An Improved Algorithm for Scattered Data Interpolation using Quartic Triangular Bézier Surfaces

Krassimira Vlachkova

Abstract We revisit the problem of interpolation of scattered data in $\mathbb{R}^3$ and propose a solution based on Nielson's minimum norm network and triangular Bézier patches. We aimed at solving the problem using the least number of polynomial patches of the smallest possible degree. We propose an alternative to the previously known algorithms, see (Clough and Tocher, 1965) and (Shirman and Séquin, 1987,1991). Although conceptually similar, our algorithm differs from the previous in all its steps. As a result the complexity of the resulting surface is reduced and its smoothness is improved. We present results of our numerical experiments.

1 Introduction

Scattered data interpolation is an important problem in Computer Aided Geometric Design and finds applications in various areas including automotive, aircraft and ship design, architecture, archeology, computer graphics, biomedical informatics, scientific visualization, and others. In general the problem can be formulated as follows: Given a set of points $\mathbf{d}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, $i = 1, \ldots, N$, find a bivariate function $F(x,y)$ defined in a certain domain D containing points $\mathbf{v}_i = (x_i, y_i)$, such that F possesses continuous partial derivatives up to a given order and $F(x_i, y_i) = z_i$.

Various methods for solving this problem were proposed and applied, see [10, 11, 13, 14]. Further information and treatment more recently have been presented in [1, 2, 3, 7]. A standard approach to solve the problem consists of two steps:

Step 1. Construct a triangulation $T = T(\mathbf{v}_1, \ldots, \mathbf{v}_N)$;

Step 2. For every triangle in T construct a surface (patch) which interpolates the data at the three vertices of T.

Krassimira Vlachkova
Faculty of Mathematics and Informatics, Sofia University "St. Kliment Ohridski", Sofia, Bulgaria
e-mail: `krassivl@fmi.uni-sofia.bg`

The interpolation surface constructed in Step 2 is usually polynomial or piecewise polynomial. Typically, the patches are computed with a priori prescribed normal vectors at the data points. G^1 or G^2-smoothness of the resulting surface is achieved either by increasing the degree of the patches, or by the so called *splitting* in which for each triangle in T a macro-patch consisting of a fixed number of Bézier sub-patches is constructed. Splitting was originally proposed by Clough and Tocher [6] and further developed by Percell [16] and Farin [8] for solving different problems. Known splitting algorithms apply splitting to all macro-patches defined in D. We point out that splitting decreases the smoothness of a polynomial macro-patch to G^1.

Shirman and Séquin [18, 20] construct a G^1-smooth surface consisting of quartic triangular Bézier surfaces. Their method assumes that the normal vectors at points $\mathbf{d}_i$, $i = 1,\ldots,N$, are given as part of the input. Shirman and Séquin construct a smooth cubic curve network defined on the edges of T, first, and then degree elevate it to quartic. This increases the degrees of freedom and allows them to connect smoothly the adjacent Bézier patches. Next, they apply splitting where for each triangle in T a macro-patch consisting of three quartic Bézier sub-patches is constructed. To compute the inner Bézier control points closest to the boundary of the macro-patch, Shirman and Séquin use a method proposed by Chiyokura and Kimura [5, 4]. The interpolation surfaces constructed by Shirman and Séquin's algorithm often suffer from unwanted bulges, tilts, and shears as pointed out by the authors in [19] and more recently by Hettinga and Kosinka in [12].

Nielson [15] proposes a method which computes a smooth interpolation curve network defined on the edges of T so as to have common tangent planes at the data points and to satisfy an extremal property. This curve network is called *minimum norm network* (MNN) and is cubic. Nielson extends the MNN to a smooth interpolation surface using a *blending* method based on convex combination schemes. The interpolant obtained is a rational function on every triangle in T.

In this paper we aim at constructing an interpolation surface that is piecewise polynomial and consists of the least number of patches of the smallest possible degree. Such interpolants are computationally tractable and desired in practice. We propose an algorithm that constructs interpolants consisting of quartic triangular Bézier patches and improves on Shirman and Séquin's method in various ways. More precisely, our contributions are as follows.

(i) We use the MNN and then degree elevate it to quartic. An important advantage of this approach is that the MNN is obtained through a global optimization of the curve networks defined on the edges of T which improves the shape of the patches. Another advantage is that the normal vectors at the data points are obtained through the computation of the MNN.

(ii) Our algorithm does not necessarily apply splitting in all triangles of T. We identify a subset of triangles $T_s \subset T$ where splitting needs to be applied. Finding the minimum size set T_s is computationally hard, and instead, we adopt an efficient incremental heuristic, where at each step the best candidate triangle is added to T_s. Decreasing the size of T_s reduces the complexity of the resulting surfaces and improves their quality.

(iii) We compute the control points of the Bézier patches so that the oscillations of the resulting interpolation surface near edges and vertices are minimized and distortions and twists are avoided.

(iv) Shirman and Séquin impose an additional condition that the three quartic curves defined on the common edges of the three sub-patches must be degree elevated cubic curves. This condition is not necessary to obtain G^1-continuity across the common edges of the sub-patches. We use different condition for the inner points of the sub-patches and believe that such a choice would facilitate the construction of convex macro-patches.

The remainder of this paper is organized as follows. In Sect. 2 we present some preliminary results. Our algorithms are presented in Sect. 3. In the final Sect. 4 examples of some of our numerical experiments are presented and compared with surfaces generated by Shirman and Séquin's method.

2 Preliminaries

2.1 Nielson's MNN

Let $N \geq 3$ be an integer and $\mathbf{d}_i := (x_i, y_i, z_i),\ i = 1, \ldots, N$ be different points in $\mathbb{R}^3$. We call this set of points *data*. The data are *scattered*[1] if the projections $\mathbf{v_i} := (x_i, y_i)$ onto the plane Oxy are different and non-collinear. Hereafter we assume that a triangulation T of the points $\mathbf{v}_i,\ i = 1, \ldots, N$, is given and fixed. Furthermore, for the sake of simplicity, we assume that the domain D formed by the union of the triangles in T is simply connected. The set of the edges in T is denoted by E. If there is an edge between $\mathbf{v}_i$ and $\mathbf{v}_j$ in E, it will be referred to by e_{ij} or simply by e if no ambiguity arises. A *curve network* is a collection of real-valued univariate functions $\{f_e\}_{e \in E}$ defined on the edges in E. With any real-valued bivariate function F defined on D we naturally associate the curve network defined as the restriction of F on the edges in E, i.e. for $e = e_{ij} \in E$,

$$\begin{gathered} f_e(t) := F\Big(\big(1 - \frac{t}{\|e\|}\big) x_i + \frac{t}{\|e\|} x_j,\ \big(1 - \frac{t}{\|e\|}\big) y_i + \frac{t}{\|e\|} y_j \Big), \\ \text{where } 0 \leq t \leq \|e\| \text{ and } \|e\| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \end{gathered} \tag{1}$$

Furthermore, according to the context F will denote either a real-valued bivariate function or a curve network defined by (1). We introduce the following class of *smooth interpolants* defined on D

$$\mathscr{F} := \big\{ F(x,y) \in C^1(D) \,|\, F(x_i, y_i) = z_i,\ i = 1, \ldots, N, f'_e \in AC,\ f''_e \in L^2,\ e \in E \big\},$$

[1] Note that this definition of scattered data slightly misuses the commonly accepted meaning of the term. It allows data with some structure among points $\mathbf{v}_i$. We have opted to do this in order to cover all cases where our presentation and results are valid.

where $C^1(D)$ is the class of bivariate functions defined in D which possess continuous first order partial derivatives, *AC* is the class of univariate absolutely continuous functions defined in $[0, \|e\|]$, and L^2 is the class of univariate functions defined in $[0, \|e\|]$ whose second power is Lebesgue integrable.

The restrictions on E of the functions in $\mathscr{F}_p$ form the corresponding class of so-called *smooth interpolation curve networks*

$$\mathscr{C}(E) := \left\{F_{|E} = \{f_e\}_{e\in E} \mid F(x,y) \in \mathscr{F}\right\}.$$

The smoothness of the interpolation curve network $F \in \{\mathscr{C}(E)\}$ geometrically means that at each point $\mathbf{d}_i$ there is a *tangent plane* to F, where a plane is *tangent* to the curve network at the point $\mathbf{d}_i$ if it contains the tangent vectors at $\mathbf{d}_i$ of the curves incident to $\mathbf{d}_i$.

For $F \in \{\mathscr{C}(E)\}$ we denote the curve network of second derivatives of F by $F'' := \{f''_e\}_{e\in E}$. The L^2-norm of F'' is defined by

$$\|F''\|_{L^2(T)} := \|F''\| = \left(\sum_{e\in E} \int_0^{\|e\|} |f''_e(t)|^2 dt\right)^{1/2}.$$

Nielson [15] considered and solved the following extremal problem

$$(\mathbf{P}) \quad \textit{Find } F^* \in \mathscr{C}(E) \ \textit{ such that } \|F^{*\prime\prime}\| = \inf_{F\in\mathscr{C}(E)} \|F''\|.$$

The unique solution (MNN) to ($\mathbf{P}$) is a cubic curve network and is obtained by solving a linear system of equations.

2.2 The G^1-continuity conditions

Let C_1 and C_2 be cubic triangular Bézier patches whose common boundary is the cubic curve $q(t)$. Let $q(t) = \sum_{i=0}^3 \mathbf{q}_i B_i^3(t)$ where $\mathbf{q}_i,\ i = 0,\dots,3$ be the control points of $q(t)$, and $B_i^m(t)$ be the Bernstein polynomials of degree m, $m \in \mathbb{N}$, defined for $0 \le t \le 1$ by

$$B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}, \qquad \binom{m}{i} = \begin{cases} \frac{m!}{i!(m-i)!}, & \text{for } i = 0,\dots,m, \\ 0, & \text{otherwise.} \end{cases}$$

Let us degree elevate C_1 and C_2 to quartic Bézier patches and denote the control points of the degree elevated $q(t)$ by $\hat{\mathbf{q}}_i,\ i = 0,\dots,4$, where $\hat{\mathbf{q}}_0 \equiv \mathbf{q}_0$ and $\hat{\mathbf{q}}_4 \equiv \mathbf{q}_3$. Then $q(t) = \sum_{i=0}^4 \hat{\mathbf{q}}_i B_i^4(t)$ where $\hat{\mathbf{q}}_i = \frac{i}{4}\mathbf{q}_{i-1} + \left(1 - \frac{i}{4}\right)\mathbf{q}_i,\ i = 0,\dots,4.$

Let $\mathbf{p}_i$ and $\mathbf{r}_i$, $i = 0,\dots,3$, be the nearest to $q(t)$ control points of C_1 and C_2, respectively, see Fig. 1(left). Farin ([9], pp. 368-371) proposed the following sufficient conditions for G^1-continuity between C_1 and C_2.

$$\frac{i}{4}\mathbf{b}_{i,4} + \left(1 - \frac{i}{4}\right)\mathbf{b}_{i,0} = 0,\ i = 0,\ldots,4,\ \text{where} \tag{2}$$

$$\begin{aligned}\mathbf{b}_{i,0} &= \alpha_0\mathbf{p}_i + (1-\alpha_0)\mathbf{r}_i - \big(\beta_0\hat{\mathbf{q}}_i + (1-\beta_0)\hat{\mathbf{q}}_{i+1}\big),\\ \mathbf{b}_{i,4} &= \alpha_1\mathbf{p}_{i-1} + (1-\alpha_1)\mathbf{r}_{i-1} - \big(\beta_1\hat{\mathbf{q}}_{i-1} + (1-\beta_1)\hat{\mathbf{q}}_i\big),\end{aligned}$$

and $0 < \alpha_i < 1,\ i = 1,2$. From (2) for $i = 0$ and $i = 4$ we obtain

$$\mathbf{b}_{0,0} = 0 \Rightarrow \alpha_0\mathbf{p}_0 + (1-\alpha_0)\mathbf{r}_0 = \beta_0\hat{\mathbf{q}}_0 + (1-\beta_0)\hat{\mathbf{q}}_1, \tag{3}$$

$$\mathbf{b}_{4,4} = 0 \Rightarrow \alpha_1\mathbf{p}_3 + (1-\alpha_1)\mathbf{r}_3 = \beta_1\hat{\mathbf{q}}_3 + (1-\beta_1)\hat{\mathbf{q}}_4 \tag{4}$$

The geometric meaning of (2) is that the four shaded quadrilaterals in Fig. 1(left) are planar. Moreover, $\alpha_i, \beta_i,\ i = 1,2$, are uniquely determined by the intersection point of the diagonals of the first and the last quadrilaterals, respectively.

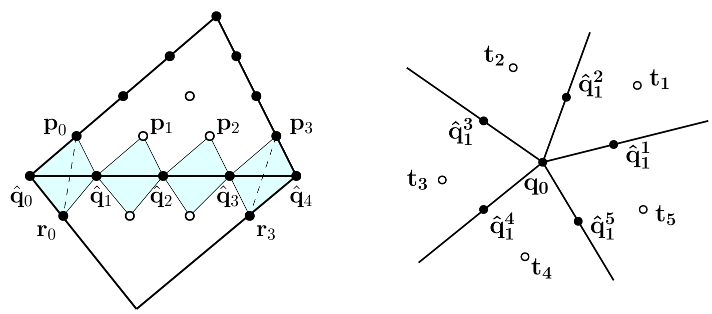

Fig. 1: (left) The G^1-continuity conditions: the four shaded quadrilaterals are planar; (right) The vertex enclosure problem for an odd degree vertex always has a solution.

2.3 The vertex enclosure problem

Let $\mathbf{v}$ be an inner vertex in T, i.e. $\mathbf{v} \equiv \mathbf{v}_i$ for some i, $1 \le i \le N$, and $\mathbf{q}_0$ be the corresponding point $\mathbf{d}_i$. Let n be the degree of $\mathbf{v}$, i.e. the number of the edges incident to $\mathbf{v}$. Let τ_k, $k = 1,\ldots,n$, be the triangles in T with common vertex $\mathbf{v}$ listed in counterclockwise order around $\mathbf{v}$, where τ_1 is arbitrarily chosen. For $k = 1,\ldots,n$, let $\mathscr{T}_k$ be the quartic Bézier patch defined in τ_k, and $q^k(t) = \sum_{i=0}^{4}\hat{\mathbf{q}}_i^k B_i^4(t)$ be the degree elevated cubic curves of the MNN with starting point $\mathbf{q}_0$ defined on the edges of τ_k. We denote by α_i^k, β_i^k, $i = 0,1$, the corresponding coefficients for $q^k(t)$ defined by (3) and (4). Let $\mathbf{t}_k$ be the nearest to $\mathbf{q}_0$ inner control point of $\mathscr{T}_k$, $k = 1,\ldots,n$, see Fig. 1(right) for $n = 5$.

By applying (3) to $\mathscr{T}_k$, $k = 1,\ldots,n$, we obtain the following linear system for the unknowns $\mathbf{t}_k$,

$$\begin{pmatrix} 1-\alpha_0^2 & \alpha_0^2 & 0 & \dots & 0 & 0 \\ 0 & 1-\alpha_0^3 & \alpha_0^3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1-\alpha_0^n & \alpha_0^n \\ \alpha_0^1 & 0 & 0 & \dots & 0 & 1-\alpha_0^1 \end{pmatrix} \begin{pmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \vdots \\ \mathbf{t}_{n-1} \\ \mathbf{t}_n \end{pmatrix} = \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \\ \vdots \\ \mathbf{s}_{n-1} \\ \mathbf{s}_n \end{pmatrix}, \tag{5}$$

where the points $\mathbf{s}_k$ depend on α_i^k, β_i^k, $\hat{\mathbf{q}}_i^k$, $k = 1, \dots, n$; $i = 1, 2$.

The existence of a solution to system (5) is known as the *vertex enclosure problem*. Peters [17] proved that the rank of (5) in n for odd n, and $n-1$ for even n. Therefore, if n is odd then system (5) always has a unique solution. If n is even then by Gauss elimination we obtain that (5) has a solution if and only if

$$\begin{aligned} &\mathbf{s}_n(1-\alpha_0^2)\dots(1-\alpha_0^n) - \mathbf{s}_{n-1}\alpha_0^1\dots\alpha_0^{n-1} \\ &+ \sum_{k=1}^{n-2}(-1)^k \mathbf{s}_k \alpha_0^1 \dots \alpha_0^k (1-\alpha_0^{k+2})\dots(1-\alpha_0^n) = 0. \end{aligned} \tag{6}$$

3 Outline of our algorithms

In this section we propose algorithms to construct G^1-continuous surface consisting of Bézier patches in the presence of even degree vertices. In [17] it is shown that although splitting does not eliminate even degree vertices, the vertex enclosure problem can be solved successfully since splitting adds more degrees of freedom. However, it is not necessary to split all triangles in T. In case the vertex is of even degree and (6) doesn't hold, it suffices to split just one triangle incident to the vertex. Our goal is to reduce the number of the splitted triangles since splitting decrease the smoothness of the macro-patch to G^1.

3.1 Determining the set T_s

First, using degree elevation we represent all cubic curves comprising the MNN as quartic Bézier curves. Next, we identify an optimized set T_s of triangles where splitting needs to be applied. For the remaining triangles, those in $T \setminus T_s$, we simply construct the standard quartic Bézier patch.

Let V_1 be the set of all inner vertices in T. First, for each edge e incident to V_1 for which the corresponding coefficients $\alpha_0(e)$ and $\alpha_1(e)$ defined by (3) and (4) are different, we insert the triangles incident to e in T_s and mark the endpoint vertices of e. Next, we identify a set $V_2 \subset V_1$ consisting of vertices that:

(i) are not marked;
(ii) have even degree;
(iii) (6) does not hold.

Let T_2 be the set of triangles incident to vertices in V_2. We assign weights w, $w = 1,2$, or 3, to triangles in T_2 equal to the number of vertices in T_2 incident to the corresponding triangle. We organize T_2 in a priority queue with respect to the weights. While T_2 is nonempty we insert a triangle with maximum weight in T_s, remove its vertices from V_2, and update the weights and the queue.

Algorithm 1 below takes the MNN as an input and construct a G^1-continuous interpolating surface $F(x,y)$ defined on D which consists of triangular quartic Bézier patches.

Algorithm 1

Step 1. Compute the control points of the curves in the MNN.
Step 2. Degree elevate all curves to quartic curves.
Step 3. Compute an optimized set T_s as described above.
Step 4. For each $\tau \in T \setminus T_s$ compute 3 inner control points and construct a macro-patch.
Step 5. For each $\tau \in T_s$ compute 19 inner control points and construct three sub-patches using Algorithm 2 for splitting as described in Sect. 3.2.

3.1.1 Solving the vertex enclosure problem

Let $\mathbf{v}$ be an inner vertex of T of degree n and $\mathbf{q}_0$ be the corresponding data point. First, we compute the control points $\mathbf{t}_1,\ldots,\mathbf{t}_n$ which are the nearest to $\mathbf{q}_0$, see Fig. 2. If no triangle with vertex $\mathbf{v}$ belongs to T_s then we compute $\mathbf{t}_1,\ldots,\mathbf{t}_n$ by solving the corresponding system (5), see Fig. 1(right). Let there be a triangle in T_s with vertex $\mathbf{v}$. In Fig. 2 an inner vertex $\mathbf{v}$ of degree 7 incident to four triangles in T_s is shown. Let $\tau_1,\ldots,\tau_r$ be the triangles with vertex $\mathbf{v}$ listed in counterclockwise order around $\mathbf{v}$, where τ_1 is arbitrarily chosen. Let $\mathbf{t}_k$ be the nearest to $\mathbf{q}_0$ point for τ_k, $k = 1,\ldots,r$. We divide $\tau_1,\ldots,\tau_r$ into groups of consecutive triangles such that the first and the last triangles in each group are sub-patches, and the intermediate triangles are macro-patches, see Fig. 2 where the groups are four: $\tau_1,\tau_2,\tau_3,\tau_4$; τ_5,τ_6; τ_7,τ_8,τ_9; τ_{10},τ_{11}. The groups are uniquely defined and their number is equal to the number of the triangles in T_s incident to $\mathbf{v}$. Let $\tau_1,\ldots,\tau_m$ be any of the groups. We compute $\mathbf{t_1},\ldots,\mathbf{t}_m$ by solving the corresponding system (5). Since τ_1 and τ_m do not have a common edge then the last equation of (5) is excluded. Then we can represent $\mathbf{t_1},\ldots,\mathbf{t}_m$ as linear functions of $\mathbf{t}_m$ using (5). We aim at minimizing the unwanted oscillations between patches $Q_1,\ldots,Q_m$ using $\mathbf{t}_m$ as a shape parameter. We compute its first two coordinates as shown in Sect. 3.2. Let z_i denote the third coordinate of $\mathbf{t}_i$, $i = 1,\ldots,m$. We compute z_m by minimizing the function $g(z_m) = \sum_{i=1}^{m-1}(z_{i+1} - z_i)^2$. The minimum is found readily since $g'(z_m) = 0$ is a linear equation.

In the case where $\mathbf{v}$ is a boundary vertex we compute the corresponding points $\mathbf{t}_1,\ldots,\mathbf{t}_r$ analogously to the case of an inner vertex. In this case τ_1 is uniquely determined and it is possible that either the first group of triangles starts, or the last group ends with a macro-patch instead of a sub-patch.

We note that if a triangle doesn't belong to T_s then by solving the vertex enclosure problem for its three vertices we completely determine the corresponding macro-patch since we obtain its three inner control points.

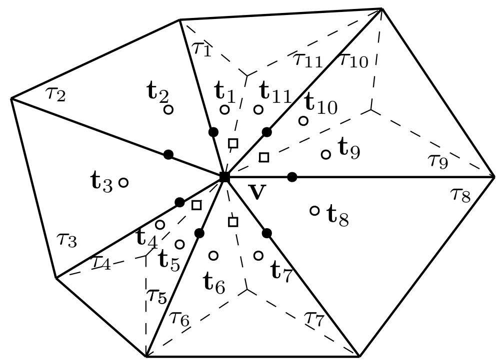

Fig. 2 An inner vertex $\mathbf{v}$ of degree 7 with four incident triangles in T_s. The points $\mathbf{t}_i$ are the nearest control points to the corresponding data point $\mathbf{q}_0$, $i = 1, \ldots, 11$.

3.2 Computing the control points of the sub-patches

Let τ be a triangle in T_s and $\mathscr{T}$ be the corresponding macro-patch defined in τ. We compute the control points of the three Bézier sub-patches consecutively in four layers as shown in Fig. 3(left). The first layer consists of inner control points that are the nearest to the boundary of τ. The last fourth layer contains a single point $\mathbf{z}$. This point $\mathbf{z}$ is the splitting point and will be computed as a center of the triangle with vertices in the previous third layer. We use the following notation.

- $\blacksquare$ vertices of the sub-patches;
- $\bullet$ inner control points on the boundary of the macro-patch;
- $\square$ inner control points of the three inner boundary curves of the sub-patches;
- $\circ$ inner control points of the sub-patches.

Next we explain in detail our choice of the control points in the first level. There are three points of type $\square$ and six points of type $\circ$ in this layer, see Fig. 3(left). First, we compute points of type $\square$ as centers of the three small triangles with vertices $\bullet\blacksquare\bullet$ on the boundary of the macro-patch. Then we compute the points $\mathbf{z}_i$, $i = 1, 2$, of type $\circ$ as follows.

Let the corresponding edge of τ be inner for T and $\tilde{\mathbf{q}}_1^i$, $\tilde{\mathbf{z}}_i$, $i = 1, 2$, be the corresponding control points of the neighbouring patch, see Fig. 3(right). We compute consecutively the following points.

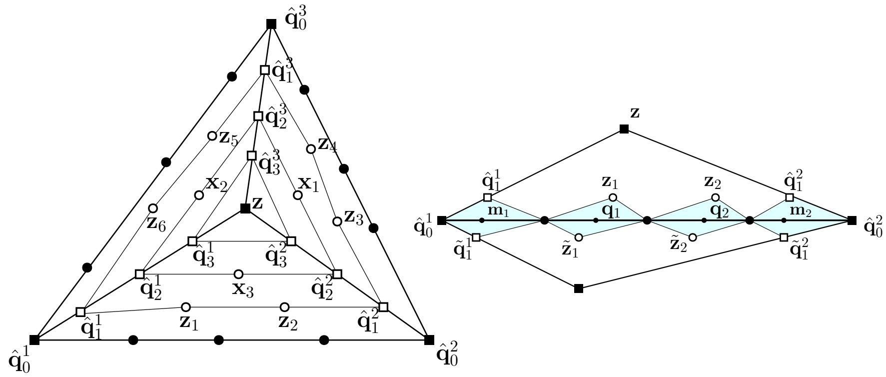

Fig. 3: (left) Construction of a G^1-continuous Bézier macro-patch by splitting to three sub-patches; (right) The control points in the first layer are computed to avoid unwanted twisting, tilting, and oscillations between the adjacent patches.

$$
\begin{aligned}
&\mathbf{z}_i' := \hat{\mathbf{q}}_1^1 + \mathbf{q}_i - \mathbf{m}_1,\ \tilde{\mathbf{z}}_i' := \tilde{\mathbf{q}}_1^1 + \mathbf{q}_i - \mathbf{m}_1,\\
&\mathbf{z}_i'' := \hat{\mathbf{q}}_1^2 + \mathbf{q}_i - \mathbf{m}_2,\ \tilde{\mathbf{z}}_i'' := \tilde{\mathbf{q}}_1^2 + \mathbf{q}_i - \mathbf{m}_2,\\
&\mathbf{a}_i := (1-\frac{i}{3})\mathbf{z}_i' + \frac{i}{3}\mathbf{z}_i'',\ \tilde{\mathbf{a}}_i := (1-\frac{i}{3})\tilde{\mathbf{z}}_i' + \frac{i}{3}\tilde{\mathbf{z}}_i'',\ i = 1,2,
\end{aligned}
$$

where $\mathbf{q}_i$, $i = 1,2$, are control points of the cubic curve defined on the common edge of τ and its neighbouring triangle, and $\mathbf{m}_i$, $i = 1,2$, are the intersection points of the diagonals of the quadrilaterals $\hat{\mathbf{q}}_0^1\hat{\mathbf{q}}_1^1\bullet\tilde{\mathbf{q}}_1^1$ and $\bullet\hat{\mathbf{q}}_1^2\hat{\mathbf{q}}_0^2\tilde{\mathbf{q}}_1^2$, respectively, as shown in Fig. 3(right).

Let $\mathbf{z}_i := (\xi_i, \eta_i, \zeta_i)$ and $\tilde{\mathbf{z}}_i := (\tilde{\xi}_i, \tilde{\eta}_i, \tilde{\zeta}_i)$. We choose ξ_i, η_i and $\tilde{\xi}_i, \tilde{\eta}_i$ to be equal to the corresponding coordinates of $\mathbf{a}_i$ and $\tilde{\mathbf{a}}_i$, respectively, $i = 1,2$. In this way the projections of $\mathbf{z}_i$, $i = 1,2$, onto Oxy lie inside τ. Hence, we avoid unwanted twisting and tilting of the patch. We compute the third coordinates ζ_i, $\tilde{\zeta}_i$ so that $\zeta_i = \tilde{\zeta}_i$ and $\mathbf{z}_i$, $\tilde{\mathbf{z}}_i$, $\mathbf{q}_i$ are collinear, $i = 1,2$. In this way we avoid unwanted oscillations between the adjacent patches.

In the case where the corresponding edge of τ is boundary for T, i.e. there is no neighbouring patch of $\mathscr{T}$, we compute $\mathbf{z}_i$, $i = 1,2$ as follows.

$$
\mathbf{r}_i' := \hat{\mathbf{q}}_1^1 - \hat{\mathbf{q}}_0^1 + \mathbf{q}_i,\ \mathbf{r}_i'' := \hat{\mathbf{q}}_1^2 - \hat{\mathbf{q}}_0^2 + \mathbf{q}_i,\ \mathbf{z}_i := (1-\frac{i}{3})\mathbf{r}_i' + \frac{i}{3}\mathbf{r}_i'',\ i = 1,2.
$$

The rest of the points $\mathbf{z}_i$, $i = 3,\dots,6$, are computed analogously.

Algorithm 2 below takes a triangle τ in T_s and the degree-elevated quartic boundary control points of the corresponding patch and computes 19 control points of the three G^1-continuous quartic Bézier sub-patches.

Algorithm 2

Step 1. Compute the control points in the first layer:
1.1 Points of type □ are centers of the three small triangles with vertices ●■●.
1.2 Then points of type ○ are computed as described in Sect. 3.2.
Step 2. Compute the control points in the second layer:
2.1 Points of type □ are centers of the three small triangles with vertices ○□○ in the first layer.
2.2 Then points of type ○ are mid-points of the segments with vertices of type □ in the second layer.
Step 3. Compute the control points in the third layer: The three points of type □ are centers of the small triangles with vertices ○□○ in the second layer.
Step 4. Compute the splitting point of type ■ as a center of the triangle with vertices □ in the third layer.

4 Examples

To demonstrate the results of our work we present here two examples and compare the surfaces generated by our algorithms and Shirman and Séquin's algorithm. For the latter we also use the MNN as input.

Example 1 We consider data obtained from a regular triangular pyramid. We have $N = 4$, $\mathbf{v}_1 = (-1/2, -\sqrt{3}/6)$, $\mathbf{v}_2 = (1/2, -\sqrt{3}/6)$, $\mathbf{v}_3 = (0, \sqrt{3}/3)$, $\mathbf{v}_4 = (0,0)$, and $z_i = 0$, $i = 1,2,3$, $z_4 = -1$. The corresponding MNN is shown in Fig. 4(left). In this case T_s is empty and no triangle has been splitted. The corresponding Shirman and Séquin's surface is shown in Fig. 5(left). The surface generated by our Algorithms 1 and 2 is shown in Fig. 5(right).

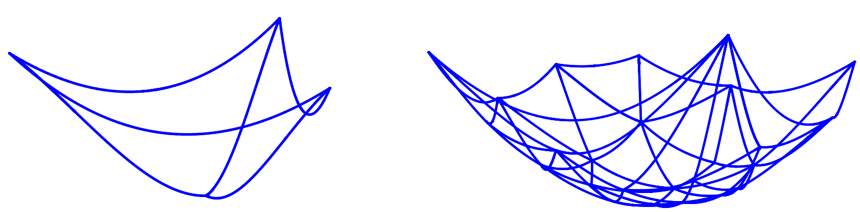

Fig. 4: (left) The MNN for Example 1; (right) The MNN for Example 2.

Example 2 We have $N = 25$ and data sampled from the function $F(x,y) = 5 * \exp\left((x-0.5)^2 + (y-0.5)^2\right)$. The triangulation is the Delaunay triangulation, it consists of 41 triangles and is shown in Fig. 6(left). The corresponding MNN is shown in Fig. 4(right). The set T_s consists of 6 triangles, see Fig. 6(right). The corresponding Shirman and Séquin's surface is shown in Fig. 6(left). The surface generated by our Algorithms 1 and 2 is shown in Fig. 6(right).

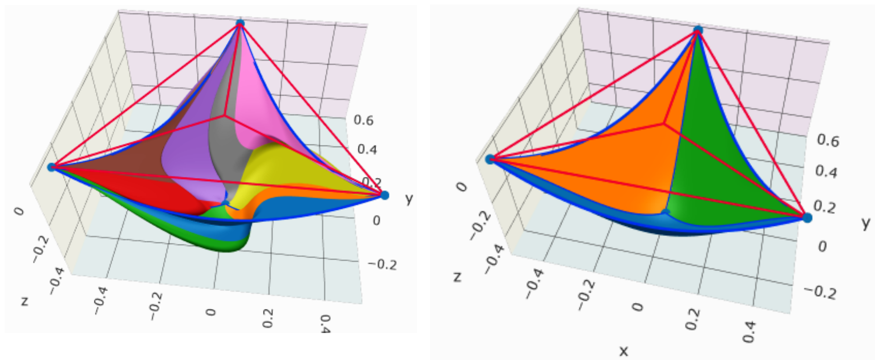

Fig. 5: Comparison of the two surfaces for the data in Example 1: (left) The surface generated using Shirman and Séquin's algorithm; (right) The surface generated using our Algorithms 1 and 2.

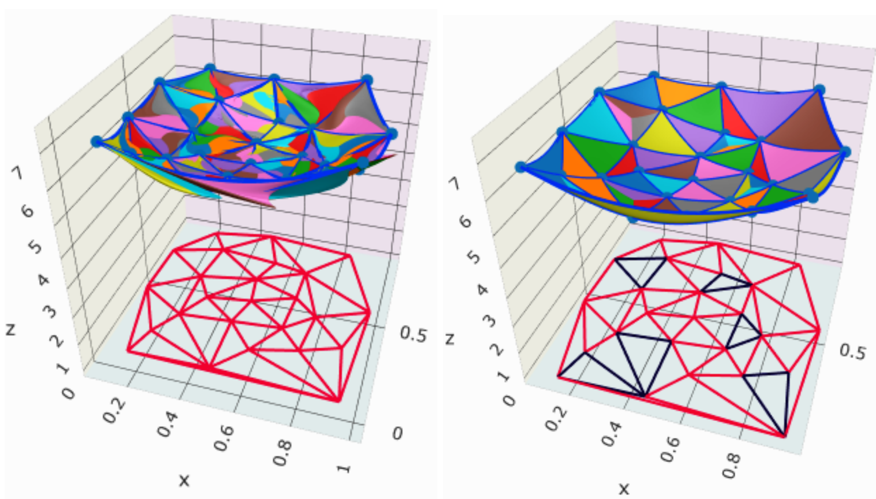

Fig. 6: Comparison of the two surfaces for the data in Example 2: (left) The surface generated using Shirman and Séquin's algorithm; (right) The surface generated using our Algorithms 1 and 2. The set T_s is marked in black and consists of 6 triangles.

Acknowledgements This research was supported by Sofia University Science Fund Grant No. 80-10-171/2020, and by European Regional Development Fund and the Operational Program "Science and Education for Smart Growth" under Contract BG05M2OP001-1.001-0004 (2018-2023). The author acknowledges Krum Radev's work and support in implementation and testing of the algorithms.

References

1. Amidror, I.: Scattered data interpolation methods for electronic imaging systems: a survey. J. of Electron. Imaging **11**, 157–176 (2002). DOI 10.1117/1.1455013
2. Anjyo, K., Lewis, J., Pighin, F.: Scattered data interpolation for computer graphics, SIGGRAPH 2014 course notes (2014). URL `http://olm.co.jp/rd/research_event/scattered-data-interpolation-for-computer-graphics`. Last accessed August 22, 2020
3. Cazals, F., Giesen, J.: Delaunay triangulation based surface reconstruction. In: J.D. Boissonat, M. Teillaud (eds.) Effective computational geometry for curves and surfaces, pp. 231–276. Springer, Berlin Heidelberg (2006). DOI 10.1007/978-3-540-33259-6_6
4. Chiyokura, H.: Localized surface interpolation method for irregular meshes. In: T. Kunii (ed.) Advanced Computer Graphics, Proceedings of Computer Graphics Tokyo'86, vol. 66–80, pp. 3–19. Springer, Tokyo (1986). DOI 10.1007/978-4-431-68036-9_1
5. Chiyokura, H., Kimura, F.: Design of solids with free-form surfaces. In: P.P. Tanner (ed.) SIGGRAPH '83 Proceedings of the 10th annual conference on Computer graphics and interactive techniques, vol. 17, pp. 289–298. ACM, New York (1983). DOI 10.1145/964967.801160
6. Clough, R., Tocher, J.: Finite elements stiffness matrices for analysis of plate bending. In: Proceedings of the 1st Conference on Matrix Methods in Structural Mechanics, vol. 66–80, pp. 515–545. Wright-Patterson A. F. B., Ohio (1965). URL `http://contrails.iit.edu/reports/8574`
7. Dell'Accio, F., Tommaso, F.D.: Scattered data interpolation by Shepard's like methods: classical results and recent advances. Dolomites Res. Notes Approx. **9**, 32 – 44 (2016). DOI 10.14658/pupj-drna-2016-Special_Issue-5
8. Farin, G.: A modified Clough-Tocher interpolant. Comput. Aided Geom. Des. **2**(1–3), 19–27 (1985). DOI 10.1016/0167-8396(85)90003-2
9. Farin, G.: Curves and Surfaces for CAGD: A Practical Guide, 5th edn. Morgan-Kaufmann, San Francisco (2002). DOI 10.1017/CBO9780511546860
10. Foley, T., Hagen, H.: Advances in scattered data interpolation. Surv. Math. Ind. **4**, 71–84 (1994)
11. Franke, R., Nielson, G.: Scattered data interpolation and applications: a tutorial and survey. In: H. Hagen, D. Roller (eds.) Geometric Modeling, pp. 131–160. Springer, Berlin (1991). DOI 10.1007/978-3-642-76404-2_6
12. Hettinga, G., Kosinka, J.: Multisided generalisations of Gregory patches. Comput. Aided Geom. Des. **62**, 166 – 180 (2018). DOI 10.1016/j.cagd.2018.03.005
13. Lodha, S., Franke, K.: Scattered data techniques for surfaces. In: Proceedings of Dagstuhl Conference on Scientific Visualization, pp. 182–222. IEEE Computer Society Press, Washington (1997). URL `https://ieeexplore.ieee.org/document/1423115`
14. Mann, S., Loop, C., Lounsbery, M., Meyers, D., Painter, J., DeRose, T., Sloan, K.: A survey of parametric scattered data fitting using triangular interpolants. In: H. Hagen (ed.) Curve and Surface Design, pp. 145–172. SIAM, Philadelphia (1992). DOI 10.1137/1.9781611971651.ch8
15. Nielson, G.: A method for interpolating scattered data based upon a minimum norm network. Math. Comput. **40**, 253–271 (1983). DOI 10.2307/2007373
16. Percell, P.: On cubic and quartic Clough-Tocher finite elements. SIAM J. Numer. Anal. **13**(1), 100–103 (1976). DOI 10.1137/0713011
17. Peters, J.: Smooth interpolation of a mesh of curves. Constr. Approx. **7**(1), 221–246 (1991). DOI 10.1007/BF01888155
18. Shirman, L., Séquin, C.: Local surface interpolation with Bézier patches. Comput. Aided Geom. Des. **4**(4), 279–295 (1987). DOI 10.1016/0167-8396(87)90003-3
19. Shirman, L., Séquin, C.: Local surface interpolation with shape parameters between adjoining gregory patches. Comput. Aided Geom. Des. **7**(5), 375–388 (1990). DOI 10.1016/0167-8396(90)90001-8
20. Shirman, L., Séquin, C.: Local surface interpolation with Bézier patches: errata and improvements. Comput. Aided Geom. Des. **8**(3), 217–221 (1991). DOI 10.1016/0167-8396(91)90005-V