# Convexity preserving interpolatory subdivision with conic precision

Gudrun Albrecht[a], Lucia Romani[b,*]

[a]Univ. Lille Nord de France, UVHC, LAMAV-CGAO, FR no. 2956, F-59313 Valenciennes, France
[b]Department of Mathematics and Applications, University of Milano-Bicocca, Via R. Cozzi 53, 20125 Milano, Italy

**Abstract**

The paper is concerned with the problem of shape preserving interpolatory subdivision. For arbitrarily spaced, planar input data an efficient non-linear subdivision algorithm reproducing conic sections and respecting the convexity properties of the initial data, is here presented. Significant numerical examples are included to illustrate the effectiveness of the proposed method and the smoothness of the limit curves.

*Keywords:* Subdivision, interpolation, convexity preservation, conic reproduction

## 1. Introduction and state of the art

Subdivision schemes constitute a powerful alternative for the design of curves and surfaces over the widely studied parametric and implicit forms. In fact, they offer a really versatile tool that is, at the same time, very intuitive and easy to use and implement. This is due to the fact that subdivision schemes are defined via iterative algorithms which exploit simple refinement rules to generate denser and denser point sequences that, under appropriate hypotheses, converge to a continuous, and potentially smooth, function.

In the univariate case, the iteration starts with a sequence of points denoted by $\mathbf{p}^0 = (\mathbf{p}_i^0 \ : \ i \in \mathbb{Z})$, attached to the integer grid, and then for any $k \geq 0$ one subsequently computes a sequence $\mathbf{p}^{k+1} = S\,\mathbf{p}^k$, where $S : \ell(\mathbb{Z}) \to \ell(\mathbb{Z})$ identifies the so-called subdivision operator.

Subdivision operators can be broadly classified into two main categories: *interpolating* and *approximating* [1, 2]. Interpolating schemes are required to generate limit curves passing through all the vertices of the given polyline $\mathbf{p}^0$ (note that throughout the paper the terms "polygon" and "polyline" are used as an alternative to refer to point sequences). Therefore they are featured by refinement rules maintaining the points generated at each step of the recursion in all the successive iterations. Approximating schemes, instead, are not required to match the original position of vertices on the assigned polyline $\mathbf{p}^0$ and thus they adjust their positions aiming at very smooth and visually pleasing limit shapes that approximate it. As a consequence, while in the case of *approximating* subdivision the newly generated vertices are not on the limit shape, in the case of *interpolatory* subdivision, in every iteration a finer data set $\mathbf{p}^{k+1}$ is obtained by taking the old data values $\mathbf{p}^k$ and inserting new points in between them, so that the limit curve not only interpolates the initial set of points but also all the points generated through the whole process. Every such new point is calculated using a finite number of existing, usually neighboring points. In particular, if the computation of the new points is carried out through a linear subdivision operator $S$, the scheme is said to be *linear*, otherwise *non-linear*. Then, inside the above identified categories, the schemes can also be further classified. More specifically, the refinement rules of a scheme can be distinguished between *stationary* (when they do not alter from level to level) and *non-stationary*; between *uniform* (when they do not vary from point to point) and *non-uniform*; between *binary* (when they double the number of points at each iteration) and *N-ary*, namely of arity $N > 2$.

Most of the univariate subdivision schemes studied in the literature are binary, uniform, stationary and linear. These characteristics, in fact, make it easier to study the mathematical properties of the limit curve, but seriously limit the applications of the scheme. Exceptions from a binary, or a uniform, or a stationary, or a linear approach, have

*Corresponding author.
*Email addresses:* gudrun.albrecht@univ-valenciennes.fr (Gudrun Albrecht), lucia.romani@unimib.it (Lucia Romani)

already appeared (see for example [3] and references therein), but none of the methods proposed so far provides an interpolatory algorithm that can fulfill the list of *all* fundamental features considered essential in applications. These features can briefly be summarized as:

(i) generating a visually-pleasing limit curve which faithfully mimics the behaviour of the underlying polyline without creating unwanted oscillations;

(ii) preserving the shape, i.e., the convexity properties of the given data;

(iii) identifying geometric primitives like circles and more generally conic sections, the starting polyline had been sampled from, and reproducing them.

Requirement (i) derives from the fact that, despite interpolating schemes being considered very well-suited for handling practical models to meet industrial needs (due to their evident link with the initial configuration of points representing the object to be designed), compared to their approximating counterparts, they are more difficult to control and tend to produce bulges and unwanted folds when the initial data are not uniformly spaced. Recently this problem has been addressed by using non-uniform refinement rules [4, 5] opportunely designed to take into account the irregular distribution of the data. But, despite their established merit of providing visually pleasing results, there is no guarantee that such methods are convexity-preserving, i.e., that if a convex data set is given, a convex interpolating curve can be obtained. This is due to the fact that, such non-uniform schemes are linear and, as it is well-known [6], linear refinement operators that are $C^1$ cannot preserve convexity in general.

The property (ii) of convexity preservation is of great practical importance in modelling curves and surfaces tailored to industrial design (e.g. related to car, aeroplane or ship modelling where convexity is imposed by technical and physical conditions as well as by aesthetic requirements). In fact, if shape information such as convexity is not enforced, interpolatory curves, though smooth, may not be satisfactory as they may contain redundant wiggles and bumps rather than those suggested by the data points, i.e., they feature unacceptable visual artifacts. Preserving convexity, while a curve is interpolating a given data set, is far from trivial. But much progress has been made in this field, evidence of which is given by the recent burgeoning literature. In most publications, the introduction of subdivision schemes fulfilling requirement (ii) has been achieved through the definition of non-linear refinement rules. In fact, although linear subdivision schemes turn out to be simple to implement, easy to analyze and affine invariant, they have many difficulties to control the shape of the limit curve and avoid artifacts and undesired inflexions that usually occur when the starting polygon $\mathbf{p}^0$ is made of highly non-uniform edges. Non-linear schemes, instead, offer effective algorithms to be used in shape-preserving data interpolation [6–12].

On the basis of the well–known, linear Dubuc-Deslauriers interpolatory 4-point scheme [13], for example, several non-linear analogues have been presented in order to accomplish at least one of the three above listed properties. On the one hand, non-linear modifications of the classical 4-point scheme have been introduced to reduce the oscillations that usually occur in the limit curve when applying the refinement algorithm to polylines with short and long adjacent edges. These have been presented in [14] and [15], and as concerns the case of convexity-preserving strategies (which are the ones capable of completely eliminating the artifacts arising during the subdivision process), we find the papers [16] and [11]. On the other hand, for the purpose of enriching the Dubuc-Deslauriers 4-point scheme with the property (iii) of geometric primitives preservation, a non-linear 4-point scheme reproducing circles and reducing curvature variation for data off the circle, has been defined [17]. With the same intent, another modification of the classical 4-point scheme in a non-linear fashion, had been given in [18].

With these papers, the theoretical investigation of non-linear interpolatory subdivision has only begun. A lot is still to be done, in particular as concerns the use of non-linear rules for reproducing salient curves other than circles, considered of fundamental importance in several applications. So far, it has been shown that non-linear updating formulas can be used in the definition of non-stationary subdivision schemes aimed at reproducing polynomials and some common transcendental functions. In particular, [19] respectively [20–25] present subdivision algorithms that turn out to be circle-preserving respectively able to exactly represent any conic section. While the first is able to guarantee reproduction starting from given samples with any arbitrary spacing, for the latter ones the property of conic precision is confined to the case of equally-spaced samples. Most recently a shape and circle preserving scheme for any arbitrary data points has been presented in [7].

Therefore, an outstanding issue that should be considered is the possibility of defining an interpolatory subdivision scheme that is at the same time shape-preserving and artifact free, as well as capable of generating conic sections starting from any arbitrarily-spaced samples coming from a conic. This is exactly the purpose of this paper. Based on an approximation order four strategy presented in [26] for estimating tangents to planar convex data sequences, we propose a convexity-preserving interpolatory subdivision scheme with conic precision. This turns out to be a new kind of non-linear and geometry-driven subdivision method for curve interpolation.

The remainder of the paper is organized as follows. In Section 2 we start by describing the refinement strategy, which relies on a classical cross-ratio property for conic sections and uses the tangent estimator from [26], for the case of globally convex data. In Section 3 we adapt the scheme to general, not necessarily convex data by segmenting the given polygon into linear and globally convex segments, and in Section 4 we summarize the whole subdivision algorithm in all its steps. Section 5 contains proofs for the scheme's shape preservation and conic reproducing properties, as well as a proof for the $C^0$ continuity of its limit curve. Moreover, a conjecture for the $C^1$ continuity is also included. Section 6 is devoted to illustrating the scheme by several significant application examples, and we conclude in Section 7.

## 2. Definition of the scheme for globally convex data

In this section we define a convexity preserving subdivision scheme for globally convex data which will then be the basis for the final shape preserving subdivision algorithm for general data.

Curve subdivision schemes iteratively apply a subdivision operator $S$ to a starting point sequence $\mathbf{p}^0 = (\mathbf{p}_i^0 \,:\, i \in \mathbb{Z})$ yielding a new sequence $\mathbf{p}^{k+1} = S\,\mathbf{p}^k$ for any level $k \geq 0$. Our scheme, being interpolatory, has refinement rules of the following form:

$$\begin{aligned} \mathbf{p}_{2i}^{k+1} &= \mathbf{p}_i^k, \\ \mathbf{p}_{2i+1}^{k+1} &= \varphi(\mathbf{p}_{i-\nu}^k, ..., \mathbf{p}_i^k, \mathbf{p}_{i+1}^k, ..., \mathbf{p}_{i+\nu}^k, \mathbf{p}_{i+\nu+1}^k;\ \mathbf{p}) \end{aligned} \tag{1}$$

where $\nu = 2$ is the number of points taken into account in the left and right hand neighborhoods of the segment $\mathbf{p}_i^k \mathbf{p}_{i+1}^k$ in order to define the newly inserted vertex $\mathbf{p}_{2i+1}^{k+1}$, and $\mathbf{p}$ is a parameter point specified later. $\varphi$ is a non linear function, which we will define in form of an algorithm.

In order to detail the idea at the basis of this convexity–preserving scheme, we thus consider the following problem where, for simplicity, we omit the upper indices $k$.

**Problem 1.** *Given $n$ points $\mathbf{p}_i((p_i)_x, (p_i)_y)$, $i = 1, \ldots, n$ ($n \geq 5$), in convex position in the affine plane, we wish to obtain one new point $\mathbf{u}_i$ related to the $i$-th edge $\mathbf{p}_i\,\mathbf{p}_{i+1}$.*

We will carry out the construction of the new points in the projectively extended affine plane. To this end we denote the projective counterparts of the affine points $\mathbf{p}_i((p_i)_x, (p_i)_y)$, $i = 1, \ldots, n$ by

$$P_i(p_{i,0}, p_{i,1}, p_{i,2}) \quad \text{where} \quad p_{i,0} = 1,\ p_{i,1} = (p_i)_x,\ p_{i,2} = (p_i)_y\,. \tag{2}$$

By projective geometry's principle of duality, a line $L$ may be represented either by a linear equation $l_0x_0 + l_1x_1 + l_2x_2 = 0$ in variable *point coordinates* $(x_0, x_1, x_2)$ or by a triple $(l_0, l_1, l_2)$ of constant *line coordinates*. The line coordinates of the line $L(l_0, l_1, l_2)$ joining two points $X_1(x_{1,0}, x_{1,1}, x_{1,2})$ and $X_2(x_{2,0}, x_{2,1}, x_{2,2})$ may simply be calculated by the vector product $X_1 \wedge X_2 = L$. In the same way, the point coordinates of the intersection point $P(p_0, p_1, p_2)$ of two lines $L_1(l_{1,0}, l_{1,1}, l_{1,2})$ and $L_2(l_{2,0}, l_{2,1}, l_{2,2})$ is obtained as $L_1 \wedge L_2 = P$. This elegant way of calculating line intersections and point connections is well known from projective geometry, see, e.g., [26, 27]. Without loss of generality we apply a normalization to the homogeneous point coordinates such that $x_0 \in \{0, 1\}$ for all calculated points.

We use these notions of projective geometry in the following preprocessing procedure in order to preserve global convexity of the input point set.

In every given point $P_i$ we estimate a tangent from a subset of five points (including the point $P_i$) by the conic tangent estimator presented in [26]. If the given points represent a closed polygon, then the five-point subset is composed of the point $P_i$, its preceding two points $P_{i-1}, P_{i-2}$ as well as its successive two points $P_{i+1}, P_{i+2}$ (by considering

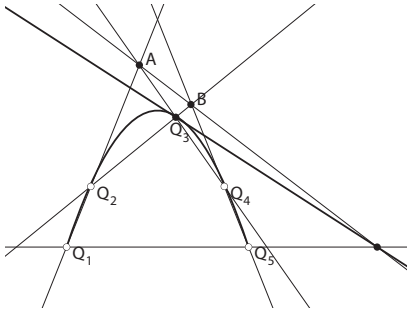



Figure 1: Illustration of the tangent construction in $Q_3$.

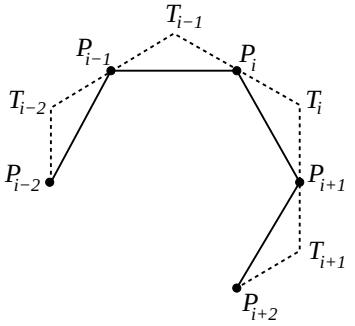



Figure 2: Convex delimiting polygon (dashed line) for a given convex data set (solid line).

$P_0 = P_n$, $P_{-1} = P_{n-1}$, $P_{n+1} = P_1$, $P_{n+2} = P_2$). If the given points represent an open polygon, then for $i = 3, \ldots, n-2$ we proceed as above and for $i \in \{1, 2\}$ (respectively $i \in \{n-1, n\}$) the points $\{P_1, P_2, P_3, P_4, P_5\}$ (respectively $\{P_{n-4}, P_{n-3}, P_{n-2}, P_{n-1}, P_n\}$) are taken.

In order to apply the conic tangent estimator from [26] we locally rename the five points around $P_i$ by $Q_3 = P_i$, and by arbitrarily mapping the remaining four points to $Q_1, Q_2, Q_4, Q_5$ by a one–to–one map. The desired tangent in the point $Q_3$ is then calculated by the formula (see [26])

$$M_{33} := Q_3 \wedge (M_{15} \wedge (A \wedge B)), \tag{3}$$

where $M_{ij} = Q_i \wedge Q_j$ for $i \neq j$, $A = M_{12} \wedge M_{34}$ and $B = M_{54} \wedge M_{32}$. See Figure 1 for an illustration.

We then denote the obtained line in the point $P_i$ by $L_i$, and intersect every two consecutive lines generating the intersection points $T_i = L_i \wedge L_{i+1}$, see Figure 2.

The dashed lines in Figure 2 constitute a convex delimiting polygon for the new points generated in the next subdivision level. If the initial points come from a conic section, the constructed lines $L_i$ are the tangents to this conic in the respective points $P_i$. Otherwise the lines $L_i$ approximate the tangents with approximation order 4, see [26].

After this preprocessing step we now get back to the initial subdivision Problem 1, i.e., between every two points $P_i$ and $P_{i+1}$ insert a new point $U_i$ by applying a classical result from projective geometry which, for the readers convenience, we recall in the following Proposition (see, e.g., [28–30]).

**Proposition 1.** *a) Let $X, E, E_0, E_1$ be four points of a projective line $\mathcal{P}_1$, where the points $E, E_0, E_1$ are mutually distinct, and let $(x_0, x_1)$ be the projective coordinates of the point $X$ with respect to the projective coordinate system $\{E_0, E_1; E\}$ of $\mathcal{P}_1$. Then, the cross ratio $cr(X, E, E_0, E_1)$ of the four points $X, E, E_0, E_1$ in this order is defined by $cr(X, E, E_0, E_1) = \frac{x_1}{x_0}$.*

*b) Let $P_1, P_2$ be two points on a conic section $C$, and $t_1, t_2$ the tangents of $C$ in $P_1, P_2$ respectively, and let $T$ be the intersection point of $t_1$ and $t_2$ ($T = t_1 \cap t_2$). Then, the point $T$ and the line $P_1P_2$ are pole and polar with respect to the conic $C$. If we further denote the intersection points of any line $l_T$ through $T$ with the conic $C$ by $P$ and $U$, and the intersection point of $l_T$ and $T$'s polar $P_1P_2$ by $X$ ($l_T \cap C = \{P, U\}$, $l_T \cap P_1P_2 = X$), then $cr(U, P, X, T) = -1$ and the four points $(U, P, X, T)$ are said to be in harmonic position.*

Therefore, chosen any point $P$ on the conic section $C$, Proposition 1 gives us the means of constructing a point $U$ from the known collinear points $P, X, T$ such that the harmonic cross ratio condition for conic sections is satisfied, i.e. $U$ is a point between $P_1$ and $P_2$ that lies on the conic section $C$. For an illustration see Figure 3.

More precisely, according to Proposition 1, for any pair of points $P_i$, $P_{i+1}$ in the given polyline, we need to choose another point $P_{j_i}$ (that we call *parameter point*) in order to get a new point $U_i$ inside the triangle $\Delta(P_i, T_i, P_{i+1})$, with $T_i = L_i \wedge L_{i+1}$. For the choice of the parameter point $P_{j_i}$ the whole region bounded by the lines $L_i$, $L_{i+1}$ and $P_iP_{i+1}$ containing the given convex polygon (see Figure 4) is suitable. In particular, any point $P_{j_i}$ of the given convex polygon can be taken as parameter point (with exception of $P_i$ and $P_{i+1}$), and since such a choice guarantees the reproduction of conic sections whenever the given points are samples coming from a conic, we opt for it.

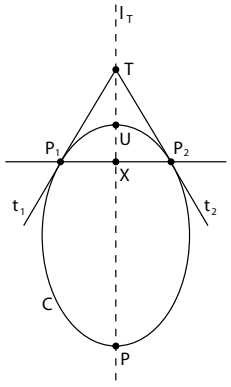



Figure 3: Harmonic cross ratio condition for conic sections.

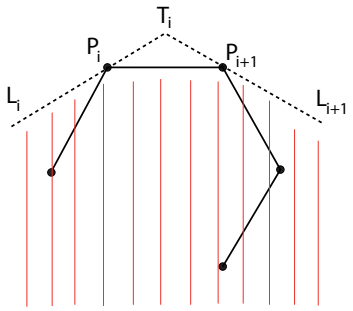



Figure 4: Suitable region for the choice of the parameter point $P_{j_i}$.

Let us thus denote by $X_i$ the intersection point of the lines $P_iP_{i+1}$ and $P_{j_i}T_i$ for a chosen $j_i \in \{1,\dots,n\} \setminus \{i, i+1\}$, i.e., $X_i = P_iP_{i+1} \cap P_{j_i}T_i$.

In order to guarantee a regular distribution of the inserted points we propose to choose the index $j_i$ by the following angle criterion. Let $\mathbf{m}_i$ be the midpoint of each segment $\mathbf{p}_i\mathbf{p}_{i+1}$. Then let $g_i = \mathbf{t}_i\mathbf{m}_i$, $h_{ij} = \mathbf{t}_i\mathbf{p}_j$ be the connecting lines of the points $\mathbf{t}_i$ and $\mathbf{m}_i$ respectively $\mathbf{t}_i$ and $\mathbf{p}_j$, and $\alpha^i_j = \angle(g_i, h_{ij})$ the angle between these two lines (the smaller one of the two complementary angles is taken in each case) for $j = i+2,\dots,n+i-1$ by considering $\mathbf{p}_{n+r} = \mathbf{p}_r$ for $r \geq 1$. For every $i \in \{1,\dots,n\}$ in the case of a closed polygon and for every $i \in \{1,\dots,n-1\}$ in the case of an open polygon, we then obtain a value $j_i$ from the condition

$$\alpha^i_{j_i} = \min_{j=i+2,\dots,n+i-1} \alpha^i_j. \tag{4}$$

Once the point $P_{j_i}$ has been selected by exploiting the illustrated criterion, we then establish the projective coordinate system $\{X_i, T_i; P_{j_i}\}$ on the straight line $P_{j_i}T_i$ by calculating the projective representatives of $X_i$ and $T_i$ by solving the linear equation system $\gamma_i X_i + \mu_i T_i = P_{j_i}$ for $\gamma_i$ and $\mu_i$. We obtain $\gamma_i = D_{i,1}/D_i$ , $\mu_i = D_{i,2}/D_i$, where

$$D_i = \det\begin{pmatrix} x_{i,l} & t_{i,l} \\ x_{i,m} & t_{i,m} \end{pmatrix},\ D_{i,1} = \det\begin{pmatrix} p_{j_i,l} & t_{i,l} \\ p_{j_i,m} & t_{i,m} \end{pmatrix},\ D_{i,2} = \det\begin{pmatrix} x_{i,l} & p_{j_i,l} \\ x_{i,m} & p_{j_i,m} \end{pmatrix}, \qquad l \neq m \in \{0,1,2\}. \tag{5}$$

Let us recall that in the above matrices $y_{i,k}$ are the coordinates of the projective counterpart $Y_i$ of a point $\mathbf{y}_i$, see (2). For details on projective coordinate systems see, e.g, [28]. By Proposition 1 the point $U_i$ is thus obtained as $U_i = D_{i,1}X_i - D_{i,2}T_i$.

## 3. Modification of the scheme for non-convex data

In Section 2 we have described the fundamental steps of our new interpolatory subdivision scheme in the case of globally convex data. This section is devoted to an accurate illustration of the modification of the scheme in case of general, non-convex data.

If the input data are not convex, the idea is to segment them according to the following criteria in order to obtain a sequence of rectilinear and piecewise convex segments.

First of all, consecutive collinear points are identified in the following way.

| We check if the inner angle between any two consecutive segments of the given polyline is close to 180° within a given threshold and, if this is the case, we select the outer end points of these collinear segments as the delimiting points of a sequence of collinear vertices. Let us denote them by $\mathbf{p}^0_j$ and $\mathbf{p}^0_l$. | (6) |
|---|---|

Since in CAD applications linear features are usually intentional, we do not smooth the angle between a subpolygon consisting of (at least 3) collinear points and its neighbors. The insertion rule for these straight line subpolygons between the points $\mathbf{p}_j^k$ and $\mathbf{p}_l^k$ ($k = 0, 1, 2, 3, \ldots$) simply reads as:

$$\mathbf{p}_{2i+1}^{k+1} = \frac{1}{2}(\mathbf{p}_i^k + \mathbf{p}_{i+1}^k), \; i = j, \ldots, l-1. \tag{7}$$

The remaining subpolygons no longer contain collinear segments. For these remaining subpolygons, inflection edges are identified by the following criterion, see also [7, 31].

An edge $\mathbf{p}_i^0\mathbf{p}_{i+1}^0$ is identified as *inflection edge* if the points $\mathbf{p}_{i-1}^0$ and $\mathbf{p}_{i+2}^0$ lie in different half planes with respect to it. From a computational viewpoint, this translates into calculating $v_l = (p_{i-1}^0)_y - (p_i^0)_y + ((p_i^0)_x - (p_{i-1}^0)_x)\frac{(p_{i+1}^0)_y-(p_i^0)_y}{(p_{i+1}^0)_x-(p_i^0)_x}$, $v_r = (p_{i+2}^0)_y - (p_i^0)_y + ((p_i^0)_x - (p_{i+2}^0)_x)\frac{(p_{i+1}^0)_y-(p_i^0)_y}{(p_{i+1}^0)_x-(p_i^0)_x}$ and checking if $\text{sign}(v_l\, v_r)$ is negative. (8)

On an inflection edge we insert a new point, e.g. the midpoint of the edge corners, and we split the polygon at the newly inserted point. In the sequel we refer to such kind of inserted junction point as *inflection point*. Notice that, once all inflection points have been inserted, we have a sequence of subpolygons without inflections.

Last, we check each of these subpolygons for global convexity (since by construction, see (6), the considered subpolygons no longer contain collinear points) by means of the following criterion:

If for every edge of the subpolygon all the points of the subpolygon lie either on the edge or in the same half plane with respect to the edge, then the subpolygon is *globally convex*, otherwise it is not globally convex. (9)

Since in case of not globally convex subpolygons the tangent estimation in (3) does not produce a convex delimiting polygon, we need to divide the given subpolygon into two new consecutive convex subpolygons that satisfy criterion (9) and repeat the procedure until we have a sequence of globally convex subpolygons, each of which we suppose to be composed of at least five points. If this is not the case we insert additional points in the following way. Note that two cases are possible, depending if the globally convex subpolygon consists of either three or four points, see Figure 5.

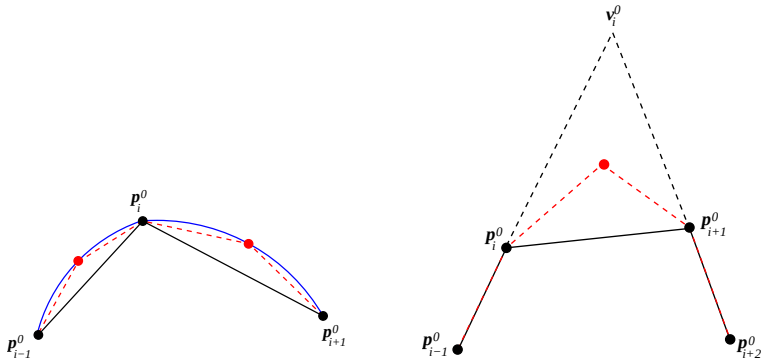



Figure 5: Rules to define an initial globally convex subpolygon of five vertices starting from three (left) or four (right) given points.

In the case of three points $\mathbf{p}_{i-1}^0$, $\mathbf{p}_i^0$, $\mathbf{p}_{i+1}^0$, we insert the two new points by sampling the parametric quadratic curve interpolating the given points at the centripetal parameter values $\{\tau_{i-1}, \tau_i, \tau_{i+1}\}$ at the intermediate parameter values $\frac{\tau_{i-1}+\tau_i}{2}$ and $\frac{\tau_i+\tau_{i+1}}{2}$. In the case of four points $\mathbf{p}_{i-1}^0, \ldots, \mathbf{p}_{i+2}^0$ we choose as fifth point the barycenter of the triangle $\Delta(\mathbf{p}_i^0, \mathbf{v}_i^0, \mathbf{p}_{i+1}^0)$, where $\mathbf{v}_i^0 = \mathbf{p}_{i-1}^0\mathbf{p}_i^0 \cap \mathbf{p}_{i+1}^0\mathbf{p}_{i+2}^0$.

In the sequel we refer to the vertex where two globally convex subpolygons meet as *convex junction point*. We point out that it is necessary to introduce these points because the triangles of the convex delimiting polygon around a

convex junction point might not correctly be positioned, see Figure 10 (a), (b). We let the reader notice that any point of the given polygon that identifies two consecutive globally convex subpolygons is a good candidate to be a convex junction point. So the user can arbitrarily select this point, taking into account that the visual quality of the curve is only very slightly influenced by the selection of different candidates.

In conclusion, the result of the proposed segmentation process is:

| an initial polygon $\mathbf{p}^0$ made of straight line segments (consisting of at least three points) as well as globally convex segments, the latter ones separated by inflection points or convex junction points and consisting of at least five points each. | (10) |
|---|---|

Whereas in the interior of every globally convex subpolygon we apply the tangent estimation algorithm detailed in the previous section, the next subsections are devoted to the description of the method we use to compute tangents in inflection points and convex junction points.

### *3.1. Tangent computation in inflection points*

In the case of an *inflection point* $\mathbf{p}_i^0$ let us denote the line of the inflection edge $\mathbf{p}_{i-1}^0\mathbf{p}_i^0\mathbf{p}_{i+1}^0$ by $\mathbf{e}_i$, and the convex subpolygons meeting in $\mathbf{p}_i^0$ by $\mathbf{s}_{l,i}^0$ and $\mathbf{s}_{r,i}^0$. We estimate a left and a right tangent in $\mathbf{p}_i^0$, $\mathbf{l}_{l,i}^0$ and $\mathbf{l}_{r,i}^0$ respectively, by applying the tangent estimation method from [26] to the five points $\mathbf{p}_{i-4}^0, \ldots, \mathbf{p}_{i-1}^0, \mathbf{p}_i^0$ of polygon $\mathbf{s}_{l,i}^0$, respectively $\mathbf{p}_i^0, \ldots, \mathbf{p}_{i+3}^0, \mathbf{p}_{i+4}^0$ of polygon $\mathbf{s}_{r,i}^0$. We then combine these two lines $\mathbf{l}_{l,i}^0$ and $\mathbf{l}_{r,i}^0$ for defining an initial tangent $\mathbf{l}_i^0$ in $\mathbf{p}_i^0$ (see Figure 6) as follows:

$$\mathbf{l}_i^0 = \lambda_i^0 \mathbf{l}_{l,i}^0 + (1-\lambda_i^0)\mathbf{l}_{r,i}^0\,, \qquad \lambda_i^0 \in (0,1). \tag{11}$$

In all the computational examples presented in this paper we have chosen $\lambda_i^0 = \frac{1}{2}$. This choice of $\lambda_i^0$ defines $\mathbf{l}_i^0$ as the bisector of the angle between $\mathbf{l}_{l,i}^0$ and $\mathbf{l}_{r,i}^0$, and yields a symmetric behavior of the curve around the inflection point.

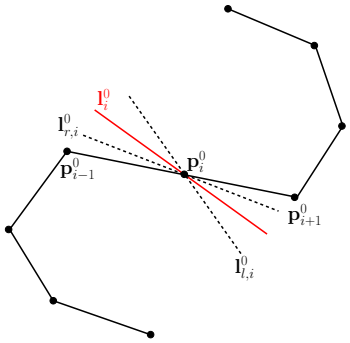



Figure 6: Definition of an initial tangent $\mathbf{l}_i^0$ in an inflection point $\mathbf{p}_i^0$.

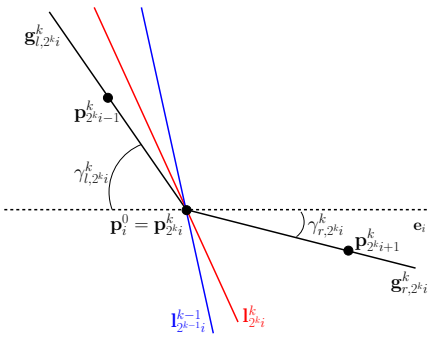



Figure 7: Tangent definition in an inflection point.

The tangents in the other vertices of the polygons $\mathbf{s}_{l,i}^0$ and $\mathbf{s}_{r,i}^0$, and thus their new vertices, are calculated as described in the previous section by treating $\mathbf{s}_{l,i}^0$ and $\mathbf{s}_{r,i}^0$ separately. In this way we obtain the new polygons $\mathbf{s}_{l,i}^1$ and $\mathbf{s}_{r,i}^1$. Let us now describe how to compute the tangents $\mathbf{l}_{2^k i}^k$ in the point $\mathbf{p}_i^0 = \mathbf{p}_{2^k i}^k$ in the following iterations ($k = 1, 2, 3, \ldots$), see Figure 7 for an illustration. Let

$$\mathbf{g}_{l,2^k i}^k = \mathbf{p}_{2^k i-1}^k \mathbf{p}_{2^k i}^k \;\text{ and }\; \mathbf{g}_{r,2^k i}^k = \mathbf{p}_{2^k i}^k \mathbf{p}_{2^k i+1}^k \tag{12}$$

be the edges that are incident in $\mathbf{p}_i^0 = \mathbf{p}_{2^k i}^k$, and $\gamma_{r,2^k i}^k = \angle(\mathbf{g}_{r,2^k i}^k, \mathbf{e}_i)$ and $\gamma_{l,2^k i}^k = \angle(\mathbf{g}_{l,2^k i}^k, \mathbf{e}_i)$ their respective angles with the initial inflection edge $\mathbf{e}_i$. We then define the line $\mathbf{g}_{2^k i}^k$ by choosing that one of the lines $\mathbf{g}_{l,2^k i}^k$ and $\mathbf{g}_{r,2^k i}^k$ from (12) yielding the maximum angle

$$\gamma_{2^k i}^k = \max\{\gamma_{l,2^k i}^k, \gamma_{r,2^k i}^k\}\,. \tag{13}$$

The tangent $\mathbf{l}_{2^k i}^k$ in the point $\mathbf{p}_i^0 = \mathbf{p}_{2^k i}^k$ is then defined as

$$\mathbf{l}^k_{2^k i} = \lambda^k_{2^k i}\mathbf{l}^{k-1}_{2^{k-1} i} + (1-\lambda^k_{2^k i})\mathbf{g}^k_{2^k i}\,, \qquad \lambda^k_{2^k i} \in (0,1). \tag{14}$$

In all the computational examples presented in this paper we have chosen $\lambda^k_{2^k i} = \frac{1}{2}$ for all $k > 0$, so that $\mathbf{l}^k_{2^k i}$ identifies the bisector of the angle formed by the lines $\mathbf{l}^{k-1}_{2^{k-1} i}$ and $\mathbf{g}^k_{2^k i}$, and guarantees a symmetric behavior of the curve around the inflection point.
In the other vertices of $\mathbf{s}^k_{l,i}$ and $\mathbf{s}^k_{r,i}$ we proceed as in Section 2 for estimating the tangents; this allows us to calculate the new polygons $\mathbf{s}^{k+1}_{l,i}$ and $\mathbf{s}^{k+1}_{r,i}$ by separately applying the "convex" procedure from the previous section.

In Figure 8 we show an application example of the presented subdivision algorithm in the case of a given starting polygon containing an inflection point.

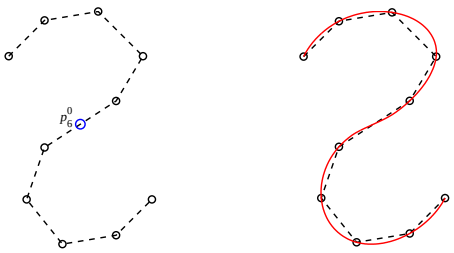



Figure 8: Application of the subdivision algorithm in case of a polyline with an inflection point ($\mathbf{p}^0_6$): given polyline (left); refined polyline after 6 steps of the subdivision algorithm (right).

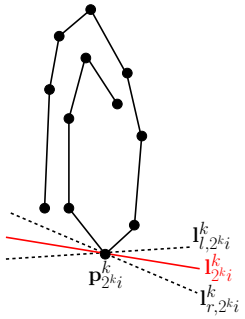



Figure 9: Tangent definition in a convex junction point.

### *3.2. Tangent computation in convex junction points*

Let us denote the convex subpolygons meeting in $\mathbf{p}^0_i$ by $\mathbf{s}^0_{l,i}$ and $\mathbf{s}^0_{r,i}$. As in the case of an inflection point we estimate a left and a right tangent in $\mathbf{p}^0_i$, $\mathbf{l}^0_{l,i}$ and $\mathbf{l}^0_{r,i}$ respectively, by applying the tangent estimation method from [26] to the five points $\mathbf{p}^0_{i-4}, \ldots, \mathbf{p}^0_{i-1}, \mathbf{p}^0_i$ of polygon $\mathbf{s}^0_{l,i}$, respectively $\mathbf{p}^0_i, \ldots, \mathbf{p}^0_{i+3}, \mathbf{p}^0_{i+4}$ of polygon $\mathbf{s}^0_{r,i}$. If the points $\mathbf{p}^0_{i-1}$ and $\mathbf{p}^0_{i+1}$ lie in different half planes with respect to $\mathbf{l}^0_{l,i}$ ($\mathbf{l}^0_{r,i}$ respectively) we replace $\mathbf{l}^0_{l,i}$ ($\mathbf{l}^0_{r,i}$ respectively) by the line $\mathbf{p}^0_i\mathbf{p}^0_{i+1}$ ($\mathbf{p}^0_{i-1}\mathbf{p}^0_i$ respectively). We then combine these two lines $\mathbf{l}^0_{l,i}$ and $\mathbf{l}^0_{r,i}$ for defining an initial tangent $\mathbf{l}^0_i = \lambda^0_i\mathbf{l}^0_{l,i} + (1-\lambda^0_i)\mathbf{l}^0_{r,i}$ in $\mathbf{p}^0_i$. The tangents in the other vertices of the polygons $\mathbf{s}^0_{l,i}$ and $\mathbf{s}^0_{r,i}$, and thus their new vertices, are calculated as described in the previous section by treating $\mathbf{s}^0_{l,i}$ and $\mathbf{s}^0_{r,i}$ separately. In this way we obtain the new polygons $\mathbf{s}^1_{l,i}$ and $\mathbf{s}^1_{r,i}$. We then iterate this procedure and in all the following iterations ($k = 1, 2, 3, \ldots$) we compute the tangents $\mathbf{l}^k_{2^k i}$ in the point $\mathbf{p}^0_i = \mathbf{p}^k_{2^k i}$ as

$$\mathbf{l}^k_{2^k i} = \lambda^k_{2^k i}\mathbf{l}^k_{l,2^k i} + (1-\lambda^k_{2^k i})\mathbf{l}^k_{r,2^k i}\,, \qquad \lambda^k_{2^k i} \in (0,1) \tag{15}$$

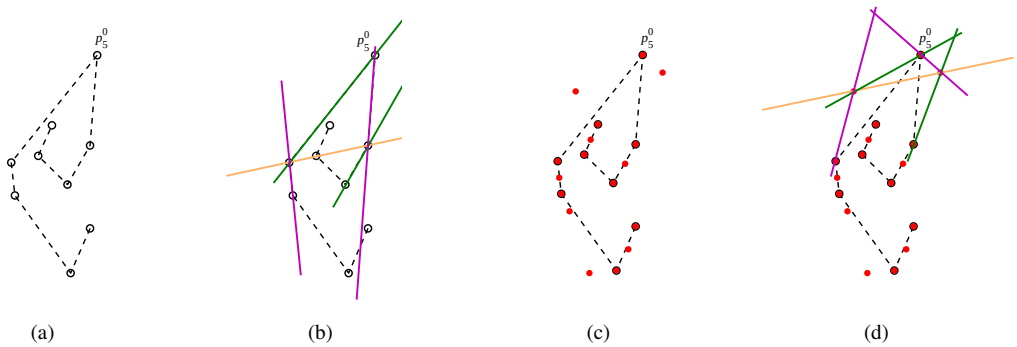
where $\mathbf{l}^k_{l,2^k i}$ and $\mathbf{l}^k_{r,2^k i}$ is the respective left and right tangent in $\mathbf{p}^k_{2^k i}$ (see Figure 9 for an illustration). In all the computational examples presented in this paper we have chosen $\lambda^k_{2^k i} = \frac{1}{2}$ for all $k \geq 0$.

Note that, by the nature of the subdivision rules, after a few steps and in all remaining iterations, the following condition is satisfied in a convex junction point $\mathbf{p}^0_i = \mathbf{p}^k_{2^k i}$ for $k \geq k_0$ (see Figure 10):

| |
|---|
| the intersection points $\mathbf{p}^k_{2^k i-2}\mathbf{p}^k_{2^k i-1} \cap \mathbf{p}^k_{2^k i}\mathbf{p}^k_{2^k i+1}$ and $\mathbf{p}^k_{2^k i-1}\mathbf{p}^k_{2^k i} \cap \mathbf{p}^k_{2^k i+1}\mathbf{p}^k_{2^k i+2}$ lie in the same half plane with respect to the line $\mathbf{p}^k_{2^k i-1}\mathbf{p}^k_{2^k i+1}$ as the point $\mathbf{p}^0_i$. |

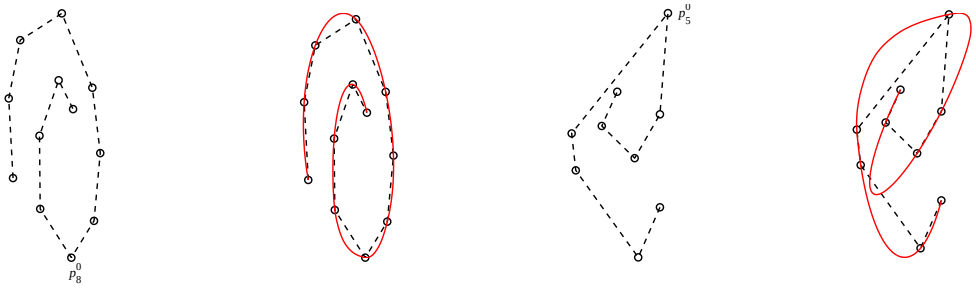
(16)

Application examples of the above described subdivision algorithm in the case of a starting polygon with a convex junction point are shown in Figure 11.




(a) (b) (c) (d)

Figure 10: Illustration of condition (16): (a) initial polyline with convex junction point $\mathbf{p}_5^0$; (b) condition (16) not satisfied for $k = 0$; (c) situation after one refinement step ($k = 1$, $\mathbf{p}_5^0 = \mathbf{p}_{10}^1$); (d) condition (16) satisfied for $k = 1$.




(a) (b) (c) (d)

Figure 11: Application of the subdivision algorithm to a polygon with a convex junction point: (a) given polygon with junction in $\mathbf{p}_8^0$; (b) refined polygon obtained from the data set in (a) after 6 steps of the algorithm; (c) given polygon with junction in $\mathbf{p}_5^0$; (d) refined polygon obtained from the data set in (c) after 6 steps of the algorithm.

## 4. The subdivision algorithm

The basic subdivision scheme with its generalization previously detailed in Sections 2 and 3, may be summarized in the following algorithm, which we apply to the given point sequence $\mathbf{p}^0 = (\mathbf{p}_i^0 \ : \ i \in \mathbb{Z})$. For the sake of clarity we will first describe the procedure to be used in the case of globally convex data (Subsection 4.1), which constitutes an essential ingredient of the final algorithm for general non-convex data presented in Subsection 4.2.

### *4.1. Algorithm for globally-convex data*

Let $\mathbf{p}^k = (\mathbf{p}_i^k \ : \ i = 1, \ldots, n_k)$ be the vertices of the globally convex polygon at the $k$-level refinement. Hereafter we denote by $P_i^k$ the projective counterparts of the affine points $\mathbf{p}_i^k$, see Section 2.
The algorithm that implements the function $\varphi$ from (1) proceeds as follows in order to calculate the points $\mathbf{p}_{2i+1}^{k+1} = \varphi(\mathbf{p}_{i-2}^k, \mathbf{p}_{i-1}^k, \mathbf{p}_i^k, \mathbf{p}_{i+1}^k, \mathbf{p}_{i+2}^k, \mathbf{p}_{i+3}^k \ ; \ \mathbf{p})$.

*Step 1:* Preprocessing

a) For a closed polygon we set $P_0^k = P_{n_k}^k$, $P_{-1}^k = P_{n_k-1}^k$, $P_{n_k+1}^k = P_1^k$, $P_{n_k+2}^k = P_2^k$.
For an open polygon we set $P_0^k = P_5^k$, $P_{-1}^k = P_4^k$, $P_{n_k+1}^k = P_{n_k-4}^k$, $P_{n_k+2}^k = P_{n_k-3}^k$.
For $i = 1, \ldots, n_k$ we then assign $Q_1 = P_{i-2}^k$, $Q_2 = P_{i-1}^k$, $Q_3 = P_i^k$, $Q_4 = P_{i+1}^k$, $Q_5 = P_{i+2}^k$, and apply formula (3) for obtaining the tangent $L_i^k$ in $P_i^k$. Please note that, according to [26], we can arbitrarily number the five points for calculating the tangent at any one of them.

b) According to the angle criterion, we calculate the angles $\alpha_j^i$ for $j = i+2, \ldots, n_k + i - 1$ and for $i = 1, \ldots, n_k$ in the case of a closed polygon, while for $i = 1, \ldots, n_k - 1$ in the case of an open polygon. We then select the value $\alpha_{j_i}^i$ satisfying condition (4).

*Step 2:* For a closed polygon we set $L_{n_k+1}^k = L_1^k$ and we consider $i = 1, \ldots, n_k$, whereas for an open polygon we consider $i = 1, \ldots, n_k - 1$. We thus calculate the intersection points $T_i^k = L_i^k \wedge L_{i+1}^k$.

*Step 3:* Calculate the lines (for $i = 1, \ldots, n_k$ for a closed polygon, and for $i = 1, \ldots, n_k - 1$ for an open polygon) $N_i^k = P_i^k \wedge P_{i+1}^k$, $\Lambda_i^k = P_{j_i}^k \wedge T_i^k$, as well as their intersection points $X_i^k = N_i^k \wedge \Lambda_i^k$.

*Step 4:* Calculate the points $P_{2i+1}^{k+1}$ as $P_{2i+1}^{k+1} = D_{i,1}^k X_i^k - D_{i,2}^k T_i^k$ with $D_{i,1}^k, D_{i,2}^k$ according to (5).

*4.2. Algorithm for non-convex data*

*Step 1:* Preprocessing
We preprocess the data according to the criteria (6), (8) and (9) thus identifying subpolygons contained in a straight line and introducing and /or identifying inflection vertices and convex junction vertices within the initial data points of the remaining subpolygons yielding a sequence composed of globally convex subpolygons and "straight line" subpolygons as described in (10).

*Step 2:* For a globally convex subpolygon $(\mathbf{p}_j^0, \ldots, \mathbf{p}_l^0) = \ldots = (\mathbf{p}_{2^k j}^k, \ldots, \mathbf{p}_{2^k l}^k)$ (where $l \geq j + 4$) we apply the algorithm of Subsection 4.1 (Step 1, open case) in order to calculate the tangents in its points. In inflection points and convex junction points tangents are computed by applying rule (14) and (15) respectively. Finally in transition points to straight line segments we fix the tangents equal to the straight line subpolygon.

*Step 3:* In order to compute the new points for a straight line subpolygon we apply the insertion rule (7), while for globally convex subpolygons we apply the procedure of Subsection 4.1.

The algorithm has been implemented in Matlab, and its most expensive parts are steps 1-4 of subsection 4.1 which employ the following number of operations (for each refinement level):

- The computation of each tangent $L_i^k$ in step 1.a) requires 10 cross products, which correspond to 30 additions/subtractions and 60 multiplications/divisions.
- The computation of each angle $\alpha_j^i$ in step 1.b) requires 9 additions/subtractions, 10 multiplications/divisions and 2 square roots.
- The computation of the point $P_{2i+1}^{k+1}$ in step 4 requires 27 additions/subtractions and 50 multiplications/divisions.

## 5. Properties of the scheme

In this section we will be showing that the presented scheme is

- shape preserving,
- conic reproducing,
- convergent to a continuous limit curve.

Moreover, we will present numerical tests to support a $C^1$ continuity conjecture on the limit curves. In fact, since the proposed scheme is non-linear and of a new geometry-driven type, it cannot be analyzed by existing techniques.

*5.1. Theoretical analysis*

We start this section by proving the shape preserving and conic reproducing properties of the proposed subdivision scheme.

**Proposition 2.** *For an initial polygon* $\mathbf{p}^0$ *that satisfies condition (10), the presented interpolatory subdivision scheme produces a limit curve that preserves the shape of the initial data, i.e., if the starting point sequence* $\mathbf{p}^0$ *consists of convex, straight line and concave segments, then all generated subsequent point sequences* $\mathbf{p}^k$*, for* $k \geq 1$*, respect the same behavior.*

*Proof.* By construction straight line segments are reproduced. The convexity of an initial data set is preserved by inserting the new points $\mathbf{p}_{2i+1}^{k+1}$ in the triangles $\Delta(\mathbf{p}_i^k, \mathbf{t}_i^k, \mathbf{p}_{i+1}^k)$ of the convex delimiting polygon with vertices $\mathbf{p}_i^k$, $\mathbf{t}_i^k$. □

**Proposition 3.** *The presented interpolatory subdivision scheme reproduces conic sections, i.e., if the data come from a conic section* $C$*, then the limit curve coincides with* $C$*.*

*Proof.* If the point sequence $\mathbf{p}^k$ comes from a conic section $C$, then the lines $\mathbf{l}_i^k$ generated in the preprocessing step are the tangents of $C$ in the points $\mathbf{p}_i^k$. Since the parameter points $\mathbf{p}_{j_i}^k$ come from the conic $C$, by Proposition 1 the constructed points $\mathbf{p}_{2i+1}^{k+1}$ lie on $C$. □

**Remark 1.** *If the data come from a curve composed by contiguous conic sections our scheme reproduces the single conics in the case where the individual conic segments can be identified upfront and every point sequence corresponding to a single conic is treated separately. Otherwise, if the whole polyline is processed all together the inner parts of the single conics are reproduced, but next to the junctions we have slight deviations. See Figure 12 for an illustration.*

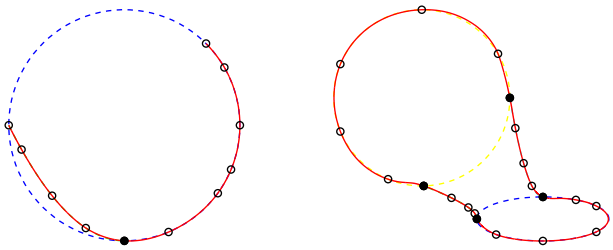


Figure 12: Result of the proposed subdivision algorithm when applied to a polyline including a parabolic and a circular arc (left), and contiguous conic sections of different type, namely parabolic, circular, hyperbolic and elliptic arcs (right). Full small circles are used to identify vertices separating the different pieces of conics drawn with dashed lines.

As concerns the proof for $C^0$ continuity, we start by considering the case where the starting polygon $\mathbf{p}^0$ is globally convex, which means that it might contain convex junction points but no inflection points. In the convex junction points we suppose condition (16) to be satisfied from a certain iteration level $k_0 \geq 0$ onwards. Thus, denoted by $\mathbf{q}_i^k$ the intersection point of the edges $\mathbf{p}_{i-1}^k\mathbf{p}_i^k$ and $\mathbf{p}_{i+1}^k\mathbf{p}_{i+2}^k$, it follows by construction that the point $\mathbf{t}_i^k$ lies inside the triangle $\Delta(\mathbf{p}_i^k, \mathbf{q}_i^k, \mathbf{p}_{i+1}^k)$ for $k \geq k_0$ and the point $\mathbf{p}_{2i+1}^{k+1}$ lies inside the triangle $\Delta(\mathbf{p}_i^k, \mathbf{t}_i^k, \mathbf{p}_{i+1}^k)$. Denoted by $\mathbf{p}^k$ the polygon with vertices $\mathbf{p}_i^k$ and by $\mathbf{q}^k$ the polygon with vertices $\mathbf{q}_i^k$, we can formulate the following:

**Proposition 4.** *If the starting polygon* $\mathbf{p}^0$ *is convex, then all the polygons* $\mathbf{p}^k$ *are convex and they are bounded by the polygon* $\mathbf{q}^{k_0}$*.*

*Proof.* Being $\mathbf{p}^0$ a convex polygon, then, by construction (see Proposition 2), this property is transferred from level $k$ to level $k+1$ for all $k \geq 0$. Thus each $\mathbf{p}^k$ is convex. Moreover, the triangles $\Delta(\mathbf{p}_{2i}^{k+1}, \mathbf{q}_{2i}^{k+1}, \mathbf{p}_{2i+1}^{k+1})$ and $\Delta(\mathbf{p}_{2i+1}^{k+1}, \mathbf{q}_{2i+1}^{k+1}, \mathbf{p}_{2i+2}^{k+1})$ lie inside the triangle $\Delta(\mathbf{p}_i^k, \mathbf{q}_i^k, \mathbf{p}_{i+1}^k)$ for all $k \geq k_0$. Therefore the polygon $\mathbf{q}^{k_0}$ is an outer bound of all polygons $\mathbf{p}^k$. □

Since by construction $\mathbf{p}^k \subset \mathbf{p}^{k+1}$, the sequence of polygons $\{\mathbf{p}^k\}$ is a monotone and bounded sequence of convex polygons yielding the following proposition.

**Proposition 5.** *If the starting polygon* $\mathbf{p}^0$ *is convex, then* $\lim_{k\to\infty} \mathbf{p}^k$ *exists and is a* $C^0$ *convex curve.*

For the proof of this result we refer the reader to [8, Prop. 2].

We now turn our attention to non-convex starting polygons $\mathbf{p}^0$. As described in Section 3, given an initial non-convex polygon $\mathbf{p}^0$, we identify straight line segments, introduce inflection points and thus treat every convex subsegment separately as a standalone convex polygon in the subdivision procedure. Since both in the transition points to straight line segments and in the inflection points the limit curve is $C^0$ by construction, with Proposition 5 we obtain the following result.

**Corollary 1.** *The limit of the presented interpolatory subdivision scheme is a* $C^0$ *curve.*

*5.2. Experimental analysis of smoothness*

As explained in [11], for non-linear schemes the conventional tools for checking the regularity of the limit curves are not applicable. Thus we exploit experimental methods for analyzing the smoothness properties of the proposed scheme.
Following the work done in the recent papers [11, 15, 17, 31], we present two numerical tests which strongly suggest that the limit curves generated by the proposed subdivision scheme are $C^1$ continuous with respect to chord length parameterization.

In our first test we let $\theta_i^k$ be the angle of the local, inward-pointing unit normal vector with respect to the direction $(1, 0)$

$$\mathbf{n}_i^k = (\cos(\theta_i^k), \sin(\theta_i^k)) \perp (\mathbf{p}_{i+1}^k - \mathbf{p}_i^k) \tag{17}$$

and let $\theta^k : \mathbb{R} \to \mathbb{R}$ be the piecewise linear function that interpolates the data $(\tilde{t}_i^k, \theta_i^k)$, with $\tilde{t}_i^k$ denoting the chordal parameter values associated with the points $\mathbf{p}_i^k$.

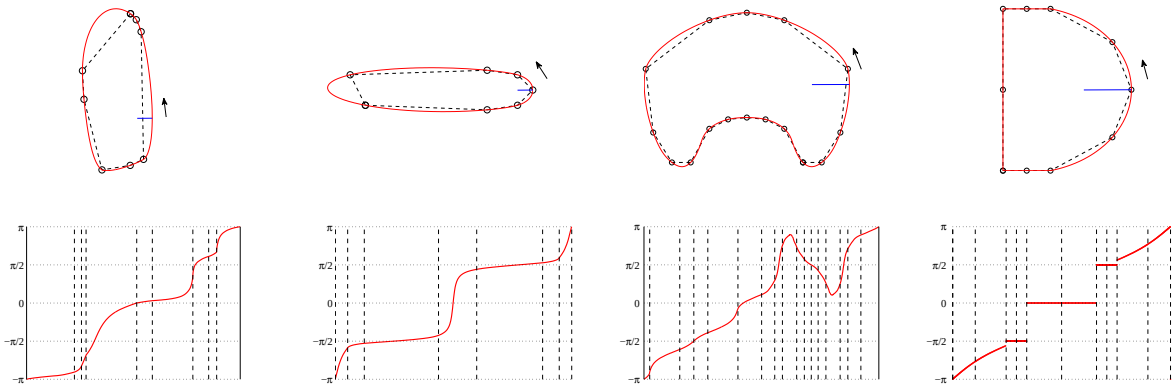



Figure 13: First row: refined polylines displayed after 6 subdivision steps. Second row: plot of normal angle over arc length for all the examples in top row.

For any initial data not containing straight line segments, the sequence $\{\theta^k\}_{k\geq 0}$ converges to a continuous limit $\theta$, as shown in Figure 13. Here each plot starts at the rightmost point of the curve where the normal is (-1,0) and follows the curve in counterclockwise orientation, as indicated by the arrow. We assume the positive $\theta_i^k$ to be the counterclockwise angle from the vector $(1, 0)$ to $\mathbf{n}_i^k$. Moreover, to better understand the normal angle behaviour, we insert in all the plots dashed vertical lines referring to the positions of the initial polyline vertices. We observe that, only in the last picture, the normal angle curve is discontinuous in correspondence to the transition points to straight line segments where we know the limit curve being $C^0$ by construction and on purpose.
From the results obtained by applying this first test on a wide range of examples, we can conclude that the sequence

of the piecewise linear functions $\tilde{\mathbf{f}}^{k,[1]}$ which interpolate the data $(\tilde{t}_i^k, \tilde{\mathbf{p}}_i^{k,[1]})$, with $\tilde{\mathbf{p}}_i^{k,[1]}$ denoting the first divided differences with respect to the chordal parameter values $\tilde{t}_i^k$,

$$\tilde{\mathbf{p}}_i^{k,[1]} = \frac{\mathbf{p}_{i+1}^k - \mathbf{p}_i^k}{\tilde{t}_{i+1}^k - \tilde{t}_i^k} = (-\sin(\theta_i^k), \cos(\theta_i^k)), \tag{18}$$

converges to a continuous limit, namely $\tilde{\mathbf{f}}^{[1]} = (-\sin(\theta), \cos(\theta))$.
In the second test we show that the lengths of the difference vectors $\boldsymbol{\delta}_i^{k,[1]} = \tilde{\mathbf{p}}_{i+1}^{k,[1]} - \tilde{\mathbf{p}}_i^{k,[1]}$ converge to 0 as the refinement level $k$ increases, which is a necessary condition for the uniform convergence of the functions $\tilde{\mathbf{f}}^{k,[1]}$. As we can see from Figure 14, in all our tests, the maximum length $\Delta^{k,[1]} = \sup_{i\in\mathbb{Z}} \|\boldsymbol{\delta}_i^{k,[1]}\|_2$ decreases towards 0 when $k$ increases, but we observe that, in some examples (see for instance the second example in Figure 14), it may happen that in the first few subdivision steps $\Delta^{k,[1]}$ increases before finally going to 0, which suggests that it might be hard to establish a general proof for $C^1$ continuity.

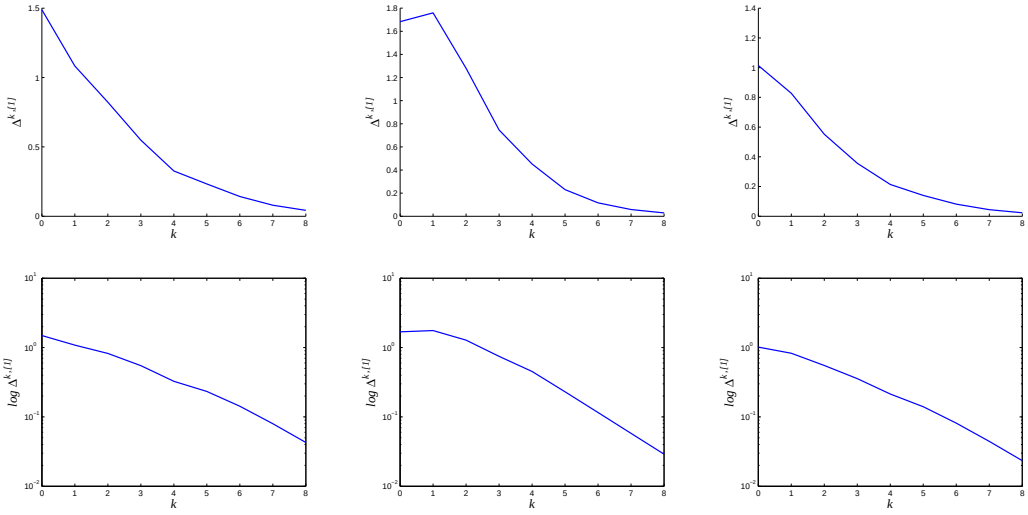



Figure 14: Maximum difference of the first divided differences with respect to chordal parameterization over the number of iterations for the first three examples in Figure 13. First row: plots of $\Delta^{k,[1]}$; second row: log-plots of $\Delta^{k,[1]}$.

## 6. Numerical examples

Subdivision curves are visualized by drawing a highly refined polyline which gives the impression of being visually smooth. For a high quality rendering, the requirement is therefore to reach a level of refinement which guarantees a visually sufficient approximation of the limit shape. The subdivision algorithm presented in Section 4 provides a process of global refinement at every level. Therefore, when the starting polyline is highly non-uniform (namely it is made of adjacent edges with highly non-uniform lengths) the required level of refinement is determined by those locations which approximate the limit curve most unfavorably. Obviously, these may cause unnecessary fine subdivisions at other locations of the curve, thus leading to an unreasonable resource demanding algorithm. To overcome this problem, in the case of highly non–uniform data we propose a uniformizing variant of our subdivision algorithm: when starting from a highly non-uniform polyline, the same refinement rules presented before are applied at each step to introduce new points only where needed, in order to produce a sequence of refined polylines having vertices with a more and more regular distribution. In this way we apply the mechanism of subdivision only at those locations of the starting polyline that are not approximated with the desired visual quality. The decision where a higher resolution refinement is needed, strongly depends on the underlying application. As concerns our algorithm, it may be controlled either by the user or by an automatic criterion. In fact, the user may specify which portions of the polygon should

be subdivided or the process may be automated by controlling whether the length of an edge is greater or not than a specified threshold. Only in the positive case we insert a new point in correspondence to the considered edge. As a consequence, this version of the scheme strongly reduces the computational cost of the algorithm (for instance, the last shape in Figure 20 may be represented either by a refined polyline connecting 1153 points obtained after 6 steps of the basic algorithm, or by only 257 points produced via its uniformizing variant).
As concerns the forthcoming examples, we start by applying the subdivision algorithm of Subsection 4.1 to globally convex closed and open polylines with nearly uniform edges (Figure 15, first row), and successively we exploit the uniformizing version of the scheme in the case of polylines with highly non-uniform edges (Figure 15, second row). As it appears, the generated curves are always convex and visually pleasing. Moreover, in Figure 16, we illustrate the conic precision property of the proposed algorithm focussing our attention on very general non-uniform examples.
We then continue by showing that the curves computed by the algorithm presented in Subsection 4.1 are really artifact free. In fact, although a limit curve can be apparently artifact free, it is hard to tell from the display on the screen if it is acceptable or not. Two curves may look very similar on the screen, but their curvature plots may reveal important differences. The most commonly used tool for revealing significant shape differences is provided by the curvature comb of the curve. In pictures 17–19 we have used the graphs of the discrete curvature combs of the refined polylines to show that the limit curves generated by our algorithm are indeed artifact free. In particular, if we compare the results we get by refining the polyline in Figures 19(a) and 17(c) (representing data that do not come from a conic section) through our uniformizing algorithm and through the subdivision algorithms in [4], [15] and [17] (Figures 19(b), (c) and (d) respectively, as well as Figures 18(a), (b) and (c) left images respectively), they are only apparently very similar. Yet their curvature combs reveal substantial differences showing that neither every non-linear nor every non-uniform subdivision scheme is indeed artifact free (see Figures 19 (e)-(f)-(g)-(h) and 17(c)-18(a)-(b)-(c) right images). Whereas Figures 17(a)-(b) and 19(e) suggest an even higher order smoothness of the presented algorithm, Figure 17(c), right image, shows that the smoothness is in fact limited to $C^1$.

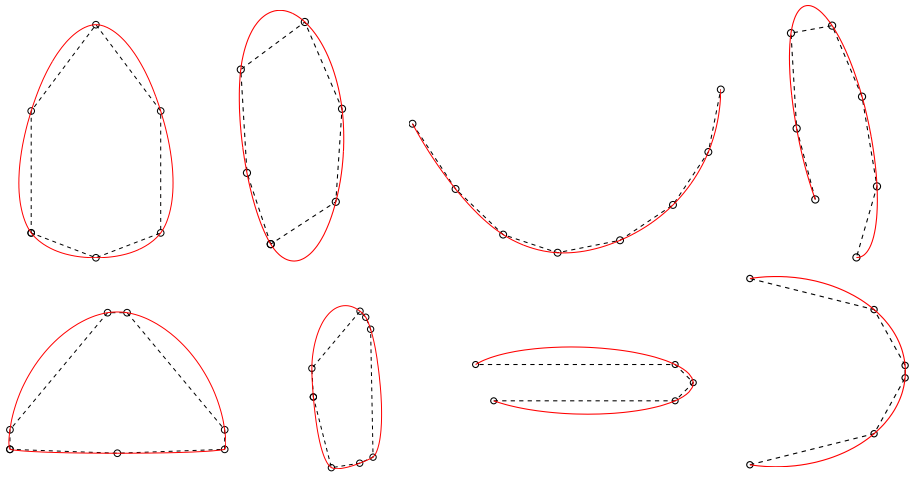


Figure 15: Application examples of the subdivision algorithm of Subsection 4.1 (first row) and its uniformizing variant (second row). The displayed results have been obtained after 6 steps of the algorithm.

In the following we proceed by illustrating the results of the subdivision algorithm of Subsection 4.2 and its uniformizing version. In the first example of Figure 20 we take the D-shape polyline of [7] in order to show the ability of the scheme to reproduce collinear vertices. In the second and third examples we apply the subdivision scheme to an open, respectively closed, sequence of non-convex data to illustrate its smoothness and shape-preserving interpolation properties.
In Figure 21 we show the results of the uniformizing variant of the subdivision algorithm of Subsection 4.2 when applied to a sequence of non-convex data containing a convex subpolygon with less than five vertices (center and right pictures). In this case the convex subpolygon is preprocessed via the procedure described in Figure 5 such that it contains five initial points.

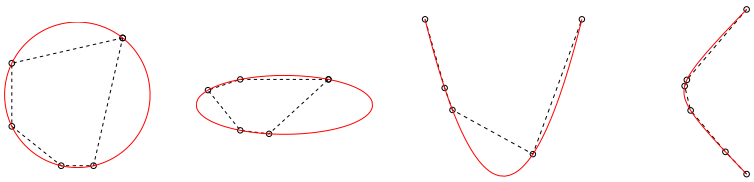
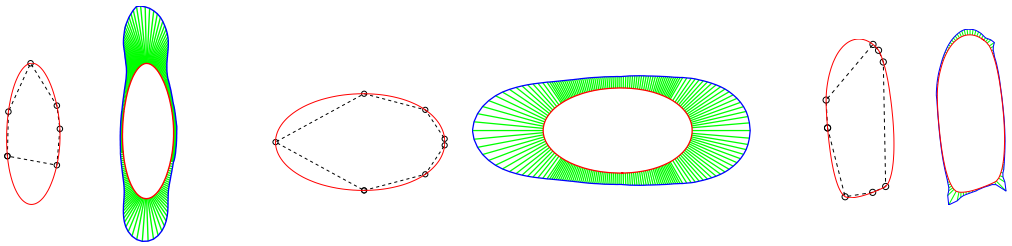
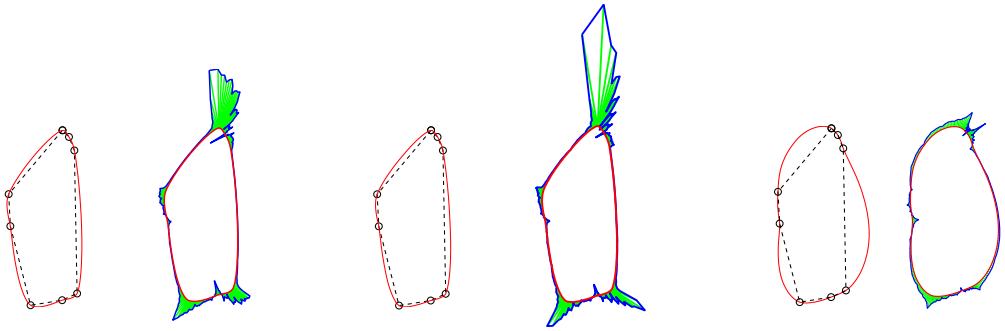


Figure 16: Reproduction of conic sections from non-equispaced samples by applying the uniformizing version of the subdivision algorithm of Subsection 4.1. The displayed results have been obtained after 6 steps of the algorithm.



(a) (b) (c)

Figure 17: Application examples of the subdivision algorithm of Subsection 4.1 (a) and its uniformizing variant (b, c) to globally convex closed polylines that do not come from a conic section, with the corresponding curvature combs.



(a) (b) (c)

Figure 18: Application example of the subdivision algorithms in [4, 15] (a, b) using centripetal parameterization and that in [17] (c), with the corresponding curvature combs.

We close this section by showing the performance of the algorithm on real world data. In order to illustrate its versatility we apply the scheme to two rather different application scenarios. On the one hand we consider profile curves from mechanical engineering, see Figures 22 and 23 for an airfoil and a gear. On the other hand we look at profile curves of so called "organic" shapes used for freeform object design, see Figure 24 for the cover of a mobile phone, Micky Mouse head and a bottle opener. The models in the first two rows of Figure 24 contain several convex segments where most of them have been sampled from conic sections. The shape in the third row is made of 4 independent closed polygons, some of which are non-convex. The data of the first and third example are courtesy of the CAD company think3 (www.think3.com).

For the big variety of models considered in this section, the proposed subdivision scheme turns out to work very well and clearly manifests its versatility as well as its characteristic features of (i) reproducing conic sections, (ii) preserving convexity of the data and (iii) producing smooth curves.

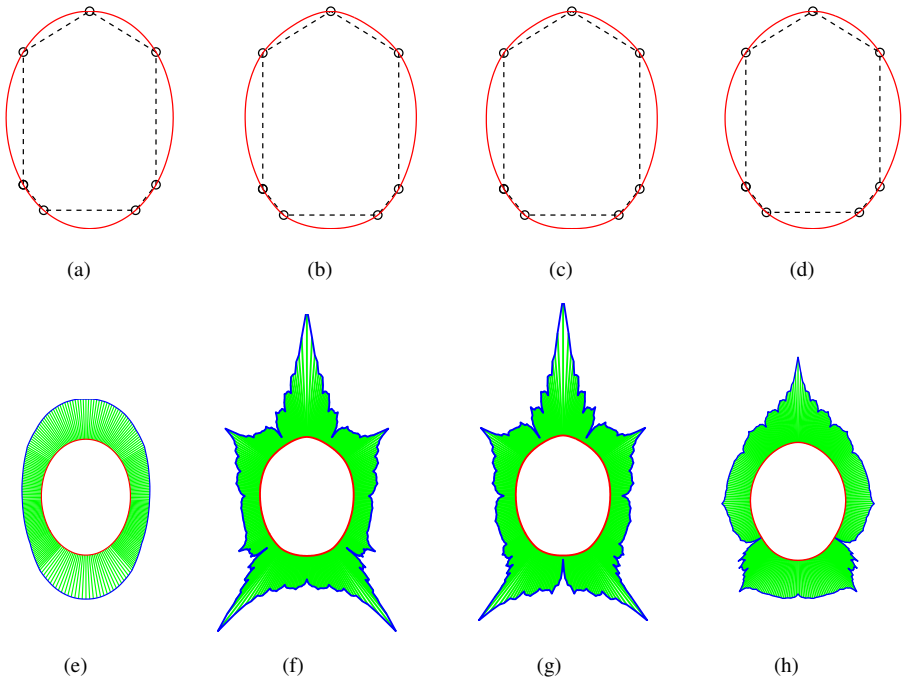



Figure 19: Comparison with the linear, non-uniform scheme in [4] and with the non-linear schemes in [15] and [17]. First row: refined polylines obtained after 6 steps of (a) our uniformizing algorithm; (b) algorithm in [4] with chord length parameterization; (c) algorithm in [15] with chord length parameterization; (d) algorithm in [17]. Second row: corresponding curvature combs.

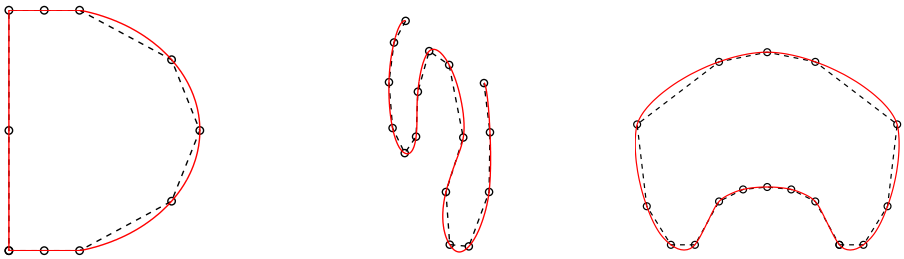


Figure 20: Refined polylines after 6 steps of the subdivision algorithm of Subsection 4.2 (left and center) and its uniformizing variant (right).

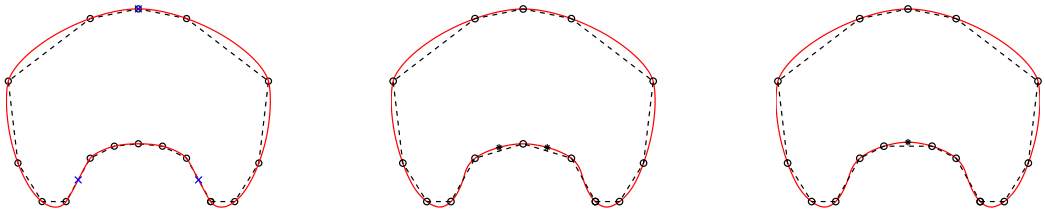


Figure 21: Examples of application of the uniformizing variant of the subdivision algorithm of Subsection 4.2 to a piecewise globally convex polyline (dashed line) with a convex subpolygon (confined between the two inflection points marked by the blue crosses in the lower part of the subfigure on the left) consisting of five points (left) and less than five points (three and four respectively in the center and right subfigures). Black stars depict the initial vertices inserted by the procedure described in Section 3.

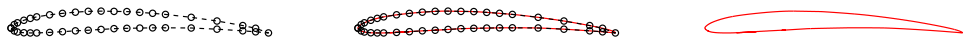


Figure 22: Application example of the subdivision algorithm of Subsection 4.2 for the reconstruction of an airfoil. Left: starting polyline taken from the UIUC Airfoil Coordinates Database (www.ae.illinois.edu/m-selig/ads/coord_database.html). Center and Right: refined polyline obtained after 7 steps of our algorithm.

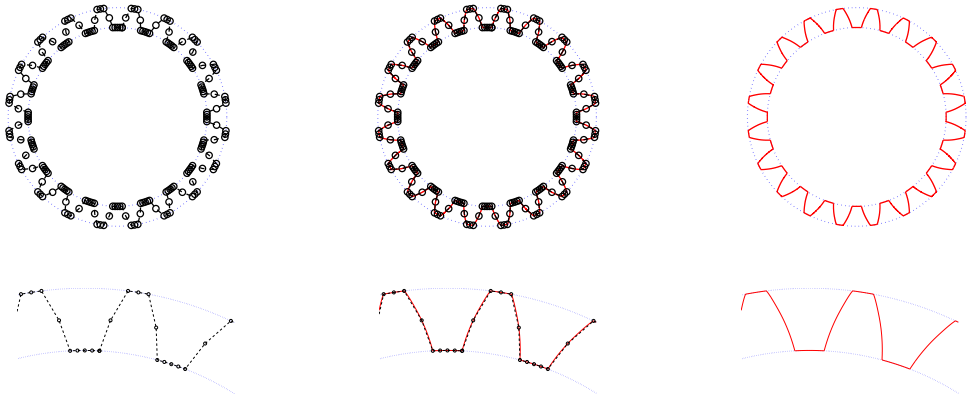


Figure 23: Application example of the subdivision algorithm of Subsection 4.2 for the reconstruction of a 20 tooth gear. First line: starting polyline taken from the CAD drawings library www.traceparts.com (left) and refined polyline obtained after 7 steps of our algorithm (center and right). Second line: zoom of the pictures in the first line.

## 7. Conclusions and ongoing research

Even though in the last years important steps forward have been taken both in the construction and analysis of interpolatory subdivision schemes [3], several problems are still open and need to be tackled in order to increase the strength and popularity of subdivision in more and more fields of application.

First of all, unlike the non-interpolatory subdivision schemes, the interpolatory ones usually generate shapes of inferior quality because, if applied to points with an irregular distribution, they provide a limit curve with more convexity changes than the starting polygon. Since, in several applications it is important to guarantee shape preservation, in this paper we have described a new interpolating subdivision algorithm enjoying this important property.

Because in CAGD it is also often necessary to have schemes able to generate classical geometric shapes, we have enriched our interpolating subdivision scheme with the capability of including the exact representation of all conic sections. The different methods of the literature [20–25] give solutions only when the assigned points have a regular distribution. These linear subdivision schemes use non-stationary refinement rules associated with functional spaces defined via the union between polynomial and exponential functions with a free parameter.

The idea we have explored in this paper is to provide a subdivision scheme in which the conic section reproduction is obtained by adaptive geometric constructions on the given points. The advantage of doing this is that the presented non–linear scheme is able to adapt itself to any data configuration, i.e., to arbitrary irregularly distributed point sequences. Due to the underlying construction, the properties of shape preservation and conic reproduction follow straightforwardly. The method has been illustrated by several significant examples which strongly suggest that the limit curves are $C^1$. Due to the acknowledged difficulty of the problem, $C^1$ continuity is only conjectured and supported by a wide range of numerical tests. A detailed smoothness analysis of the proposed subdivision scheme is a topic of ongoing research since it requires the development of more general theoretical tools that so far are not available in the literature.

At hand of a big variety of application examples, the method shows excellent performance and high versatility. To the best of our knowledge this is the first interpolatory subdivision scheme combining the properties of reproducing conics, preserving convexity and producing smooth and pleasing shapes.

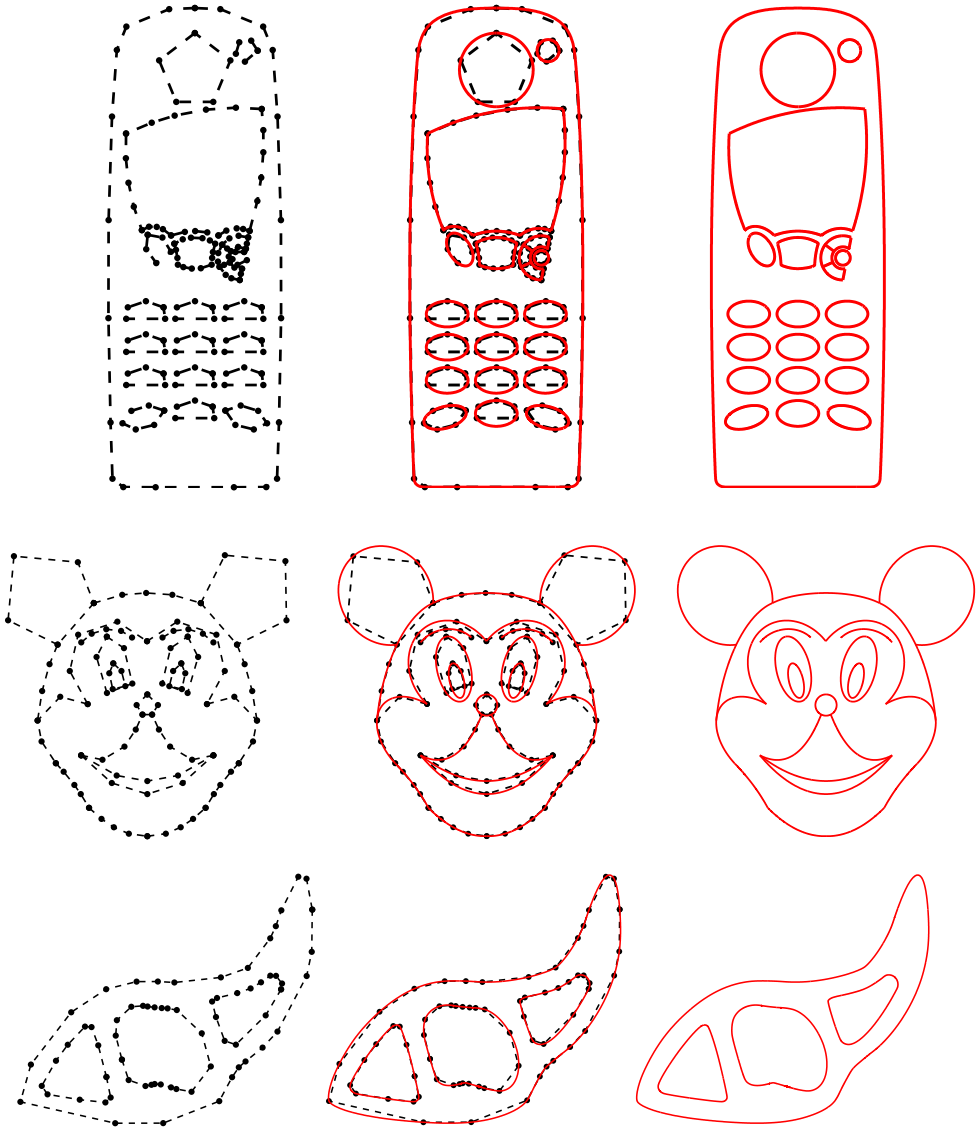


Figure 24: Application examples of the subdivision algorithm of Subsection 4.2 to real world data. Left: starting polylines. Center and Right: refined polylines obtained after 7 steps of our algorithm. Data of first and third row are courtesy of think3.


## Acknowledgements

L. Romani thanks the University of Valenciennes for financing a one month visit during the academic year 2010–2011.


## References


[1] N. Dyn, D. Levin, Subdivision schemes in geometric modelling, Acta Numer. 11 (2002) 73-144.

[2] D. Zorin, P. Schröder, Subdivision for modeling and animation, in SIGGRAPH Course Notes, 2000.

[3] M.A. Sabin, Recent progress in subdivision: a survey, in: N.A. Dodgson, M.S. Floater, M.A. Sabin (Eds.), Advances in Multiresolution for Geometric Modelling, Springer-Verlag, 2005, pp. 203-230.

[4] C.V. Beccari, G. Casciola, L. Romani, Non-uniform interpolatory curve subdivision with edge parameters built upon compactly supported fundamental splines, BIT Numer. Math. 51(4) (2011) 781-808.

[5] C. Beccari, G. Casciola, L. Romani, Polynomial-based non-uniform interpolatory subdivision with features control, J. Comput. Appl. Math. 235(16) (2011) 4754-4769.

[6] F. Kuijt, Convexity preserving interpolation - stationary nonlinear subdivision and splines, PhD Thesis, University of Twente, 1998.
[7] C. Deng, G. Wang, Incenter subdivision scheme for curve interpolation, Comput. Aided Geom. Design 27(1) (2010) 48-59.
[8] N. Dyn, D. Levin, D. Liu, Interpolatory convexity-preserving subdivision schemes for curves and surfaces, Comput. Aided Design 24(4) (1992) 211-216.
[9] N. Dyn, Three families of nonlinear subdivision schemes, in: K. Jetter, M. Buhmann, W. Haussman, R. Schaback, J. Stoeckler (Eds.), Topics in Multivariate Approximation and Interpolation, Elsevier, 2005, pp. 23-38.
[10] F. Kuijt, R. Van Damme, Shape-preserving interpolatory subdivision schemes for nonuniform data, J. Approx. Theory 114 (2002) 1-32.
[11] M. Marinov, N. Dyn, D. Levin, Geometrically controlled 4-point interpolatory schemes, in: N.A. Dodgson, M.S. Floater, M.A. Sabin (Eds.), Advances in Multiresolution for Geometric Modelling, Springer-Verlag, 2005, pp. 301-315.
[12] M. Floater, C. Beccari, T. Cashman and L. Romani, A smoothness criterion for monotonicity-preserving subdivision, Adv. Comput. Math. doi:10.1007/s10444-012-9275-y (in press).
[13] G. Deslauriers, S. Dubuc, Symmetric iterative interpolation processes, Constr. Approx. 5 (1989) 49-68.
[14] A. Cohen, N. Dyn, B. Matei, Quasilinear subdivision schemes with applications to ENO interpolation, Appl. Comput. Harmon. Anal. 15 (2003) 89-116.
[15] N. Dyn, M. Floater, K. Hormann, Four-point curve subdivision based on iterated chordal and centripetal parameterizations, Comput. Aided Geom. Design 26(3) (2009) 279-286.
[16] M. Floater, C. Micchelli, Nonlinear stationary subdivision, in: N. Govil, R. Mohapatra, Z. Nashed, A. Sharma, J. Szabados (Eds.), Approximation Theory: in memory of A.K. Varma, Elsevier, 1998, pp. 209-224.
[17] M.A. Sabin, N. Dodgson, A circle-preserving variant of the four-point subdivision scheme, in: M. Dæhlen, K. Mørken, L.L. Schumaker (Eds.), Mathematical Methods for Curves and Surfaces: Tromsø 2004, Nashboro Press, 2005, pp. 275-286.
[18] N. Aspert, T. Ebrahimi, P. Vandergheynst, Non-linear subdivision using local spherical coordinates, Comput. Aided Geom. Design 20(3) (2003) 165-187.
[19] L. Romani, A circle-preserving $C^2$ Hermite interpolatory subdivision scheme with tension control, Comput. Aided Geom. Design 27(1) (2010) 36-47.
[20] C. Beccari, G. Casciola, L. Romani, A non-stationary uniform tension controlled interpolating 4-point scheme reproducing conics, Comput. Aided Geom. Design 24(1) (2007) 1-9.
[21] C. Beccari, G. Casciola, L. Romani, Shape-controlled interpolatory ternary subdivision, Appl. Math. Comput. 215(3) (2009) 916-927.
[22] C. Conti, L. Gemignani, L. Romani, From approximating to interpolatory non-stationary subdivision schemes with the same generation properties, Adv. Comput. Math. 35(2) (2011) 217-241.
[23] C. Conti, L. Romani, Affine combination of B-spline subdivision masks and its non-stationary counterparts, BIT Numer. Math. 50(2) (2010) 269-299.
[24] C. Conti, L. Romani, Algebraic conditions on non-stationary subdivision symbols for exponential polynomial reproduction, J. Comput. Appl. Math. 236(4) (2011) 543-556.
[25] L. Romani, From approximating subdivision schemes for exponential splines to high-performance interpolating algorithms, J. Comput. Appl. Math. 224(1) (2009) 383-396.
[26] G. Albrecht, J.-P. Bécar, G. Farin, D. Hansford, On the approximation order of tangent estimators, Comput. Aided Geom. Design 25 (2008) 80-95.
[27] G. Farin, NURBS - From projective geometry to practical use, second edition, AK Peters, Natick, Massachusetts, 1999.
[28] G. Albrecht, Géométrie projective. In Encyclopédie Techniques de l'Ingénieur, AF 206, Editions T.I., 2008, http://www.techniques-ingenieur.fr/book/af206/geometrie-projective.html.
[29] M. Chasles, Traité des sections coniques. Paris, Gauthier Villars, 1865.
[30] E. Pascal, Repertorium der Höheren Mathematik. Teubner, B.G., Leipzig/Berlin, 1910.
[31] V. Hernández, J. Estrada, I. Ivrissimtzis, S. Morales, Curve subdivision with arc-length control, Computing 86(2-3) (2009) 151-169.