Université de Montréal

Visualization and prediction of spatial deformation using thin-plate splines in the context of scoliosis

par

Di Jiang

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la faculté des études supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)

en informatique

Août 2003

Université de Montréal

Faculté des études supérieures

Ce mémoire de maîtrise intitulé

Visualization and prediction of spatial deformation using thin-plate splines in the context of scoliosis

présenté par

Di Jiang

a été évalué par un jury composé des personnes suivantes :

Max Mignotte
président-rapporteur

Neil F. Stewart
directeur de recherche

Farida Cheriet
membre du jury

Sommaire

La scoliose est une déformation tridimensionnelle de la colonne vertébrale qui engendre des déformations du torse. Le risque d'un cancer associé à la méthode de diagnostique actuelle, les rayons X, appelle aux changements dans ce domaine. Ceci est particulièrement important considérant le grand nombre de cas chez les adolescents. L'ordinateur a pris un rôle de plus en plus important dans le domaine médical, au point d'avoir été utilisé pour presque toutes les différentes parties du corps humain.

Plusieurs techniques de déformation spatiale ont été développées: les déformations sans contraintes (*free-form*) et leurs multiples extensions, les modèles de lissage par splines, la technique ¨space deformation¨, *etc.* Elles offrent à l'usager un contrôle sur la déformation par divers moyens: points marqueurs, treillis de contrôle et manipulations directes.

Dans ce projet, pour résoudre le problème de visualisation et de simulation de la scoliose, une méthode de déformation spatiale, les plaques minces (*thin-plate spline models*) a été employée. Pour la prédiction de la scoliose, nous travaillons à la fois sur un ensemble d'indexes de R^d, utilisé pour l'interpolation et l'approximation spatiale, et un ensemble d'indexes de l'intervalle [0,1] pour l'interpolation et l'extrapolation dans le temps. Nous testons et validons nos modèles avec des données de vrais patients. Les résultats des tests sur les données réelles que nous avons obtenus sont raisonnables, en se basant sur le niveau de précision des données disponibles: les résultats de déformation externe sont plutôt bons, tant dis que les résultats de déformation interne montrent quelques erreurs. Des travaux ultérieurs pourraient se concentrer sur ces deux points, augmenter

la précision des données réelles et prendre en compte plus d'information interne physique.

Une contribution particulière de ce projet est que nous montrons le lien entre les plaques minces sur un ensemble d'indexes de l'intervalle [0,1] et un ensemble d'indexes de R^d. Il s'agit de deux modèles differents de splines, mais après tout le travail mathématique (permutation et comparaison), nous pouvons voir qu'ils partagent vraiment quelque chose en commun, ce qui est très utile.

Mots clefs:
simulation médical, visualisation, modèls de plaques minces, modélisation d'objets déformarbles, scoliose.

Abstract

Scoliosis is a common 3D spinal deformity that leads to aesthetic deformity of the torso. The cancer risk associated with the current diagnosis method, X-rays, motivates modifications of this method. This is especially important because of the high occurrence of scoliosis among teenagers. The computer has begun to take more and more important roles in the medical field; it has been used to study almost every part of the human body.

Many spatial deformation techniques have been developed: Free-form deformation and its several extensions, smoothing spline models, space deformations,*etc.* They offer the user control over deformation by different means: marker points, control lattices and even direct manipulations.

This project approached the problem of visualization and prediction of scoliosis with a method of spatial deformation, specifically, thin-plate spline models. For the prediction of scoliosis, we work on both the R^d index set, which we use for spatial interpolation and approximation, and the time index set, which we use for time-interpolation and time-extrapolation. We test and validate our models with real patient data. The real testing results we got are reasonable, based on the accuracy level of the available data: the external deformation results are pretty good, while the internal deformation results show some errors. Later work could extend from these two points, increase the accuracy of the real data and take more internal physical information into account.

A special contribution of this project is: we show the link between the thin-plate spline models based on [0,1] index set and R^d index set. These are two different spline models, but after some mathematical work (permutation and com-

parison), we can see that they really share something in common, which is quite useful.

Keywords:
medical simulation, visualization, thin-plate spline models, modeling of deformable objects, scoliosis.

Contents

List of Figures

List of Tables

Acknowledgment

I would like to express my sincere appreciation to Professor Neil F. Stewart, my research supervisor, for his invaluable guidance, his encouragement and his great support throughout all the course of this thesis work. I would like express my thanks to Professor Farida Cheriet who gave me a lot of help and support during my work. This thesis benefits from their careful reading and constructive advice.

Also, I would like to thank Dr. Hubert Labelle, who gave me the suggestions for my test cases; Christian Bellefleur, who helped me for the data processing; Valerie Pazos, Luc Duong and all the other persons in the research center of Ste-Justine hospital for their help. And, I would like to thank Jacob Jaremko, University of Calgary, for his help with the data problem.

Plus, I wish to thank all the people in the Laboratoire d'Informatique Graphique de l'Université de Montréal (LIGUM), especially François Duranleau, for all the help they offered me for my work and the wonderful environment they provided.

Finally, I wish to express my special acknowledgment for my parents, for their great support and encouragement!

Chapter 1

Introduction

1.1 Background

Medicine is an extremely challenging field of research, which has been — more than any other discipline — of fundamental importance in human existence. The variety and inherent complexity of unsolved problems, along with the obvious intrinsic interest of improved human health, have made it a major driving force for many natural and engineering sciences. Therefore, the medical field has become one of the most important application areas with an enduring supply of exciting research challenges for computer scientists. Computers have been used to study almost every part of the human body: the head, face, spine, wrist, hand, knee, foot, even some internal soft tissue organs.

1.1.1 Scoliosis

Amongst all of these research topics, study of the spine has become more and more attractive because of the intrinsic complexity and invisibility of the spine (see Fig.1.1). There are four main kinds of disorders concerning the spine [3]:

1. Scoliosis

   Scoliosis is a 3D abnormality of the trunk in which the spine loses its normal left-right symmetry and instead develops a lateral curvature associated with

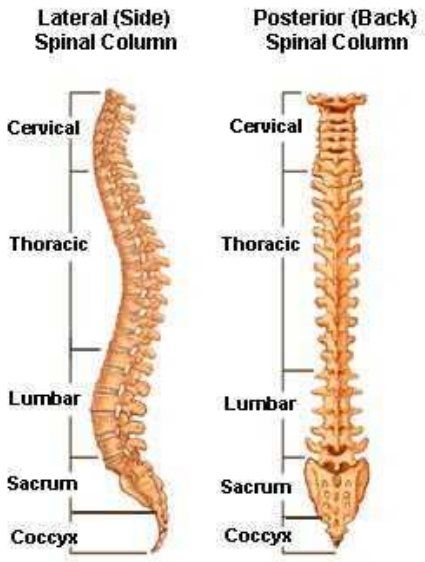


Figure 1.1: Human spine illustration — lateral view and posterior view [10]

rotation and deformity of the vertebrae, rib cage and torso. It is a condition involving lateral curve or angular derivation of one or more vertebral segments, often with twisting of the spinal column.

2. Lordosis

   An exaggeration of the posterior concavity of the spine, characteristic of the lumbar region. It is also called "sway back", indicating extreme anterior curvature of the lumbar spine.

3. Kyphosis

   An exaggeration of the posterior convexity of the thoracic vertebral column (humpback). It may be due to the absence of a vertebral body, malformation by incomplete segmentation of vertebral bodies, absence of a corner or flattening by compression.

4. Osteoporosis

   A disease of the bone due to deficiency of bony matrix.

Amongst all these, scoliosis, especially idiopathic scoliosis, is quite important due to its frequency among children and teenagers, and it therefore interests many

computer researchers[1].

Figure 1.2: A typical scoliotic skeleton figure [2].

1.1.2 Motivations

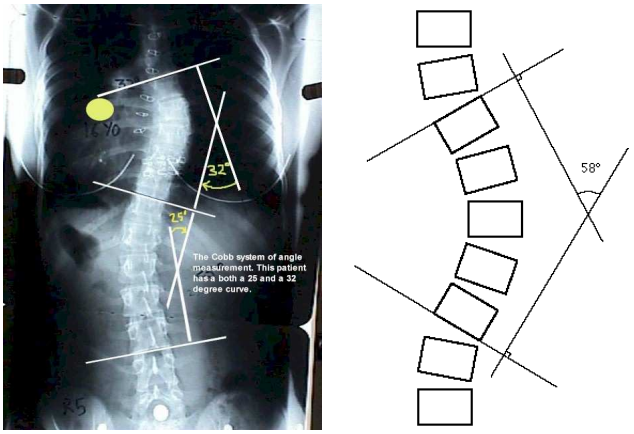
The original motivation of our project is partially based on the limitations (the side effects described below) of current methods in the diagnosis of scoliosis, and also because of its frequency.

First, the current method of diagnosis of scoliosis is a traditional and still-reasonable way — X-rays. But many medical researchers have warned that repeated exposure to X-ray radiation may lead to an increased risk of breast, bone and thyroid cancer. This risk is more important to children, among whom scoliosis has a higher frequency of occurrence. Also, radiography provides only a planar projection: for a three-dimensional deformation like scoliosis, a two-dimensional image can provide only limited information. There has been some work done in this field, the 3D reconstruction from the 2D radiography (see [1]).

Second, the current method of evaluation of scoliosis uses the Cobb angle (see Fig 1.3) from posterior-anterior or anterior-posterior radiography. To use the Cobb angle method, one must first decide which vertebrae are the end-vertebrae of the curve. These end-vertebrae are the vertebrae at the upper and lower limits of the curve which tilt most severely toward the concavity of the curve. Once

[1]The adjective "idiopathic" means only that the cause is unknown.

these vertebrae have been selected, one then draws a line along the upper end-plate of the upper body and along the lower end-plate of the lower body [11]. But as already mentioned, most scoliosis appears as a distortion of the spine, and the Cobb angle cannot express such three-dimensional information involving rotation of the vertebra. Also, the Cobb angle depends on the (horizontal) view direction of the doctor, which makes the Cobb angle very prone to error: spines with the same Cobb angle may vary a lot in the real figure. This shows the limitation of this evaluation method.



(a) (b)

Figure 1.3: Cobb angle — (a) Cobb angle based on radiography; (b) Cobb angle calculation [11].

The computer has already been used for the 3D reconstruction from the 2D radiography. Here we want to focus on simulating the internal changes based on the given external data, trying to reduce the use of X-rays. This is what we call non-invasive visualization, diagnosis and prediction by computer. It would not only help the doctor to grasp the ongoing progress of the patient's disease, but also it could help the patient to fully understand his or her condition.

1.2 Available deformation techniques

1.2.1 Spatial deformation

Free-form and smoothing methods

- **Free Form Deformation(FFD) and its extensions**

  Free-Form Deformation as a technique for deforming solid geometric models in a free-form manner was first brought out by T. W. Sederberg and S. R. Parry in 1986 [20]. From that day on this method has attracted considerable interest. The method uses a control lattice, which will be described below. Because of its many advantages FFD has become a standard technique, known as "virtual clay sculpting" suggesting that target solids or surfaces can be shaped with flexibility akin to clay in a sculptor's hands.

  Recently, the method of FFD has been more widely used and developed. Several versions and extensions of the simple FFD have appeared: Extended free-form deformation (EFFD) [9] and Rational free-form deformation (RFFD) [13][2]. But the key point that interests us here is the most basic property of the FFD: it gives a way of deforming space. Just based on this, we chose FFD as our first method to try. Because of its flexibility to simulate any form of deformation, we considered this technique for simulation of the different kinds of deformation of the human torso in the circumstances of idiopathic scoliosis.

- **Smoothing Spline Models**

  Smoothing splines provide a general method of prediction that permits a compromise between smoothness of the predictor, and accuracy of the interpolation of given data. The thin-plate spline is a conventional tool for surface interpolation over scattered data. It involves an elegant algebraic expression for the dependence of the physical bending energy of a thin metal plate under point constraints [5]. This is the second method we tried after

[2] All these methods will be discussed later in Chapter 2.

abandoning the FFD model.

Deformation by direct manipulation

Direct-manipulation deformation is another group of deformation techniques. It realizes the control over the deformation by directly manipulating the object without any intermediate control-lattice. Examples are the space-deformation model defined by Borrel and Bechmann [7] and direct free-form deformation (DFFD) [12].

1.2.2 Physics-based deformation

In 1975, Versprille proposed the Non-Uniform Rational B-Splines (NURBS) [16]. NURBS quickly gained popularity because of their power to represent free-form shapes as well as common analytic shapes, and they were soon incorporated into several commercial modeling systems [21]. However, the drawback of NURBS also showed up: the user is faced with the tedium of indirect shape manipulation through a bewildering variety of geometric parameters; shape design to required specifications by manual adjustment of available geometric degrees of freedom is often elusive and typical design requirements may be stated in both quantitative and qualitative terms [21, 17, 18, 17].

Then an extension of NURBS was introduced: Dynamic NonUniform Rational B-spline (D-NURBS). They extend the basic NURBS to the physics-based models that incorporate mass distributions, internal-deformation energies, and other physical quantities. In this way, the behavior of the deformable model is governed by physical laws, and this on top of the standard geometric foundation makes the whole method more convenient for use.

1.3 Outline of the thesis

In this thesis we have given:

1. a theoretical study of certain methods that might be used for visualization and simulation in the context of scoliosis.

2. a preliminary implementation with a user interface.

3. a generalization of Rohr's anisotropic error measures that seems appropriate in the case of scoliosis (affine-affine matching).

New (although not necessarily profound) mathematical results appearing in this thesis will be indicated within relevant sections, in a footnote.

The rest of the thesis is organized as follows: in Chapter 2, we will discuss the spatial deformation models, especially focusing on the Free-form deformation, which is our first test model, and its extensions. We will describe some of our preliminary test results of the FFD there. Chapter 3 is the theoretical discussion of the Smoothing spline model, which is our second model. In Chapter 4, we will give a slight generalization of the models we tried, i.e., the models for simulation of deformation. We will talk about the interface of our preliminary software in Chapter 5. All the experimental results will come in Chapter 6, together with the analysis of the results. We will give our conclusions in Chapter 7.

Chapter 2

Spatial deformation

Spatial deformation is a transformation technique for 3D geometric data [4]. It has a number of useful traits, such as continuity guarantees (ensuring that the corrected portion of the model is still "smoothly" related to the uncorrected area), and local/global control over the transformation, allowing for the preservation of fine detail in the areas being corrected [15]. It is a deformation technique independent of the underlying object representation. Here we separate the techniques into two groups: deformation using free-form and smoothing methods, which will be discussed in Section 2.1; and deformation realized by directly manipulating the object, which will show up in Section 2.2 [4].

2.1 Thin-plate spline model

Thin-plate spline (TPS) model is one of the deformation models that uses marker points to control the deformation [19]. Each time the user change the positions of the marker points, the deformation model changes the coefficient vectors, which defines the final deformation figure. We will discuss this model in detail in Chapter 3. Kriging is another method, which also realizes spatial deformation [22]. It is quite closely related to TPS [23].

2.2 Free-form deformation and its extensions

2.2.1 Theoretical foundation

Free-Form Deformation (FFD) belongs to the group of deformation models requiring a deformation tool, such as a lattice. A lattice is represented by a trivariate volume regularly subdivided and defined by a three-dimensional array of control points [20]. The object that has to be deformed is embedded inside the lattice. To deform an object the user deforms the lattice by moving its control points. Any point lying inside the lattice is deformed according to the lattice deformation. In particular, the deformation of an object inside the lattice follows the displacement of the lattice control points [20]. This is the unique aspect of FFD: instead of deforming the object directly, the object is embedded in a space that is then deformed and the deformation is realized in an indirect way. A good physical analogy for FFD is to consider a parallelepiped of clear, flexible plastic in which is embedded an object, or several objects, which we wish to deform [20]. The object is imagined to also be flexible, so that it deforms along with the plastic that surrounds it [6]. Mathematically, the FFD is defined in terms of a tensor-product trivariate Bernstein polynomial. We begin [20] by imposing a local coordinate system on a parallelepiped region; then, any point $\mathbf{X}$ that has (s, t, u) coordinates in this system is transformed as:

$$\mathbf{X} = \mathbf{X}_0 + s\mathbf{S} + t\mathbf{T} + u\mathbf{U}. \tag{2.1}$$

The (s, t, u) coordinates of $\mathbf{X}$ can then be found using linear algebra. For any point interior to the parallelepiped, we have $0 < s < 1$, $0 < t < 1$, $0 < u < 1$ and a vector solution is

$$s = \frac{(\mathbf{T} \times \mathbf{U}) \cdot (\mathbf{X} - \mathbf{X}_0)}{(\mathbf{T} \times \mathbf{U}) \cdot \mathbf{S}}, t = \frac{(\mathbf{U} \times \mathbf{S}) \cdot (\mathbf{X} - \mathbf{X}_0)}{(\mathbf{U} \times \mathbf{S}) \cdot \mathbf{T}}, u = \frac{(\mathbf{S} \times \mathbf{T}) \cdot (\mathbf{X} - \mathbf{X}_0)}{(\mathbf{S} \times \mathbf{T}) \cdot \mathbf{U}}. \tag{2.2}$$

Next we impose a grid of control points $\mathbf{P}_{ijk}$ on the parallelepiped. These form $l + 1$ planes in the $\mathbf{S}$ direction, $m + 1$ planes in the $\mathbf{T}$ direction, and $n + 1$ planes in the $\mathbf{U}$ direction. These control points lie on a lattice, and their locations are

defined by

$$\mathbf{P}_{ijk} = \mathbf{X}_0 + \frac{i}{l}\mathbf{S} + \frac{j}{m}\mathbf{T} + \frac{k}{n}\mathbf{U}, \tag{2.3}$$

where $i = 0, \ldots, l$, $j = 0, \ldots, m$, and $k = 0, \ldots, n$.

The deformation is specified by moving the $\mathbf{P}_{ijk}$ from their original latticial positions. The deformed position $\mathbf{X}_{ffd}$ of an arbitrary point $\mathbf{X}$ is found by first computing its (s, t, u) coordinates from equation (2.2), and then evaluating the vector

$$\mathbf{X}_{ffd} = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \left[\sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \left[\sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k \mathbf{P}_{ijk} \right] \right] \tag{2.4}$$

where $\mathbf{X}_{ffd}$ is a vector containing the Cartesian coordinates of the deformed point, and where $\mathbf{P}_{ijk}$ is a vector containing the Cartesian coordinates of the control points [20].

2.2.2 FFD in our application

With the FFD method[1], provided two groups of control points which form the two control lattices (one original and one deformed) and the point-based original figure, we can get the final deformed figure.

In our application we need one extra step — to get the deformed control lattice, given the two groups of marker points (one original and one deformed) on the object. What we want to achieve is to choose the deformed control lattice so as to deform the whole space containing all the original marker points to either interpolate or approximate all the deformed marker points. So the whole process of this method is:

1. Get the deformed control lattice:

   Construct the original control lattice for the whole object, and get the coordinates of all the original control points which are just the vertices of the original control lattice. Match the transformed coordinates of the original

[1] The approach described here, for controlling the FFD, is new. It was, however, unsuccessful, for the reasons given at the end of the subsection.

marker points and that of the deformed marker points to get the coordinates of the deformed control points, which form the deformed control lattice, as[2]:

$$[...f^{\star}(s_q,t_q,u_q)...]_{3\times M} = [...P_r^{\star}...]_{3\times N} \begin{bmatrix} W_{11}W_{12}...W_{1M} \\ W_{21}W_{22}...W_{2M} \\ \cdots \\ W_{N1}W_{N2}...W_{NM} \end{bmatrix}_{N\times M} \tag{2.5}$$

where

$$W_{rq} = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \left[\sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \left[\sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k \right] \right],$$

$r = 1, \ldots, N = (l+1)(m+1)(n+1)$, and $q = 1, \ldots, M$; M is the number of control points. Here the original marker points act as the weight coefficient for the deformed control points by forming the weight matrix. The unknowns in (2.5) are the control point $P_r^{\star}$ of the deformed lattice.

2. Get the final deformed figure:

   Use the deformed control points to calculate the deformed coordinates of all the points on the object using equation (2.4).

Advantages of FFD

The advantages of FFD as a method of space deformation are quite obvious. First, FFD is a representation-independent deformation technique. It can be applied not only to any solid-modeling system, such as Constructive Solid Geometry (CSG) or Boundary Representation (B-rep), but also surfaces or polygonal data [20]. Second, the deformation can be formulated in terms of any polynomial basis, such as tensor-product B-splines or non-tensor-product Bernstein polynomials, which means the deformation can be applied either globally or locally, based on need. Each time we can choose the best and most convenient way to express the deformation. Third, the deformation method works by space deformation. This is the most important point in our application, because our purpose is to choose the deformed control lattice so as to deform the line (corresponding to the spine

[2]This is in effect the inverse of the standard FFD.

in the application) in the middle according to the degree of deformation of all the points on the cylinder (corresponding to the torso in the real application). This can keep the whole deformation space continuous such that both parts can have consistent deformations.

Disadvantages of FFD

The disadvantage of the FFD method itself is that it is difficult to control [8]. It is a global deformation method in that it takes a space and bends it. According to [25], it works well in simple cases, but fails when complex, subtle, local deformations are required over a surface. Plus, this method requires editing a lattice to match a specific object and deforming it to produce the desired object deformation. This process may prove difficult, particularly when the object's shape does not fall into a simple combination of predefined types of lattices, and when the deformation is so complex that the correspondence between the lattice deformation and the object deformation is not straightforward [6].

The disadvantages of FFD in our application are quite crucial and finally led us to abandon this method. First, as part of our application actually we need the inverse of the standard FFD; this requires calculation of the inverse of a matrix, or at least the LU decomposition. In the either case, we have to worry about matrix singularity, especially in our application: sometimes it is quite possible that more than two original marker points lie on a line. The second disadvantage is related to the number M of control points, which may cause (2.5) to be over- or under-determined. Also, for the method of FFD, the more control points we have, the better deformation we can get, but there is a practical constraint concerning the number of marker points (both the original and the deformed ones). The limited number of marker points led to part of the final deformation being out of control (in regions where there are no control points)[3]. The very poor results of our preliminary experiments with this method led us to change our research to other methods of deformation. We give a brief description of these experiments

[3]This will be discussed later in Section 2.1.3.

in the next subsection.

2.2.3 Preliminary experimental results using FFD

We begin with the local coordinate system (s, t, u) on a parallelepiped region defined by $\mathbf{X}_0$, $\mathbf{S}$, $\mathbf{T}$ and $\boldsymbol{U}$ [20, p. 153]. There are $N = (l+1)(m+1)(n+1)$ control points P_{ijk} defining the original control lattice grid as equation (2.3). Here we take $l = m = n = 1$ so that in total there are $2 \cdot 2 \cdot 2 = 8$ control points. Within this parallelepiped region, an open cylinder is defined, in terms of the (s, t, u) coordinates (see Fig. 2.1). In the simplest case, this might of course just

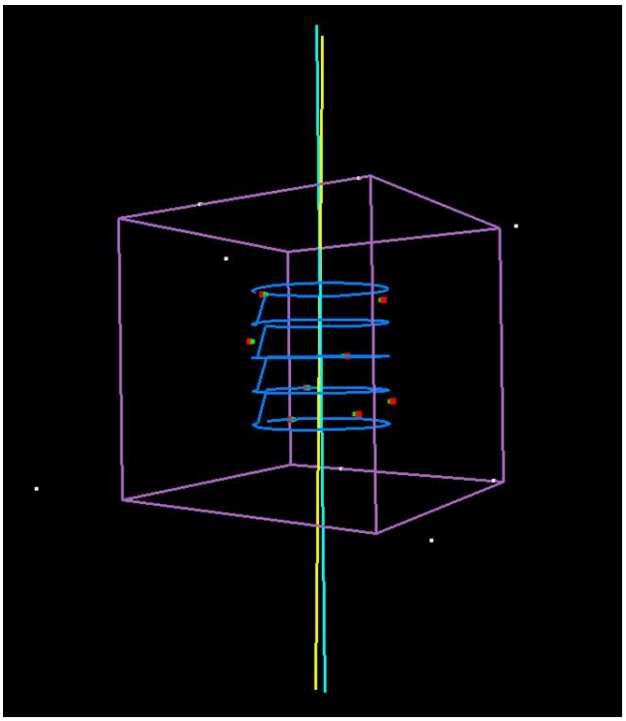


Figure 2.1: Preliminary test model for FFD model — a blue open cylinder, which represents the deformation target (original cylinder), embedded inside a cube, which is the original control lattice: white points represent the deformed control points; green points represent the original marker points; red points represent the deformed marker points; gray cylinder (not shown here) will represent the deformed cylinder; the turquoise line is the original spine and the yellow line represents the deformed spine.

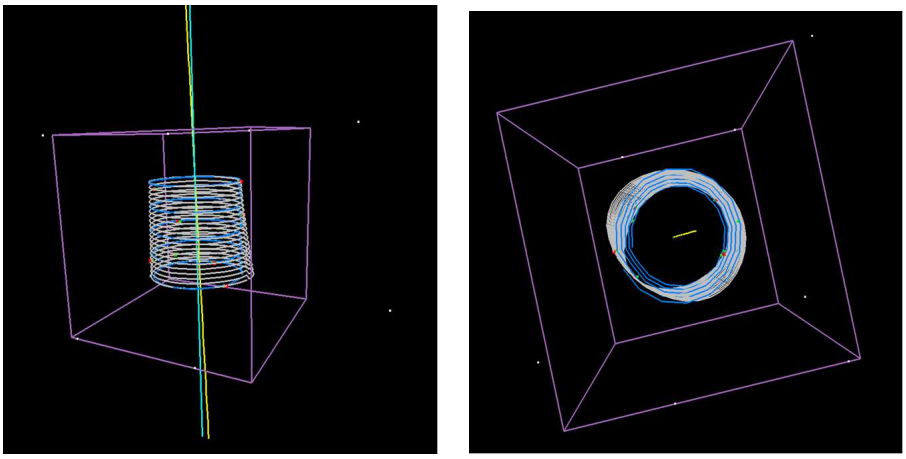
be the ordinary Cartesian system:

$$\boldsymbol{X}_0 = [0,0,0], \boldsymbol{S} = [1,0,0], \boldsymbol{T} = [0,1,0], \boldsymbol{U} = [0,0,1].$$

We would then like to find new values of $\boldsymbol{P}_{ijk}$ for all the points on the cylinder which will cause given marker points on the (original) open cylinder to be transformed into given (observed) values in (s,t,u)-space R^3. In our experiments, we chose the number of marker points $M = 8$ to simplify the calculation, since if the number of marker points M is smaller than the number of control points N, the solution may not be unique. When M is greater than N, the solution may not exist at all, and the Least Square technique needs to be applied.

Here are some of our preliminary test results:

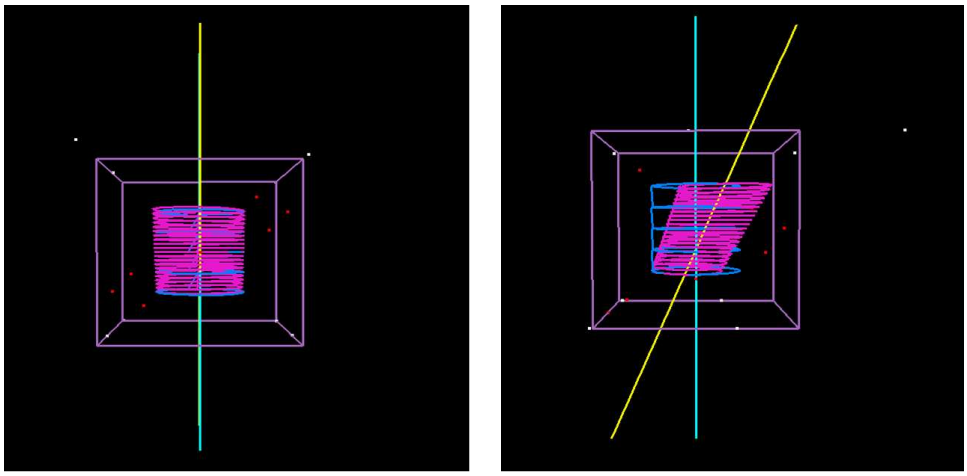
1. Test case 1: Here we choose the number of marker points $M = 8$, move two of the control points outward (see Fig.2.2 (a), showing the two white control points on the right side) to get some intuitive idea of how the method works. As we can see just from the figures, the deformation works quite in the way expected: one side of the lower part of the cylinder moves out, and the internal vertical line switches to the side of the cylinder, which extends out a little bit; the rest of the cylinder remains almost unchanged.



(a) (b)

Figure 2.2: FFD test case 1 — move two control points outward, (a) is the sideview; and (b) is the topview.

2. Test case 2: This is our second test with this model. We use the same number of marker points, but move four of the eight control points, two outward and two inward (see Fig. 2.3). The deformation result is pretty good. In the direction where the control points move, the object is deformed, see Fig. 2.3 (b); the whole cylinder is sheared according to the upper two outward-moved control points and the lower two inward-moved control points. And, in the direction where there is no change in the positions of the control points, the cylinder remains unchanged: see Fig. 2.3 (a).



(a) (b)

Figure 2.3: FFD test case 2 — move four control points (two outward and two inward), (a) is the view from the unchanged side; and (b) is the view from the changed side. Here, the blue cylinder is the original cylinder and the pink one represents the deformed one.

3. Problem with the FFD model. During the time we did the preliminary test for the FFD model, we met very serious problems, which led us abandon this method finally, namely, there appear large distortions in the deformed figure (see Fig. 2.4).

2.2.4 Extended versions of FFD model

In spite of the disadvantages of the FFD model as we mentioned in the previous section, this method is still quite powerful. And several extended versions of the

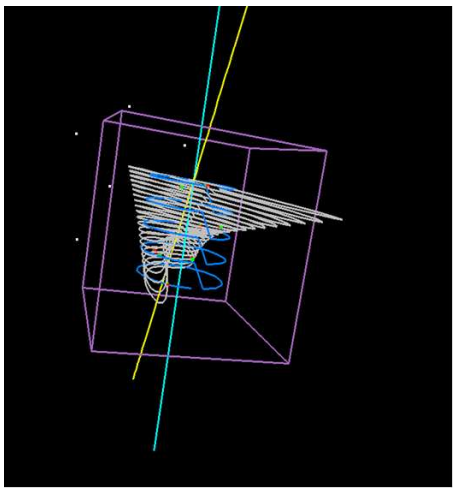


Figure 2.4: Problem with the FFD model: there appear large distortions in the deformed figure.

basic FFD model have appeared:

1. Extended free-form deformation (EFFD)

   Extended free-form deformation; this [9] works on the same mathematical foundations but increases the power of the modeling system by using any shape of initial lattices or combining them [4].

2. Rational free-form deformation (RFFD)

   Rational free-form deformation [13] is another extension of FFD. It allows incorporation of weights defined at each control point of the parallelepiped lattice. However, when the weights at each control point are unity, the deformations are equivalent to the FFD. To control the deformation, the user either moves the lattice control points or modifies their associated weights. The coordinates of a point are computed in the lattice parameter space before editing the lattice control points [4].

3. Direct free-form deformation (DFFD)

   Direct free-form deformation [12] is slightly different from the other two extensions. Part of this method belongs to the category of direct-manipulation deformation, but since it is also an extension of FFD, we list it here. DFFD also consists in embedding the object that has to be deformed inside a trivari-

ate lattice defined by an array of control points. The object deformation follows the lattice deformation but the displacements of the lattice control points are computed from actions such as: "move this point of the object to there" [4].

In summary, free-form deformation, extended free-form deformation, and rational free-form deformation techniques rely on the same mathematical formulation. The deformed area as well as the shape of the deformation inside the deformed area depend on the polynomials. The deformed area is either the whole lattice or a part of it. To localize a deformation, the original lattice should either include a limited part of the object or its subdivision in chunks to be modified [4].

2.3 Direct-manipulation deformation

Direct manipulation deformation is the other group of deformation techniques. Two models belong to this group: space deformation, which will be described here, and direct free-form deformation, which we have discussed in the previous section together with other extended versions of free-form deformation [4].

Introduced in 1991, the space deformation model defined of Borrel and Bechmann [7] provides direct manipulation of the object. Intermediate tools, such as lattices, are no longer required. The deformation of the object is simply specified by the displacement of arbitrary selected points called *constraints*. The size and the boundary of a bounding box centered around each constraint point allows control of the extent of the deformation. Depending on this extent, the whole object can be included (global deformation) or only a limited area around the constraint point (local deformation). A large range of deformation shapes such as arbitrarily shaped bumps can be designed using this technique [4].

Chapter 3

Smoothing Spline Models

Spline models can involve functions based on different index sets $\mathcal{T}$. Here mainly two cases interest us: $\mathcal{T} = [0, 1]$ and $\mathcal{T} = R^d$.

In many practical applications, using only rigid transformations is obviously far from satisfactory, and the lack of accuracy of the result may cause the whole method to be useless. This makes elastic transformations, which allow for local adaptation and which are constrained to some kind of continuity or smoothness, quite attractive. Thin-Plate Spline (TPS) is one of the methods currently being used and studied widely.

One of the goals of this thesis is to show how to adapt the TPS methods presented in the literature to our particular problem, since the link is not always obvious. This will sometimes involve generalization (for example, increasing the dimension from 2 to d, $d > 2$), and sometimes specialization (for example, assuming that data observations are always paired, or, in general, "grouped d-wise"). We will also sometimes make changes to arbitrary factors (for example, multiplying a matrix by a factor, when this change can be compensated by an arbitrary weighting factor elsewhere in the formulation). The purpose of doing this is to arrive at a similar notation for several different methods (originally presented in the literature using different formulations and notations), so that their similarities and differences can be observed.

3.1 Note on terminology

Based on the fact that certain technical terms may appear in different situations with different meanings, in this first section we make a note on terminology which will help clarify the discussion in later sections.

3.1.1 Interpolation and time-interpolation

A continuous function $y = y(x)$ can be used to represent the $n+1$ data values by passing through all the $n+1$ points $y_i, i = 0, \ldots, n$. Then one can find the value of y at any other value of x. This is *interpolation*, as opposed to approximation. In our explanation later on, the same term "interpolation" has a different meaning (interpolation as opposed to extrapolation). To differentiate these concepts, we use "time-interpolation" for the second one.

3.1.2 Recapitulation

- **Time-interpolation and time-extrapolation**

  Time-interpolation and time-extrapolation are two terms we used for the [0,1] index model. Time-interpolation refers to an estimation of a value within (viewed in the [0,1] time domain) two known values in a sequence of values. This is opposed to time-extrapolation, which is an estimation of a value based on extending a known sequence of values or facts beyond the range in [0,1] that is certainly known; this is often called "prediction".

- **Interpolation and approximation**

  Interpolation and approximation are terms used for both the [0,1] index model[1] and the R^d model. They are just standard terms: interpolation represents the case where the continuous function passes through all the given points, and approximation represents the case where a smoothing parameter

[1]Interpolation and approximation can be used in our application to do time-interpolation or time-extrapolation of the torso and spine, given marker points data, over a single time interval.

is introduced into the function, so that the continuous function just approximates the given set of data. The level of approximation depends on the value we choose for the smoothing parameter. In the approximation case, the form of the function usually is more smooth.

3.2 The case $\mathcal{T} = [0, 1]$

3.2.1 Mathematical outline

The bivariate Thin-plate spline model based on the [0,1] index set is described as [14, 24]:

$$y_{ki} = f_k(t_{ki}) + \epsilon_{ki}, k = 1, 2; i = 1, ...n_k, \tag{3.1}$$

where the i^{th} response of the k^{th} variable y_{ki} is generated by the k^{th} function f_k evaluated at the design point t_{ki} plus a random error ϵ_{ki}. Here it is assumed that $\epsilon_{ki} \overset{\text{iid}}{\sim} N(0, \sigma_k^2)$ for fixed $k = 1, 2$ and $Corr(\epsilon_{1i}, \epsilon_{2j}) = \rho$ if y_{1i} and y_{2j} are a pair and zero otherwise; also the domain of both functions is $[0, 1]$ and $f_k \in W_2$, where

$$W_2 = \{f : f, f' \text{ absolutely continuous, } \int_0^1 (f''(t))^2 dt < \infty\}.$$

This method can be quite easily extended to multi-response data: just let $k = 1, 2, \cdots, d$, where d is the dimension. In our case[2], the data corresponds to the coordinates of points in space, so ungrouped data ("unpaired data" in the bivariate case) is not relevant. Therefore, here we assume the variable d (dimension) has the value 3, and that the number of observations n_k and the trade-off parameter λ_k (see below) for the three groups are the same for each $k = 1, 2, 3$. Therefore we may then use n instead of n_k and λ instead of λ_k. Plus, we sometimes use x, y, z to represent the three components of each observation, which may be more intuitive for understanding of our particular application. Denote $\mathbf{t}_k = (t_{k1}, ..., t_{kn})^T$, $\mathbf{f}_k = (f_k(t_{k1}), ..., f_k(t_{kn}))^T$, $\mathbf{y}_k = (y_{k1}, ..., y_{kn})^T$, $\epsilon_k = (\epsilon_{k1}, ..., \epsilon_{kn})^T$, $\mathbf{f} = (\mathbf{f}_1^T, \mathbf{f}_2^T, \mathbf{f}_3^T)^T$, and $\mathbf{y} = (\mathbf{y}_1^T, \mathbf{y}_2^T, \mathbf{y}_3^T)^T$, where the superscript ^{T} refers to transpose. We will take

[2]The development in this section, showing how [24] can be adapted to our problem, is new, including the derivation of (3.6), (3.7) and (3.10), and the treatment of the case $\lambda = 0$.

the inverse of the co-variance matrix W^{-1} to be the direct sum of three matrices of the form

$$\begin{pmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_n^2 \end{pmatrix}. \tag{3.2}$$

This matrix can be obtained from the matrix W^{-1} in [24] by assuming that the correlation $\rho = 0$, that the matrix there is multiplied by the factor $\theta = \sigma_1\sigma_2$, and that the dimension has been raised from 2 to 3. The multiplicative factor can be compensated by a change in the arbitrary constant λ discussed below. In general it may be useful to permit non-zero off-diagonal elements, and different values for the σ_i in each of the three blocks. In this case the nine values corresponding to the i^{th} row and the j^{th} column of each of the three blocks should form a 3×3 covariance matrix (see Section 3.3.2).

The function $\mathbf{f}_k$ can be estimated by solving the following penalized weighted least-squares problem:

$$\min_{f_1,f_2,f_3 \in W_2} \{(\mathbf{y}-\mathbf{f})^T W (\mathbf{y}-\mathbf{f}) + \lambda \int_0^1 (f_1''(t))^2 dt + \lambda \int_0^1 (f_2''(t))^2 dt + \lambda \int_0^1 (f_3''(t))^2 dt\}, \tag{3.3}$$

where the first term is the weighted least-squares and the remaining terms are penalties for the roughness of the functions. The parameters λ control the trade-off between goodness-of-fit and the smoothness of the estimates and are referred to as smoothing parameters [24]. Now the solution giving the prediction function is [23]

$$\hat{f}_k(t) = \sum_{\nu=1}^{m} d_{k\nu}\phi_\nu(t) + \sum_{i=1}^{n} c_{ki} R^1(t, t_{ki}), k = 1, 2, 3; \tag{3.4}$$

it can be expressed in vector form as $[\ x(t), y(t), z(t)\ ] =$

$$[\ \phi_1(t), \cdots, \phi_m(t)\ ] \begin{bmatrix} d_{x1}, d_{y1}, d_{z1} \\ \vdots \\ d_{xm}, d_{ym}, d_{zm} \end{bmatrix} + [\ R^1(t,t_1), \cdots, R^1(t,t_n)\ ] \begin{bmatrix} c_{x1}, c_{y1}, c_{z1} \\ \vdots \\ c_{xn}, c_{yn}, c_{zn} \end{bmatrix}, \tag{3.5}$$

in which

$$\phi_\nu(t) = t^{\nu-1}/(\nu-1)!, \nu = 1, \ldots, m$$

defines the m-dimensional space of polynomials of degree $m-1$ or less, spanned by $\phi_1, \cdots, \phi_m$, and

$$R^1(s,t) = k_2(s)k_2(t) - k_4(s-t)$$

where $k_\nu(\cdot) = B_\nu(\cdot)/\nu!$, and B_ν is the ν'th Bernoulli polynomial: $B_2(x) = x^2 - x + \frac{1}{6}$, $B_4(x) = x^4 - 2x^3 + x^2 - \frac{1}{30}$. The vectors

$$\boldsymbol{d} = (d_{x1}, \cdots, d_{xm}, d_{y1}, \cdots, d_{ym}, d_{z1}, \cdots, d_{zm})^T$$
$$\boldsymbol{c} = (c_{x1}, \cdots, c_{xn}, c_{y1}, \cdots, c_{yn}, c_{z1}, \cdots, c_{zn})^T$$

which are chosen to minimize (3.3) when

$$\boldsymbol{f} = \mathrm{S}\ \boldsymbol{c} + \mathrm{T}\ \boldsymbol{d}.$$

are stated in [24] to be the solutions[3] to

$$\begin{pmatrix} T^TWT & T^TWS \\ SWT & SWS + \lambda S \end{pmatrix} \begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{c} \end{pmatrix} = \begin{pmatrix} T^TWy \\ SWy \end{pmatrix} \tag{3.6}$$

where $T_k = \phi_\nu(t_{ki})_{i=1\ \nu=1}^{n\quad m}, k = 1, 2, 3$; in our special case T_k is independent of k. Since for us the data represents the components of a position on the torso or spine, at a specific time, we have $t_{1j} = t_{2j} = t_{3j}$, (data paired, or "grouped 3-wise") and we just use t_j for all three. Here,

$$T = \begin{bmatrix} \phi_1(t_1)\,\phi_2(t_1)\,\ldots\,\phi_m(t_1) & & \\ \phi_1(t_2)\,\phi_2(t_2)\,\ldots\,\phi_m(t_2) & 0 & 0 \\ \cdots & & \\ \phi_1(t_n)\,\phi_2(t_n)\,\cdots\,\phi_m(t_n) & & \\ & \phi_1(t_1)\,\phi_2(t_1)\,\ldots\,\phi_m(t_1) & \\ 0 & \phi_1(t_2)\,\phi_2(t_2)\,\ldots\,\phi_m(t_2) & 0 \\ & \cdots & \\ & \phi_1(t_n)\,\phi_2(t_n)\,\cdots\,\phi_m(t_n) & \\ & & \phi_1(t_1)\,\phi_2(t_1)\,\ldots\,\phi_m(t_1) \\ 0 & 0 & \phi_1(t_2)\,\phi_2(t_2)\,\ldots\,\phi_m(t_2) \\ & & \cdots \\ & & \phi_1(t_n)\,\phi_2(t_n)\,\cdots\,\phi_m(t_n) \end{bmatrix}.$$

Also, $S = diag(S_k)$ where $S_k = R^1(t_{ki}, t_{kj})_{i=1\ j=1}^{n\quad n}, k = 1, 2, 3$, and, similarly, S_k is independent of k. Using the same simplification for t_{ki},

$$S = \begin{bmatrix} R^1(t_1,t_1)\,R^1(t_1,t_2)\,\ldots\,R^1(t_1,t_n) & & \\ R^1(t_2,t_1)\,R^1(t_2,t_2)\,\ldots\,R^1(t_2,t_n) & 0 & 0 \\ \cdots & & \\ R^1(t_n,t_1)\,R^1(t_n,t_2)\,\ldots\,R^1(t_n,t_n) & & \\ & R^1(t_1,t_1)\,R^1(t_1,t_2)\,\ldots\,R^1(t_1,t_n) & \\ 0 & R^1(t_2,t_1)\,R^1(t_2,t_2)\,\ldots\,R^1(t_2,t_n) & 0 \\ & \cdots & \\ & R^1(t_n,t_1)\,R^1(t_n,t_2)\,\ldots\,R^1(t_n,t_n) & \\ & & R^1(t_1,t_1)\,R^1(t_1,t_2)\,\ldots\,R^1(t_1,t_n) \\ 0 & 0 & R^1(t_2,t_1)\,R^1(t_2,t_2)\,\ldots\,R^1(t_2,t_n) \\ & & \cdots \\ & & R^1(t_n,t_1)\,R^1(t_n,t_2)\,\cdots\,R^1(t_n,t_n) \end{bmatrix}$$

and $\boldsymbol{c}$ and $\boldsymbol{d}$ are chosen to minimize (3.3) when

[3] To avoid a conflict in notation in the later comparison between different methods, we changed the notations here, writing S instead of Σ.

$$\boldsymbol{f} = \mathrm{S}\ \boldsymbol{c} + \mathrm{T}\ \boldsymbol{d}.$$

Here $\boldsymbol{f}$ is $(3n \times 1)$, S is $(3n \times 3n)$, $\boldsymbol{c}$ is $(3n \times 1)$, T is $(3n \times 3m)$, and $\boldsymbol{d}$ is $(3m \times 1)$. Then, the objective function in (3.3) can be written

$$[\boldsymbol{y} - (S\boldsymbol{c} + T\boldsymbol{d})]^T W[\boldsymbol{y} - (S\boldsymbol{c} + T\boldsymbol{d})] + \boldsymbol{c}^T \lambda S\boldsymbol{c}. \tag{3.7}$$

Expanding this, dropping the term $\boldsymbol{y}^T W\boldsymbol{y}$ (which does not depend on $\boldsymbol{c}$ or $\boldsymbol{d}$), and dividing by 2 gives:

$$-\boldsymbol{y}^T WS\boldsymbol{c} - \boldsymbol{y}^T WT\boldsymbol{d} + \tfrac{1}{2}\boldsymbol{c}^T \lambda S\boldsymbol{c} + \boldsymbol{c}^T SWT\boldsymbol{d} + \tfrac{1}{2}\boldsymbol{d}^T T^T WT\boldsymbol{d} + \tfrac{1}{2}\boldsymbol{c}^T SWS\boldsymbol{c}.$$

Differentiating with respect to $\boldsymbol{c}$ and setting the derivative to zero gives:

$$-SW\boldsymbol{y} + [SWS + \lambda S]\boldsymbol{c} + SWT\boldsymbol{d} = 0,$$

and differentiating with respect to $\boldsymbol{d}$ and setting the derivative to zero gives

$$-T^T W\boldsymbol{y} + T^T WS\boldsymbol{c} + T^T WT\boldsymbol{d} = 0.$$

These equations are exactly those of (3.6).

It is also stated in [24] that a solution to

$$\begin{cases} (S + \lambda W^{-1})\boldsymbol{c} + T\boldsymbol{d} = \boldsymbol{y} \\ T^T\boldsymbol{c} = \boldsymbol{0} \end{cases} \tag{3.8}$$

provides a solution to (3.6)[4]. To see this, multiply the first equation of (3.8) on the left by SW, to obtain the second equation of (3.6). Now, multiplying the same equation on the left by $T^T W$, we obtain

$$T^T WT\boldsymbol{d} + (T^T WS + \lambda T^T)\boldsymbol{c} = T^T W\mathbf{y}$$

which, given the second equation of (3.8), yields the first equation of (3.6).

To calculate the coefficients $\boldsymbol{c}$ and $\boldsymbol{d}$, we use the following transformations:

$$\tilde{S} = \tfrac{1}{\lambda}S, \lambda \neq 0$$
$$\tilde{\boldsymbol{c}} = \lambda\boldsymbol{c}.$$

[4]Equation (3.7) is simplified relative to equation (6) of [24], since $\lambda_1 = \lambda_2 = \lambda_3$.

If $\lambda = 0$, then, revising the derivation following (3.7), (3.8) will be replaced by the single equation $S\boldsymbol{c} + T\boldsymbol{d} = \boldsymbol{y}$, which guarantees (3.6). These equations are apparently inconsistent for $n > 2$.

If $\lambda \neq 0$, then equations (3.8) can be expressed in matrix-vector form as:

$$\left(\begin{bmatrix} S_1/\lambda & 0 & 0 \\ 0 & S_2/\lambda & 0 \\ 0 & 0 & S_3/\lambda \end{bmatrix} + \begin{bmatrix} W_x^{-1} & 0 & 0 \\ 0 & W_y^{-1} & 0 \\ 0 & 0 & W_z^{-1} \end{bmatrix} \right) \begin{bmatrix} \mathbf{c}_x\lambda \\ \mathbf{c}_y\lambda \\ \mathbf{c}_z\lambda \end{bmatrix} + \begin{bmatrix} T_1 & 0 & 0 \\ 0 & T_2 & 0 \\ 0 & 0 & T_3 \end{bmatrix} \begin{bmatrix} d_{x1} \\ \vdots \\ d_{xm} \\ d_{y1} \\ \vdots \\ d_{ym} \\ d_{z1} \\ \vdots \\ d_{zm} \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ y_1 \\ \vdots \\ y_n \\ z_1 \\ \vdots \\ z_n \end{bmatrix}$$

$$\begin{bmatrix} T_1 & 0 & 0 \\ 0 & T_2 & 0 \\ 0 & 0 & T_3 \end{bmatrix}^T \begin{bmatrix} \mathbf{c}_x\lambda \\ \mathbf{c}_y\lambda \\ \mathbf{c}_z\lambda \end{bmatrix} = 0. \tag{3.9}$$

Here all the $W_x^{-1}, W_y^{-1}, W_z^{-1}$ are $n \times n$ matrices. Let

$$T_k = (Q_{k1}, Q_{k2}) \begin{pmatrix} R_k \\ 0 \end{pmatrix}, k = 1, 2, 3$$

be the QR decomposition of T_k, and let

$$\begin{aligned} Q_1 &= diag(Q_{11}, Q_{21}) \\ Q_2 &= diag(Q_{12}, Q_{22}) \\ R &= diag(R_1, R_2) \\ B &= \tilde{\Sigma} + W^{-1}. \end{aligned}$$

The solution finally is [24]

$$\begin{aligned} \tilde{\mathbf{c}} &= Q_2(Q_2^T B Q_2)^{-1} Q_2^T \mathbf{y}, \\ R\mathbf{d} &= Q_1^T(\mathbf{y} - B\tilde{\mathbf{c}}). \end{aligned} \tag{3.10}$$

3.2.2 Permuting variables for later comparison

As illustrated in the previous section, all the matrices for the functions in the case of [0,1] index are ordered based on components, which means they are formed with the order of x, y and z coordinates in our special case. In this section we will show how these matrices can be permuted to another form which is based

on each single point. Our main purpose for doing this is for the later comparison of the model using [0,1] index and the model using R^d (see Section 3.4). All the components in the equations

$$\begin{cases} (S + \lambda W^{-1})\boldsymbol{c} + T\boldsymbol{d} = \boldsymbol{y} \\ T^T\boldsymbol{c} = \mathbf{0} \end{cases} \tag{3.11}$$

can be permuted to give:

$$T = \begin{bmatrix} \phi_1(t_1)\,0\,0 & \phi_2(t_1)\,0\,0 & \cdots & \phi_m(t_1)\,0\,0 \\ 0\,\phi_1(t_1)\,0 & 0\,\phi_2(t_1)\,0 & \cdots & 0\,\phi_m(t_1)\,0 \\ 0\,0\,\phi_1(t_1) & 0\,0\,\phi_2(t_1) & & 0\,0\,\phi_m(t_1) \\ \vdots & \vdots & & \vdots \\ \phi_1(t_n)\,0\,0 & \phi_2(t_n)\,0\,0 & & \phi_m(t_n)\,0\,0 \\ 0\,\phi_1(t_n)\,0 & 0\,\phi_2(t_n)\,0 & \cdots & 0\,\phi_m(t_n)\,0 \\ 0\,0\,\phi_1(t_n) & 0\,0\,\phi_2(t_n) & & 0\,0\,\phi_m(t_n) \end{bmatrix},$$

$$S = \begin{bmatrix} R^1(t_1,t_1)\,0\,0 & R^1(t_1,t_2)\,0\,0 & \cdots & R^1(t_1,t_n)\,0\,0 \\ 0\,R^1(t_1,t_1)\,0 & 0\,R^1(t_1,t_2)\,0 & \cdots & 0\,R^1(t_1,t_n)\,0 \\ 0\,0\,R^1(t_1,t_1) & 0\,0\,R^1(t_1,t_2) & & 0\,0\,R^1(t_1,t_n) \\ & \cdots & \cdots & \\ R^1(t_n,t_1)\,0\,0 & R^1(t_n,t_2)\,0\,0 & & R^1(t_n,t_n)\,0\,0 \\ 0\,R^1(t_n,t_1)\,0 & 0\,R^1(t_n,t_1)\,0 & \cdots & 0\,R^1(t_n,t_n)\,0 \\ 0\,0\,R^1(t_n,t_1) & 0\,0\,R^1(t_n,t_2) & & 0\,0\,R^1(t_n,t_n) \end{bmatrix},$$

$$\mathbf{d} = \begin{bmatrix} d_{x1} \\ d_{y1} \\ d_{z1} \\ \vdots \\ d_{xm} \\ d_{ym} \\ d_{zm} \end{bmatrix}, \mathbf{c} = \begin{bmatrix} c_{x1} \\ c_{y1} \\ c_{z1} \\ \vdots \\ c_{xn} \\ c_{yn} \\ c_{zn} \end{bmatrix}, \mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_n \end{bmatrix}.$$

Later on we will use this permuted version for the comparison between the [0,1] index model and the R^d index model.

3.2.3 Time-interpolation

Time-interpolation is our first application of the Thin-plate spline model on [0,1] index. Our goal is to get some idea of what is going on in between two given states. This is especially useful in our real application — scoliosis. Given the two different states of the patient body, this model can help both the doctor and the patient know how the disease changes.

With the function[5]

$$\boldsymbol{f}_k(t) = \sum_{\nu=1}^{m} d_\nu \phi_\nu(t) + \sum_{i=1}^{n} c_i R^1(t, t_{ki}), k = 1, 2, 3,$$

each time we input the time index list and the coordinates list of one body point (a point in R^3) at all the time points to calculate the coefficient arrays $\boldsymbol{c}$ and $\boldsymbol{d}$ of this body point. Then by changing the value of t we can get its coordinate at any corresponding in-between time.

There is more than one choice for the time index list $[t_1, t_2, \cdots, t_n]$. First, since in the time-interpolation situation, only the start and the final state are involved in the model, we could use these two groups t_{start} and t_{end}, plus the data at the closest time point, as constraints for the calculation of the deformation[6]. This method, appropriate in a test situation, has the advantage that we still have some given data at hand with which we can do some comparison. But its shortcoming is that only using three time points as constraints may actually reduce the overall accuracy of the whole method (if there are more available). Because less available data involved means less information counted, and some state that has a big influence in the interpolation problem may be ignored. Another approach is to use all the available data to do the time-interpolation, that is, all the $[t_1, t_2, \cdots, t_n]$ time points participate in the calculation of time-interpolation. So no matter which state is appointed as the start state and which is the final state, each time the algorithm uses exactly the same group of data as constraints. This method is what we would use when working with the real patient case. But, still for test purposes, we ignore one group of available data to test the accuracy of our method.

Concerning the W^{-1} matrix which contains the variance parameter σ_j^2, in our application, we arbitrarily set all the values of σ_j^2 to be 1, and for the smoothing parameter λ which appears as part of the W^{-1} matrix, we change it as necessary.

[5]For our application, all the data are grouped: the time index lists are the same for all three components.

[6]The inequality $n > m$ is one of the conditions for the model; since $m = 2$, the smallest value for n is 3.

As we discussed before in Section 3.2.1, in the case of [0,1] index Thin-plate spline model, interpolation, in which $\lambda = 0$, is impossible when $m = 2$.

Later in Chapter 6, we will show experimental results for this model, including both the preliminary and real-data test cases.

3.2.4 Time-extrapolation

Time-extrapolation is another application for which we tested the Thin-plate spline model on the [0,1] index set. By definition in the previous section on terminology, extrapolation is just to get something based on what we have, at some future point in time. And this is what attracts us most: prediction. The main idea under this model is to obtain, from certain groups of data, some trend information which helps us to know what will happen next (prediction). Applying this to our scoliosis application, the use would be quite interesting: we might help the doctor to do some prediction of the form of the spine at some later time point.

This time-extrapolation model is not much different from the time-interpolation model since they share the same [0,1] index model. We have the time index list and coordinates list as input, and get the coefficient vectors $\boldsymbol{c}$ and $\boldsymbol{d}$. Then each time we only need to input a value of t which represents the time point at which we want get the extrapolated figure. The only difference here is that under the time-extrapolation model, this input t is greater than 1. For testing purposes, there are two options available for the specification of the time index list. One is simply using the original value of t which is great than 1 as input, and get the coordinates of the point at the extrapolated state. The other one is to compress all the values down within the [0,1] interval:

$$t_{i_{new}} = t_{i_{old}} / t_{i_{extrapolated}}, i = 0, ..., n$$

where, $t_{i_{new}}$ is the "compressed" value of t_i for extrapolation, $t_{i_{old}}$ is the original value of t_i for the current model[7], and $t_{i_{extrapolated}}$ is the value of t_i that is greater

[7]Since for this method, extrapolation will use all the available data, therefore, the time index list is fixed for each test case.

than 1, which represents the target time point. This is the method we used. Similar to the time-interpolation model, there are also two ways available to get the extrapolated result, either based on all the given data or only the last three data points. We chose the former one for reasons similar to those given above the time-interpolation model.

Later, in Chapter 6, we will show the experimental results of this model including both the preliminary and real-data test cases.

3.3 The case $\mathcal{T} = R^d$

Our original motivation for the use of the Thin-plate spline model based on the R^d index was to do some prediction of the internal spine information, given some marker points around the external patient body and the full point-by-point original patient body. But the real application is much more complicated than the simple simulated test model, and there are serious problems with the real data[8]. We have, however, got some conditional good results.

3.3.1 Mathematical outline

Interpolation

Thin-plate spline model for interpolation can be stated as a multivariate interpolation problem [19]: given a number[9] N of corresponding marker points $\boldsymbol{p}_i$ and $\boldsymbol{q}_i$, $i = 1, ...N$ in two images of dimension d, find a continuous transformation $u : R^d \longrightarrow R^d$ within a suitable Hilbert space H of admissible functions, which 1) minimizes a given functional $J : H \longrightarrow R$ and 2) fulfills the interpolation conditions

$$\boldsymbol{q}_i = \boldsymbol{u}(\boldsymbol{p}_i), i = 1, ..., N, \tag{3.12}$$

[8]We will discuss the problems with the real test data later on, in Chapter 6.

[9]To avoid a conflict with the notation n we used for the [0,1] index model, which represents the number of marker points, here we use N for the number of marker points.

while minimizing the functional which represents the bending energy of a thin plate separately for each component $u_k, k = 1, ..., d$

$$J_m^d(\boldsymbol{u}) = \sum_{k=1}^{d} J_m^d(u_k). \tag{3.13}$$

Here, the single functional is

$$J_m^d(u) = \sum_{\alpha_1+...+\alpha_d=m} \frac{m!}{\alpha_1!...\alpha_d!} \times \int_{R^d} (\frac{\partial_m u}{\partial x_1^{\alpha_1}...\partial x_d^{\alpha_d}})^2 d\boldsymbol{x}. \tag{3.14}$$

The solution of minimizing the functional can be written as [19]

$$u(\boldsymbol{x}) = \sum_{\nu=1}^{M} a_\nu \phi_\nu(\boldsymbol{x}) + \sum_{i=1}^{N} w_i U(\boldsymbol{x}, \boldsymbol{p}_i), \tag{3.15}$$

and this is the function used for prediction. It can also be expressed in vector form as :

$$[u_x, u_y, u_z] = \left[\ \phi_1(\mathbf{x}), \cdots, \phi_M(\mathbf{x})\ \right] \begin{bmatrix} a_{1x}, a_{1y}, a_{1z} \\ \vdots \\ a_{Mx}, a_{My}, a_{Mz} \end{bmatrix} + \left[\ U(\mathbf{x}, p_1), \cdots, U(\mathbf{x}, p_N)\ \right] \begin{bmatrix} w_{1x}, w_{1y}, w_{1z} \\ \vdots \\ w_{Nx}, w_{Ny}, w_{Nz} \end{bmatrix}. \tag{3.16}$$

The set of functions ϕ_ν span the space $\Pi^{m-1}(R^d)$ of all the polynomials on R^d up to order $m-1$, which is the nullspace of the functional in (3.14) [19] with values: $\phi_1(\mathbf{x}) = 1$, $\phi_2(\mathbf{x}) = x$, $\phi_3(\mathbf{x}) = y$ and $\phi_4(\mathbf{x}) = z$ in the case $m = 2$. The dimension of the space is $M = \frac{(d+m-1)!}{d!(m-1)!}$ and M must be not greater than N. This is because later on, to get the solution to the prediction function, we need to have the QR decomposition of the P matrix which is of dimension $N \times M$, so a larger value of M does not work here. The basis functions $U(x, p_i)$ depend on (1) the dimension of the domain, (2) the order m of the derivatives in the functionals, and (3) the Hilbert space H of admissible functions [19, 24]. It can be expressed as follows [19]:

$$U(x,p) = \begin{cases} \theta_{m,d}|x-p|^{2m-d} ln|x-p|, & 2m-d \text{ even positive integer} \\ \theta_{m,d}|x-p|^{2m-d}, & \text{otherwise.} \end{cases} \tag{3.17}$$

The constants $\boldsymbol{a} = (a_1, ..., a_M)^T$ and $\boldsymbol{w} = (w_1, ..., w_N)^T$ corresponding the specific column in (3.16) satisfy the following system of linear equations:

$$\begin{aligned} K\boldsymbol{w} + P\boldsymbol{a} &= \boldsymbol{v} \\ P^T\boldsymbol{w} &= \mathbf{0} \end{aligned} \tag{3.18}$$

in which

$$K = \begin{bmatrix} U(\boldsymbol{p}_1,\boldsymbol{p}_1) & U(\boldsymbol{p}_1,\boldsymbol{p}_2) & \cdots & U(\boldsymbol{p}_1,\boldsymbol{p}_N) \\ U(\boldsymbol{p}_2,\boldsymbol{p}_1) & U(\boldsymbol{p}_2,\boldsymbol{p}_2) & \cdots & U(\boldsymbol{p}_2,\boldsymbol{p}_N) \\ & \cdots & & \\ U(\boldsymbol{p}_N,\boldsymbol{p}_1) & U(\boldsymbol{p}_N,\boldsymbol{p}_2) & \cdots & U(\boldsymbol{p}_N,\boldsymbol{p}_N) \end{bmatrix}$$

and

$$P = \begin{bmatrix} \phi_1(\boldsymbol{p}_1) & \phi_2(\boldsymbol{p}_1) & \cdots & \phi_M(\boldsymbol{p}_1) \\ \phi_1(\boldsymbol{p}_2) & \phi_2(\boldsymbol{p}_2) & \cdots & \phi_M(\boldsymbol{p}_2) \\ & \cdots & & \\ \phi_1(\boldsymbol{p}_N) & \phi_2(\boldsymbol{p}_N) & \cdots & \phi_M(\boldsymbol{p}_N) \end{bmatrix}.$$

Now equation (3.18) can be transformed into matrix form as:

$$\begin{bmatrix} K_{11}K_{12}\cdots K_{1N} \\ K_{21}K_{22}\cdots K_{2N} \\ \cdots \\ K_{N1}K_{N2}\cdots K_{NN} \end{bmatrix} \times \begin{bmatrix} w_{1x}w_{1y}w_{1z} \\ w_{2x}w_{2y}w_{2z} \\ \cdots \\ w_{Nx}w_{Ny}w_{Nz} \end{bmatrix} + \begin{bmatrix} P_{11}P_{12}\cdots P_{1M} \\ P_{21}P_{22}\cdots P_{2M} \\ \cdots \\ P_{N1}P_{N2}\cdots P_{NM} \end{bmatrix} \times \begin{bmatrix} a_{1x}a_{1y}a_{1z} \\ a_{2x}a_{2y}a_{2z} \\ \cdots \\ a_{Mx}a_{My}a_{Mz} \end{bmatrix} = \begin{bmatrix} q_{1x}q_{1y}q_{1z} \\ q_{2x}q_{2y}q_{2z} \\ \cdots \\ q_{Nx}q_{Ny}q_{Nz} \end{bmatrix}$$

$$\begin{bmatrix} P_{11}P_{12}\cdots P_{1M} \\ P_{21}P_{22}\cdots P_{2M} \\ \cdots \\ P_{N1}P_{N2}\cdots P_{NM} \end{bmatrix}^T \begin{bmatrix} w_{1x}w_{1y}w_{1z} \\ w_{2x}w_{2y}w_{2z} \\ \cdots \\ w_{Nx}w_{Ny}w_{Nz} \end{bmatrix} = \mathbf{0}. \tag{3.19}$$

Let

$$P = (Q_1 : Q_2)\begin{pmatrix} R \\ 0 \end{pmatrix}$$

be the QR decomposition of P. The solution finally is [23]

$$\begin{aligned} \boldsymbol{w} &= Q_2(Q_2^T K Q_2)^{-1} Q_2^T \boldsymbol{v}. \\ R\boldsymbol{a} &= Q_1^T(\boldsymbol{v} - K\boldsymbol{w}). \end{aligned} \tag{3.20}$$

Approximation

When we want to take into account landmark localization errors, we just extend the basic interpolation approach by weakening the interpolation condition. This can be achieved by introducing a quadratic approximation term in the functional

(3.13) as [19, 23]:

$$J_\lambda(\boldsymbol{u}) \equiv \frac{1}{N}\sum_{i=1}^{N}\frac{\parallel \boldsymbol{q}_i - \boldsymbol{u}(\boldsymbol{p}_i) \parallel^2}{\sigma_i^2} + \lambda J_m^d(\boldsymbol{u}). \tag{3.21}$$

The first term of the functional in (3.21), which is called the data term, measures the sum of the quadratic Euclidean distances between the deformed marker points $\boldsymbol{p}_i$ and the given marker points $\boldsymbol{q}_i$. Each distance is weighted by the variances σ_i^2 representing landmark localization errors. The second term in (3.21) measures the smoothness of the resulting transformation. The minimization of the functional yields a transformation $\boldsymbol{u}$ which (1) approximates the distance between the marker point set and (2) is sufficiently smooth. The relative weight between the approximation behavior and the smoothness of the transformation is determined by the regularization parameter $\lambda > 0$. If λ is small, we obtain a solution with good adaptation to the local structure of the deformations and if λ is large, we obtain a very smooth transformation with little adaption to the deformations. There are two limiting cases: for $\lambda \to 0$ we obtain the original interpolating Thin-plate spline transformation, and for $\lambda \to \infty$ we have a global polynomial of order up to $m-1$, which has no bending energy at all [19].

The computational scheme to compute the coefficients of the transformation $\boldsymbol{u}$ is:

$$\begin{aligned}(K + N\lambda W^{-1})\boldsymbol{w} + P\boldsymbol{a} &= \boldsymbol{v} \\ P^T\boldsymbol{w} &= \boldsymbol{0}\end{aligned} \tag{3.22}$$

where

$$W^{-1} = \begin{pmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_N^2 \end{pmatrix}. \tag{3.23}$$

So now if we use $\tilde{K}$ to represent $K + N\lambda W^{-1}$, then we get

$$\tilde{K} = \begin{bmatrix} \sigma_1^2 & U(\boldsymbol{p}_1,\boldsymbol{p}_2) & \cdots & U(\boldsymbol{p}_1,\boldsymbol{p}_N) \\ U(\boldsymbol{p}_2,\boldsymbol{p}_1) & \sigma_2^2 & \cdots & U(\boldsymbol{p}_2,\boldsymbol{p}_N) \\ & \cdots & & \\ U(\boldsymbol{p}_N,\boldsymbol{p}_1) & U(\boldsymbol{p}_N,\boldsymbol{p}_2) & \cdots & \sigma_N^2 \end{bmatrix}.$$

We get:

$$\begin{bmatrix} \tilde{K}_{11}\tilde{K}_{12}\cdots\tilde{K}_{1N} \\ \tilde{K}_{21}\tilde{K}_{22}\cdots\tilde{K}_{2N} \\ \cdots \\ \tilde{K}_{N1}\tilde{K}_{N2}\cdots\tilde{K}_{NN} \end{bmatrix} \times \begin{bmatrix} w_{1x}w_{1y}w_{1z} \\ w_{2x}w_{2y}w_{2z} \\ \cdots \\ w_{Nx}w_{Ny}w_{Nz} \end{bmatrix} + \begin{bmatrix} P_{11}P_{12}\cdots P_{1M} \\ P_{21}P_{22}\cdots P_{2M} \\ \cdots \\ P_{N1}P_{N2}\cdots P_{NM} \end{bmatrix} \times \begin{bmatrix} a_{1x}a_{1y}a_{1z} \\ a_{2x}a_{2y}a_{2z} \\ \cdots \\ a_{Mx}a_{My}a_{Mz} \end{bmatrix} = \begin{bmatrix} q_{1x}q_{1y}q_{1z} \\ q_{2x}q_{2y}q_{2z} \\ \cdots \\ q_{Nx}q_{Ny}q_{Nz} \end{bmatrix}$$

$$\begin{bmatrix} P_{11}P_{12}\cdots P_{1M} \\ P_{21}P_{22}\cdots P_{2M} \\ \cdots \\ P_{N1}P_{N2}\cdots P_{NM} \end{bmatrix}^T \begin{bmatrix} w_{1x}w_{1y}w_{1z} \\ w_{2x}w_{2y}w_{2z} \\ \cdots \\ w_{Nx}w_{Ny}w_{Nz} \end{bmatrix} = \mathbf{0}. \tag{3.24}$$

Let

$$P = (Q_1 : Q_2)\begin{pmatrix} R \\ 0 \end{pmatrix}$$

be the QR decomposition of P. The solution finally is [23]

$$\begin{aligned} \boldsymbol{w} &= Q_2(Q_2^T\tilde{K}Q_2)^{-1}Q_2^T\boldsymbol{v}, \\ R\boldsymbol{a} &= Q_1^T(\boldsymbol{v} - \tilde{K}\boldsymbol{w}). \end{aligned} \tag{3.25}$$

3.3.2 Anisotropic marker points

The approximation scheme described in the previous section uses scalar weights to represent marker point localization errors [19]. This, however, implies isotropic localization errors and is only a coarse error characterization. Generally, the errors are different in different directions and thus are anisotropic. A further extension of the approach is obtained by replacing the scalar weights σ_i^2 with matrices Σ_i representing anisotropic marker point localization errors [19]. Now the functional is:

$$J_\lambda(\boldsymbol{u}) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{q}_i - \boldsymbol{u}(\boldsymbol{q}_i))^T\Sigma_i^{-1}(\boldsymbol{q}_i - \boldsymbol{u}(\boldsymbol{q}_i)) + \lambda J_m^d(\boldsymbol{u}). \tag{3.26}$$

But the computational scheme to compute the coefficients of the transformation $\boldsymbol{u}$ is the same as before:

$$\begin{aligned} (K + N\lambda W^{-1})\boldsymbol{w} + P\boldsymbol{a} &= \boldsymbol{v} \\ P^T\boldsymbol{w} &= \mathbf{0}; \end{aligned} \tag{3.27}$$

the only change here is concerning the W^{-1} matrix which becomes:

$$W^{-1} = \begin{pmatrix} \Sigma_1 & & 0 \\ & \ddots & \\ 0 & & \Sigma_N \end{pmatrix}. \tag{3.28}$$

Note[10] that the Σ_i represent the localization errors of two corresponding marker points. A typical Σ_i^{-1} in (3.26) will be a 3×3 symmetric, positive-definite matrix with real eigenvalues $\frac{1}{(\sigma^1)^2}$, $\frac{1}{(\sigma^2)^2}$ and $\frac{1}{(\sigma^3)^2}$. This corresponds to using a norm that has a unit sphere in the shape of an ellipsoid. The principal axes of this ellipsoid are $\boldsymbol{n}^1$, $\boldsymbol{n}^2$ and $\boldsymbol{n}^3$, the eigenvectors of Σ_i^{-1}. The half-length of the ellipsoid in the direction $\boldsymbol{n}^j$ is σ_j, $j = 1, 2, 3$.

A large value of σ_j means that the error in this direction is unimportant. So if it is only the error in the direction $\boldsymbol{w}$ that is important, we would choose a pancake-shaped ellipsoid with $\boldsymbol{w}$ as the normal to the surface of the pancake. Similarly, if it is the error in the directions $\boldsymbol{v}$ and $\boldsymbol{w}$ that are important, we would choose a long thin ellipsoid with major axis equal to $\boldsymbol{v} \times \boldsymbol{w}$.

The matrix Σ_i^{-1} has an orthonormal basis of eigenvectors $\boldsymbol{n}^1$, $\boldsymbol{n}^2$ and $\boldsymbol{n}^3$.

$$\Sigma_i^{-1} = [\boldsymbol{n}^1 \boldsymbol{n}^2 \boldsymbol{n}^3] \cdot \begin{bmatrix} \frac{1}{(\sigma^1)^2} & 0 & 0 \\ 0 & \frac{1}{(\sigma^2)^2} & 0 \\ 0 & 0 & \frac{1}{(\sigma^3)^2} \end{bmatrix} \cdot [\boldsymbol{n}^1 \boldsymbol{n}^2 \boldsymbol{n}^3]^T$$

where $\| \boldsymbol{n}^j \| = 1$, $j = 1, 2, 3$.

Since after introduction of the matrix Σ_i, the value of errors on each direction changed according to different values in the Σ matrix, we must make a minor modification on the overall data structure. That is, the dimension of $\boldsymbol{w}$ is $dN \times 1$, $\boldsymbol{a}$ is $dM \times 1$, and $\boldsymbol{q}$ is $dN \times 1$. Now the modified version is:

$$\begin{bmatrix} \tilde{K} \end{bmatrix} \cdot \begin{bmatrix} w_{1x} \\ w_{1y} \\ w_{1z} \\ \vdots \\ w_{Nx} \\ w_{Ny} \\ w_{Nz} \end{bmatrix} + \begin{bmatrix} P \end{bmatrix} \cdot \begin{bmatrix} a_{1x} \\ a_{1y} \\ a_{1z} \\ \vdots \\ a_{Mx} \\ a_{My} \\ a_{Mz} \end{bmatrix} = \begin{bmatrix} q_{1x} \\ q_{1y} \\ q_{1z} \\ \vdots \\ q_{Nx} \\ q_{Ny} \\ q_{Nz} \end{bmatrix}$$

[10] The details on ellipsoidal norms given here are new.

and

$$\left[\ P\ \right]^T \cdot \begin{bmatrix} w_{1x} \\ w_{1y} \\ w_{1z} \\ \vdots \\ w_{Nx} \\ w_{Ny} \\ w_{Nz} \end{bmatrix} = \mathbf{0}, \tag{3.29}$$

where $\tilde{K} = (K + N\lambda W^{-1})$, i.e,

$$\tilde{K} = \begin{bmatrix}
N\lambda\Sigma_{11}^1 & N\lambda\Sigma_{12}^1 & N\lambda\Sigma_{13}^1 & K_{12} & 0 & 0 & \cdots & K_{1N} & 0 & 0 \\
N\lambda\Sigma_{21}^1 & N\lambda\Sigma_{22}^1 & N\lambda\Sigma_{23}^1 & 0 & K_{12} & 0 & & 0 & K_{1N} & 0 \\
N\lambda\Sigma_{31}^1 & N\lambda\Sigma_{32}^1 & N\lambda\Sigma_{33}^1 & 0 & 0 & K_{12} & & 0 & 0 & K_{1N} \\
K_{21} & 0 & 0 & N\lambda\Sigma_{11}^2 & N\lambda\Sigma_{12}^2 & N\lambda\Sigma_{13}^2 & & K_{2N} & 0 & 0 \\
0 & K_{21} & 0 & N\lambda\Sigma_{21}^2 & N\lambda\Sigma_{22}^2 & N\lambda\Sigma_{23}^2 & \cdots & 0 & K_{2N} & 0 \\
0 & 0 & K_{21} & N\lambda\Sigma_{31}^2 & N\lambda\Sigma_{32}^2 & N\lambda\Sigma_{33}^2 & & 0 & 0 & K_{2N} \\
\vdots & & & \vdots & & & & \vdots & & \\
K_{N1} & 0 & 0 & K_{N2} & 0 & 0 & & N\lambda\Sigma_{11}^N & N\lambda\Sigma_{12}^N & N\lambda\Sigma_{13}^N \\
0 & K_{N1} & 0 & 0 & K_{N2} & 0 & \cdots & N\lambda\Sigma_{21}^N & N\lambda\Sigma_{22}^N & N\lambda\Sigma_{23}^N \\
0 & 0 & K_{N1} & 0 & 0 & K_{N2} & & N\lambda\Sigma_{31}^N & N\lambda\Sigma_{32}^N & N\lambda\Sigma_{33}^N
\end{bmatrix}$$

and

$$P = \begin{bmatrix}
\phi_1(\boldsymbol{p}_1) & 0 & 0 & \phi_2(\boldsymbol{p}_1) & 0 & 0 & \cdots & \phi_M(\boldsymbol{p}_1) & 0 & 0 \\
0 & \phi_1(\boldsymbol{p}_1) & 0 & 0 & \phi_2(\boldsymbol{p}_1) & 0 & & 0 & \phi_M(\boldsymbol{p}_1) & 0 \\
0 & 0 & \phi_1(\boldsymbol{p}_1) & 0 & 0 & \phi_2(\boldsymbol{p}_1) & & 0 & 0 & \phi_M(\boldsymbol{p}_1) \\
\phi_1(\boldsymbol{p}_2) & 0 & 0 & \phi_2(\boldsymbol{p}_2) & 0 & 0 & & \phi_M(\boldsymbol{p}_2) & 0 & 0 \\
0 & \phi_1(\boldsymbol{p}_2) & 0 & 0 & \phi_2(\boldsymbol{p}_2) & 0 & \cdots & 0 & \phi_M(\boldsymbol{p}_2) & 0 \\
0 & 0 & \phi_1(\boldsymbol{p}_2) & 0 & 0 & \phi_2(\boldsymbol{p}_2) & & 0 & 0 & \phi_M(\boldsymbol{p}_2) \\
\cdots & & & \cdots & & & & \cdots & & \\
\phi_1(\boldsymbol{p}_n) & 0 & 0 & \phi_2(\boldsymbol{p}_n) & 0 & 0 & & \phi_M(\boldsymbol{p}_n) & 0 & 0 \\
0 & \phi_1(\boldsymbol{p}_n) & 0 & 0 & \phi_2(\boldsymbol{p}_n) & 0 & \cdots & 0 & \phi_M(\boldsymbol{p}_n) & 0 \\
0 & 0 & \phi_1(\boldsymbol{p}_n) & 0 & 0 & \phi_2(\boldsymbol{p}_n) & & 0 & 0 & \phi_M(\boldsymbol{p}_n)
\end{bmatrix}.$$

Here, the Σ_{mn}^i are the components of Σ_i, $1 \le m, n \le 3$. Let

$$P = (Q_1 : Q_2)\begin{pmatrix} R \\ 0 \end{pmatrix}$$

be the QR decomposition of P. The solution finally is [19]

$$\begin{aligned} \boldsymbol{w} &= Q_2(Q_2^T\tilde{K}Q_2)^{-1}Q_2^T\boldsymbol{v}, \\ R\boldsymbol{a} &= Q_1^T(\boldsymbol{v} - \tilde{K}\boldsymbol{w}). \end{aligned} \tag{3.30}$$

In the implementation, it was still possible for us to keep the same data structure for this anisotropic marker points case and this will be a memory-saving

technique. We can store the error direction information in an independent matrix called ΔW^{-1}

$$\Delta W^{-1} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \Sigma_{13} \\ \Sigma_{21} & \Sigma_{22} & \Sigma_{23} \\ \Sigma_{31} & \Sigma_{32} & \Sigma_{33} \end{bmatrix}$$

then what we need to do is some more additions based on what we get for the two previous models (interpolation and approximation) as:

$$\begin{aligned} x_{anisotropic} &= (\Sigma_{11} \times x + \Sigma_{12} \times y + \Sigma_{13} \times z) + x_{old}; \\ y_{anisotropic} &= (\Sigma_{21} \times x + \Sigma_{22} \times y + \Sigma_{23} \times z) + y_{old}; \\ z_{anisotropic} &= (\Sigma_{31} \times x + \Sigma_{32} \times y + \Sigma_{33} \times z) + z_{old}. \end{aligned}$$

where x_{old}, y_{old} and z_{old} represent the coordinates before change, that is used for both the interpolation and approximation case, and $x_{anisotropic}$, $y_{anisotropic}$ and $z_{anisotropic}$ represent the transformed anisotropic values of the coordinates.

3.4 Comparison between [0,1] index model and R^d index model

One of the goals of this thesis was to show the link between the Thin-plate spline models based on [0,1] index and R^d index[11]. Even though they are two completely different spline models, after the permutation of variables done in Section 3.2.2 and the mathematical development in Section 3.2 and 3.3, we can see that they really share something in common, which is quite useful in our application.

3.4.1 Similarities

Both the [0,1] index model and the R^d model can act as prediction functions. The objective function of the [0,1] index model can be written as

$$\sum_{i=1}^{n} (\boldsymbol{y}_i - \boldsymbol{f}(t_i))^T \Sigma_i^{-1} (\boldsymbol{y}_i - \boldsymbol{f}(t_i)) + \sum_{k=1}^{d} \lambda \int_0^1 (f_k''(t))^2 dt, \tag{3.31}$$

[11]Showing the similarity between the two models is new.

where

$$\boldsymbol{y}_i - \boldsymbol{f}(t_i) = [y_{1i} - f_1(t_i), y_{2i} - f_2(t_i), y_{3i} - f_3(t_i)]^T.$$

Then the spline model based on [0,1] index model may now be easily compared with the R^d model which is, for $k = 1, ..., d$

$$u_k(x) = \sum_{\nu=1}^{M} a_{\nu k}\phi_\nu(x) + \sum_{i=1}^{N} w_{ik}U(x, p_i), \tag{3.32}$$

by minimizing the functional

$$\sum_{i=1}^{N}(\boldsymbol{q}_i - \boldsymbol{u}(p_i))^T\Sigma_i^{-1}(\boldsymbol{q}_i - \boldsymbol{u}(p_i)) + \sum_{k=1}^{d}\lambda \cdot J_m^d(u_k), \tag{3.33}$$

where

$$J_m^d(u) = \sum_{\alpha_1+...+\alpha_d=m} \frac{m!}{\alpha_1!...\alpha_d!} \times \int_{R^d} (\frac{\partial_m u}{\partial x_1^{\alpha_1}...\partial x_d^{\alpha_d}})^2 dx.$$

Comparing (3.31) and (3.33), the methods have very similar structure.

It is easy to verify that, after permutation of variables, the $(nd \times md)$ matrix T for the [0,1] index model has exactly the same form as each $(n \times m)$ matrix T_k, with each element $\phi_\nu(t_i)$ replaced by $\phi_\nu(t_i) \cdot I_{(d\times d)}$, where I is the identity matrix. Similarly, the $(nd \times nd)$ matrix S has exactly the same form as each $(n \times n)$ matrix S_k, with each element $R^1(t_i, t_j)$ replaced by $R^1(t_i, t_j) \cdot I_{(d\times d)}$. Analogous statements apply to the matrices involved in the equations to be solved the R^d model [19], and they may be solved using the QR approach.

3.4.2 Differences

Since the [0,1] index model and the R^d index model are two different models, some differences are obvious: for example, the different definition of the coefficient vectors for (3.31) and (3.33). But here, what interests us most is the relation between different body points in the model. For the [0,1] index model, the deformation of each point is only related to its own position at different time points: there is no interactive relation between points. For the R^d index model, the deformation is defined by the relation between body points and the marker points. The deformed position of a certain body point is decided by the positions of all the

marker points and the weighting factor of the distance of this body point from all the marker points. This explains why the coefficient vector generalization test (in Section 6.3) can work with the R^d index model, while for the [0,1] index model, it does not.

Chapter 4

Models for simulation of deformation

In the previous chapters, we discussed three different models for deformation; now, based on these models, we extend a little to a model called "Affine-affine matching", which will be described in Section 4.1. Then, in Section 4.2, we will give some general discussion concerning all the prediction models[1]. The method of affine-affine matching was not, however, implemented in our prototype interface.

4.1 Affine-affine matching

In Chapter 3, we discussed the Thin-plate models based on both the [0,1] index and the R^d index, to do time-interpolation or time-extrapolation, and deformation, respectively. But in both of the two cases, we are working with individual 3-dimensional points, which is quite limited in the sense of deformation: we do not include, for example, any information about the plane in which the deformed point should be. Plus, consider the size of the difference between a given point $\boldsymbol{y}_i$ on the boundary surface and its predicted values $\boldsymbol{f}(t_i)$ (Section 3.2, [0,1] index), or the size of the difference between the desired position $\boldsymbol{q}_i$ of a point on the boundary surface and the transformed position $\boldsymbol{u}(\boldsymbol{p}_i)$ of the initial position $\boldsymbol{p}_i$ (Section 3.3,

[1]The material in this chapter is new.

R^d index). In both cases, we may wish to relax the requirement that closeness be measured using the ordinary Euclidean metric. For example, we may want a one-dimensional set (part of a line in R^3) on the object surface to be close to another given one-dimensional set, or, a two-dimensional set (part of a plane in R^3) on the object surface to be close to another given two-dimensional set.

We describe the matching done by the use of criteria which we will refer to as "affine-affine matching". An m-dimensional affine set in R^3 is specified by choosing $m+1$ affinely independent points in R^3. It is the cases $m = 0, 1$ and 2 that are of primary interest: point-point matching ($m = 0$), line-line matching ($m = 1$), and plane-plane matching ($m = 2$). For example, three affinely independent points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3 \in R^3$ define a (locally) planar section of the object surface; then, the middle point $\boldsymbol{x}_0 = \theta_1\boldsymbol{x}_1 + \theta_2\boldsymbol{x}_2 + \theta_3\boldsymbol{x}_3, (\theta_1 + \theta_2 + \theta_3 = 1, 0 \leq \theta_1, \theta_2, \theta_3)$ is to be compared with another point $\boldsymbol{y}_0$ on a surface having specified outer normal $\boldsymbol{n}$. Suppose that error in the direction $\boldsymbol{n}$ is much more significant than errors in the other two orthogonal directions. In this case, we may use a metric that has a pancake-shaped ellipsoidal unit sphere with its flat side orthogonal to $\boldsymbol{n}$, and compare the three points $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$ with three suitably chosen points in the plane $\boldsymbol{n}^T(\boldsymbol{y} - \boldsymbol{y}_0) = 0$ (see Fig. 4.1).

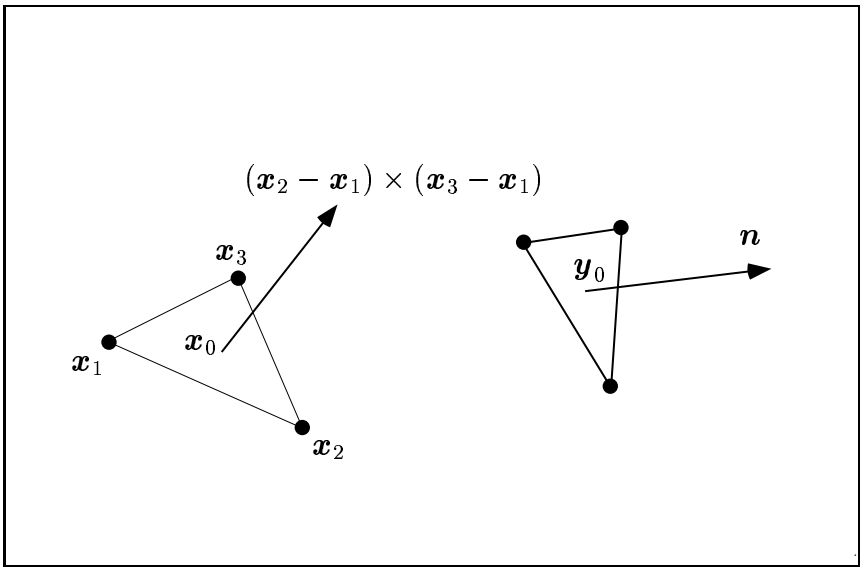


Figure 4.1: Plane-plane matching

This idea is quite similar (a slight generalization) to the idea of the anisotropic error measurement of the Thin-plate spline based on R^d model introduced by [19] and discussed above in Section 3.3.2. Specification of the error unit sphere is technically straightforward. A typical Σ_i^{-1} will be a 3×3 symmetric, positive-definite matrix with real the eigenvalues $\frac{1}{(\sigma^1)^2}$, $\frac{1}{(\sigma^2)^2}$ and $\frac{1}{(\sigma^3)^2}$. This corresponds to using a norm that has a unit sphere in the shape of an ellipsoid. The principal axes of this ellipsoid are $\boldsymbol{n}^1$, $\boldsymbol{n}^2$ and $\boldsymbol{n}^3$, eigenvectors of Σ_i^{-1}. The half-length of the ellipsoid in the direction $\boldsymbol{n}^j$ is σ_j, $j = 1, 2, 3$. Thus, for the flat ellipsoid described above, we choose $\boldsymbol{n}^1$ equal to the given $\boldsymbol{n}$ and complete the orthonormal basis with two vectors in the plane $\boldsymbol{n}^T(\boldsymbol{y} - \boldsymbol{y}_0) = 0$; then, σ^1 is chosen to be small relative to σ^2 and σ^3. The matrix Σ_i^{-1} is obtained as

$$[\boldsymbol{n}^1\boldsymbol{n}^2\boldsymbol{n}^3] \cdot \begin{bmatrix} \frac{1}{(\sigma^1)^2} & 0 & 0 \\ 0 & \frac{1}{(\sigma^2)^2} & 0 \\ 0 & 0 & \frac{1}{(\sigma^3)^2} \end{bmatrix} \cdot [\boldsymbol{n}^1\boldsymbol{n}^2\boldsymbol{n}^3]^T$$

where $\| \boldsymbol{n}^j \| = 1$, $j = 1, 2, 3$.

Similarly, in the case of line-line matching ($m = 1$), if errors in the directions $\boldsymbol{v}$ and $\boldsymbol{w}$ are important, where $\boldsymbol{v} \perp \boldsymbol{w}$, but errors along the line defined by $\boldsymbol{v} \times \boldsymbol{w}$ are not, then we choose $\boldsymbol{v}$, $\boldsymbol{w}$ and $\boldsymbol{v} \times \boldsymbol{w}$ as eigenvectors, with the eigenvalues corresponding to $\boldsymbol{v} \times \boldsymbol{w}$ chosen to be large relative to the other two.

This "affine-affine matching" model could be quite useful in practice. By defining the three error directions, it well defines the shape of the deformation, while keeping the overall figure of the object unchanged. This would be especially helpful in our special medical application, because we model our deformation based on the pictures we took for the patient, but the postures of the patient may bring errors inside the deformation, especially on the plane of the skin, which corresponds to skin slippage. By defining the special direction for the error, we can reduce this kind of error as much as we can.

4.2 Other prediction models

1. Prediction on time index set

The prediction model based on time index set [0,1] was discussed in Section 3.2. There are two different kinds of prediction in time: time-interpolation has the time point inside the time interval; time-extrapolation is the time point outside of the time interval. This model gives either an interpolation or an approximation result, depending on whether it tries to match the exact values at a certain time point. But in the case of R^d values, the model can only give us approximate values, as we discussed in Section 3.2: the case when the smoothing parameter $\lambda = 0$ does not work under this situation. And the result of the how good the approximation is still depends on the value of λ.

With both time-interpolation and time-extrapolation, we may trace all the progress of the deformation, given the status of the object at (at least) three time points, and the time index list.

2. Prediction on R^d index set

   Prediction based on R^d index set was our original motivation for this project. We want to get some prediction tool for our medical application — prediction of the deformation due to scoliosis. The theoretical illustration part of this model was given in Section 3.3. It can have two different forms: interpolation and approximation. The choice largely depends on our needs, whether we want our deformed figure to exactly fit all the given deformed marker points, or whether we want it just to approximate those given marker points, while keeping a relatively smoother shape.

3. Combination of the two models

   As we mentioned, the more related information involved, the better the deformation result will be. So here is another deformation model, a combination of the two previous models. Since with the prediction on time index set we can get some future figure of the model, and with the prediction on R^d index set, we can get the internal figure from the external ones, then what we can do is (see Fig. 4.2): first, get the external data of the object at a future time point t_{n+1} using [0,1] index model (the first row in Fig. 4.2);

choose the external marker points in the usual way; third, based on these generated external marker points, we can get the internal figure with the R^d index model (the second row in Fig. 4.2, represented by the full arrows). So with the combination of the two models, we can get the full figure (both internal and the external) of the object at any time point (the two models inside the ellipse in Fig. 4.2).

The most attractive feature of this is concerning our scoliosis application: since in our special application, external data is relatively easy to get (because there is no harmful effect on the patient), so we can get a list of the external data with quite short time intervals (just to have as many data as possible); then we can work out all the internal data with these external data with the prediction model on R^d index set. Now we have a list of the full figure of the patient, and we can use our prediction model on time index set to get the full figure at any future time point.

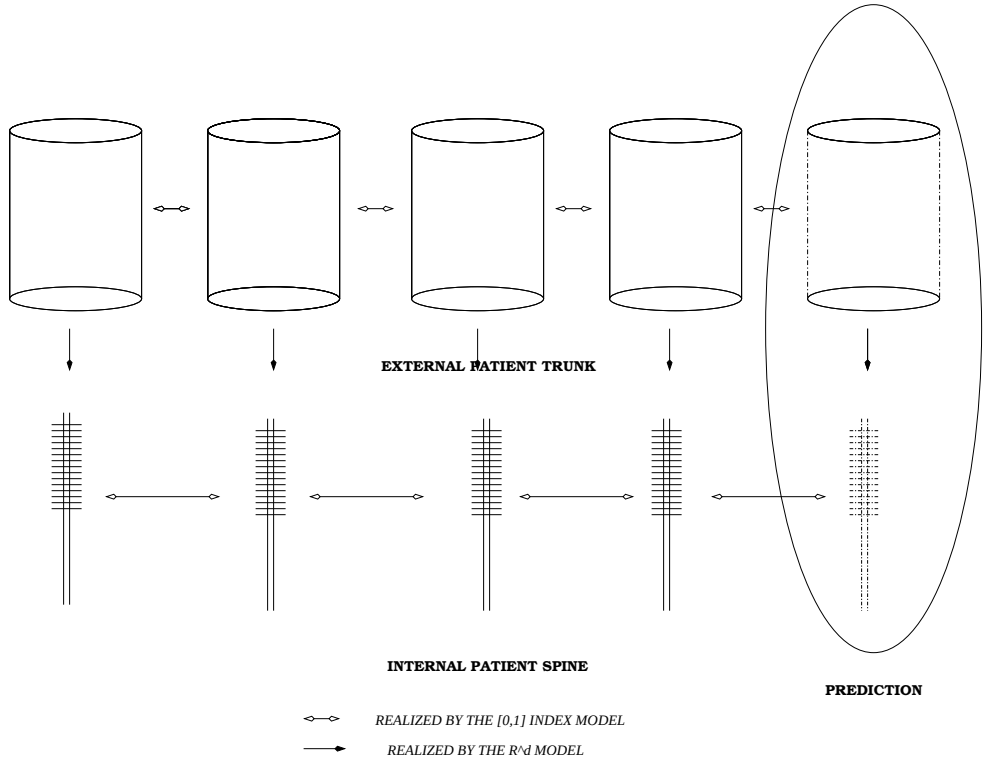


Figure 4.2: Illustration of the combination of the two models; the first row represents the external patient trunk, and the second row represents the internal patient spines. Models linked with the open arrows can be realized by the [0,1] index model (either by time-interpolation or time-extrapolation), and those linked with full arrows are realized by the R^d index model (either by interpolation or approximation). The two models inside the ellipse are the predicted results by the models. Other combinations of methods are possible if partial data (e.g. internal X-ray data at infrequent time points) is available.

Chapter 5

System interface

Part of the purpose of the project is to obtain a software implementation of spline interpolation and extrapolation model on $[0, 1]$ index [24] as well as the Thin-plate-spline prediction model on R^d index [19]. The latter method could be enhanced by a marker-point-specification method which we call "affine-affine" matching[1].

The goal of the software is to see whether the above two methods can be used to provide a convenient visualization tool in the context of deformation due to idiopathic scoliosis. The same methods could theoretically be used for *prediction* of internal deformation, given data on external deformation, but so far we are still constrained by the problem of inaccurate data. We have, however, obtained some conditional good initial results, which will be presented in Chapter 6. In this chapter we describe the system interface.

5.1 The data

We have three test cases namely: Test case A, B, and C, corresponding to the three patients available. Each Test case comprises n Internal/External models ("IE models"), where "Internal" refers to the spine and rib-cage and "External"

[1]These implementations were done from scratch, but they are not new in the sense that previous implementations exist [19, 24]. Affine-affine matching is a slight (but useful) generalization of the anisotropic error criteria of [19]. As already mentioned, this idea was not actually implemented.

refers to the external trunk. The n models represent views of the same patient at different time points t_i, $i = 1, \ldots, n$. Each such model is made up of approximately 14,000 data points in R^3 including both the internal and external data. Each patient has $n + 1$ independent files with all the available data.

For the R^d index model, besides the coordinates of all the internal and external points, there is also other information:

d the dimension of the object; for the scoliosis case, $d = 3$, but our model will also work for the case of a 2D plane, for which $d = 2$;

m the order of largest derivative in the R^d case; we tried both the case of $m = 2$ as in [19] and the case of $m = 3$;

N the number of external marker points;

totalpoints the total number of external and internal points of the current model.

For the [0,1] index model, each patient has only one file: it contains the coordinates of all the points, including both the external and the internal, at all time points. To be more consistent with our application in scoliosis, we ordered the time-point data based on time. For example, for the first test case, patient2116957, the order of data inside the file is: Nov98, May99, Nov99 and May00.

5.2 The operations

Depending on the model, the main operations can be separated as:

For the [0,1] index model:

1. *Time-interpolation*

   This operation corresponds to the time-interpolation of the [0,1] index model described in Section 3.2. Given a list of time points within the interval [0,1], and the coordinates of all the points in R^3 of each model at every time point, we can get the full point-by-point interpolation over the interval [0,1]. The input value of t is the target time point, which can be specified easily by the user.

2. *Time-extrapolation*

   This operation is quite similar to the operation in item 1, since they share the same [0,1] index model. But here the difference is that instead of input of a value of t between 0 and 1, we need a t bigger than 1 to represent the time point for which we want to predict.

For R^d index model:

1. *Alignment*

   The alignment[2] mentioned here is only a simple method. It is a prerequisite operation for the deformation operation (item 2). It permits the user to apply a rigid motion (rotation plus translation) to any one of the n IE models, viewed on top of one of the other models in a two-color overlay, and transformed in the same coordinate system.

2. *Deformation*

   This is the core operation of the Thin-plate spline model based on the R^d index. Given two groups of marker points and a complete figure in terms of coordinates of points, it permits us to get the deformed figure. To make sure the final result is comparable with the given data, we require item 1 to be its prerequisite operation.

5.3 The interface

The form of the interface is shown in Fig. 5.1.

There are three windows, each with associated buttons and/or sliders: *TopLeft* (Fig 5.2(a)), *TopRight* (Fig 5.2(b)), and *Bottom* (Fig 5.3).

1. *TopLeft* acts as an initialization window. It sets the initial figure for the deformation, which will be displayed in the *TopRight* window, and it permits

[2]The general problem of calibration in this context is likely to be very difficult: the patient may have grown, gained weight, and/or suffered further scoliotic deformation; marker points may have been placed at slightly different positions, cameras may have changed or moved, and so on. These problems are beyond the scope of this thesis.

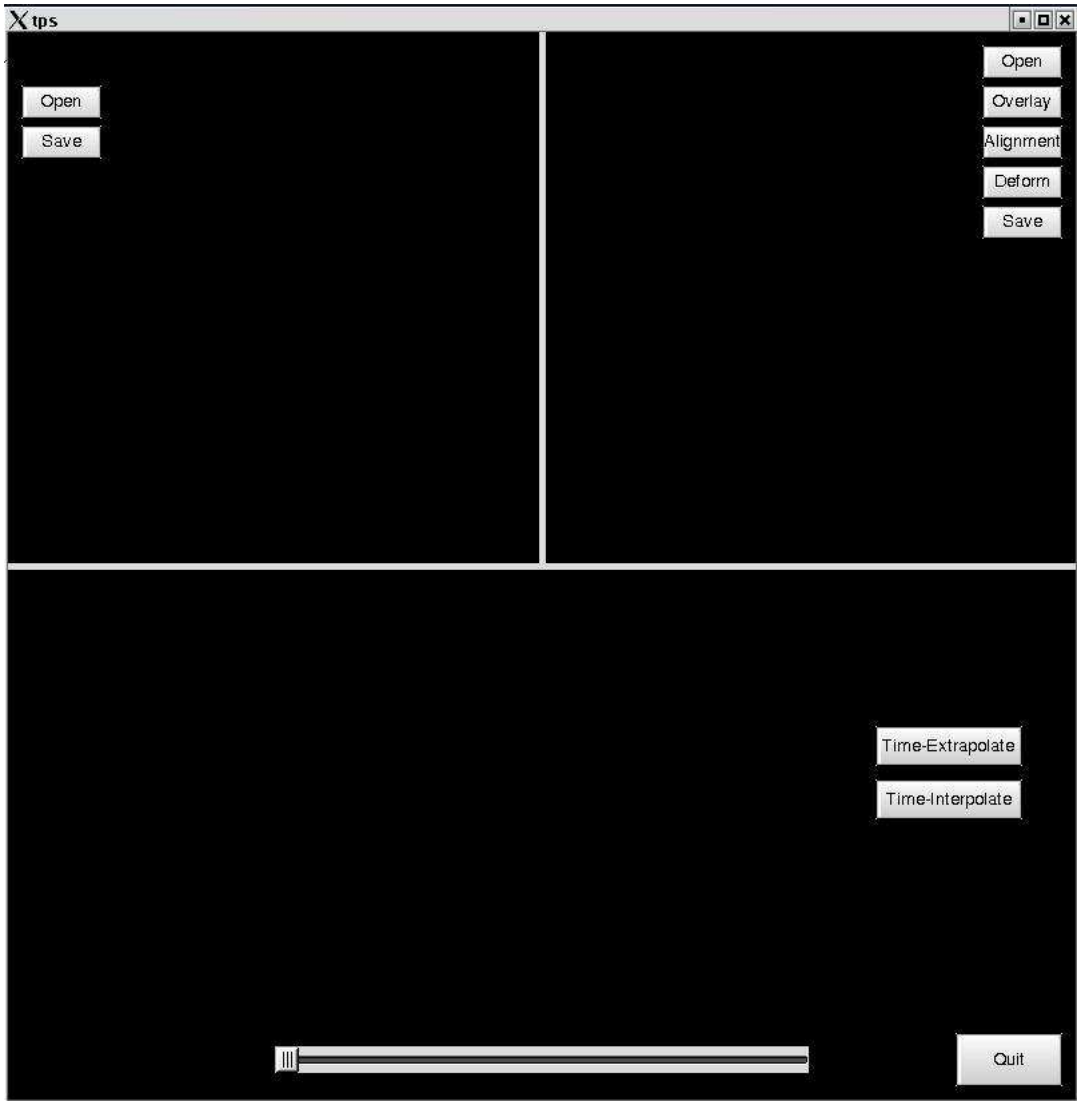


Figure 5.1: Interface which contains 3 sub-windows, each with associated buttons and/or sliders.

specification of the patientID that will be the target of the time-interpolation and time-extrapolation. The last will be displayed in the *Bottom* window. It has 2 buttons as follows:

Open Display of a file of form *idtimepoint.dat* in the *TopLeft* window, by means of a pop-up menu giving access to such files;

Save Save the data displayed in the *TopLeft* window after modification.

2. *TopRight* is the window for the display of the deformation. It has the same basic buttons as the *TopLeft* window, plus the functional buttons:

   Overlay Overlay the figure in the *TopLeft* to the *TopRight* window;

   Alignment Transform the overlayed figure into the same coordinate system as the final figure which is opened in this window;

   Deform Deform the registered figure in the window.

3. *Bottom* is the window for time-interpolation and time-extrapolation. It has a slider across the bottom to permit display of interpolated values. It also has the buttons:

   Time-interpolation Get the interpolation result for the patient that has been chosen in the *TopLeft* window: the default value of t is set to be 0.5;

   Time-extrapolation Get the extrapolation result for the patient that has been chosen in the *TopRight* window: the default value of t is set to be 1.3.

The four operations described above are available with this interface, in the following way:

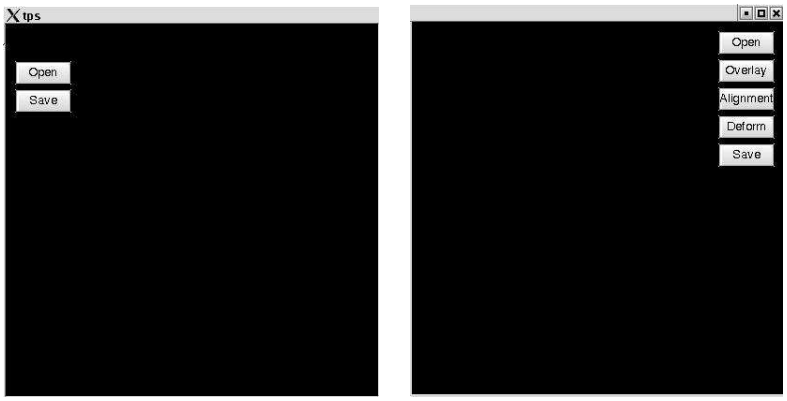
1. *Registration* is implemented by the *Alignment* button in the *TopRight* window. First the user may open a file, using *Open*, to define the initial figure and display it in the *TopLeft* window. When the user opens a different file in the *TopRight* window, he may also use the *Overlay* button in the

TopRight window to automatically load the figure from the *TopLeft* into the *TopRight* window. Thus, two different figures will be shown in the same window (*TopRight*) in different colors. By clicking the *Alignment* button, the two figures in the *TopRight* will be automatically aligned.

2. *Deformation* is implemented by the *Deform* button in the *TopRight* window. Once the user has chosen the original marker points, the deformed marker points, and the start figure that he wants to deform, then clicking the button *Deform* will cause the Thin-plate spline model based on R^d index model to be invoked, and the deformed figure will show up the *TopRight* window. At this time, if the user does not click *Time-interpolate* then the *Bottom* window remains black.

3. *Time-interpolation* is done by the *Time-interpolate* button in the *Bottom* window. Before clicking this button, the user must choose an initial figure, which is shown in the *TopLeft* window, and a target figure, which is shown in the *TopRight* window. Then, once the *Time-interpolate* button is clicked, the user can view the time-interpolated figures by means of the slider at the bottom of the *Bottom* window. (Once the initial figure is chosen by the user, all the information concerning this particular interpolated case is fixed including the patientID, the number of time points, etc..)

4. *Time-extrapolation* is implemented by the *Time-extrapolate* button in the *Bottom* window. This is quite similar to *Time-interpolation*. The difference here is that this time, the user needs to select only one state of the patient as the end state (corresponds to the time point t_n) of a list of states, based on which the time-extrapolation will work.



(a) (b)

Figure 5.2: Interface— (a) is the *TopLeft* window which contains the *Open* and *Save* buttons, and (b) is the *TopRight* window which contains *Open*, *Overlay*, *Alignment*, *Deform* and *Save* buttons.

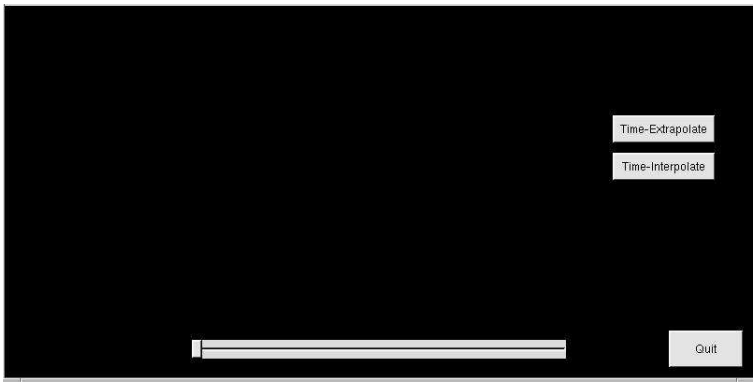


Figure 5.3: Interface — the *Bottom* window which contains *Time-Interpolation*, *Time-Extrapolation* buttons, and a slider.

Chapter 6

Experimental Results

In this chapter we will show the experimental results we got with the models described in previous chapters. The experimental results consist of two parts. The first one is a preliminary test, namely, the result we got with some simple simulated test cases for our models. We have two main purposes for this: to test how well the model works in an ideal situation and to obtain a reference for comparison with the quality of results for the real data. The second part involves the real data. The real data comes from Ste-Justine Hospital. The chapter is organized as follows: we will devote one special section (Section 6.1) to a description of the data-related topics. Then we will give the results for the [0,1] index model in Section 6.2 and those for the R^d index model in Section 6.3.

6.1 Test case analysis

6.1.1 Preliminary test case

For the preliminary test cases, usually we choose simple models simulating our real-data cases.

For the [0,1] index model, we chose four co-centric cylinders with radius 1, 2, 3 and 4 respectively to simulate the change in the figure of human body as time passes (see Fig. 6.1).

We choose a certain value of t which is between 0 and 1 to simulate the time-

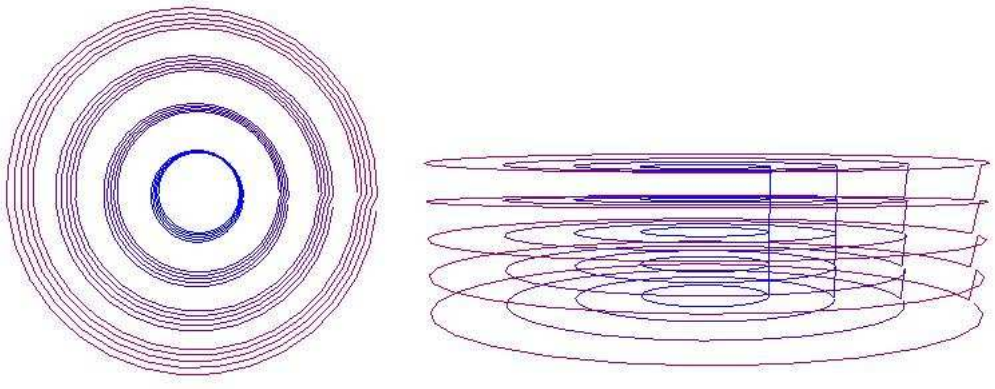
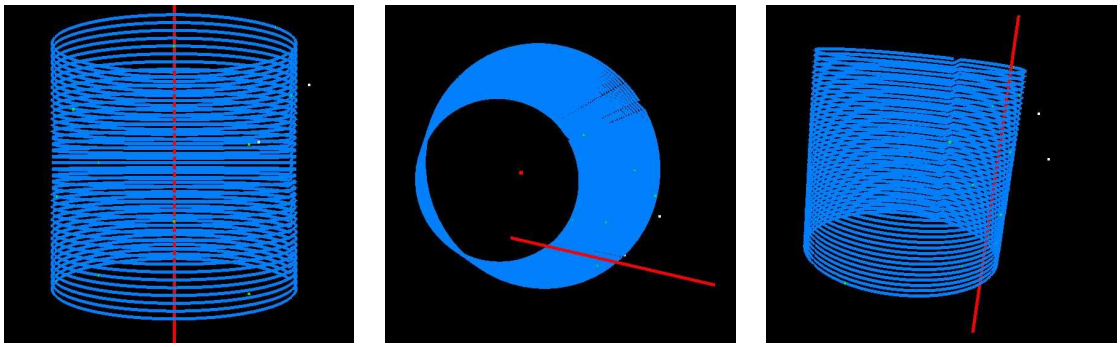


Figure 6.1: Preliminary test model for [0,1] index model — four co-centric cylinders representing the four models at different time points with radius 1, 2, 3 and 4 respectively; (a) is the topview and (b) is the sideview.

interpolation case, and a t which is greater than 1 to simulate time-extrapolation. The advantage of this simulated test model is that we can easily figure out where a certain time-interpolated or time-extrapolated cylinder layer should be by comparing the values with all the given time points.

For the R^d model ($d = 3$), we use a cylinder to simulate the human trunk, and one internal vertical line which is located quite close to the side of the cylinder to simulate the internal spine (see Fig. 6.2). Since the ultimate goal of the R^d



(a) (b) (c)

Figure 6.2: Test model for the 3-dimensional object with $m = 2$ (where m was defined on p. 29) $N = 12$ for R^d index model; (a) is the straight sideview, (b) is the topview and (c) is the sideview at a slight angle.

model is to get the internal spine deformation information provided the external

body deformation, we can deform a certain part of the cylinder as scoliosis does and see what will happen with the vertical internal line which is supposed to be the spine.

6.1.2 Real-data test case

In contrast to the preliminary test case, which we defined by ourselves, we do not have so many choices in the case of the real data, since all the data comes from Ste-Justine Hospital, Montreal. Later we will list all the contraintes and problems of these real data.

1. **Information contained in the test data**

   Totally we got 48 test cases (the number of patients) and several groups of data at different time points for each patient. Fortunately we got very detailed medical information concerning the physical condition of each patient, specified by 126 parameters. This helped us in the selection of test cases. The parameters include patient height, growth velocity, ever braced or not, primary curve type, Cobb angle for major curve, etc... Among all these parameters, 15 of them are quite related to our project including the coordinates of all the points that form the patient trunk.

2. **Criteria for test-case selection**

   Not all the 48 test cases are suitable for our application. After consultation with Dr. Hubert Labelle who is an expert in this field, we got some criteria for our test-case selections:

   (a) *mcobbttl* – Cobb angle for major curve in T/TL zone [3] from clinical chart (degrees, + to right).

   Its absolute value should be greater or equal to 40, and the change in values between different time points should be greater than 10. This is to make sure that the change in form of the patient figure is big enough for our model to simulate.

(b) *mcobbl* – Cobb angle for major curve in L zone [3] from clinical chart (degrees, + to right).

Its absolute value should be greater or equal to 40, and the change in values between different time points should be greater than 10. This is to make sure that the change in form of the patient figure is big enough for our model to simulate.

(c) *mcurvetype* – Primary curve type, 1-8 (LuT, RuT, LT, RT, LTL, RTL, LL, RL) [3], 0 if N/A, from chart.

Here the types of interest to us are either RT or RTL;

(d) *mclincobb* – Cobb angle of primary curve, from chart (degrees, + to right).

Its absolute Cobb angle should be greater or equal to 40, and the change in values between different time points should be greater than 10. This is to make sure that the change in form of the patient figure is big enough for our model to simulate.

(e) t_i – Time points available.

Both for the [0,1] index model and the R^d index model, we hope to have as many time points as possible, especially for the [0,1] index model. Since the number of time points n must be greater than the index m of the highest derivative, for the possible test case, we must have at least 3 times points available, because we still want to have one in-between for comparison purposes.

Based on all these conditions, we finally got 3 test cases from all the 48 which have the same type of scoliosis, and a similar degree of change in the figure. These offer us the opportunity to do the test of "coefficient vector generalization" of our model[1]. For the first test case, we got data for 4 time points, but only 3 for the other 2 test cases. The information is listed in the following tables:

[1]This will be discussed later in Section 6.3.2.

Table 6.1: Physical information available for each patient

PatientID	Sex	Age	Surgery	mcobbttl t1 t2 t3 t4	mcobb1 t1 t2 t3 t4	mcurvetype	mclincobb t1 t2 t3 t4
patient1	F	16.3	F	43 56 56 62	-34 -34 -38 -45	4	43 56 56 62
patient2	F	13.5	F	38 44 47 –	38 44 47 –	4	38 44 47 –
patient3	F	13.0	F	32 40 48 –	-26 0 0 –	4	32 40 48 –

Table 6.2: Modeling information available for each patient

PatientID	NumberOfPoints t1 t2 t3 t4	NumberOfContours t1 t2 t3 t4	NumberOfPoints /Contour
patient1	14040 12600 13680 14040	39 35 38 39	360
patient2	14400 14400 14760 —–	40 40 41 –	360
patient3	14040 14400 14040 —–	39 40 39 –	360

Here, F for the Surgery means till the time the scans were taken, the patient did not undergo any surgery; and the Points represent points in R^3.

3. **Testing data treatment**

   (a) Problems with the test case

   As we can see from the tables above, there are several problems concerning the test cases. The first and quite obvious one is that the number of points in R^3 that form the same patient at different time points is not the same, which means that there are some points that do not have their corresponding points available at some other time point. This may cause the method to fail. Second, since the data are produced at different time points, it is impossible for the patient to keep exactly the same posture as the previous time and the camera to be exactly in the same place with the same orientation. So this means it is unlikely that we will have the same coordinate system for the same patient at different time points. Third, as we mentioned at the very beginning, we got our external patient trunk data and the internal spine and rib-cage information separately from Ste-Justine Hospital. Thus, we have the problem of the alignment of the external and internal data, i.e, bringing all the data into the same coordinate system. Fourth, for the R^d model, we need to have certain number of marker points around the patient's

body, which is part of the external model for both the start figure and the final deformed figure. The problems concerning the marker points are two: finding the correspondence marker points of the original figure and the final figure and the number of marker points. In the original data from Calgary, we have a list of marker points, but their number is far from enough. As we will discuss in later sections, the number of marker points is quite important for the final result of the model: the more and the better distributed the marker points, the better the final result.

(b) Cleaning process

The cleaning process is the process that we implemented to solve our first two problems described above.

For the external data, the process contains two steps: first, by doing a rigid transformation, we brought the central contours of all the models at different time points for the same patient to the same place. To do this, we used our t_0 time point as reference, transforming all the later models into its coordinate system. Then, the second step was to cut off several contours of certain models to make all the models have exactly the same number of points (actually, because the numbers of points of each contour is the same, we just need to ensure that all the models have the same number of contours around the patient trunk).

For the internal data, since the data around the rib cage are not sorted, it is impossible to process it based on the contour parameter which does not exist, so, what we could do was to cut the points from both sides of the rib cage.

(c) Internal and external data registration

This process is for the third problem described above. For the internal and external data registration problem, it is a little bit complicated. We wrote a small program to do the internal and external data registration process. Here the main idea was to use the positions of certain marker

points as reference, transforming the external to the same coordinate system as the internal one. For the R^d model, this process is not really necessary, because for the R^d model, all the points (both the internal and the external) share the same coefficient vectors $\boldsymbol{a}$ and $\boldsymbol{w}$. Instead of doing the registration, we still can get the coefficient matrices based on the external marker points, and apply them to the internal data. The disadvantage of doing this is that when we do the display, the internal and external are not in the same system which is inconvenient for viewing. But meanwhile, we increase the accuracy of the data we get, since we do not need to do any kind of transformation calculation on the coordinates of the points.

(d) Marker-point generation

To solve the fourth problem, we wrote another small program to generate marker points at certain intervals around the patient trunk. This helps us solve the problem of the lack of marker points. But the problem of lack of the exact corresponding relationship between marker points remains, even though we tried to line up the two models as much as we could before we generated the marker points.

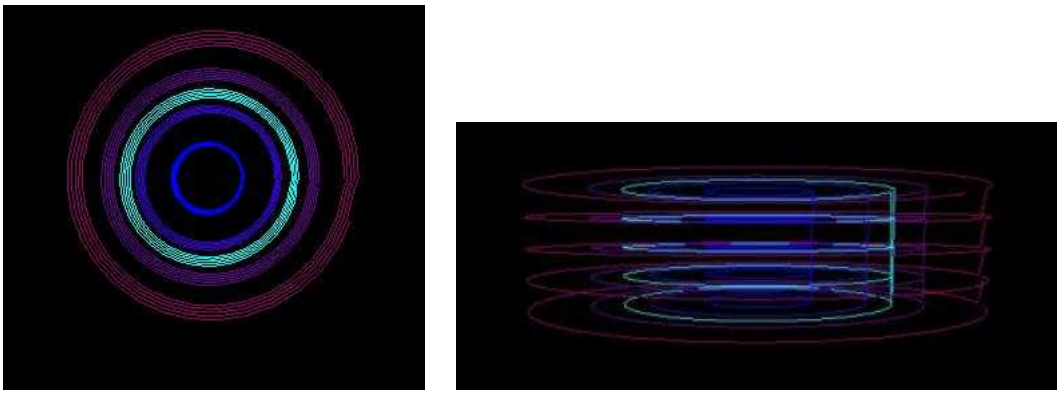
6.2 The case $\mathcal{T} = [0, 1]$

6.2.1 Preliminary experimental results

Now we are going to present the preliminary experimental results for the Thin-plate spline models based on the [0,1] index. This includes both the time-interpolation and time-extrapolation model results. Here we used four co-centric cylinders with radius 1, 2, 3 and 4 respectively as models for different time points (this test case is illustrated with no deformation in Fig. 6.1).

Test results

The results for the Thin-plate spline model on [0,1] index are quite satisfying both for the time-interpolation and the time-extrapolation (see Fig. 6.3, 6.4, 6.5 and the related discussion on validation).



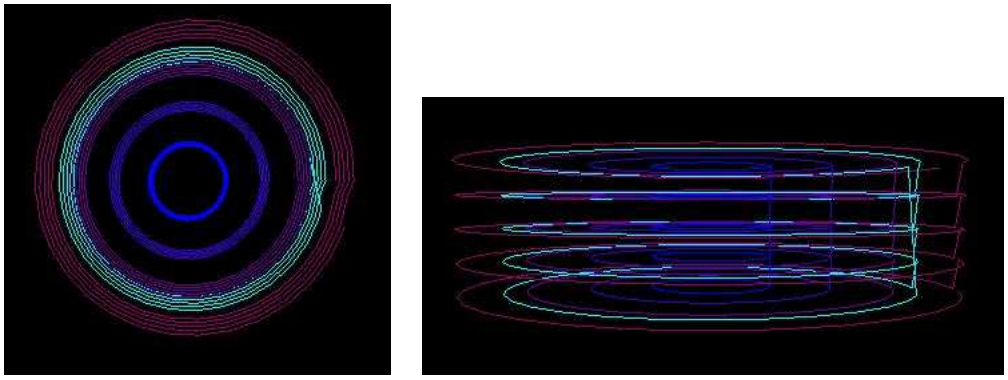
(a) (b)

Figure 6.3: Preliminary test results for [0,1] index model — Time-interpolation, four co-centric cylinders representing the given data, with a turquoise one representing the interpolated model at $t = 0.5$ and $\lambda = 1$; (a) is the topview and (b) is the sideview.

Validation

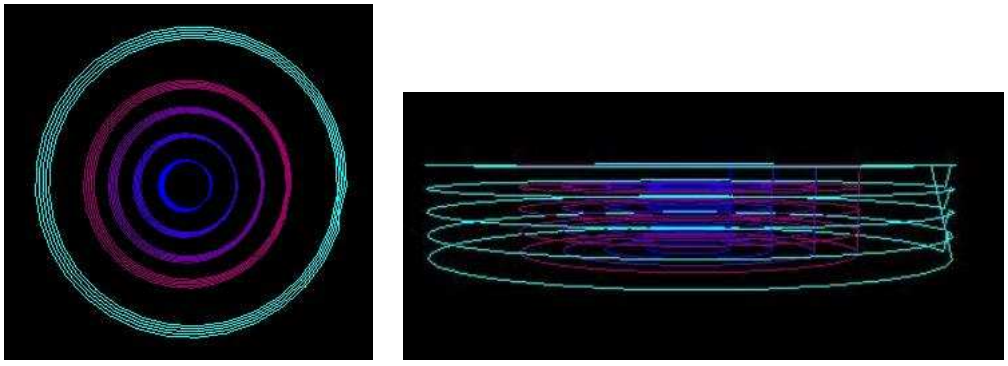
1. Result

   For the validation of our [0,1] index model, what we did with the preliminary test case was this: we ignore the given data at a certain time point t_i between t_0 and t_n which will be the interpolated time point, use our model to get that data, and compare with the available data which has been ignored. For the time-extrapolation test, we ignore the data at the last time point, and try to extrapolate from all the other given data. The parameters we use for the validation are: *totaldistance* is the sum of the errors (the error is the distance between two corresponding points), *totalpoints* is the total number of points including both the external and the internal data and *averagedistance RMS* is the average error (see Fig. 6.6 for time-interpolation and Fig. 6.7 for



(a) (b)

Figure 6.4: Preliminary test results for [0,1] index model — Time-interpolation, four co-centric cylinders representing the given data, with a turquoise one representing the interpolated model at $t = 0.8$ and $\lambda = 1$; (a) is the topview and (b) is the sideview.



(a) (b)

Figure 6.5: Preliminary test results for [0,1] index model — Time-extrapolation, four co-centric cylinders representing the given data, with a turquoise one representing the extrapolated model at $t = 1.3$ and $\lambda = 1$; (a) is the topview and (b) is the sideview.

time-extrapolation). We define

$$RMS = \frac{\sum_{i=1}^{N} \sqrt{(x_{i_{cal}} - x_{i_{given}})^2 + (y_{i_{cal}} - y_{i_{given}})^2 + (z_{i_{cal}} - z_{i_{given}})^2}}{N};$$

here, N is the total number of points, $(x_{i_{cal}}, y_{i_{cal}}, z_{i_{cal}})$ represents the coordinates of the calculated result (either time-interpolated or time-extrapolated) point and $(x_{i_{given}}, y_{i_{given}}, z_{i_{given}})$ represents the given coordinates of the point (either time-interpolated or time-extrapolated respectively), with the units are cm.

2. Analysis

   Time-interpolation: $RMS = 0.107$ at $t = 0.35$; at this moment, the radius of the time-interpolated cylinder is 2, and the outermost cylinder has a radius of 4; compared with these, the average distance (RMS) between the two corresponding points. which is 0.107, is acceptable.

   Time-extrapolation: $RMS = 0.830$ at $t = 1.3$; at this moment, the radius of the time-extrapolated cylinder is 4. Even though compared with the interpolation case, this result is worse, the overall result is still reasonable.

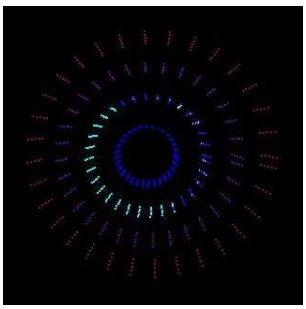


Figure 6.6: Validation of the time-interpolation result of the preliminary test case for [0,1] index model, $t = 0.35$. Analysis result: $totaldistance = 17.047$, $totalpoints = 160$, $averagedistanceRMS = 0.107$.

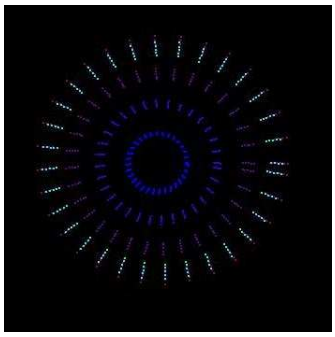


Figure 6.7: Validation of the time-extrapolation result of the preliminary test case for [0,1] index model, $t = 1.35$. Analysis result: $totaldistance = 132.280$, $totalpoints = 160$, $averagedistanceRMS = 0.830$.

6.2.2 Real-data test

In this subsection we will present the real-data test results for the Thin-plate spline model based on the [0,1] index including both the time-interpolation and time-extrapolation models[2]. In all tests described in this subsection, the smoothing parameter λ is equal to 1.

1. **Test case A**: *patient 1*

   This patient is a 16 year old girl (year 2000), 160.2cm in height and 54.7kg in weight. Also, she has been identified as a scoliotic patient and has used a brace before all the scans were taken. We have four groups of data available for her. Based on the time intervals, the time index list of this test case is $[0, 0.35, 0.71, 1]$. All of her available data is displayed in Fig. 6.8.

   We tried 3 different tests with our first real test case. The first two are concerning time-interpolation, one at $t = 0.35$ (see Fig. 6.9) and one at $t = 0.5$ (see Fig. 6.10). Test 1 is just for validation purposes because we already have the data available at $t = 0.35$; we want to know how well our model can work out on the real data. As we can see from the analysis: $RMS_{external}$ is 0.175 based on the minimum distance around the patient trunk at $t = 0.35$

[2]In this section, when we discuss the real test data, we use the term ¨Test case A¨, ¨Test case B¨ and ¨Test case C¨ to represent the 3 patients, and use ¨Test 1¨ or ¨Test 2¨ etc.. to represent the different tests we did for each patient.

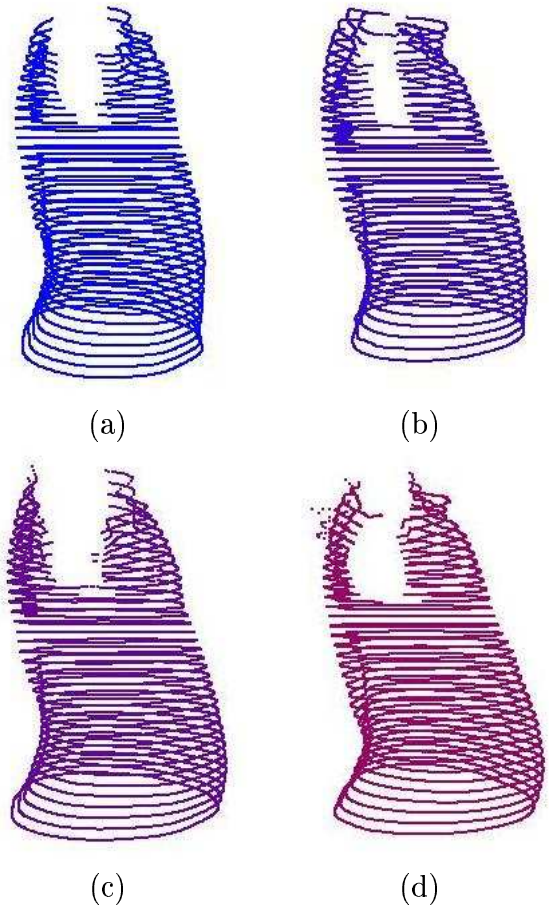
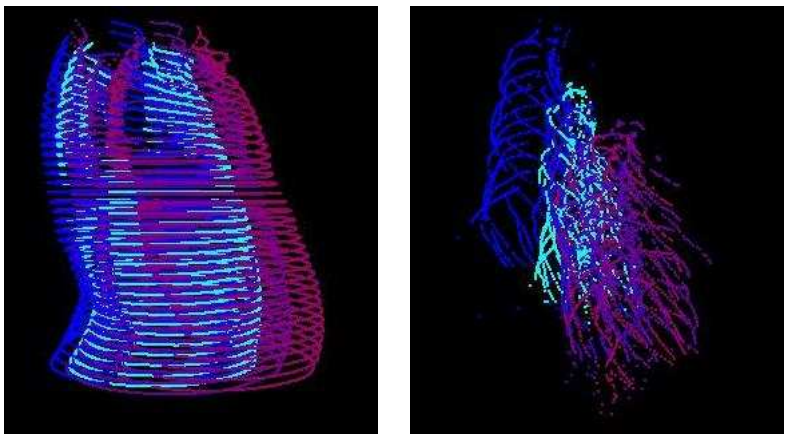


Figure 6.8: Available data for Test case A – patient 1, (a) is the figure at the first time point which is Nov 1998, (b) is the figure at the second time point which is May 1999, (c) is the figure at the third time point which is Nov 1999, and (d) is the figure at the fourth time point which is May 2000.

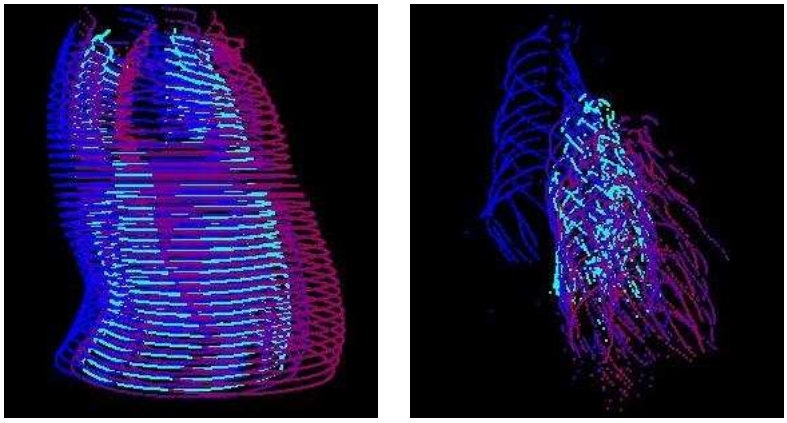
of $minDistance_{external} = 2.051$, so the external time-interpolation result is quite good; while for the internal data: $RMS_{internal}$ is 0.585 based on the minimum radius size of the vertebra $minDistance_{internal} = 0.143$, the error is big.

Test 2 was just to get some sense of time-interpolation (see Fig. 6.10). Test 3 is concerning the time-extrapolation at $t = 1.3$. Test case A is the one and only one available for the time-extrapolation test, because as we mentioned before, we need at least three time points for either time-interpolation or the time-extrapolation. We did the same validation for extrapolation on the real test cases as for the preliminary test cases: we ignore the group data at the last time point, get the data with our model, and then compare the two. The result is shown in Fig. 6.11; similar to the time-interpolation result, the external time-extrapolation is quite good: we have $RMS_{external} = 0.283$, based on the minimum distance around the patient trunk at $t = 0.35$ – $minDistance_{external} = 2.263$. But the internal time-extrapolated result is quite bad, we got $RMS_{internal} = 1.229$ based on the minimum radius size of the vertebra $minDistance_{internal} = 0.131$.



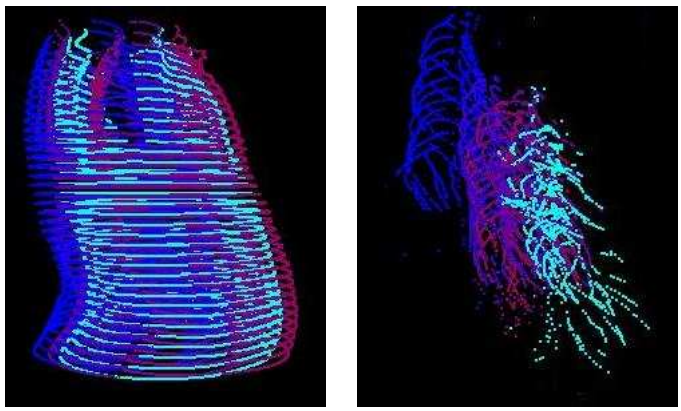
(a) (b)

Figure 6.9: Time-interpolation result 1 for Test case A for [0,1] index model at $t = 0.35$ which is exactly the second scan; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.175$, $RMS_{internal} = 0.585$, $minDistance_{external} = 2.051$, $minDistance_{internal} = 0.143$.



(a) (b)

Figure 6.10: Time-interpolation result 2 for Test case A for [0,1] index model at $t = 0.5$ which means 2.5 months after the second scan (around mid Aug 1999); (a) is the figure of the external body, (b) is the figure of the internal rib cage.

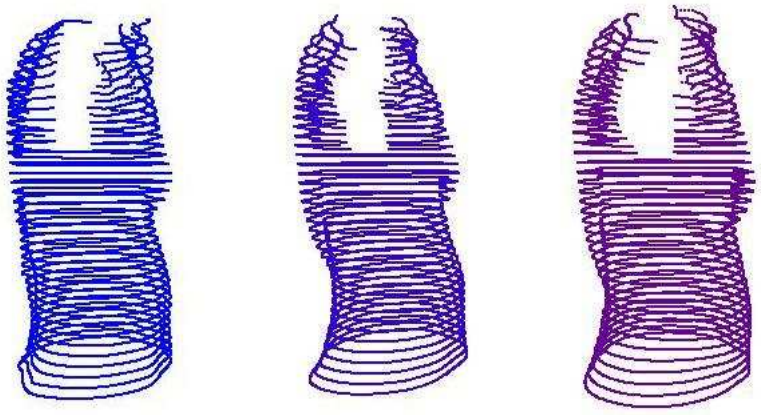


(a) (b)

Figure 6.11: Time-extrapolation result for Test case A for [0,1] index model at $t = 1.3$ which is exactly the fourth scan, (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.283$, $RMS_{internal} = 1.229$, $minDistance_{external} = 2.263$, $minDistance_{internal} = 0.131$.

2. **Test case B**: *patient 2*

   This patient is a 13 year old girl (year 2000), 160cm in height and 37.4kg in weight. She has been identified as a scoliotic patient and has used a brace before all the scans were taken. We have three groups of data available. Based on the time intervals, the time index list of this test case is $[0, 0.35, 1]$. All of her available data is displayed in Fig. 6.12:



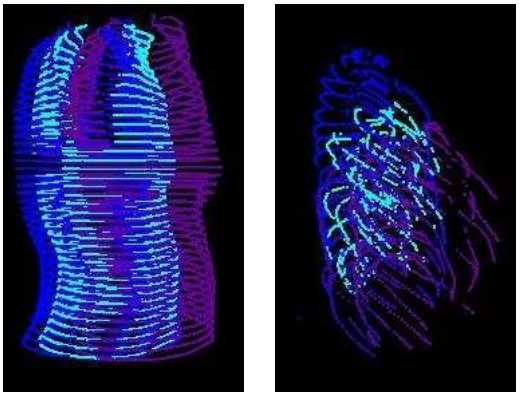
(a) (b) (c)

Figure 6.12: Available data for Test case B — patient 2, (a) is the figure at the first time point, which is Nov 1998, (b) is the figure at the second time point, which is May 1999, and (c) is the figure at the third time point, which is May 2000.

   For Test case B, we did two tests; the first was for time-interpolation at $t = 0.35$ with the validation results (see Fig. 6.13). The $RMS_{external}$ of the time-interpolation test is 0.085, based on the minimum distance around the body of 1.444; this is a very good time-interpolation. Still, the internal result is not good: $RMS_{internal} = 0.646$ and $minDistance_{internal} = 0.138$. Test 3 is for the time-extrapolation at $t = 1.3$ (see Fig. 6.14).

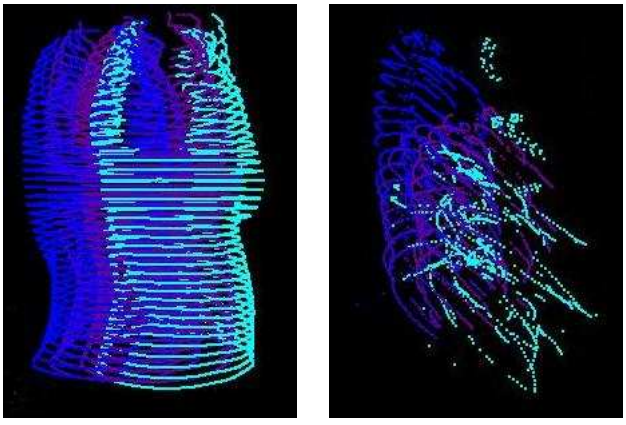
3. **Test case C**: *patient 3*

   This patient is a 13 year old girl (year 2000), 149cm in height and 49.8kg in weight. She has been identified as a scoliotic patient and has used brace before all the scans were taken. We have three groups of data available. Based on the time intervals, the time index list of this test case is $[0, 0.35, 1]$.



(a) (b)

Figure 6.13: Interpolation result for Test case B for [0,1] index model at $t = 0.35$ which is exact the second scan, (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.085$, $RMS_{internal} = 0.646$, $minDistance_{external} = 1.444$, $minDistance_{internal} = 0.138$.



(a) (b)

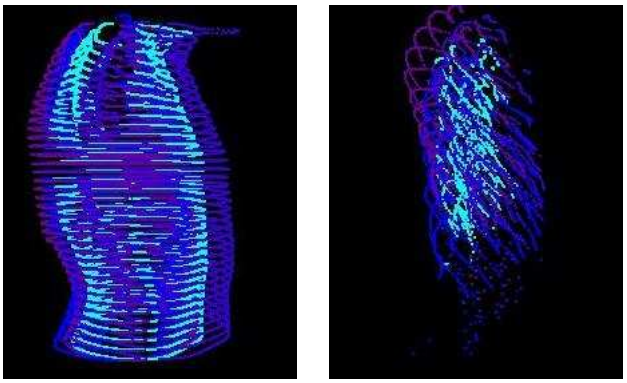
Figure 6.14: Extrapolation result for Test case B for [0,1] index model at $t = 1.3$, (a) is the figure of the external body, (b) is the figure of the internal rib cage.

All of her available data displayed in Fig. 6.15:

(a) (b) (c)

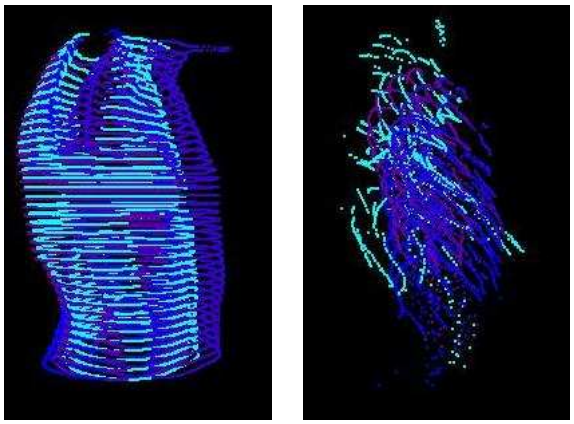
Figure 6.15: Available data for Test case C — patient 3; (a) is the figure at the first time point, which is May 1999, (b) is the figure at the second time point, which is Nov 1999, and (c) is the figure at the third time point, which is May 2000.

For Test case C, we did the same thing as for Test case B: the first test is for the time-interpolation at $t = 0.35$ with the validation results (see Fig. 6.16). The $RMS_{external}$ of the time-interpolation test is 0.300, based on the minimum distance around the body $minDistance_{external} = 1.321$. The $RMS_{internal} = 0.770$ and the $minDistance_{internal} = 0.127$. The other test is for the time-extrapolation at $t = 1.3$ (see Fig. 6.17).



(a) (b)

Figure 6.16: Time-interpolation result for Test case C for [0,1] index model at $t = 0.35$ which is exactly the second scan; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.300$, $RMS_{internal} = 0.770$, $minDistance_{external} = 1.321$, $minDistance_{internal} = 0.127$.



(a) (b)

Figure 6.17: Time-extrapolation result for Test case C for [0,1] index model at $t = 1.3$; (a) is the figure of the external body, (b) is the figure of the internal rib cage.

6.2.3 Possible applications for the [0,1] index model

From all the tests we have done with our data, the following applications of this model seem reasonable:

For time-interpolation, what the model can provide us is: given two groups of data specifying the start and the final status (either direction), then we can get any in-between status. This can help us to trace the process of the deformation between two time points. For example, in the medical application, it can help the doctor to better see how the internal spine changes as time passes, and meanwhile, this helps reduce the number of X-rays, since we can prolong the time interval between two scans, and use this model for the in-between information.

For extrapolation, what the model can provide us is: given a list of data at several time points (at least 2), we can predict the figure at a certain time point after that. Of course, the closer the time point to those available ones, the better the result will be. This application is quite interesting: it can help us to predict what will happen after a certain time period. This is especially useful in the medical context, since it may help the doctor to choose the most appropriate treatment for the patient.

6.3 The case $\mathcal{T} = R^d$

6.3.1 Preliminary experimental results

In this section we will present the preliminary experimental results of the Thin-plate spline model that has been implemented for both the cases of interpolation and approximation. We tried the model with both two-dimensional and three-dimensional objects. For the case of 2D, we use a 14×14 plane as the target (see Fig. 6.18), and a cylinder of radius 2 with a vertical line inside for the 3D case (see Fig. 6.19). Our goal here was to obtain some intuitive understanding of the behavior of the method.

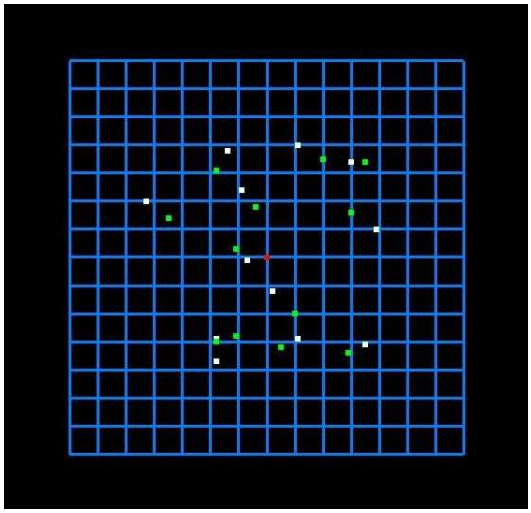


Figure 6.18: Preliminary 2D test case for R^d index model — 2D plane, $N = 12$ marker points with white dots representing the original marker points, and green dots representing the deformed ones.

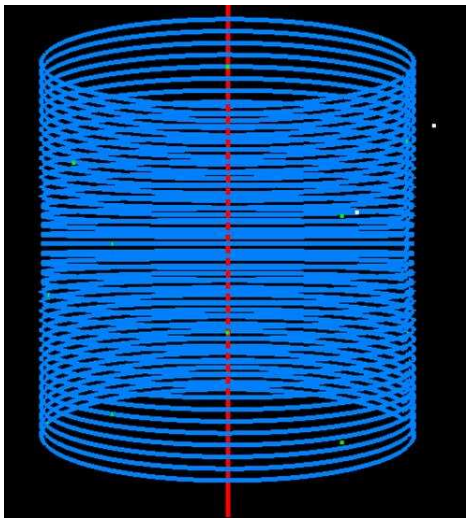


Figure 6.19: Preliminary 3D test case for R^d index model — 3D cylinder, $N = 12$ marker points with white representing the original marker points, and green representing the deformed ones.

- **Two–dimensional Test ($d = 2$)**

  When $m = 2$ (where m was defined on p. 29) ($M = 3$), $\lambda = 0$, a small number of marker-points ($N = 5$) permits the interpolation of marker points to work well, but the overall deformation behaves badly: the small quantity of marker points makes some part of the deformation out of control, which leads to unexpected changes in the final figure. But when we test with the data similar to [19] with $N = 12$, the whole image looks almost the same as shown in [19, p. 528]. Introducing values of λ which are not equal to zero (we used the values 0.001, 0.01 and 0.1) changes the overall method from interpolation to approximation. Besides getting the same visual results as in [19] (see Fig. 6.20), we also found out that the larger the value of λ the smoother the lines appear. In particular we tried with $\lambda = 1$ which according to [19] is a quite large value and the grid contains almost straight lines only. A value of λ equal to 500, which causes the $\boldsymbol{w}$ vector to have only zero values, leads to almost no visible change in the figure, but the quality of the approximation of marker points decreases slightly (see Fig. 6.20 (f)). In summary, our implementation appears to reproduce the results given in [19].

  When $m = 3$ ($M = 6$), $\lambda = 0$, we used the same group of data as for $m = 2$, only changing the value of m (see Fig. 6.21). The case $m = 3$ is not discussed in [19], so we cannot compare results. The interpolation of marker points presents no problem, but concerning the overall figure, the deformation causes very bad distortion. Introducing non-zero values of λ again makes the overall deformation look more reasonable. Still, when we increase the value of λ from 1 to some large value like 500, the image shows no obvious change, but this large λ makes the smoothing part weigh more and more, and finally the $\boldsymbol{w}$ vector contains zeroes only. As in the case $m = 2$, when we increase the number of marker points, the deformation works better in all cases. The more uniform the distribution of the marker points, the smoother is the overall deformation. The method with $m = 3$

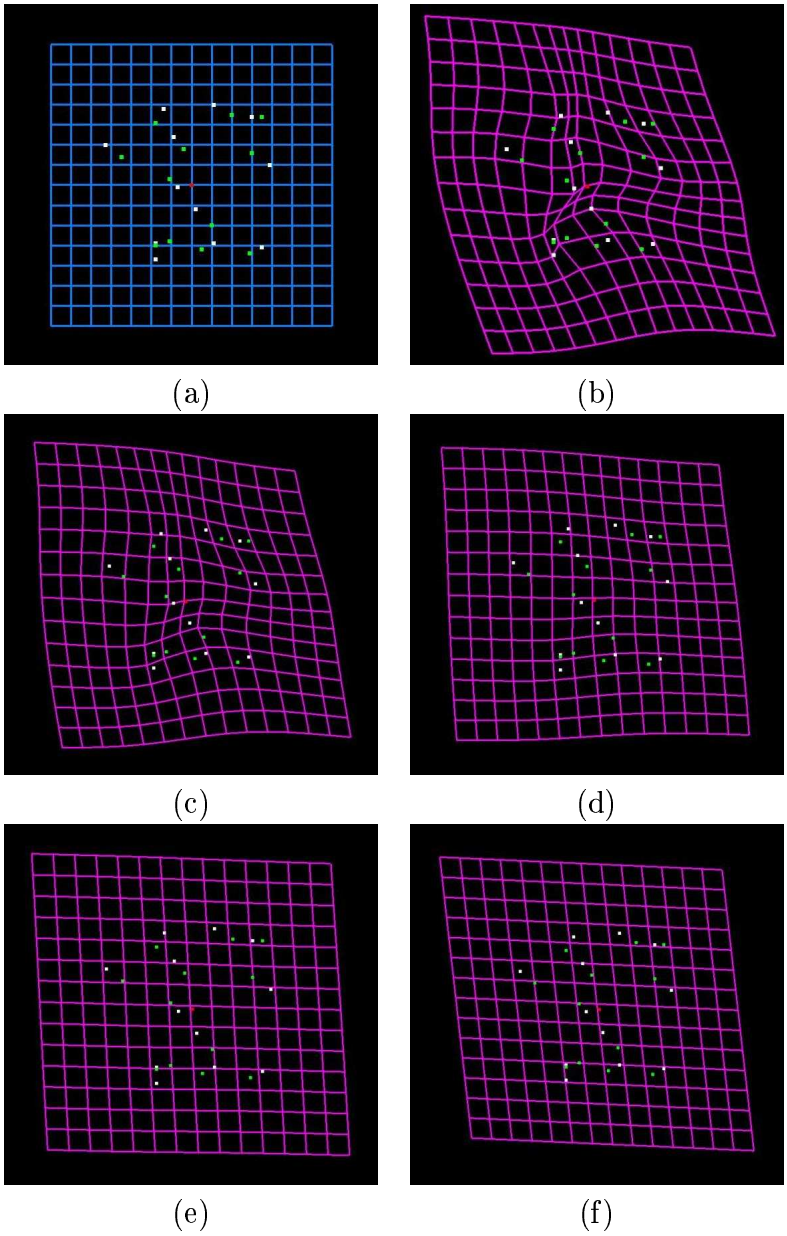


Figure 6.20: Test results for the 2-dimensional case with m=2 for R^d index mode, $d = 2$. Two different marker point sets (original and deformed) represented by green dots and white dots respectively. (a) original 2D plane; (b) interpolation $\lambda = 0$; (c) approximation $\lambda = 0.01$; (d) approximation $\lambda = 0.1$; (e) approximation $\lambda = 1$; (f) approximation $\lambda = 500$.

seems to be more sensitive in this respect than the method with $m = 2$.

- **Three-dimensional Test ($d = 3$)**

  The case $m = 2$ ($M = 4$), $\lambda = 0$, with 12 marker points (see Fig. 6.2) is the prototype for the test model. Both the interpolation and the overall deformation work pretty well. The only shortcoming is that the deformed cylinder is not smooth enough (see Fig. 6.22 for interpolation). When we increase the value of λ, the method changes from interpolation to approximation. This is quite visible in the results (see Fig. 6.23 – 6.25). As for the overall deformation, the result is not so good because the smoothing factor has made the figure (the external cylinder which corresponds to the patient body in our application) include only straight lines. This is true in particular when the value of λ becomes quite large, e.g. 100 (see Fig. 6.25), in which case the $\boldsymbol{w}$ array has zeroes everywhere.

  When $m = 3$ ($M = 10$), the minimum number of marker points is $N = 10$. We started our test with 11 marker points. When $\lambda = 0$, the interpolation of the marker points is quite good, while the overall deformation, on the contrary, for some cases, shows very large distortions. The spine which is located inside the cylinder now appears outside of the original cylinder (see Fig. 6.26). While for some examples, the deformation is quite acceptable, the badly deformed part comes from the very small number of marker points.

  After testing with several groups of data, we found that the quality of the deformation is largely decided by the location of the marker points. A well-located set of marker points can maintain a very nice deformation, although the *number* of marker points is also very important. We even tried a test case of 54 marker points distributed around the whole cylinder (see Fig. 6.27). But meanwhile we also found another solution for this problem (uncontrolled deformation) — put some marker points, at least one, inside the cylinder. Such marker points can help to realize the constraint on the final transformation; this is especially useful for us concerning the

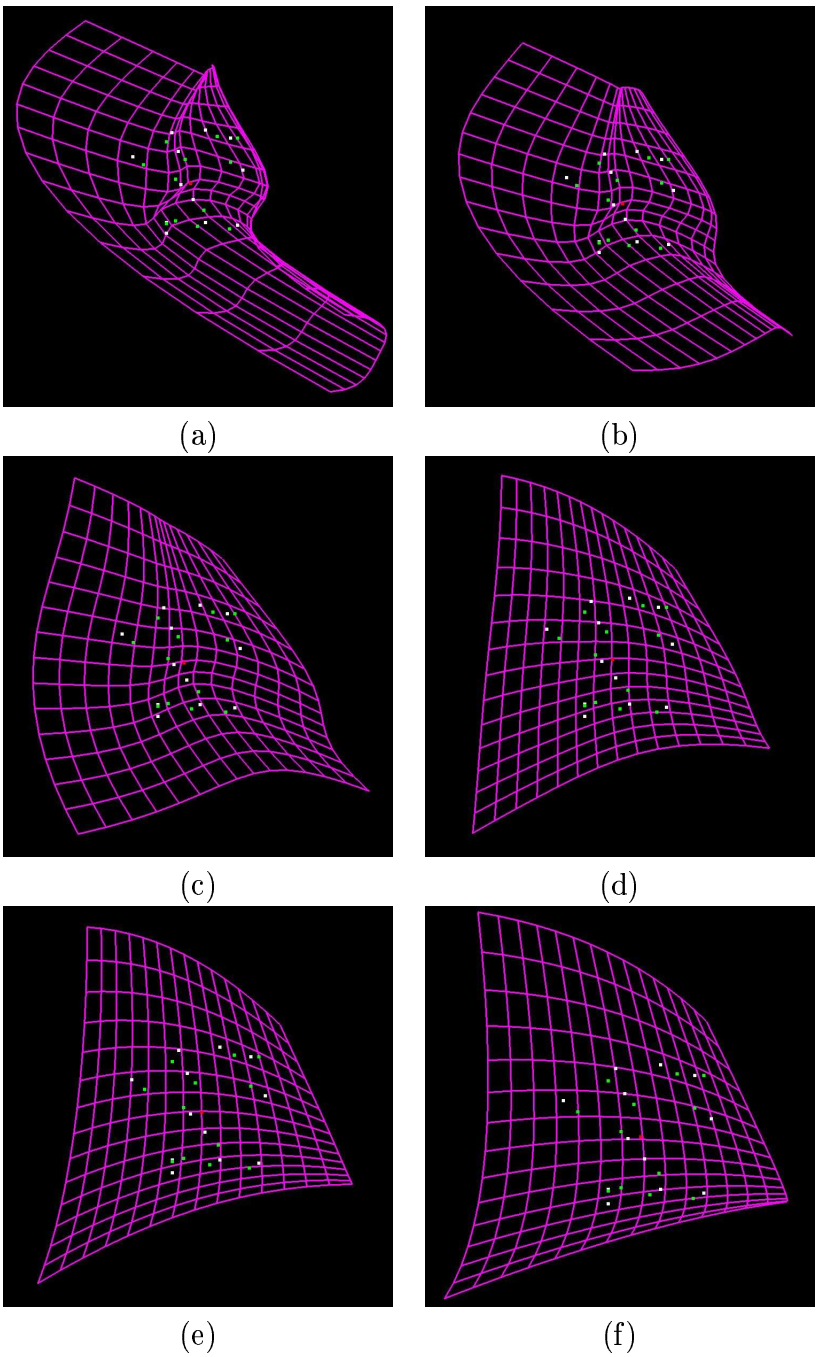
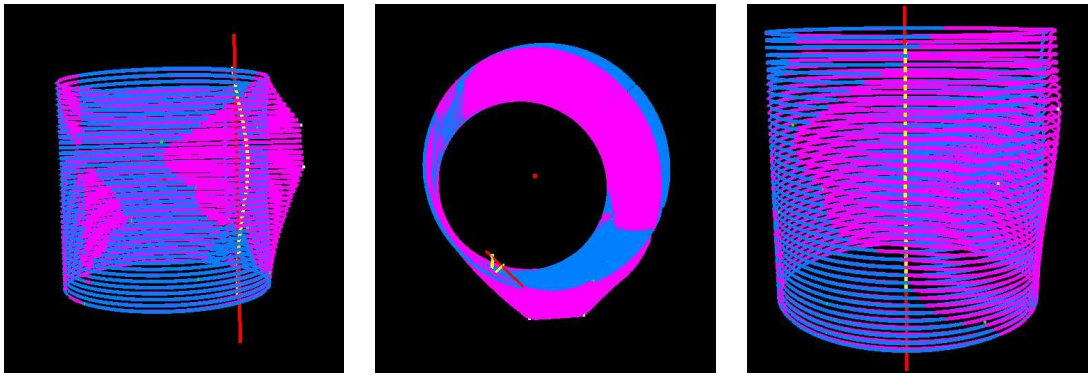
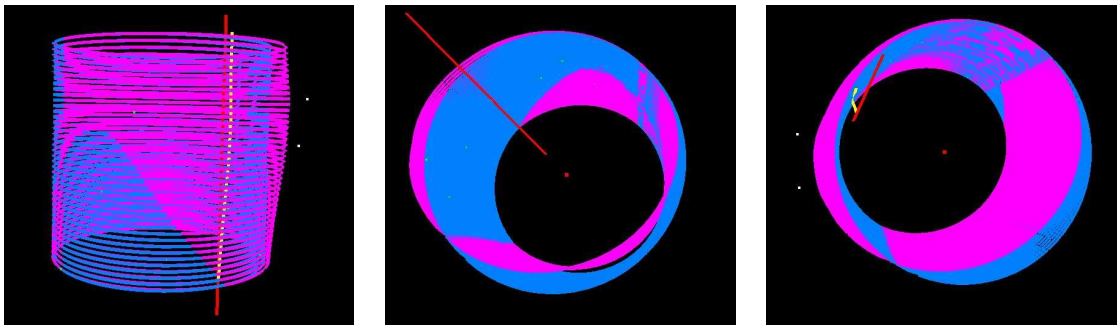


Figure 6.21: Test results for the 2-dimensional case with m=3 for R^d index model, $d = 2$. Two different marker point sets (original and deformed) represented by green dots and white dots respectively. (a) original 2D plane; (b) interpolation $\lambda = 0$; (c) approximation $\lambda = 0.01$; (d) approximation $\lambda = 0.1$; (e) approximation $\lambda = 1$; (f) approximation $\lambda = 500$.



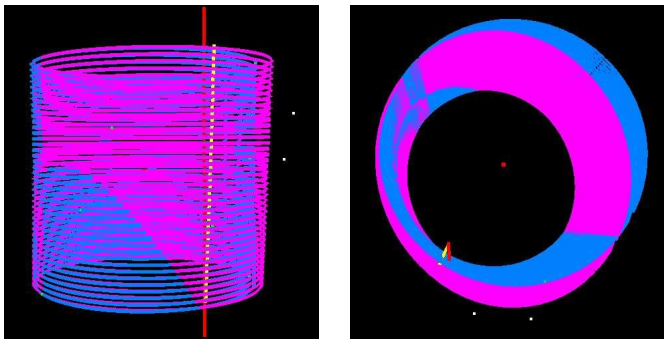
(a) (b) (c)

Figure 6.22: Test result for the 3-dimensional object with $m = 2$, $N = 12$ and $\lambda = 0$ — interpolation for R^d index model.



(a) (b) (c)

Figure 6.23: Test case for the 3-dimensional object with $m = 2$, $N = 12$ and $\lambda = 0.01$ — approximation for R^d index model.



(a) (b)

Figure 6.24: Test case for the 3-dimensional object with $m = 2$, $N = 12$ and $\lambda = 0.1$ — approximation for R^d index model.

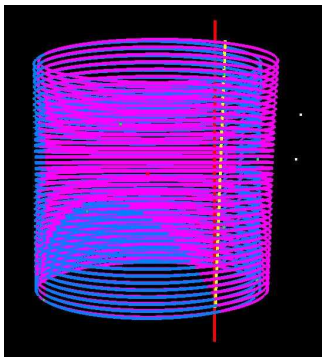
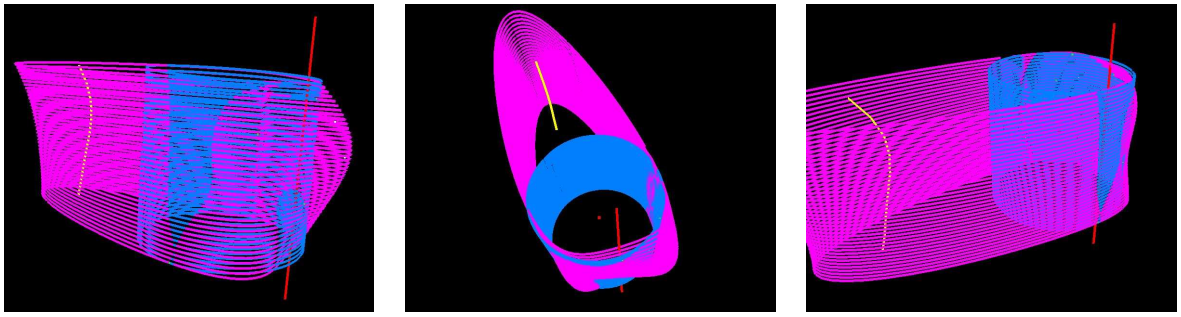


Figure 6.25: Test case for the 3-dimensional object with $m = 2$, $N = 12$ and $\lambda = 100$ — approximation for R^d index model, a deformed cylinder, in which the smoothing parameter makes the external cylinder almost straight.

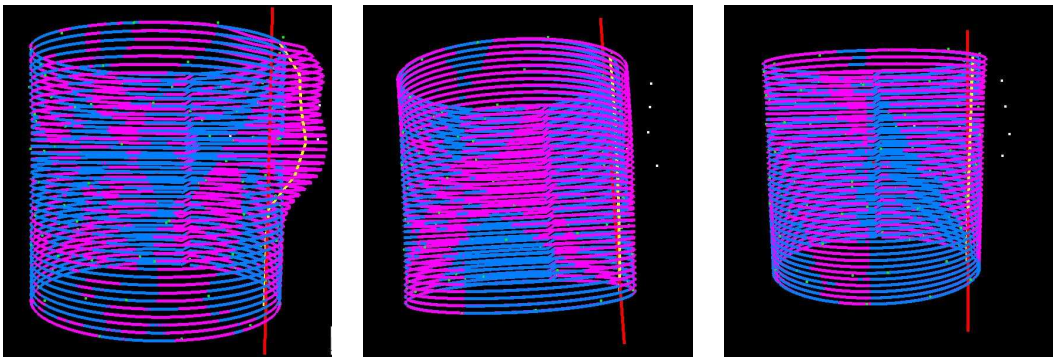
deformation of the spine inside the cylinder. But this solution is often not reasonable in practice, for example, in our application, it is impossible for us to put a marker point inside the patient's body (but see the remark on trend data, below). Finally, when we change the value of λ, the whole result goes from interpolation to approximation, similar to the case of 2D (see Fig. 6.27 (b) and (c)).



(a) (b) (c)

Figure 6.26: Test case for the 3-dimensional object with $m = 3$, $N = 11$ and $\lambda = 0$ — interpolation for R^d index model, deformation shows large distortions.

In summary, the method of TPS works quite well in both cases of $d = 2$ and $d = 3$ when $m = 2$ for both interpolation and approximation. But the number of marker points and their locations matters a lot. Usually the more the number of marker points the better the final result, and the better the distribution of the marker points the better the final result. But we have to point out that when



(a) (b) (c)

Figure 6.27: Test case for the 3-dimensional object with $m = 3$, $N = 54$ for R^d index model, (a) is the interpolation result in which $\lambda = 0$, (b) is the approximation result with $\lambda = 1$ and (c) is approximation with $\lambda = 10$.

more than two marker points lie in one line the whole method will crash. For the cases when $m = 3$, the validation of the TPS is largely based on the location and the number of the marker points. For $m = 3$, the deformed figure looks better than in the case when $m = 2$, in the sense that it is more smooth, because of the second derivatives in the objective functions.

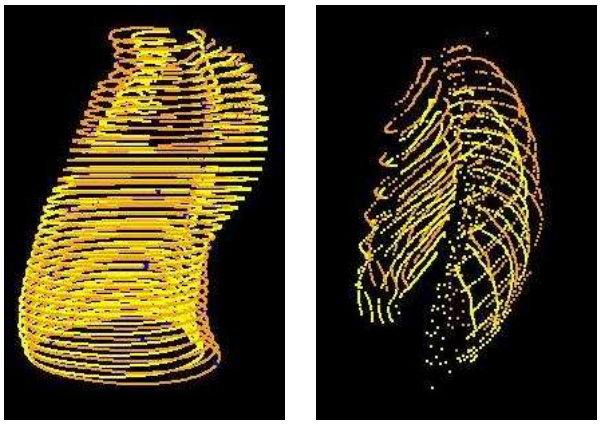
There appear to be two ways to obtain a well-behaved overall transformation. The first is to distribute the marker equally points throughout the space of interest. We did the test with 108 marker points uniformly distributed around the cylinder, and the deformation behaves very well; even in the case of 54 marker points on half of the cylinder only, we still get quite reasonable results. All of this showed that it is quite possible to obtain a good deformation when $m = 3$. The second solution (in the context of human trunk) is to have marker points located in the interior of the trunk (in practice, a possible way is to get this interior information from the X-rays). For purposes of developing a method to predict the spinal deformation from external data, it appears that we should well-locate the marker points, and distribute them uniformly over the patient's back. We should also try to obtain some data related to the interior of the trunk, perhaps obtained from the trend data or time-series data of the spine. Finally, we should also make the number of marker points as large as practically possible.

6.3.2 Real-data test

In this subsection we will present the real-data test results of the Thin-plate spline model based on the R^d index including both the interpolation and approximation models together with the validations for the interpolation test. In this subsection, in each result display, there are two images: the orange one represents the initial figure and the yellow one represents the deformed result. These two images should be the same, or as close as possible.

1. **Test case A**: *patient 1*

   For Test case A, we tried 3 different tests. The first one (Test 1) is interpolation with 40 marker points around the trunk. We get the $RMS_{external} = 0.247$ based on the minimum distance around the body $minDistance_{external} = 2.041$; and for the internal spine interpolation result: $RMS_{internal} = 0.695$ with $minDistance_{internal} = 0.143$ (see Fig. 6.28). Again the external interpolation result is pretty good, while the internal one is bad.

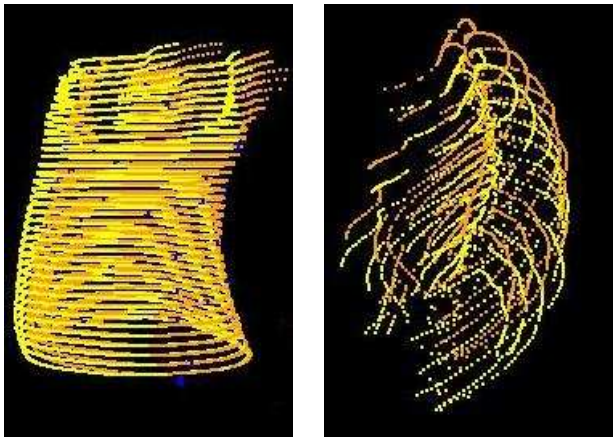


(a) (b)

Figure 6.28: Interpolation test 1 for Test case A for R^d index model from Nov98 to May99, with the orange one representing the given final figure, and the yellow image the figure as deformed by the model, 40 marker points, $\lambda = 0$; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.247$, $RMS_{internal} = 0.695$, $minDistance_{external} = 2.041$, $minDistance_{internal} = 0.143$.

Another test (Test 2) involved increasing the number of marker points

around the body to 76 marker points. Here as we expected, the error for both the internal and external interpolation decreases: $RMS_{external} = 0.180$ based on the minimum distance around the body $minDistance_{external} = 2.262$; and for the internal spine interpolation result: $RMS_{internal} = 0.495$ with $minDistance_{internal} = 0.131$ (see Fig. 6.29).



(a) (b)

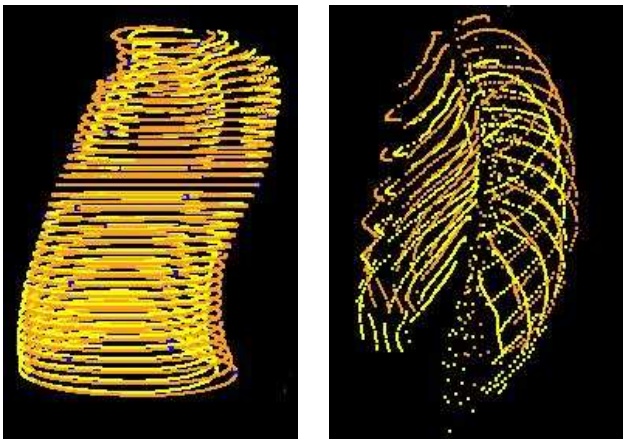
Figure 6.29: Interpolation test 2 for Test case A for R^d index model from Nov99 to May00, 76 marker points, (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.180$, $RMS_{internal} = 0.495$, $minDistance_{external} = 2.263$, $minDistance_{internal} = 0.131$.

The third test we did involved approximation, with exactly the same group of test data, only changing the value of λ from 0 to 0.1 (see Fig. 6.30).

2. **Test case B**: *patient 2*

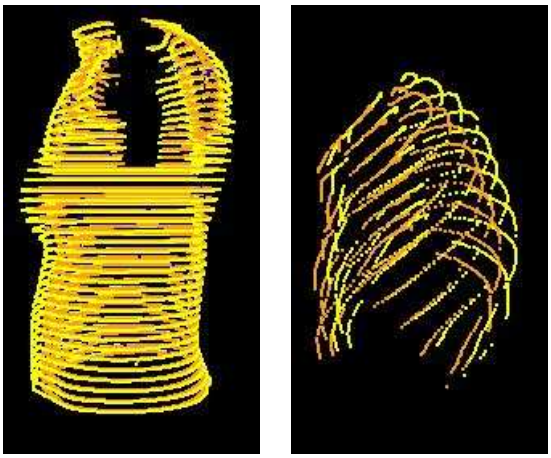
   For Test case B, we worked with 40 marker points on all the groups of data, and the performance of the model is relatively stable. Fig. 6.31 is the test result of our first test for interpolation case. Here, $RMS_{external} = 0.329$ based on $minDistance_{external} = 1.361$, a quite good result. And $RMS_{internal} = 0.656$ based $minDistance_{internal} = 0.131$.

   For the rest of the interpolation test cases, we will not show all the figures here, but only the validation results.



(a) (b)

Figure 6.30: Approximation test for Test case A for R^d index model from Nov98 to May99, 40 marker points, $\lambda = 0.1$, (a) is the figure of the external body, (b) is the figure of the internal rib cage.



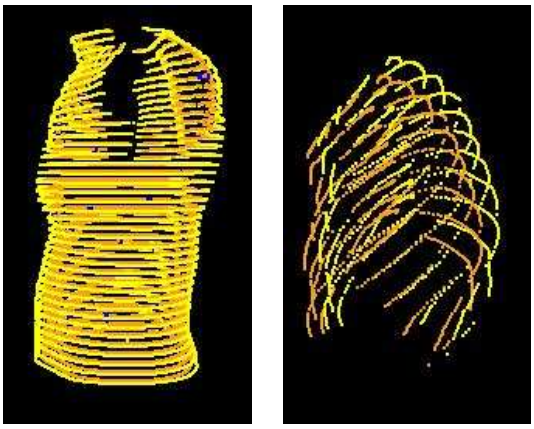
(a) (b)

Figure 6.31: Interpolation test for Test case B for R^d index model from Nov98 to May00, 40 marker points, $\lambda = 0$, (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.329$, $RMS_{internal} = 0.656$, $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$.

Table 6.3: Test results of Test case B

Parameter	May99–May00	May00–Nov98
ExternalRMS	0.282144	0.193.83
InternalRMS	0.646784	0.704741
ExternalMinDistance	1.361056	1.24849
InternalMinDistance	0.130843	0.152557

Similarly, the last test we tried with this test case was approximation, with $\lambda = 0.1$ (see Fig. 6.32).

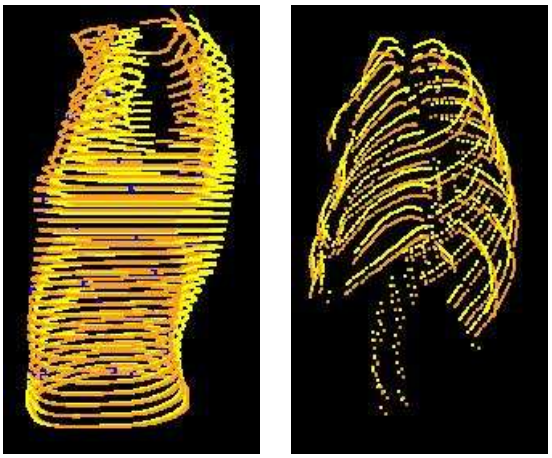


(a) (b)

Figure 6.32: Approximation test for Test case B for R^d index model from Nov98 to May00, 40 marker points, $\lambda = 0.1$, (a) is the figure of the external body, (b) is the figure of the internal rib cage.

3. **Test case C**: *patient 3*

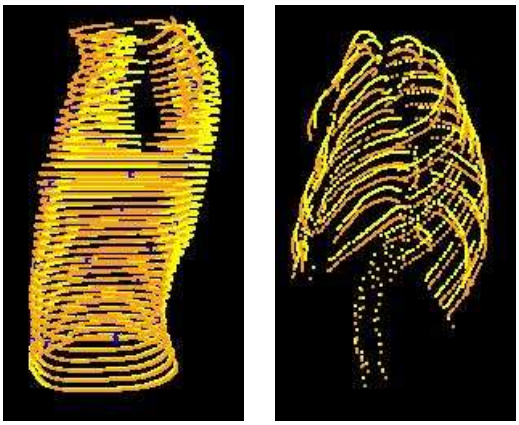
   For Test case C, we worked with 40 marker points on all the groups of data. Fig. 6.33 is the test result of our first test for the interpolation case. Here, $RMS_{external}$ is 0.270 based on the minimum distance around the body $minDistance_{external} = 1.263$. And $RMS_{internal} = 0.779$ based on $minDistance_{internal} = 0.127$.



(a) (b)

Figure 6.33: Interpolation test for Test case C for R^d index model from May99 to May00, 40 marker points, $\lambda = 0$; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.270$, $RMS_{internal} = 0.779$, $minDistance_{external} = 1.263$, $minDistance_{internal} = 0.127$.

   Finally, the last test we tried with this test case was approximation, with $\lambda = 0.1$ (see Fig. 6.34).



(a) (b)

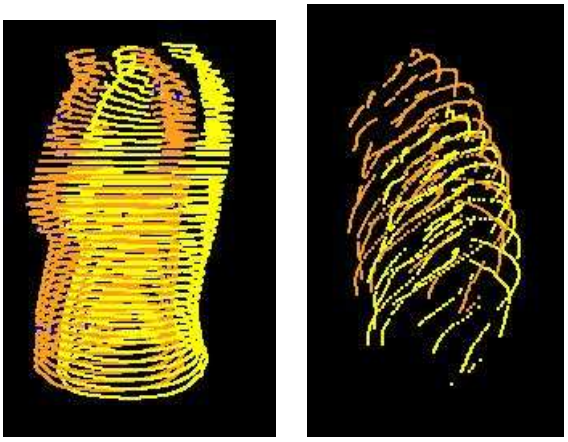
Figure 6.34: Approximation test for Test case C for R^d index model from May99 to May00, 40 marker points, $\lambda = 0.1$; (a) is the figure of the external body, (b) is the figure of the internal rib cage.

4. **Test case D:** "Special coefficient vector generalization" test

   A special test we did for the Thin-plate spline model on R^d was a "coefficient vector generalization" test. In contrast to the [0,1] model, in the R^d model, all the space points in one model share the same two coefficient vectors **a** and **w**. This fact interests us a lot, since all three test cases are of the same type of scoliosis and similar degree of change in figure (see Table 6.1), and there should be some similarities amongst these vectors. And if we extend this idea further, there may be some possibility of characterizing the coefficient vectors for each kind of deformation.

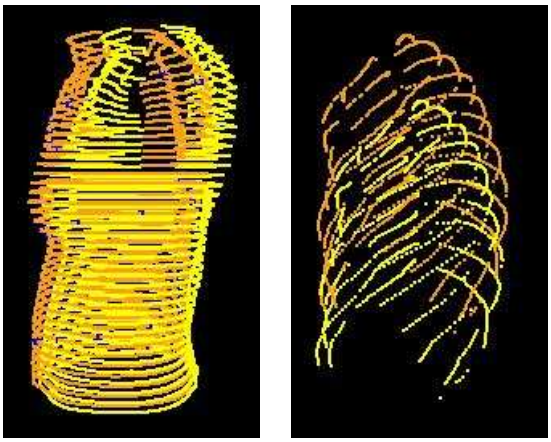
   Based on this motivation, what we tried was to apply the coefficient vectors **a** and **w** of Test case A (patient1 from Nov98 to May99) to the latter two test cases (Test case B and C). The results for Test case B — patient2 are not bad, especially for the internal result (see Fig. 6.35 and 6.36). Here $RMS_{external} = 0.619$ and $RMS_{internal} = 0.489$, based on $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$ respectively for Test 1. And $RMS_{external} = 0.640$ and $RMS_{internal} = 0.477$, based on $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$ respectively.

   Another special test we did for this case was to use the same coefficient vec-



(a) (b)

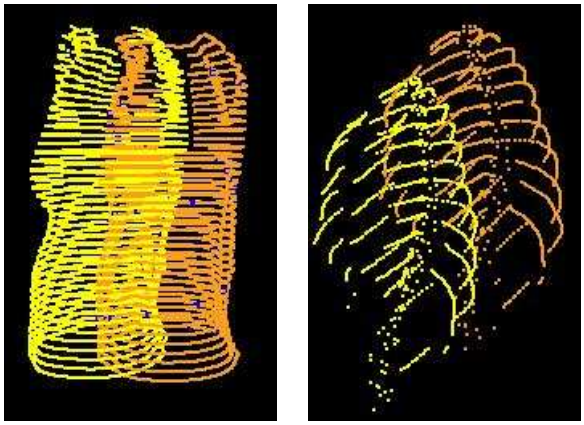
Figure 6.35: Test 1 with Test case B for R^d index model — patient 2 from Nov98 to May00, 40 marker points with the coefficient matrices of Test case A (patient1 from Nov98 to May99); (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.619$, $RMS_{internal} = 0.489$, $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$.



(a) (b)

Figure 6.36: Test 2 with Test case B for R^d index model — patient 2 from May99 to May00, 40 marker points with the coefficient matrices of Test case A (patient1 from Nov98 to May99); (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.640$, $RMS_{internal} = 0.477$, $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$.

tors as the previous tests, which are those of patient1 from Nov98 to May00, for the case of patient2 from May00 to Nov98. Since the two deformations are in the opposite time direction, we expected the RMS to have a quite big value, and this was confirmed: we got $RMS_{external} = 0.725$ based on 1.736, and $RMS_{internal} = 0.482$ based on 0.131 (see Fig. 6.37).

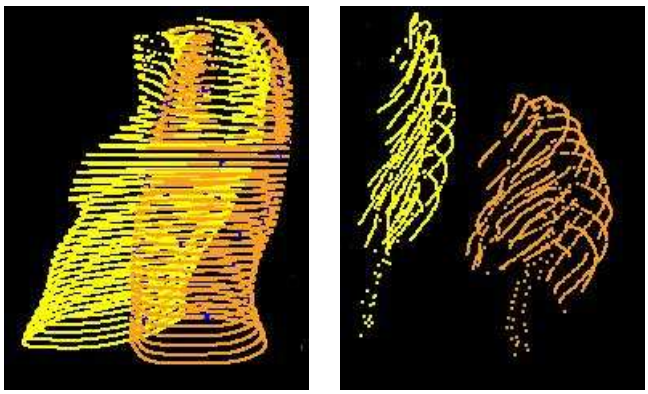


(a) (b)

Figure 6.37: Test 3 with Test case B for R^d index model— patient 2 from May99 to May00, 40 marker points with the coefficient matrices of patient1 from Nov98 to May99, (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.725$, $RMS_{internal} = 0.482$, $minDistance_{external} = 1.361$, $minDistance_{internal} = 0.131$.

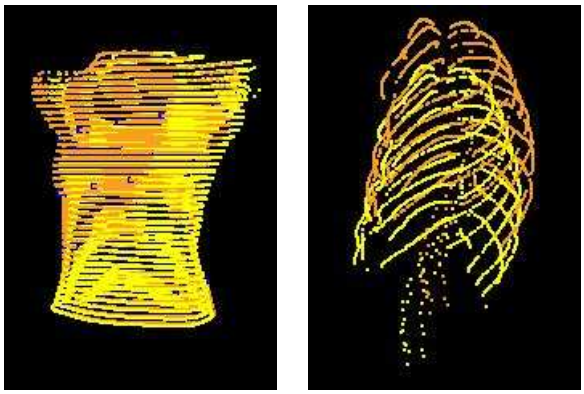
Similar test results (using the coefficient vectors of Test case A) with Test Case C patient 3 are not that promising for the case of interpolation. The value of RMS is quite big, $RMS_{external} = 0.649$, $RMS_{internal} = 0.719$ based on $minDistance_{external} = 0.126$ and $minDistance_{internal} = 0.127$ respectively (see Fig. 6.38). For the approximation case, however, when $\lambda = 0.1$, the result is much better, $RMS_{external} = 0.270$, $RMS_{internal} = 0.779$ based on $minDistance_{external} = 1.263$ and $minDistance_{internal} = 0.127$ respectively (see Fig.6.39).

Generally speaking, the Thin-plate spline model on R^d works quite well, especially for the simple deformation case. For the ¨coefficient vector generalization¨ case,



(a) (b)

Figure 6.38: Test 1 with Test case C for R^d index model — patient 3 from May99 to May00, 40 marker points with the coefficient matrices of patient1 from Nov98 to May99; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.649$, $RMS_{internal} = 0.719$, $minDistance_{external} = 1.262$, $minDistance_{internal} = 0.127$.



(a) (b)

Figure 6.39: Test 2 with Test case C for R^d index model — patient 3 from May99 to May00, 40 marker points with the coefficient matrices of patient1 from Nov98 to May99, both are in the case of approximation at $\lambda = 0.1$; (a) is the figure of the external body, (b) is the figure of the internal rib cage. Analysis result: $RMS_{external} = 0.270$, $RMS_{internal} = 0.779$, $minDistance_{external} = 1.262$, $minDistance_{internal} = 0.127$.

our tests indicate that the situation is at least promising, since Test case A and Test case B have quite similar forms.

6.3.3 Possible applications for the R^d index model

There are two main possible applications for our R^d index model.

The first one, which was our original motivation, is to avoid X-rays as much as possible. By applying the R^d model, what we can have is with two groups of marker points and one group of complete figure points containing one of the two groups of marker points, we can predict the other group of complete figure points. In this way, we might predict the form of the internal rib cage from the external figure. Again, in our medical application, the external figure is much easier to get than the internal, and this can largely reduce the number of X-rays that need to be taken. But the data accuracy problem may be a limitation here.

The second application is classification of different kinds of deformations by means of the coefficient vectors. This may help a lot with medical research, and it also reduces the number of X-rays that the patient needs to take.

Chapter 7

Conclusions

This research was devoted to scoliosis predictions, including those based on the external data to get the internal spine information, and those based on a series of data at different time points to get information at a future point. The main conclusions from this work can be summarized as follows.

Scoliosis, with its high occurrence among teenagers, has become one of those subjects that interest medical researchers a lot. The limits of current diagnosis methods induce us to find some prediction models, in order to avoid potentially harmful X-rays. Based on all the information available for the patient, we proposed two different models: Thin-plate spline based on R^d index model and based on [0,1] index model.

For the R^d index model, the main working process can be summarized in Fig. 7.1: given two groups of marker points and a complete point-based figure, we can hope to get the deformed figure including both the internal spine and external trunk.

For the [0,1] index model, the main working process can be summarized in Fig. 7.2: given a list of time points and the complete point-based figure, we can get the figure at any in-between time point (corresponds to time-interpolation), and later-on time points (corresponds to time-extrapolation).

The validation we did with our experiments, which was illustrated in Chapter 6, shows that for the R^d model: when $d = 2$, the method works very well; when

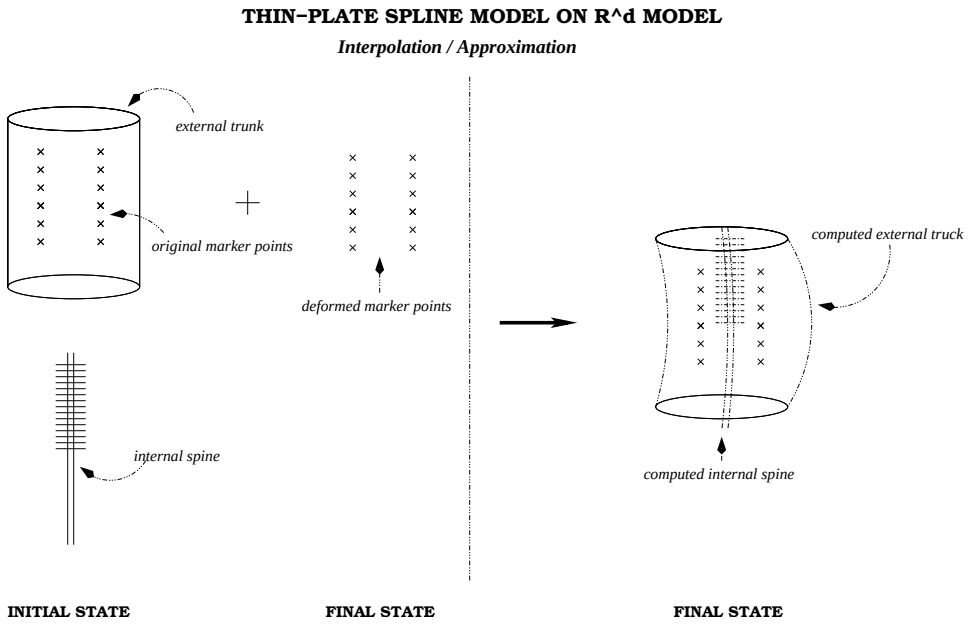


Figure 7.1: Illustration of the working process of Thin-plate spline model on R^d index set.

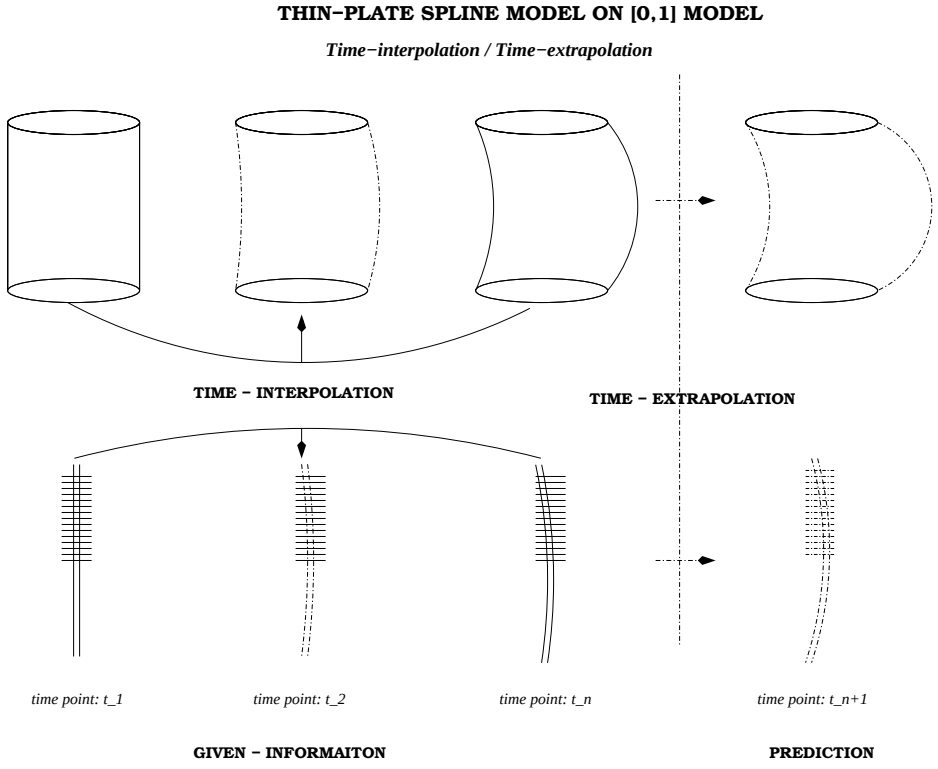


Figure 7.2: Illustration of the working process of Thin-plate spline model on [0,1] index set.

$d = 3$, for the preliminary models, the result is still quite good; but for the real test data, the deformation results for the external trunk is pretty good, while that for the internal spine is not that good. But if we look at the figure, we can see part of the RMS comes from the position of the deformed model, while the shapes of the figure are quite close actually. And for the [0,1] index model, the ideal preliminary test gave us a very good result. But for the real test case, similar results to those of the R^d model show up again: the external time-interpolated and time-extrapolated are both quite good, but the internal result shows large errors. For the time-extrapolation test, since we only have one real test case, Test case A, the result is not that convincing. Part of these large errors come from the problems intrinsic to the real test data, as we described in early Chapter 6. Since we are constrained by the inaccuracy of the data, the final results are reasonable.

We used the programming language C++ and OpenGL to implement the models. And for part of the mathematical calculation, we used Matrix Template Library (MTL).

The models described here could be quite useful in medical applications. They can reduce the frequency of the X-ray taken for the patient, it can help the doctor to trace the in-between deformation between any two states, plus, it also provides the function to do prediction at a future time point.

The problems of data accuracy must be resolved, however, and future work could focus on this problem. For example, as we described in Chapter 3, the "Affine-affine matching" model, and in Chapter 4, the combination of the two models, are promising ideas to explore. All of these could increase the accuracy of the results; also, we can work on the real data, and improve increase its quality.

The less good deformation results of the internal rib-cage may be due to the absence of markers, but also because the deformation of the bone structure is not the same as that of the soft tissues of the trunk. Then a possible solution could be to characterize both types of deformation.

Bibliography

[1] www.grbb.polymtl.ca.

[2] www.spineuniverse.com.

[3] Anatomical Chart Co., Skokie, Illinois. *The human spine — disorders*, 1996.

[4] D. Bechmann. Space deformation models survey. *Computers and Graphics*, 18(4):571–586, 1994.

[5] F. L. Bookstein. Principle warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989.

[6] P. Borrel and D. Bechmann. Deformation of n-dimensional objects. Technical report, IBM Research Division, 1990.

[7] P. Borrel and D. Bechmann. Deformation of n-dimensional objects. *International Journal of Computational Geometry and Applications*, 1(4):137–155, 1991.

[8] J. X. Chen, H. Wechsler, J. M. Pullen, Y. Zhu, and E. B. MacMahon. Knee surgery assistance: patient model construction, motion simulation. and biomechanical visualization. *IEEE Transaction on Biomedical Engineering*, 48(9):1042–1052, September 2001.

[9] S. Coquillart. Extended free-form deformations: A sculpturing tool for 3d geometric modeling. *Computer Graphics*, 24(4):187–193, August 1990.

[10] P. H. Dangerfield, D. Scutt, J. C. Dorgan, Y. Li, J. D. Pearson, and D. A. Brodie. The effect of body position on the three-dimensional surface deformity of scoliosis. *Bone and Joint Surgery*, 307(74-B):139–147, 1992.

[11] C. J. Goldberg, M. Kaliszer, D. P. Moore, E. E. Eogarty, and F. E. Dowling. Surface topography, Cobb angles, and cosmetic change in scoliosis. *Spine*, (26):55–63, 2001.

[12] W. M. Hsu, J. F. Hughes, and H. Kaufman. Direct manipulation on free-form deformation. *Computer Graphics*, 26(2):177–184, 1992.

[13] P. Kalra, A. Mangili, N. M. Thalmann, and D. Thalmann. Simulation of facial muscle actions based on rational free-form deformation. *EUROGRAPHICS*, 11(3):59–69, 1992.

[14] A. Liu and Y. Wang. Hypothesis testing in smoothing spline models. Technical Report 387, Department of statistics and applied probability, UCSB, 2002.

[15] R. Neeser. The use of spatial deformation for correcting the taphonomic distortion of hominid crania in south Africa: a proposal. 2002. www.cs.uct.ac.za/Research/CVC/Projects/current/rneeser/rneeser.html.

[16] L. Piegl and W. Tiller. *The NURBS Book*. Springer, 1997.

[17] H. Qin and D. Terzopoulos. Dynamic NURBS swung surfaces for physics-based shape design. *Computer aided design*, 27(2):111–127, 1995.

[18] H. Qin and D. Terzopoulos. D-NURBS: a physics-based framework for geometric design. *IEEE Transactions on visualization and computer graphics*, 2(1):85–96, March 1996.

[19] K. Rohr, H. S. Stiehl, R. Sprengel, T.M. Buzug, J. Wesse, and M. H. Kuhn. Landmark-based elastic registration using approximating thin-plate splines. *IEEE Transactions on Medical Imaging*, 20(6):526–534, June 2001.

[20] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. *SIGGRAPH*, pages 151–160, August 1986.

[21] D. Terzopoulos and H. Qin. Dynamic NURBS with geometric contraints for interactive sculpting. *ACM transactions on Graphics*, pages 103–136, 1994.

[22] F. Trochu. Krigeage en CAO et FAO. Course notes for MEC6310, École Polytechnique, Montreal, Sept 2001.

[23] G. Wahba. *Spline Models for Observational Data.* CBMS-NFS Regional Conference Series in Applied Mathematics. SIAM, Philadelphia, PA, 1990.

[24] Y. Wang, W. Guo, and M. B.Brown. Spline smoothing for bivariate data with applications to association between hormones. *Statistica sinica*, 10(2):377–399, April 2000.

[25] A. Watt and M. Watt. *Advanced Rendering and Animation Techniques: Theory and Practice.* Addison-Wesley, 1991.