Comput. Aided Geom. Des. 111 (2024) 102347

Contents lists available at ScienceDirect

Computer Aided Geometric Design

journal homepage: www.elsevier.com/locate/cagd

A point-normal interpolatory subdivision scheme preserving conics

Niels Bügel [a], Lucia Romani [b], Jiří Kosinka [a,*]

[a] Bernoulli Institute, University of Groningen, Nijenborgh 9, 9747AG Groningen, the Netherlands
[b] Dip. di Matematica, Alma Mater Studiorum Università di Bologna, Piazza Porta San Donato 5, 40126 Bologna, Italy

ARTICLE INFO

Keywords:
Curve design
Subdivision of 2D point-normal pairs
Interpolation
Conic reproduction

ABSTRACT

The use of subdivision schemes in applied and real-world contexts requires the development of conceptually simple algorithms that can be converted into fast and efficient implementation procedures. In the domain of interpolatory subdivision schemes, there is a demand for developing an algorithm capable of (i) reproducing all types of conic sections whenever the input data (in our case point-normal pairs) are arbitrarily sampled from them, (ii) generating a visually pleasing limit curve without creating unwanted oscillations, and (iii) having the potential to be naturally and easily extended to the bivariate case. In this paper we focus on the construction of an interpolatory subdivision scheme that meets all these conditions simultaneously. At the centre of our construction lies a conic fitting algorithm that requires as few as four point-normal pairs for finding new edge points (and associated normals) in a subdivision step. Several numerical results are included to showcase the validity of our algorithm.

1. Introduction

Subdivision schemes are efficient numerical methods for rapidly generating smooth curves/surfaces as the limit of an iterative process that starts from a given control polygon/mesh and recursively updates it by applying simple refinement rules. We refer the interested reader to Peters and Reif (2008); Sabin (2010); Warren and Weimer (2001) for a comprehensive treatment of univariate and bivariate subdivision schemes.

During the last thirty years, interpolatory subdivision schemes, i.e., schemes whose limit curves/surfaces interpolate the vertices of the input polygon/mesh, have been the subject of extensive research. This is due to the fact that many applications require the final shape to not merely approximate the input data, but exactly match them. However, despite this progress, we are still not aware of a conceptually simple subdivision scheme capable of efficiently producing a smooth interpolatory curve/surface that does not exhibit unwanted oscillations, fully respects the behaviour of the given polygon/mesh, and preserves conics/quadrics whenever the refinement process is applied to any arbitrarily-spaced samples of a quadratic shape.

In this paper, we aim to construct a univariate interpolatory subdivision scheme that:

- reproduces all types of conic sections whenever the starting data (point-normal pairs in our context) are arbitrarily sampled from them;

* Corresponding author.
E-mail address: j.kosinka@rug.nl (J. Kosinka).

https://doi.org/10.1016/j.cagd.2024.102347

Available online 23 May 2024

- generates a visually pleasing limit curve which faithfully mimics the behaviour of the underlying control polygon without creating unwanted oscillations;
- can be easily implemented;
- and that prepares the ground for a natural extension to the bivariate case.

One of the potential applications of such a subdivision scheme would be the derivation of active contours (and successively of active surfaces) for segmenting shapes in biomedical images and volumetric structures in 3D biomedical data, improving what has already been done in Badoual et al. (2017, 2021); Conti et al. (2015).

The remainder of this article is organized as follows. In Section 2 we review several existing univariate subdivision schemes, paying special attention to the interpolatory ones. Next, in Section 3, we introduce our new interpolatory subdivision scheme. The results and properties of the new scheme are evaluated in Section 4. Finally, we provide concluding remarks and directions for future investigations in Section 5.

2. Related work

In this section, we briefly review existing univariate subdivision schemes that are relevant to our work. We start with an overview of univariate interpolatory schemes with various shape-preserving properties (Section 2.1), and then continue by illustrating the conic-preserving schemes that are known to us (Section 2.2).

2.1. Interpolatory versus approximating univariate subdivision schemes

Univariate subdivision schemes can broadly be classified as interpolatory or approximating according to whether they generate a limit curve that passes through all the vertices of the control polygon or not. At each refinement step, approximating schemes, in addition to inserting new edge points, also update the location of the old vertex points. A prime example are the subdivision schemes derived from uniform B-splines. Because of this, approximating schemes tend to require smaller local support compared to interpolatory ones. And although approximating schemes often rely on simple rules for positioning new vertices, they are able to ensure smooth limit curves, which usually turn out to be much smoother than those provided by interpolatory schemes with the same support. However, while approximating subdivision methods tend to produce smoother curves (Pan et al., 2012), the resulting shapes might not be an accurate representation of the original control polygon due to the significant shrinkage effect, which is unacceptable in applications where the limit curve has to pass through the original sample points/vertices.

In contrast, interpolatory subdivision schemes produce limit curves that pass through the vertices of their control polygon, which is simply achieved by leaving these points at their original location as subdivision proceeds. And although these limit curves can be expected to give users intuitive and direct control over their final shape, they very often suffer from undesired artefacts since they tend to exhibit more convexity changes than the underlying control polygon, especially when its edges have highly non-uniform lengths.

To alleviate this, the research of the last three decades has focused on developing algorithms capable of ensuring shape-preserving (i.e., convexity preserving as well as artefact free) interpolants, with many valuable schemes (Albrecht and Romani, 2012; Bebarta and Jena, 2021, 2023; Cai, 2009; Dyn et al., 1992; Goodman and Ong, 2005; Jena, 2021; Kuijt and Van Damme, 2002; Marinov et al., 2005; Novara and Romani, 2018; Yang, 2006). However, except for Albrecht and Romani (2012) which we discuss further below, none of these schemes is capable of preserving all types of conic sections when applied to arbitrarily-spaced conic samples.

2.2. Conic-preserving interpolatory curve subdivision

As far as we are aware, the earliest reference to an interpolatory subdivision scheme capable of reproducing geometric entities other than lines, is the paper of Sabin and Dodgson (2004). It was designed as a variant of the original four-point subdivision scheme (Dyn et al., 1987), with the rule to insert the new edge vertices constructed in such a way that the curvature at each new vertex equals the mean of the curvatures at the adjacent old vertices. The resulting limit curve reproduces circles and is almost C^2-continuous. While Sabin and Dodgson's method is capable of reproducing circles from any arbitrary sequence of points sampled from them, other types of conic sections are not explicitly preserved. The same applies to several other successively proposed subdivision schemes, including Aihua et al. (2016); Chalmovianský and Jüttler (2007); Conti et al. (2015); Deng and Ma (2014); Deng and Wang (2010); Hernández-Mederos et al. (2013); Lipovetsky and Dyn (2016); Romani (2010); Song et al. (2013); Yang (2023), which are able to reproduce (only) circles and in some cases also ellipses.

In contrast, the interpolatory subdivision schemes proposed in Beccari et al. (2007, 2009); Donat and López-Ureña (2019); Novara and Romani (2015); Romani (2009) are able to reproduce all types of conic sections, but reproduction is guaranteed only when the vertices of the control polygon are uniformly sampled on the desired conic.

Differently from the above-mentioned schemes, the interpolatory subdivision scheme introduced in Albrecht and Romani (2012) reproduces conic sections also when the starting data are irregularly distributed. For every edge, it first calculates the tangents at its endpoints using a conic tangent estimator. Then, the two vertices of each edge and their two tangent lines are used to construct a conic and insert a new edge point lying on it by exploiting a result from projective geometry. As non-convex curves cannot be approximated by a second-order curve, non-convex data are first segmented into convex segments in order to be handled independently as convex-only parts. The examples provided in Albrecht and Romani (2012) suggest that the generated limit curve is at least C^1 continuous,

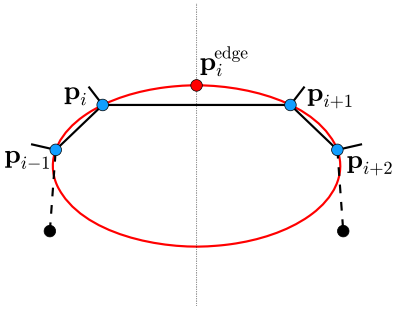

Fig. 1. The local neighbourhood of the edge $\mathbf{p}_i\mathbf{p}_{i+1}$ and the insertion rule of the edge point $\mathbf{p}_i^{\text{edge}}$.

but due to the complexity of the method, a mathematical proof is lacking. Another limitation of Albrecht and Romani (2012) is that it is difficult to extend the method to the bivariate setting, i.e., when control polygons are replaced by quadrilateral/triangular control meshes and quadric-preservation is desired.

To the best of our knowledge, there are no bivariate interpolatory subdivision schemes capable of reproducing quadrics when the vertices of the control mesh are arbitrarily-spaced samples coming from a quadric. The problem we aim to solve in this paper is thus to construct a simple and efficient interpolatory scheme for preserving conics, with an eye towards a natural bivariate extension with reproduction of quadrics.

3. Conic subdivision

In order to derive an interpolatory subdivision scheme for curves, we need to design a rule that governs the insertion of new edge points (and associated normals) at each subdivision step (as the vertex points and their normals stay fixed) so that conics are preserved. More precisely, given an edge and a local neighbourhood with point-normal pairs sampled from a certain conic, the new edge point of said edge (and the associated normal) should also come from this conic (see Fig. 1). This problem can be divided into two parts. First, a conic is constructed that fits the local neighbourhood (Section 3.1). Second, a point-normal pair is sampled from the found conic to determine the location of the new edge point and its associated normal (Section 3.2). We then discuss a modification to the scheme that allows it to work with non-convex input polygons (Section 3.3).

3.1. Conic fitting

A conic section (or simply a conic) in the (x, y) plane can be described by the implicit equation $f(x, y) = 0$ where $f(x, y)$ is defined as a quadratic polynomial:

$$f(x,y) = q_{20}x^2 + q_{02}y^2 + 2q_{11}xy + 2q_{10}x + 2q_{01}y + q_{00}, \tag{1}$$

where $q_{00}, q_{01}, q_{10}, q_{11}, q_{02}, q_{20}$ are real coefficients. The aim is to find a certain 'best-fitting' conic that is defined by the local neighbourhood of an edge in a given control polygon. While there are synthetic methods that make use of projective geometry, such as the one proposed by Albrecht and Romani for conic sections (Albrecht and Romani, 2012), these methods are generally very complex and not flexible. As such, we opt for an analytical method that involves creating a system of linear equations based on (1) and solving for the coefficients.

To construct these linear equations, the coordinates of a number of points in $\mathbb{R}^2$ are required. These can be obtained from the vertices in the neighbourhood of the edge in question. Given a sequence $\mathbf{P} = (\mathbf{p}_1, ..., \mathbf{p}_n)$ of n points with coordinates $[x_i, y_i]^T$, $i = 1, \dots, n$, one could construct the homogeneous linear system

$$\mathbf{A}\mathbf{q} = \mathbf{0}, \tag{2}$$

where

$$\mathbf{A} = \begin{bmatrix} x_1^2 & y_1^2 & 2x_1y_1 & 2x_1 & 2y_1 & 1 \\ x_2^2 & y_2^2 & 2x_2y_2 & 2x_2 & 2y_2 & 1 \\ & & \vdots & & & \\ x_n^2 & y_n^2 & 2x_ny_n & 2x_n & 2y_n & 1 \end{bmatrix}_{n\times 6}, \qquad \mathbf{q} = \begin{bmatrix} q_{20} & q_{02} & q_{11} & q_{10} & q_{01} & q_{00} \end{bmatrix}^T,$$

and attempt to solve it. However, there are two main issues with this strategy.

First, (1) has 5 degrees of freedom. Therefore, in general, 5 vertices are required to construct $\mathbf{A}$ such that the linear system can be solved. This is not ideal as it requires an asymmetrical neighbourhood around a given edge and many singular configurations exist. In the case that we opt for 6 points (so three on either side of the edge), the system is, in general, over-constrained and thus leads to no solutions.

Secondly, the resulting conic has to pass through all the given points exactly (since we are after an interpolatory scheme). As the size of the neighbourhood has to be relatively large, the chance of finding a good quality conic for arbitrary curves is relatively small.

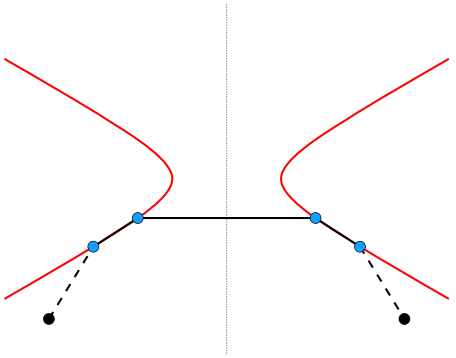

Fig. 2. The found conic is a hyperbola where the vertices lie on different branches, resulting in no line-conic intersection solutions. This situation can be partially averted by growing the fitting neighbourhood when this situation occurs.

Additionally, the constructed conic could be a (two-branch) hyperbola with different parts of the neighbourhood being on different branches of said hyperbola (see Fig. 2). Preventing these undesirable cases is a challenging issue. Restricting the fitting process to certain classes of conics is not a valid strategy, since the conic itself is perfectly valid, provided that the entire neighbourhood lies on a single branch. As such, we cannot constrain the type of conic we obtain to a specific type, as done for example by Harker et al. (2008).

All in all, considering the local neighbourhood just as a point collection does not seem to lead to the desired results. We therefore augment the neighbourhood information with the (unit) normals of the vertices. Not only does this allow for a reduction in the neighbourhood size, but it also provides extra (implicit) information about local shape. We now leverage the strategy used by Birdal et al. (2020) to use this oriented point collection for conic fitting. While their method is intended for fitting quadrics to oriented point clouds, the method is easily adjusted to our setting.

Precisely, we augment the system in (2) by setting the gradient of the conic at each point $\mathbf{p}_i$ equal to a multiple $\alpha_i \in \mathbb{R} \setminus \{0\}$ of the normal $\mathbf{n}_i = [n_i^x, n_i^y]^T$ at said point:

$$\nabla f(x,y) = \begin{bmatrix} 2q_{20}x + 2q_{11}y + 2q_{10} \\ 2q_{02}y + 2q_{11}x + 2q_{01} \end{bmatrix} = \alpha_i \begin{bmatrix} n_i^x \\ n_i^y \end{bmatrix}. \tag{3}$$

This introduces a unique scaling factor α_i for every normal. To include these equations in the conic fitting process, we change (2) to

$$\mathbf{A}'\mathbf{q}' = \mathbf{0}$$

with

$$\mathbf{A}' = \begin{bmatrix} x_1^2 & y_1^2 & 2x_1y_1 & 2x_1 & 2y_1 & 1 & 0 & 0 & \dots & 0 \\ x_2^2 & y_2^2 & 2x_2y_2 & 2x_2 & 2y_2 & 1 & 0 & 0 & \dots & 0 \\ & & & & & \vdots & & & & \\ x_n^2 & y_n^2 & 2x_ny_n & 2x_n & 2y_n & 1 & 0 & 0 & \dots & 0 \\ 2x_1 & 0 & 2y_1 & 2 & 0 & 0 & -n_1^x & 0 & \dots & 0 \\ 0 & 2y_1 & 2x_1 & 0 & 2 & 0 & -n_1^y & 0 & \dots & 0 \\ 2x_2 & 0 & 2y_2 & 2 & 0 & 0 & 0 & -n_2^x & \dots & 0 \\ 0 & 2y_2 & 2x_2 & 0 & 2 & 0 & 0 & -n_2^y & \dots & 0 \\ & & & & & \vdots & & & & \\ 2x_n & 0 & 2y_n & 2 & 0 & 0 & 0 & 0 & \dots & -n_n^x \\ 0 & 2y_n & 2x_n & 0 & 2 & 0 & 0 & 0 & \dots & -n_n^y \end{bmatrix}_{(3n)\times(n+6)}, \tag{4}$$

and

$$\mathbf{q}' = \begin{bmatrix} q_{20} & q_{02} & q_{11} & q_{10} & q_{01} & q_{00} & \alpha_1 & \dots & \alpha_n \end{bmatrix}^T. \tag{5}$$

The resulting homogeneous linear system is in general over-constrained, and thus cannot be solved directly. To circumvent this, we use the least-squares solution to $\mathbf{A}'\mathbf{q}' = \mathbf{0}$ that can be found using Singular Value Decomposition (SVD):

$$\mathbf{A}' = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

where $\mathbf{U}$ is a $3n \times 3n$ orthogonal matrix, $\mathbf{\Sigma}$ is a $3n \times (n+6)$ matrix with non-negative real numbers on the diagonal (the so-called singular values of $\mathbf{A}'$), and $\mathbf{V}$ is a $(n+6) \times (n+6)$ orthogonal matrix.

The best-fitting solution for $\mathbf{q}'$ (i.e., the vector $\mathbf{q}'$ that minimizes the 2-norm of $\mathbf{A}'\mathbf{q}'$ under the constraint $||\mathbf{q}'|| = 1$) is the right-singular vector of $\mathbf{A}'$ corresponding to the smallest singular value. In other words, the sought solution turns out to be the vector

from the columns of $\mathbf{V}$ (i.e., the right-singular vectors) that corresponds to the smallest singular value of $\mathbf{\Sigma}$. The advantage of this approach is that with the introduction of normals, the neighbourhood generally needs to be no larger than 4 point-normal pairs, i.e., two on either side of the processed edge (see Fig. 1). It also points to a bivariate generalisation.

Due to the approximating nature of the above method, the found conics are no longer guaranteed to interpolate the two vertices of the edge. In order to tackle this interpolation problem, we introduce a diagonal weight matrix $\mathbf{W}$ to steer the solution to better fit certain equations in the system. The matrix $\mathbf{W}$ contains as its diagonal elements the weights for each equation. With this, the homogeneous linear system becomes

$$\mathbf{W}\mathbf{A}'\mathbf{q}' = \mathbf{0},$$

and the best-fitting solution for $\mathbf{q}'$ becomes the right-singular vector of $\mathbf{W}\mathbf{A}'$ that corresponds to the smallest singular value.

While we could introduce different weights for each equation, we found that it is sufficient to differentiate between only four different weights: the edge weights w_e^v and w_e^n for the edge vertices and normals respectively, and the 'outer' weights w_o^v and w_o^n for the outer vertices and normals. By setting w_e^v to a large value, the system will find solutions that favour the interpolation of the edge vertices. In addition to interpolating the edge vertices, we found that interpolating the corresponding two normals produces better results, as the found conic better fits the local shape. Additionally, for the outer vertices and normals, more emphasis is put on the vertices as opposed to the normals. This ensures that large variations in the curvature around the local shape do not disproportionately affect the conic. As such, we set

$$w_o^v = \tau w_o^n \quad \text{with} \quad \tau > 1. \tag{6}$$

We have found that $\tau = 100$ works well (see Section 4 and Fig. 6) and it is the default setting in all our examples.

3.2. Edge point and normal computation

Having constructed a locally best-fitting conic, the location of the new edge point needs to be determined. Ideally, the edge point should be close to the midpoint $\mathbf{p}_i^{\text{mid}}$ of the edge to promote evenly spaced geometry and prevent unnecessary curvature oscillations. Given an edge defined by two points $\mathbf{p}_i = [x_i, y_i]^T$ and $\mathbf{p}_{i+1} = [x_{i+1}, y_{i+1}]^T$, we calculate its midpoint

$$\mathbf{p}_i^{\text{mid}} = \frac{\mathbf{p}_i + \mathbf{p}_{i+1}}{2}.$$

Next, we construct the exact normal of the edge $\mathbf{n}_i^{\text{edge}}$ as

$$\mathbf{n}_i^{\text{edge}} = [y_i - y_{i+1}, x_{i+1} - x_i]^T.$$

We now construct the line $\mathbf{l}(t)$ that is perpendicular to the edge and passes through its midpoint:

$$\mathbf{l}(t) = \mathbf{p}_i^{\text{mid}} + t\mathbf{n}_i^{\text{edge}},\ t \in \mathbb{R}. \tag{7}$$

Using this, we find the line-conic intersections by rewriting (1) in matrix form $f(\mathbf{x}) = 0$, where

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}\mathbf{x} \tag{8}$$

with

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} q_{20} & q_{11} & q_{10} \\ q_{11} & q_{02} & q_{01} \\ q_{10} & q_{01} & q_{00} \end{bmatrix}.$$

We can find the solutions for t that describe the line-conic intersection by evaluating:

$$f(\mathbf{l}(t)) = 0 \tag{9}$$

as described by Trettner and Kobbelt (2021). Since conics are defined by a quadratic polynomial, there will be generically at most two real intersection points, corresponding to t_1 and t_2. The location of the new edge point $\mathbf{p}_i^{\text{edge}}$ is determined by:

$$\mathbf{p}_i^{\text{edge}} = \begin{cases} \mathbf{l}(t_1), & \text{if } \left|\left|\mathbf{p}_i^{\text{mid}} - \mathbf{l}(t_1)\right|\right| \leq \left|\left|\mathbf{p}_i^{\text{mid}} - \mathbf{l}(t_2)\right|\right|, \\ \mathbf{l}(t_2), & \text{otherwise.} \end{cases} \tag{10}$$

This describes that the location of the new edge point is the intersection point closest to the midpoint. We have also considered setting the new edge point as the geodesic mid-point on the found conic between the two vertices of the edge in question. While this is arguably the best solution as far as geometry and uniformity are concerned, it comes with a non-trivial computational effort, and we thus did not pursue this further. The (new) normal associated with $\mathbf{p}_i^{\text{edge}}$ is also sampled from the fitted conic.

Despite the various techniques described here, the undesirable case of the conic being a (two-branch) hyperbola can still occur. This can result in no solutions being found; see Fig. 2. If this is the case, we incrementally increase the neighbourhood size until

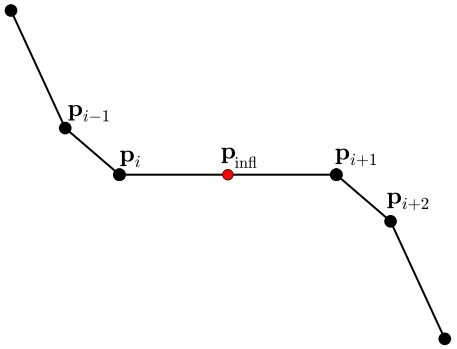

Fig. 3. A non-convex curve segment with the corresponding inflection point $\mathbf{p}_{\text{infl}}$.

a solution is found. If no solution is found and it is not possible to add points to the neighbourhood without breaking the global convexity property, then we set $\mathbf{p}_i^{\text{edge}} = \mathbf{p}_i^{\text{mid}}$. This situation occurs only extremely rarely.

3.3. Modification for non-convex input polygons

As we have introduced a separate scaling factor for each normal, we have added a significant number of extra degrees of freedom. This, combined with the fact that conics do not possess, and thus cannot represent, inflection points, may lead to poor quality results for non-convex input polygons, as already observed by Birdal et al. (2020) when fitting quadrics to oriented point clouds. Their solution for a smoother surface is to enforce $\alpha = 1$ for each normal, which forces the system to find quadrics whose gradients match the unit normals. However, this unit normal constraint results in the system not being able to reproduce all conics. As such, we use a different approach to ensure smooth curves in the case of non-convex input polygons.

We follow the idea proposed by Albrecht and Romani (2012) to split the control polygon into a collection of globally convex sub-polygons before starting the subdivision process. We do this by first identifying the inflection edges. In their paper, an inflection edge is described as an edge $\mathbf{p}_i\mathbf{p}_{i+1}$ for which the points $\mathbf{p}_{i-1}$ and $\mathbf{p}_{i+2}$ lie on different half planes with respect to this edge. For each inflection edge, we introduce a new point, called the inflection point $\mathbf{p}_{\text{infl}}$. We set the location of $\mathbf{p}_{\text{infl}}$ to be the midpoint of the edge; see Fig. 3.

In places where these inflection points are inserted, we observe that the neighbourhood would have three consecutive points that are collinear. This configuration causes the system to find undesirable (singular) conics, so instead we opt to remove the outer collinear point for any given neighbourhood surrounding an inflection point. For example, point $\mathbf{p}_i$ is not included in the neighbourhood of edge $\mathbf{p}_{\text{infl}}\mathbf{p}_{i+1}$.

As our method uses both points and normals to find a conic, the normal $\mathbf{n}_{\text{infl}}$ at the inflection point also needs to be set appropriately as it influences the resulting curve. If $\mathbf{n}_{\text{infl}}$ is chosen to be close to orthogonal with respect to the edge $\mathbf{p}_i\mathbf{p}_{i+1}$, the curve at the inflection point should be close to flat. In contrast, if $\mathbf{n}_{\text{infl}}$ points in the same direction as the edge, then there will be a 'wiggle' in the curve. In our case, a normal that is parallel to the edge is undesirable, as this does not produce good quality conics. As such, we restrict the angle between $\mathbf{n}_{\text{infl}}$ and the direction $\mathbf{n}_i^{\text{edge}}$ perpendicular to the edge to be between 0 and $\frac{1}{4}\pi$. To ensure this, we proceed as follows.

We wish to set a blending factor $0 \leq \gamma_0 \leq 0.5$ to linearly blend between the normalised direction of $\mathbf{p}_i\mathbf{p}_{i+1}$ (denoted as $\widehat{\mathbf{p}_{i+1} - \mathbf{p}_i}$) and the normalised $\mathbf{n}_i^{\text{edge}}$ (denoted as $\widehat{\mathbf{n}_i^{\text{edge}}}$) to obtain

$$\mathbf{n}_{\text{infl}}^0 = (1-\gamma_0)\widehat{\mathbf{n}_i^{\text{edge}}} + \gamma_0(\widehat{\mathbf{p}_{i+1} - \mathbf{p}_i}). \tag{11}$$

Although our method introduces a separate scaling factor per normal, making the (inflection) normals $\mathbf{n}_{\text{infl}}^0$ and $-\mathbf{n}_{\text{infl}}^0$ essentially equivalent, we flip $\mathbf{n}_i^{\text{edge}}$ if $\mathbf{n}_i^{\text{edge}} \cdot (\mathbf{p}_{i+1} - \mathbf{p}_i) < 0$ for direction blending in (11). With $\phi_i = \angle \mathbf{p}_{i-1}\mathbf{p}_i\mathbf{p}_{i+1}$ the angle between $\mathbf{p}_{i-1}\mathbf{p}_i$ and $\mathbf{p}_i\mathbf{p}_{i+1}$, we set

$$\gamma_0 = \frac{1}{2} - \left|\frac{\phi_i}{\pi} - \frac{1}{2}\right|.$$

The resulting value for γ_0 is then used in (11) to obtain $\mathbf{n}_{\text{infl}}^0$. Analogously, we compute γ_1 and $\mathbf{n}_{\text{infl}}^1$ based on ϕ_{i+1}. The final normal $\mathbf{n}_{\text{infl}}$ at the inflection point is then set as

$$\mathbf{n}_{\text{infl}} = \begin{cases} \mathbf{n}_{\text{infl}}^0 & \text{if } \angle \mathbf{n}_{\text{infl}}^0 \mathbf{n}_i^{\text{edge}} < \angle \mathbf{n}_{\text{infl}}^1 \mathbf{n}_i^{\text{edge}}, \\ \mathbf{n}_{\text{infl}}^1 & \text{otherwise.} \end{cases} \tag{12}$$

This ensures good behaviour in case of flat sections or sharp edges near the inflection points.

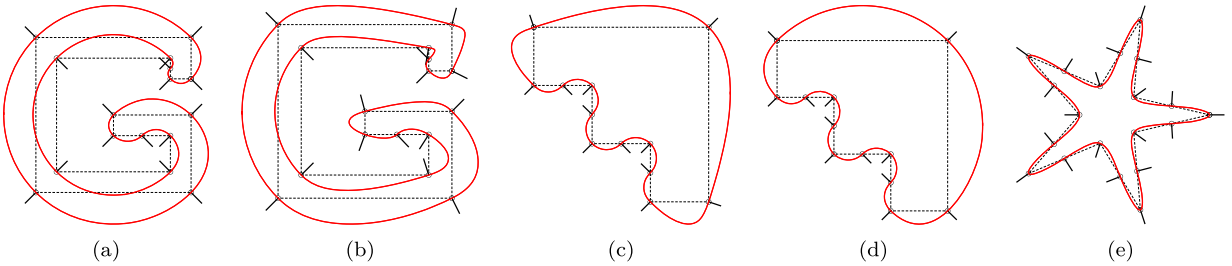

Fig. 4. Results using our subdivision scheme. In (a), (c), and (e) the normals at the control points are given by simply averaging the adjacent edge normals of the control polygon. In (b) and (d) the averaging of the normals at control points takes into account the length of the two incident edges at each vertex. Due to the symmetry of the star control polygon (e), both options lead to the same normals and thus the same subdivision curve.

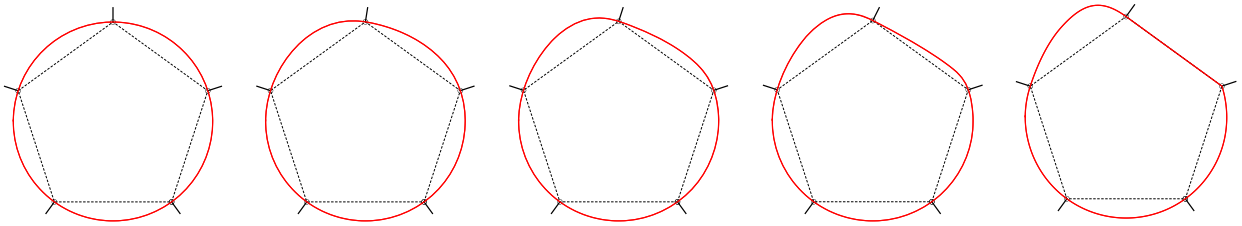

Fig. 5. The effect of adjusting the normal of the top control point. The regular pentagonal control polygon is initially equipped with normals sampled from its circumscribed circle. From left to right: The normal is adjusted in uniform steps from its original vertical position (far left) all the way to being orthogonal to one of the edges of the polygon (far right). The other normals and all points stay fixed.

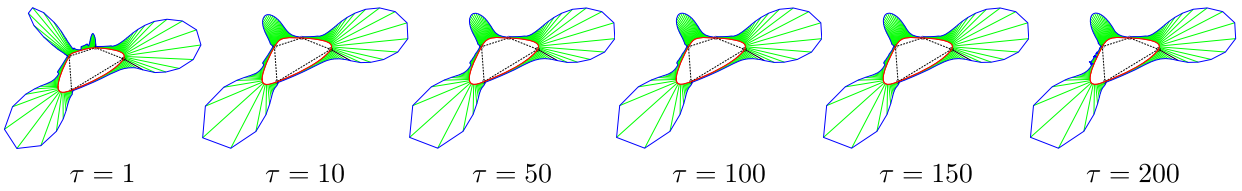

Fig. 6. The influence of parameter τ; see (6). Low and high values of τ lead to suboptimal results, whereas $\tau \in [10, 150]$ provides good and nearly indistinguishable limit curves and curvature behaviour. We have thus made $\tau = 100$ our default setting.

4. Results and evaluation

We start by showing several examples of limit curves generated by our subdivision scheme in Fig. 4. The normals at the control points were set automatically from the geometry of the control polygon by (edge-length-adjusted) averaging of edge normals. The change of the normal direction influences the subdivision curves in a natural way. This is further investigated in Fig. 5, where a single normal is adjusted. Note that the first configuration results in the exact circle the data was sampled from. The figure also shows the reasonable range for setting normals at a vertex: the normal should be set within the angle bounded by orthogonal directions to the two edges incident with the vertex under consideration. Given the geometric construction of the scheme and the fact the conics cannot capture inflection points, other configurations cannot be expected to lead to reasonable results.

Further examples are provided in Fig. 7, some of which include challenging configurations, such as with non-uniform distributions of points, a double loop, with inflection points, and with a sharp corner and three collinear points. In the figure, top row, we also show the curvature combs of the curves. The middle row plots the angle between consecutive segments along the curve, and the bottom row shows corresponding curvature plots (estimated using osculating circles passing through three consecutive points).

The angle variation plots suggest that our scheme produces tangent-line, i.e., G^1, continuous limit curves (unless sharp corners are explicitly desired). The corresponding curvature plots provide an insight into the second-order behaviour of the scheme. While the scheme does not, in general, produce curvature-continuous results, some of the less challenging configurations of control polygons and normals seem to lead to near-curvature-continuous results. An investigation into how to set the normals to achieve curvature continuity or to at least minimise the curvature jumps is beyond the scope of the present paper.

We now discuss the choice of parameter τ in (6); see Fig. 6 for an example. Other tested input data exhibited similar behaviour. Although τ has little influence on the general shape of the limit curve with the exception of τ values close to one, it has a noticeable effect on curvature behaviour. Low τ values put too much emphasis on outer normals, whereas high τ values start introducing numerical instabilities due to a wide range of values in the system. We have thus set $\tau = 100$ as our default setting. Note that (similar) numerical instabilities can occur when the range of the input data spans many orders of magnitude. This is not limited to our scheme; it applies to all (geometric) subdivision schemes.

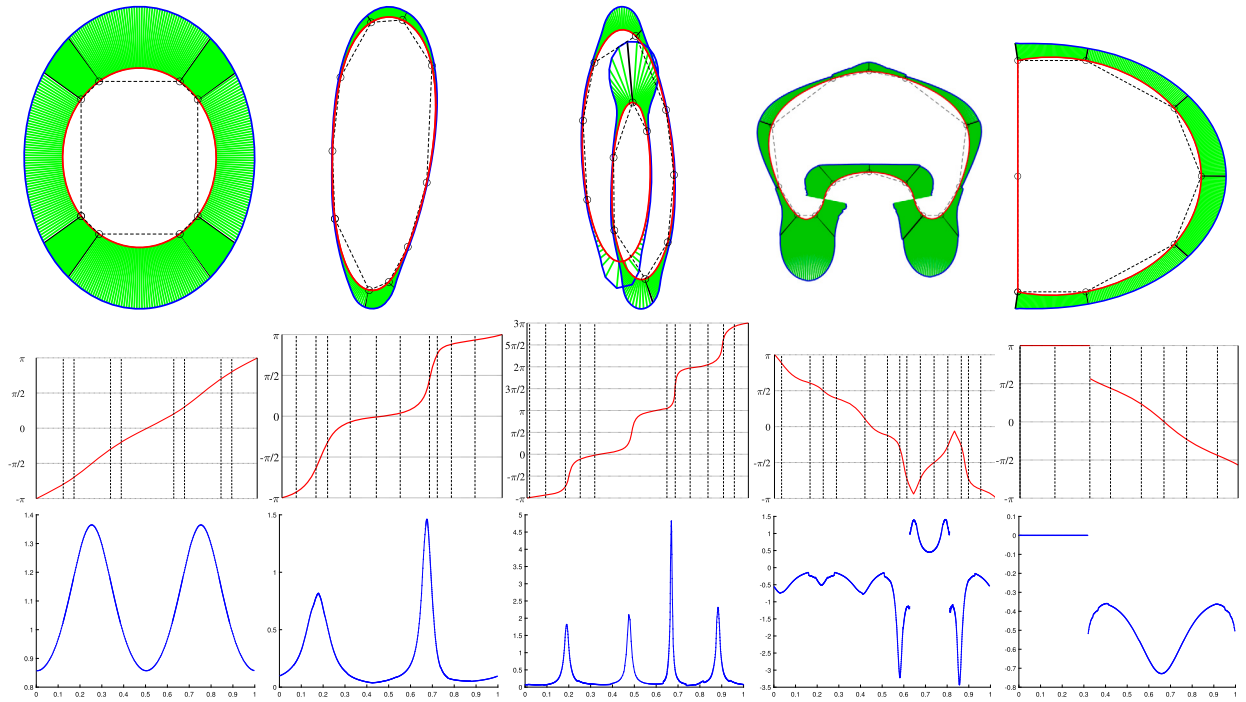

Fig. 7. Top row: Control polygon (dashed black) with control normals (black), limit curve (red; after 6 subdivision steps), and curvature comb (green and blue). Middle row: Angle variation along the subdivision curve. Bottom row: Curvature plot (discrete estimate using an osculating circle passing through three consecutive points). Following Albrecht and Romani (2012), we use counter-clockwise chordal parametrisations along the curves for the angle and curvature plots. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

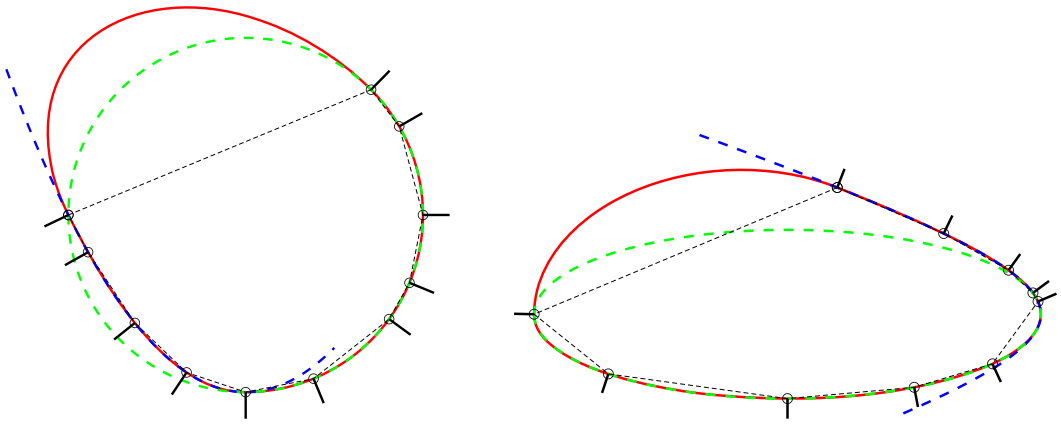

Fig. 8. Preservation of conics. Left: Circle (green) and parabola (blue). Right: Ellipse (green) and hyperbola (blue). Our scheme preserves conics when consecutive point-normal pairs are sampled from them. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

Finally, we focus on the conic-preservation property of our scheme; see Fig. 8. On the left, we show an example with a control polygon with a sequence of points and normals sampled from a circle, followed by a sequence of points and normals coming from a parabola. On the right, points and normals have been sampled from an ellipse and a hyperbola. Both examples confirm that our scheme reproduces conics when sufficiently many consecutive point-normal pairs come from a single conic. They also show that our scheme produces good transitions between conic arcs.

5. Conclusion

We have presented a new conic-preserving, interpolatory subdivision scheme that is based on a simple geometric construction with the advantage of being extendable to the bivariate case. The proposed construction relies on a small local neighbourhood of each edge and allows the user to insert edge points (and associated normals) in such a way that all types of conic sections are preserved.

To the best of our knowledge such an interpolatory subdivision scheme has no competitors in the existing literature. Indeed, none of the univariate schemes proposed so far is able to satisfy all the properties listed above.

In our future work, we plan to attack the difficult problem of establishing formal conditions on G^1 continuity of the scheme with accompanying proofs, in the spirit of Ewald et al. (2015). We also believe that our current handling of inflection points leaves room for improvement. Another focus point could be to improve the efficiency of our implementation, which has so far not been considered. Finally, we will explore the extension of our scheme to the bivariate setting. While this is often difficult for geometric subdivision schemes due to the number of configurations that need to be considered, we believe that our scheme opens the door for a generalisation as the locally fitted conic can be replaced by a locally fitted quadric at a mesh vertex.

CRediT authorship contribution statement

Niels Bügel: Data curation, Investigation, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Lucia Romani:** Conceptualization, Funding acquisition, Investigation, Methodology, Software, Supervision, Visualization, Writing – original draft, Writing – review & editing. **Jiří Kosinka:** Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Software, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by the “Italian National Group for Scientific Calculation” (GNCS - INDAM) and received funds under the 2023 Visiting Professor Program - CUP E53C22001930001. The subdivision scheme is loosely based on the first author’s internship project co-supervised also by Marcello Seri and Roland van der Veen at the University of Groningen.

References

Aihua, M., Jie, L., Jun, C., Guiqing, L., 2016. A new fast normal-based interpolating subdivision scheme by cubic Bézier curves. Vis. Comput. 32 (9), 1085–1095.
Albrecht, G., Romani, L., 2012. Convexity preserving interpolatory subdivision with conic precision. Appl. Math. Comput. 219, 4049–4066.
Badoual, A., Novara, P., Romani, L., Schmitter, D., Unser, M., 2017. A non-stationary subdivision scheme for the construction of deformable models with sphere-like topology. Graph. Models 94, 38–51.
Badoual, A., Romani, L., Unser, M., 2021. Active subdivision surfaces for the semiautomatic segmentation of biomedical volumes. IEEE Trans. Image Process. 30, 5739–5753.
Bebarta, S., Jena, M.K., 2021. Shape preserving Hermite subdivision scheme constructed from quadratic polynomial. Int. J. Appl. Comput. Math. 7, 222.
Bebarta, S., Jena, M.K., 2023. Shape preserving rational [3/2] Hermite interpolatory subdivision scheme. Calcolo 60, 8.
Beccari, C., Casciola, G., Romani, L., 2007. A non-stationary uniform tension controlled interpolating 4-point scheme reproducing conics. Comput. Aided Geom. Des. 24 (1), 1–9.
Beccari, C., Casciola, G., Romani, L., 2009. Shape controlled interpolatory ternary subdivision. Appl. Math. Comput. 215 (3), 916–927.
Birdal, T., Busam, B., Navab, N., Ilic, S., Sturm, P.F., 2020. Generic primitive detection in point clouds using novel minimal quadric fits. IEEE Trans. Pattern Anal. Mach. Intell. 42 (6), 1333–1347.
Cai, Z., 2009. Convexity preservation of the interpolating four-point C^2 ternary stationary subdivision scheme. Comput. Aided Geom. Des. 26 (5), 560–565.
Chalmovianský, P., Jüttler, B., 2007. A non-linear circle-preserving subdivision scheme. Adv. Comput. Math. 27 (4), 375–400.
Conti, C., Romani, L., Unser, M., 2015. Ellipse-preserving Hermite interpolation and subdivision. J. Math. Anal. Appl. 426 (1), 211–227.
Deng, C., Ma, W., 2014. A biarc based subdivision scheme for space curve interpolation. Comput. Aided Geom. Des. 31 (9), 656–673.
Deng, C., Wang, G., 2010. Incenter subdivision scheme for curve interpolation. Comput. Aided Geom. Des. 27 (1), 48–59.
Donat, R., López-Ureña, S., 2019. Nonlinear stationary subdivision schemes reproducing hyperbolic and trigonometric functions. Adv. Comput. Math. 45 (5), 3137–3172.
Dyn, N., Levin, D., Gregory, J.A., 1987. A 4-point interpolatory subdivision scheme for curve design. Comput. Aided Geom. Des. 4 (4), 257–268.
Dyn, N., Levin, D., Liu, D., 1992. Interpolatory convexity-preserving subdivision schemes for curves and surfaces. Comput. Aided Des. 24 (4), 211–216.
Ewald, T., Reif, U., Sabin, M., 2015. Hölder regularity of geometric subdivision schemes. Constr. Approx. 42 (3), 425–458.
Goodman, T.N.T., Ong, B.H., 2005. Shape-preserving interpolation by splines using vector subdivision. Adv. Comput. Math. 22 (1), 49–77.
Harker, M., O’Leary, P., Zsombor-Murray, P., 2008. Direct type-specific conic fitting and eigenvalue bias correction, 15th Annual British Machine Vision Conference. Image Vis. Comput. 26 (3), 372–381.
Hernández-Mederos, V., Estrada-Sarlabous, J., Ivrissimtzis, I., 2013. Generalization of the incenter subdivision scheme. Graph. Models 75 (2), 79–89.
Jena, M.K., 2021. A Hermite interpolatory subdivision scheme constructed from quadratic rational Bernstein–Bézier spline. Math. Comput. Simul. 187, 433–448.
Kuijt, F., Van Damme, R., 2002. Shape preserving interpolatory subdivision schemes for nonuniform data. J. Approx. Theory 114 (1), 1–32.
Lipovetsky, E., Dyn, N., 2016. A weighted binary average of point-normal pairs with application to subdivision schemes. Comput. Aided Geom. Des. 48, 36–48.
Marinov, M., Dyn, N., Levin, D., 2005. Geometrically controlled 4-point interpolatory schemes. In: Dodgson, N.A., Floater, M.S., Sabin, M.A. (Eds.), Advances in Multiresolution for Geometric Modelling. Springer, pp. 301–315.
Novara, P., Romani, L., 2015. Building blocks for designing arbitrarily smooth subdivision schemes with conic precision. J. Comput. Appl. Math. 279, 67–79.
Novara, P., Romani, L., 2018. On the interpolating 5-point ternary subdivision scheme: a revised proof of convexity-preservation and an application-oriented extension. Math. Comput. Simul. 147, 194–209.

Pan, J., Lin, S., Luo, X., 2012. A combined approximating and interpolating subdivision scheme with C^2 continuity. Appl. Math. Lett. 25 (12), 2140–2146.
Peters, J., Reif, U., 2008. Subdivision Surfaces. Springer.
Romani, L., 2009. From approximating subdivision schemes for exponential splines to high-performance interpolating algorithms. J. Comput. Appl. Math. 224 (1), 383–396.
Romani, L., 2010. A circle-preserving C^2 Hermite interpolatory subdivision scheme with tension control. Comput. Aided Geom. Des. 27 (1), 36–47.
Sabin, M., 2010. Analysis and Design of Univariate Subdivision Schemes. Springer.
Sabin, M., Dodgson, N., 2004. A circle-preserving variant of the four-point subdivision scheme. In: Mathematical Methods for Curves and Surfaces. Tromsø 2004.
Song, Q., Zheng, H.C., Peng, G.H., 2013. A nonlinear ternary circle-preserving interpolatory subdivision scheme. Appl. Mech. Mater. 427–429, 2170–2173.
Trettner, P., Kobbelt, L., 2021. Sampling from quadric-based CSG surfaces. Comput. Graph. Forum 40, 41–56.
Warren, J., Weimer, H., 2001. Subdivision Methods for Geometric Design: A Constructive Approach. Morgan Kaufmann Publishers Inc., San Francisco.
Yang, X., 2006. Normal based subdivision scheme for curve design. Comput. Aided Geom. Des. 23 (3), 243–260.
Yang, X., 2023. Point-normal subdivision curves and surfaces. Comput. Aided Geom. Des. 104, 102207.