Multivariate Splines for Curve and Surface Interpolation and Fitting

Chongyang Deng* Qingfei Hong† Ming-Jun Lai ‡ Clayton Mersmann§
Yidong Xu¶

January 8, 2020

Abstract

We explain a level curve method to construct 2D smooth interpolatory and/fitting curve from any given 2D data set by using bivariate splines over triangulation. Similarly, we explain a level surface method to construct 3D smooth interpolatory/fitting surfaces from any given 3D point cloud by using trivariate splines over tetrahedralization. The theory and implementation of these splines functions are matured nowadays and hence they can be used for constructing smooth 2D curves and 3D surfaces rather easily as demonstrated in this paper. In addition to the convenience of generating fitting curves and surfaces, one significance of our methods is to use spline functions of high order smoothness to interpolate or fit current G^1 or C^1 surfaces which are already constructed so that one can obtain their counterpart of C^2 or smoother surfaces.

1 Introduction

We are interested in constructing a smooth, e.g. C^2 interpolatory or fitting surface of a given point cloud in the 3D setting. One of our motivations is to be able to connect the wings to the body of airplane in C^2 fashion. The smooth boundary of the exterior domain of the airplane enables smooth air flows. Also the smoother surface of the airplane reduces more air friction and hence produces more fuel efficiency. Mathematically, we would like to have C^2 smooth surfaces to interpolate and/or fit these given data sets in Figure 1.

During last several decades, a lot of theories and computational methods have been proposed to find desired smooth and interpolated curves and surfaces. The literature are too many to survey in this article. A quick and easy on-line research on this topic will be much more convenient to the reader. Typically, one starts with a piecewise linear curve L and construct a smooth curve fitting or interpolating the given points in parametric form (cf. e.g. [6] and [13]). For example, one uses B-spline functions and tensor products of B-splines to construct desired curves and surfaces. Aerospace and car companies adopt the B-spline technology ([6] and [13]), and more generally nonuniform rational B-splines (NURB) to create their desired curves and surfaces due to their nice properties (cf. [25]). The concept of G^1 continuity instead of C^1 continuity for connecting smooth surfaces together was introduced and studied (cf. e.g. [26]). T-splines make B-spline surfaces more convenient to use (cf.[31] and [32]). IGA introduced in [11] combined the surface design and PDE modeling togehter to speed up the computation for engineering experimentation. Poly-splines are recently used to generate 3D

[*] School of Science, Hangzhou Dianzhi University, Xiasha, Hangzhou 310018, China

[†] School of Science, Hangzhou Dianzhi University, Xiasha, Hangzhou 310018, China

[‡] mjlai@uga.edu, Department of Mathematics, University of Georgia, Athens, GA30602

[§] Department of Mathematics, University of South Florida, St. Petersburg, FL33701

[¶] Department of Mathematics, University of Georgia, Athens, GA30602

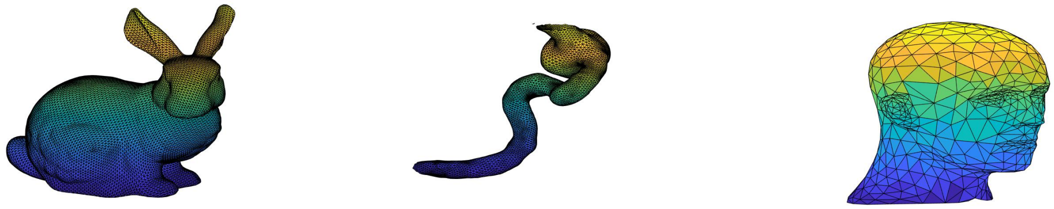

Figure 1: A bunny-like data set (courtesy Meng Mu), an intestine data set (courtesy Bree Ettinger), a human head data set (courtesy Scott Kersey)

surfaces and solids (cf. e.g. [27]). Another popular technique is to use subdivision schemes. One can use interpolatory subdivision schemes, e.g. butterfly scheme to interpolate the given gridded data data points directly or compute a control mesh whose limit surface interpolates the given data points (cf. e.g. [12]). Also, when a space surface, more precisely, a triangulated surface of the given data is centeralizable, or is of star-like shape, one can use spherical splines to interpolate or fit the given data locations. Here a data set is centeralizable or of star-like shape if the interior of the triangulated surface is a star-like domain, i.e. there exists a center $\mathbf{v}_0$ such that all line segment from $\mathbf{v}_0$ to any point on the triangulated surface is completely contained in the closure of the interior of the triangulated surface. We refer to [3], [4], [10], [17], [9] and the references therein. However, these tools are not flexible enough in connection of two smooth surface patches together in a smooth, say C^2 fashion. In particular, a major difficulty is to ensure the water tight and in C^1 or even in C^2 joint fashion between two smooth surface patches over some extraordinary points(EP), construct pants-like surfaces and double-torus like surfaces.

In this paper, we propose to use multivariate spline functions to construct interpolatory surfaces in C^r smooth fashion, where $r \geq 1$. Notice that the theory of multivariate splines and their implementation are matured enough nowadays. We refer to [20] for theory and [5] and [29] for implementation and applications to numerical solution of partial differential equations. Although multivariate spline functions based on triangulation or tetrahedral partition have been used for data fitting which can be viewed as a construction of curve or surface as explained in [5], they have not been used to construct more complicated curves like figure 8 curve, double-torus like surface and arbitrary 3D surfaces as the data sets shown in Figure 1. That is, to the best of the authors' knowledge, the bivariate splines have not been used for constructing smooth 2D closed curves and the trivariate splines have not been used for constructing 3D closed surfaces in the literature so far.

Due to the power of computer nowadays and the effort of the authors of this paper who make the implementation of multivariate splines much efficient and effective to use (cf. [5], [24], [33]), we are able to use them easily for curve and surface construction as demonstrated in this paper. For convenience, we explain our approach in detail. The ideas of the approach discussed in this paper are simple and well-known. Any desired smooth curve can be viewed as a contour or a part of contour of the smooth spline function defined on a 2D polygon including the given data locations and any smooth surface can be treated an iso-surface of a smooth spline function defined over a 3D solid including all given data locations. For example, consider a planar closed curve $C : \{(x(t), y(t)), 0 \leq t \leq 1\}$ first. Let us assume that C is a smooth curve. We can image that there is a function $z = z(x, y)$ such that the level curve $\{(x, y) : z(x, y) = c\}$ is C or a good approximation of C. In practice, we are given a set of points $\mathcal{D} = \{(x_i, y_i), i = 1, \cdots, n\}$ and interested in constructing a C^r smooth curve $S : \{(s_1, s_2)\}$ such that S interpolates the given points $\mathcal{D}$, where $r \geq 1$. Our approach is to start with a polygonal domain Ω, say a rectangle which contains all the given data points in $\mathcal{D}$. For example, one can use an enlarged rectangle of the bounding box of the given data points. If the connectivity L of data points is known, e.g. a piecewise linear interpolation is given. Let $\triangle$ be a constrained triangulation of Ω with the piecewise linear segments of the points in $\mathcal{D}$ being a part of edges of $\triangle$. See Figure 2 for a given piecewise linear curve L in red as well as triangulation with edges along the L.

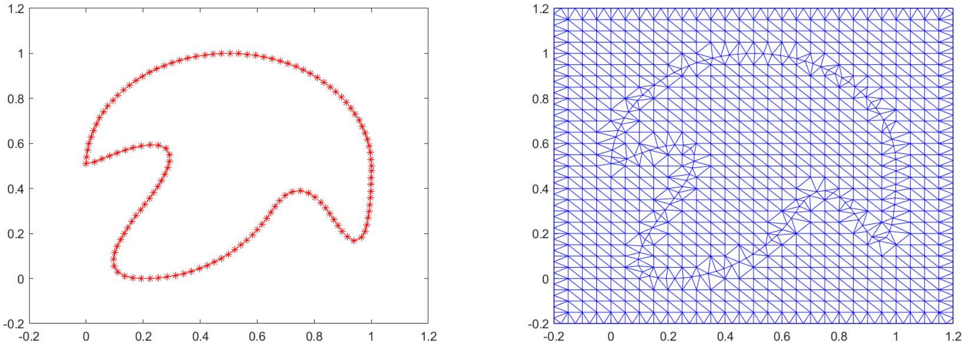

Figure 2: An Example of Piecewise Linear Curve L and a Triangulation

In general, the connectivity of a set of given data points may not be given. In this case, we still let $\triangle$ be a triangulation of Ω with vertices including the given data points. For simple curves, we may even use any regular triangulation such as a type 1 triangulation of the rectangular domain Ω. Let

$$S^r_d(\triangle) = \{s \in C^r(\Omega), s|_T \in \mathbb{P}_d, T \in \triangle\}$$

be the bivariate spline space of smoothness r and degree d, where $d > r$, e.g. $d \geq 3r + 2$. It is known (cf. [5]) one can use them for scattered data fitting and interpolation as well as numerical solution of partial differential equations. For how to use multivariate splines for various partial different equations, we refer the reader to [23], [16], [15], [2], [1], [29], [22], [24], [33], [30] and etc.. Note that our approach proposed in this paper is completely different from the methods described in [5].

Letting $(x_i, y_i), i = 1, \cdots, n$ be points of vertices of $\triangle$ associated with the given piecewise linear curve L, we find a spline S interpolates or fits the data values, say 1 at the given points and additional auxiliary data points in the following senses:

$$S(x_i, y_i) = 1, i = 1, \cdots, n \tag{1}$$

and

$$s(x_j, y_j) = 0, \text{ for some } j \in \mathcal{B} \text{ and } s(x_j, y_j) = 2, \text{ for other } j \in \mathcal{C} \tag{2}$$

where $\mathcal{B}$ and $\mathcal{C}$ are auxiliary points in $\triangle$, $\mathcal{B}$ are near the boundary of the Ω and $\mathcal{C}$ are points near the center of Ω as shown in the left one of Figure 3.

According to the data fitting methods in [5], we use the energy functional

$$\mathcal{E}(s) = \int_\Omega |\frac{\partial^2}{\partial x^2} s|^2 + 2|\frac{\partial^2}{\partial x \partial y} s|^2 + |\frac{\partial^2}{\partial y^2} s|^2 \tag{3}$$

to construct the fitting spline by using the following penalized least square method:

$$\min_{s \in S^r_d(\triangle)} \sum_{i=1}^n |s(x_i, y_i) - 1|^2 + \sum_{i \in \mathcal{B}} |s(x_i, y_i)|^2 + \sum_{i \in \mathcal{C}} |s(x_i, y_i) - 2|^2 + \lambda \mathcal{E}(s) \tag{4}$$

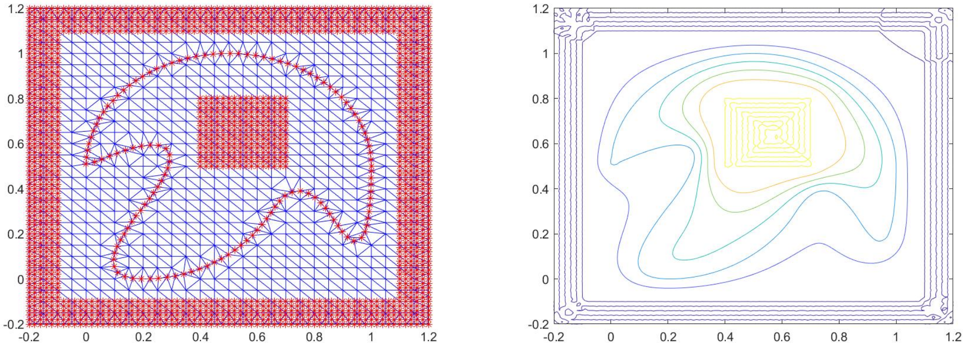

Figure 3: The data fitting locations and the contour curves of the fitting spline including the level curve $\{(x,y) : S(x,y) = 1\}$

where $\lambda > 0$ is a parameter, $r \geq 1$ and $d \geq 3r + 2$. In general, we can add weights to the first three summation terms to combine interpolatory and fitting methods together.

Let us say S is the solution. See, e.g. the contour curves of $S \in S^1_5(\triangle)$ shown in the right one of Figure 3. The desired curve is the level curve $\{(x,y), S(x,y) = 1\}$ shown on the left of Figure 4. An enlarged part of the level curve is shown on the right of Figure 4 to demonstrate the smoothness of the curve. We can see that the level curve is a good C^1 curve passing through the given data points. For more smooth curve, say, C^2 curve, we can use $S^2_8(\triangle)$ to fit the given data points in (1) and (2). Similarly, we can use $S^r_{3r+2}(\triangle)$ to find any fixed smoothness curve passing through the interpolatory points in (1). Indeed, the MATLAB programs developed in [24] and [33] are able to find interpolatory splines in 2D and 3D settings for arbitrary smoothness and arbitrary degree as long as $d > r$ and $S^r_d(\triangle$ is not empty and the memory of a computer large enough. In particular, a domain decomposition method is developed in [33] to construct data fitting/interpolatory spline curves and surfaces.

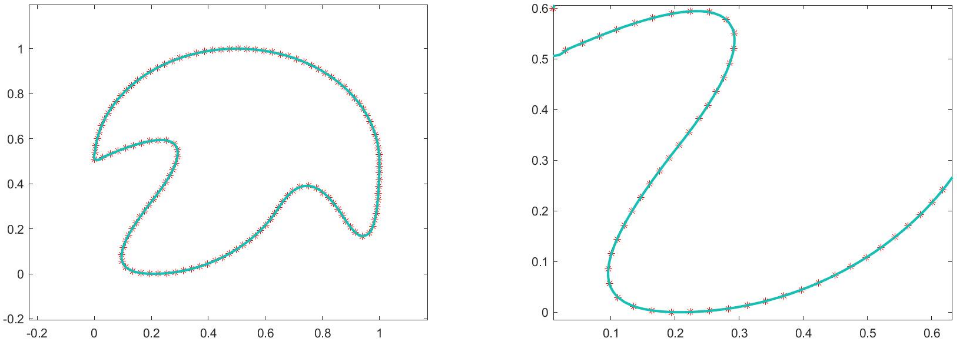

Figure 4: The contour curve $\{(x,y) : S(x,y) = 1\}$ and an enlarged level curve $\{(x,y) : S(x,y) = 1\}$

We shall discuss how to deal with various aspects of curve design in §2 by choosing various purposeful points and values to influence the shape of the interpolatory curves. Also, we shall show that if the given points are from a circle the level curve will reproduce the circle as long as $d \geq 5$ and usnig energy functional $\mathcal{E}_3$ instead of $\mathcal{E}_2$. Similar for any ellipse. Next we shall explain how to handle piecewise smooth curves with corners. More examples will be given in §2 to demonstrate the effectiveness of this level curve method. However, how to design a curve using tension control and/or convexity/concave control is still under investigated.

Similarly, for surface reconstruction from a given data set, e.g. triangular surface in $\mathbb{R}^3$, we assume that $\Omega = [-1, 1]^3$ is an enlarged cube containing the given piecewise linear surface S in $\mathbb{R}^3$. We compute a 3D spline fitting or interpolatory function over Ω and then find the level surface called iso-surface. This approach will produce a smooth desired surface. In particular, if a given surface is G^1, we can use this approach to find a C^1 or C^2 or smoother surface by using an higher order smooth spline to interpolate or fit the data locations obtained from G^1 surface. In §4 and its appendix, we present several examples to show the reproduction of the surface of ball, the connection of pipes in different patterns, and various surfaces from bunny-like data set, human head data set, intestine data set, and coffee mug data set. These examples demonstrate the effectiveness of this level surface method. Finally, we end the this paper by giving several remarks on the open problems of how to handle curves and surfaces with tension control and/or convexity/concave control.

2 Construction of 2D Smooth Curves

When designing a curve, there are many basic requirements for a good software/approach to have. The most common requirements for a curve are

- (1) the smoothness of the curve;
- (2) interpolatory property;
- (3) locality;
- (4) reproduction of quadratic curves such as conics;
- (5) how to handle corners
- (6) monotonicity preserving property;
- (7) convex/concavity preserving property;
- (8) etc..

It is easy to understand that our bivariate spline method for level curves satisfies the properties (1), (2), (3). Indeed, we can have C^r curves for $r = 1, 2, 3$ and etc.. if we use spline space $S^r_{3r+2}(\triangle)$ for an appropriate degree d. We can ensure the interpolatory property. As spline functions have a set of locally supported basis functions when $d \geq 3r + 2$, any local change will not affect to the curve globally.

We next discuss how to reproduce circles/ellipse. The data fitting method in (4) can reproduce linear polynomials, but not be able to reproduce quadratic polynomials. To overcome this difficulty, we have to use an higher order energy functional $\mathcal{E}_3$.

$$\mathcal{E}_3(s) = \sum_{i+j=3} \int_\Omega |\frac{\partial^3}{\partial x^i \partial y^j} s|^2 dxdy \tag{5}$$

and consider the minimization:

$$\min_{s\in S^r_d(\triangle)} \sum_{i=1}^n |s(x_i, y_i) - 1|^2 + \sum_{i\in\mathcal{B}} |s(x_i, y_i)|^2 + \sum_{i\in\mathcal{C}} |s(x_i, y_i) - 2|^2 + \lambda\mathcal{E}_3(s). \tag{6}$$

To ensure the interpolatory condition of the given data locations, we may add the weights to the first term of the functional above. That is, we consider

$$\min_{s\in S_d^r(\triangle)} \sum_{i=1}^{n} w_{1,i}|s(x_i,y_i)-1|^2 + \sum_{i\in\mathcal{B}} w_{2,i}|s(x_i,y_i)|^2 + \sum_{i\in\mathcal{C}} w_{3,i}|s(x_i,y_i)-2|^2 + \lambda\mathcal{E}_3(s), \tag{7}$$

where $w_{1,i}, w_{2,i}, w_{3,i} > 0$ for all i. For example, to interpolate at those (x_i, y_i), we place a large weight $w_{1,i}$ while for other (x_i, y_i), we use a smaller value for $w_{2,i}$ and $w_{3,i}$. Under this new minimization, the penalized least square fitting method in (7) will be able to reproduce quadratic polynomials and hence, reproduce the circle or ellipse from the piecewise linear interpolation of the given points which are from a circle or ellipse if the number of the given data points is enough.

Next we consider the situation that the desired curve is not smooth overall. It has a few corners. See Figure 5 and 5 as the spline fitting curve is smooth and simply use this approach without any modification will not produce a sharp corner. To overcome this difficulty, we can create some holes inside the domain of interest at the corners so that the minimization will skip the spline functions inside the holes and hence, create the desired sharp corners.

Example 2.1. In this example, we show how to handle the corners of a curve. First let use use the level curve method directly to generate a smooth curve. See several corners are not sharp enough as we use smooth spline functions in Figure 5.

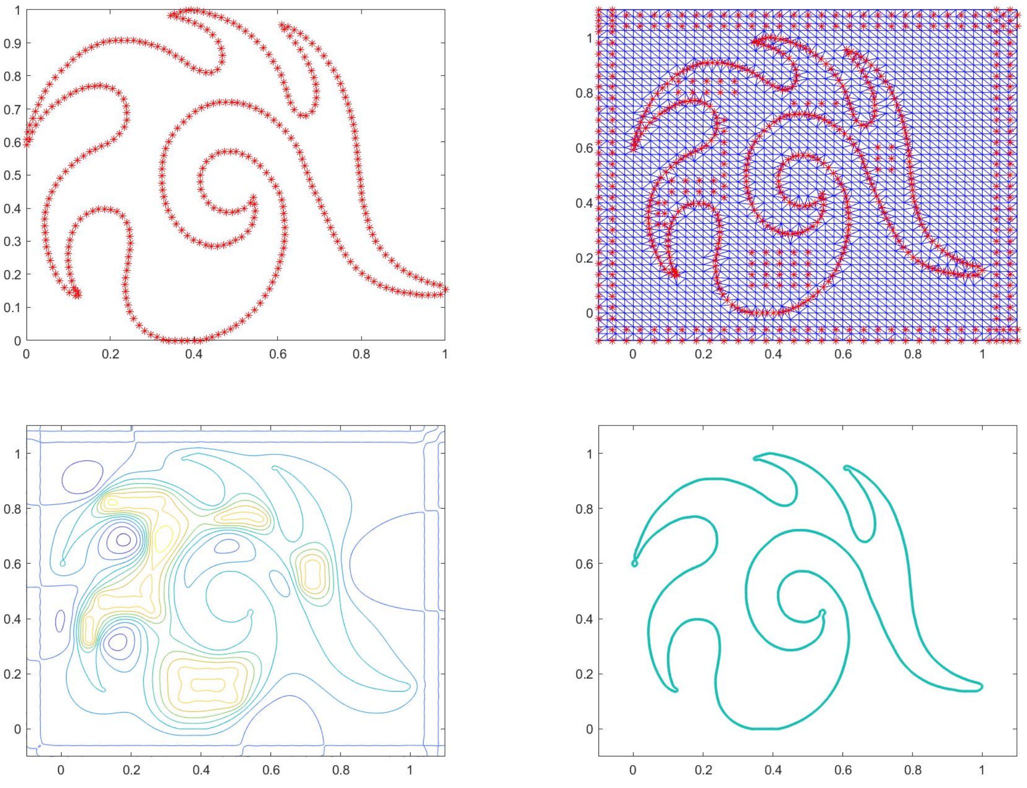

Figure 5: Top row: the given data locations and a triangulation with given locations with auxiliary points; Bottom row: a contour of the spline fitting based on our approach and the level curve which interpolates the given data locations.

We now modify our approach by creating some holes in the triangulation at the corners of the curve.

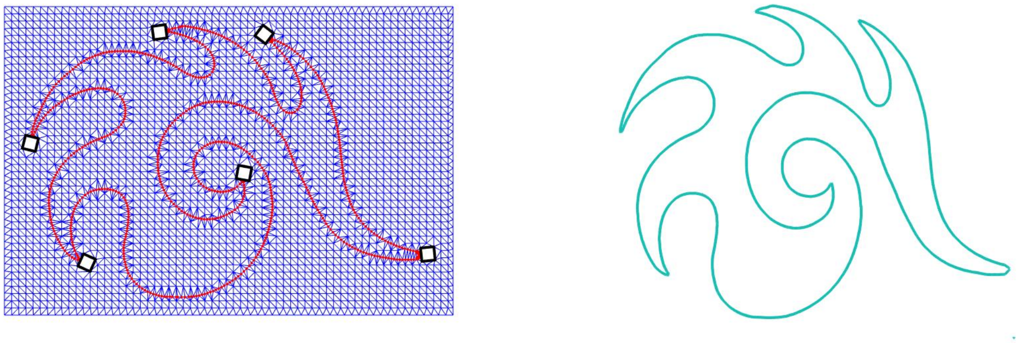

Figure 6: Top row: a triangulation with holes and the level curve with corners; Bottom row: magnified corners

In Figure 7, we can see the smooth tips of the curves in the last row in Figure 5 which can be compared with the tips of the curves in Figure 6. The tips in Figure 6 are much sharper.

Next we show various points in designing curves in the following examples.

Example 2.2. We are given a piecewise linear curve as shown in the left of Figure 8. We find a constrained triangulation $\triangle$ with the linear curve L as some edges of $\triangle$. There is a portion of points which forms a cluster. Our method is able to follow the given piecewise linear curve L to find a smooth curve due to the constrained triangulation. See Figure 9.

We can see that the curve in middle section is near to touch itself and the given data locations in the area are very close to themselves. Our spline method is able to find the desired curve. In general, because we are able to choose a good triangulation as fine as possible to separate the points and we can even add auxiliary points in between with values > 1 to separate the curve.

Example 2.3 (Self-Intersected Curve)**.** In this example, we will show that our construction method of smooth curve is able to find a curve which is self-intersected. Our approach can still work.

Let us enlarge the cross point to see the curve more closely. We can see that the curve of figure 8 intersects itself sharply.

Example 2.4. In this example, we discuss how to make a change of the curve. We start with a circle like curve. In this example, we do not use the constrained triangulation. Instead, we simply use the standard triangulation. With value zero at the points around the boundary of the domain, and 1 at the locations on a circle and 2 in the middle of the domain, we obtain a circle like curve. Next we start to make a change of the circle. We add a new point shown in blue and choose 1.5 to push the circle like curve to make a change.

We can see that the value at the auxiliary point and the location of the auxiliary point make a significant change of the curve. In this way, we can design and modify a curve to our taste.

Finally in this section, let us look at another situation that multiple smooth curves which need to be expressed. This is an example which is close to a real life application which is included to demonstrate the performance of our approach. We show that our approach can find multiple curves at the same time.

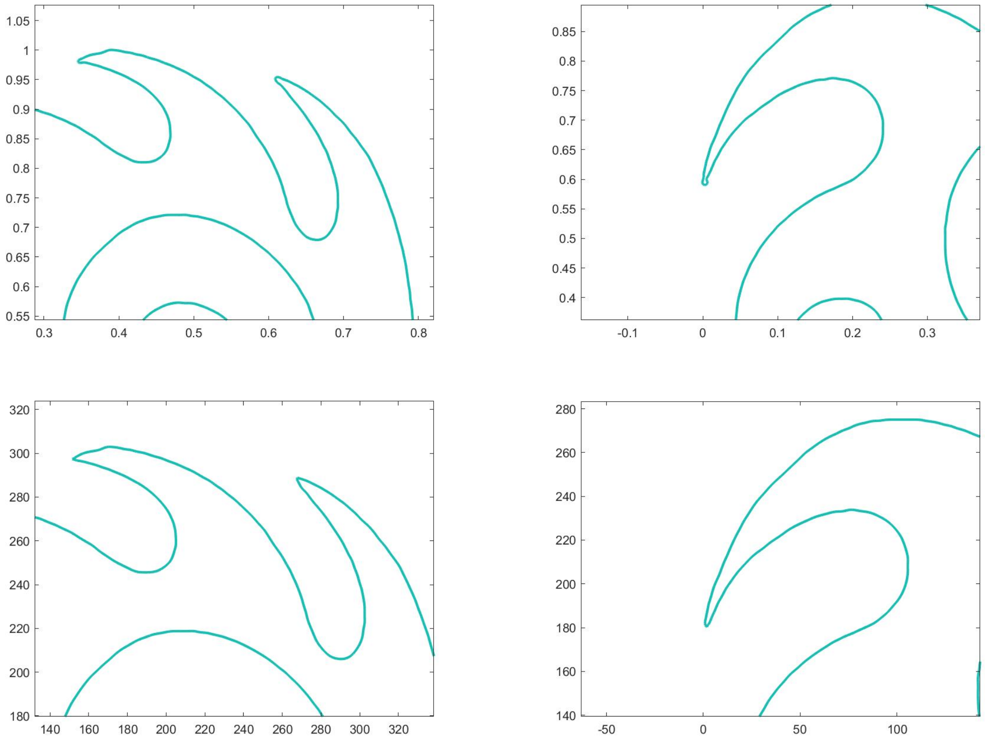

Figure 7: Top row: smooth tips of some corners of curves of Figure 5; Bottom row: tips of some corners of curves of Figure 6

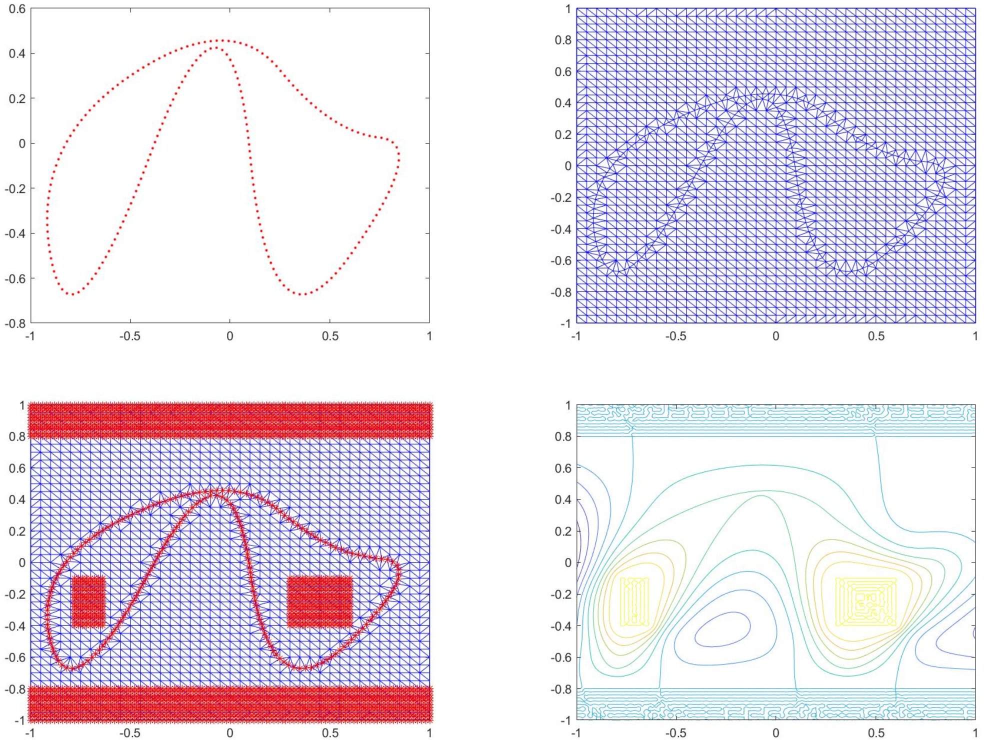

Figure 8: Top row: another set of point cloud and a constrained triangulation; Bottom row: The data fitting locations and the contour curves of the fitting spline including the level curve $\{(x, y) : S(x, y) = 1\}$

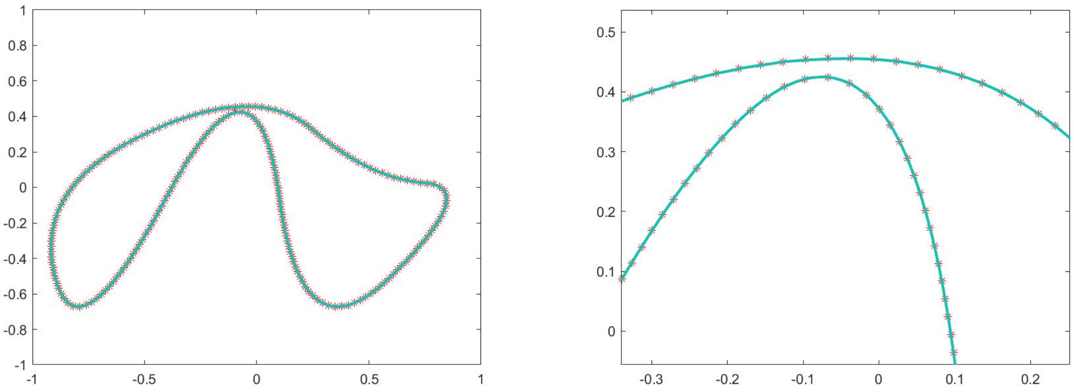

Figure 9: The contour curve $\{(x, y) : S(x, y) = 1\}$ and an enlarged level curve $\{(x, y) : S(x, y) = 1\}$

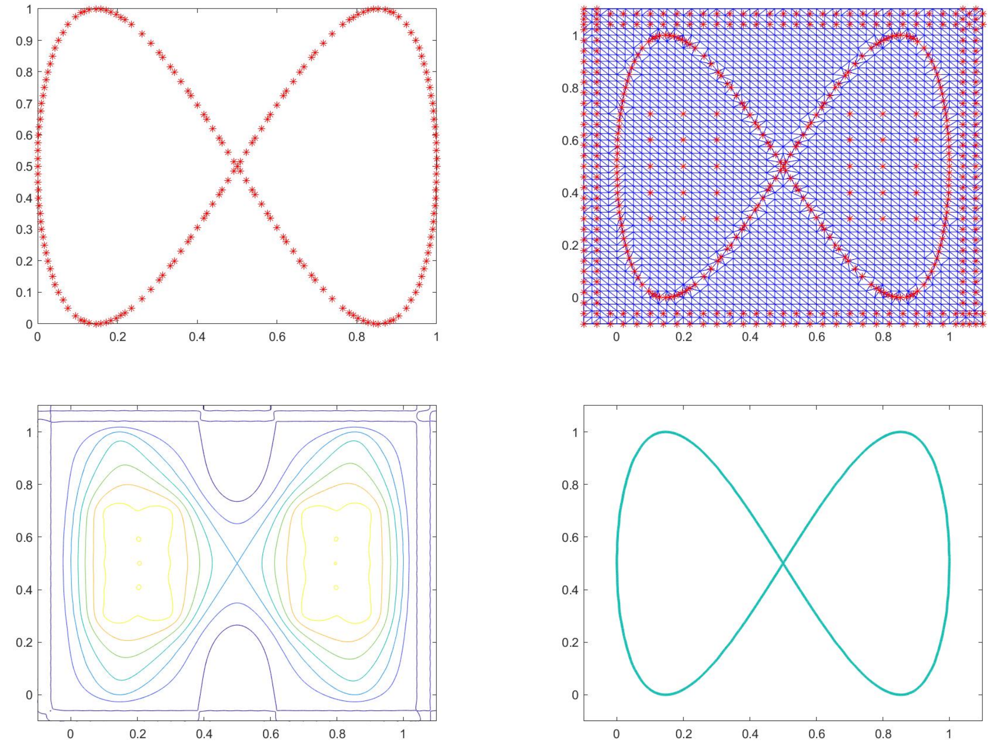

Figure 10: A contour of the spline fitting based on our approach and the level curve which interpolates the given data locations

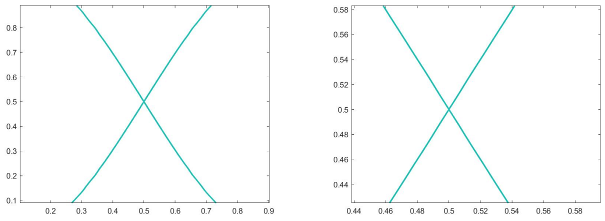

Figure 11: Two enlarged areas of cross point

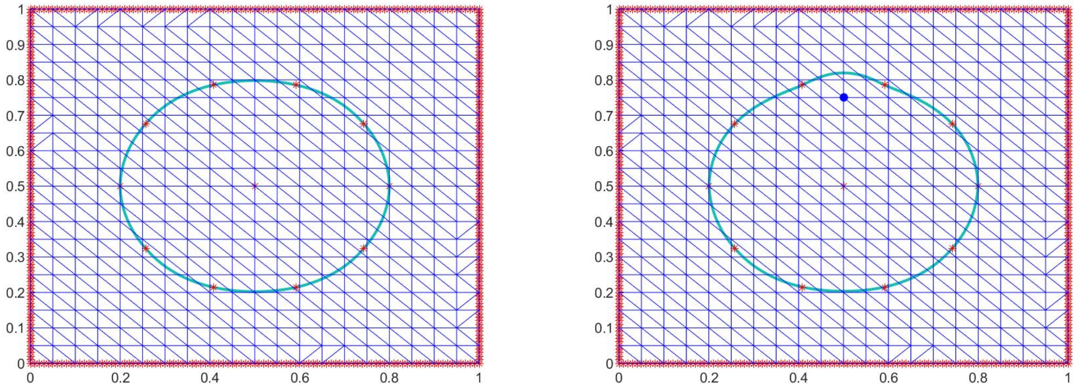

Figure 12: The level curve L shown in blue from the a few give locations (in red) and a new curve based on the red points and a blue point (auxiliary point) which has a value 1.5.

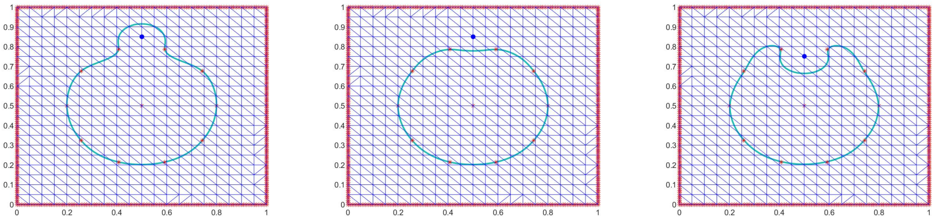

Figure 13: The level curves from the red points together with the blue point (auxiliary points) with various values

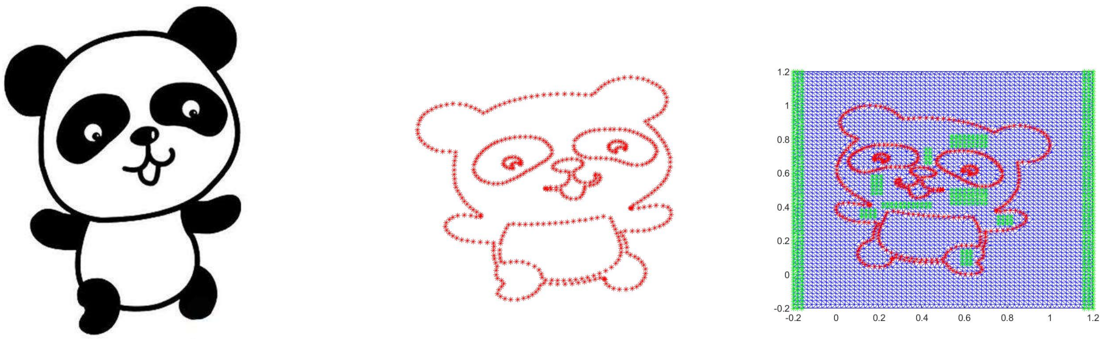

Figure 14: A cartoon panda, the data locations, and data locations with triangulation.

Example 2.5. Consider a small panda as shown in Figure 14 together with data locations.

Our approach produces smooth curves shown in Figure 15. It is easy to see that this approach works well. More realistic designs of various curves close to real life can be found in [7].

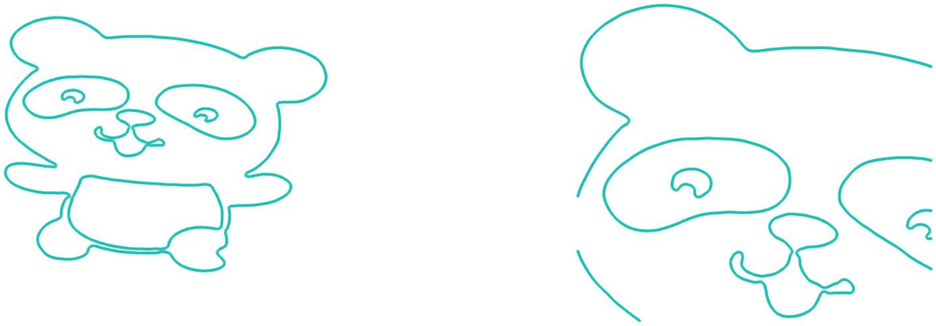

Figure 15: Our smooth spline level curves and an enlargement of a part of these curves

3 Construction of 3D Smooth Surfaces

In this section, we extend the construction of curves in the previous section to the construction of smooth surfaces in 3D. Letting $\mathcal{D} = \{(x_i, y_i, z_i), i = 1, \cdots, N\}$ be the given vertices in $\mathbb{R}^3$, we choose a cube which is an enlarged bounding cube of the given points in $\mathcal{D}$. We let $\triangle$ be a constrained tetrahedral partition of Ω with each data locations being vertices of $\triangle$. If a triangular surface of the given data is given, we let $\triangle$ be a constrained tetrahedral partition with each triangle in the triangular face $\mathcal{T}$ being a triangular face of $\triangle$. Although these tetrahedral partitions make spline interpolation easier, we can use any regular tetrahedral partition $\triangle$ of the cube if the targeted surface is simple enough. That is, we simply use a delaunay tetrahedron $\triangle$ based on equally-spaced vertices of the cube to have a tetrahedral partition of the cube.

For a tetrahedral partition $\triangle$ in $\mathbb{R}^3$, let

$$S^r_d(\triangle) = \{s \in C^r(\Omega), s|_T \in \mathbb{P}_d, T \in \triangle\}$$

be the trivariate spline space of smoothness r and degree d, where $d > r$, e.g. $d \geq 6r + 3$. It is known one can use them for scattered data fitting and interpolation as well as numerical solution of partial differential equations. We refer the reader to [5] for how to use trivariate splines for various computation. In particular, we are able to speed up the computation with trivariate splines for numerical solution of partial differential equations as in [24] and data fitting methods in [33]. We use the trivariate spline packages developed in [24] and [33] for data interpolating and fitting to find a spline function which satisfying $S(x_i, y_i, z_i) = 1$ for all $i = 1, \cdots, N$ and $S(\bar{x}_j, \bar{y}_j, \bar{z}_j) = 0$ for $j = 1, \cdots, M_1$ and $S(\hat{x}_k, \hat{y}_k, \hat{z}_k) = 2$ for $k = 1, \cdots, M_2$ for auxiliary points $(\bar{x}_j, \bar{y}_j, \bar{z}_j)$ and $(\hat{x}_k, \hat{y}_k, \hat{z}_k)$. All ideas of constructing 3D surfaces are the same as the constructing curves in the previous section. We only need to show some surfaces to demonstrate the method in the 3D setting.

Example 3.1. For simplicity, let us use a spherical surface to explain how we do in the 3D setting. See Figure 16 (left), where we find a tetrahedral partition enclosed the given data of a ball. In Figure 16 (right), we use the auxiliary data around the ball. For auxiliary points inside of the ball, see Figure 17 (left). By our spline data fitting method, we find the isosurface at level $L = 1$. See the graph in Figure 18, where the spline space of $d = 6$ and $r = 1$ over the tetrahedral partition was used. One reason that we are able to use degree 6 spline to achieve C^1 surface is that there is such a spline function which can reproduce the surface of the ball. From the graphs (in particular, the graph on the right of Figure 18, we can see that the iso-surface is very smooth as it is indeed the surface of the ball. We used the energy functional $\mathcal{E}_3$, the third order functional instead of the standard second order energy in our data fitting method to achieve the reproduction of the quadratic polynomial surface.

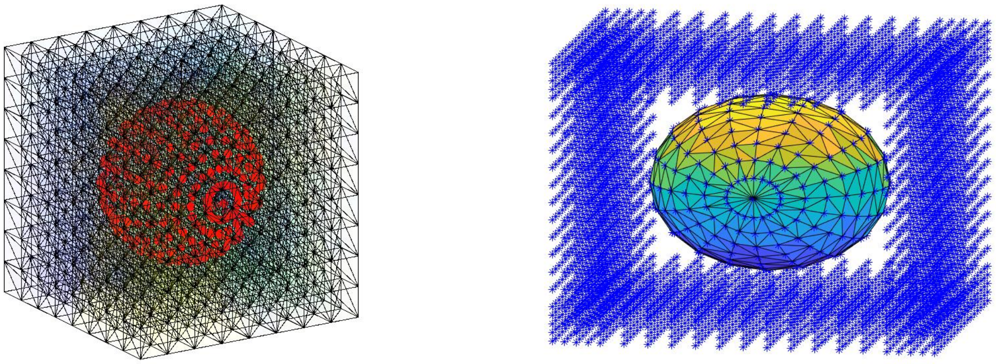

Figure 16: A tetrahedral partition with the vertices in $\mathcal{D}$ shown in red and auxiliary points in blue.

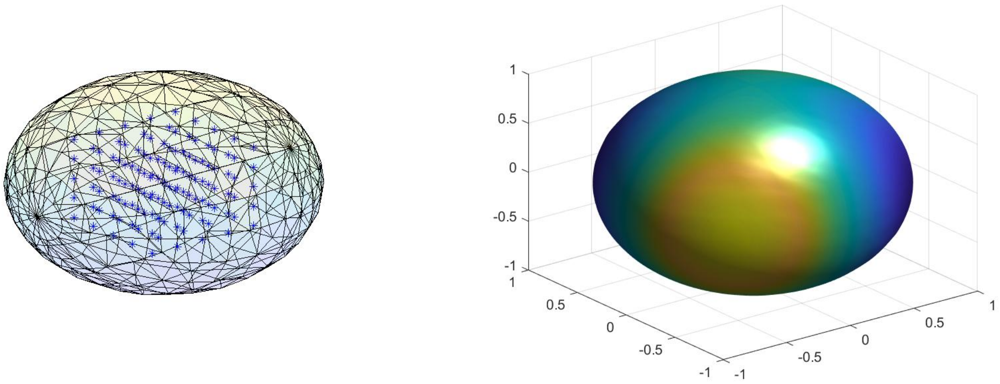

Figure 17: Additional (auxiliary) points inside of the ball and the level $L = 1$ isosurface using E_2.

Note that we did not use a constrained tetrahedral partition in the construction above for this simple surface. That is, we did not use a triangular surface of the ball to guide the construction of this spline iso-surface as the surface of ball is very simply. When a surface is complicated, e.g. a surface almost touching itself, a guiding triangular surface may be needed and more complicated tetrahedral partition will be useful.

To demonstrate the convenience of using this approach to construct various surfaces, we shall generate more surfaces of various kinds. Let us start with a double torus. For a double torus with the same radii for both large circles and the same radii for the smaller circle, we refer to [33]. In the following example, we present a double torus with different radii.

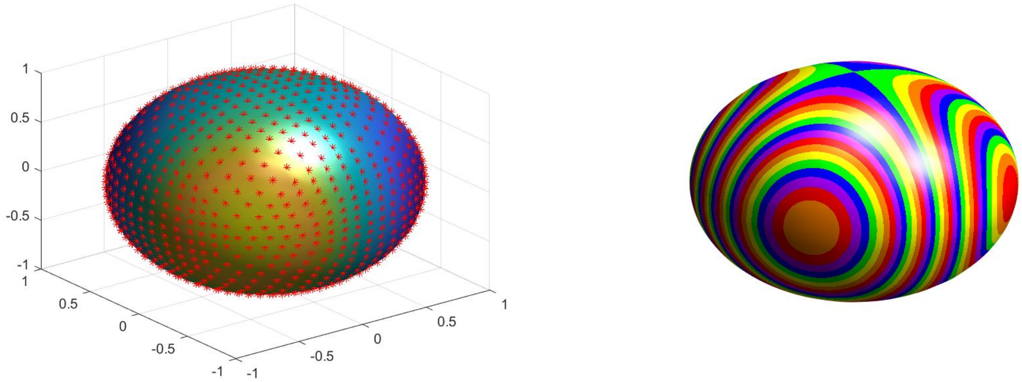

Figure 18: The iso-surface with given data locations (left) and the iso-surface with rainbow light on(right). This iso-surface is generated by using E_3.

Example 3.2 (Double Torus)**.** In Figure 19, we simply present the surface constructed by using the 3D splines.

Example 3.3. We are given 9907 locations from the surface of 3 pipes connected together which was generated in [8] by a subdivision scheme. We choose a standard cube containing all the data locations and find a tetrahedral partition $\triangle$ as shown in Figure 20 (right). $\triangle$ has 3703 tetrahedra. We use the trivariate spline space of $S_6^1(\triangle)$ to find an interpolatory spline with 1 at the given data locations and other values at two groups of auxiliary data locations. More precisely, we use zero for the green data locations and 3 for the red data locations as shown in Figure 21. We can see the isosurface of the interpolatory spline when level value equal to 1. To see all the given data locations are on the spline surface, we present a graph on the left of Figure 22.

In Appendix, we shall present more examples to convince the reader that the approach discussed in this paper is very convenient to use.

4 Conclusions and Remarks

In this paper, we proposed a numerical method to compute smooth interpolatory curves and surfaces. One of the advantages of this approach is that it can generate smooth interpolatory curves and surfaces rather easily. In addition, it can find multiple curves and surfaces simultaneously. See, e.g. Example 2.5. Although the method give us satisfactory results, we have a couple of remarks on this method in order.

Remark 4.1. We used a rectangular domain containing the curves to do interpolation and fitting. This way of construction is just for convenience, not necessary. One can certainly use a smaller non-rectangular domain containing the curves of interest which will enable to save computational time and storage of spline coefficients. Similar to the 3D setting to construct an isosurface to interpolate/fitting the given data.

Remark 4.2. For large problems such as surfaces with high smoothness, we have to use spline functions of large degree. This will lead to an intensive computational problem. One way to do is to use a domain decomposition method for finding a good data fitting splines in the 2D and 3D settings. For example, we can extend the DDM method for data fitting explained in [21] to the 3D

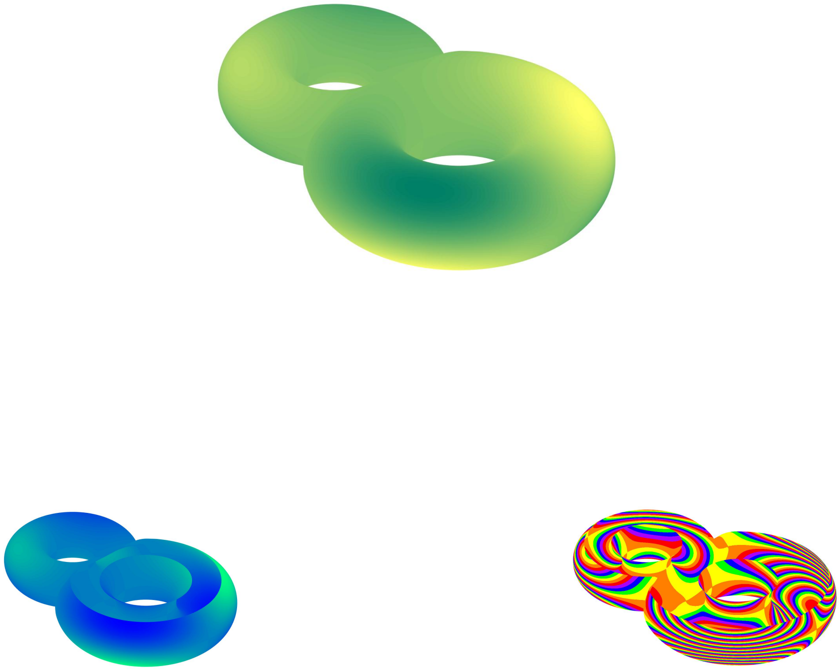

Figure 19: The double torus surface (top), the surface with a cut at the bigger torus to see the inside of the surface (low-left), and the half surface of the double torus with rainbow light on(low-right).

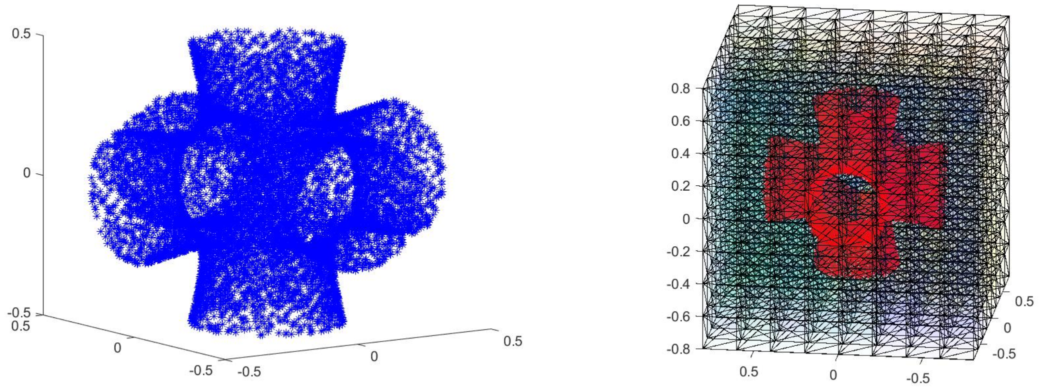

Figure 20: A given set of scattered vertices in $\mathcal{D}$ shown in the left and a tetrahedral partition of the cube containing the given data locations

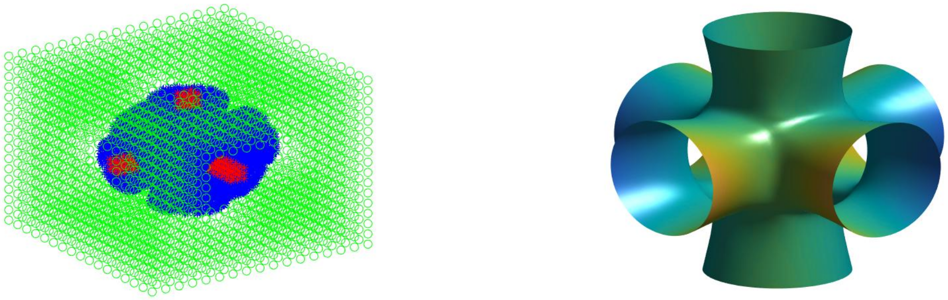

Figure 21: Data interpolation locations (various values for various colors) and the spline isosurface

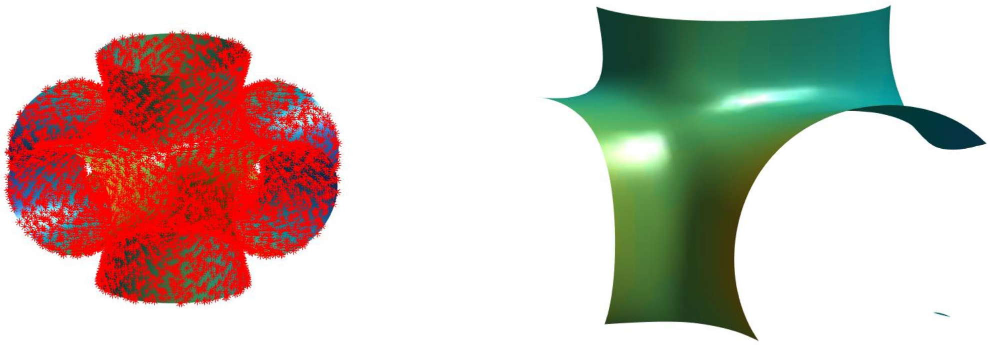

Figure 22: The spline isosurface with given data locations (left) and an enlarged isosurface to see that the surface is very smooth

setting. For another example, in [33], a randomized DDM method for data fitting and numerical solution of Poisson equations are explained based on bivariate and trivariate splines. We leave the detail to a future publication.

Remark 4.3. We can extend the construction of a piecewise smooth curve with corners to the construction of piecewise smooth surfaces with corners. However, the nonsmoothness in the 3D surface is more complicated in general. For example, we need to construct surfaces with ridges, e.g. the rim of coffee mug. As we use smooth spline functions, the corners are smoothed and the rim of mug is not sharp enough which can be seen from Figure 27 given in Appendix. In order to have the sharpness, we can choose a hole at the rim which has to be chosen carefully. More study will be needed.

Remark 4.4. We have shown how to do self-intersection, reproduction of quadratic circle, and use various values at auxiliary points to adjust the shape of curves. Similar things can be done in the surface setting. The details are left to the interested reader. However, we do not know how to use this method to construct convex curves or tension splines.

Remark 4.5. We can compute the curvature of the spline surfaces. It is interesting to use these curvature information to help design surfaces. For example, instead of the energy functional in (4), we can use the mean curvature or Gauss curvature functional to find interpolatory surfaces with minimal mean curvature or minimal Gauss curvature. We leave this topic to a future research.

Remark 4.6. It is interesting to know how to use this approach for curve/surface deformation. We again leave it to a future research.

Remark 4.7. Finally, let us point out that we are not able to construct 3D curves using this approach. We leave it to the interested reader.

Acknowledgment: The authors would like to thank Larry Schumaker for providing nurb curve data sets we used in this paper. Also, they would like to thank Meng Wu for providing a data set of bunny for us to test our methods. Finally, Ming-Jun Lai would like to thank Prof. Thomas Hughes of University of Texas at Austin for the generosity of his time in the summer of 2017 for discussing 3D data fitting and surface generating problems such as surfaces at extraordinary points and pants-like surface.

References

[1] M. Ainsworth, O. Davydov, and L. L. Schumaker, Bernstein-Bézier finite elements on tetrahedral-hexahedral-pyramidal partitions. Comput. Methods Appl. Mech. Engrg. 304 (2016), 140–170.

[2] M. Ainsworth, G. Andriamaro, and O. Davydov, A Bernstein-Bézier basis for arbitrary order Raviart-Thomas finite elements. Constr. Approx. 41 (2015), no. 1, 1–22.

[3] P. Alfeld, M. Neamtu, and L. L. Schumaker, Bernstein-Bézier polynomials on spheres and sphere-like surfaces, Comput. Aided Geom. Design, 13 (1996), pp. 333–349.

[4] P. Alfeld, M. Neamtu, and L. L. Schumaker, Fitting scattered data on sphere-like surfaces using spherical splines, J. Comput. Appl. Math., 73 (1996), pp. 5–43.

[5] G. Awanou, M. -J. Lai, P. Wenston, The multivariate spline method for scattered data fitting and numerical solutions of partial differential equations, Wavelets and splines: Athens (2005), 24–74.

[6] C. de Boor, A Practical Guide to Splines, Springer, 1977.

[7] C. Deng, Q. Hong, and M. -J. Lai, Bivariate spline method for curve deformation, under preparation, 2019.

[8] C. Deng and W. Ma, A unified interpolatory subdivision scheme for quadrilateral meshes, ACM Transaction on Graphs, vol. 32 (2013), article 23.

[9] Baramidze, V. and Lai, M. -J., Nonnegative Data Interpolation by Spherical Splines, J. Applied and Comput. Math., vol. 342 (2018) pp. 463–477

[10] Baramidze, V., Lai, M. -J. and Shum, C. K., Spherical Splines for Data Interpolation and Fitting, SIAM Journal of Scientific Computing, vol. 28 (2006) pp. 241–259.

[11] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. Isogeometric analysis: Toward Integration of CAD and FEA. Wiley, Chichester,, 2009.

[12] N. Dyn, D. Levin, and J. A. Gregory, A butterfly subdivision scheme for surface interpolation with tension control. ACM Trans. Graph. 9 (1990), 160–169.

[13] G. Farin, Curves and Surfaces for Computer-Aided Geometric Design, A Practical Guide, Academic Press, 1992.

[14] M. Floater and Lai, M.-J., Polygonal spline spaces and the numerical solution of the Poisson equation, SIAM J. Numer. Anal. 54, 797–824 (2016).

[15] J. Gutierrez, M.-J. Lai, G. Slavov: Bivariate spline solution of time dependent nonlinear PDE for a population density over irregular domains. Math. Biosci. 270, 263–277 (2015)

[16] X. Hu, Han, D. and Lai, M. -J., Bivariate Splines of Various Degrees for Numerical Solution of PDE, SIAM Journal of Scientific Computing, vol. 29 (2007) pp. 1338–1354.

[17] M. -J. Lai, Shum, C. K., Baramidze, V. and Wenston, P., Triangulated Spherical Splines for Geopotential Reconstruction, Journal of Geodesy, vol. 83 (2009) pp. 695–708.

[18] M. -J., Lai, and Lanterman, J., A polygonal spline method for general 2nd order elliptic equations and its applications, in Approximation Theory XV: San Antonio, 2016, Springer Verlag, (2017) edited by G. Fasshauer and L. L. Schumaker, pp. 119–154.

[19] M. -J. Lai and C. Mersmann, Bivariate Splines for Numerical Solution of Helmholtz Equation with Large Wave Number, submitted, 2019.

[20] M. -J. Lai and L. L. Schumaker, Spline Functions over Triangulations, Cambridge University Press, 2007.

[21] M. -J. Lai and L. L. Schumaker, Domain Decomposition Method for Scattered Data Fitting, SIAM Journal on Numerical Analysis, vol. 47 (2009) pp. 911–928.

[22] M. -J. Lai and C. Wang, A Bivariate Spline Method for Second Order Elliptic Equations in Non-Divergence Form, Journal of Scientific Computing, (2018) pp. 803–829.

[23] M. -J. Lai and Wenston, P., Bivariate Splines for Fluid Flows, Computers and Fluids, vol. 33 (2004) pp. 1047–1073.

[24] C. Mersmann, Numerical Solution of Helmholtz equation and Maxwell equations, Ph.D. Dissertation, 2019, University of Georgia, Athens, GA.

[25] L. Piegl and W. Tiller, The NURBS book. 2nd ed. Springer-Verlag, New York, USA. 1997.

[26] Micchelli, Charles A., Mathematical aspects of geometric modeling, SIAM Publication, Philadelphia, PA, 1995.

[27] T. Schneider, J. Dumas, X. Gao, M. Botsch, D. Panozzo, and D. Zorin, Poly-Spline Finite Element Method, ACM Transactions on Graphics, vol. 38(2019), Article no. 19.

[28] L. L. Schumaker, Spline Functions, Academic Press, 1978.

[29] L. L. Schumaker, Spline Functions: Computational Methods, SIAM Publication, 2015.

[30] L. L. Schumaker, Solving Elliptic PDE's on Domains with Curved Boundaries with an Immersed Penalized Boundary Method, J. Scientific Computing, online on May 24, 2019.

[31] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCSs. ACM Transactions on Graphics, 22 (3):477–484, 2003.

[32] T. W. Sederberg, D. L. Cardon, G. T. Finnigan, N. S. North, J. Zheng, and T. Lyche. T-spline simplification and local refinement. ACM Transactions on Graphics, 23 (3):276–283, 2004.

[33] Y. D. Xu, Multivariate Splines for Scattered Data Fitting. Eigenvalue Problems, and Numerical Solution to Poisson equations, Ph.D. Dissertation, 2019, University of Georgia, Athens, GA.

5 Appendix

In this section, we collect more examples to convince the reader that the approach discussed in this paper is very convenient to use. We are able to generate a lot of figures in a very short of time.

Example 5.1. Instead of 9907 data locations in Example 3.3, we now show that only a few data locations enable us to generate pipe connection surfaces as shown Figures 23 and 24.

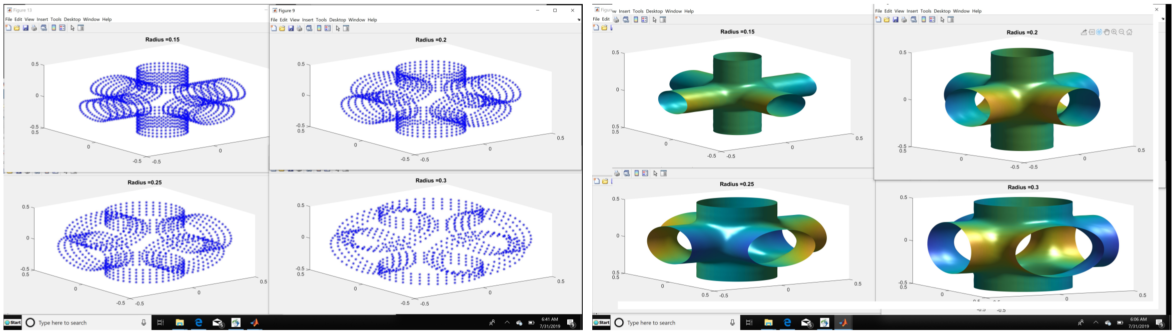

Figure 23: The spline isosurfaces(right) with pipes of various sizes and the associated given data locations (left)

Example 5.2. Next let us work on more realistic examples. We aim to produce a C^1 surface fitting the data from a bunny as shown in Figure 25. We first find a tetrahedral partition around the bunny data locations shown in Figure 25 (left). Then we choose the additional data locations inside the body of the bunny as well as the auxiliary data locations near the boundary of the bounding box. We use $S_8^1(\triangle)$ to find a data fitting spline. Then we find the isosurface at level $L = 1$. See the graphs in Figure 26 (right).

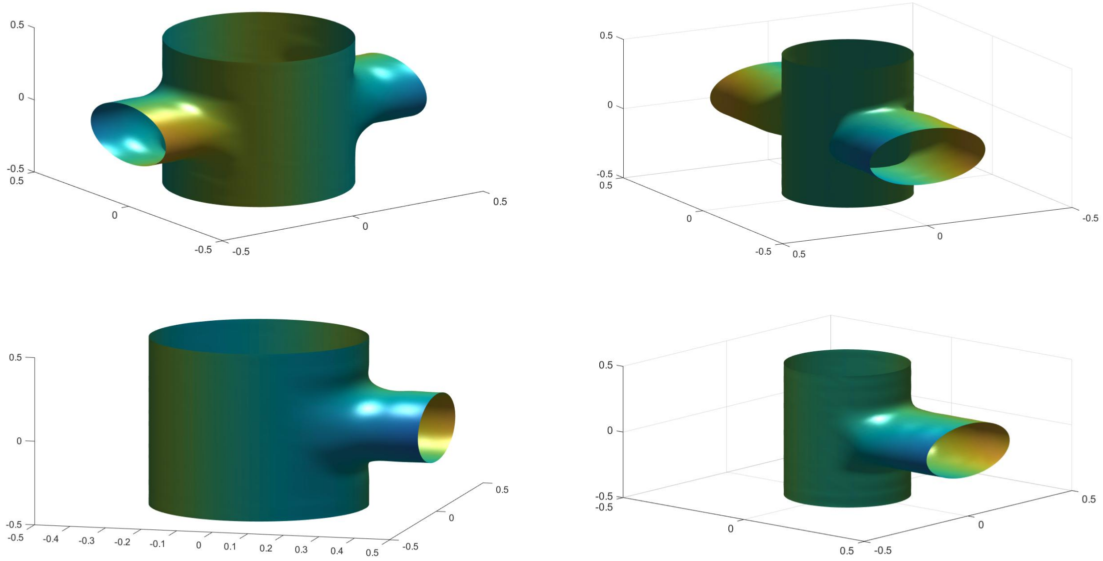

Figure 24: The spline pipe isosurfaces(right)

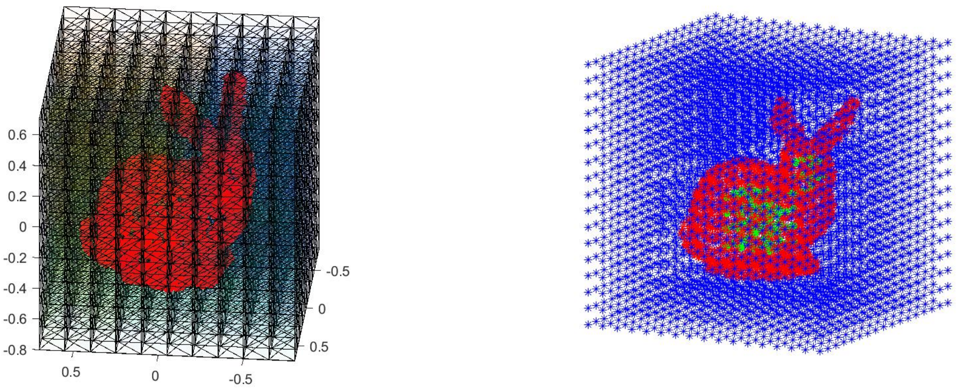

Figure 25: A tetrahedral partition with the bunny vertices in $\mathcal{D}$ shown in red (left) and the bunny data with auxiliary locations in blue and green (right).

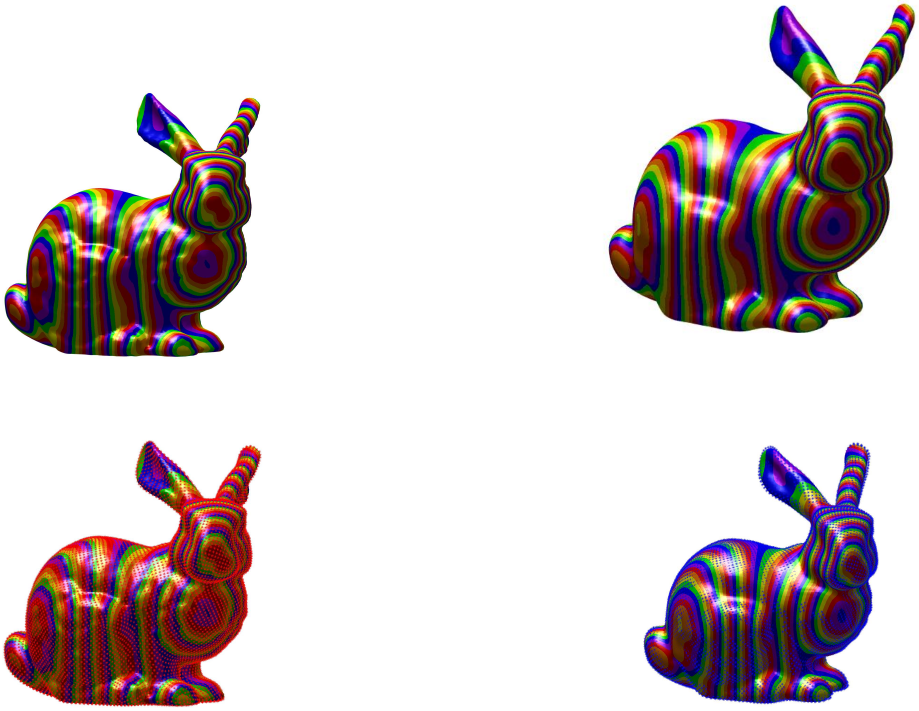

Figure 26: The isosurface of trivariate spline data interpolation using energy functional $\mathcal{E}_2$ and using $\mathcal{E}_3$ (top row) and these spline bunny surfaces with given data locations (bottom row)

From Figure 26, we can see that the trivariate spline interpolates the given data (the size of the bunny data is 11936) very well. Also, we can see that using energy functional $\mathcal{E}_3$ produces more smooth surface than using using energy functional $\mathcal{E}_2$.

Example 5.3. Let us present more examples to show that our method is usual for 3D surface reconstruction. See Figures 27, 28, 29. In these three cases, we used $\mathcal{E}_3$ for the energy functional when using the penalized least squares method.

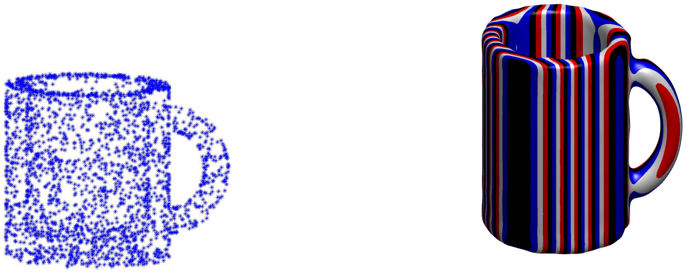

Figure 27: A data set of coffee mug-like surface and a spline surface of coffee mug.

Figure 28: A spline surface of human head and a different view with given data locations

Figure 29: A spline surface of human intestine and the surface with given data location

These figures show that the iso-surfaces satisfy the interpolatory condition very well. They are visually smooth as well as computationally smooth as the smoothness conditions were checked to have $1e-6$ accuracy.