve. During the deformation, intersection may be occurred when two planes are not parallel and they intersect in a line. But the mesh may not have self-intersection when the planes intersect.

Therefore, the distance between the vertices to the axial curve has to be considered in the detection. Then a sphere is constructed on the axial curve point. And its radius is the distance between the mesh vertex to the curve point. By projecting the sphere on to a normal plane, gives a circle on the normal plane. Finally, using the lengths comparison, we can compare the length between the intersection line to the curve point and the diameter of the circle. If self intersection occurs on an object mesh, the intersection line passes through the circles on the planes.

論文摘要

在電腦輔助結構設計和動畫中，自由變形技術是當中一項重要的工具。不同的表示，例如: 表示和骨骼表示會運用在自由變形技術。不過，軸向變形是指透過軸線的調整令三維模型變形。在電腦圖像和電腦輔助設計中，這項技術扮演著重要的角色。 最近，介紹了以一對曲線爲變形的技術,令坐標可以真實地控制。但是，這項技術會造成線對的控制點扭轉一團。在電腦輔助設計中，需要能自由地控制已成的物件，所以真實的變形工具是必需的。

在這份論文中，我們將會介紹一項技術利用軸線去變形三維模型，這項技術允許直接控制在軸線局部坐標。使用者可以隨心所欲地移動和扭動局部坐標，同時維持形狀的持續性和平滑。這樣避免想像之外的扭動，結果座標失去了真實的控制。軸線的持續和平順，去估計局部座標在使用者訂下的，從而維持相關結構的形狀。

在我們的研究下，變形的物件的表面可能重疊，當軸向變形沒有任何邊界測試，結果這樣造成自交情況。所以, 我們介紹利用軸線去偵測自交情況。當變形時，因爲系統可以利用一組可控制的局部坐標偵測被軸線變形的物件的自交。首先,利用軸線上的局部坐標去建立一組與軸線垂直的平面。當兩塊平面的垂直向量不是平行, 它們可能發生貫穿的情況貫穿成一條直線。 但兩塊平面貫穿不等於物件上的點發生自交。所以當模型變型時, 我們需要考慮物件上的點與軸條上的點成形成的距離。之後一個圓球體會建在軸線的點上, 圓球體的半徑是物件上的點與軸條上的點成形成的距離。另外把圓球體投射在垂直的平面上, 結果一個圓形放在平面上。最後通過比較長度, 偵測貫穿線與軸條點和圓形的直徑的長度。如果在垂直的平面上,貫穿線穿過圓形, 物件上的點就會發生自交情況。

Contents

List of Figures

1. Introduction

1.1 Motivation

Axial Deformation is a popular technique for deforming existing geometric shapes. This technique is widely used for the modeling of objects in aesthetic design, 3D modeling and animation. Modeling and animation of 3D object has long been a research goal in computer-aided design. Various deformation techniques have been proposed for the deformation of object, such as lattice-based deformation, skeleton-based deformation and point-based deformation. Axial deformation is an efficiently technique for modeling soft objects. Nevertheless, the details (fairness and smoothness) of the deformed geometric model may be undesirably distorted in the deformation.

Axial Deformation uses an axial curve to adjust the shape of an object. An intuitive axial deformation maintains the smoothness and continuities of the 3D model. Nerveless, accurate deformation requires expensive computation which is not an efficiency method. Although different approaches have been introduced for modeling 3D object by an axial curve, the basic problem of specifying local coordinate frames in an axial deformation is still not well addressed.

Recently, a curve pair-based deformation is proposed basing on the axial deformation technique. A curve-pair is used to control the twisting and bending effect in a deformation. However, unexpected twisted may be obtained when the curve of curve-pair overlap. Moreover, self-intersection may be obtained in an axial deformation. Therefore, a technique to detect the self-intersection in a deformation is required. Most popular methods for detecting self-intersection use radii and distance map to detect the self intersection on the surface of the model. There is not much work using the deformation technique to detect the self intersection in real time.

1.2 Objectives

The objective of this thesis is to develop an intuitive axial deformation technique which also detects the self intersection during the deformation. Although the axial deformation [1] is an effective tool for 3D modeling, classical axial deformation technique does not control on the twist of an object. Curve-pair based deformation technique [2] improve the axial deformation technique in order to provide an intuitive twisting deformation, but an unexpected twisted may be obtained by the overlap of the curves. In this research, we proposed to use a set of controllable local coordinate frame (LCF) to improve the axial deformation technique.

An axial curve and a set of user defined local coordinate frame constitute the basic elements of our framework. An object attached to the axial curve can be deformed by adjusting the curve. One important issue is the control of the twist of the curve. Instead of using an additional orientation curve [2], we make use of a set of local coordinate frame defined on the axial curve. The twist of the curve can be directly controlled by manipulating the local coordinate frames. By attaching an object to the axial curve, each vertex of the object is specified relative to a local coordinate frame, and hence the object can be deformed according adjusting the local coordinate frame. To maintain the smoothness of the model, an interpolation scheme is adopted to obtain a smooth and continue orientation of the LCF between the users defined LCFs.

Self-intersection is always a major problem in various deformation methods. Approach for detecting self-intersection include the use of distance map [3], modifying the shape of an axial curve [1] and offset curves [4]. Preventing the self intersection problem by adjusting the position of the control point on an axial curve, this is an efficient method for preventing self intersection, but the shape of the axial curve cannot be control intuitively. And since the curve is a non-trivial task to detect and trim all local and global self intersections, detecting and eliminate the self intersection problem by an offset curve and surface is a difficult task.

An algorithm is proposed to detect the self intersection using LCFs-based axial deformation. Firstly, we use local coordinate frames for constructing a set of normal planes. Each vertex of the object mesh has one LCF reference of curve point and a normal plane. The normal planes of the LCF may intersect among themselves in the deformation. A line is always formed when two planes are intersected. But, planes intersection method may not be accurate when the tangents of those planes are parallel. Therefore, sphere is used to lying on the normal plane and its radius is the distance between the object mesh to an axial curve. The spheres and the intersection line are then used for detecting the self intersection on the object mesh by comparing the distance from the sphere or the intersection line to the axial curve.

The proposed axial deformation technique uses a set of local coordinate frames to control an axial curve. This provides an accurate and intuitive deformation. Besides, this technique can prevent self intersection in the object mesh. Experimental results show that the proposed method is intuitive and can be used for deforming complex freeform objects.

1.3 Report Organization

This thesis first reviews the current status on axial deformation and the detection of self intersection in deformation. These are given in chapter 2. This chapter is divided into two main sections. The first section provides an overview the Free-Form Deformation, the Extended Free-Form Deformation, the axial deformation, and the axial curve-pair deformation. And we introduce the technique of self intersection detection in section 2.2.

Chapter 3 presents an algorithm for deforming a 3D model using a set of controllable local coordinate frames. An axial space is briefly specified in section 3.1. Then we describe the method to define a local coordinate frame in section 3.2. And we introduce the construction of a set of LCFs on an axial curve. Section 3.3 presents the advantages of using LCF as a deformation tool. Experimental results using a set of LCF to deform a 3D model is presented in section 3.5.

Chapter 4 presents an algorithm for preventing self intersection during object deformation. This chapter has five sections. In section 4.1, we introduce using the LCFs to construct a normal plane on the axial curve. Then a set of planes may be intersected during the deformation in section 4.2. The method for detecting possible intersection between planes constraining LCFs is presented in section 4.3 – 4.6

Chapter 5 gives the conclusions and proposes further works to enhance the performance of the technique.

2. Related Works

2.1 Axial and Free Form Deformation

An efficient and intuitive shape manipulation technique is vital to the shape editing process. Free-Form Deformation (FFD) is a popular technique for 3D modeling and animation. The FFD method allows complex objects to be deformed by adjusting a number of control points. The coordinates of an object point relative to the control lattice remain unchanged even if the control points are moved. An alternative approach called Extended Free-form Deformation (EFFD) [6] adopted a more intuitive interface. EFFD replaced the parallelepiped-shaped control lattice used in the FFD with a lattice of spline control points. It is referred to as lattice-based axial curve deformation and will be discussed in section 2.1.1 and 2.1.2.

An alternative class of deformation method deforms a 3D object by associating its shape components to a skeleton [7,8] and in character modeling and animation. In Computer Vision, axial representation is frequently used to describe the shape and features of a 3D object. Axial representation is an efficient and compact shape descriptor as it preserves the basic topology and geometry of an object. Work related to axial curve deformation can also be classified into polygon-based [9] and point-based [10].

Lazarus et al. [1] presented an axial deformation technique (AxDf) which will be discussed in section 2.1.3. This approach allows deforming a 3D object by adjusting the shape of an axial curve. It allows an object to be deformed by adjusting an axial curve. However, the object cannot be twisted by manipulating the axial curve. K.C. Hui [2] introduced an axial curve-pair based deformation technique for modeling objects. The use of a curve-pair allows the local coordinate frames to be controlled intuitively. However, the expected twist of object may still be obtained when the primary and orientation curves intersect.

2.1.1 The Free-Form Deformation

In Computer-aided geometric design, Free-Form deformation (FFD) is a shape editing tool widely used for modeling and animation. Sederberg and Parry [5] introduced FFD for deforming geometric model in a free-form manner. This technique is based on the use of a control lattice defined with a trivariate Bernstein polynomial. Objects embedded in the lattice can be deformed by manipulating the control points of the lattice. The advantage of FFD is the deformations of the FFD lattice are then automatically passed to the object. However, it cannot provide an intuitive deformation.

2.1.2 The Lattice-based Representation

Free-From Deformation is an intuitive technique for modeling 3D object. It solves the size and the position problem. However, the deformation that can be achieved is restricted by the parallelpiped shape of the lattice. The Extended Free-From Deformation (EFFD) technique uses a non-parallelpipedical lattice to allow more complex shaped deformations. The processes of the EFFD technique are:

1. Define the shape of the lattice
2. Deform and associate a lattice to the object surface
3. Freeze the lattice
4. Deform the surface using the lattices

Although EFFD is quite effective for creating impressions and simple deformations, EEFD is based on the notion of deformation the underlying space of an object lies in. Therefore, the control lattice used to manipulate the underlying space is not directly related to the object being deformed. Some control points may be too closed to the object surface and made an unexpected result to the users.

2.1.3 The Axial Deformation

Lazarus et al. [1] introduced a deformation technique which uses a 3D axis for deforming an existing object. This technique is called the Axial Deformation (AxDf). Suppose a 3D model **O** and an axial curve $\mathbf{c}(t)$, $t_{start} \le t \le t_{end}$ of S is predefined by a set of I points $p_i, i \in [1, I]$. And the axial curve $\mathbf{c}(t)$ is a B-spline curve which can be defined by

$$\mathbf{c}(t) = \sum_{i=0}^{n} \mathbf{q}_i N_{i,k}(t) \tag{1}$$

where $\mathbf{q}_i$, $i \in [1, I]$ are the control points, and $N_{i,k}(t)$ is a B-spline basis function of order k with $t_{start} \le t \le t_{end}$.

There are two main steps in an AxDf. First, each vertex of an object is attached to a point on the axial curve and is specified relative to the local coordinate frame of the curve point. Second, object points are deformed by adjusting the shape of the axial curve since each object point is homologous to a curve point. One limitation of AxDf is that the object cannot be twisted by manipulating the axial curve.

Step 1. Attaching the Vertex to the Axial

Normally, object vertices are attached to their closest points of the axial curve. Different methods are available for computing the closest point on a curve [12,13]. To avoid more than one closest point, adjacent vertices need to be considered, and the attached point is represented by its parametric value on the axial curve.

Step 2. Constructing Local Coordinate Frames on the Axial Curve

Assume C(s), $s \in [0, L]$ is a regular curve, Klok [9] defined the rotation minimizing orthogonal frame t(s), f(s), g(s) is defined along C, so that

$$
\begin{aligned}
t(s) &= C'(s) / \|C'(s)\| \\
f'(s) &= -(C''(s) \cdot f(s)) C'(s) / \|C'(s)\|^2 \\
g'(s) &= -(C''(s) \cdot g(s)) C'(s) / \|C'(s)\|^2
\end{aligned}
\tag{2}
$$

Step 3. Compute Coordinates of the Deformed Vertices

A zone of influence is defined by two radii values, $R_{\min}$ and $R_{\max}$ of two circular cross section along the axis. The radii values can be defined by the users.

Although this technique is suitable for modeling flexible or deformable objects, the limitation is the object cannot be twisted by manipulating the axial curve. Moreover, the local coordinate frame of an axial may cause unexpected twist of an object.

2.1.4 Curve Pair-based Representation

Although the axial deformation technique allows an object to be deformed by manipulating axial curve, unexpected twist may be obtained in a deformation. The lack of control on the local coordinate frame of the curve is the main problem of axial deformation. Hui [2] proposed to use axial curve-pairs to perform freeform design. This is extended for constructing an axial skeletal representation of an object. An axial curve-pair is composed of a primary curve $\mathbf{c}(t)$ and an orientation curve $\mathbf{c}_D(t)$. The primary curve defines the position of a local coordinate frame along the curve. The orientation curve is an approximate offset of the primary curve. The orientation curve defines the orientation of the local coordinate frame. The orientation curve is used to define the normal vector.

Using the axial curve-pair deformation, the control point of the orientation curve is changed automatically when the primary is modified in order to maintain the curves relationship. For example: When the control point of the orientation curve is rotated around the primary curve, the local coordinate frame is oriented automatically. Therefore the shape of an object can be modified easily by adjusting the curve pair.

The axial curve-pair based deformation is suitable for the bending and twisting of object shapes to the expected form. But, one limitation of this approach is that an undesirable twist of the object may be obtained when the primary curve and the orientation curve intersects each other. Figure 1 shows an example of this undesirable twist where distortions are found in the intersecting region.

Figure 1: Self-intersection of an object deformed with a curve-pair.

2.2 Self Intersection Detection

Free-Form Deformation can be classified into lattice-based, point-based, curve-based and skeleton-based. One problem in all forms of spatial deformation is the potential cause of self intersection of an object. Technique for the detection of self-intersection is hence essential for detecting possible self-intersection deformation process.

Self-intersection has been discussed a lot of deformation literature. In the FFD, the lattice provides an indication to experienced users of the degree of deformation. Since the lattices is not always a transparent guide to self-intersection. A lattice with overlapping faces does not necessary imply to self intersection. Conversely though, a lattice without overlap does provided by an FFD lattice is not available in curve and point-based spatial deformation.

There are several methods for detecting self-intersection in the spatial deformation literature. Samoilov and G. Elber [14] introduced an algorithm for eliminating self-intersection in freeform curve metamorphosis. In this approach a homotopy between the two original curves is build. These curves are composited of ruled surface with an appropriate subjective continuous function that causes the curve of homotopy to be self-intersection-free. The best correspondence between the relative parameterizations of the original curves is constructed. Finally, a flipping method is used to eliminate the remaining self-intersections.

Seong et al. [15] presented an algorithm for trimming the local and global self-intersections of offset curves and surfaces. This method is used an analytic distance map between the original curve/surface and its offset. The global and local self intersection region can be identified by solving a bivariate polynomial equation for an offset curve/surface. The limitation of this trimming algorithm is that small self-intersections in the offset curves and surfaces may not be detected in the trimming procedure.

Pekerman et al. [16] introduced an algorithm for detecting and eliminating global self-intersection in freeform curve and surfaces. A bi-normal line criterion is used to detect antipodal points on an intersection loop through solving a system of five multivariate polynomial constraints. Applying an optimization procedure is satisfying the constraints, the location of the antipodal points are flipped to eliminate all self intersection. Later, they [17] introduced several algorithms for self-intersection detection. The method used binormal-line criterion and a simple direct algebraic elimination procedure to obtain a solution for the self-intersection.

3. Axial Deformation with a Set of Controllable LCFs

3.1 Previous Methods

An intuitive deformation tool is essential in design applications where freeform shape editing of an existing object is required. Although the axial deformation method [1] can be used for deforming complex objects, the lack of control on the twist of an axial curve may result in deformed shapes with undesirable distortions. Hui [2] proposed to use axial curve-pairs to perform freeform design. An axial curve-pair is composed of a primary curve and an orientation curve. The orientation curve is an approximate offset of the primary curve. The object shape can be modified by adjusting the curve-pair. In order to maintain the relationship between the two curves, synchronized movement of the control points of the two curves is required. One limitation of this approach is that an undesirable twist of the object may be obtained when the primary curve and the orientation curve intersects each other. Figure 1 shows an example of this undesirable twist where distortions are found in the intersecting region.

Our method is related different from the classical axial deformation technique by allowing users to directly control a set of local coordinate frames (LCF) on the axial curves. The associated model can be deformed and twisted by adjusting the LCFs, while maintaining the continuity and the smoothness of the shape. This avoids undesirable twists as a result of the lack of intuitive control on the LCFs. The continuity and smoothness of the axial curve, and hence the associated geometric shape is maintained by adopting a suitable interpolation scheme to determine the LCF between user defined LCFs. Experiments show that the proposed method is capable of maintaining the shape's smoothness while eliminating undesirable twist and distortion.

3.2 Axial Space

Burtnyl N, Wein M [18] introduced a technique for defining 2D object using an axial curve. The pixels of an image are defined relative to a skeleton curve. When the curve is changed, the positions of the pixels and the shape of the image are changed. Coquillart [6] used the same approach for deforming 3D object using an axial curve. In his method, a 3D object was represented with a polygon mesh. The shape of the polygon mesh is deformed by adjusting the shape of an axial curve.

Considering an axial curve **c** as a B-spline curve defined with a set of control points:

$$\mathbf{c}(t) = \sum_{i=0}^{n} \mathbf{q}_i N_{i,k}(t) \tag{3}$$

where $\mathbf{q}_i$ are the control points, and $N_{i,k}(t)$ is the B-spline basis function of order k with $t_{start} \le t \le t_{end}$.

Given a curve $\mathbf{c}(t)$ and a local coordinate system $\mathbf{l}(t) = \left[\mathbf{l}_x(t), \mathbf{l}_y(t), \mathbf{l}_z(t)\right]$, we can define the axial space of **c**(t) as a 4-dimentional space which is a subset of R^4. A point **p**=(x, y, z), can be represented in the axial space of **c**(t) with a 4-tuple **p**=(t, u, v, w) where t specifies the coordinate frame the point **p** is belonging to, and (u, v, w) are the local coordinates with respect to that coordinate frame.

A function $f : R^4 \rightarrow R^3$ relates the axial space and the Euclidean space:

$$\mathbf{p} = f(t,u,v,w) = \mathbf{c}(t) + u\mathbf{l}_x(t) + v\mathbf{l}_y(t) + w\mathbf{l}_z(t) \tag{4}$$

Given the value of t, we can obtain a curve point $\mathbf{c}(t)$. Since an instance of a local coordinate system $\mathbf{l}(t)$ is defined on a curve point $\mathbf{c}(t)$, by subtracting $\mathbf{c}(t)$ from **p**, where

$\mathbf{p}' = \mathbf{p} - \mathbf{c}(t)$, $\mathbf{p}'$ is then aligned with $\mathbf{l}(t)$, the values of u, v and w can then be computed by projecting $\mathbf{p}'$ to the three axis $\left[\mathbf{l}_x(t), \mathbf{l}_y(t), \mathbf{l}_z(t)\right]$.

$$
\begin{aligned}
u &= \mathbf{p}' \cdot \mathbf{l}_x(t) \\
v &= \mathbf{p}' \cdot \mathbf{l}_y(t) \\
w &= \mathbf{p}' \cdot \mathbf{l}_z(t)
\end{aligned}
\tag{5}
$$

3.3 Definition of Local Coordinate Frame

A popular technique for defining the local coordinate frame of a curve is to use the Frenet Frame. Given a curve **c**(t), the Frenet Frame is defined in terms of the first derivative **c**'(t) and the second derivative **c"**(t) of the curve:

$$\mathbf{T}(t) = \frac{\mathbf{c}'(t)}{\|\mathbf{c}'(t)\|}$$
$$\mathbf{B}(t) = \frac{\mathbf{c}'(t) \times \mathbf{c}''(t)}{\|\mathbf{c}'(t) \times \mathbf{c}''(t)\|}$$
$$\mathbf{N}(t) = \mathbf{B}(t) \times \mathbf{T}(t) \qquad (6)$$

These vectors are also referred to as the tangent **T**, normal **N**, and binormal **B** of **c**(t). Frenet Frame sufferers from two major limitations: Firstly, the frame is completely defined by the curve **c**(t), users can only adjust it via modifying the curve. There is no direct control on the orientation of the frame. Secondly, the second derivative may not be always available, **c**"(t) may vanish when the curve contains a straight segment which makes the frame undefined.

Another similar approach proposed by Lossing and Eshleman[19] uses an additional direction curve **d**(t) to define the normal **n**(t) of the curve:

$$\mathbf{n}(t) = \mathbf{d}(t) \times \mathbf{c}'(t) \qquad (7)$$

Although it allows the control of the orientation of the curve through controlling **d**(t), it is still hard to control it intuitively since this extra curve **d**(t) is independent of the primary curve.

One recent approach proposed by K.C Hui [2] modifies the Lossing and Eshleman's approach [19] by using an additional curve called the orientation curve. This orientation curve forms an axial-pair together with the primary curve. In contrast to the Lossing and Eshleman's method where **d**(t) is independent of the primary curve **c**(t), the orientation

curve is an approximate offset of the primary curve **c**(t). The local coordinate frame is defined by:

$$\mathbf{l}_z(t) = \frac{\mathbf{c}'(t)}{|\mathbf{c}'(t)|}$$

$$\mathbf{l}_x(t) = \mathbf{n}(t) \times \mathbf{l}_z(t) \qquad \text{where} \quad \mathbf{n}(t) = \frac{\mathbf{c}_D(t) - \mathbf{c}(t)}{|\mathbf{c}_D(t) - \mathbf{c}(t)|}$$

$$\mathbf{l}_y(t) = \mathbf{l}_z(t) \times \mathbf{l}_x(t) \tag{8}$$

and $|\mathbf{c}_D(t) - \mathbf{c}(t)|$ is less than a user defined values, i.e. $|\mathbf{c}_D(t) - \mathbf{c}(t)| \leq r$

The primary and the orientation curves contain the same number of control points which are used to control the shapes of the curve-pair. His research proposed an algorithm to synchronize the modification of the two curves. It is essential to maintain the relationship between the two curves, a synchronized movement of the control points of two curves is required. There is one limitation when the primary curve and the orientation curve are intersecting to each other, an undesirable twist may be resulted of the object.

3.4 Construction of Axial Curve with LCFs

Beside Frenet Frame, there exists others approach for defining the coordinate frames on a curve, such as rotation minimizing method [20], tangent reference method [21] and normal projection method [22]. However, these methods may not provide a satisfactory output when the curvature of the curve is high. In this research, the normal projection method is adopted. The normal vector is constructed by using the Axis Projection method. The normal N(s) is defined by projecting the Y-Axis to the tangent plane.

and the local coordinate frame at t is given by

$$\mathbf{l}_z(t) = \frac{\mathbf{c}'(t)}{|\mathbf{c}'(t)|}$$

$$\mathbf{l}_y(t) = \frac{\mathbf{N}(t)}{|\mathbf{N}(t)|} \qquad \text{where } \mathbf{N}(t) = \mathbf{Y} - (\mathbf{Y} \cdot \mathbf{l}_z(t))\mathbf{l}_z(t) \qquad (9)$$

$$\mathbf{l}_x(t) = \mathbf{l}_y(t) \times \mathbf{l}_z(t)$$

However, the normal may not be defined if the tangent plane is perpendicular to the Y-axis. This situation happens when the tangent vector is parallel to the Y-axis, such that the dot product of the Y-axis and the tangent vector is zero. This means the normal vector is parallel to the tangent vector. This is impossible, and the normal vector is define as (-1, 0, 0) when the axial curve is a vertical line.

3.5 Point Projection Method

This section describes the point projection method which associates each vertex of an object to the axial curve, $\mathbf{c}(t)$. One straight forward method is to associate vertices to their closest point the axial curve.

Given that knot sequence

$$[t_0, t_1, \cdots t_i, \cdots t_{N-1}], \tag{10}$$

A direct linear search is used. The time complexity of the process is O(N), where N is the number of t_i. This is computational expensive when the N is large. In order to reduce the time complexity, we adopt a technique called the nearest neighbor search [17], which reduce the searching time to O(logN). Given the knots t_i, a set of discrete curve points is obtained. The nearest neighbor search can then be performed on the set of curve points. Based on our experiments, with the use of the nearest neighbor search, real time response is attained even with large point set in the order of thousands. However, one limitation of the above method is that the value of t obtained may not be optimal if the optimal value is not in the set of t_i, and hence only an approximation can be obtained.

Although the approximation is usually enough for freeform design, one may desire to have an optimal solution for the value of t. In this case, we propose to use a modified version of the method introduced by Wang et al. [18]. This method combines a quadratic minimization problem with the Newton's method. Given three initial values of t, quadratic minimization converges slowly but it is good for refining coarse estimates. On the other side, the Newton's method converges quickly with a good initial value, but provides a rough estimation.

Our method replaces the quadratic minimization process by the above discrete approximation. The first reason is that the quadratic minimization method suffers from

the problem of local minima, and may converge to a wrong estimate for the Newton's method. Instead of a local minimum, the discrete approximation method always provides a solution that approximates the global optimal solution. The second reason is that no initial guess is required for the discrete approximation while it is essential for the quadratic minimization. A bad initial guess may cause the algorithm to fail.

3.5.1 Optimum Reference Axial Curve Point

Consider an object O, and a point **p** in O, where **p**=(x, y, z), is in 3D Euclidean space. In order to associate **p** to an axial curve, **p** has to be transformed into the 4D axial space in the form of **p**=(t, u, v, w). As mentioned in section 2.1, once t is found, u, v and w can be determined easily. The parameter t is defined as the value such that the curve point **c**(t) is closest to the point **p**.

Our proposed two-step algorithm to find t:

1. Discrete approximation: Given a discrete set of curve points, and an object point **p**, the closest point **c*** to **p** is obtained using the nearest neighbor search. The approximate optimal t^* is the value of t associated with **c***.

2. Newton's Method: Given a point **p**, and a curve point **c**(t), the square of the distance between them is:

$$D(t) = (c_x(t) - x)^2 + (c_y(t) - y)^2 + (c_z(t) - z)^2 \tag{11}$$

The Newton's method attempts to find the optimal value t^* by minimizing D(t). It leads to the following iterative equation:

$$t^{*,n+1} = t^{*,n} + \frac{D'(t^{*,n})}{D''(t^{*,n})}, \quad n = 0,1,2,\ldots \tag{12}$$

with the above method, an initial value $t^{*,0}$ is obtained from step 1. This value is usually close to the optimal value t^*. If a good initial guess is obtained after step 1, this iterative process can converge quickly - usually within 3 iterations. Moreover, the local minimum problem which is usually found in quadratic minimization is also avoided by the good initial value.

3.6 Advantages using LCFs in Axial Deformation

Axial Deformation is a popular technique in modeling and animation. The shape of an object can be modified by adjusting the shape of an axial curve. However, there is no control on the twist of the axial curve. In this research, local coordinate frame is used to control the shape of the axial curve.

3.6.1 Deformation with a Smooth Interpolation

After attaching an object to an axial curve, each point of the object is associated with its closest point on the axial curve, and the local coordinate frame (LCF) at the corresponding curve point is regarded as the orientation frame reference. When the LCF $(\mathbf{l}_x(t),\mathbf{l}_y(t),\mathbf{l}_z(t))$ is transformed to $(\mathbf{l}^*_x(t),\mathbf{l}^*_y(t),\mathbf{l}^*_z(t))$ with a rotation transformation $\mathbf{R}$, an attached object point $\mathbf{p}_a$ will be moved to a new position $\mathbf{p}_a^*$ with the same transformation relative to the LCF.

$$\begin{aligned}
\mathbf{l}^*_x(t) &= \mathbf{l}_x(t)\mathbf{R} \\
\mathbf{l}^*_y(t) &= \mathbf{l}_y(t)\mathbf{R} \\
\mathbf{l}^*_z(t) &= \mathbf{l}_z(t)\mathbf{R} \\
\mathbf{p}_a^* &= \mathbf{p}_a\mathbf{R}(\theta(t))
\end{aligned} \tag{13}$$

To maintain the smoothness of the object, when one particular local coordinate frame is modified, an interpolation scheme is adopted to propagate the changes to the other LCF. Assume two LCFs are rotated for an angle θ_0 and θ_1 respectively, an algorithm is presented to determine the rotation angle $\theta(t)$ of the LCF at any t between the modified LCFs.

In our system, the axial curve is approximated with a set of linear curve segment. Each segment is constructed with consecutive LCFs. Since the knots may not be

positioned evenly, the rotation angle can be computed based on the arc length $l(t)$ and the total arc length $l(n)$ of the curve.

$$\sigma = \frac{l(t)}{l(n)},$$
$$\theta_t = (1-\sigma)\theta_0 + \sigma\theta_1 \tag{14}$$

However, the rate of change of θ_t may not be continuous at the knot t. Consider the first derivative of the above equation, the rate of change of θ_t depends only on the angle difference (or twist angle), i.e. $\theta_t' = (\theta_1 - \theta_0)\sigma'$. Hence, when two consecutive segments are having large difference in their twist angles, a sharp turn will be obtained as shown in Figure 2.

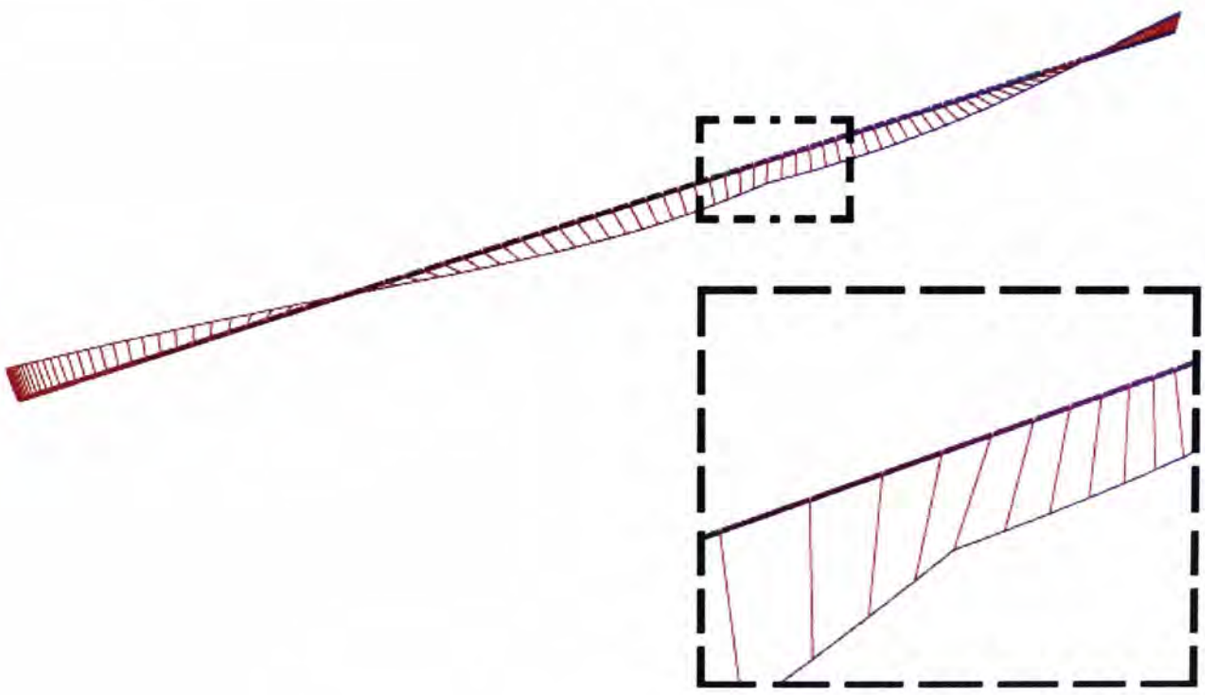

Figure 2: The linear interpolation

In order to maintain a smooth change in the rotation angle, one possible solution is to use polynomial interpolation. In general an $n-1$ degree polynomial can be used to interpolate n data points. Given 4 angles $(\theta_0, \theta_1, \theta_2, \theta_3)$, the interpolated angle can be obtained with the following equation.

$$\theta(t) = (1-t)^3\theta_0 + 3t(1-t)^2\theta_1 + 3t^2(1-t)\theta_2 + t^3\theta_3 \tag{15}$$

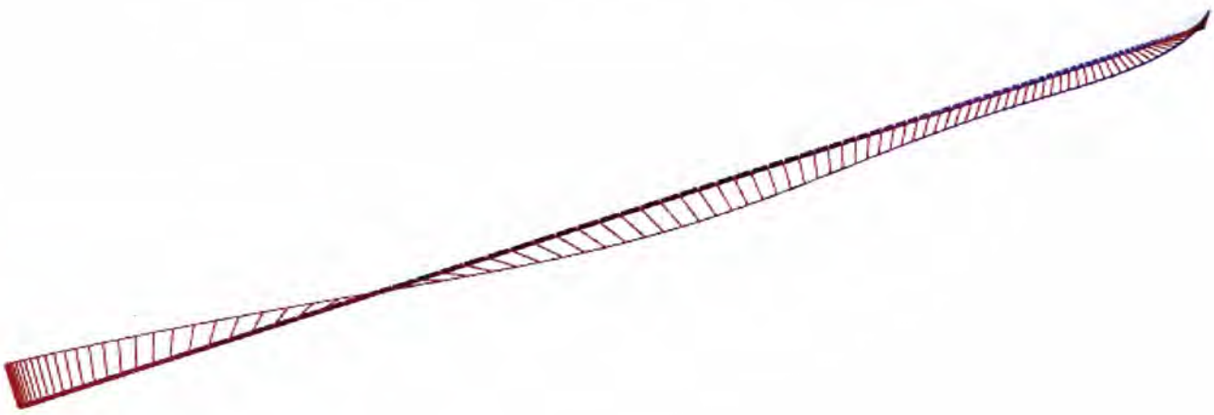

Figure 3: The cubic interpolation

Since the cubic interpolation is twice continuously differentiable, it ensures that the change of rotation angle is smooth and continuous. However, comparing to linear interpolation, polynomial interpolation is relatively computational expensive. Therefore, the cosine interpolation [19] is adopted in this research which is defined as follows:

$$\sigma_1 = \frac{l(t)}{l(n)}, \qquad \text{where } 0 \le \sigma_1 \le 1$$

$$\sigma_2 = \frac{1-\cos(\sigma_1\pi)}{2}, \qquad \text{where } 0 \le \sigma_2 \le 1$$

$$\theta_t = (1-\sigma_2)\theta_0 + \sigma_2\theta_1 \tag{16}$$

One important advantage of using cosine interpolation is that the angle is guaranteed to change smoothly especially at the junction point of two segments. Consider the rate of change of the rotation angle, that is, the first derivative of θ_t:

$$\theta_t' = (\theta_1 - \theta_0)\sigma_2' = \frac{1}{2}(\theta_1 - \theta_0)\pi\sigma_1' \sin(\sigma_1\pi) \tag{17}$$

We can observe that at the two end points of each segment, where the values of σ_1 are 0 and 1, the rate of change is always zero. In other words, at the connection point of each segment, the rate of change is always zero. The angle change is thus always smooth on the whole curve. Therefore, a suitable cosine function serves to provide a smooth transition between adjacent segments as shown in Figure 4.

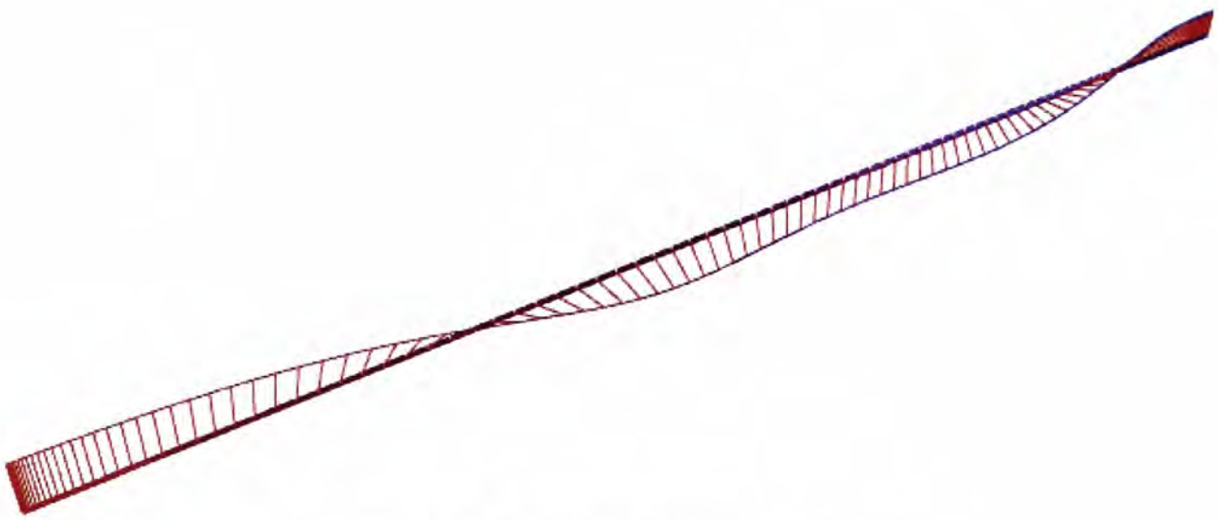

Figure 4: The cosine interpolation

3.6.2 Used in Closed-curve Deformation

In previous sections, an open curve is assumed for the axial curve. Given the control points of the curve as $q_0, q_1, \cdots, q_{epnum-1}$, a closed b-spline curve can be formed by setting the first 3 control points to be the same as the last 3 control points of the axial curve.

$$\begin{aligned} q_0 &= q_{epnum-1} \\ q_1 &= q_{epnum-2} \\ q_2 &= q_{epnum-3} \end{aligned} \tag{18}$$

Therefore, the new control points sequence is updated to become $q_0, q_1, \cdots, q_{epnum-3}$. And the local coordinate frames are built by the normal projection method which is discussed in section 3.4.

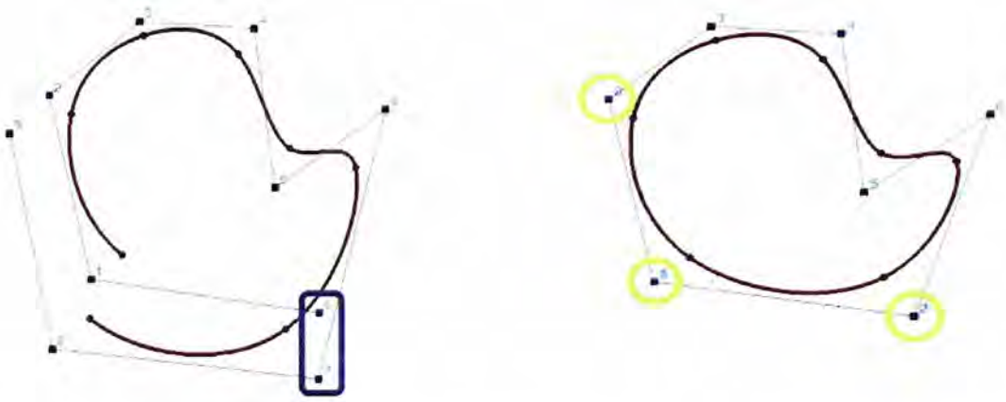

Figure 5: Construction of closed axial curve: (a) the original control points; (b) merging the first and last three control points to form a closed curve.

Twisting in a closed curve can also be performed by using the proposed axial deformation method. Assume the rod is twisted by a set of controllable local coordinate frames. And then the rod is bent to from a ring. The first three and last three control points of the axial curve are merged together. The shape of the rod is adjusted according to the closed loop form of the axial curve. As a result, a ring shape rod is formed by a

closed axial curve. The local coordinate frames at the front and end of the curve are adjusted in order to provide a smooth segment at the connection point of the axial curve. Besides, the circular object can be twisted naturally.

3.6.3 Object Representation with a Hierarchy of the Axial Curve

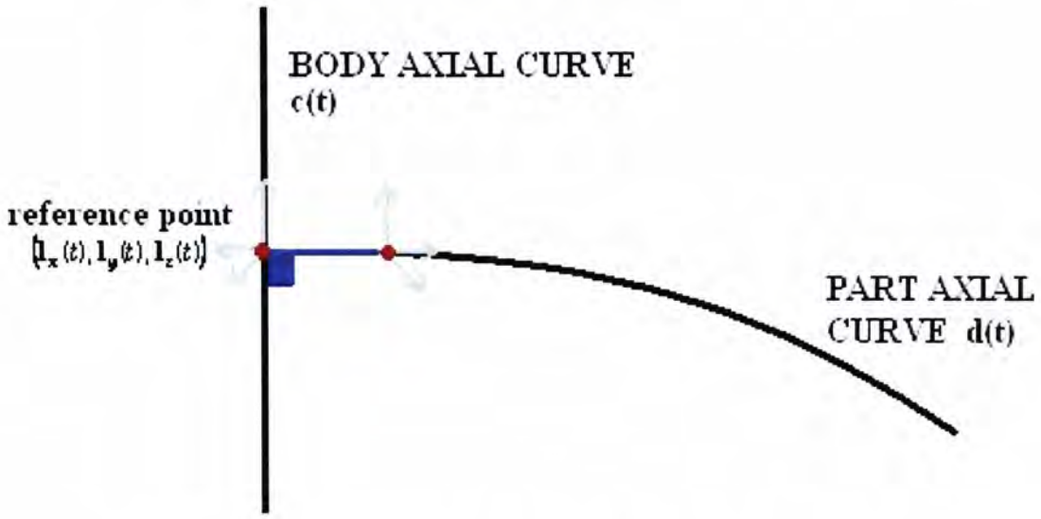

Figure 6: The orientation of a LCF of a hierarchy axial

Assume **c**(t) is an axial curve of the main body of an object and **d**(t) is an axial curve of a component of the object, **d**(t) is attached to a reference point on the axial curve $\mathbf{c}(t)$ that is the closest to the end point of **d**(t). Given the LCF $(\mathbf{l}_x(t),\mathbf{l}_y(t),\mathbf{l}_z(t))$ at the reference point of the axial curve $\mathbf{c}(t)$ is transformed to $(\mathbf{l}^*_x(t),\mathbf{l}^*_y(t),\mathbf{l}^*_z(t))$ with a rotation transformation R, then the rotation transformation will be applied to the local coordinate frames of the axial curve **d**(t) and .

$$\begin{aligned}\mathbf{l}^*_x(t) &= \mathbf{l}_x(t)\mathbf{R}\\ \mathbf{l}^*_y(t) &= \mathbf{l}_y(t)\mathbf{R}\\ \mathbf{l}^*_z(t) &= \mathbf{l}_z(t)\mathbf{R}\end{aligned} \tag{19}$$

3.6.4 Application in Soft Object Deformation

In general, the animations of soft objects can be controlled by an axial curve, such as fish swimming, snake movement and the swing of tree. Lazarus et al. [1] applied Axial Deformation technique to deform the shape of soft objects. In Axial deformation, the shape of an object is adjusted according to the shape of an axial curve. When the local coordinate frame on the axial curve is changed, the associated object points will be transformed to new positions corresponding to the new orientation of the local coordinate frame and its reference curve point on the axial curve.

In the proposed approach, users can adjust the orientation of a local coordinate frame and other local coordinate frames will be adjusted with an interpolation scheme in order to maintain the smoothness and continuity of the orientation.

3.7 Experiments and Results

An experimental system was implemented in a standard personal computer. The surface mesh of an object can be constructed using commercial CAD systems, such as Maya, 3D Studio Max etc. The mesh data file of the object is then exported from the CAD system and passed into our experimental system. The axial curve is constructed interactively by specifying a number of data points. The system then constructs a set of local coordinate frames automatically by using Normal Plane Projection method. Imported object can be attached to the axial curve which forms an axial representation of the object surfaces. Figure 7 shows a S- shape ribbon constructed with the experimental system. An axial curve is attached to the ribbon. The shape of the axial curve can be modified by adjusting the LCFs of the curve. The corresponding positions of the other frames are adjusted in accordance automatically.

A ribbon is modified to form a butterfly shape as shown in Figure 8. The deformation is obtained by orientating the local coordinate frames of the axial curve. Figure 9 shows a twisted ribbon. The local coordinate frame at the mid-point of the curve is rotated 180 degrees to obtain the twist, and the in-between frames are interpolated by using Cosine interpolation. The result shows that no self intersection occurs on the ribbon.

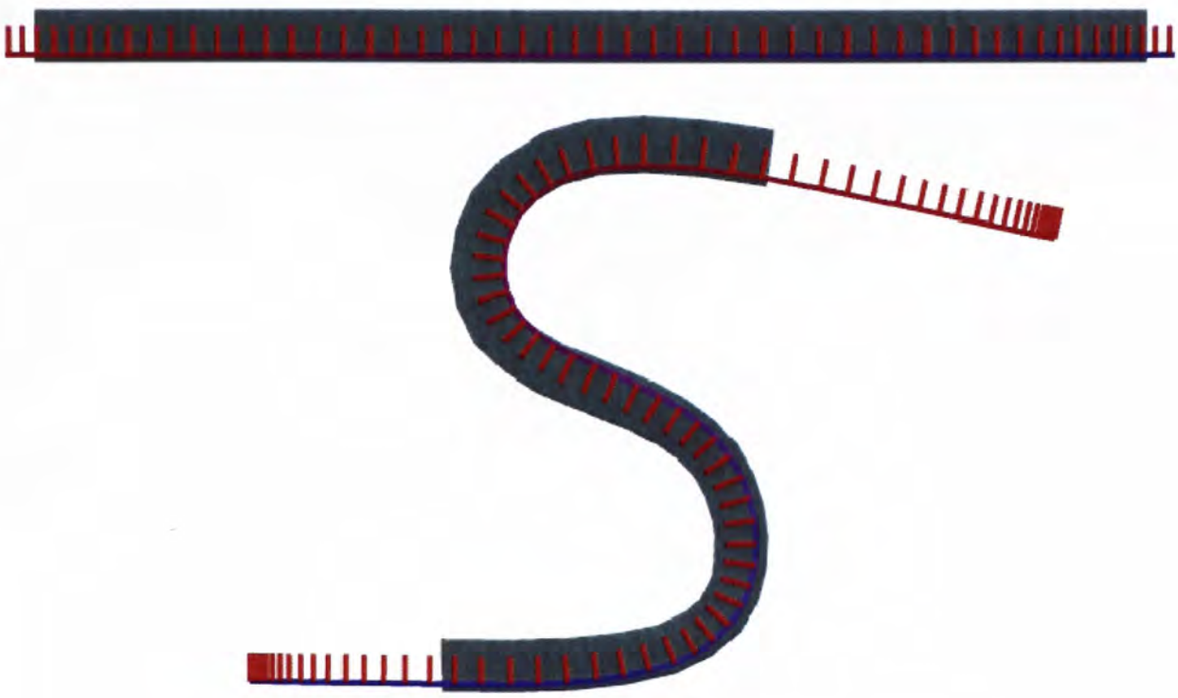

Figure 7: A ribbon is bent to form a S-shape

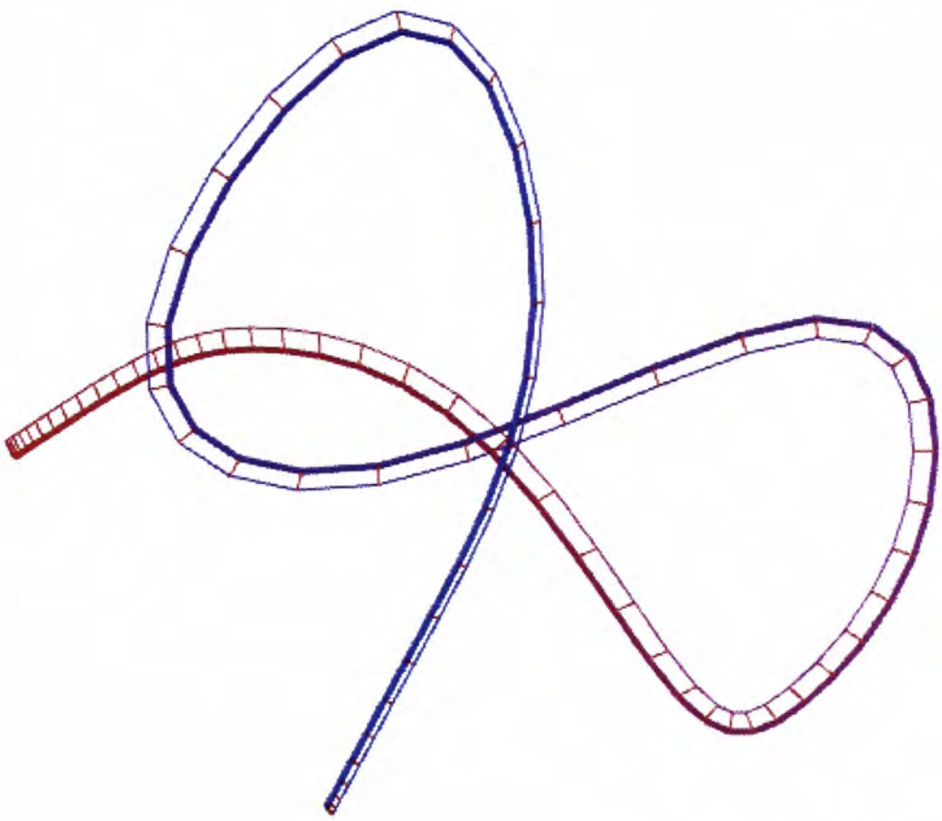

Figure 8: Deforming a ribbon to a butterfly shape

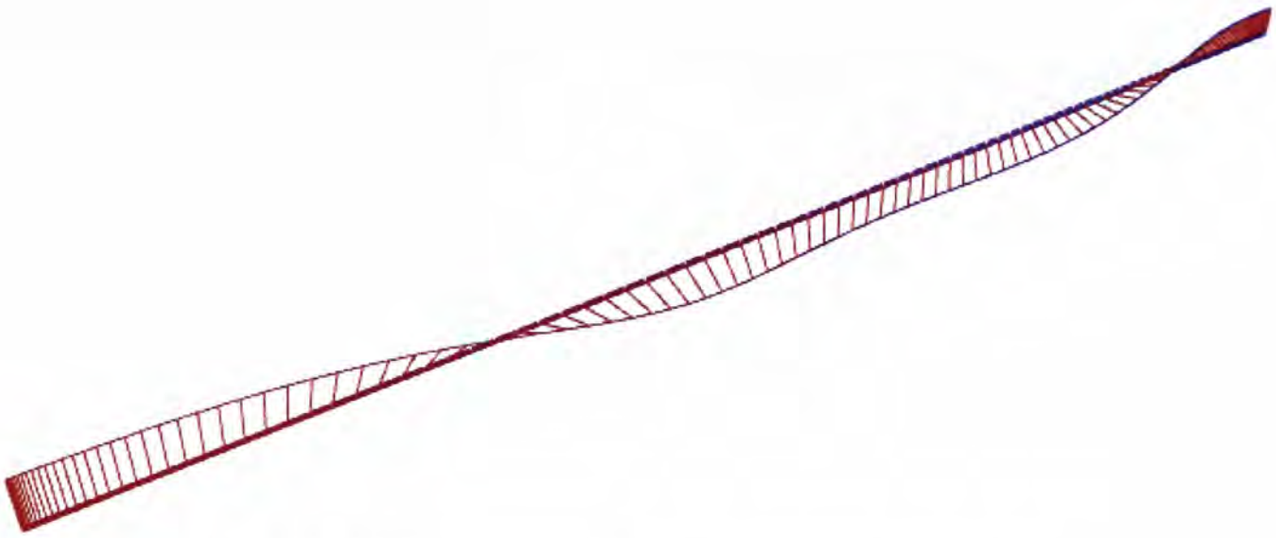

Figure 9: A ribbon twisted 180 degrees, no intersection occurs.

Figure 10 shows the deformation of a set of leave. The initial structure of the leave is shown in Figure 10(a). The leave is attached to an axial curve (Figure 10(b)). The local coordinate frames are deformed to obtain a S-shape leave as shown in Figure 10(c). The axial curve is further twisted to obtain the shape as shown in Figure 10(d).

Figure 11 illustrates the design of the motions of a dolphin. The original posture of the dolphin is shown in Figure 11(a). An axial curve is defined through the body of the dolphin as shown in Figure 11(b). In Figure 11(c), the tail of the dolphin is bent and moved upwards. Different postures can be obtained by orientating the axial curve. Figure 11(d) shows the tail of the dolphin being twisted 45 degrees about the axial curve.

Figure 12(a) shows the construction of an octopus. The head of the octopus is attached to the main axial curve. Each arm is attached to its own axial curve. The arms are deformed to obtain the shape in Figure 12(b).

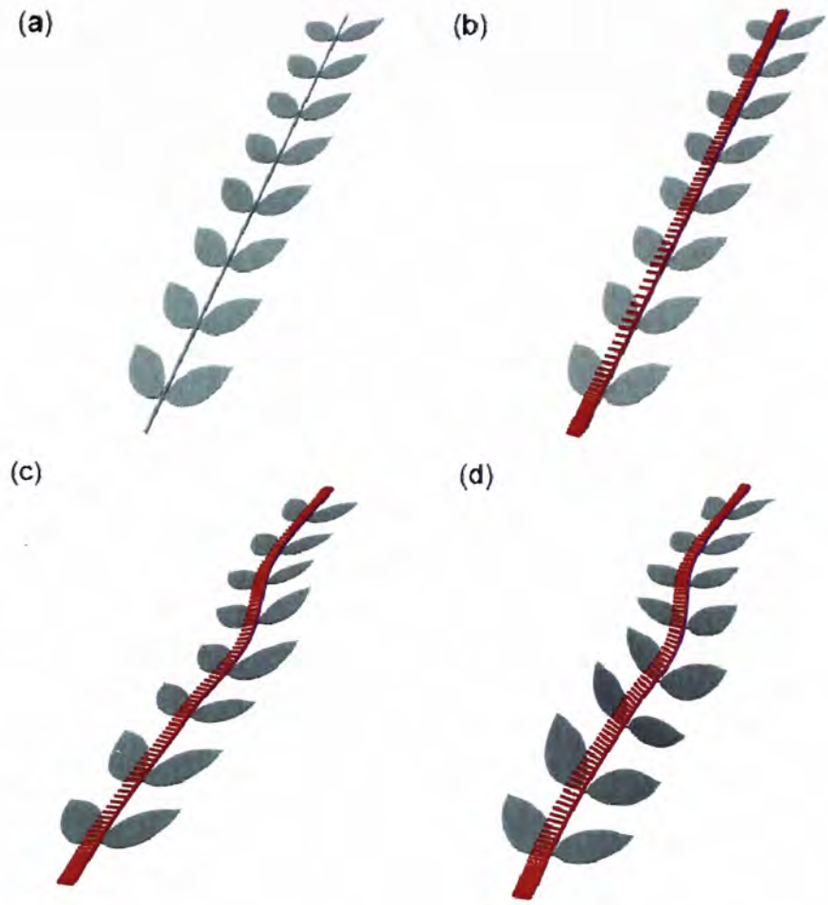

Figure 10: Deformation of a leaf: (a) the original imported leaf; (b) the leaf stem defined with an axial curve; (c) result of deforming the leaf stems; (d) result of twisting the deformed stem to form a new shape.

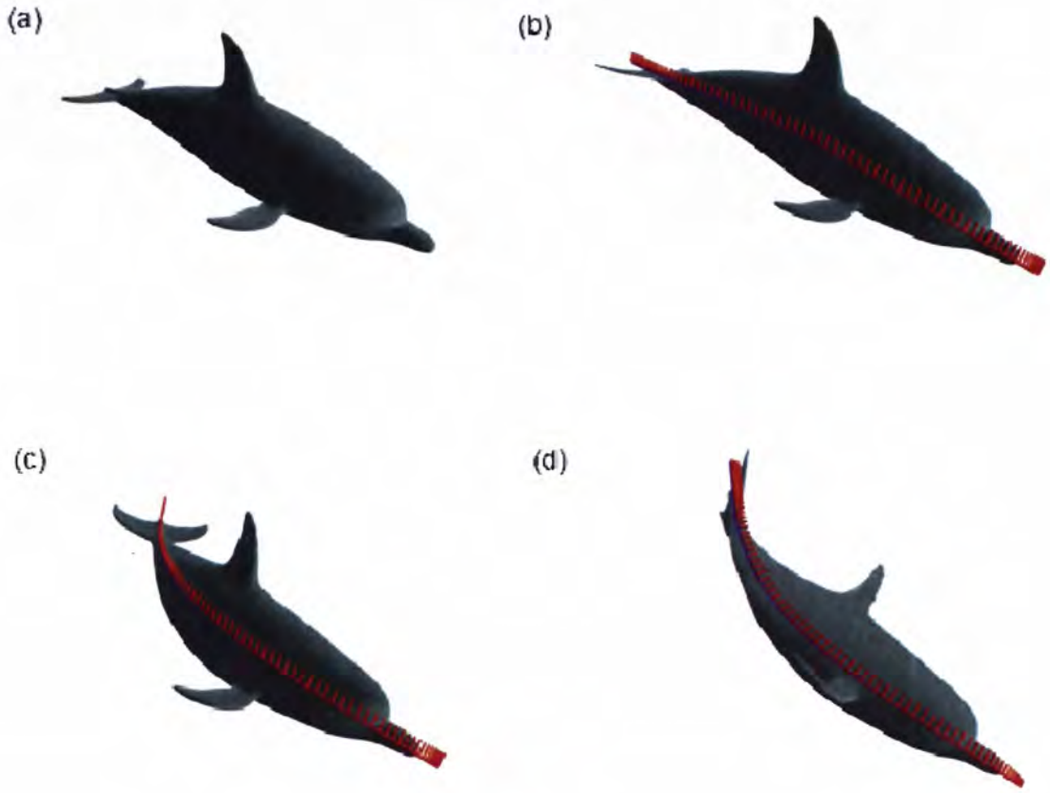

Figure 11: Deformation of a dolphin: (a) the original dolphin; (b) the dolphin modified by a single axial curve; (c) the tail of the dolphin is bent and moved upwards; (d) twisting the tail for 45 degrees.

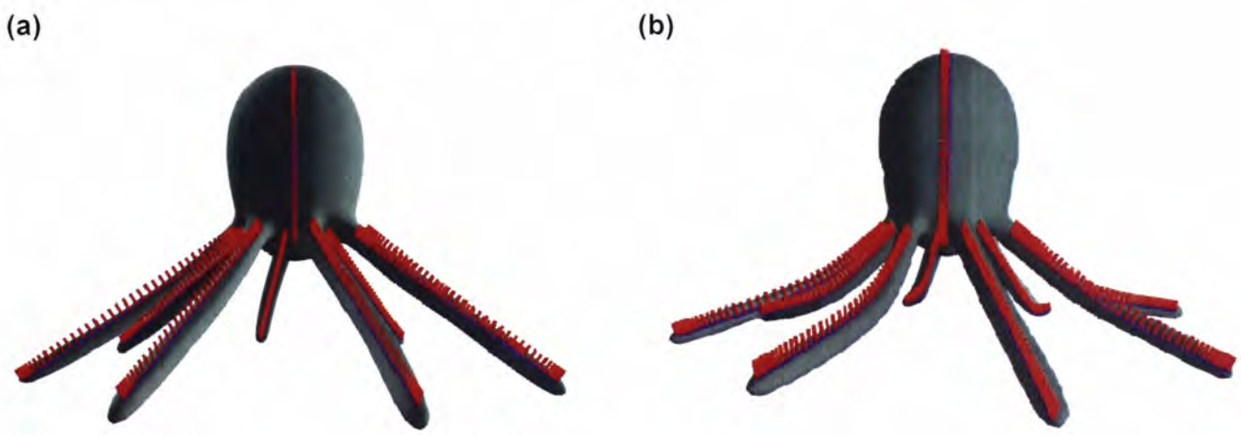

Figure 12: Deformation of an octopus: (a) the original octopus;
(b) the octopus with bent arm.

4. Self Intersection Detection of an Axial Curve with LCFs

Self-intersection is a fundamental problem in the shape editing of geometric models using axial deformation. In the previous chapter, we presented an axial representation of a geometric object using a set of local coordinate frames (LCFs) defined on an axial curve, and the object can then be deformed by adjusting the attached axial curve with a set of controllable LCFs.

To a certain extent, self-intersection can be largely and easily detected by visual inspection. And sometimes minor editing on the original model can be done to solve the self-intersection problem. However, expensive redesign or working around may be required. And minor self intersections may still remain in the CAD model. Therefore, a system to locate those regions with self-intersection is desirable; in our experimental system, those areas with self-intersection are highlighted in order to help reducing the problems when the CAD model is being deformed.

This chapter focuses on developing an algorithm for detecting self intersection using a set of local coordinate frame on an axial curve

4.1 Related Works

Self-intersection is a serious problem in Free-form Deformation. Borrel and Repoport [23] presented a model for producing controlled spatial deformation, Simple Constrained Deformation (ScoDef). The users defined a set of constraint points by giving a desired displacement and radius of influence for each. However, self-intersection problem has to be identified and corrected by the users.

In the Free-Form Deformation, James [24] proposed an injectivity (one-to-one) test for preventing self intersection under lattice-based spatial deformation. To solve the problem of injectivity test which is accurate but computation expensive, a technique called the Directly Manipulated Free-Form Deformation (DMFFD) is proposed. It composes many small injective deformations. The DMFFD can prevent self-intersection under a range of possible deformation without sacrificing the speed of the approximate test.

Lazarus et al. [1] proposed using an axial curve to deform the complex geometry. There are two types of self-intersections: local self intersection and global self intersection. In local self intersection, the axis curvature is the main mainly causes of the self intersection. Global self intersection is mainly caused by a narrow space between two different sections of the axis curve.

Ji et al. [25] presented a noval axial deformation algorithm without local self-intersection. The algorithm used the curvature of an axial curve to detect the self intersection. When the curvature is too large, the control points of the axis are relocated and the shape of the axial curve is modified. Therefore, the local self intersection can be avoided by changing the curvature of the curve.

Lee et al. [26] presented a new approach to realistic hand modeling. Self intersections may occur in a hand model when the fingers touch the palm. A simple approximation of the hand is constructed using spheres and planes. This allows self intersection around the

knuckles to be detected using cross-sectional plane at each joint, and sphere-sphere intersections.

Yoshizawa et al. [27] proposed a mesh evolution scheme for avoiding self intersections. The original mesh is then represented as the set of displacements applied to the vertices of the improved (smoothed) skeletal mesh. The mesh deformation process is combined from deformations of the smoothed skeletal mesh and the displacement field. And the local and global self-intersections of the deformed mesh can be removed by the mesh evolutions method.

Charlie et al. [28] presented a view-dependent method to perform object deformation. In the context of self intersection, their approach adopted the Correa et al. [29] model. The warp distorts the model in two dimensions to match artwork from a given camera perspective. With reference to the method of Correa et al., they compute the tangent curve to avoid the self intersection.

4.2 Algorithms for Solving Self-intersection Problem with a set of LCFs

In axial deformation, self-intersection may lead to undesirable distortion of the object mesh may be broken during deformation. Ji et al. [25] proposed to relocate the control points of an axial curve to avoid the self-intersection problem. The problem is that the relocated control point cannot be predicted by the users. The deformation is hence not intuitive. The proposed method displays the self -intersection region to the users, so that the user can control the shape of an object all the time without self-intersection.

In this chapter, we propose an algorithm for detecting self-intersection on object mesh. The algorithm uses a set of normal planes constructed at the position of the local coordinate frames on the axial curve. By observing that two normal planes overlap when self-intersection occurs, this algorithm can accurately detect local self-intersection on the mesh. Unfortunately, self-intersection cannot be detected, for example, when the object is bent into a U shape or S shape, such that the planes are parallel. In the proposed algorithm, an enclosing sphere is used to detect the overlapping region in the deformation. The radius of the sphere is the distance between the vertex on the object mesh and the corresponding axial curve point, and its center is the reference curve point. When the two planes intersect, a line is formed. The intersection line also cuts the spheres when there is self-intersection. Our algorithm can detect both local and global self-intersection on the object. Figure 13 shows that a self-intersection occurs when the intersection line passes through both spheres.

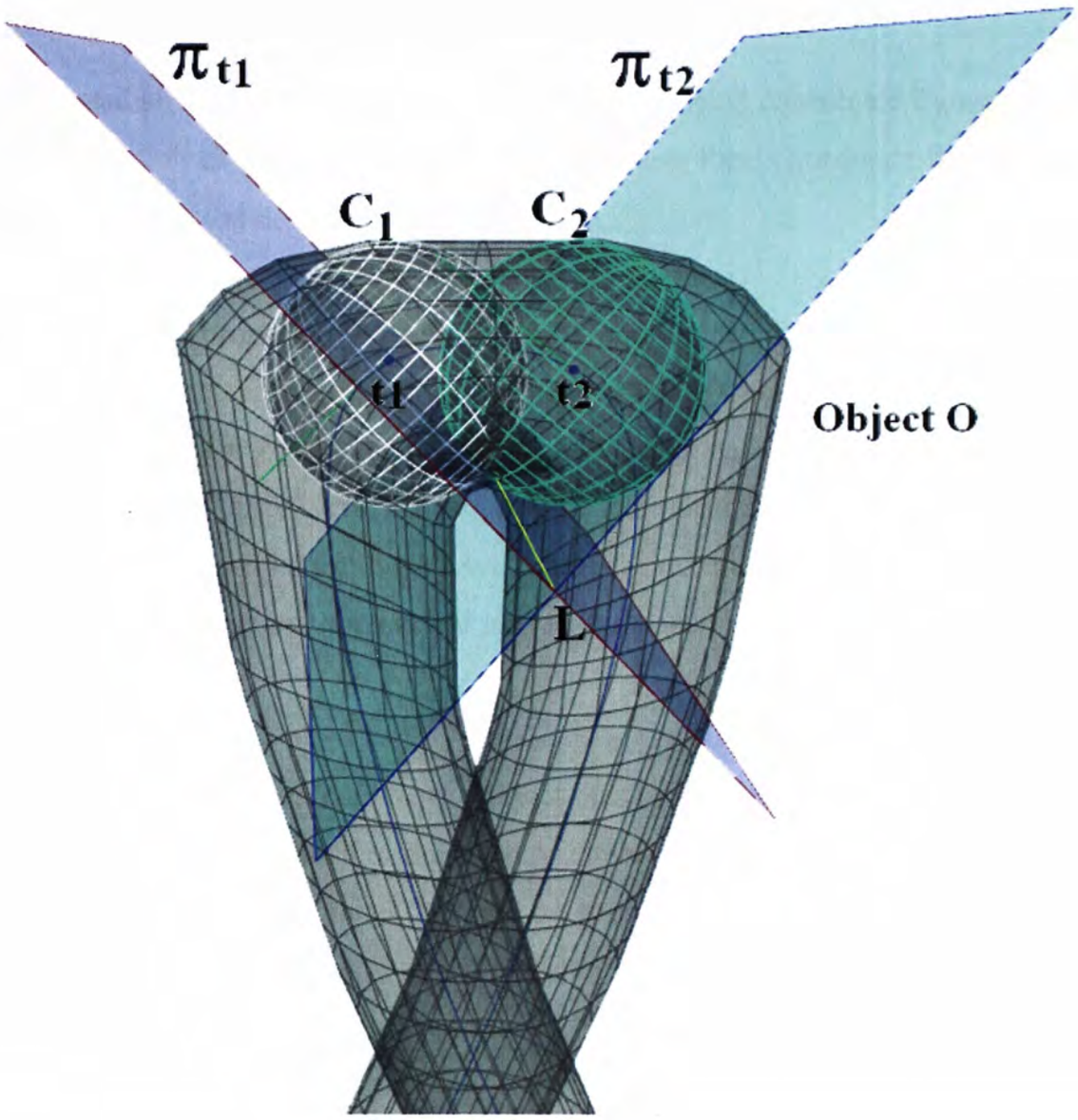

Figure 13: Detecting self-intersection on an object surface by the proposed algorithm.

4.2.1 The Intersection of Two Planes

In the proposed an axial deformation technique, a set of local coordinate frames is used to specify the location of the mesh vertices. Besides, these local coordinate frames can also act as a self-intersection detector.

In this thesis, we proposed to use the local coordinate frames to detect the self-intersection on an object mesh. A set of local coordinate frames are used to construct a set of normal planes on an axial curve. Each mesh vertex has a reference local coordinate frame. As a result, each vertex has a reference normal plane. The normal plane construction is discussed in section 4.2.1.1. When the normal planes are not parallel, they must be intersection between the normal planes. It means possible self-intersection on the object.

In some cases, no overlap occurs on the object mesh although the planes are intersecting each other. This is because the planes may intersect in region outside of the object. Therefore, the sizes of the planes have to be considered in order to have an accurate detection. However, in general, when two planes are not parallel, self-intersection may occur. A line is always formed when two planes are intersecting as will be discussed in section 4.2.1.2.

4.2.1.1 Constructing the Normal Plane

In the detection of self-intersection, the local coordinate frames are used to construct a set of normal planes on the axial curve. When the LCFs on the axial curve are modified, the normal planes are automatically updated. The mesh of the object is deformed according to the position and orientation of the LCFs. Since, the LCFs are associated with the object mesh, each vertex of the object is associated with the normal plane of its corresponding LCF.

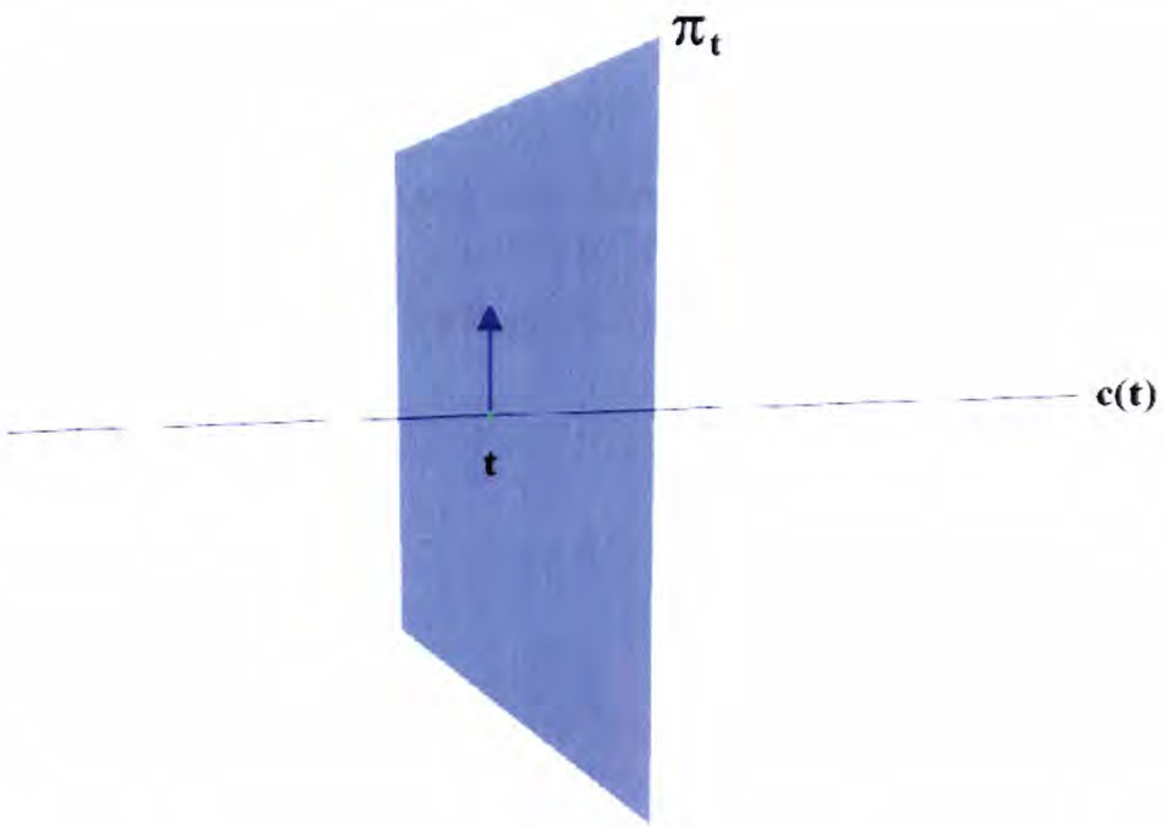

Figure 14: A normal plane is constructed at a curve point t on an axial curve c

Assume that **p** is the vector representing the position of a point on the plane, and let **n** be a nonzero normal vector lying on the plane. The center of the normal plane is at its reference curve point. The normal plane $\prod$ containing the normal vector $\mathbf{n} = (a, b, c)$ that passes through an axial curve point $\mathbf{x_0} = (x_0, y_0, z_0)$ is

$$\mathbf{n} \cdot (\mathbf{x} - \mathbf{x}_0) = 0 \tag{20}$$

where $\mathbf{x} = (x, y, z)$. This gives the general equation of a plane,

$$ax + by + cz + d = 0 \tag{21}$$

where $d = -ax_0 - by_0 - cz_0$

There is intersection between two planes with plane normals $\mathbf{n}_i, \mathbf{n}_j$ if

$$\mathbf{n}_i \times \mathbf{n}_j \neq \mathbf{0} \qquad i \neq j, i, j = 1,2,3,..... \tag{22}$$

Normally the planes must cross each other when the normals of the two planes are not parallel. Some regions of the mesh do not need to be considered in subsequent processes if the normal planes are parallel in that region.

4.2.1.2 A Line Formed by the Intersection of Two Planes

Intersection between the normal plane may occur when $\mathbf{n}_i \times \mathbf{n}_j$ is not zero. Figure 16 shows an intersection line between two planes.

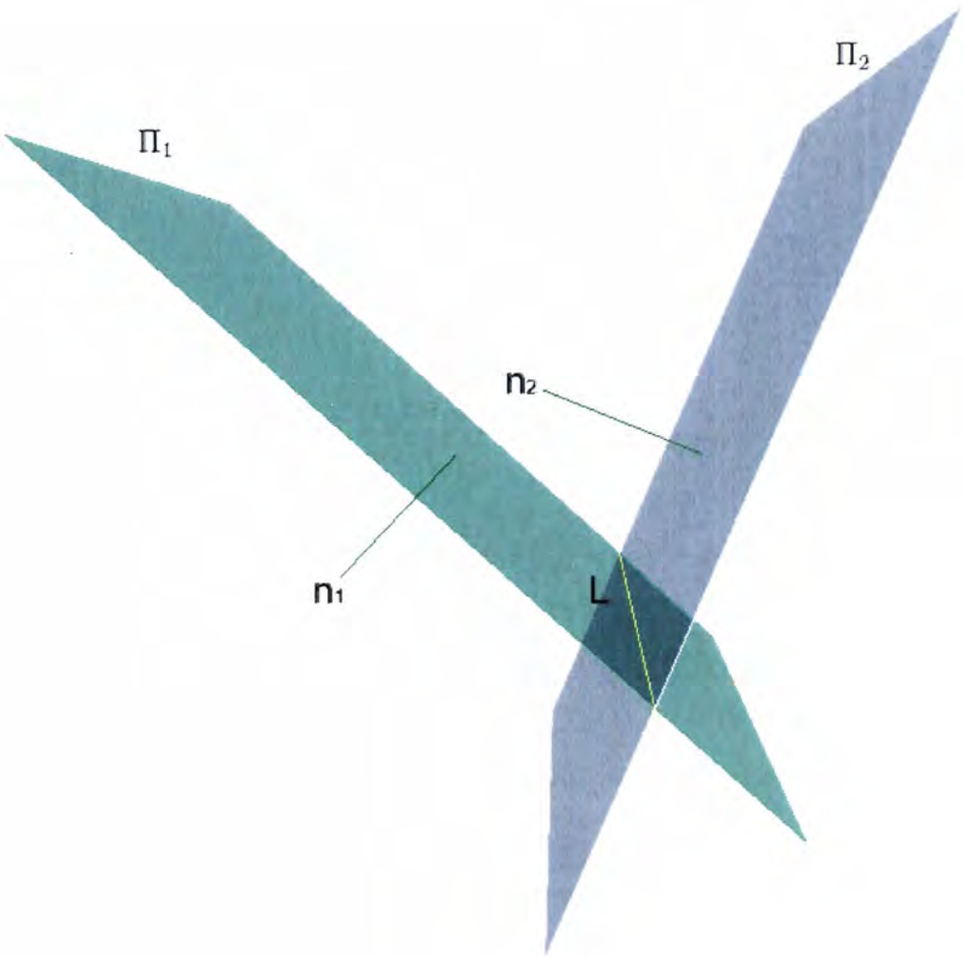

Figure 15: A line is formed when two planes are intersecting each other.

Consider there are two planes Π_i, Π_j with normal $\mathbf{n_1}$, $\mathbf{n_2}$ respectively. The equation of the intersection line can be obtained by the following steps.

Plane equation of Π_1

$$a_1 x + b_1 y + c_1 z + d_1 = 0 \tag{23}$$

Plane equation of Π_2

$$a_2 x + b_2 y + c_2 z + d_2 = 0 \tag{24}$$

From Eq. (23)

$$x = \frac{-b_1 y - c_1 z - d_1}{a_1}$$

Eq. (24) then gives

$$a_2 \frac{-(b_1 y + c_1 z + d_1)}{a_1} + b_2 y + c_2 z + d_2 = 0$$

$$a_2 \frac{-(b_1 y + c_1 z + d_1)}{a_1} + b_2 y + c_2 z + d_2 = 0$$

$$\frac{-a_2 b_1 + a_1 b_2}{a_1} y = \frac{a_2 c_1 - a_1 c_2}{a_1} z + \frac{a_2 d_1 + a_1 d_2}{a_1}$$

$$(-a_2 b_1 + a_1 b_2) y = (a_2 c_1 - a_1 c_2) z + a_2 d_1 + a_1 d_2$$

$$y = \frac{(a_2 c_1 - a_1 c_2) z + a_2 d_1 + a_1 d_2}{(-a_2 b_1 + a_1 b_2)}$$

Using Eq. (23) and express in term of x and y

$$z = \frac{-a_1 x - b_1 y - d_1}{c_1}$$

Eq. (24) gives

$$\frac{a_2c_1}{c_1}x+\frac{b_2c_1}{c_1}y+\frac{-a_1c_2x-b_1c_2y-c_2d_1}{c_1}+\frac{c_1d_2}{c_1}=0$$

$$a_2c_1x+b_2c_1y+-a_1c_2x-b_1c_2y-c_2d_1+c_1d_2=0$$

$$(a_2c_1-a_1c_2)x-(b_1c_2-b_2c_1)y+c_1d_2-c_2d_1=0$$

$$y=\frac{(a_2c_1-a_1c_2)x+c_1d_2-c_2d_1}{(b_1c_2-b_2c_1)}$$

Equation of the intersection line **L** is

$$\frac{(a_2c_1-a_1c_2)x+c_1d_2-c_2d_1}{(b_1c_2-b_2c_1)}=y=\frac{(a_2c_1-a_1c_2)z+a_2d_1+a_1d_2}{(-a_2b_1+a_1b_2)} \qquad (25)$$

4.2.1.3 Problems

When the two planes are not parallel, they must intersect each other. In most cases, the normal planes intersect when the axial curve is not straight. For example, a straight rod is bent into a high curvature shape (U shape or S shape).

The existence of intersections between planes containing the LCF is a necessary but not sufficient condition for self-intersection. As shown in Figure 16, Plane 1 and Plane 2 intersect while there is no self-intersection in the object. Further tests are required to decide if a self-intersection occurred.

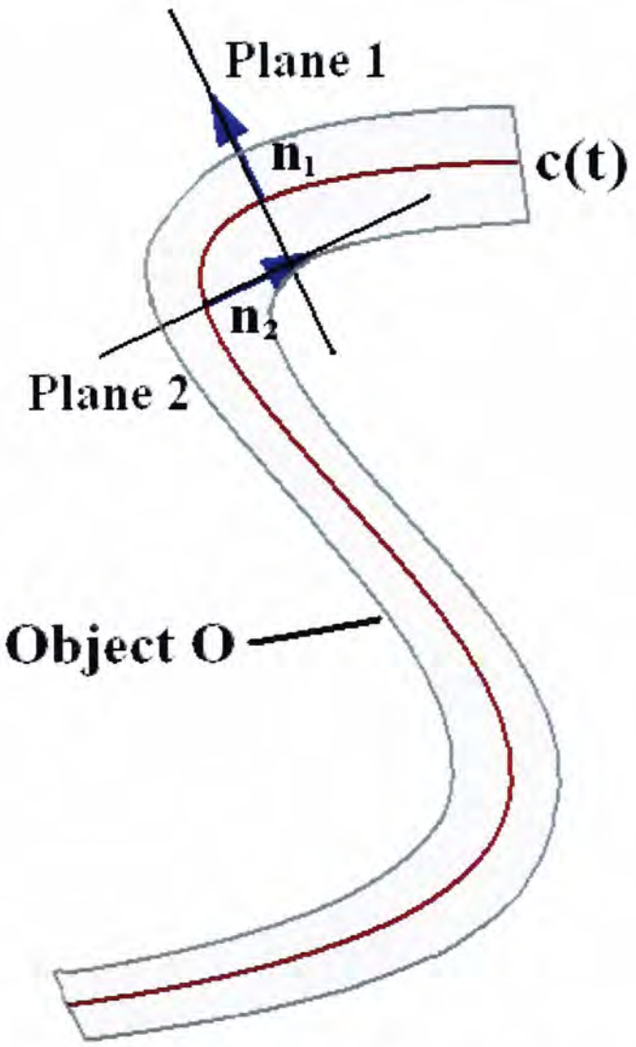

Figure 16: An axial curve is bent into a S shape with intersecting normal planes.

4.2.1.4 Approximating Axially Represented Object with Sphere

In order to provide a more accurate estimate of self-intersection, a set of spheres are constructed at the reference curve points. The radius of the sphere is the distance between the mesh vertices and their reference axial curve point. It shows in Figure 17. Self-intersection may occur if the intersection line between the normal planes passes through the spheres of the two reference curve points.

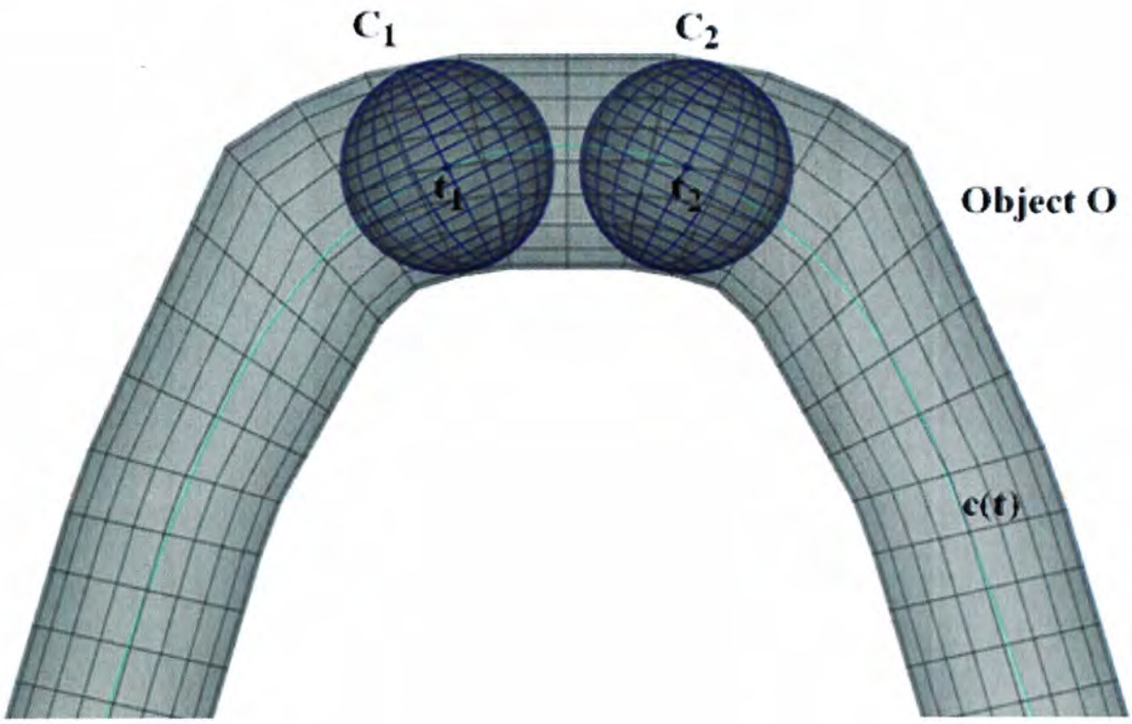

Figure 17: An axial curve is bent into S shape, there are intersected normal planes.

Assume r to be the radius of a sphere. The center is at its reference curve point $\mathbf{x}_0 = (x_0, y_0, z_0)$ on the axial curve and the equation of the sphere is

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = r^2 \tag{26}$$

Now, we can use Eq. 25 and Eq. 26 to check if the intersection line passes through two spheres. To simplify the process, the sphere is projected onto the normal plane. As a result, a circle is obtained on the plane after the projection which is shown in Figure 18.

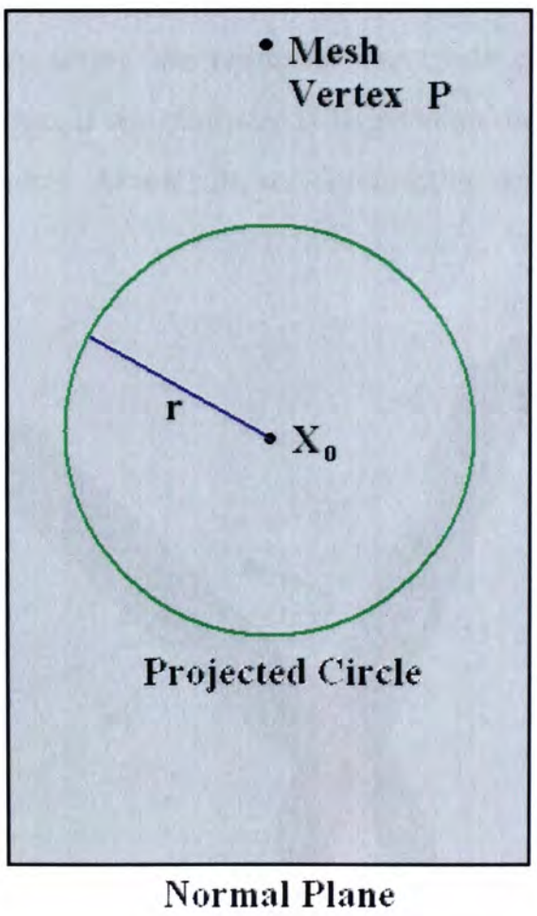

Figure 18: A sphere is projected onto the normal plane.

4.2.1.5 An Intersection Line with Two Circles

The center of the projected circle lies on its reference axial curve point and its radius is the distance between the mesh vertex and the axial curve point. In the proposed axial deformation method, each vertex is deformed according to the location and orientation of its reference LCFs. A set of normal planes are constructed at the LCFs and different sizes of circles are formed on the planes after the projection. The intersection line between two normal planes represents the common space between two planes, and the projected circle on the normal plane represents a region between the object and the axial curve. Since the center of the circle lies on the axial curve like the sphere, self-intersection detection can be detected by simply comparing the radius of the circle and the distance from the intersection line to the vertex. If the diameter is larger than the distance, the intersection line passes through the spheres. As a result, self-intersection occurs on the object mesh.

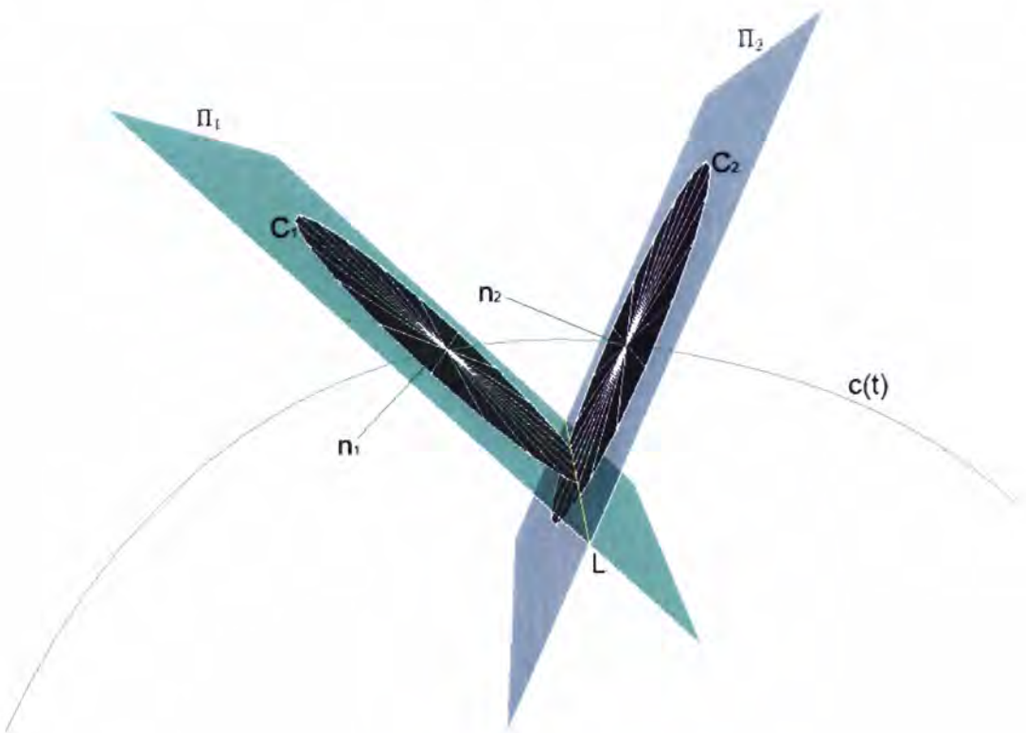

Figure 19: The intersection line passes through two circles on normal planes in a self-intersection.

4.2.2 Distance between a Vertex to a Curve Point

Consider $\mathbf{p} = (x_p, y_p, z_p)$ is the vertex on the object mesh and $\mathbf{x_0} = (x_0, y_0, z_0)$ is the curve point on the axial curve. Assume d as the distance between the vertex and the intersection line.

The radius of the circle is

$$r = |\mathbf{p} - \mathbf{x_0}| \tag{27}$$

Denote the intersection line as

$$\mathrm{L}: \; Ax + By + Cz + D = 0 \tag{28}$$

Then distance from a vertex point $\mathbf{p} = (x_p, y_p, z_p)$ to the intersection line **L** is

$$d = \left| \frac{Ax_p + By_p + Cz_p + D}{\sqrt{A^2 + B^2 + C^2}} \right| \tag{29}$$

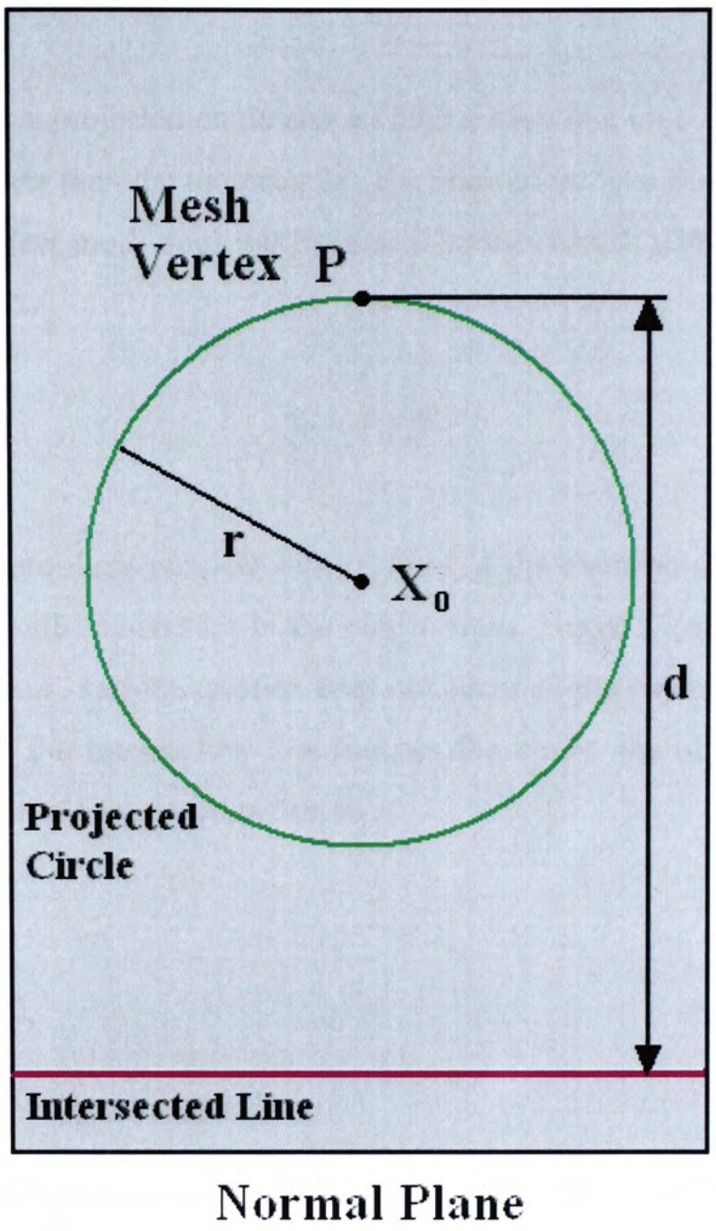

Figure 20: The projected circle and the intersection line are placed on a normal plane.

4.2.2.1 Possible Cases of a Line and a Circle

Figure 20 shows that a projected circle and an intersection line on a normal plane. When the distance d is larger than the diameter $2r$, the intersection line does not cut the circle. It means that the object mesh does not have self-intersection. Eq.30 is used to compare the values of d and $2r$.

$$2r - d > 0 \tag{30}$$

Using Eq. 30, there are three possible results. First, if the diameter $2r$ is larger than the distance d, there is self-intersection in the object mesh. Figure 21, if the diameter $2r$ is less than the distance d, self-intersection does not occur on the object mesh which shows in Figure 22. When the intersection line touches the circle, the object mesh does not overlap. Therefore no self-intersection occurs.

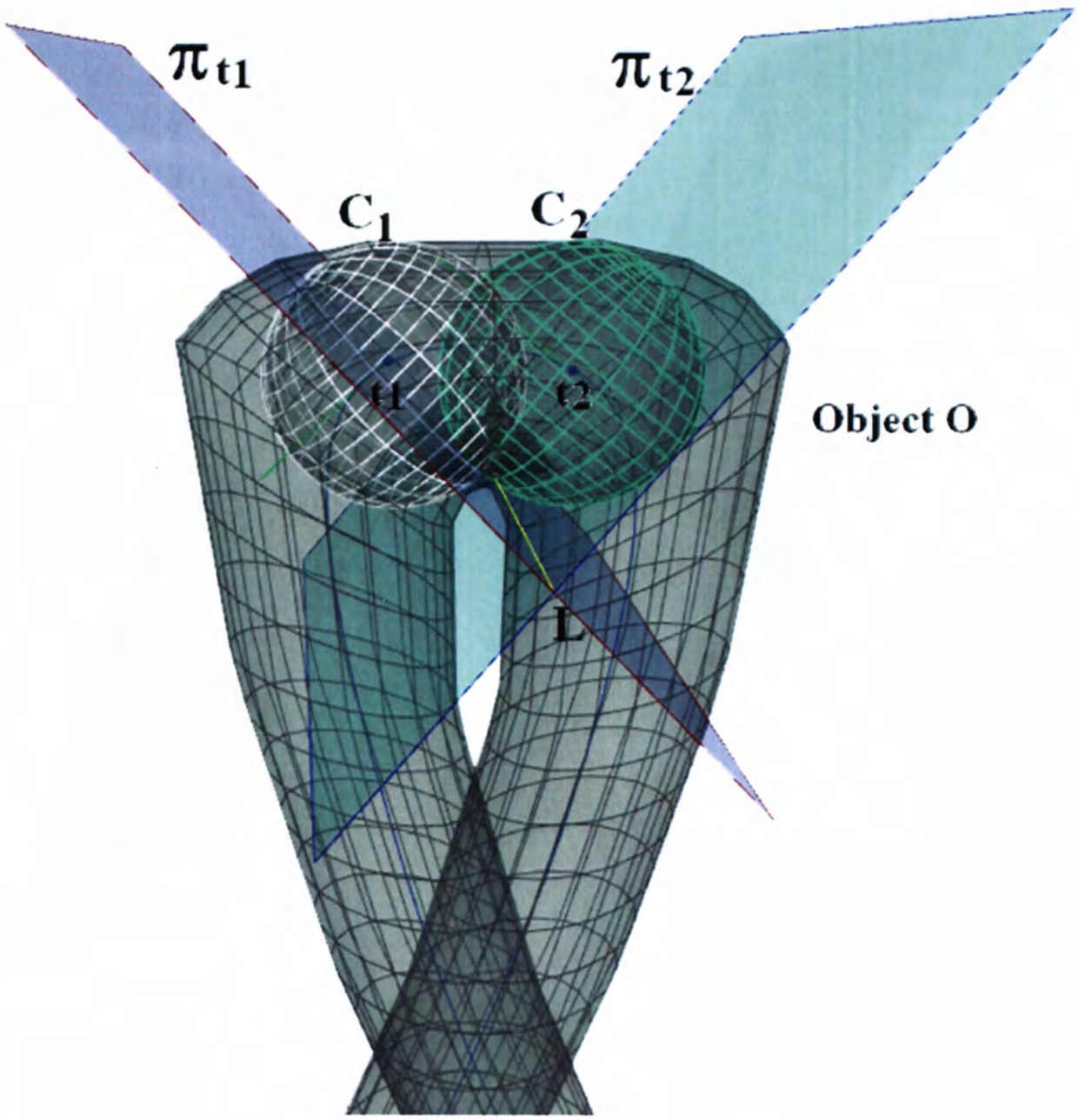

Figure 21: When the intersection line passes through the projected circle, self-intersection occurs.

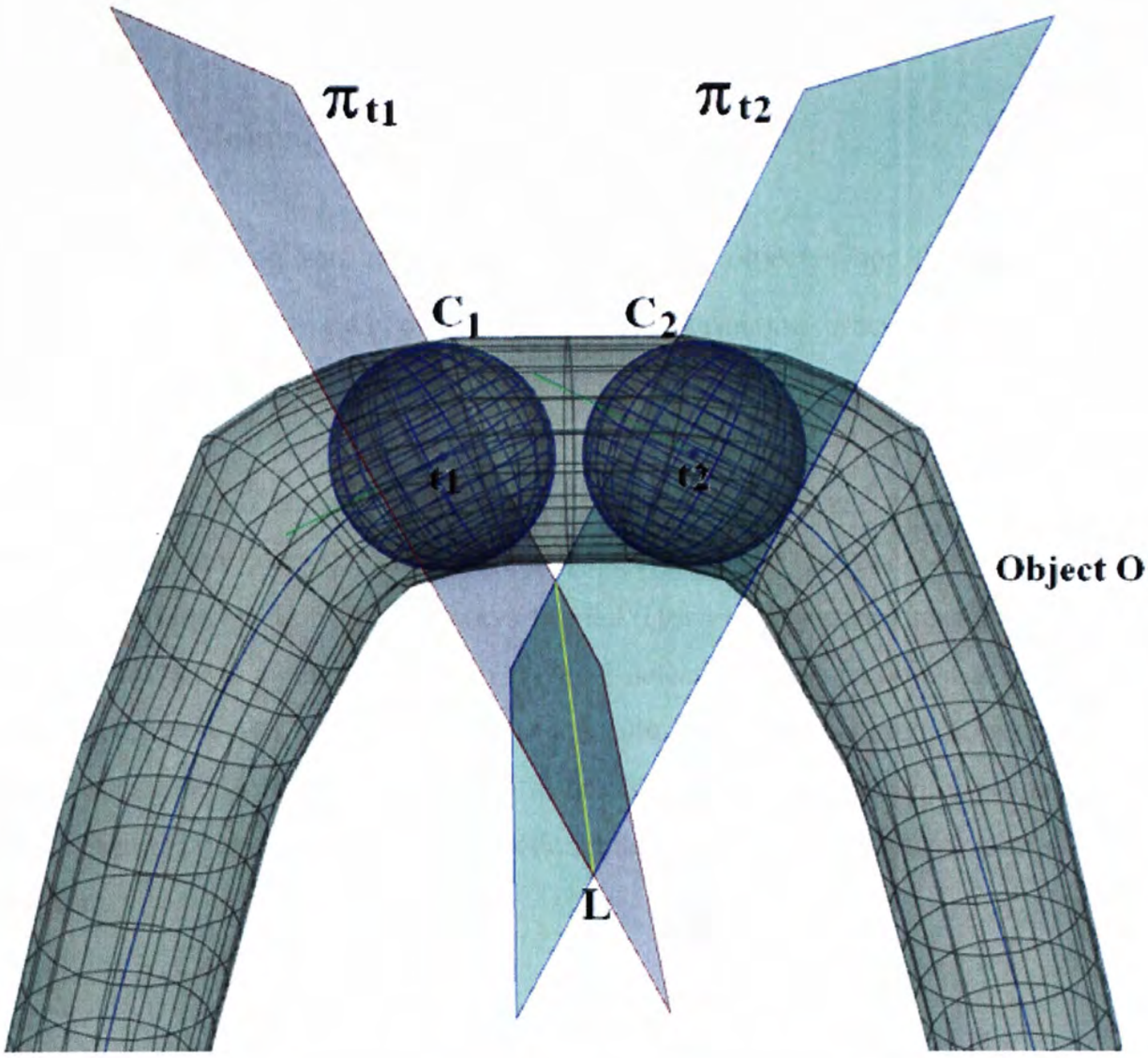

Figure 22: The intersection line does not pass through the circle, no self-intersection occurs.

4.3 Definition Proof

4.3.1 Define the Meaning of Self-intersection

Given two neighboring vertices point p_1 and p_2 of an object, when the location of p_1 and p_2 are flipped with each other during the deformation process, self-intersection occurs on the object surface.

In our proposed self-intersection algorithm, we introduced to use a set of certain size spheres and normal planes to detect the self-intersection. When two normal planes are intersected, an intersection line is always formed. However, by detecting the intersection of the normal planes cannot provide an accurate detection when an object is with S or U shape. Therefore a set of reference spheres are built and the radius are defined according to the distance between the object mesh to an axial curve. And then, we proposed that self-intersection occurs when an intersection line can pass through two spheres at the same time.

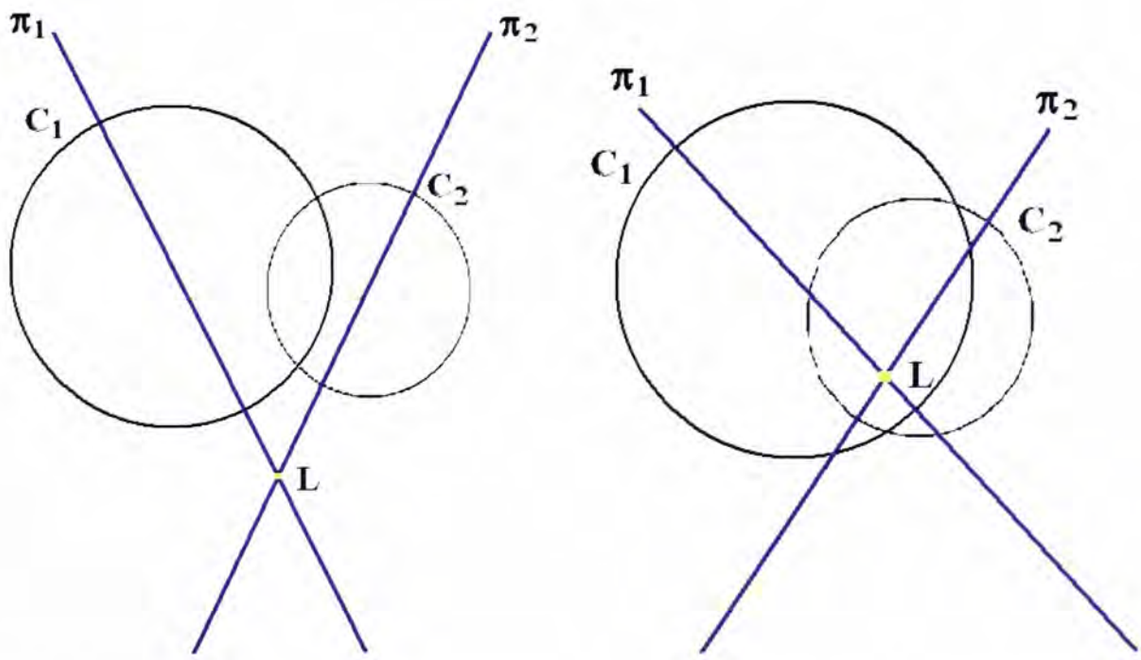

Figure 23: Two spheres are intersected: (a) the intersected line is out of both spheres; (b) the intersected line is inside the spheres.

4.3.2 Cross Product of Two Vectors

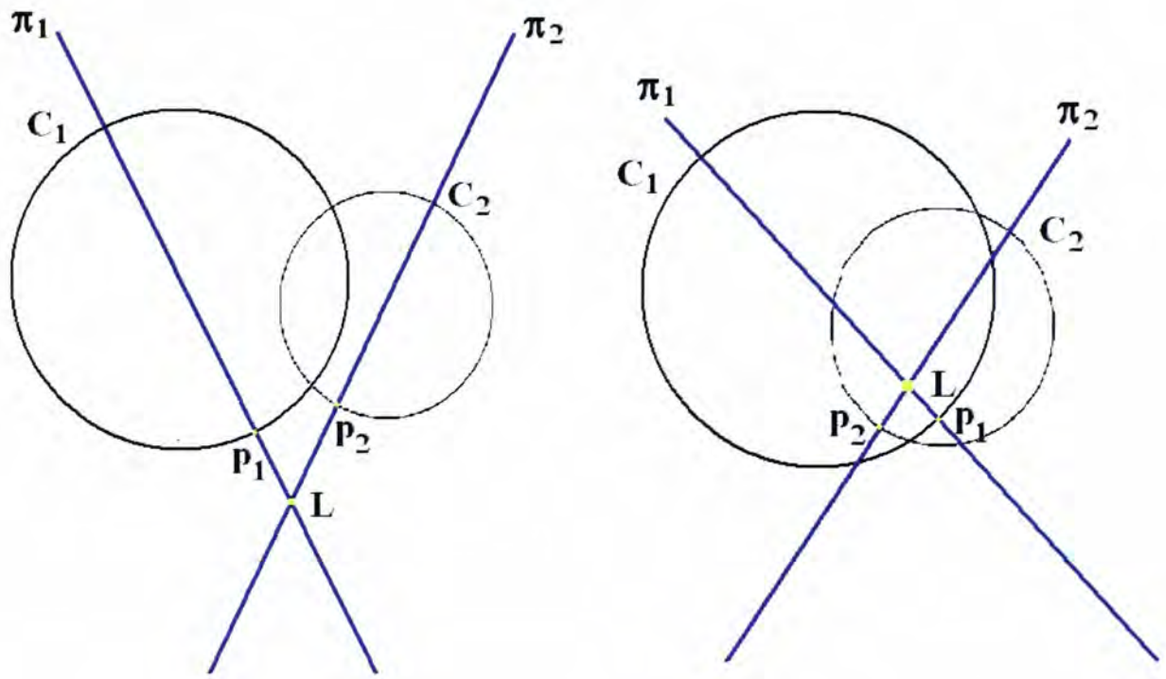

Figure 24: Two vertices of the intersected spheres: (a) two vertices are on the spheres; (b) the location of the vertices are flipped when the intersected line passes through both spheres.

Assume that p_1 and p_2 be the original object vertices and p_1' and p_2' be the deformed object vertices. Two vectors are formed by crossing p_1, p_2 and p_1', p_2'. When self-intersection occurs, two cross product vectors must be opposite to each other since the locations of two object points are flipped.

$$p_1 \times p_2 > -(p_1' \times p_2')$$

4.4 Factors Affecting the Accuracy of the Algorithm

The presented algorithm uses the local coordinate frame on an axial curve for detecting self-intersection. In this section, we will discuss three factors – curvature, thickness and vertices distance which can affect the accuracy of the algorithm.

4.4.1 High Curvature of the Axial Curve

Assume κ be the curvature of the curve point at parametric value t of an axial curve c. The curvature can be obtained by

$$\kappa(t) = \frac{\|c'(t) \times c''(t)\|}{\|c'(t)\|^3} \qquad \text{where } t = 1,23,\ldots \qquad (31)$$

Self-intersection usually occurs in the region with a high curvature on the axial curve. When the distance d between the mesh vertex $\mathbf{p} = (x_p, y_p, z_p)$ of the object O to the axial curve is larger than the radius of curvature κ at its reference point $\mathbf{x}_0 = (x_0, y_0, z_0)$ on the axial curve c, the region near that reference curve point may exhibit self-intersection.

$$d > \frac{1}{\kappa} \qquad \text{where } t = 1,23,\ldots \qquad (32)$$

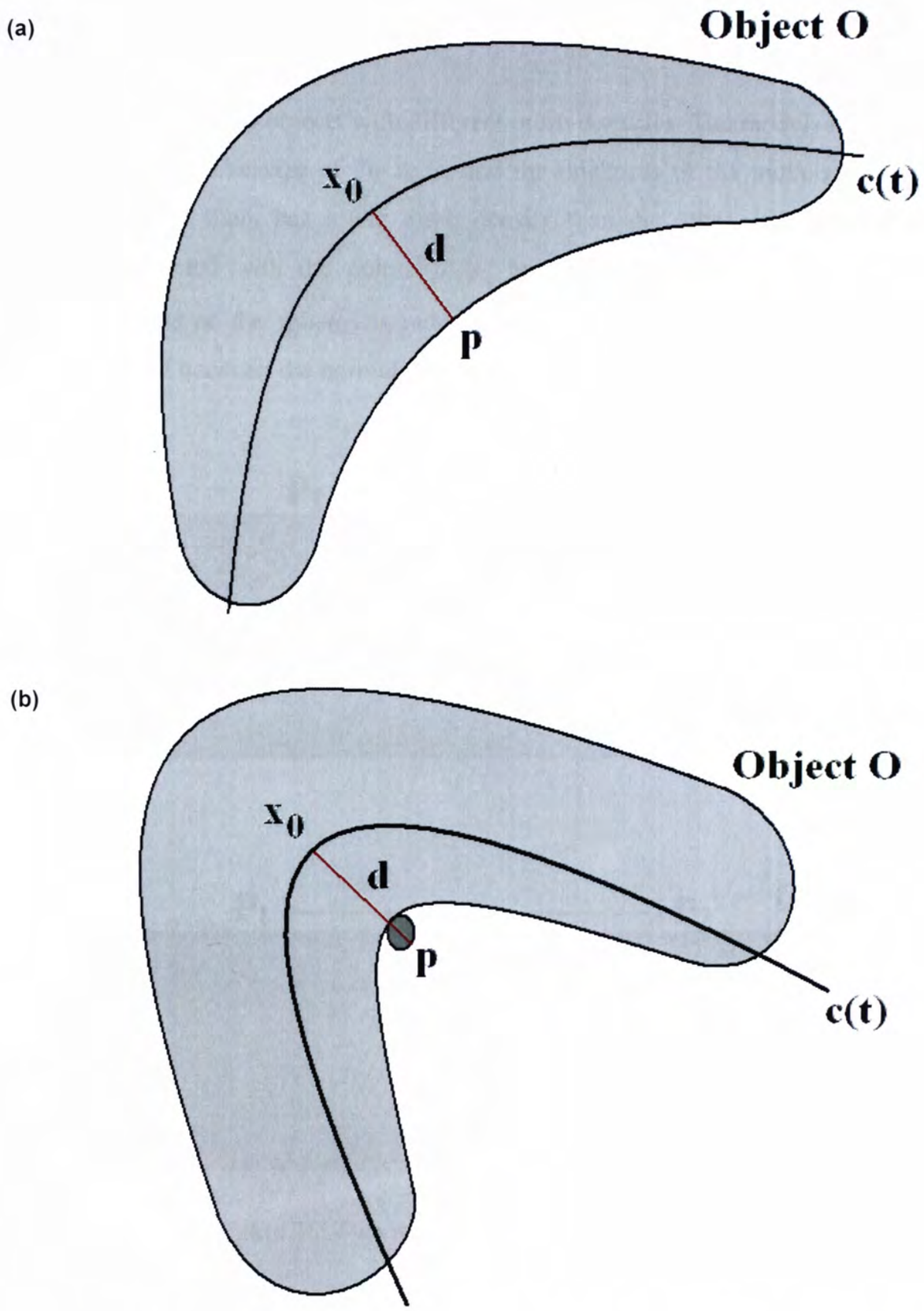

Figure 25: The result of an object which deformed by an axial curve: (a) The original shape of an object and an axial curve; (b) When the curvature of the axial curve is too high, self-intersection occurs on the object.

4.4.2 Mesh Density of an Object

In Figure 26 shows two objects with different mesh densities. The models in Figure 26(a) and (b) have the same size of the rods, and the thickness of the models are the same. However, one of them has lower mesh density than the other. In Figure 26(a), two spheres constructed with the points p_1, p_2 touched each other. In Figure 26(b), the vertices distance of the spheres is reduced and the two spheres will not interest. The intersection line between the normal planes will never intersect the spheres.

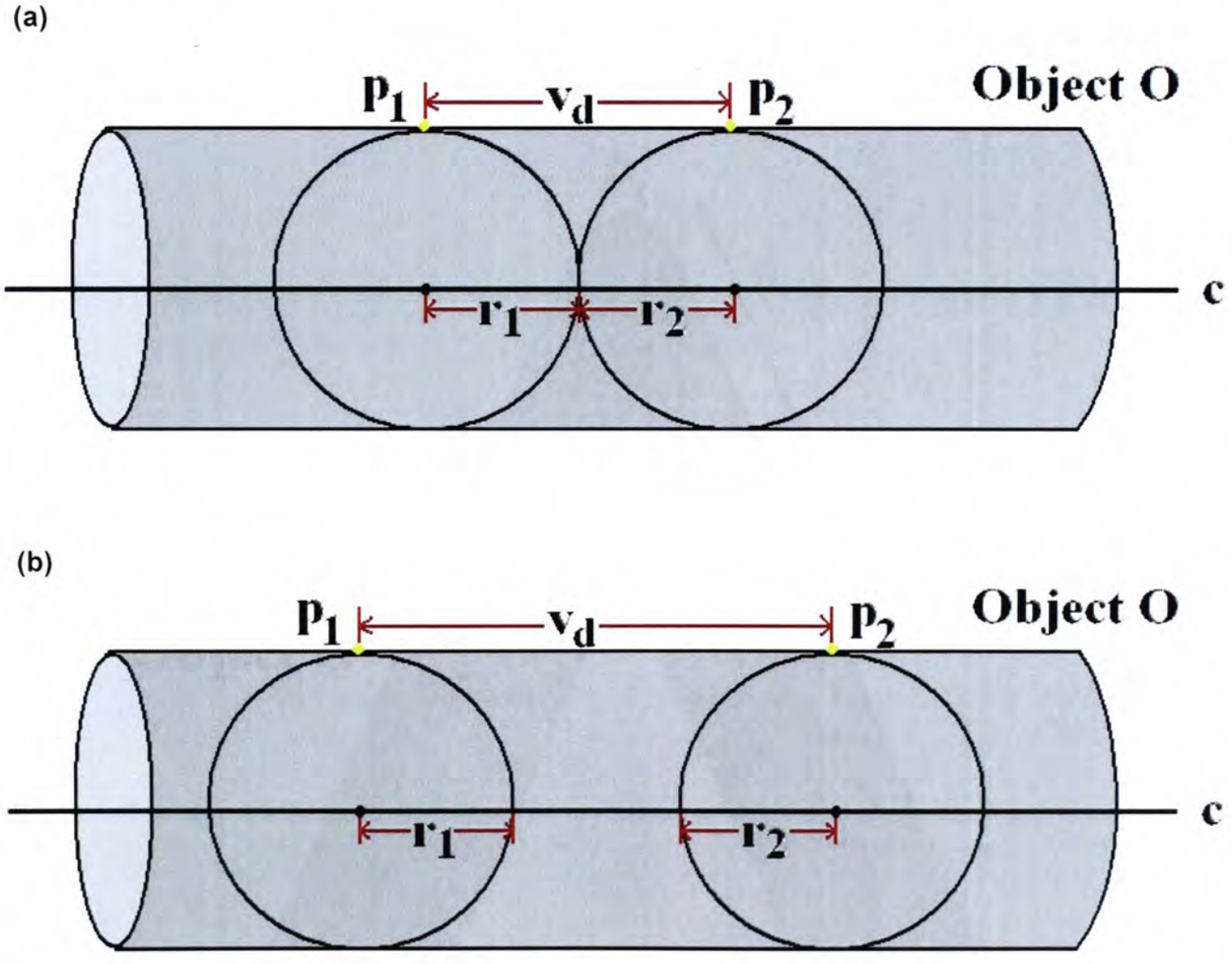

Figure 26: Two rods with different mesh densities.

Assume v_d be the distance between the vertices and r_i, r_j be the radius of the sphere. Self-intersection can be detected when the vertices distance $v_d = |p_2 - p_1|$ is less than or equal to the minimum value of the diameter $2r_j$ of a set of spheres, such that

$$v_d \leq \min[2r_i] \qquad \text{where } i = 1,2,3,\ldots \qquad (33)$$

The problem of Figure 26(b) object is that the system can not detect the self-intersection when the object bent into a high curvature shape. In Figure 27, it shows that a ribbon is bent with high curvature. The region between p_{i-1} and p_{i+1} occurs self intersection. But, since the distance between the vertices is too large, the sphere at a curve point t_i does not intersect with its neighboring spheres. Therefore, the vertices distance of the object mesh is one of the factors affecting the accuracy of our proposed algorithm.

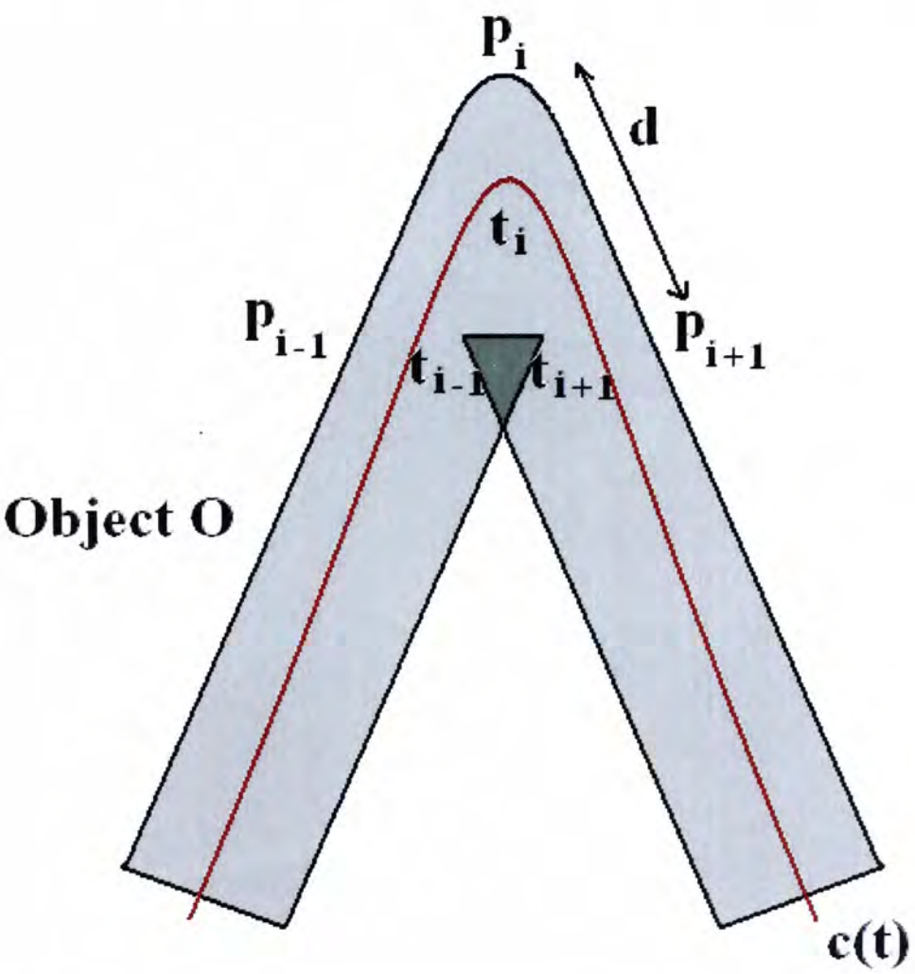

Figure 27: A ribbon is bent with a high curvature. Self-intersection occurs near the region of curve point p_i.

Assume v_d be the distance between the vertices and r_i, r_j be the radius of the corresponding sphere. Since each vertex has its own reference curve point and a sphere is constructed on that point, the vertices distance should be less then or equal to the minimum value of the sum of the distance between two neighboring spheres, or

$$v_d \leq \min[r_i + r_j] \quad \text{where } i, j = 1,2,3,\ldots \cap i \neq j \quad (34)$$

in order that self-intersection can be detected.

4.5 Architecture of the Self-intersection Algorithm

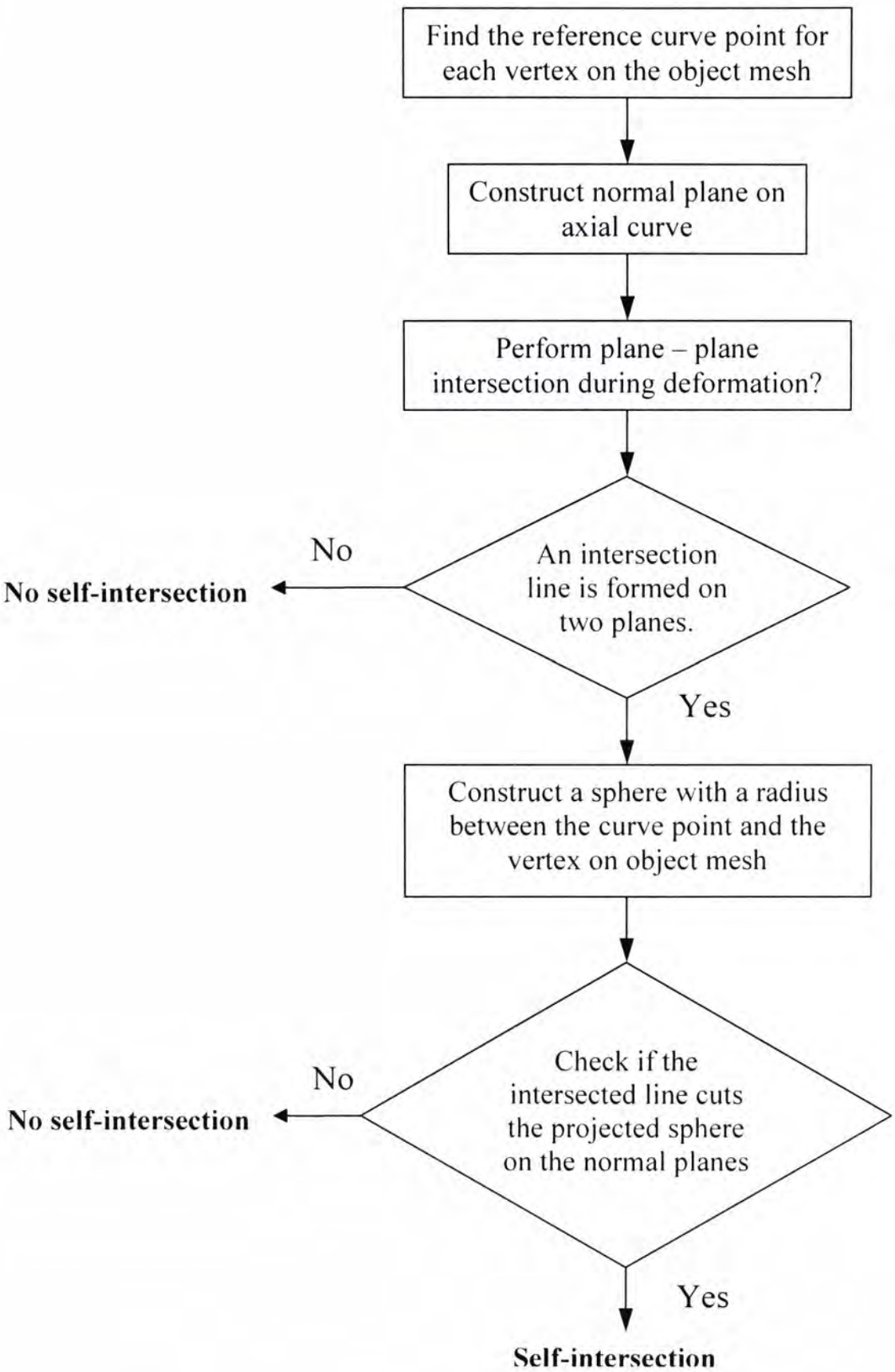

Table 1: Detecting self-intersection

4.6 Experimental Results

In this section, we show the results of our presented algorithm. In the examples shown, the regions of self-intersection are displayed in red.

In Figure 28, the region of high curvature where self-intersection occurs, are displayed in red. In Figure 29, an axial curve is defined on a straight rod. The rod is then bent such that there is self-intersection on the mesh.

A hand model with fingers is deformed in Figure 30. Multiple axial curves are included to model the skeletons of the hand. Each vertex of the hand model is attached to one axial curve point. When one of the axial curves is moved towards another axial curve, intersection may occur between the two axial curves. The system displays red color in the regions of the intersection.

A rod is bent to form a ring in Figure 31. In our method, we check the binormal of the normal planes. When two ends are touched each other, the tangents of normal planes are parallel. Using our algorithm, the region of two ends can discard the self-intersection detection occurs.

In Figure 32, a leaf ring is used to show the effect of the density of mesh. Two rings are with different mesh density. The one with higher density can provide accurate self-intersection detection than another.

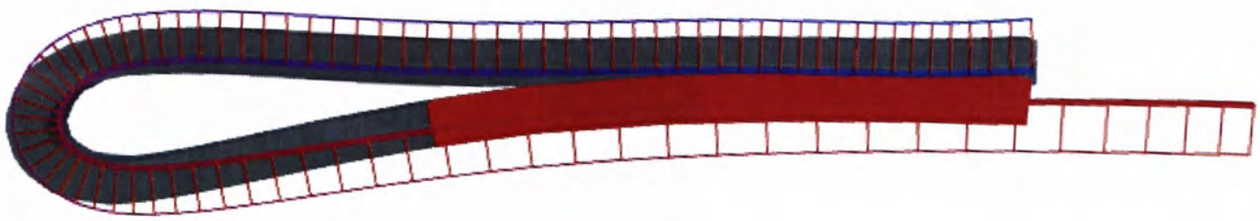

Figure 28: Self-intersection in a U-shaped rod.

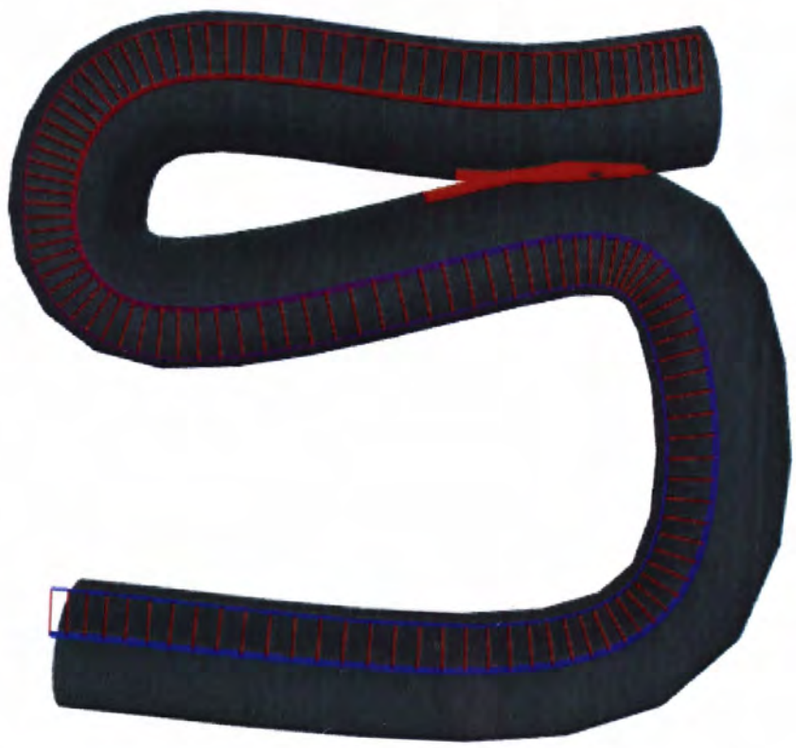

Figure 29: Bending a rod to a S shape with self-intersection.

Figure 30: A hand model with self-intersection. One finger is bent to collide with another fingertip. The red color shows the intersecting region of the hand model.

Figure 31: A rod deformed into a closed ring: (a) In perspective and front view, the rod ends do not touch each other; (b) Move one end to another, the mesh of the rod overlaps and causes self-intersection.

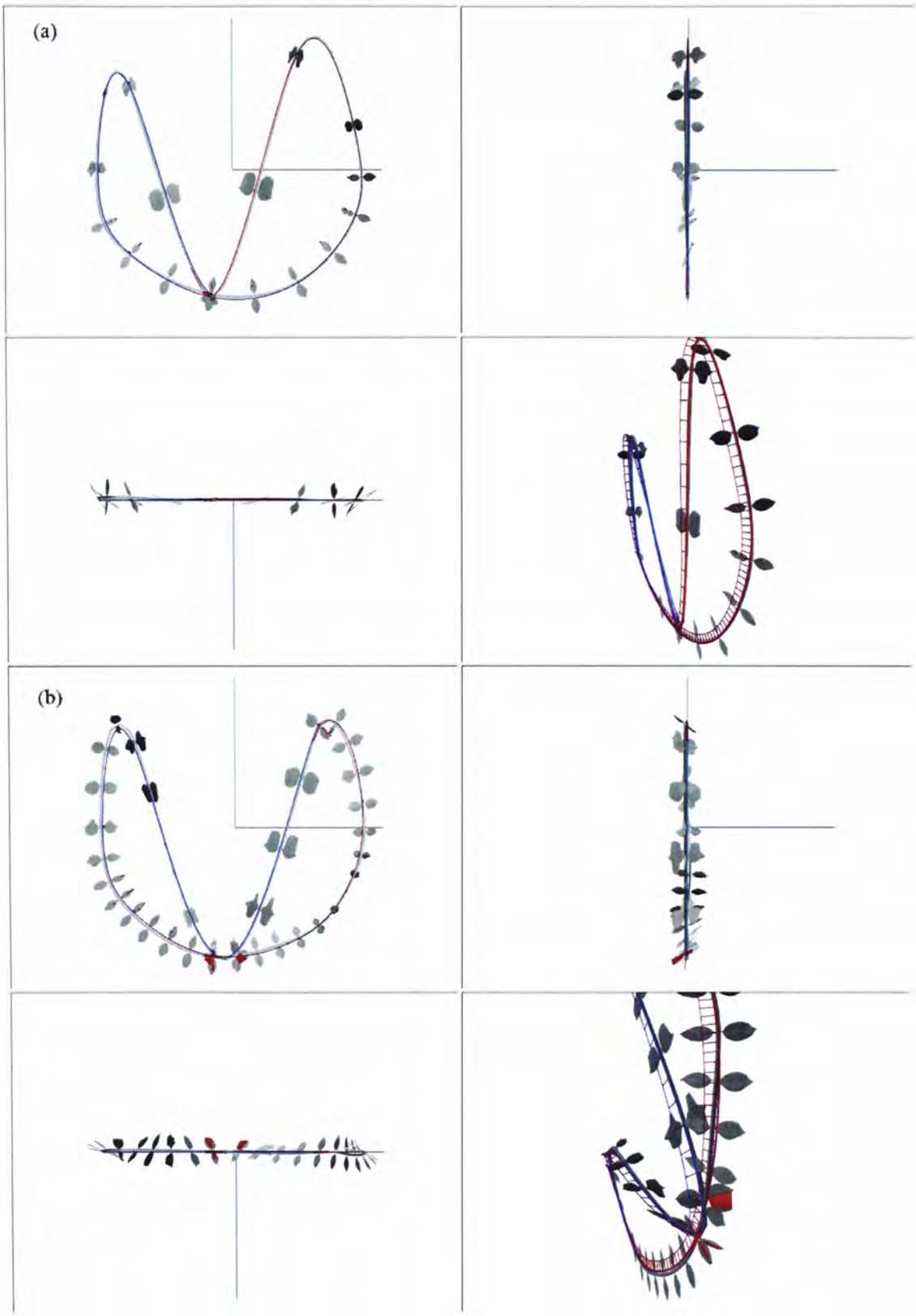

Figure 32: A leaf ring is deformed: (a) The ring with lower density of mesh, self-intersection in some parts is missed; (b) The model has higher mesh density, our proposed algorithm provides accurate self-intersection detection.

5. Conclusions and Future Developments

5.1 Contribution and Conclusions

An axial deformation technique is proposed in this thesis. A set of controllable local coordinate frames is used to deform a 3D object. Self-intersection in the deformed object is also one of the considered by using planes constructed at the local coordinate frames.

Axial deformation with controllable local coordinate frames is a reliable technique for deforming freeform shape. An axial curve is constructed with a NURBS curve which is manipulated by a set of control points. However, deforming axial curve by control points may result in undesirable twist or distortion of the associated object. In the proposed algorithm for the axial deformation of an object, a set of local coordinate frames along an axial curve is constructed by using the Normal Plane Projection method. When one particular local coordinate frame is modified, the modification is propagated to the whole set of local coordinate frames. An algorithm for interpolating the local coordinate frames is presented to maintain the smoothness and continuity of the coordinate frames along the axial curve. To associate object mesh points to points on the axial curve, each vertex on the object is mapped to the closest point on the axial curve. Users can directly control the LCFs to modify the shape of the axial curve. This modification is propagated to the attached mesh of the object. The new position of the mesh is updated automatically whenever the LCFs of the axial curve are modified. The smoothness and continuity of the axial curve can effectively reduce undesirable distortions of the shape. Experiments show that the technique can be used for blending and twisting regular or freeform objects and gives better results compared with the result of deforming object with curve pairs.

Self-intersection is one of the main problems in deformation. In the second part of this thesis, we present an algorithm for detecting the self-intersection in a deformation by using a set of constructed local coordinate frames. Using a set of LCFs on an axial curve, a set of normal planes is constructed. Plane-Plane intersection always occurs between normal planes and a line is formed when the planes are not parallel. Since the intersection

on the normal planes may not lead to self-intersection especially on the models with high curvature (U shape and S shape), the distance between the object mesh and the axial curve is also used for the detection. A sphere is constructed between the axial curve and the object. Its radius is the distance between the object mesh to the axial curve. When there is self-intersection in the object mesh is self-intersected, the intersection line between normal planes should pass through two spheres at the same time. By comparing the length between the mesh vertex to the axial curve with the diameter of the projected circle to the axial curve, the self-intersection can be detected. Experimental results show that this algorithm can be applied on freeform object with complex shapes.

5.2 Limitations and Future Developments

This thesis presents an axial deformation technique which uses a set of controllable local coordinate frames. It also presents an algorithm to prevent self-intersection on the object mesh by using the local coordinate frames. However, there are limitations in these approaches as listed below:

1. In proposed axial deformation method, when one LCF is modified, the neighboring LCFs are also adjusted. Therefore the user cannot specifically deform certain region of an object and the object mesh is always smooth.

2. The number of the mesh vertices can affect the accuracy of the algorithm for self-intersection detection. When more vertices are placed on the surface, more LCFs are constructed on the axial curve. But, if the object contains fewer vertices, fewer spheres are used to detect the self-intersection. This may result in inaccurate self-intersection detection.

There are also several potential works that may be considered in the future. They are described as follows:

1. In the proposed system, the continuity and smoothness of the object is always maintained. However, some designs may require non-smooth surface. Some extra constraints will have to be included to provide more control on the deformation process.

2. If the object has more mesh vertices, the self-intersection detection is more accurate. However, the process is time consuming. Some further study can be performed to locate an optimum number of object vertices for accurate self-intersection.

References

1 Lazarus F. et al., Axial deformations: an intuitive deformation technique, *Computer-Aided Design*, 1994, 26(8), pp.607-613.

2 Hui K.C., Free-form design using axial curve pairs, *Computer-Aided Design*, 2002, 34(8), pp.583-595.

3 Gershon Elber, Distance Separation Measures Between Parametric Curves and Surfaces Toward Intersection and Collision Detection Applications, *2005 ACM*, pp.63-75.

4 Wallner J. et al., Self intersections of offset curves and surfaces, *International Journal of Shape Modeling*, 2001, 7(1), pp. 1-21.

5 Sederberg T. and Parry S., Free-form deformations of solid geometric models, *Computer Graphics*, 1986, 20, pp.151-160.

6 Coquillart S., Extended free-form deformations: A sculpting tool for 3D geometric modeling, *Computer Graphics*, 1990, 24(4), pp.187-196.

7 Bloomenthal J., Lim C., "Skeletal Methods of Shape Manipulation", *Proc. Shape Modeling International '99*, 1999, pp.44-47.

8 Yoshizawa S., Belyaev A.G., Seidel H.-P., Free-form skeleton-driven mesh deformations, *ACM SIGGRAPH 2003*, pp.247-253.

9 Wang C.C.L., Wang Y., and Yuen M.M.F., Design automation for customized apparel products, *Computer-Aided Design*, 2005, 37, pp.675-691.

10 Yoshizawa S., Belyaev A.G., Seidel H.P., A simple approach to interactive free-form shape deformations, *Pacific Graphics 2002*, 2002, October 9-11, pp.471-474.

11 Klok F., Two moving coordinate frames for sweeping along a 3D trajectory, *Computer Aided Geometric Design*, 1986, 3, pp.217-229.

12 Cohen E., Lyche T. and Riesenfeld R., Discrete B-splines and subdividsion techniques in computer-aided geometric design and computer graphics, *Comput. Graph. & Image Proc.*, 1980, Vol 14, pp.87-111.

13 Farin G., Curves and Surfaces for Computer Aided Geometric Design Academic Press, 1988, USA.

14 T. Samoilov and G. Elber, Self-intersection elimination in metamorphosis of two-dimension curves, *The Visual Computer*, 1998, 12, pp.415-428.

15 J.K. Seong, G. Elber and M.S. Kim, Trimming local and global self-intersections in offset curves/surfaces using distance maps, *Computer-Aided Design*, 2006, 38, pp.283-193.

16 Pekerman Diana et al., Global Self-Intersection Detection and Elimination in Freeform Curves and Surfaces with Applications in Metamorphosis, 1996

17 Pekerman Diana et al., Self-Intersection Detection and Elimination in Freeform Curves and Surfaces, *Computer-Aided Design*, 2008, v.40 n.2, pp.150-159.

18 N. Burtnyk and M. Wein, Interactive skeleton techniques for enhancing motion dynamics in key frame animation, *CACM*, October 1976, pp.546-569.

19 Lossing D.L., Eshleman A.L., Planning a common data base for engineering and manufacturing. *SHARE XLIII*, Chicago, 1974, Aug.

20 Wang W. and Joe B., Robust computation of the rotation minimizing frame for sweep surface modeling, *Computer-Aided Design*, 1997, 29(5): pp.379-391.

21 Gray A., Modern Differential Geometry of Curves and Surfaces, 1993, *CRC Press.*

22 Bloomenthal J., Calculation of Reference Frames along a Space curve, *Graphics Gems*, 1990, pp.567–571.

23 Borrel, P., Rappoport, A simple constrained deformations for geometric modeling and interactive design, *ACM Trans. Graph*, 1994, 13(2), pp. 137–155.

24 James E. Gain and Neil A. Dodgson, Preventing Self-Intersection under Free-Form Deformation, *IEEE Transactions on Visualization and Computer Graphics*, 2001, 7(4): pp.289-298.

25 Ji et al., A Novel Axial Deformation Algorithm without Local Self-Intersection, *Chinese Journal of Computer*, 2006, 29(5), pp. 828-834.

26 Lee et al., Realistic Human Hand Deformation, *Comp. Anim. and Virtual Worlds*, 2006, 17, pp. 479–489.

27 Yoshizawa S. et al., Free-form skeleton-driven mesh deformations, *Proceedings of Pacific Graphics*, 2002, pp.471-474.

28 Charlie Wang C.L. et al, Virtual human modeling from photographs for garment industry, *Computer-Aided Design*, 2003, 35(6), pp577-589.

29 W.T. Correa, R.J. Jensen, C.E. Thayer, & A Finkelstein, Texture mapping for cel animation, *SIGGRAPH 98 Conference Proceedings*, 1998, pp. 435-456.

30 Funkhouser T. et al., Modeling by Example, *ACM Transactions on Graphics*, 2004, 23(3), pp.652-663.

31 Faux I.D., Pratt M.J., Computation geometry for design and manufacturing, *Wiley*, 1979.

32 Woodward C., Skinning techniques for interactive B-spline interpolation, *Computer-Aided Design*, 1988, 20(8), pp.441-451.

33 Tao Yu Fei et al., Continuous Nearest Neighbor Search, *Proceedings of 28th VLDB Conference*, 2002.

34 Wang H., An analytical solution for free-form roads in driving simulation, *Technical Report 01-04*, Department of Computer Science, the University of Iowa, 2001.

35 Waters K., Levergood T.M., "DECface: An Automatic Lip- Synchronization Algorithm for Synthetic Faces", Tech. Report, CRL 93/4, DEC *Cambridge Research Lab.*, 1993.

36 Siltanen P. and Woodward C., Normal orientation methods for 3D offset curves, sweep surfaces and skinning, *EUROGRAPHICS '92*, 1992, 11(3), pp.449-457.

37 Peng Qunsheng, Jin Xiaogang, and Jieqing Feng, Arc-Length-Based Axial Deformation and Length Preserving Deformation, *In Computer Animation'97, IEEE Computer Society*,1997, pp.86-92.

38 Conquillart S., Computer offsets of B-spline curves, *Computer-Aided Design*, 1987, 19(6), pp.305-309.

39 Hui K.C., Axial Representations of 3D Shapes, *IEEE Workshop on Statistical Signal Processing 2003*, 2003, pp.311-314.

40 Ignacio Llamas , Alexander Powell , Jarek Rossignac , Chris D. Shaw, Bender: a virtual ribbon for deforming 3D shapes in biomedical and styling applications, *Proceedings of the 2005 ACM symposium on Solid and physical modeling SPM '05* , 2005, pp.88-99.

41 Remco C. Veltkamp and Wieger Wesswlink, Modeling 3D Curves of Minimal Energy, *EUROGRAPHICS '95*, 1995, 14(3), pp.97-110.

42 Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popovic, Mesh Based Inverse Kinematics, *SIGGRAPH 2005*.

43 Gerold Wesche, A User Interface for Free-Form shape Design at the Responsive Workbench, *ASME*, 2004, 4, pp.178–185.

44 Hui K.C., Axial Representation for Modeling 3D Shapes, *IEEE 2000*, 2000, pp.401-406.

45 Jung W.Y. et al., Self-intersection Removal in Triangular Mesh Offsetting, pp.477-484.

46 Diana Pekerman et al., Global Self-Intersection Detection and Elimination in Freeform Curves and Surfaces with Applications in Metamorphosis.

47 A. Galligo and J.P. Pavone, A sampling algorithm computing self-intersections of parametric surfaces, pp.185-204.

48 Paul Bourke, The Intersection Of Two Planes, February 2000.

49 Shin Yoshizawa et al., Skeleton-based Variational Mesh Deformations, *EUROGRAPHICS 2007*, 2007, 26(3), pp.255–264.

50 K.C. Hui, Smooth blending of subdivision surfaces, *Computer-Aided Design*, 2006, 38, pp.786-799.

51 Paul Bourke, Interpolation Methods, December 1999.

52 Samoilov T. and Elber G., Self-intersection elimination in metamorphosis of two-dimensional curves, *The Visual Computer*, 1998, 14, pp.415-428.

53 Jung W.Y. et al., Self-intersection Removal in Triangular Mesh Offsetting, pp.477-484.

54 Lewis J.P. et al., Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation, *In Proceedings of the ACM SIGGRAPH 2000 Conference*, pp.165-172.

55 Maryann S. and Carlo H., 2D Shape Decomposition and The Automatic Generation of Hierarchical Representations, *International Journal of Shape Modeling*, 1998.

56 Wang H.L. et al., Robust and Efficient Computation of the Closest Point on a Spline Curve, *Curve and Surface Design*, 2002, pp.397-405.

57 K.C. Hui, Smooth blending of subdivision surfaces, *Computer-Aided Design*, 2006, 38, pp.786-799.

58 M.L. Chan and K.C. Hui, Dynamic Axial Curve-Pair Based Deformation, *Edutainment 2008*, 2008, LNCS 5093, pp.593-602.

59 Wang W.P. and Joe Barry, Robust computation of the rotation minimizing frame for sweep surface modeling, *Computer-Aided Design*, 1997, 29(5), pp.379-391.

60 Andrew J. Hanson, Quaternion Frenet Frames: Making Optimal Tubes and Ribbons from Curves, 2004, AJ Hanson.

61 Pekka Siltanen and Charles Woodward, Normal orientation methods for 3D offset curves, sweep surfaces and skinning, *EUROGRAPHICS '92*, 1992, 11(3), pp.449-457.

62 Liu H.Z. et al., Parameterized Freeform Shape Design and Deformation, *MUE'07*, 2007, pp.626-632.

63 Gabriel J. et al., Novel Skeleton Representation for Articulated Creatures, *ECCV 2004*, LNCS 3023, pp.66-78.

64 Li H. et al., Interactive Spline-Driven Deformation for Free-Form Surface Styling, *2006 ACM*, 06-08 June 2006, pp.139-147.