UNIVERSITY OF LIMOGES
DEPARTMENT OF SCIENCE AND TECHNOLOGY

3D Free Form Method and Applications to Robotics

Thesis

Master of Mathematics ACSYON

Supervisors: **Ouiddad Labbani-Igbida**
Olivier Ruatta

Presented by: **PHAN Tran Duc Minh**

Limoges, 2014.

Acknowledgments

I would like to express my deep gratitude to Professor Ouiddad Labbani-Igbida and Professor Olivier Ruatta, my internship supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work.

I would like to extend my thanks to the staff of the XLIM Research Institute for their assistance during the internship.

I would also like to thank the Director of the Department of Science and Technology of the University of Limoges, who enabled me to complete this thesis.

Finally, I am much indebted to my family and my friends for their support and heartily encouragement throughout my work.

Limoges, June 2014.

The author.

Abstract

This thesis is a detailed study on the geometry and the deformation of the shapes represented by uniform piecewise Bézier surfaces and applications to the shape optimization arising in image segmentation. Evolving deformable surfaces constructed by Bézier surfaces are moved from initially defined surfaces to attracted features which define the boundaries between different regions of a given three dimensional image. This evolution is characterized by local deformation. The different regions of the image are then detected and represented.

This thesis also contributes to robotics, typically, robot path planning in three-dimensional, shortly 3D, settings, where planning the movement of a mechanical arm avoiding obstacles is mentioned as a preliminary step. We propose an approach based on 3D free form model and deformation, called 3D Free Form method. In this method, we develop the geometric algorithms in order to manipulate Bézier free form models. This Bézier local structures allows the algorithms convergence with almost linear complexity and adapt to several complex boundaries of free space regions. The first experiments of our method are done with 3D simplified toy images with the objective of approximating the shapes of what could be considered free space regions.

Thematic: *Rational surfaces, 3D images and active surfaces.*

Table of Contents

Chapter 1

Introduction

Since the early 1960s, the French engineer Pierre Bézier used Bézier surfaces to design automobile bodies, [2]. Nowadays, Bézier surfaces are popularized in computer graphics, computer-aided design and finite element modeling. The property to be polynomially parametrized allows us to use Bézier surfaces to represent smooth surfaces. Moreover, this representation is geometrically intuitive and leads to numerically robust algorithms. Thus, geometric shapes that represented by uniform piecewise Bézier surfaces inherit meaningful properties. Significantly, Bézier modeling also permits to represent evolving deformable surfaces. This contributes to solve the shape optimization problem arising in image segmentation and leads to important applications to robotics.

In order to illustrate the potential applications in robotics, we describe an important problem for which the present work is an interesting contribution. An autonomous robot must be able to determine its free space, it is to say the region where it can freely move. For instance, in a factory, if a robot needs to catch something with an arm, it needs to determine a possible movement of the arm avoiding collision. This requires not only delicate techniques, simple algorithms but also fast processes. Free space perception of an unknown environment is still a big issue challenging mobile robotics. There are several significant efforts supporting to free space perception. For instance, using telemetric sensors such as sonars or laser allows to detect the depth of objects and to avoid accident while many approaches by vision permit to recover this depth estimation. One of such impressing contributions is two-dimensional Free form method. This method is introduced in [12]. One used the method in combination with processing 2D images of monocular omnidirectional vision. This method typically based on active contours' deformation for image segmentation in two dimensions. Figure 1.1 gives an illustration for this method. By using the topology change and the structural topology test, the 2D free form method is effective to adapt to numerous complex boundaries of free space regions, furthermore, to detect obstacles, see in Figure 1.2.

In three dimensions, the problem of free space approximations in unknown environments is attracting not only on the expansion of number of dimensions but also on the difficulties of topological structure of objects and regions in the space. We propose a 3D approach to this problem, called 3D Free form method. This 3D version of free form method is also based on free form models and deformation. The goal is to move an active surface from an initial center to detect free regions in three-dimensional unknown space. The method needs significant supports from 3D image segmentation to generate a pressure force orienting the evolution of the active surfaces. Deformation of free form active surfaces is typically characterized by their local representation and it is forced to minimize a functional energy. There are many formulation of energy functional, [4], [5], [1], but their common main desire is to ensure the convergence of the active surfaces to complex boundaries of free regions.

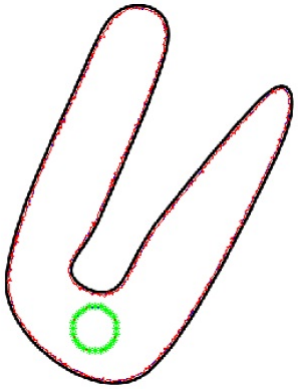


Figure 1.1: 2D Free Form method, [11]. The black curve defines the boundary of a free space regions. The green circle is a initially defined curve. The red illustrates the deformed curve from the green.

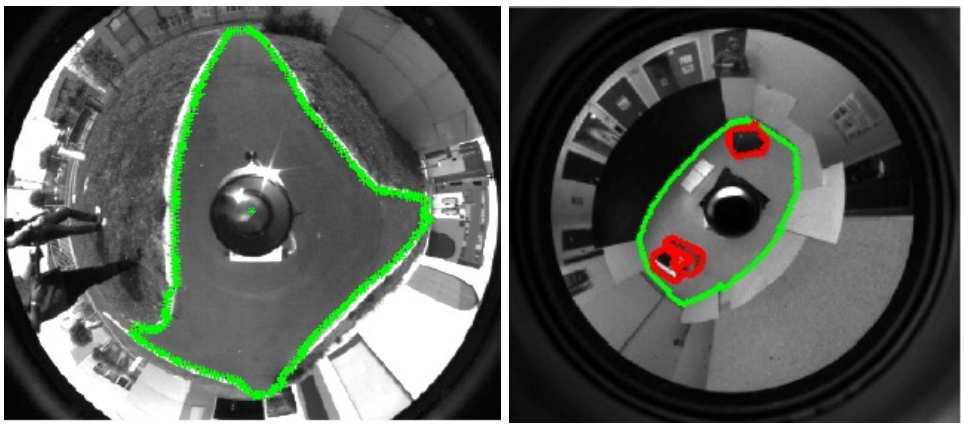


Figure 1.2: The 2D Free Form method adapts to a complex boundary region (the left, [11]) and detects obstacles (the right, [12]).

In 3D free form method, our work is to define a free form model of active surfaces so that deformation is smooth and the convergence adapt to many complex boundaries of the space regions. The active surfaces are constructed by Bézier patches. The free form active surfaces inherit from this local structures the parametric representation and the flexible and smooth deformation. The algorithms of our method require linear computational cost. They are fast processes. The contribution of this study particularly includes using the Free form based on active surfaces for 3D free space segmentation. This is applied to 3D free space perception and autonomous safe navigation of a robot.

This thesis contains six chapters. The first one is the introduction. Chapter 2 provides the mathematical backgrounds of free forms and active surfaces. In this chapter, we recall the basic definitions of Bézier curves and Bézier surfaces. After that, we study on the geometry and the deformation of shapes represented by uniform piecewise Bézier surfaces. This leads to introduce the approach of the 3D free form method. Its content is related to Bézier surface evaluation, interpolation, sampling, deformation and subdivision. Chapter 3 is the application of

the knowledge of Bézier surfaces in Chapter 2 for solving a class of geometric shape optimization problems. Such geometric shape optimization problems arise in image segmentation. Chapter 4 introduces 3D Canny edge detector. This detector is then used to generate a gradient edge map of a 3D image which is needed for 3D free form method. In Chapter 5, we will sketch the formulation of the shape optimization problem coming from image segmentation and the algorithm of 3D free form method. Manny results of experiments validating 3D free form method are presented here. Finally, Chapter 6 is to conclude the study and draws future works.

Chapter 2

Geometry and Deformation of Uniform Piecewise Bézier Surfaces

This chapter provides material and basic tools for 3D free form modeling and deformation. It contains evaluation, interpolation, sampling, subdivision and deformation of Bézier surfaces and uniform piecewise Bézier surfaces. This is mentioned as the mathematical background in order to formulate the 3D free form method. In this method, an active surface will be constructed with many linked patches, each of them is described by a continuous Bézier surface of bi-degree (D, D), called a 3D free form. Hence, the evolution of active surfaces will intrinsically inherit geometric characteristics of Bézier surfaces.

In this chapter, we work on three-dimensional real space. We denote that $E = \mathbb{R}^3$.

2.1 Bézier surfaces

In this section, we will present the definition and interesting geometric properties of Bézier surfaces. Typically, every polynomially parametrized surface can be represented by a Bézier surface. The recursive equations of Bézier surfaces generate a robust way to evaluation them by using the De Casteljau algorithm. Furthermore, Bézier interpolation allows to recover the Bézier surface passing through given points.

Bézier surface is a generalized concept of Bézier curve. It is necessary to recall some basic related definitions of Bézier curves. This will make the approach on three dimension become easier. A Bézier curve is determined by its order, D, and a set of $(D + 1)$ control points, $P_0, P_1, ..., P_D \in \mathbb{R}^2$, which is called the control polygon of the curve. The definition of Bézier curves bases on the following recursive equations, for $t \in [0, 1]$,

$$\begin{cases} B(P_0; t) = P_0, \\ B(P_0, P_1, ..., P_D; t) = (1 - t)B(P_0, P_1, ..., P_{D-1}; t) + tB(P_1, P_2, ..., P_D; t). \end{cases} \tag{2.1}$$

More clearly that a Bézier curve is a set of points which is defined by $\{B(P_0, P_1, ..., P_D; t) | t \in [0, 1]\}$. We remark that coordinates of $B(P_0, P_1, ..., P_D; t)$ are polynomials of degree at most D. So by this way, every set of points in the plane $\mathbb{R}^2$ will generate a polynomially parametrized curve whose degree is bounded, [11]. Trivially, for all point $P_0 \in \mathbb{R}^2$, the polynomially parametrized curve $B(P_0; t) = P_0$ is a Bézier curve of degree 0. For another example, we consider given points in $\mathbb{R}^2$,

$$P_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad P_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad P_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad P_3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$

By using Equations (2.1), we obtain

$$\begin{aligned}
B(P_0, P_1, P_2, P_3; t) =& (1-t)B(P_0, P_1, P_2; t) + tB(P_1, P_2, P_3; t) \\
=& (1-t)[(1-t)B(P_0, P_1; t) + tB(P_1, P_2)] \\
&+ t[(1-t)B(P_1, P_2; t) + tB(P_2, P_3)]. \\
=& (1-t)^2[(1-t)B(P_0; t) + tB(P_1; t)] \\
&+ (1-t)t[(1-t)B(P_1; t) + tB(P_2; t)] \\
&+ t(1-t)[(1-t)B(P_1; t) + tB(P_2; t)] \\
&+ t^2[(1-t)B(P_2; t) + tB(P_3; t)] \\
=& (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t)t^2 P_2 + t^3 P_3.
\end{aligned}$$

It shows that $B(P_0, P_1, P_2, P_3; t)$ defines a parametrized curve, whose coordinates are polynomials of degree 3,

$$B(P_0, P_1, P_2, P_3; t) = \begin{bmatrix} 3(1-t)^2 t + 3(1-t)t^2 - t^3 \\ (1-t)^3 + 3(1-t)^2 t - t^3 \end{bmatrix}.$$

This curve is a Bézier curve of degree 3. We call such a curve a cubic Bézier curve. Figure 2.1 gives us an geometric illustration of this cubic curve.

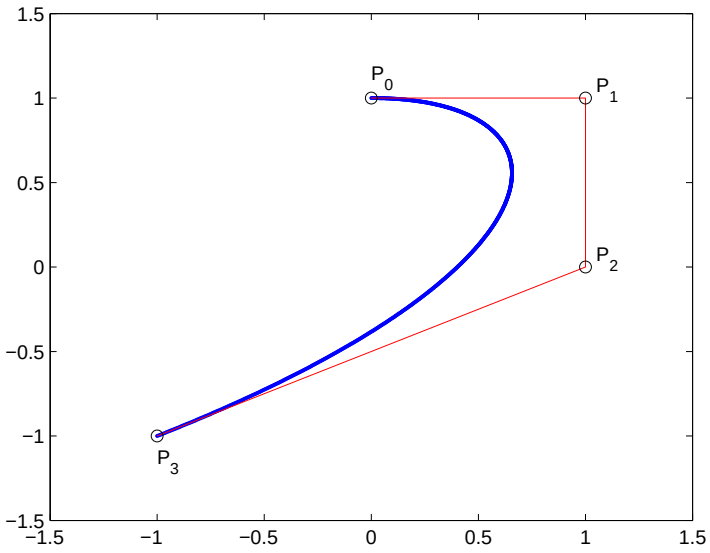


Figure 2.1: A Bézier curve. P_i are the control points. The red is the control polygon. The blue are the points on the curve.

As we have seen in the figure, the first and last control points are always the end points of the curve. However, the intermediate control points (if any) generally do not lie on the curve. Beside that, it is also notable that the curve points lie inside the convex hull of the control points.

Equations (2.1) means that a Bézier curve of degree D is a linear interpolation between two Bézier curves of degree $(D-1)$. This recursive definition leads to a way to evaluate Bézier curves by using the De Casteljau algorithm, [12]. It is a robust and numerically stable algorithm. This algorithm is shown in Algorithm 1.

Similarly, a Bézier surface is defined by a set of control points in $\mathbb{R}^3$,

$$\begin{array}{ccccc} P_{00} & P_{01} & P_{02} & ... & P_{0n} \\ P_{10} & P_{11} & P_{12} & ... & P_{1n} \\ P_{20} & P_{21} & P_{22} & ... & P_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{m0} & P_{m1} & P_{m2} & ... & P_{mn}. \end{array}$$

These points form a so-called control polytope. The Bézier surface is described by Equations (2.2). This is a polynomial parametrization in two variables $(s,t) \in [0,1] \times [0,1]$,

$$\begin{cases} B_2(P_{00}; s, t) = P_{00}, \\ B_2(P_{00}, ..., P_{mn}; s, t) = & (1-s)B(B(P_{00}, ..., P_{0n}; t), ..., B(P_{m-1,0}, ..., P_{m-1,n}; t); s) \\ & +sB(B(P_{10}, ..., P_{1n}; t), ..., B(P_{m0}, ..., P_{mn}; t); s), \end{cases} \tag{2.2}$$

where $B(.;.)$ is given by Equations (2.1).

Recursive equations (2.2) allow us to evaluate a Bézier surface by using the De Casteljau algorithm. This is done by Algorithm 2.

The Bézier surface generated by $(m+1)(n+1)$ control points by using Equations (2.2) is a surface which is parametrized with polynomials in s and t of degree $(m+n)$. Bézier surfaces can be of any degree, but in our approach to the geometry and the deformation of shapes, we will focus on Bézier surfaces of multi-degree (D, D).

Algorithm 1 De Casteljau algorithm for Bézier curves: $Eval(P;t)$, [12].

Input: $[P_0, ..., P_n]$ the list of the points of control polygon of the Bézier curve $B([P_0, ..., P_n], t)$ and $t \in [0,1]$.
Output: Coordinates of a curve point in $\mathbb{R}^2$

```
if n = 0 then
  return P_0
else
  return (1 − t) ∗ Eval([P_0, ..., P_{n−1}], t) + t ∗ Eval([P_1, ..., P_n], t)
end if
```

Algorithm 2 De Casteljau algorithm for Bézier surfaces: $Eval2(P;s,t)$.

Input: $\begin{bmatrix} P_{00} & ... & P_{0n} \\ \vdots & \ddots & \vdots \\ P_{m0} & ... & P_{mn} \end{bmatrix}$ the matrix of the points of control polytope of the Bézier surface $B([P_{00}, ..., P_{mn}]; s, t)$ and $(s,t) \in [0,1] \times [0,1]$;
Output: Coordinates of a surface point in $\mathbb{R}^3$

```
if m = 0 then
  return Eval([P_00, ..., P_0n], t)
else
  return
    (1−s)∗Eval2( [ P_00 ... P_0n ; ⋮ ⋱ ⋮ ; P_{m−1,0} ... P_{m−1,n} ] ; s, t ) + s∗Eval2( [ P_10 ... P_1n ; ⋮ ⋱ ⋮ ; P_m0 ... P_mn ] ; s, t )
end if
```

2.1.1 Bézier surfaces of bi-degree (D, D)

A (D, D) Bézier surface is defined by a set of $(D+1)(D+1)$ control points. It can be seen as a mapping which maps the unit square $[0,1] \times [0,1]$ into a smooth-continuous surface embedded within the space E.

Definition 1. *Given a $(D+1) \times (D+1)$ matrix of points in E,*

$$\begin{array}{ccccc} P_{00} & P_{01} & P_{02} & ... & P_{0D} \\ P_{10} & P_{11} & P_{12} & ... & P_{1D} \\ P_{20} & P_{21} & P_{22} & ... & P_{2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{D0} & P_{D1} & P_{D2} & ... & P_{DD}, \end{array} \tag{2.3}$$

the set $\{B_2(P_{00}, ..., P_{DD}; s, t) | (s,t) \in [0,1] \times [0,1]\}$ is called a Bézier surface of bi-degree (D, D), where $B_2(.; s, t)$ is defined by Equations (2.2). The list of points $[P_{00}, ..., P_{DD}]$ forms the control polytope of the Bézier surface and the points $P_{00}, ..., P_{DD}$ are control points.

It is remarkable that each set of $(D+1) \times (D+1)$ points generates a parametrized surface. Especially, this surface is a polynomially parametrized surface with bounded degree.

Proposition 1. *Let $P_{00}, ..., P_{DD} \in E$, then $B_2(P_{00}, ..., P_{DD}; s, t)$ is a polynomial parametrization. Moreover, its coordinates have degree at most $2D$ and of degree D with respect to both s and t.*

Proof. It is a fact that $B(P_0, ..., P_d; .)$ is a polynomial parametrization of degree at most d. Hence, by definition,

$$\begin{aligned} B_2(P_{00}, ..., P_{DD}; s, t) =& (1-s)B(B(P_{00}, ..., P_{0D}; t), ..., B(P_{D-1,0}, ..., P_{D-1,D}; t); s) \\ &+ sB(B(P_{10}, ..., P_{1D}; t), ..., B(P_{D0}, ..., P_{DD}; t); s), \end{aligned}$$

$B_2(P_{00}, ..., P_{DD}; s, t)$ is a polynomial parametrization in s, t, of degree at most $2D$. ■

For example, consider the Bézier surface generated by a given set of control points in E as following,

$$P_{00} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \quad P_{01} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \quad P_{02} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$P_{10} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad P_{11} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad P_{12} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$P_{20} = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \quad P_{21} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \quad P_{22} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

By definition, we use Equations (2.2) to obtain

$$\begin{aligned} B_2(P_{00}, ..., P_{22}; s, t) =& (1-s)B(B(P_{00}, P_{01}, P_{02}; t), B(P_{10}, P_{11}, P_{12}; t); s) \\ &+ sB(B(P_{10}, P_{11}, P_{12}; t), B(P_{20}, P_{21}, P_{22}; t); s). \end{aligned}$$

Then, we have

$$\begin{aligned}
B_2(P_{00},...,P_{22};s,t) =&(1-s)^2B(P_{00},P_{01},P_{02};t)+(1-s)sB(P_{10},P_{11},P_{12};t)\\
&+s(1-s)B(P_{10},P_{11},P_{12};t)+s^2B(P_{20},P_{21},P_{22};t)\\
=&(1-s)^2[(1-t)B(P_{00},P_{01};t)+tB(P_{01},P_{02};t)]\\
&+2(1-s)s[(1-t)B(P_{10},P_{11};t)+tB(P_{11},P_{12};t)]\\
&+s^2[(1-t)B(P_{20},P_{21};t)+tB(P_{21},P_{22};t)]\\
=&(1-s)^2[(1-t)^2P_{00}+2(1-t)tP_{01}+t^2P_{02}]\\
&+2(1-s)s[(1-t)^2P_{10}+2(1-t)tP_{11}+t^2P_{12}]\\
&+s^2[(1-t)^2P_{20}+2(1-t)tP_{21}+t^2P_{22}].
\end{aligned}$$

Then, the surface is parametrized by polynomials in s and t, of degree 4, i.e, of bi-degree $(2,2)$,

$$B_2(P_{00},...,P_{22};s,t) =$$
$$\begin{bmatrix}
-(1-s)^2(1-t)^2+2(1-s)s[2(1-t)t+t^2]+s^2[-(1-t)^2+2(1-t)t+t^2]\\
(1-s)^2[2(1-t)t+t^2]+2(1-s)s[2(1-t)t+t^2]-s^2(1-t)^2\\
(1-s)^2[(1-t)^2+2(1-t)t]+2(1-s)s[(1-t)^2+2(1-t)t]+s^2[(1-t)^2+2(1-t)t]
\end{bmatrix}.$$

We call such a Bézier surface a biquadratic Bézier surface. In practice, bicubic Bézier surfaces are used more popularly. These surfaces is generated by 4×4 control points. Figure 2.2 illustrates a Bézier surface of this type.

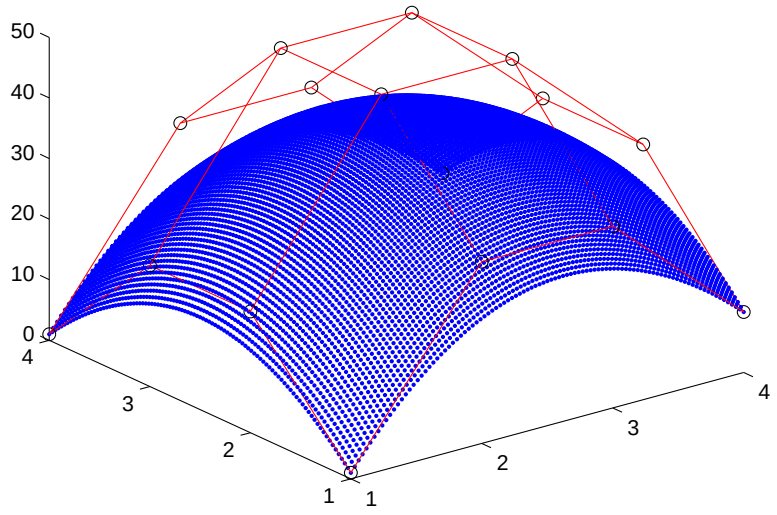


Figure 2.2: A bicubic Bézier surface. The circles are the control points. The red is the control polytope. The blue are the points on the surface.

2.1.2 Bernstein's polynomials

In this section, we will show how a (D,D) Bézier surface is expressed in terms of Bernstein's polynomials. These polynomials are advantageous not only to algebraically represent Bézier surfaces but also to evaluate them.

Definition 2. *Let D be an nonnegative integer and $i \in \{0, ..., D\}$, the polynomial*

$$b_{iD}(s) := \binom{i}{D}(1-s)^{D-i}s^i$$

is called Bernstein polynomial.

We denote $\mathbb{R}[s]_D$ the set of polynomials of degree less or equal to D. This is a real vector space of dimension $(D+1)$ and its natural basis is $\{1, s, s^2, ..., s^D\}$. As we have known that the set of Bernstein polynomials $\{b_{iD}(s)|i \in \{0, ..., D\}\}$ is linearly independent, thus, they also form a basis of $\mathbb{R}[s]_D$, [11].

Lemma 1. *$\{b_{iD}(s)|i \in \{0, ..., D\}\}$ form a basis of $\mathbb{R}[s]_D$.*

Notation 1. *We denote $\mathbb{R}[s,t]_{(D,D)}$ the $\mathbb{R}$-vector space generated by the linearly independent set $\{s^i t^j | i, j \in \{0, ..., D\}\}$.*

The vector space $\mathbb{R}[s,t]_{(D,D)}$ has dimension $(D+1)^2$. We will show that the polynomials $b_{iD}(s)b_{jD}(t)$'s form a basis of the space $\mathbb{R}[s,t]_{(D,D)}$.

Proposition 2. *The set $\{b_{iD}(s)b_{jD}(t)\}_{i,j\in\{0,...D\}}$ is a basis of $\mathbb{R}[s,t]_{(D,D)}$.*

Proof. It is the fact that $\mathbb{R}[s] \otimes \mathbb{R}[t] \cong \mathbb{R}[s,t]$. This isomorphism is given by $p(s) \otimes q(t) \longmapsto p(s)q(t)$. Since $b_{iD}(s)$'s is a basis of $\mathbb{R}[s]_D$ and $b_{jD}(t)$'s is a basis of $\mathbb{R}[t]_D$, the set $\{b_{iD}(s) \otimes b_{jD}(t)\}$ forms a basis of $\mathbb{R}[s]_D \otimes \mathbb{R}[t]_D$. Then through the isomorphism, $\{b_{iD}(s)b_{jD}(t)\}$ is a basis of $\mathbb{R}[s,t]_{(D,D)}$. ■

The Bernstein polynomials form a basis of the vector space $\mathbb{R}[s,t]_{(D,D)}$. And the recursive definition given by Equations (2.2) can be expressed in terms of Bernstein polynomials, as in Proposition 3. Till now, these are to say that every polynomially parametrized surface can be represented as a Bézier surface, said in Corollary 1.

Proposition 3. *Let $P_{00}, ..., P_{DD} \in E$, then*

$$B_2(P_{00}, ..., P_{DD}; s, t) = \sum_{i=0}^{D} \left[\sum_{j=0}^{D} P_{ij} b_{jD}(t) \right] b_{iD}(s) \tag{2.4}$$

for all $(s,t) \in [0,1] \times [0,1]$.

Proof. By definition in Equations (2.2),

$$\begin{aligned}
B_2(P_{00}, ..., P_{DD}; s, t) =& (1-s)B(B(P_{00}, ..., P_{0D}; t), ..., B(P_{D-1,0}, ..., P_{D-1,D}; t); s) \\
&+ sB(B(P_{10}, ..., P_{1D}; t), ..., B(P_{D0}, ..., P_{DD}; t); s) \\
=& (1-s) \sum_{i=0}^{D-1} B(P_{i0}, ..., P_{iD}; t) b_{i,D-1}(s) \\
&+ s \sum_{i=1}^{D} B(P_{i0}, ..., P_{iD}; t) b_{i-1,D-1}(s) \\
=& (1-s)^D B(P_{00}, ..., P_{0D}; t) + s^D B(P_{D0}, ..., P_{DD}; t) \\
&+ \sum_{i=1}^{D-1} B(P_{i0}, ..., P_{iD}; t) \left[(1-s)b_{i,D-1}(s) + sb_{i-1,D-1}(s) \right].
\end{aligned}$$

Since $(1-s)b_{i,D-1}(s) + sb_{i-1,D-1}(s) = b_{iD}(s)$, we obtain that

$$\begin{aligned} B_2(P_{00},...,P_{DD};s,t) &= \sum_{i=0}^{D} B(P_{i0},...,P_{iD};t)b_{iD}(s) \\ &= \sum_{i=0}^{D} \left[\sum_{j=0}^{D} P_{ij}b_{jD}(t) \right] b_{iD}(s). \end{aligned}$$

■

Corollary 1. *Every polynomially parametrized surface can be represented as a Bézier surface.*

The equation (2.4) can be rewritten as below,

$$B_2(P_{00},...,P_{DD};s,t) = \sum_{ij} b_{iD}(s)b_{jD}(t)P_{ij}.$$

This means that each point of Bézier surface, $B_2(P_{00},...,P_{DD};s,t)$, is a weighted average of the control points, P_{ij}. Hence, a Bézier surface will lie completely within the convex hull of its control points. Note that the corner points of the surface are the four corner ones of the control polytope. And generally, the surface does not pass through other control points. Moreover, the equation (2.4) also says that the curves $B_2(P_{00},...,P_{DD};s_0,t)$ and $B_2(P_{00},...,P_{DD};s,t_0)$ lying on the surface are Bézier curves. Typically, the Bézier curves $B_2(P_{00},...,P_{DD};0,t)$, $B_2(P_{00},...,P_{DD};1,t)$, $B_2(P_{00},...,P_{DD};s,0)$ and $B_2(P_{00},...,P_{DD};s,1)$ are the boundary of the surface. We will call them boundary curves.

2.1.3 Interpolation

We have already known that we can take sampled points of a Bézier surface using the De Casteljau algorithm. Conversely, a question is proposed that with some given points, whether there exists a Bézier surface passing through them. Since a Bézier surface of bi-degree (D,D) is defined by $(D+1)^2$ control points, we can hope to recover $(D+1)^2$ control points of a surface with a given sampling of $(D+1)^2$ points on the surface. This section will show that it is possible, said in Proposition 4. In fact, there are numerous satisfied Bézier surfaces. Each of them corresponds to a specific subdivision of the unit square $[0,1]\times[0,1]$, as mentioned in Lemma 2. This will be displayed by Figure 2.3.

Proposition 4. *Let $M_{00}, M_{01}, ..., M_{DD} \in E$, then there exists Bézier surface of bi-degree (D,D) passing through these points.*

Lemma 2. *Let $0 = s_0 < s_1 < ... < s_D = 1$ and $0 = t_0 < t_1 < ... < t_D = 1$, then there exists one and only one Bézier surface $B_2(P_{00},...,P_{DD};s,t)$ of bi-degree (D,D) such that $B_2(P_{00},...,P_{DD};s_i,t_j) = M_{ij}$ for all $i,j \in \{0,...,D\}$.*

Proof. We denote by M the $(D+1)\times(D+1)$ matrix of points M_{ij},

$$M = \begin{bmatrix} M_{00} & M_{01} & M_{02} & ... & M_{0D} \\ M_{10} & M_{11} & M_{12} & ... & M_{1D} \\ M_{20} & M_{21} & M_{22} & ... & M_{2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ M_{D0} & M_{D1} & M_{D2} & ... & M_{DD} \end{bmatrix},$$

and by P the $(D+1)\times(D+1)$ matrix of points P_{ij},

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & ... & P_{0D} \\ P_{10} & P_{11} & P_{12} & ... & P_{1D} \\ P_{20} & P_{21} & P_{22} & ... & P_{2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{D0} & P_{D1} & P_{D2} & ... & P_{DD} \end{bmatrix}. \tag{2.5}$$

Now, we consider the following matrices associated with $\boldsymbol{s} = (s_0, ..., s_D)$ and $\boldsymbol{t} = (t_0, ..., t_D)$,

$$B_{sD} = \begin{bmatrix} b_{0D}(0) & b_{1D}(0) & b_{2D}(0) & ... & b_{DD}(0) \\ b_{0D}(s_1) & b_{1D}(s_1) & b_{2D}(s_1) & ... & b_{DD}(s_1) \\ b_{0D}(s_2) & b_{1D}(s_2) & b_{2D}(s_2) & ... & b_{DD}(s_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{0D}(1) & b_{1D}(1) & b_{2D}(1) & ... & b_{DD}(1) \end{bmatrix}, \tag{2.6}$$

$$B_{tD} = \begin{bmatrix} b_{0D}(0) & b_{1D}(0) & b_{2D}(0) & ... & b_{DD}(0) \\ b_{0D}(t_1) & b_{1D}(t_1) & b_{2D}(t_1) & ... & b_{DD}(t_1) \\ b_{0D}(t_2) & b_{1D}(t_2) & b_{2D}(t_2) & ... & b_{DD}(t_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{0D}(1) & b_{1D}(1) & b_{2D}(1) & ... & b_{DD}(1) \end{bmatrix}. \tag{2.7}$$

As the matrices B_{sD}, B_{tD} are Vandermonde matrices expressed in the Bernstein basis, they are invertible, [11]. Hence, if P is such that $B_{sD}PB_{tD}^T = M$, then $B_2(P_{00}, ..., P_{DD}; s, t)$ gives the wanted surface. ■

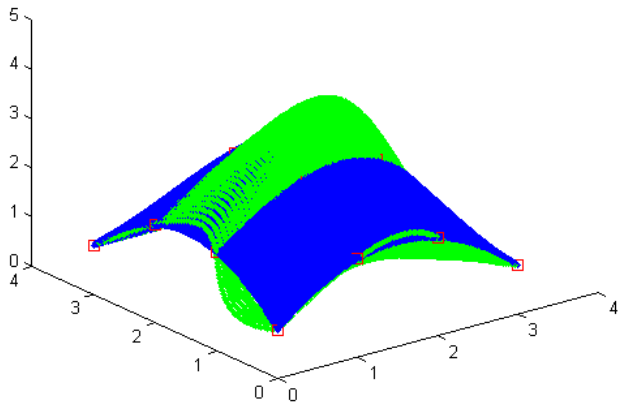


Figure 2.3: Two distinct bicubic Bézier surfaces pass through 4×4 points. The red squares are given points. The blue surface corresponds to the regular subdivision of the unit square, that $\boldsymbol{s} = [0, \frac{1}{3}, \frac{2}{3}, 1]$ and $\boldsymbol{t} = [0, \frac{1}{3}, \frac{2}{3}, 1]$. The green is given by $\boldsymbol{s} = [0, \frac{4}{8}, \frac{7}{8}, 1]$ and $\boldsymbol{t} = [0, \frac{1}{5}, \frac{3}{5}, 1]$

Remark that the pair of matrices B_{sD}^{-1} and B_{tD}^{-T} defines a linear map associating to $(D+1)^2$ distinct points a Bézier surface of bi-degree (D, D) going through those points.

Notation 2. *We denote θ_{st} the linear operator on $E^{(D+1)^2}$ defined by*

$$\begin{aligned}\theta_{st} : E^{(D+1)^2} &\longrightarrow E^{(D+1)^2}\\ P &\longmapsto B_{sD} P B_{tD}^T.\end{aligned}$$

From now, we will represent each element of the space $E^{(D+1)^2}$ by matrix form shown in Equation (2.8).

Proposition 5. *θ_{st} is an linear isomorphism.*

Proof. θ_{st} is linear because of the fact that,

$$B_{sD}(P+Q)B_{tD}^T = B_{sD}PB_{tD}^T + B_{sD}QB_{tD}^T, \forall P, Q \in E^{(D+1)^2}.$$

Moreover, since the matrices B_{sD} and B_{tD} are invertible, the inverse of θ_{st} is given by $E^{(D+1)^2} \ni M \longmapsto B_{sD}^{-1} M B_{tD}^{-T}$. ■

The map θ_{st}^{-1} gives the interpolation formula for Bézier points. It is used in Algorithm 3. Note that in implementation, if the quantities $\boldsymbol{s}$ and $\boldsymbol{t}$ are known and fixed, the matrices B_{sD}^{-1} and B_{tD}^{-T} will be evaluated only once in the initialization and used for all. There are several choices of the quantities $\boldsymbol{s}$ and $\boldsymbol{t}$, but here we focus on the regular subdivision of the unit square, i.e, $s_i = \frac{i}{D}, t_j = \frac{j}{D}$. When the quantities $\boldsymbol{s}$ and $\boldsymbol{t}$ are fixed in the initialization, the computational cost of the interpolation at bi-degree (D, D) is given by the following lemma.

Lemma 3. *Let $\boldsymbol{s} = [s_0, s_1, ..., s_D]$ and $\boldsymbol{t} = [t_0, t_1, ..., t_D]$ are fixed subdivisions of the interval $[0, 1]$, the cost of interpolation by a Bézier surface of bi-degree (D, D), named $\mathcal{I}(D)$, is the cost of multiplications of three $(D+1) \times (D+1)$ matrices in three times. This cost is in $\mathcal{O}(D^3)$ and is constant for a fixed D.*

Algorithm 3 Interpolation of (D, D) Bézier surfaces.

Input:

$$\begin{bmatrix} M_{00} & ... & M_{0D} \\ \vdots & \ddots & \vdots \\ M_{D0} & ... & M_{DD} \end{bmatrix}$$ the matrix of given points;

Subdivisions of the interval $[0, 1]$: $\boldsymbol{s} = [0, s_1, ..., s_D]$ and $\boldsymbol{t} = [0, t_1, ..., t_D]$;

Output: The Bézier surface passing through these points;

return

$$B_{sD}^{-1} \begin{bmatrix} M_{00} & ... & M_{0D} \\ \vdots & \ddots & \vdots \\ M_{D0} & ... & M_{DD} \end{bmatrix} B_{tD}^{-T};$$

2.2 Uniform piecewise Bézier surfaces

In this section, we will describe uniform piecewise Bézier surfaces and study on their geometric properties including sampling, interpolation, subdivision and deformation. Thank to Bézier parametrization, the set of all piecewise Bézier surfaces is a good approximating for the space of continuous functions $\mathcal{C}^0([0, 1]^2, E)$. Moreover, this set is a manifold, indeed a finite dimensional space. In addition, the geometric properties of uniform piecewise Bézier surfaces are naturally

induced by the ones of local Bézier structures. 3D free form method then benefit these ideal characteristics to become an efficient method.

An active surface will be described as a uniform piecewise Bézier surface. It is a set of $(M+1)(N+1)$ linked Bézier surfaces of the same bi-degree (D, D) such that they form a regular surface. The regularity here means that if any two patches of a uniform piecewise Bézier surface have intersection, then the common one is either the end point or the entire boundary curve of theirs. Figures 2.5 shows a regular surface and Figure 2.6 gives an example of nonregular one. From now, we only focus on regular uniform piecewise Bézier surfaces. The regularity is strictly required for all presentation and deformation of uniform piecewise Bézier surfaces.

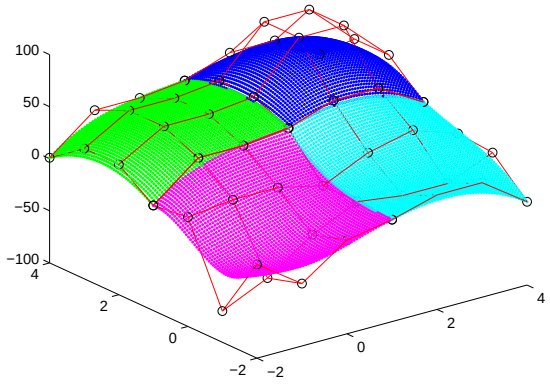


Figure 2.4: A uniform piecewise bicubic Bézier surface with 4 patches. Each color is a path of the uniform piecewise Bézier surface. The circles are the control points. The red is the control polytope.

2.2.1 Definitions

We now construct a uniform piecewise Bézier surface by gluing Bézier surfaces together such that the regularity is satisfied.

Given a matrix of points in E,

$$P = \begin{bmatrix} P_{00}^{00} & \dots & P_{0D}^{00} & P_{00}^{01} & \dots & P_{0D}^{01} & \dots & \dots & \dots & P_{00}^{0N} & \dots & P_{0D}^{0N} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ P_{D0}^{00} & \dots & P_{DD}^{00} & P_{D0}^{01} & \dots & P_{DD}^{01} & \dots & \dots & \dots & P_{D0}^{0N} & \dots & P_{DD}^{0N} \\ P_{00}^{10} & \dots & P_{0D}^{10} & P_{00}^{11} & \dots & P_{0D}^{11} & \dots & \dots & \dots & P_{00}^{1N} & \dots & P_{0D}^{1N} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ P_{D0}^{10} & \dots & P_{DD}^{10} & P_{D0}^{11} & \dots & P_{DD}^{11} & \dots & \dots & \dots & P_{D0}^{1N} & \dots & P_{DD}^{1N} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ P_{00}^{M0} & \dots & P_{0D}^{M0} & P_{00}^{M1} & \dots & P_{0D}^{M1} & \dots & \dots & \dots & P_{00}^{MN} & \dots & P_{0D}^{MN} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ P_{D0}^{M0} & \dots & P_{DD}^{M0} & P_{D0}^{M1} & \dots & P_{DD}^{M1} & \dots & \dots & \dots & P_{D0}^{MN} & \dots & P_{DD}^{MN} \end{bmatrix} \tag{2.8}$$

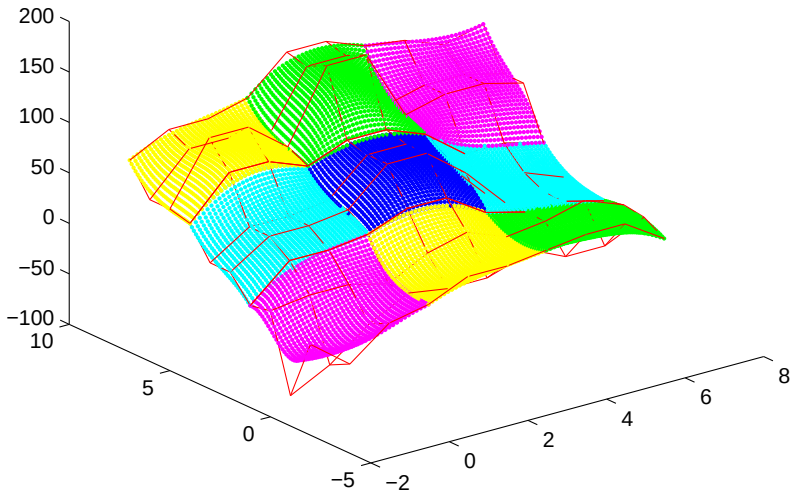


Figure 2.5: A regular uniform piecewise Bézier surface with 9 patches. Each path of the uniform piecewise Bézier surface is described by color. The red is the control polytope.

such that

$$\begin{cases} P_{0k}^{i+1,j} = P_{Dk}^{ij}, & \forall i \in \{0,...,M-1\}, k \in \{0,...,D\}, \\ P_{l0}^{i,j+1} = P_{lD}^{ij}, & \forall j \in \{0,...,N-1\}, l \in \{0,...,D\}. \end{cases} \tag{2.9}$$

We denote that, for all $i \in \{0,...,M\}$ and $j \in \{0,...,N\}$,

$$P^{ij} = \begin{bmatrix} P_{00}^{ij} & ... & P_{0D}^{ij} \\ \vdots & & \vdots \\ P_{D0}^{ij} & ... & P_{DD}^{ij} \end{bmatrix}.$$

Each block P^{ij} of P will describe a Bézier surface. The condition (2.9) is to keep the regularity of the uniform piecewise Bézier surface defined by P.

Let $0 = s_{00} < s_{0D} = s_{10} < s_{1D} = s_{20} < ... < s_{MD} = 1$ and $0 = t_{00} < t_{0D} = t_{10} < t_{1D} = t_{20} < ... < t_{ND} = 1$ be two subdivision of the interval $[0,1]$, and α be a transformation given by

$$\begin{cases} \alpha(s) = \frac{s-s_{i0}}{s_{iD}-s_{i0}}, & \forall s \in [s_{i0}, s_{iD}], \\ \\ \alpha(t) = \frac{t-t_{j0}}{t_{jD}-t_{j0}}, & \forall t \in [t_{j0}, t_{jD}], \end{cases} \tag{2.10}$$

we define

$$\Gamma(P^{00},...,P^{MN},s,t) = B_2(P^{ij};\alpha(s),\alpha(t)) \tag{2.11}$$

where $s \in [s_{i0}, s_{iD}]$ and $t \in [t_{j0}, t_{jD}]$ for all $i \in \{0,...,M\}$, $j \in \{0,...,N\}$.
$\Gamma(P,s,t)$ is a continuous parametrization. It is called a uniform piecewise Bézier surface. The surfaces parametrized by $B_2(P^{ij};\alpha(s),\alpha(t))$ are called the patches of $S = \Gamma(P,[0,1],[0,1])$. For

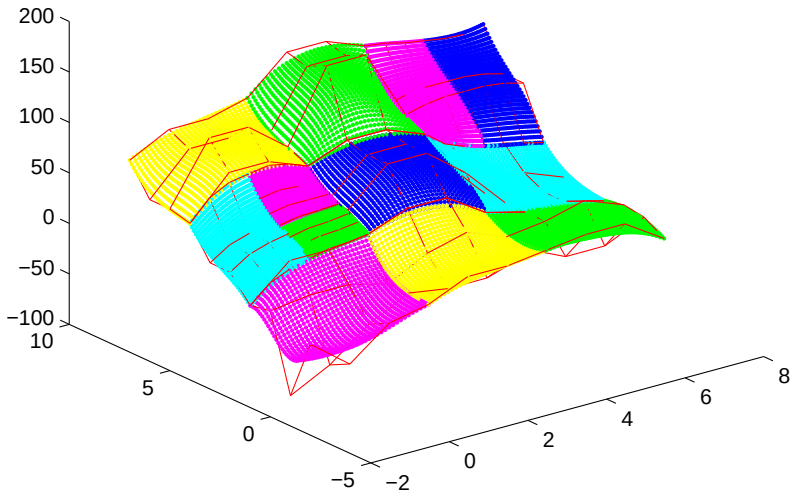


Figure 2.6: A non-regular uniform piecewise Bézier surface with 12 patches. Each path of the uniform piecewise Bézier surface is described by color. The red is the control polytope.

example, Figure 2.4 illustrates a uniform piecewise Bézier surface with 4 patches. Moreover, if it is satisfied that

$$\begin{aligned} P_{k0}^{i0} &= P_{kD}^{iN}, \quad \forall i \in \{0, ..., M\}, k \in \{0, ..., D\} \\ P_{0l}^{0j} &= P_{Dl}^{Mj}, \quad \forall j \in \{0, ..., N\}, l \in \{0, ..., D\}, \end{aligned}$$

then S is a closed surface, i.e, on topology, S separates the space E into two domains, interior and exterior. Figure 2.7 displays such a closed uniform piecewise Bézier surface.

Remark that there exist numerous choices of the subdivisions $(s_{00}, s_{0D} = s_{10}, s_{1D}, ..., s_{MD})$ and $(t_{00}, t_{0D} = t_{10}, t_{1D}, ..., t_{ND})$ of the interval $[0, 1]$. However, in practice, we use the subdivisions $(0, \frac{1}{M+1}, \frac{2}{M+1}, ..., 1)$ and $(0, \frac{1}{N+1}, \frac{2}{N+1}, ..., 1)$. They are the so-called regular subdivisions of [0,1]. Then, the transformation α becomes

$$\begin{cases} \alpha(s) = (M+1)s - i, & \forall s \in [\frac{i}{M+1}, \frac{i+1}{M+1}] \\ \alpha(t) = (N+1)t - j, & \forall t \in [\frac{j}{N+1}, \frac{j+1}{N+1}], \end{cases}$$

for all $i \in \{0, ..., M\}$ and $j \in \{0, ..., N\}$. Using them makes the implementation simpler.

Notation 3. *We denote $\mathcal{B}_{MN}^{D}$ the set of uniform piecewise Bézier surfaces built from $(M+1)(N+1)$ patches of bi-degree (D, D).*

This set, $\mathcal{B}_{MN}^{D}$, is the image of the following map,

$$\Psi_{MN} : \begin{cases} (E^{(D+1)^2})^{(M+1)(N+1)} & \longrightarrow \mathcal{C}^0([0,1] \times [0,1], E) \\ P & \longmapsto \quad \Gamma(P, s, t). \end{cases} \tag{2.12}$$

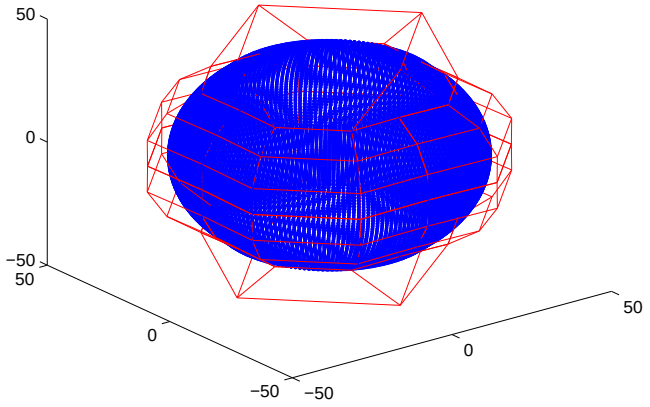


Figure 2.7: A closed uniform piecewise Bézier surface. The red is the control polytope. The blue are the points on the surface.

In particular, the set of all Bézier surfaces of bi-degree (D, D), $\mathcal{B}^D_{00}$ is the image of the map $\Psi_{00} : E^{(D+1)^2} \to \mathcal{C}^0([0,1]^2, E)$.

Notably, the set of polynomials in two variables s and t on the unit square $[0,1] \times [0,1]$ is dense in the space of continuous functions $\mathcal{C}^0([0,1] \times [0,1], \mathbb{R})$, see in [7]. This implies that for each continuous function $f : [0,1] \times [0,1] \to E$, there exists a sequence $\{\Gamma_n(s,t)\}_{n\in\mathbb{N}}$ such that $\lim_{n\to\infty} ||f - \Gamma_n||_2 = 0$. It is to say that piecewise Bézier surfaces are a good approximating set for the space $\mathcal{C}^0([0,1] \times [0,1], E)$.

In order to do calculus on the set $\mathcal{B}^D_{MN}$, as well as to deform the uniform piecewise Bézier surfaces, $\mathcal{B}^D_{MN}$ need equipped a smooth manifold structure. In fact, $\mathcal{B}^D_{MN}$ is a finite dimensional subspace of the normed linear space $\mathcal{C}^0([0.1]^2, E)$. This is the consequence of Proposition 9, where sampling map and the map Ψ_{MN} play central roles.

2.2.2 Sampling map and retraction to Ψ_{MN}

Each sampling of a surface given by a function $f \in \mathcal{C}^0([0,1] \times [0,1], E)$ corresponds to particular samples on the unit square.

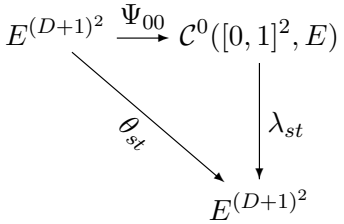
Definition 3. *Let $0 = s_0 < s_1 < ... < s_D = 1$ and $0 = t_0 < t_1 < ... < t_D = 1$, denote $\boldsymbol{s} = (s_0, ..., s_D)$ and $\boldsymbol{t} = (t_0, ..., t_D)$ the associated subdivisions of $[0,1] \times [0,1]$, we define the sampling map $\lambda_{st} : \mathcal{C}^0([0,1]^2, E) \to E^{(D+1)^2}$ by*

$$f \longmapsto \lambda_{st}(f) = \begin{bmatrix} f(s_0,t_0) & ... & f(s_0,t_D) \\ \vdots & \ddots & \vdots \\ f(s_D,t_0) & ... & f(s_D,t_D) \end{bmatrix}.$$

For each $(\boldsymbol{s}, \boldsymbol{t})$, the points $\lambda_{st}(f)$ is called a $(D+1) \times (D+1)$ sampling of the surface f.

The sampling map λ_{st} is linear. Once the quantities $\boldsymbol{s}$ and $\boldsymbol{t}$ are known, the map will assign to each surface $(D+1)^2$ sampled points.

Proposition 6. *The following diagram is commutative,*



(2.13)

Moreover, Ψ_{00} is an linear isomorphism between $E^{(D+1)^2}$ and $\mathcal{B}_{00}^D = Im(\Psi_{00})$, and its inverse is $\Psi_{00}^{-1} = \theta_{st}^{-1} \circ \lambda_{st}$.

Proof. Passing the map Ψ_{00}, each matrix P of points in E,

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & ... & P_{0D} \\ P_{10} & P_{11} & P_{12} & ... & P_{1D} \\ P_{20} & P_{21} & P_{22} & ... & P_{2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{D0} & P_{D1} & P_{D2} & ... & P_{DD} \end{bmatrix},$$

is assigned to a Bézier surface determined by $\Gamma(s,t) = \Psi_{00}(P)(s,t) = B_2(P;s,t)$. Since Γ is of the form $\Gamma(s,t) = \sum_{i=0}^{D} \left[\sum_{j=0}^{D} P_{ij} b_{jD}(t)\right] b_{iD}(s)$, the matrix form of the sampling map λ_{st} is given by

$$\lambda_{st}(\Gamma) = B_{sD} P B_{tD}^T = \theta_{st}(P).$$

This shows that $\lambda_{st} \circ \Psi_{0D} = \theta_{st}$.
Now, we consider the map $\Psi_{0D} : E^{(D+1)^2} \longrightarrow \mathcal{B}_{00}^D = Im(\Psi_{00})$. Obviously, Ψ_{00} is a linear monomorphism. Since the map θ_{st} is linearly isomorphic and its inverse defined by $\theta_{st}^{-1}(P) = B_{sD}^{-1} P B_{tD}^{-T}$, the map $\theta_{st}^{-1} \circ \lambda_{st}$ is the inverse of Ψ_{00}. Hence, Ψ_{00} is an linear isomorphism between $E^{(D+1)^2}$ and $\mathcal{B}_{00}^D$. ■

In fact, given a Bézier surface $\Gamma(s,t) \in \mathcal{B}_{00}^D$, the map λ_{st} gives a sampling of Γ, that is $\lambda_{st}(\Gamma)$. By Lemma 2, there exists one and only one Bézier surface, whose control polytope is $\theta_{st}^{-1}(\lambda_{st}(\Gamma))$, and whose sampled points are $\lambda_{st}(\Gamma)$. The mentioned surface is indeed $\Gamma(s,t)$.

The followings are to extend the sampling map λ_{st} to the case of uniform piecewise Bézier surfaces. Indeed, this is application of λ_{st} to each of patches.

Definition 4. *Let $0 = s_{00} < s_{01} < ... < s_{0D} = s_{10} < s_{11} < ... < s_{1D} = s_{20} < ... < s_{MD} = 1$ and $0 = t_{00} < t_{01} < ... < t_{0D} = t_{10} < t_{11} < ... < t_{1D} = t_{20} < ... < t_{ND} = 1$. Then, we define the sampling map*

$$\Lambda_{st} : \mathcal{C}^0([0,1]^2, E) \longrightarrow (E^{(D+1)^2})^{(M+1)(N+1)}$$

by

$$\Gamma \longmapsto \Lambda_{st}(\Gamma) = \begin{bmatrix} \Gamma(s_{00}, t_{00}) & ... & \Gamma(s_{00}, t_{ND}) \\ \vdots & \ddots & \vdots \\ \Gamma(s_{MD}, t_{00}) & ... & \Gamma(s_{MD}, t_{ND}) \end{bmatrix}.$$

Notation 4. *Denote that $\boldsymbol{s} = (s_0, ..., s_M)$ where $s_i = (s_{i0}, ..., s_{iD})$ and $\boldsymbol{t} = (t_0, ..., t_N)$ where $t_i = (t_{i0}, ..., t_{iD})$.*

Many choices of $\boldsymbol{s}$, $\boldsymbol{t}$ exist. In practice, we use ones that, for all $j \in \{0, ..., D\}$,

$$s_{i0} = \frac{i}{M+1}, \quad s_{iD} = \frac{i+1}{M+1}, \quad s_{ij} = t_{i0} + \frac{j}{D(M+1)}, \quad \forall i \in \{0, ..., M\},$$
$$t_{k0} = \frac{k}{N+1}, \quad t_{kD} = \frac{k+1}{N+1}, \quad t_{kj} = t_{k0} + \frac{j}{D(N+1)}, \quad \forall k \in \{0, .., N\}.$$

This is the so-called regular subdivision.

It is significant to consider the transformation α given by Formula (2.10). Depending on $\boldsymbol{s}$ and $\boldsymbol{t}$, α transforms the activities done on Γ into ones done in each of its patches. This implies that we can do directly what we want to do on the entire surafce Γ, on each of its patches.

Proposition 7. *Let Γ be a uniform piecewise Bézier surface with the control polytope P, then the sampling map Λ_{st} is determined by*

$$\Lambda_{st}(\Gamma) = (B_{\alpha(s_0)D} \times B_{\alpha(s_1)D} \times ... \times B_{\alpha(s_M)D}) P (B_{\alpha(t_0)D} \times B_{\alpha(t_1)D} \times ... \times B_{\alpha(t_N)D})^T,$$

where $B_{\alpha(s_0)D} \times B_{\alpha(s_1)D} \times ... \times B_{\alpha(s_M)D} = \operatorname{diag}(B_{\alpha(s_0)D}, B_{\alpha(s_1)D}, ..., B_{\alpha(s_M)D})$ and $B_{\alpha(t_0)D} \times B_{\alpha(t_1)D} \times ... \times B_{\alpha(t_N)D} = \operatorname{diag}(B_{\alpha(t_0)D}, B_{\alpha(t_1)D}, ..., B_{\alpha(t_N)D})$, as shown below,

$$B_{\alpha(s_0)D} \times B_{\alpha(s_1)D} \times ... \times B_{\alpha(s_M)D} = \begin{bmatrix} B_{\alpha(s_0)D} & & & 0 \\ & B_{\alpha(s_1)D} & & \\ & & \ddots & \\ 0 & & & B_{\alpha(s_M)D} \end{bmatrix}.$$

Proof. Given a uniform piecewise Bézier surface $\Gamma \in \mathcal{B}^D_{MN}$. Assume that the control polytope P of the surface Γ is of the form shown in (2.8). It can be rewritten shortly, as below,

$$P = \begin{bmatrix} P^{00} & ... & P^{0N} \\ \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{MN} \end{bmatrix}.$$

Then, the matrix form of the sampling map Λ_{st} is derived from the followings.

$$\begin{aligned}
&\operatorname{diag}(B_{\alpha(s_0)D}, ..., B_{\alpha(s_M)D}) P \operatorname{diag}(B^T_{\alpha(t_0)D}, ..., B^T_{\alpha(t_N)D}) \\
&= \begin{bmatrix} B_{\alpha(s_0)D} & & 0 \\ & \ddots & \\ 0 & & B_{\alpha(s_M)D} \end{bmatrix} \begin{bmatrix} P^{00} & ... & P^{0N} \\ \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{MN} \end{bmatrix} \begin{bmatrix} B^T_{\alpha(t_0)D} & & 0 \\ & \ddots & \\ 0 & & B^T_{\alpha(t_N)D} \end{bmatrix} \\
&= \begin{bmatrix} B_{\alpha(s_0)D} P^{00} B^T_{\alpha(t_0)D} & ... & B_{\alpha(s_0)D} P^{0N} B^T_{\alpha(t_N)D} \\ \vdots & \ddots & \vdots \\ B_{\alpha(s_M)D} P^{M0} B^T_{\alpha(t_0)D} & ... & B_{\alpha(s_M)D} P^{MN} B^T_{\alpha(t_N)D} \end{bmatrix} \\
&= \begin{bmatrix} \theta_{\alpha(s_0)\alpha(t_0)}(P^{00}) & ... & \theta_{\alpha(s_0)\alpha(t_N)}(P^{0N}) \\ \vdots & \ddots & \vdots \\ \theta_{\alpha(s_M)\alpha(t_0)}(P^{M0}) & ... & \theta_{\alpha(s_M)\alpha(t_N)}(P^{MN}) \end{bmatrix}
\end{aligned}$$

Note that $\theta_{st}(P^{ij}) = \lambda_{st}(\Psi_{00}(P^{ij}))$.

We obtain

$$\begin{aligned}
&\operatorname{diag}(B_{\alpha(s_0)D},...,B_{\alpha(s_M)D})P\operatorname{diag}(B^T_{\alpha(t_0)D},...,B^T_{\alpha(t_N)D}) \\
&\quad= \begin{bmatrix} \lambda_{\alpha(s_0)\alpha(t_0)}(\Psi_{00}(P^{00})) & ... & \lambda_{\alpha(s_0)\alpha(t_N)}(\Psi_{00}(P^{0N})) \\ \vdots & \ddots & \vdots \\ \lambda_{\alpha(s_M)\alpha(t_0)}(\Psi_{00}(P^{M0})) & ... & \lambda_{\alpha(s_M)\alpha(t_N)}(\Psi_{00}(P^{MN})) \end{bmatrix} \\
&\quad= \begin{bmatrix} \Psi_{00}(P^{00})(\alpha(s_0),\alpha(t_0)) & ... & \Psi_{00}(P^{0N})(\alpha(s_0),\alpha(t_N)) \\ \vdots & \ddots & \vdots \\ \Psi_{00}(P^{M0})(\alpha(s_M),\alpha(t_0)) & ... & \Psi_{00}(P^{MN})(\alpha(s_M),\alpha(t_N)) \end{bmatrix} \\
&\quad= \begin{bmatrix} \Gamma(s_0,t_0) & ... & \Gamma(s_0,t_N) \\ \vdots & \ddots & \vdots \\ \Gamma(s_M,t_0) & ... & \Gamma(s_M,t_N) \end{bmatrix} \\
&\quad= \Lambda_{st}(\Gamma).
\end{aligned}$$

This proof also shows that the sampling on the uniform piecewise Bézier surface reduced to samplings on its Bézier patches. ■

Notation 5. *We denote the extension of θ_{st} the map Θ_{st} defined by*

$$\Theta_{st}(P) = (B_{\alpha(s_0)D} \times B_{\alpha(s_1)D} \times ... \times B_{\alpha(s_M)D})P(B_{\alpha(t_0)D} \times B_{\alpha(t_1)D} \times ... \times B_{\alpha(t_N)D})^T,$$

where $P \in (E^{(D+1)^2})^{(M+1)(N+1)}$.

Then, Θ_{st} is an linear isomorphic operator on the space $(E^{(D+1)^2})^{(M+1)(N+1)}$ with the inverse given by

$$\Theta^{-1}_{st}(Q) = (B^{-1}_{\alpha(s_0)D} \times B^{-1}_{\alpha(s_1)D} \times ... \times B^{-1}_{\alpha(s_M)D})Q(B^{-T}_{\alpha(t_0)D} \times B^{-T}_{\alpha(t_1)D} \times ... \times B^{-T}_{\alpha(t_N)D}),$$

where $Q \in (E^{(D+1)^2})^{(M+1)(N+1)}$. Moreover, as assumption in Proposition 7, we obtain $\Lambda_{st}(\Gamma) = \Theta_{st}(P)$. This is to show that, once the quantities $\boldsymbol{s}$ and $\boldsymbol{t}$ are fixed, the associated sampling of the surface Γ can be taken directly by using the map Θ_{st}.

Return to the problem as in the case of Bézier surfaces mentioned in the section 2.1.3, that how to find a uniform piecewise Bézier surface passing through given points. This problem is stated in Problem 1 and solved by Proposition 8.

Problem 1. *Given a matrix of points $Q = (Q^{00}_{00},...,Q^{MN}_{DD})$ in E as shown in (2.14), find a uniform piecewise Bézier surface $\Gamma \in \mathcal{B}^D_{MN}$ such that $\Lambda_{st}(\Gamma) = Q$, for some $\boldsymbol{s},\boldsymbol{t}$ as assumption in Definition 4.*

Let us represent the points $Q^{00}_{00},...,Q^{MN}_{DD}$ as a matrix of points, in blocks of points:

$$\begin{bmatrix} Q^{00} & ... & Q^{0N} \\ \vdots & \ddots & \vdots \\ Q^{M0} & ... & Q^{MN} \end{bmatrix}$$

and in full of points as below,

$$\begin{bmatrix}
Q_{00}^{00} & \dots & Q_{0D}^{00} & Q_{00}^{01} & \dots & Q_{0D}^{01} & \dots & \dots & \dots & Q_{00}^{0N} & \dots & Q_{0D}^{0N} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
Q_{D0}^{00} & \dots & Q_{DD}^{00} & Q_{D0}^{01} & \dots & Q_{DD}^{01} & \dots & \dots & \dots & Q_{D0}^{0N} & \dots & Q_{DD}^{0N} \\
Q_{00}^{10} & \dots & Q_{0D}^{10} & Q_{00}^{11} & \dots & Q_{0D}^{11} & \dots & \dots & \dots & Q_{00}^{1N} & \dots & Q_{0D}^{1N} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
Q_{D0}^{10} & \dots & Q_{DD}^{10} & Q_{D0}^{11} & \dots & Q_{DD}^{11} & \dots & \dots & \dots & Q_{D0}^{1N} & \dots & Q_{DD}^{1N} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
Q_{00}^{M0} & \dots & Q_{0D}^{M0} & Q_{00}^{M1} & \dots & Q_{0D}^{M1} & \dots & \dots & \dots & Q_{00}^{MN} & \dots & Q_{0D}^{MN} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
Q_{D0}^{M0} & \dots & Q_{DD}^{M0} & Q_{D0}^{M1} & \dots & Q_{DD}^{M1} & \dots & \dots & \dots & Q_{D0}^{MN} & \dots & Q_{DD}^{MN}
\end{bmatrix}. \tag{2.14}$$

Proposition 8. *The solution of Problem 1 is given by the image by the map Ψ_{MN} of:*

$$\begin{bmatrix} B_{\alpha(s_0)D}^{-1} & & 0 \\ & \ddots & \\ 0 & & B_{\alpha(s_M)D}^{-1} \end{bmatrix} Q \begin{bmatrix} B_{\alpha(t_0)D}^{-1} & & 0 \\ & \ddots & \\ 0 & & B_{\alpha(t_N)D}^{-1} \end{bmatrix}^T. \tag{2.15}$$

Proof. The Formula (2.15) can be rewritten as $\Theta_{st}^{-1}(Q)$. As a consequence of the Proposition 7, Q is the sampling corresponding to the given $\boldsymbol{s}$, $\boldsymbol{t}$ of the surface $\Psi_{MN}(\Theta_{st}^{-1}(Q))$. In fact, since $\Theta_{st}^{-1}(Q)$ is the control polytope of the surface $\Psi_{MN}(\Theta_{st}^{-1}(Q))$, the sampling is given by $\Lambda_{st}(\Psi_{MN}(\Theta_{st}^{-1}(Q))) = \Theta_{st}(\Theta_{st}^{-1}(Q)) = Q$. ■

Proposition 9. *Λ_{st} and Ψ_{MN} define linear isomorphisms between $\mathcal{B}_{MN}^D$ and $(E^{(D+1)^2})^{(M+1)(N+1)}$.*

Proof. In this proof, we consider the restriction of Λ_{st} to $\mathcal{B}_{MN}^D$ and the retraction of Ψ_{MN} onto $\mathcal{B}_{MN}^D$.

It is clear that Λ_{st} is linear. And its inverse is $\Psi_{MN} \circ \Theta_{st}^{-1}$. In fact, given a uniform piecewise Bézier surface $\Gamma \in \mathcal{B}_{MN}^D$ with its control polytope P, we have $\Psi_{MN} \circ \Theta_{st}^{-1} \circ \Lambda_{st}(\Gamma) = \Psi_{MN} \circ \Theta_{st}^{-1}(\Theta_{st}(P)) = \Psi_{MN}(P) = \Gamma$, i.e,

$$\Psi_{MN} \circ \Theta_{st}^{-1} \circ \Lambda_{st} = Id_{\mathcal{B}_{MN}^D}. \tag{2.16}$$

For some $Q \in (E^{(D+1)^2})^{(M+1)(N+1)}$, the Proposition 8 says that the surface $\Psi_{MN}(\Theta_{st}^{-1}(Q))$ with the control points $\Theta_{st}^{-1}(Q)$ has the associated sampling which is Q, i.e, $\Lambda_{st}(\Psi_{MN}(\Theta_{st}^{-1}(Q))) = Q$. This means that

$$\Lambda_{st} \circ \Psi_{MN} \circ \Theta_{st}^{-1} = Id_{(E^{(D+1)^2})^{(M+1)(N+1)}}. \tag{2.17}$$

So Λ_{st} is an isomorphism. From Equation (2.17), since Θ_{st} is a linear isomorphism,

$$\Theta_{st}^{-1} \circ \Lambda_{st} \circ \Psi_{MN} = Id_{(E^{(D+1)^2})^{(M+1)(N+1)}}. \tag{2.18}$$

Notation 6. *Denote that*

$$\chi_{st} = \Theta_{st}^{-1} \circ \Lambda_{st} : \mathcal{C}^0([0,1]^2, E) \longrightarrow (E^{(D+1)^2})^{(M+1)(N+1)}.$$

By Equations (2.16), (2.18), we obtain $\Psi_{MN} \circ \chi_{st} = Id_{\mathcal{B}^D_{MN}}$ and $\chi_{st} \circ \Psi_{MN} = Id_{(E^{(D+1)^2})^{(M+1)(N+1)}}$. This shows that Ψ_{MN} is an isomorphism between $\mathcal{B}^D_{MN}$ and $(E^{(D+1)^2})^{(M+1)(N+1)}$. ■

Proposition 9 is important since it equips the set of uniform piecewise Bézier surfaces $\mathcal{B}^D_{MN}$ with a smooth manifold structure. Precisely, $\mathcal{B}^D_{MN}$ is a finite dimensional space. Moreover, together with Proposition 8, they allow to project any elements of $\mathcal{C}^0([0,1]^2, E)$ on $\mathcal{B}^D_{MN}$ using the sampling map Λ_{st}, by the map

$$\Psi_{st} \circ \chi_{st} : \mathcal{C}^0([0,1]^2, E) \longrightarrow \mathcal{B}^D_{MN}.$$

Let $\Delta \in \mathcal{C}^0([0.1]^2, E)$, denote $Q = \Lambda_{st}(\Delta)$ a sampling of Δ for some $\boldsymbol{s, t}$, then we have that $\Psi_{MN} \circ \Theta^{-1}_{st}(Q) \in \mathcal{B}^D_{MN}$ is such that the surface $\Psi_{MN} \circ \chi_{st}(\Delta) = \Psi_{MN} \circ \Theta^{-1}_{st}(Q)$ coincides with $\Delta([0,1])$ at least $(D+1)^2(M+1)(N+1)$ points counted with multiplicities on each patch.

What we have done shows that, instead of working directly with $\mathcal{B}^D_{MN}$ it is convenient to work on the "set of control polytopes", which is named $(E^{(D+1)^2})^{(M+1)(N+1)}$, by using sampling, Λ_{st}, and interpolation, Θ_{st}, that define linear isomorphism between control polytopes and the set of sampling points on surfaces. And we will keep this point of view until the end.

2.2.3 Tangent space $T\mathcal{B}^D_{MN}$ and deformation of surfaces

As we have already known, the map Ψ_{MN} defines a linear isomorphism between the space of control polytopes $(E^{(D+1)^2})^{(M+1)(N+1)}$ and the space of uniform piecewise Bézier surfaces $\mathcal{B}^D_{MN}$. For any $\gamma(s,t) \in \mathcal{B}^D_{MN}$, there exists $P \in (E^{(D+1)^2})^{(M+1)(N+1)}$ such that $\Psi_{MN}(P) = \gamma$ is given by $\chi_{st}(\gamma) = \Theta^{-1}_{st} \circ \Lambda_{st}(\gamma)$. This gives the following proposition.

Proposition 10. *Consider the map*

$$T\Psi_{MN} : T(E^{(D+1)^2})^{(M+1)(N+1)} \longrightarrow T\mathcal{B}^D_{MN}.$$

Then, for any $\gamma \in \mathcal{B}^D_{MN}$, the map

$$T\Psi^{-1}_{MN}(\gamma) : T_\gamma \mathcal{B}^D_{MN} \longrightarrow T_{\chi_{st}(\gamma)}(E^{(D+1)^2})^{(M+1)(N+1)}$$

is given by $T\Psi_{MN}(\chi_{st}(\gamma))^{-1}(\epsilon) = \Theta^{-1}_{st} \circ \Lambda_{st}(\epsilon) = \chi_{st}(\epsilon)$ for any $\epsilon(s,t) \in T_\gamma \mathcal{B}^D_{MN}$. Moreover, it is a linear isomorphism.

Proof. Since Ψ_{MN} is an linear isomorphism between $(E^{(D+1)^2})^{(M+1)(N+1)}$ and $\mathcal{B}^D_{MN}$, we obtain that,

$$T\Psi^{-1}_{MN}(\gamma)(\epsilon) = T\Psi_{MN}(\chi_{st}(\gamma))^{-1}(\epsilon) = \Psi^{-1}_{MN}(\epsilon) = \Theta^{-1}_{st} \circ \Lambda_{st}(\epsilon) = \chi_{st}(\epsilon).$$

And hence, $T\Psi^{-1}_{MN}(\gamma)$ is an isomorphism. ■

An element of $\epsilon(s,t) \in T_\gamma \mathcal{B}^D_{MN}$ is called a deformation surface. Proposition 10 is an essential step proving that manipulating a uniform piecewise Bézier surface, it is enough to manipulate its control polytope. This is shown in the lemma below.

Lemma 4. *Let $P \in (E^{(D+1)^2})^{(M+1)(N+1)}$, $\gamma \in \mathcal{B}^D_{MN}$ such that $\gamma(s,t) = \Psi_{MN}(P)(s,t) = B_2(P; s, t)$ and $\epsilon(s,t) \in T_\gamma \mathcal{B}^D_{MN}$, then:*

i. $\epsilon(s,t) = \Psi_{MN}(\chi_{st}(\epsilon))(s,t)$,

ii. $\gamma(s,t) + \epsilon(s,t) = \Psi_{MN}(P + \chi_{st}(\epsilon))(s,t)$.

Lemma 4 shows how a deformation on the space of uniform piecewise Bézier surfaces is lifted to the space of control polytopes. This profits the convenience of the vector space structure of both the space of uniform piecewise Bézier surfaces $\mathcal{B}^D_{MN}$ and the space of control polytopes $(E^{(D+1)^2})^{(M+1)(N+1)}$. Moreover, this structure also helps to define the distance between two such surfaces.

It is a fact that a deformation of a surface implies the perturbation at its sampled points. Conversely, a perturbation on a sampling of the surface can represent to a deformation of the entire surface. This is shown in Proposition 11.

Proposition 11. *Let $\gamma \in \mathcal{B}^D_{MN}$ with the control polytope P, $Q = \Lambda_{st}(\gamma)$ be a sampling of γ and δQ be a perturbation at the points Q, then the control points of the uniform piecewise Bézier surface passing through $Q + \delta Q$ at $(\boldsymbol{s}, \boldsymbol{t})$ are given by $P + \delta P$ where $\delta P = \Theta^{-1}_{st}(\delta Q)$.*

Proof. The perturbation δQ defines a deformation $\epsilon \in \mathcal{B}^D_{MN}$ by $\epsilon = \Psi_{MN} \circ \Theta^{-1}_{st}(\delta Q)$. According to Lemma 4, the deformed surface is then determined by

$$\gamma + \epsilon = \Psi_{MN}(P + \chi_{st}(\Psi_{MN} \circ \Theta^{-1}_{st}(\delta Q))).$$

Since $\chi_{st} \circ \Psi_{MN} = Id_{(E^{(D+1)^2})^{(M+1)(N+1)}}$, we obtain $\gamma + \epsilon = \Psi_{MN}(P + \Theta^{-1}_{st}(\delta Q))$. In addition,

$$\begin{aligned}\Lambda_{st}(\gamma + \epsilon) &= \Lambda_{st}(\Psi_{MN}(P + \Theta^{-1}_{st}(\delta Q))) \\ &= \Lambda_{st}(\Psi_{MN}(P)) + \Lambda_{st}(\Psi_{MN}(\Theta^{-1}_{st}(\delta Q))).\end{aligned}$$

As $\Lambda_{st} \circ \Psi_{MN} \circ \Theta^{-1}_{st} = Id_{(E^{(D+1)^2})^{(M+1)(N+1)}}$, see Equation (2.17), and it is trivial that $\Lambda_{st}(\Psi_{MN}(P)) = Q$. Hence, $\Lambda_{st}(\gamma + \epsilon) = Q + \delta Q$. The proposition is proved. ■

Reducing to the case of (D, D) Bézier surfaces, a deformation of a (D, D) Bézier surface is described by Proposition 12. Figure 2.8 and Figure 2.9 give an illustration of deformation of Bézier surfaces.

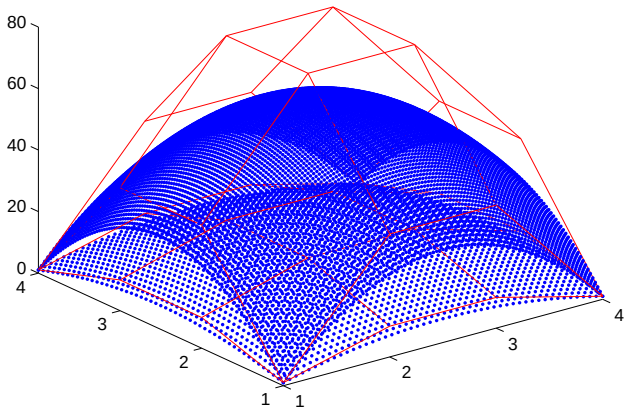


Figure 2.8: Deformation of a Bézier surface. The end points are fixed.

Proposition 12. *Given a (D, D) Bézier surface Γ with control polytope P. Let $Q = \lambda_{st}(\Gamma)$ be a sampling of Γ and δQ be a perturbation of sampled points Q, then the control points of the Bézier surface passing through $Q + \delta Q$ at $(\boldsymbol{s}, \boldsymbol{t})$ is given by $P + \delta P$ where*

$$\delta P = \theta_{st}^{-1}(\delta Q) = B_{sD}^{-1} \begin{bmatrix} \delta Q_{00} & \dots & \delta Q_{0D} \\ \vdots & \ddots & \vdots \\ \delta Q_{D0} & \dots & \delta Q_{DD} \end{bmatrix} B_{tD}^{-T}.$$

Note that the matrices B_{sD}^{-1} and B_{tD}^{-1} are also computed only once in the initialization for fixed $\boldsymbol{s}$, $\boldsymbol{t}$. The deformation of the control polytope of a (D, D) Bézier surface then has the cost of multiplication of three $(D+1) \times (D+1)$ matrices in three times. Thus, the computational cost of the deformation of a (D, D) Bézier surface is the same with the cost of interpolation of the same degree. This result is shown in Lemma 5.

Lemma 5. *The computational cost of the deformation of a (D, D) Bézier surface is $\mathcal{I}(D)$.*

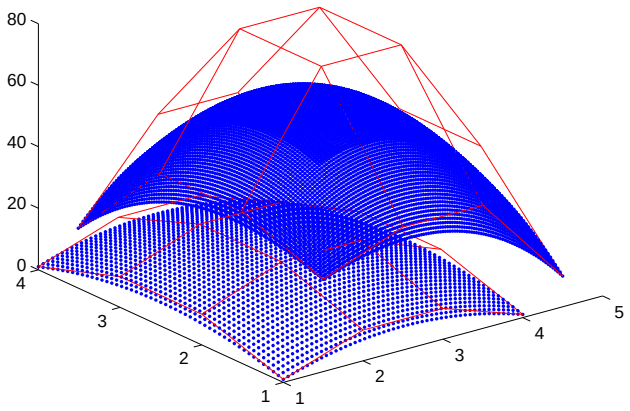


Figure 2.9: Deformation of a Bézier surface. The end points are moved.

2.2.4 Matrix form and subdivision of uniform piecewise Bézier surfaces

At first, we will show how to take a subdivision of a Bézier surface. We have known that the surface is parametrized by two variables s and t. Then, each subdivision on s and t will give a subdivision on the surface respectively. The subdivision corresponds to parameter s is called horizontal subdivision and the one respects to t is called vertical subdivision. This section will prove that it is possible to subdivide a Bézier surface with only its control polytope. This will be done after interpreting its definition into matrix form.

$$\begin{aligned} B_2(P; s, t) &= \sum_{j=0}^{D} \sum_{i=0}^{D} P_{ij} b_{iD}(s) b_{jD}(t) \\ &= \sum_{j=0}^{D} \left(\sum_{i=0}^{D} P_{ij} b_{iD}(s) \right) b_{jD}(t) \end{aligned}$$

$$
\begin{aligned}
B_2(P;s,t) &= \sum_{j=0}^{D}\left(\sum_{i=0}^{D} P_{ij}\begin{pmatrix} i \\ D \end{pmatrix}(1-s)^{D-i}s^i\right) b_{jD}(t) \\
&= \sum_{j=0}^{D}\left(\sum_{i=0}^{D} P_{ij}\begin{pmatrix} i \\ D \end{pmatrix} s^i \sum_{k=0}^{D-i}\begin{pmatrix} k \\ D-i \end{pmatrix}(-1)^k s^k\right) b_{jD}(t) \\
&= \sum_{j=0}^{D}\left(\sum_{i=0}^{D} P_{ij}\sum_{k=0}^{D-i}\begin{pmatrix} i \\ D \end{pmatrix}\begin{pmatrix} k \\ D-i \end{pmatrix}(-1)^k s^{i+k}\right) b_{jD}(t).
\end{aligned}
$$

Denote that

$$
C_D^i = \begin{pmatrix} i \\ D \end{pmatrix}, \quad \boldsymbol{s} = \begin{bmatrix} 1 & s & \dots & s^D \end{bmatrix}, \quad \boldsymbol{t}^T = \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix},
$$

and

$$
C = \begin{bmatrix}
1 & 0 & 0 & \dots & 0 \\
-C_D^1 & C_D^1 & 0 & \dots & 0 \\
C_D^2 & -C_D^1 C_{D-1}^1 & C_D^2 & \dots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
(-1)^D & (-1)^{D-1}C_D^1 & (-1)^{D-2}C_D^2 & \dots & 1
\end{bmatrix}, \tag{2.19}
$$

then,

$$
B_2(P;s,t) = \sum_{j=0}^{D}\begin{bmatrix} 1 & s & \dots & s^D \end{bmatrix} C \begin{bmatrix} P_{0j} \\ P_{1j} \\ \vdots \\ P_{Dj} \end{bmatrix} b_{jD}(t).
$$

Similarly, one can do the same with respect to t to obtain that

$$
B_2(P;s,t) = \begin{bmatrix} 1 & s & \dots & s^D \end{bmatrix} C \begin{bmatrix}
P_{00} & P_{01} & \dots & P_{0D} \\
P_{10} & P_{11} & \dots & P_{1D} \\
\vdots & \vdots & \ddots & \vdots \\
P_{D0} & P_{D1} & \dots & P_{DD}
\end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix}. \tag{2.20}
$$

Equation (2.20) gives the matrix form of a Bézier surface. A sampling of Bézier surface can be obtained by using its matrix form. This is described in Algorithm 4.

Suppose that we want to subdivide the surface $B_2(P;s,t)$ along the constant Bézier curve $s = \xi$, then the reparametrization of Equation (2.20) represents the so-called left-half part of the surface $B_2(P;s,t)$,

$$
B_2(P;s\xi,t) = \begin{bmatrix} 1 & s\xi & \dots & (s\xi)^D \end{bmatrix} C \begin{bmatrix}
P_{00} & P_{01} & \dots & P_{0D} \\
P_{10} & P_{11} & \dots & P_{1D} \\
\vdots & \vdots & \ddots & \vdots \\
P_{D0} & P_{D1} & \dots & P_{DD}
\end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix}
$$

Algorithm 4 Sampling of (D, D) Bézier surfaces.

Input:

$\begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix}$ the matrix of the points of control polytope of the Bézier surface $B_2(P_{00}, P_{01}, ..., P_{DD}; s, t)$ and $(s, t) \in [0, 1] \times [0, 1]$;

Output: Coordinates of a surface point in $\mathbb{R}^3$;

return

$$\begin{bmatrix} 1 & s & ... & s^D \end{bmatrix} C \begin{bmatrix} P_{00} & P_{01} & ... & P_{0D} \\ P_{20} & P_{21} & ... & P_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ P_{D0} & P_{D1} & ... & P_{DD} \end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix};$$

Thus, the left-half surface is defined by

$$B_2(P; s\xi, t) = \begin{bmatrix} 1 & s & ... & s^D \end{bmatrix} \begin{bmatrix} 1 & 0 & ... & 0 \\ 0 & \xi & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & \xi^D \end{bmatrix} CPC^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix}. \tag{2.21}$$

We denote that

$$S_{L\xi} = C^-1 \begin{bmatrix} 1 & 0 & ... & 0 \\ 0 & \xi & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & \xi^D \end{bmatrix} C, \tag{2.22}$$

and rewrite Equation (2.21) as

$$B_2(P; s\xi, t) = \boldsymbol{s} C S_{L\xi} \begin{bmatrix} P_{00} & P_{01} & ... & P_{0D} \\ P_{10} & P_{11} & ... & P_{1D} \\ \vdots & \vdots & \ddots & \vdots \\ P_{D0} & P_{D1} & ... & P_{DD} \end{bmatrix} C^T \boldsymbol{t}^T. \tag{2.23}$$

Equation (2.23) is to say that $B_2(P; s\xi, t)$, the left-half part of the surface $B_2(P; s, t)$, is also a Bézier surface and its control polytope is $S_{L\xi}P$. The computation for the right-half part of $B_2(P; s, t)$ can be done by the same way. Reparametrize Equation (2.20) by substituting s by $(\xi + (1 - \xi)s)$, we obtain

$$B_2(P; \xi + (1-\xi)s, t) = \begin{bmatrix} 1 & \xi + (1-\xi)s & ... & (\xi + (1-\xi)s)^D \end{bmatrix} CPC^T \boldsymbol{t}^T. \tag{2.24}$$

Since

$$\begin{bmatrix} 1 & \xi + (1-\xi)s & ... & (\xi + (1-\xi)s)^D \end{bmatrix} = \boldsymbol{s} \begin{bmatrix} 1 & \xi & \xi^2 & ... & \xi^D \\ 0 & 1-\xi & 2\xi(1-\xi) & ... & C_D^1 \xi^{D-1}(1-\xi) \\ 0 & 0 & (1-\xi)^2 & ... & C_D^2 \xi^{D-2}(1-\xi)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & ... & (1-\xi)^D \end{bmatrix},$$

the Equation (2.24) can be rewritten as

$$B_2(P;\xi+(1-\xi)s,t) = \begin{bmatrix} 1 & s & ... & s^D \end{bmatrix} CS_{\xi R}PC^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix} \tag{2.25}$$

where

$$S_{\xi R} = C^{-1} \begin{bmatrix} 1 & \xi & \xi^2 & ... & \xi^D \\ 0 & 1-\xi & 2\xi(1-\xi) & ... & C_D^1\xi^{D-1}(1-\xi) \\ 0 & 0 & (1-\xi)^2 & ... & C_D^2\xi^{D-2}(1-\xi)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & ... & (1-\xi)^D \end{bmatrix} C. \tag{2.26}$$

As consequence of Equation (2.25), the right-half part of $B_2(P;s,t)$ is also a Bézier surface determined by the control polytope $S_{\xi R}P$.

Hence, the horizontal subdivision of the surface $B_2(P;s,t)$ along the constant Bézier curve $s=\xi$ is given by

$$\begin{bmatrix} S_{L\xi}P \\ S_{\xi R}P \end{bmatrix}.$$

With respect to t, the vertical subdivision of $B_2(P;s,t)$ along the constant Bézier curve $t=\zeta$ will be

$$\begin{bmatrix} PS_{L\zeta}^T & PS_{\zeta R}^T \end{bmatrix}$$

where $PS_{L\zeta}^T$, $PS_{\zeta R}^T$ are respectively the control polytopes of two following Bézier surfaces,

$$B_2(P;s,\zeta t) = \begin{bmatrix} 1 & s & ... & s^D \end{bmatrix} CPS_{L\zeta}^T C^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix}$$

and

$$B_2(P;s,\zeta+(1-\zeta)t) = \begin{bmatrix} 1 & s & ... & s^D \end{bmatrix} CPS_{\zeta R}^T C^T \begin{bmatrix} 1 \\ t \\ \vdots \\ t^D \end{bmatrix}.$$

These subdivisions of $B_2(P;s,t)$ corresponding to s and t are respectively given by Algorithm 5 and Algorithm 6. The combination of these two methods give a segmentation of the considered surface into quarters,

$$\begin{bmatrix} S_{L\xi}PS_{L\zeta}^T & S_{L\xi}PS_{\zeta R}^T \\ S_{\xi R}PS_{L\zeta}^T & S_{\xi R}PS_{\zeta R}^T \end{bmatrix}.$$

They are associated with subdividing on the unit square $[0,1]\times[0,1]$, that the first one is of $0\le s\le\xi$ and $0\le t\le\zeta$, the second is of $0\le s\le\xi$ and $\zeta\le t\le 1$, etc. This is implemented by Algorithm 7. Usually, ξ and ζ are chosen at $\frac{1}{2}$. Once ξ and ζ are fixed, the matrices

Algorithm 5 Horizontal subdivision of (D, D) Bézier surfaces.

Input: $\begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix}$ the matrix of the points of control polytope of the Bézier surface $B([P_{00}, ..., P_{DD}]; s, t)$ and $\xi \in (0, 1)$;

Output: Two patches of the surface subdivided with respect to s at $s = \xi$;

return

$$\begin{bmatrix} S_{L\xi} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} \\ S_{\xi R} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} \end{bmatrix}$$

Algorithm 6 Vertical subdivision of (D, D) Bézier surfaces.

Input: $\begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix}$ the matrix of the points of control polytope of the Bézier surface $B([P_{00}, ..., P_{DD}]; s, t)$ and $\zeta \in (0, 1)$;

Output: Two patches of the surface subdivided with respect to t at $t = \zeta$;

return

$$\begin{bmatrix} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{L\zeta}^T & \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{\zeta R}^T \end{bmatrix}$$

Algorithm 7 Quadrangular subdivision of (D, D) Bézier surfaces.

Input: $\begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix}$ the matrix of the points of control polytope of the Bézier surface $B([P_{00}, ..., P_{DD}]; s, t)$ and $(\xi, \zeta) \in [0, 1] \times [0, 1]$;

Output: Four patches of the surface subdivided with both s and t along the curves $s = \xi$ and $t = \zeta$;

return

$$\begin{bmatrix} S_{L\xi} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{L\zeta}^T & S_{L\xi} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{\zeta R}^T \\ S_{\xi R} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{L\zeta}^T & S_{\xi R} \begin{bmatrix} P_{00} & ... & P_{0D} \\ \vdots & \ddots & \vdots \\ P_{D0} & ... & P_{DD} \end{bmatrix} S_{\zeta R}^T \end{bmatrix}$$

$S_{L\xi}, S_{\xi R}, S_{L\zeta}, S_{\zeta R}$ is determined. Then, the subdivisions of (D, D) Bézier surfaces will be done with only their control polytopes, without using sampling.

What we can do to split a uniform piecewise Bézier surface at $s = \xi$ or $t = \zeta$? Fortunately, through the map α, each split on the surface reduces to a subdivision on the associated (D, D) Bézier surface. But, remark that because of the singularity of uniform piecewise Bézier surfaces, the subdivision must be taken for all associated Bézier surfaces along these curve $s = \xi$ and, respectively, $t = \zeta$. For instance, given a uniform piecewise Bézier surface, named $\Gamma(s, t)$, with its control polytope denoted by P as following,

$$
\begin{array}{ccccccccccccccc}
P_{00}^{00} & \dots & P_{0D}^{00} & \dots & \dots & \dots & P_{00}^{0j} & \dots & P_{0D}^{0j} & \dots & \dots & \dots & P_{00}^{0N} & \dots & P_{0D}^{0N} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\
P_{D0}^{00} & \dots & P_{DD}^{00} & \dots & \dots & \dots & P_{D0}^{0j} & \dots & P_{DD}^{0j} & \dots & \dots & \dots & P_{D0}^{0N} & \dots & P_{DD}^{0N} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
P_{00}^{i0} & \dots & P_{0D}^{i0} & \dots & \dots & \dots & P_{00}^{ij} & \dots & P_{0D}^{ij} & \dots & \dots & \dots & P_{00}^{iN} & \dots & P_{0D}^{iN} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\
P_{D0}^{i0} & \dots & P_{DD}^{i0} & \dots & \dots & \dots & P_{D0}^{ij} & \dots & P_{DD}^{ij} & \dots & \dots & \dots & P_{D0}^{iN} & \dots & P_{DD}^{iN} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
P_{00}^{M0} & \dots & P_{0D}^{M0} & \dots & \dots & \dots & P_{00}^{Mj} & \dots & P_{0D}^{Mj} & \dots & \dots & \dots & P_{00}^{MN} & \dots & P_{0D}^{MN} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\
P_{D0}^{M0} & \dots & P_{DD}^{M0} & \dots & \dots & \dots & P_{D0}^{Mj} & \dots & P_{DD}^{Mj} & \dots & \dots & \dots & P_{D0}^{MN} & \dots & P_{DD}^{MN}
\end{array}
$$

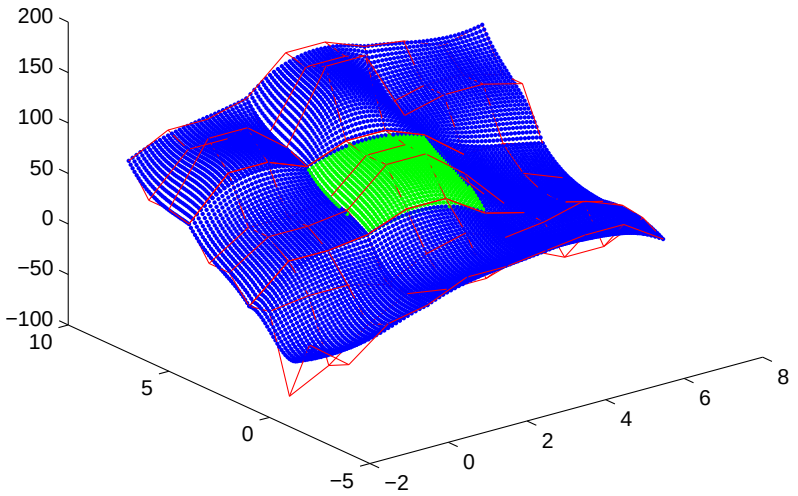


Figure 2.10: A patch of a uniform piecewise Bézier surface need subdivided.

and shortly,

$$P = \begin{bmatrix} P^{00} & ... & P^{0j} & ... & P^{0N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ P^{i0} & ... & P^{ij} & ... & P^{iN} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{Mj} & ... & P^{MN} \end{bmatrix}$$

where

$$P^{ij} = \begin{bmatrix} P^{ij}_{00} & ... & P^{ij}_{0D} \\ \vdots & \ddots & \vdots \\ P^{ij}_{D0} & ... & P^{ij}_{DD} \end{bmatrix}.$$

Recall that each P^{ij} defines a Bézier surface. They are the patches of the surface Γ. Assume that we want to split Γ along the curve $s = \xi$, then the obtained surface Γ_ξ is determined by its control polytope as below,

$$P_\xi = \begin{bmatrix} P^{00} & ... & P^{0j} & ... & P^{0N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ P^{i-1,0} & ... & P^{i-1,j} & ... & P^{i-1,N} \\ S_{L\alpha(\xi)}P^{i0} & ... & S_{L\alpha(\xi)}P^{ij} & ... & S_{L\alpha(\xi)}P^{iN} \\ S_{\alpha(\xi)R}P^{i0} & ... & S_{\alpha(\xi)R}P^{ij} & ... & S_{\alpha(\xi)R}P^{iN} \\ P^{i+1,0} & ... & P^{i+1,j} & ... & P^{i+1,N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{Mj} & ... & P^{MN} \end{bmatrix}$$

where α is given by Equations (2.10) and i is such that $\xi \in [s_{i0}, s_{iD}]$. If Γ is split along the curve $t = \zeta$ then the resulted surface Γ_ζ has control points being of the form, for $\zeta \in [t_{j0}, t_{jD}]$,

$$P_\zeta = \begin{bmatrix} P^{00} & ... & P^{0,j-1} & P^{0j}S^T_{L\alpha(\zeta)} & P^{0j}S^T_{\alpha(\zeta)R} & P^{0,j+1} & ... & P^{0N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P^{i0} & ... & P^{i,j-1} & P^{ij}S^T_{L\alpha(\zeta)} & P^{ij}S^T_{\alpha(\zeta)R} & P^{i,j+1} & ... & P^{iN} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{M,j-1} & P^{Mj}S^T_{L\alpha(\zeta)} & P^{Mj}S^T_{\alpha(\zeta)R} & P^{M,j+1} & ... & P^{MN} \end{bmatrix}$$

In particular, if the surface Γ is simultaneously subdivided along the curves $s = \xi$ and $t = \zeta$ then the new surface $\Gamma_{\xi\zeta}$ is determined by $P_{\xi\zeta}$ as following,

$$\begin{bmatrix} P^{00} & ... & P^{0,j-1} & P^{0j}S^T_{L\alpha(\zeta)} & P^{0j}S^T_{\alpha(\zeta)R} & P^{0,j+1} & ... & P^{0N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P^{i-1,0} & ... & P^{i-1,j-1} & P^{i-1,j}S^T_{L\alpha(\zeta)} & P^{i-1,j}S^T_{\alpha(\zeta)R} & P^{i-1,j+1} & ... & P^{i-1,N} \\ S_{L\alpha(\xi)}P^{i0} & ... & S_{L\alpha(\xi)}P^{i,j-1} & S_{L\alpha(\xi)}P^{ij}S^T_{L\alpha(\zeta)} & S_{L\alpha(\xi)}P^{ij}S^T_{\alpha(\zeta)R} & S_{L\alpha(\xi)}P^{i,j+1} & ... & S_{L\alpha(\xi)}P^{iN} \\ S_{\alpha(\xi)R}P^{i0} & ... & S_{\alpha(\xi)R}P^{i,j-1} & S_{\alpha(\xi)R}P^{ij}S^T_{L\alpha(\zeta)} & S_{\alpha(\xi)R}P^{ij}S^T_{\alpha(\zeta)R} & S_{\alpha(\xi)R}P^{i,j+1} & ... & S_{\xi R}P^{iN} \\ P^{i+1,0} & ... & P^{i+1,j-1} & P^{i+1,j}S^T_{L\alpha(\zeta)} & P^{i+1,j}S^T_{\alpha(\zeta)R} & P^{i+1,j+1} & ... & P^{i+1,N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P^{M0} & ... & P^{M,j-1} & P^{Mj}S^T_{L\alpha(\zeta)} & P^{Mj}S^T_{\alpha(\zeta)R} & P^{M,j+1} & ... & P^{MN} \end{bmatrix}$$

We can see how these subdivisions work by illustrations in Figures 2.11-2.16.

We note that for given D, we can compute the matrix C and then, use it for all. In practice, we use bicubic Bézier patches. This means $D = 3$. And when we fix $\xi = \zeta = \frac{1}{2}$, then the matrices C, $S_{L\frac{1}{2}}$ and $S_{\frac{1}{2}R}$ will be computed explicitly. They are used during the implementation without computing again. We will discuss more about this in Chapter 5.

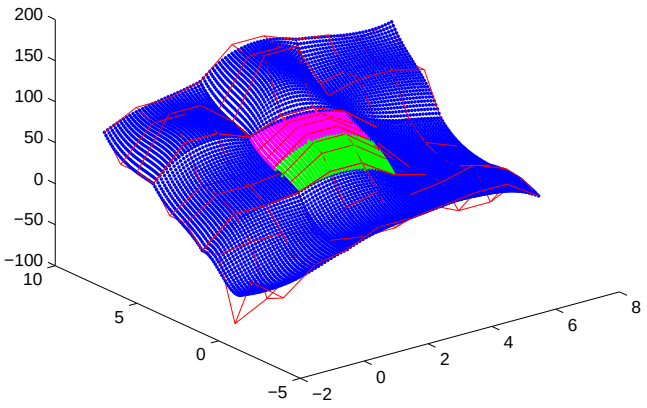


Figure 2.11: The considered patch of the uniform piecewise Bézier surface, in Figure 2.10, is subdivided in horizontal direction.

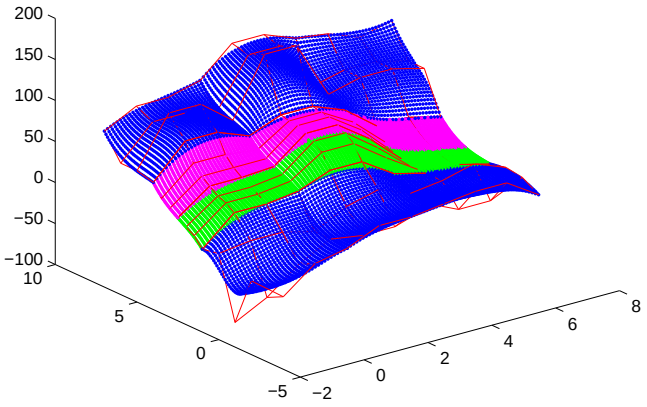


Figure 2.12: A regular subdivision of the uniform piecewise Bézier surface, in Figure 2.10, in horizontal direction.

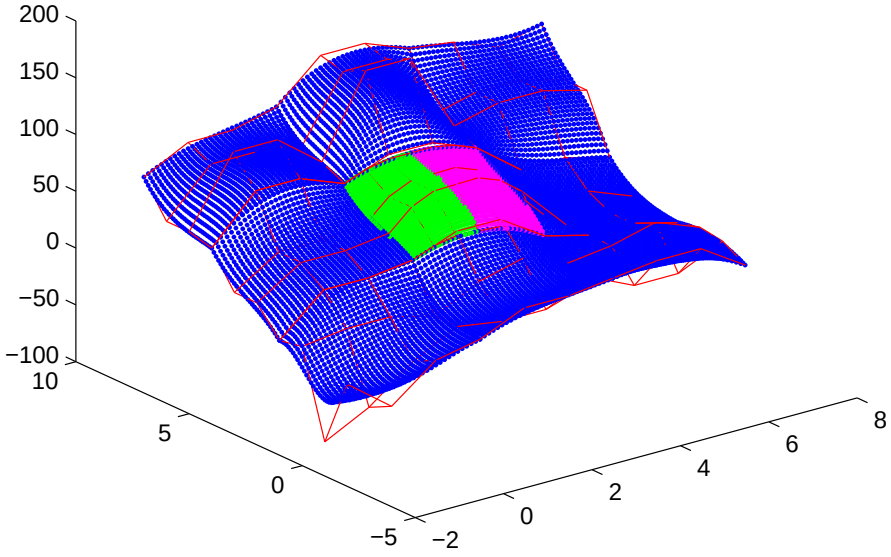


Figure 2.13: The considered patch of the uniform piecewise Bézier surface, in Figure 2.10, is subdivided in vertical direction.

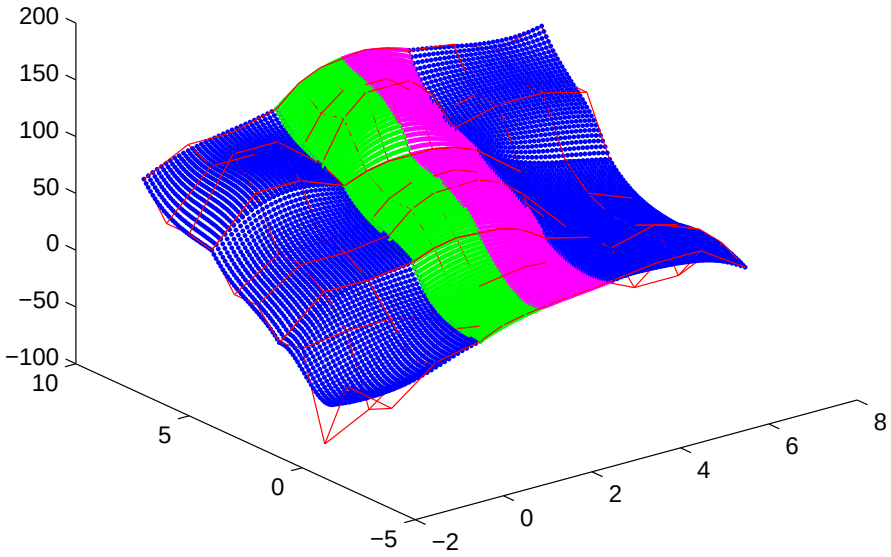


Figure 2.14: A regular subdivision of the uniform piecewise Bézier surface, in Figure 2.10, in vertical direction.

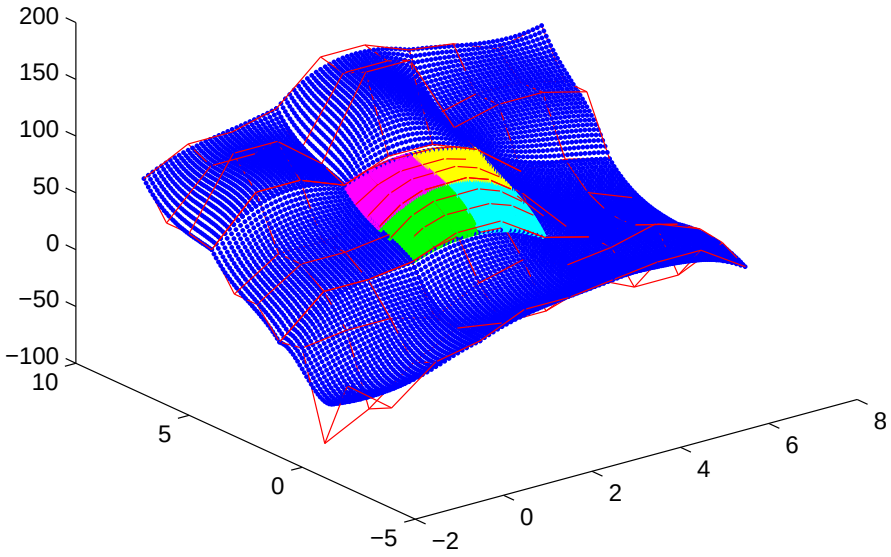


Figure 2.15: The considered patch of the uniform piecewise Bézier surface, in Figure 2.10, is subdivided into four new patches.

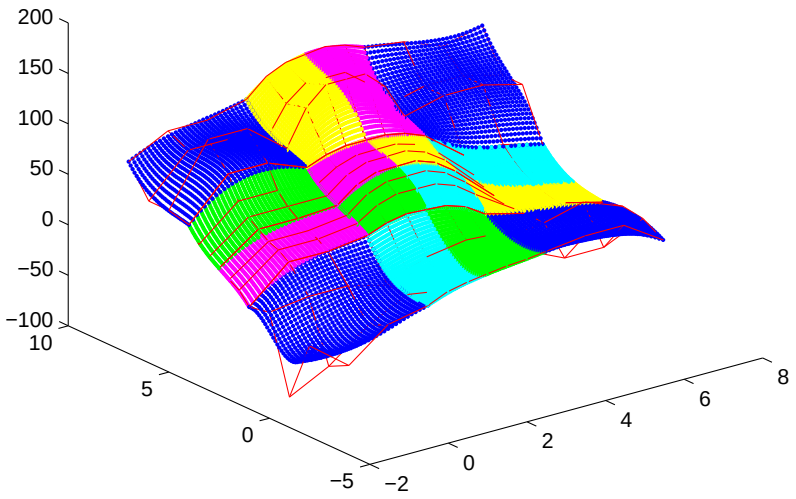


Figure 2.16: A regular subdivision of the uniform piecewise Bézier surface, in Figure 2.10, in both horizontal and vertical directions.

Chapter 3

Applications to Shape Optimization

This chapter will show how to exploit what we have done in Chapter 2, in context of shape optimization. The set of all shapes whose boundary is a simple closed uniform piecewise Bézier surface is an admissible set. A computable shape gradient will generate a vector field on the set of simple closed uniform piecewise Bézier surfaces. Thus, local extrema of shape cost functional induce at least a fixed point of the vector field. The illustration for this one will be given by an application to a problem of image segmentation presented in Chapter 5.

In this chapter, the considered space is still three-dimensional real space. We denote that $E = \mathbb{R}^3$.

3.1 Shape optimization problem

A typical shape optimization problem can be posed as the problem of finding the shape which is optimal in that it minimizes a certain cost functional while satisfying given constraints, as followings: given a set of admissible shapes $\mathcal{A}$ and a functional $F : \mathcal{A} \to \mathbb{R}^+$, find a shape $\alpha \in \mathcal{A}$ such that for all other shapes $\beta \in \mathcal{A}$, $F(\alpha) \leq F(\beta)$. Usually, the space of admissible shapes need equipped a structure of manifold. This will enable to compute the shape gradient $\nabla F(\beta)$ which expresses the evolution of F with respect to a deformation of the shape β. It can be considered that $\nabla F(\beta)$ assigns each point $M \in \beta$ to a deformation vector $\nabla F(\beta)(M) \in T_M E$. The computation of such a gradient often results in solving a system of partial differential equations. Commonly, a gradient method is used to solve this kind of problem when $\nabla F(\beta)$ is computable.

Here we consider a shape optimization problem related to image segmentation and shape recognition. The admissible shapes are ones whose boundary are closed compact oriented surfaces, and we will focus on geometric optimization, i.e, keep the topology of the shapes fixed.

Notation 7. *We denote $\mathcal{C}_c^0([0,1]^2, E)$ the set of closed compact oriented surfaces, and*

$$\mathcal{B}_c = \left\{ \gamma \in \mathcal{B}_{MN}^D \;\middle|\; \begin{array}{c} \gamma(1,.) = \gamma(0,.), \\ \gamma(.,1) = \gamma(.,0), \\ \forall (s,t),(u,v) \in (0,1)^2, \gamma(s,t) = \gamma(u,v) \Rightarrow (s,t) = (u,v) \end{array} \right\}.$$

Let $\Omega \subset \mathcal{P}(E)$ be such that for each $\omega \in \Omega$ the boundary $\partial\omega$ of ω is a closed compact oriented surface, i.e,

$$\Omega = \{\omega \in \mathcal{P}(E) | \partial\omega \in \mathcal{C}_c^0([0,1]^2, E)\}.$$

Let $F : \Omega \to \mathbb{R}^+$ be a smooth function such that $\nabla F(\omega) : \partial\omega \to TE$ is well defined. Every $\gamma \in \mathcal{B}_c$ is associated with the shape $\bar{\gamma}$ such that $\partial\bar{\gamma} = \gamma$. Then $\Omega^D_{MN} = \{\omega \in \mathcal{P} | \partial\omega \in \mathcal{B}_c\}$ is a set of admissible shapes. Recall that piecewise Bézier surfaces well approximate the space $\mathcal{C}^0([0,1]^2, E)$. Thus, Ω^D_{MN} are rather good approximations of Ω. The shape optimization will be stated in Problem 2.

Problem 2. *Find $\omega_0 \in \Omega$ such that for all $\omega \in \Omega$, $F(\omega_0) \leq F(\omega)$.*

Consider the shape gradient ∇F, for each $\omega \in \Omega$, the geometric gradient $\nabla F(\omega)$, then, assigns to each point Q on the boundary $\partial\omega$ a perturbation vector $\nabla F(\omega)(Q) \in T_Q E$ which needs to decrease the objective functional,

$$\nabla F(\omega) : \partial\omega \ni Q \longmapsto \nabla F(\omega)(Q) \in T_Q E.$$

The basic idea of the approach is to use $\nabla F(\omega)$ as a deformation of the boundary $\partial\omega$ to obtain a better shape.

Let $\bar{\gamma} \in \Omega^D_{MN}$, then $\partial\bar{\gamma} = \gamma \in \mathcal{B}_c$. Let Q is of the form shown in (2.14) be a sampling of γ for some $\boldsymbol{s}$, $\boldsymbol{t}$, i.e, $Q = \Lambda_{st}(\gamma)$. The control polytope of γ is given by $P = \chi_{st}(\gamma)$ for given $\boldsymbol{s}$, $\boldsymbol{t}$. A perturbation on Q is determined by the perturbation $\nabla F(\bar{\gamma})$ on the boundary γ of the shape $\bar{\gamma}$,

$$\delta Q = \begin{bmatrix} \nabla F(\bar{\gamma})(Q^{00}_{00}) & \dots & \nabla F(\bar{\gamma})(Q^{0N}_{DD}) \\ \vdots & \ddots & \vdots \\ \nabla F(\bar{\gamma})(Q^{M0}_{00}) & \dots & \nabla F(\bar{\gamma})(Q^{MN}_{DD}) \end{bmatrix}$$

The control polytope of the deformation surface δQ will be computed by $\delta P = \chi_{st}(\delta Q)$. Then, we obtain the sampling on the surface $\sigma(s,t) = \Psi_{MN}(P + \delta P)(s,t)$ by following Proposition 13.

Proposition 13. *$\sigma(s_{ij}, t_{kl}) = Q^{ik}_{jl} + \nabla F(\bar{\gamma})(Q^{ik}_{jl})$ for all $i \in \{0, ..., M\}$, $k \in \{0, ..., N\}$ and $j, l \in \{0, ..., D\}$.*

Proof. In fact, for given $\boldsymbol{s}$, $\boldsymbol{t}$, we have

$$\begin{aligned} \Lambda_{st}(\sigma) &= \Lambda_{st}(\Psi_{MN}(P + \delta P)) \\ &= \Theta_{st}(P + \delta P) \\ &= \Theta_{st}(P) + \Theta_{st}(\delta P) \\ &= \Lambda_{st}(\Psi_{MN}(P)) + \Lambda_{st}(\Psi_{MN}(\delta P)) \\ &= Q + \delta Q. \end{aligned}$$

The proposition is proved. ■

3.2 Vector field on $\mathcal{B}_c$ and local extrema of shape cost functional

Each shape gradient generates a vector field on B_c. The necessary condition to obtain a local minimum of functional F is that this associating vector field vanishes at this point.

Proposition 14. *For some compatible subdivision $\boldsymbol{s}$, $\boldsymbol{t}$, the map $\Psi_{MN} \circ \Theta^{-1}_{st}$ associates to each shape gradient ∇F a vector field $V_F : \mathcal{B}_c \to T\mathcal{B}_c$, determined by*

$$\mathcal{B}_c \ni \gamma \longmapsto \Psi_{MN} \circ \Theta^{-1}_{st}(\nabla F(\bar{\gamma})(\Lambda_{st}(\gamma))).$$

Proposition 15. *Assume that ∇F is such that $\nabla F(\omega_1)(q) = \nabla F(\omega_2)(q)$ for all $\omega_1, \omega_2 \in \Omega$ and for all $q \in E$. Let $\omega \in \Omega$ such that $\nabla F(\omega) \equiv 0$, then for all M, N nonnegative integers there is $\gamma \in \mathcal{B}_c$ satisfied $V_F(\gamma) = 0$. In other words, every optimum of F induces at least a fixed point of V_F over $\mathcal{B}_c$.*

Proof. Since $\partial\omega \in \mathcal{C}_c^0([0,1]^2, E)$, there exists a closed compact surface $\gamma \in \mathcal{B}_c$ such that they coincides at least $(D+1)^2(M+1)(N+1)$ points. Denote these points by Q being of the form shown in (2.14), indeed, Q is determined by $\Lambda_{st}(\partial\omega)$ for some $\boldsymbol{s}$, $\boldsymbol{t}$ and γ is given by $\Psi_{st} \circ \Theta_{st}^{-1}(Q)$. This means $Q = \Lambda_{st}(\partial\omega) = \Lambda_{st}(\gamma)$. Thus,

$$\begin{aligned} V_F(\gamma) &= \Psi_{MN} \circ \Theta_{st}^{-1}(\nabla F(\bar{\gamma})(\Lambda_{st}(\gamma))) \\ &= \Psi_{MN} \circ \Theta_{st}^{-1}(\nabla F(\bar{\gamma})(Q)) \\ &= \Psi_{MN} \circ \Theta_{st}^{-1}(\nabla F(\omega)(Q)) \\ &= 0. \end{aligned}$$

Moreover, the deformation surface of γ induced by de gradient of F vanishes at least $(D+1)^2(M+1)(N+1)$ points while it is a uniform piecewise Bézier surface of bi-degree (D, D). So it is parametrized by zero polynomials. Hence, its control polytope is reduced to origin. ■

Proposition 15 is to say that a local extremum of F induces a local extremum of its restriction to $\mathcal{B}_c$.

Based on Proposition 13 and Proposition 15, an algorithm is developed to find a good approximate solution of Problem 2. This is used in Algorithm 8.

Algorithm 8 Algorithm for shapes optimization

Input: An initial shape ω such that $\partial\omega = \Psi_{MN}(P) \in \mathcal{B}_c$
Output: The control polytope P of the boundary of a local minimum of $F(\omega)$.

```
σ ← Ψ_MN(P)
while criterium not satisfied do
    δP ← χ_st(∇F(σ̄)(Λ_st(σ)))
    P ← P + δP
    σ ← Ψ_MN(P)
end while
```

Chapter 4

3D Image Segmentation

In this chapter, we try to approach image segmentation in case of dimension 3, about 3D images and Canny edge detection for 3D images. 3D edge detectors play an important role in 3D free form method. They produce the gradient edge map needed for 3D free form method. The gradient edge map is then used to orient the deformation of active surfaces.

4.1 Images

Definition 5. *An image is a function given by*

$$\begin{aligned} I : \mathbb{R}^m &\longrightarrow \mathbb{R}^n \\ x &\longmapsto I(x) \end{aligned}$$

where x is a spatial point with coordinates of dimension m. Vector $I(x)$ is called the color of the point x and the value $||I(x)||$ is called the intensity of the image at x.

For example, with $m = 2$ and $n = 1$, we have a regular black-and-white image. When $m = 2$ and $n = 3$, the function I presents a color image. For these images, x has coordinates of dimension 2 and represents image position. We call them 2D images. In the case of color image, given a point x, the quantity $I(x)$ identify the color of the image at that point by a vector with three components. They are the red, the green, the blue components of the desired color. If x and $I(x)$ are finite and discrete quantities, we call the image a digital image. A digital image is composed of a finite number of elements named pixels. By this mean, a digital image can be considered a matrix whose component presents a pixel. For instance, a gray image is a matrix whose components are in the interval $[0, 1]$, as shown in Figure 4.1.

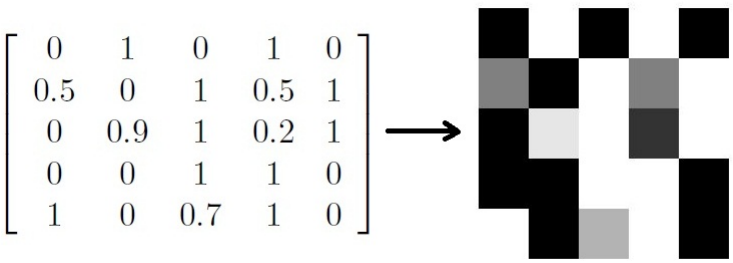


Figure 4.1: A 2D gray image.

In addition, when $m = 3$, the function I presents a three-dimensional image. Then a 3D digital image is an $(p \times q \times r)$ matrix whose component corresponds to a voxel. Like a pixel in 2D images, a voxel is a smallest unit of 3D images which has informations of image position and image color, i.e, it represents a value on a regular grid in three-dimensional space. Figure 4.2 illustrates a 3D color image.

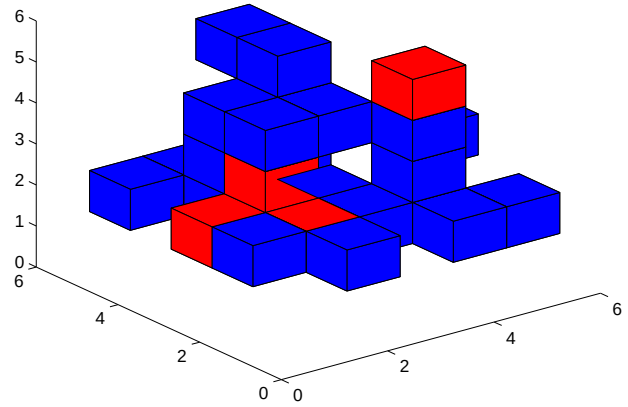


Figure 4.2: A 3D image.

3D binary image is a particular example for 3D images. This image satisfies that each of voxels has intensity level of either 0 or 1. This means 3D binary image is a $(p \times q \times r)$ matrix whose components are in the set $\{0, 1\}$. For instance, a sphere in three dimensional space is described by a binary image shown in Figure 4.3.

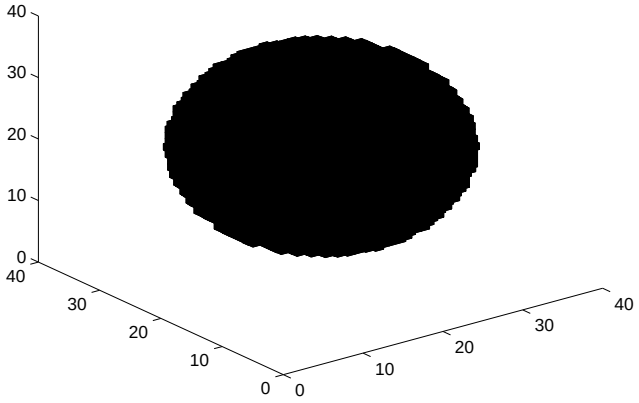


Figure 4.3: A binary image of a sphere in 3D space represented by voxels.

Binary images are produced from color images by segmentation. Segmentation is the process of assigning each pixel (corresponds to voxel for 3D images) in the source image to two or more classes. If there are more than two classes then the usual result is several binary images. Typically, edge detection creates a binary image with some pixels (corresponds to voxels) assigned to edge pixels (corresponds to edge voxels), and is also a first step in further segmentation. More clearly, edge is the discontinuity of intensity in the image and edge detector is used to detect them. Here we consider a particular one called Canny edge detector.

4.2 3D Canny edge detection

Many image processes work only based on the topological properties of images which are involving to the edges in the images. Now several edge detectors exist. In general, their purpose is to significant reduce the amount of data in an image while keeping the structural properties to be used for further image processing. Canny edge detector is a such one developed by John F. Canny in 1986. It uses a multi-stage algorithm to detect a wide range of edges in images.

In this section, we focus on gray images, i.e, $I : \mathbb{R}^m \longrightarrow \mathbb{R}$.

Canny used the three following criteria to obtain an optimal edge detection algorithm:

1. **Detection:** The probability of detecting real edge points should be maximized while the probability of falsely detecting non-edge points should be minimized, i.e the algorithm should mark as many real edges in the image as possible.
2. **Localization:** The detected edges should be as close as possible to the edges in real image.
3. **Responses:** A real edge in the image should not result in more than one detected edge.

Canny edge detection can be summarized in 5 separate steps:

1. **Smoothing:** Image noise reduction.
2. **Finding gradients:** Marking the edges where the gradients of the image has large magnitudes.
3. **Non-maximum suppression:** Only local maxima should be marked as edges.
4. **Double thresholding:** Potential edges are determined by thresholding.
5. **Edge tracking by hysteresis:** Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge.

Now, we go on to detail these five steps of 3D Canny edge detection algorithm.

4.2.1 Smoothing

Noise largely affects on edge detection. Certainly, it is mistaken for edges. And all images contain some amount of noise. Thus, the important step, also the first step is to remove the noise. Usually, one uses Gaussian filter to smooth images. Gaussian filtering kernel is given by Gaussian function,

$$g(x) = \frac{1}{(\sqrt{2\pi\sigma^2})^m} \exp\left(-\frac{||x||^2}{2\sigma^2}\right), \tag{4.1}$$

where m is the dimension of x, and $||.||$ is Euclidean norm.

In higher dimension, indeed, Gaussian filter is composed of one-dimensional Gaussian filterings on each direction with the same σ,

$$g_1(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{(-\frac{t^2}{2\sigma^2})}.$$

Moreover, the smoothing effect of Gaussian filtering is generated by convolving an image with a kernel of Gaussian values. Denote S the obtained image, then

$$S(x) = (I * g)(x) = \int_{\mathbb{R}^m} I(x-u)g(u)du.$$

In practice, because of working on digital images, one needs a discretization of the convolution,

$$S[x] = (I * g)[x] = \sum_i I[x-i]g[i].$$

This requires a discrete approximation of Gaussian kernel. For instance, the following is a sampling of one-dimensional Gaussian function at discrete points, with $\sigma = 1.04$,

$$K_{g_1} = [\ 0.0060 \quad 0.0604 \quad 0.2416 \quad 0.3836 \quad 0.2416 \quad 0.0604 \quad 0.0060\ ]. \tag{4.2}$$

And in two dimension, Patrice Delmas mentioned a 5×5 approximating Gaussian matrix in his lectures on Gaussian filtering [13], with $\sigma = 1$,

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix} \tag{4.3}$$

When taking the discrete Gaussian values needed for a kernel from the continuous Gaussian function, the sum of the values will be different from 1. This will lead to a darkening or brightening of the image. In order to remedy this, the kernel should be normalized by dividing each term in the kernel by the sum of all terms.

For 3D images, a three-dimensional Gaussian kernel can be used. But in fact, one can use 1D Gaussian kernel to each direction separably. This is the best way to take advantage of Gaussian filtering's property, and requires fewer calculations while the resulting effect is the same as convolving with a three-dimensional kernel.

4.2.2 Finding gradients

Basically, Canny algorithm finds the gap of the intensity in the smoothed image. This is found by determining the gradients of the image. In the discrete case, image gradients can be approximated by the difference between the left and the right voxels [9],

$$\frac{\partial I}{\partial x_1}(i_1) \approx \frac{1}{2}\left(I(i_1+1, i_2, ..., i_m) - I(i_1-1, i_2, ..., i_m)\right),$$

then the associated convolution kernel is:

$$\partial_{x_1} = \frac{1}{2}[\ -1 \quad 0 \quad 1\ ].$$

Canny algorithm uses Sobel operators, [3], to find the gradients of smoothed image. For 2D images, they are given by the following kernels for each direction,

$$K_{Gx} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

and

$$K_{Gy} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

The convolutions of the image with K_{Gx} and K_{Gy} give the image gradients in the x- and the y-direction respectively. Edge strengths characterize to edges. They are needed by Canny edge detector to track the edges. Indeed, they are the gradient magnitudes computed by Euclidean norm or simplified by Manhattan norm of the gradient vectors $G = (G_x, G_y)$,

$$||G||_2 = \sqrt{G_x^2 + G_y^2},$$
$$||G||_1 = |G_x| + |G_y|,$$

where G_x is the gradient in the x-directions and G_y is the one corresponds to the y-direction.

Note that Sobel kernel is a combination of a differentiation kernel for a given direction and a smoothing kernel for the orthogonal directions. For example, the kernel K_{Gx} can be rewritten as a product,

$$K_{Gx} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}.$$

This gives a way to transfer the 2D version to 3D case. A 3D Sobel operator can be obtained by calculating the partial derivative in a given direction and smoothing in the 2 orthogonal directions. For instance, the Sobel kernel for z-direction is as below,

$$K_{Gz}(:,:,-1) = \begin{bmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ -1 & -2 & -1 \end{bmatrix}, \qquad K_{Gz}(:,:,0) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$K_{Gz}(:,:,1) = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

Then we also have a 3D version to calculate the gradient magnitudes,

$$||G||_2 = \sqrt{G_x^2 + G_y^2 + G_z^2},$$
$$\text{or} \quad ||G||_1 = |G_x| + |G_y| + |G_z|,$$

where G_x , G_y, G_z are the gradients in the x-, y- and z-directions respectively.

4.2.3 Non-maximum suppression

This step is to make the edges in the image of gradient magnitudes become "sharper". Since only the locally biggest gaps of image intensity need found, the step will be done by keeping all local maxima in the gradient image and deleting others. For 2D images, each pixel of the gradient image is locally compared with two points which are located in the gradient direction at this considered pixel. For instance, the considered pixel is at i, then it will be compared with two points, $i_- = i - \frac{G(i)}{||G(i)||}$ and $i_+ = i + \frac{G(i)}{||G(i)||}$, where $G(i) = (G_x(i), G_y(i))$ is the gradient vector at i. If it is the largest, it is kept. Otherwise, it is suppressed, i.e, removed. In practice, we use linear interpolation to compute $G(i_-)$ and $G(i_+)$ and then the comparison follows, source from David Forsyth, UC Berkeley, [14].

In the case of three dimension, one can do by the same way. Each voxel i will be compared with two points in its gradient direction, $i_- = \left(i - \frac{G(i)}{||G(i)||}\right)$ and $i_+ = \left(i + \frac{G(i)}{||G(i)||}\right)$ where $G(i) = (G_x(i), G_y(i), G_z(i))$ is the gradient vector at i, so as to determine whether it is a local maximum, see in Figure 4.4. If $G(i) > G(i_-)$ and $G(i) > G(i_+)$, it is kept. Otherwise, it is suppressed.

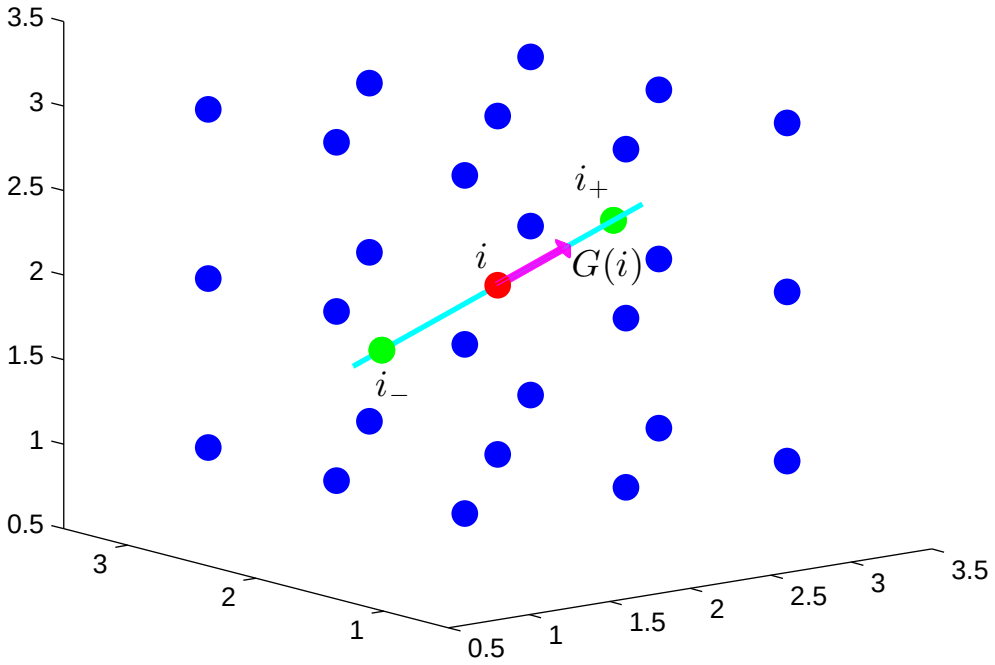


Figure 4.4: 3D non-maximum suppression.

4.2.4 Double thresholding

After non-maximum suppression step, the resulting image contains the edge-voxels which are marked with their strength voxel-by-voxel. Some of them can be true edges in the image, but others may come from noise and color variations for example due to rough surfaces. Thus, as possible, Canny edge detection algorithm uses double-thresholding to decrease the number of

false ones. Edge voxels stronger than the high threshold are marked as strong, edge voxels weaker than the low threshold are removed and others are marked weak.

4.2.5 Edge tracking by hysteresis

Right now, strong edges become "certain edges" which are present in the final edge image, while weak edges are either true edges or false ones coming from noise or color variations. Logically, there is no reason that noise and other small variations result in a strong edge. They are distributed independently of edges on the entire image. So weak edges are chosen if and only if they are connected to strong edges. By observing the neighborhood voxels (8-connected neighborhood for 2D and 26-connected neighborhood for 3D), see in Figure 4.5, a weak edge is present in the final edge image if and only if there is a strong one in its neighborhood.

Some resulting image of Canny edge detector are presented in Figures 4.6-4.11.

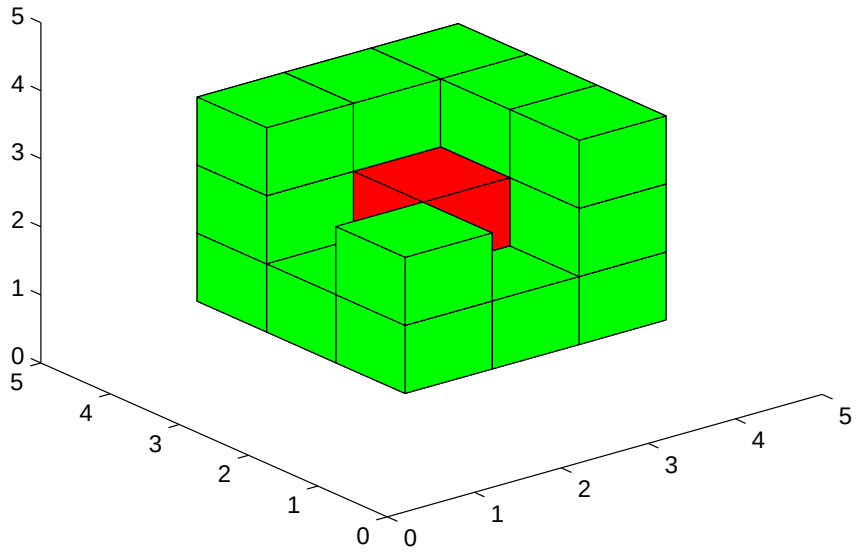


Figure 4.5: Neighborhood voxels of the red one.

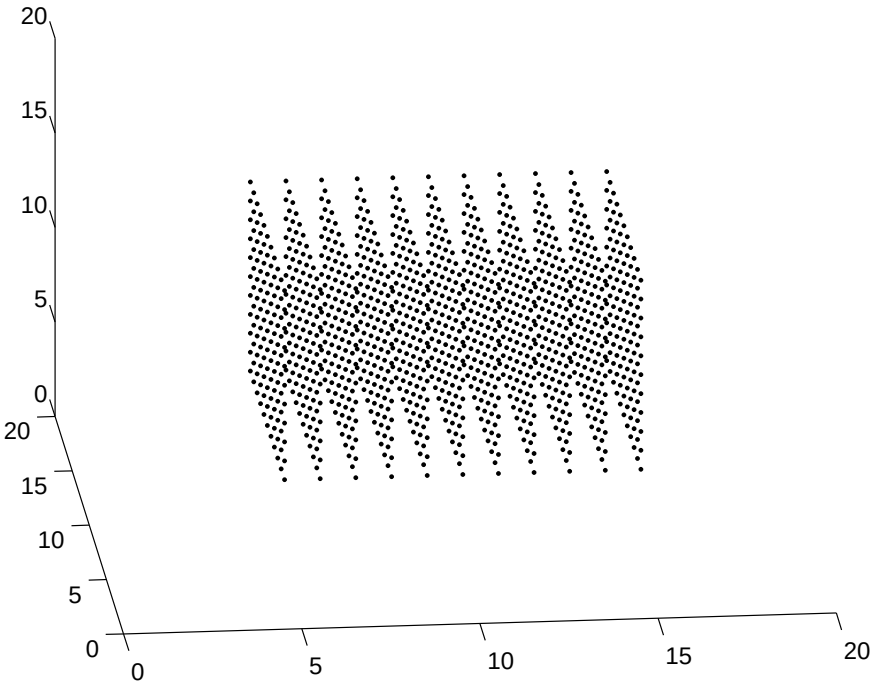


Figure 4.6: 3D image of a solid box represented by points.

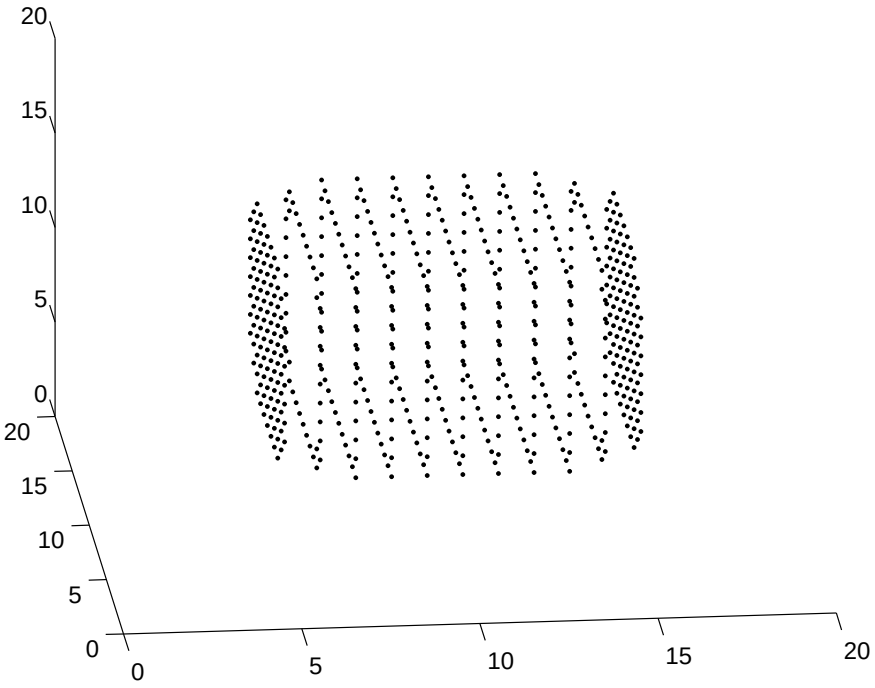


Figure 4.7: 3D Canny edge detector's result. A gradient edge image of the image shown in Figure 4.6, represented by points.

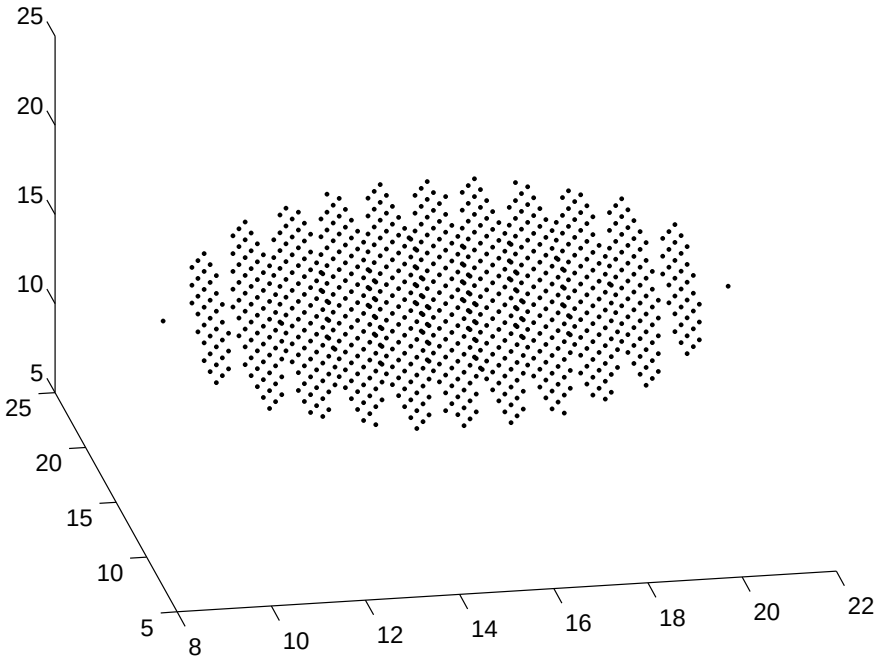


Figure 4.8: 3D image of a solid sphere represented by points.

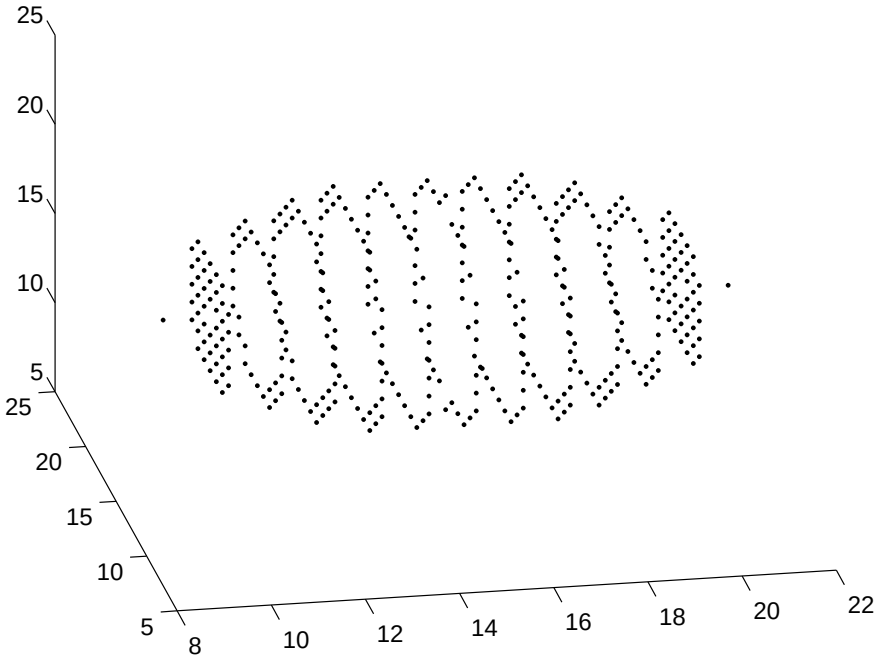


Figure 4.9: 3D Canny edge detector's result. A gradient edge image of the image shown in Figure 4.8, represented by points.

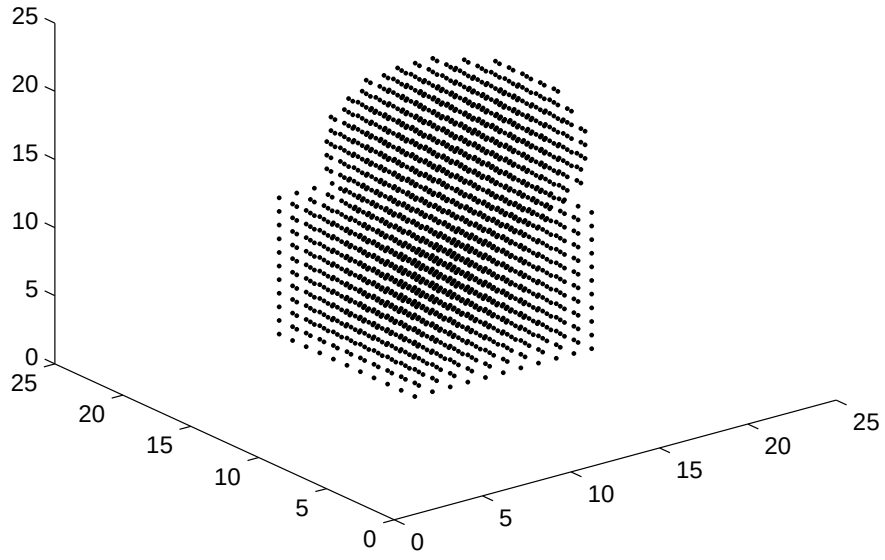


Figure 4.10: A 3D image represented by points.

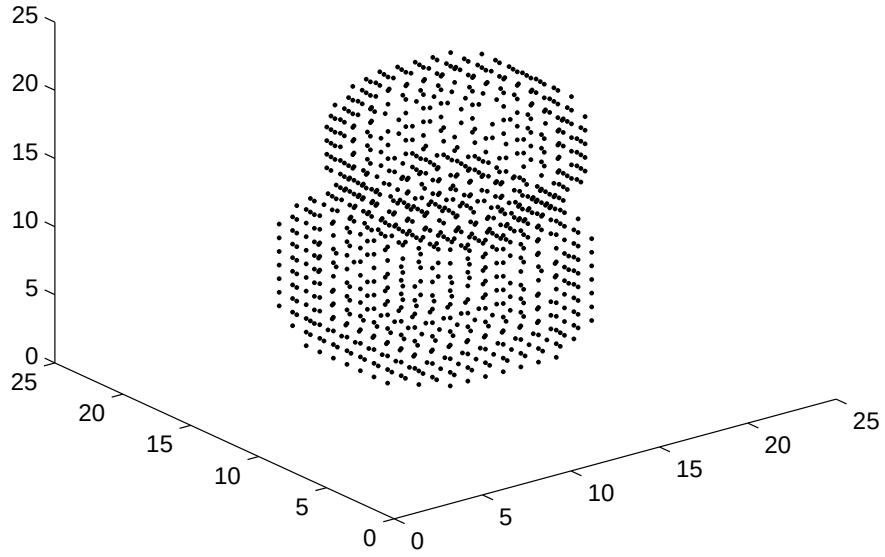


Figure 4.11: 3D Canny edge detector's result. A gradient edge image of the image shown in Figure 4.10, represented by points.

Chapter 5

3D Free Form Method

This method is a generalization of the 2D free form segmentation method. This last method is an extension of snake method, [12]. The improvements of free form method compare to snake method is that we do not need to change the energy function to maintain the set of points to be a curve. Here, the data structure guaranties that. This allows also extension to 3D-case which is very hard for snake method.

Recall that an active surface is constructed with many linked patches, each of them is described with a continuous Bézier surface of bi-degree (D, D), called a 3D free form. In previous chapters, we have already discussed about the geometry, the deformation of free form active surfaces and some results on image segmentation. This chapter will show how to realize our method to solve the shape optimization problem from image segmentation. In implementation, we use closed bicubic piecewise Bézier surfaces as a parametrization of an active surface. The active surface is then deformed to detect the boundaries of regions in the image grace to the gradient edge map of the image. Followings are in details the computation of bicubic Bézier surfaces and description of 3D free form method.

5.1 Bicubic Bézier surfaces

A bicubic Bézier surface is a $(3, 3)$ Bézier surface. This kind of surfaces is determined by a (4×4) matrix of control points. We denote the control polytope by P as below,

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix}.$$

Repeat the computation mentioned in Section 2.2.4, Chapter 2, so that we obtain the matrix form of a bicubic Bézier surface,

$$\begin{aligned} B_2(P; s, t) &= \sum_{j=0}^{3} \sum_{i=0}^{3} P_{ij} b_{i3}(s) b_{j3}(t) \\ &= \sum_{j=0}^{3} \left(\sum_{i=0}^{3} P_{ij} b_{i3}(s) \right) b_{j3}(t) \end{aligned}$$

$$B_2(P;s,t) = \sum_{j=0}^{3} \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_{0j} \\ P_{1j} \\ P_{2j} \\ P_{3j} \end{bmatrix} b_{j3}(t)$$

$$= \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}.$$

In short,

$$B_2(P;s,t) = \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} C \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix},$$

where

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}. \tag{5.1}$$

Suppose that we want to subdivide the surface at the point $s = \frac{1}{2}$. A reparametrization of the matrix equation above given by substituting $\frac{s}{2}$ for s is to represent the first half of the surface,

$$B_2(P;\frac{s}{2},t) = \begin{bmatrix} 1 & \frac{s}{2} & \left(\frac{s}{2}\right)^2 & \left(\frac{s}{2}\right)^3 \end{bmatrix} C \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} CPC^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}.$$

We denote $S_{L_{\frac{1}{2}}}$ the matrix derived from following,

$$S_{L_{\frac{1}{2}}} = C^{-1} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} C$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{16} & 0 & 0 \\ 1 & \frac{1}{3} & \frac{1}{12} & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

that

$$S_{L_{\frac{1}{2}}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix}. \tag{5.2}$$

Then, we obtain

$$B_2(P; \frac{s}{2}, t) = \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} CS_{L_{\frac{1}{2}}} \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}.$$

Hence, the subsurface $B_2(P; \frac{s}{2}, t)$ is also a Bézier surface and its control points is defined by $S_{L_{\frac{1}{2}}} P$,

$$S_{L_{\frac{1}{2}}} P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix}.$$

Now, calculation for the second half of the surface is obtained by the same way. To do this, we reparametrize the surface by substituting $(\frac{1}{2} + \frac{s}{2})$ for s,

$$\begin{aligned} B_2(P; \frac{1}{2} + \frac{s}{2}, t) &= \begin{bmatrix} 1 & \left(\frac{1}{2} + \frac{s}{2}\right) & \left(\frac{1}{2} + \frac{s}{2}\right)^2 & \left(\frac{1}{2} + \frac{s}{2}\right)^3 \end{bmatrix} CPC^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix} \\ &= \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} & \frac{3}{8} \\ 0 & 0 & \frac{1}{4} & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} CPC^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}. \end{aligned}$$

Thus, the second half of the surface is determined by

$$B_2(P; \frac{1}{2} + \frac{s}{2}, t) = \begin{bmatrix} 1 & s & s^2 & s^3 \end{bmatrix} CS_{\frac{1}{2}R} \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} C^T \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix},$$

where $S_{\frac{1}{2}R}$ is given by

$$S_{\frac{1}{2}R} = C^{-1} \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} & \frac{3}{8} \\ 0 & 0 & \frac{1}{4} & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} C = \begin{bmatrix} \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.3}$$

And the control points are defined by $S_{\frac{1}{2}R} P$,

$$S_{\frac{1}{2}R} P = \begin{bmatrix} \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix}.$$

Similarly, a subdivision of the surface can be done with respect to t. The first and second portions of the surface are $PS^T_{L\frac{1}{2}}$ and $PS^T_{\frac{1}{2}R}$. We can combine these two methods to obtain a segmentation of the surface into quarters,

$$\begin{bmatrix} S_{L\frac{1}{2}}PS^T_{L\frac{1}{2}} & S_{L\frac{1}{2}}PS^T_{\frac{1}{2}R} \\ S_{\frac{1}{2}R}PS^T_{L\frac{1}{2}} & S_{\frac{1}{2}R}PS^T_{\frac{1}{2}R} \end{bmatrix}.$$

The first one corresponds to the quarter of $0 \leq s \leq \frac{1}{2}$, $0 \leq t \leq \frac{1}{2}$ and the second is to $0 \leq s \leq \frac{1}{2}$, $\frac{1}{2} \leq t \leq 1$, etc.

As before, the De Casteljau algorithm is used to evaluate a bicubic Bézier surface. Now, we can also use Algorithm 4 with the matrix C given by Equation (5.1) to get a sampling on bicubic surfaces. The matrices $S_{L\frac{1}{2}}$ given by Equation (5.2) and $S_{\frac{1}{2}R}$ given by Equation (5.3) also allow to take subdivisions on bicubic Bézier surfaces by Algorithm 5 and Algorithm 6. A quadrangular subdivision of bicubic Bézier surfaces is then implemented by Algorithm 7.

5.2 3D free form method's applications to Image Segmentation

This section will briefly describe the geometric shape optimization arising in image segmentation. The goal is to minimize an energy functional,

$$E^*_\Gamma = \min_\Gamma \left\{ \iint_\Gamma E_{int}(\Gamma(s,t)) + E_{ext}(\Gamma(s,t))dsdt \right\} \tag{5.4}$$

where $\Gamma(s,t)$ is a parametric active surface. $\Gamma(s,t)$ is a deformable surface. The evolution of the active surface $\Gamma(s,t)$ is constrained by the internal energy $E_{int}(\Gamma)$ and oriented by the external energy $E_{ext}(\Gamma)$.

The internal energy involves to the so-called internal forces. These forces ensure the smoothness and regularity constraints on the surface propagation. By using free form deformation models, the formulation of the geometric shape optimization becomes simplified. Since the regularization constraints are naturally derived from Bézier surfaces, it is indeed not necessary to include them in the formulation. Within free form deformation models, an active surface is described by a net of linked Bézier surface. Each Bézier surface defines a patch of the active surface called free form. Bézier surfaces carry the characteristics of polynomial surfaces and they has several advantages. It is the fact that we obtain an efficient way to evaluate the parametrization of a free form using the De Casteljau algorithm for each patch of the active surface. Moreover, the active surface inherits the smoothness from the regularity of Bézier surfaces, that it is continuous and differentiable almost everywhere (except the set of points lying on the boundaries of patches). Furthermore, the evolution of the active surface is induced by local deformation of free forms which are Bézier surfaces constructing it. Fortunately, a local deformation of Bézier surfaces is efficiently done on their control polytopes. A deformation on control polytopes generates an essential movement of Bézier surfaces needed for the evolution of the active surface toward its destination.

The local deformation of free forms is only derived from a pressure force and a gradient edge map of the image. The pressure force is some type of the so-called external forces. There exist many types of dynamic external forces, [1]. Here, we just focus on the simple one, called "balloon force". It is introduced in the "balloon model",[4]. The balloon model needs a gradient edge map of the image to realize its algorithms. It can be said that the gradient edge map is

a dial for the dynamic behavior of balloon forces. The edge map is defined such that its values is only large in the instant neighborhood of edges, almost zero in homogeneous regions and that the gradient vectors are pointing toward and are normal to the edges. As usual, one uses three-dimensional Canny edge detector to obtain an edge map of a 3D image. The acting of the balloon model is defined by a computational diffusion process, shown in Equation (5.5),[12], in the image so that the boundaries of regions in the image attract free forms. Then, the active free form surface is attracted to edges and stopped.

$$F_{diff} = \frac{1}{1 + F_{edge}} \text{with } F_{edge} = |\nabla G_\sigma * I|^p, p \geq 1 \tag{5.5}$$

where $G_\sigma * I$ is a Gaussian smoothing of image I and $p \in \mathbb{N}$.
A balloon force generates a pressure force that moves the active surface along the normal direction to Bézier surfaces defining active patches. This effect of the balloon force only acts in the homogeneous domains of the gradient edge map. The computation of the normal directions inherits the benefits from the polynomial parametrization of the active surface. The normal vector at a considered surface point is derived from computing the partial derivative of Equation (2.4). This is done only once. Moreover, this computation reduces to compute the derivative of Bernstein polynomials since the control points are constant. The points of the active surface $\Gamma(s,t)$ are then moved to a new position by local movement of its patches. Indeed, this is achieved by the movement of control points of the active Bézier patches. Control points of these patches at the new position obtained by Lemma 4 will define the new position of the entire active surface by Bézier surface parametrization. This process executes Algorithm 8 until convergence. Remark that our process maintain the regularity of the active surface during its evolution. And the smoothness properties of Bézier surface significantly contribute to this effect.

The evolution of the active surface $\Gamma(s,t)$ is indeed done through some selected points and depends on the gradient edge map. When the active surface is deformed to expand, the density of these particular points significantly influences on the convergence of the process. Because these points is used not only to move free forms but also to recognize the boundaries of the regions so as to attract free forms toward them. Moreover, we do not know much information of the regions in the image. Thus, in order to increase the density of these points, more Bézier patches are dynamically added into the active surface during the evolution such that its regularity is maintained. This is realized locally by consider the distance between adjacent points of the free form patches. New patches will be added if the distances are large. Since free forms are surfaces, imagine that they look like rectangles, we have to check the distances in two directions: one along the width and one along the height. In our method, this test is based on the maximal distances separating control points of the same patch in two mentioned directions. They are then compared with a fixed distance threshold. The considered patch will be split into two different patches of the same bi-degree (D, D) as the original one along the associating mentioned direction. This technique is done locally on Bézier patches by Algorithm 5 and Algorithm 6. It is important to note that a splitting occurs at some patch will lead to the splittings on all the patches located on the Bézier constant curve passing through the former, mentioned in Section 2.2.4. This is necessary to guard the regularity of the active surface. The distance test allows to know when a free form patch need split in two ones, during the deformation of the active surface. This geometric constraint on the patch resolution advances the adaptation of the active surface to large or complex shapes of the free spatial boundaries. It is described in Algorithm 9. The complexity of this algorithm is discussed in Lemma 6.

Lemma 6. *The split procedure at a patch requires at most four multiplication of two* $(D+1) \times (D+1)$ *square matrices. So the computational cost of this procedure is in* $\mathcal{O}(D^3)$.

Algorithm 9 Split procedure.

Input: Active surface $\Gamma(s,t)$ with $(M+1)\times(N+1)$ free form patches of degree $(D+D)$ (represented by its control polytope $[P^{ij}]$) and a density threshold ϵ;

Output: New active surface (represented by its control points);

$nCP \leftarrow []$
$mp \leftarrow 0$
for $i \in \{0,...,M\}$ **do**
  for $j \in \{0,...,N\}$ **do**
    $CP \leftarrow P^{ij}$
    if $\max_{k\neq k'} d(P^{ij}_{kl}, P^{ij}_{k'l}) > \epsilon$ **then**
      do Horizontal subdivision on P^{pq} for all $p = i$ by Algorithm 5;
      $CP \leftarrow \begin{bmatrix} P^{i*}_L \\ P^{i*}_R \end{bmatrix}$
      $mp \leftarrow mp + 1$
      break
    end if
    Concatenate nCP and CP;
  end for
end for
$M \leftarrow M + mp$
$NCP \leftarrow []$
$np \leftarrow 0$
for $j \in \{0,...,N\}$ **do**
  for $i \in \{0,...,M\}$ **do**
    $CP \leftarrow nCP^{ij}$
    if $\max_{l\neq l'} d(nCP^{ij}_{kl}, nCP^{ij}_{kl'}) > \epsilon$ **then**
      do Vertical subdivision on nCP^{pq} for all $q = j$ by Algorithm 6;
      $CP \leftarrow \begin{bmatrix} nCP^{*j}_L & nCP^{*j}_R \end{bmatrix}$
      $np \leftarrow np + 1$
      break
    end if
    Concatenate NCP and CP;
  end for
end for
$N \leftarrow N + np$
return NCP (the new control points);

Algorithm 10 summarizes our method using the active surface deformation based on Bézier free forms. At each patch, the computation requires at most the cost in $\mathcal{O}(D^3)$. This is the cost of split and insert algorithm or of interpolation algorithm. Thus, the global cost of the method is bounded by $\mathcal{O}(D^3\mu)$ where $\mu = (M+1) \times (N+1)$ is the number of active patches.. When D is fixed, this cost is linear with respect to the number of patches, shown in Lemma 7. Some experiments with 3D toy images presented in Figures 5.1-5.8 are to say that the method adapts to many complex boundary regions.

Lemma 7. *The global computational cost of the method using the active surface deformation based on Bézier free forms is linear $\mathcal{O}(\mu)$ where $\mu = (M+1) \times (N+1)$ is the number of active patches.*

Algorithm 10 Free form active surface deformation.

Input: Active surface Γ;
Output: Deformed active surface Γ;
  while non convergence **do**
    Split and insert Γ (Algorithm 9);
    for each patch of Γ **do**
      Sample the patch;
      Compute the deformation at each sampled point of the patch;
      Deform the patch;
    end for
  end while
  return Γ;

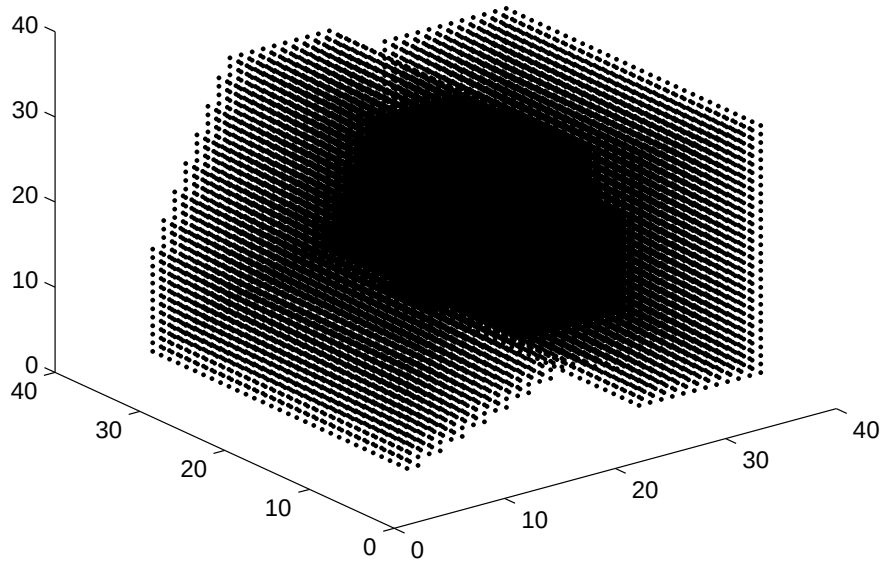


Figure 5.1: A 3D free region.

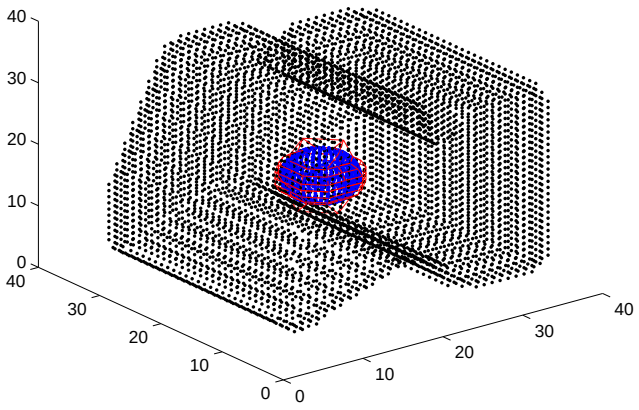


Figure 5.2: Using 3D Canny edge detector to obtain gradient edge map of the image shown in Figure 5.1, and adding initial active surface.

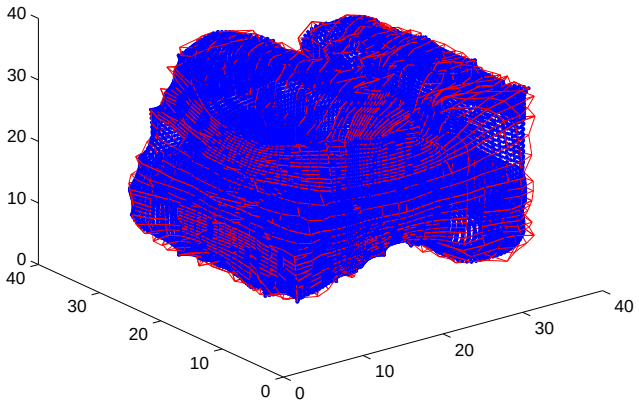


Figure 5.3: Dectected 3D free region.

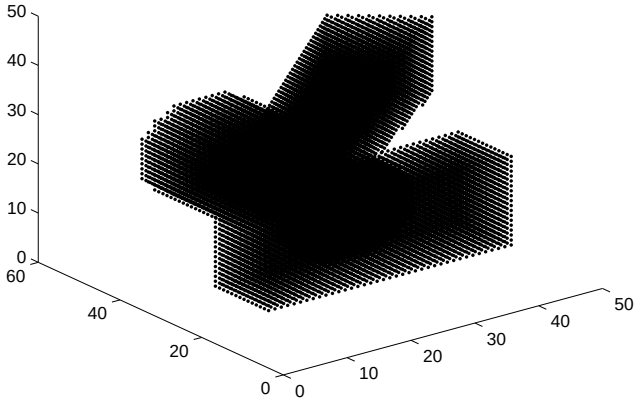


Figure 5.4: A 3D free region with complex boundary.

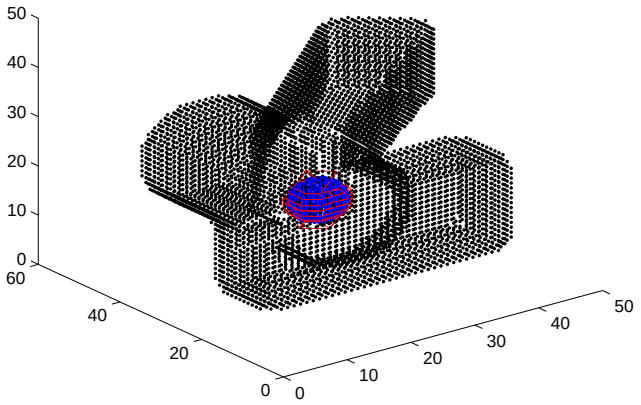


Figure 5.5: Using 3D Canny edge detector to obtain gradient edge map of the image shown in Figure 5.4, and adding initial active surface.

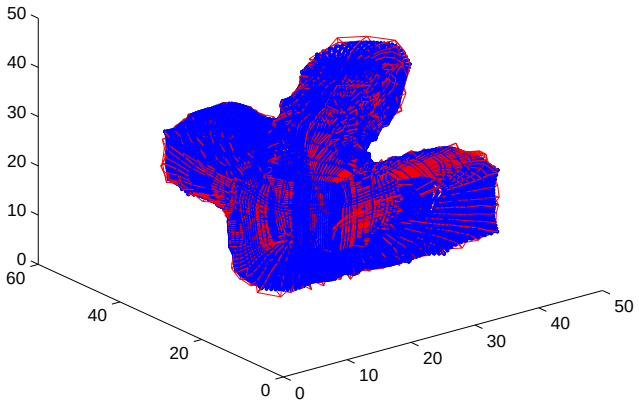


Figure 5.6: Dectected 3D free region of the image shown in Figure 5.4.

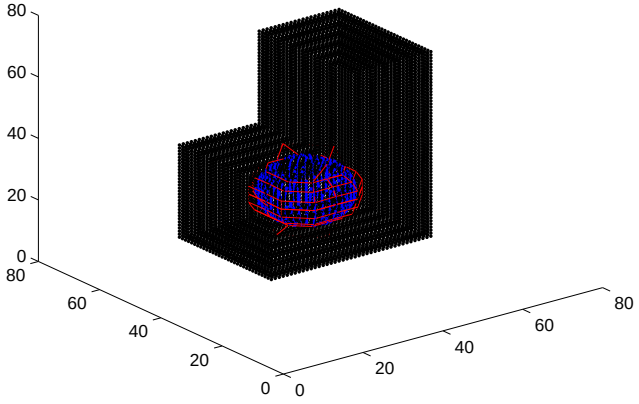


Figure 5.7: An empty box.

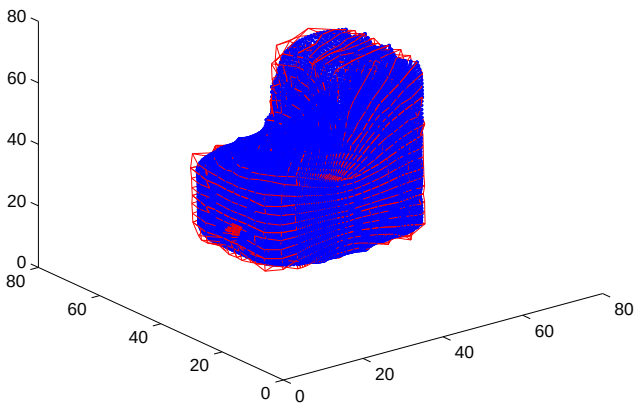


Figure 5.8: An approximation of the empty box shown in Figure 5.7.

Chapter 6

Conclusion

This thesis performed an approach on the geometry and the deformation of shapes represented by uniform piecewise Bézier surfaces. Its results contribute to solving the shape optimization problems arising in 3D images segmentation. This is a method using active surfaces based on free form modeling and deformation with three-dimensional settings. The deformation of active surfaces is controlled by local deformations which are fast and smooth. This 3D free form method can be adapted to complex shapes with convex or concave boundaries.

The complexity of the method is also discussed. The resulting algorithm of 3D free form method requires almost linear computational cost with respect to the size of the output, which is advantageous for real-time image processing and robotic applications.

The 3D free form method is applied to robotics. The deformation of free form active surface allows to detect free spatial regions around a robot so that the robot can move dynamically and autonomously from 3D images of its environment.

An autonomous robot certainly needs to recognize obstacles in the spatial neighborhood environment around it and we give a practical implementation showing the efficiency of the method. Our ongoing work is attracted by this. It requires more techniques to make topological changes on free form active surfaces during their deformations.

List of Symbols

List of Figures

List of Algorithms

References

[1] Chenyang Xu, Jerry L. Prince, *Snakes, Shapes, and Gradient Vector Flow*, IEEE Transactions on Image Processing, vol. 7, no. 3, March 1998.

[2] Gerald Farin, *A History of Curves and Surfaces in CAGD*, Computer Science and Engineering, Arizona State University, Tempe, AZ 85257-5406.

[3] Irwin Sobel, *History and Definition of the Sobel Operator*, 2014.

[4] L. D. Cohen, *On active contour models and balloons*,CVGIP: Image Understanding, vol. 53, no. 2, pp. 211-218, 1991.

[5] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, International Journal of Computer Vision (1988), vol. 1, no. 4, pp. 321-331.

[6] Marek Brejl, Milan Sonka, *Directional 3D Edge Detection in Anisotropic Data*, The University of Iowa.

[7] Matt Young, *The Stone-Weierstrass Theorem*, MATH 328 Notes, Queen's University at Kingston, 2006.

[8] Matt Zucker, Lecture *Basic Filters*, E27, 2013.

[9] Maurer, Michael, *Edge detection in two and three - dimensional images*, 2004.

[10] O. Labbani-Igbida, P. Merveilleux, O. Ruatta, *Free forms for active contours*, preprint.

[11] Olivier Ruatta, *On the Geometry and the Deformation of Shapes Represented by Piecewise Continuous Bzier Curves with Application to Shape Optimization*, LNCS 8085 GSI 2013: 112-119.

[12] Ouiddad Labbani-I., Pauline Merveilleux-O., Olivier Ruatta, *Free Form based active contours for image segmentation and free space perception.*

[13] Patrice Delmas, Lecture *Gaussian filtering*, COMPSCI 373 S1, The University of Auckland, New Zealand, 2010.

[14] Pinar Duygulu, Lecture *Edge Detection*, CS 554 - Computer Vision, Bilkent University.

[15] Prem K Kalra, Lecture *Canny Edge Detection*, CSL783, Indian Institute of Technology, Delhi, India, 2009.