Computers & Graphics 36 (2012) 642–650

Contents lists available at SciVerse ScienceDirect

Computers & Graphics

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

Technical Section

Euler arc splines for curve completion

Hailing Zhou [a], Jianmin Zheng [a,*], Xunnian Yang [b]

[a] School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore
[b] Mathematics Department, Zhejiang University, Hangzhou, China

ARTICLE INFO

Article history:
Received 21 October 2011
Received in revised form
2 April 2012
Accepted 2 April 2012
Available online 14 April 2012

Keywords:
Euler curves
Arc spline
Aesthetical curves
Shape completion

ABSTRACT

This paper introduces a special arc spline called an *Euler arc spline* as the basic form for visually pleasing completion curves. It is considered as an extension of an Euler curve in the sense that the points in the Euler curve are replaced by arcs. A simple way for specifying it, which is suitable for shape completion, is presented. It is shown that Euler arc splines have several properties desired by aesthetics of curves, in addition to computational simplicity and NURBS representation. An algorithm is proposed for curve completion using Euler arc splines. The development of the algorithm involves two optimization processes, which are converted into a single minimization problem in two variables solved by the Levenberg–Marquardt algorithm. Compared to previous methods, the proposed algorithm always guarantees the interpolation of two boundary conditions.

1. Introduction

This paper deals with the problem of *curve completion*, which is a process of completing contours beyond occlusions or across gaps [1]. It is also known as *shape completion* or *gap completion*, which is related to visual completion—a fundamental skill for human vision system [2]. Referring to Fig. 1, for example, when an object is partially occluded by others, human can automatically fill the gap by completing its contour; when an object is illusory, human can also consciously generate a subjective completed boundary. However, for a computer, this is a nontrivial task. Shape completion has wide applications in computer graphics such as shape transition or repairing for CAD [3,4] and in computer vision such as inpainting for objects with smooth curvilinear shapes [1].

The process of shape completion is actually to find an optimal curve through two specified endpoints with associated orientations, which we call *point-orientation pairs*. There exist many possible curves that meet the conditions of point-orientation pairs. The problem is under-specified despite the appeal of our vision intuition for an optimal solution [1]. The solution really depends on the criteria regarding what constitutes the most "likely" or the most "pleasing" curve [1,5]. In this paper we provide a new solution by presenting an appropriate curve representation and its associated construction algorithm.

1.1. Related work

Since two given points with associated orientations provide the first order geometric Hermite data, a straightforward approach to shape completion is to use cubic Hermite interpolation [4]. Hermite interpolation is simple to construct and compute, but it does not always provide satisfactory results as shown in [3]. It is pointed out in [6] that one reason for Hermite interpolation to produce undesired shapes is unsuitable magnitudes of the given tangent vectors. Therefore Yong and Cheng present a new class of curves called optimized geometric Hermite curves for which the magnitudes of the endpoint tangent vectors in the Hermite interpolation process are optimized to make the strain energy of the curves be minimized [6].

Ullman suggests several criteria for completion curves [7]. That is, the curves should be invariant to rigid transformation, at least differentiable once, extensible, and minimize total curvature. Based on these criteria, he then proposes to use biarcs that minimize total square curvature as completion curves. However, it is later found that in many cases biarc completions have less pleasing appearance than the cubic polynomial completions [8] and Ullman's biarc completion curve is generally not extensible [9]. Knuth considers the shape representation and construction of letters or symbols in typography from a collection of points [10], which is a problem similar to shape completion. He proposes six criteria, somewhat similar to Ullman's, which the most "pleasing" curve through a set of specified points should satisfy. These criteria are known as similar transformation invariance, symmetry, extensibility, locality, smoothness, and roundedness [10,1]. Since the last four criteria cannot be simultaneously satisfied, Knuth gives up the extensibility and roundedness properties but insists in the locality property,

* Corresponding author. Tel.: +65 67906257; fax: +65 67926559.
E-mail address: asjmzheng@ntu.edu.sg (J. Zheng).

0097-8493/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cag.2012.04.001

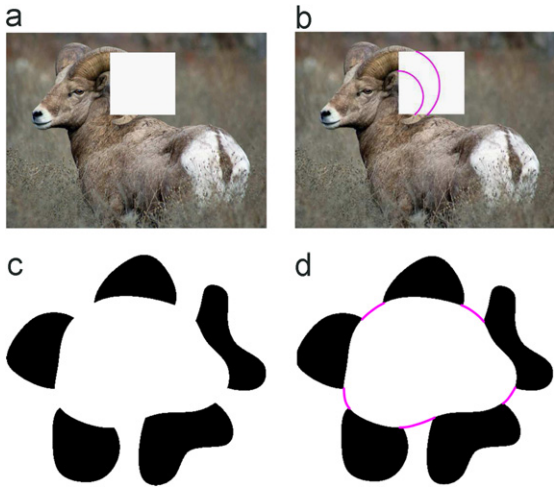

Fig. 1. Visual completion examples. (a) Object shape is partially missed; (c) illusory contours; (b), (d) the results of curve completion using our method.

which leads to a cubic spline interpolation solution. This is the base of Knuth's METAFONT system. However, psychological studies have indicated that splines may not be a satisfactory primitive for shape completion [11].

A lot of work on aesthetic curve design proposes some energy functional and defines the curve as a solution to the problem of minimizing the energy functional. Elastica is one of such examples, in which the energy functional is the integral of a linear combination of the square curvature and the arc-length [12–15]. Elastica is extensible, but not scale-invariant or rounded. On the other hand, it is argued that the energy functional capturing the elusive nature of the "most pleasing" curve should penalize curvature variation, but not necessarily curvature proper as in Elastica. In particular, minimizing the integral of the square of the derivative of curvature with respect to arc-length requires the curvature to be linear in arc-length, which leads to an Euler curve. Euler curves are also known as "Cornu spirals" or "Clothoid". They have been used in various applications including highway and railroad design, computer aided design, and computer arts. 2D Euler curves have been generalized to 3D such that the curvature and torsion of the curve are linear with arc-length [16,3] or to piecewise Euler spirals [17]. Since Euler curves are defined by complicated transcendental functions, some work devotes to approximating Euler curves by some simple representations such as Bézier curves and arc splines [18–22].

Kimia et al. [1] show that Euler curves are a suitable primitive for shape completion and various practical advantages of using Euler curve interpolation are examined. An algorithm for shape completion using Euler curves is described, which finds the completion curve by solving two nonlinear equations in two unknowns involving Fresnel integrals. A biarc fitting is used for producing an initial guess and then the algorithm performs iteratively. Walton and Meek [23] improve the work by formulating the problem into two nonlinear equations with only one unknown each, which gives a faster and more accurate algorithm.

1.2. Our work

This paper is inspired by Kimia et al.'s work of using Euler curves for shape completion [1]. Psychological studies show that Euler curves have good fit to the way that the human eyes complete curves [11]. However, there are some drawbacks with Euler curves for shape completion. First, due to complicated transcendental function representation of Euler curves, it is difficult or expensive to compute with Euler curves. In particular, for rendering purpose, Euler curves are often approximated by line segments or arc segments. Second, Kimia et al.'s algorithm finds the solution by a numerical approach that minimizes the distance between the second given point and the last point of the Euler curve. Sometimes the numerical approach might not converge or the approximate solution is not good enough. As a consequence, the resulting Euler curve will not pass through the second point. This will violate the aesthetics of shape completion since human perception is sensitive to gaps. Third, Euler curves are not compatible with nonuniform rational B-splines (NURBS) which are an industry standard in CAD/CAM.

These limitations motivate us to explore new completion curve models and shape completion algorithms. It is observed by Horn [12] that in the optimal multi-arcs that approximate the "smoothest" curve arcs tend to be of equal length and curvature changes more or less linearly along the curve. Therefore in this paper, we propose a special arc spline called an *Euler arc spline* as the completion curve primitive for shape completion. The Euler arc spline consists of G^1 continuous arcs of the same length with curvature changing linearly from one arc to another. It is a good approximation to an Euler curve. We identify and analyze the parameters needed to specify an Euler arc spline in Section 2 and show that Euler arc spline curves exhibit similar properties as Euler curves. Meanwhile, Euler arc spline curves are simple for computation, rendering and other geometric processing such as offsetting and distance query. They are nonuniform rational B-spline curves, facilitating the use with standard graphics packages. Another arc spline similar to an Euler arc spline has been introduced in [21], which is called discrete clothoid where the first and last arcs are half the length of the others. The approximation properties of discrete clothoid are analyzed. Particularly, if a clothoid is approximated by a discrete clothoid of n arcs, the approximation error is of order $O(1/n^2)$. However, no algorithm is provided to construct a discrete clothoid for shape completion.

We also propose a curve completion algorithm using the proposed Euler arc splines in Section 3. We reduce the construction of completion curve from two point-orientation pairs to minimizing the sum of squares of nonlinear functions in two unknowns, which is then solved by the Levenberg–Marquardt algorithm. The underlying idea of our approach is to distribute the approximate error between the second given point and the last point of the completion curve to all the arcs and an optimization procedure is performed to minimize the errors. Consequently, the interpolation of two given point-orientation pairs with our completion curve is always guaranteed, which overcomes the drawback of Kimia et al.'s algorithm.

2. Arc splines and Euler arc splines

This section first examines the representation of a G^1 continuous arc spline, then introduces Euler arc splines, and identifies parameters needed for the construction of an Euler arc spline in a way that is suitable for shape completion. Some properties of Euler arc splines are also analyzed.

2.1. G^1 continuous arc splines

A directed arc can be uniquely defined by its starting point, orientation angle that is a rotation angle from the positive x-axis to the tangent direction of the arc at the starting point, an arc-length and a central angle. If the direction of the arc is clockwise, the central angle is negative; otherwise, the angle is positive. Suppose there are n such directed arcs. They are connected one by

one at the endpoints with tangent continuity as shown in Fig. 2. Then they form a G^1 continuous arc spline.

Proposition 1. *A G^1 continuous arc spline consists of n arcs $\mathcal{A}_i, i=1,2,\ldots,n$. If each $\mathcal{A}_i$ has an arc-length of s_i and a central angle of $\Delta\theta_i$, then the i-th arc $\mathcal{A}_i$ $(i=1,2,\ldots,n)$ has the following ending point and orientation angle θ_i at the ending point:*

$$\theta_i = \theta_0 + \sum_{j=1}^{i} \Delta\theta_j \tag{1}$$

$$P_i = P_0 + \sum_{j=1}^{i} s_j \frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j} T(\theta_0+\phi_j) \tag{2}$$

where P_0,θ_0 are the starting point and the orientation angle at P_0 of the first arc $\mathcal{A}_1$, $\phi_j=(\theta_{j-1}+\theta_j)/2-\theta_0$, and $T(\theta)$ denotes unit vector $\left[\begin{smallmatrix}\cos\theta\\ \sin\theta\end{smallmatrix}\right]$. That is, a G^1 continuous arc spline with n arcs can be specified by $P_0,\theta_0,s_1,\Delta\theta_1,\ldots,s_n,\Delta\theta_n$.

Proof. Since $\Delta\theta_j$ is the central angle, $\theta_j-\theta_{j-1}=\Delta\theta_j$. Thus $\theta_i=\theta_{i-1}+\Delta\theta_i=\theta_{i-2}+\Delta\theta_{i-1}+\Delta\theta_i=\cdots=\theta_0+\sum_{j=1}^{i}\Delta\theta_j$.

Consider vector $P_{i-1}P_i$ (see Fig. 3). Its length is the chord-length of the arc $\mathcal{A}_i$. The radius of $\mathcal{A}_i$ is $s_i/\Delta\theta_i$. Therefore the chord-length is $2(s_i/\Delta\theta_i)\sin\Delta\theta_i/2$. The direction of $P_{i-1}P_i$ can be obtained by rotating $T(\theta_{i-1})$ by an angle of $\Delta\theta_i/2$. $T(\theta_{i-1})$ can be obtained by rotating $T(\theta_{i-2})$ by an angle of $\Delta\theta_{i-1}$. We continue this and can conclude that the direction of $P_{i-1}P_i$ is obtained by rotating $T(\theta_0)$ by an angle:

$$\frac{\Delta\theta_i}{2}+\Delta\theta_{i-1}+\cdots+\Delta\theta_1=\frac{\theta_{i-1}+\theta_i}{2}-\theta_0=\phi_i.$$

Thus $P_{i-1}P_i=2s_i/\Delta\theta_i\,\sin\Delta\theta_i/2M(\phi_i)T(\theta_0)$ where $M(\phi_i)=\left[\begin{smallmatrix}\cos\phi_i & -\sin\phi_i\\ \sin\phi_i & \cos\phi_i\end{smallmatrix}\right]$ is a rotation matrix. Then we have

$$P_i=P_{i-1}+P_{i-1}P_i=P_{i-2}+P_{i-2}P_{i-1}+P_{i-1}P_i=\cdots=P_0+\sum_{j=1}^{i}P_{j-1}P_j$$

$$=P_0+\sum_{j=1}^{i}s_j\frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j}M(\phi_j)T(\theta_0)=P_0+\sum_{j=1}^{i}s_j\frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j}T(\theta_0+\phi_j).\quad\square$$

It is easy to see that for $i=1,\ldots,n-1$, P_i and θ_i are also the starting point and its associated orientation angle of the $(i+1)$-th arc. While P_i and θ_i depend on the coordinate system, central angles $\Delta\theta_i$ and arc-lengthes s_i are coordinate system independent.

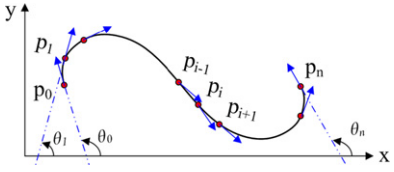

Fig. 2. A G^1 continuous arc spline curve.

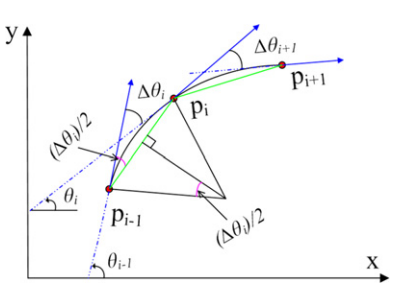

Fig. 3. One arc segment in an arc spline curve.

An arc spline can be represented as a NURBS curve. Now we show how to convert a G^1 continuous arc spline into a quadratic NURBS curve. Consider arc $\mathcal{A}_i$. Once we have P_{i-1}, P_i and $\Delta\theta_i$, we can represent $\mathcal{A}_i$ as a quadratic rational Bézier curve. To ensure the positivity of weights, we require $|\Delta\theta_i|<\pi$.

In case $|\Delta\theta_i|\geq\pi$, we split the arc into two sub-arcs by inserting a new point $P_{i-1/2}$ such that the two sub-arcs have the same central angle $\Delta\theta_i/2$. By some calculations, the new point $P_{i-1/2}$ has an expression:

$$P_{i-1/2}=P_{i-1}+s_i\frac{2\sin\frac{\Delta\theta_i}{4}}{\Delta\theta_i}T\left(\frac{\Delta\theta_i}{4}+\theta_{i-1}\right).$$

If $|\Delta\theta_i|<\pi$, then the arc can be defined by a rational Bézier curve whose three control points and weights are P_{i-1},$P_{i-1}+s_i/\Delta\theta_i\tan\Delta\theta_i/2T(\theta_{i-1})$,$P_i$ and $1,\cos\Delta\theta_i/2,1$. Refer to Fig. 4 for an illustration.

Summarizing the above discussion, we can design an algorithm that converts the arc spline into a quadratic NURBS curve:

Input: G^1 continuous arc-spline
Output: a quadratic NURBS curve
Step 1. Find P_{i-1},P_i and $\Delta\theta_i$ for each arc using Proposition 1.
Step 2. Check each arc. If $|\Delta\theta_i|\geq\pi$, split the arc into two sub-arcs.
Step 3. Re-organize all the arcs. Suppose now we have m arcs and they are defined by points $Q_0,Q_1,Q_2,\ldots,Q_m$, central angles $\Delta\theta_1,\Delta\theta_2,\ldots,\Delta\theta_m$, and arc-lengths $s_1,s_2,\ldots,s_m$. The central angles are all less than π.
Step 4. Construct the NURBS curve with control points
$Q_0,Q_{1/2},Q_1,Q_{3/2},Q_2,\ldots,Q_m$
where $Q_{i-1/2}=Q_{i-1}+\frac{s_i}{\Delta\theta_i}\tan\frac{\Delta\theta_i}{2}T(\theta_{i-1})$, weights
$1,\cos\frac{\Delta\theta_1}{2},1,\cos\frac{\Delta\theta_2}{2},\ldots,\cos\frac{\Delta\theta_m}{2},1$,
and knot sequence
$\{0,0,0,1,1,2,\ldots,m-2,m-1,m-1,m,m,m\}$.

2.2. Euler arc splines

Definition 1. An Euler arc spline (EAS) is defined to be a curve consisting of several arcs satisfying the three conditions: (1) the arcs have the same arc-length; (2) the arc are joined to form a G^1 continuous curve; and (3) the curvatures of the arcs vary linearly from one arc to another.

An Euler arc spline is a special arc spline that has constraints on arc-lengths and curvatures. It is named so because it can be considered as an extension of an Euler curve in the sense that each point on the Euler curve is replaced by an arc.

Consider an Euler arc spline with n arcs $\mathcal{A}_i, i=1,2,\ldots,n$. Assume $\mathcal{A}_i$ starts at P_{i-1} with orientation angle θ_{i-1}, ends at P_i with orientation angle θ_i, and has curvature κ_i and arc-length s.

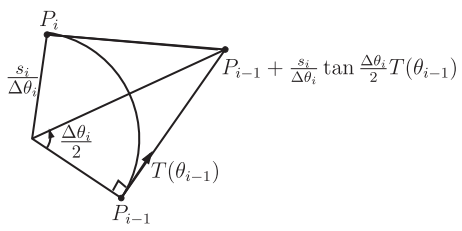

Fig. 4. One arc segment and its three Bézier control points.

Based on the definition, the curvature can be defined by $\kappa_i=\alpha s(i-1)+\kappa_1$ for $i\geq 1$ where κ_1 and α are two constants, called the initial curvature and curvature slope, respectively. From κ_i and s, we can derive

$$\Delta\theta_i=\kappa_i s=\alpha s^2(i-1)+\kappa_1 s.$$

From Proposition 1, we can derive P_i and θ_i. That is,

Proposition 2. *An Euler arc spline can be defined by six parameters*: $P_0=(x_0,y_0),\theta_0,n,s,\alpha$ *and* κ_1, *from which we have*

$$\theta_i=\theta_0+i\kappa_1 s+\frac{(i-1)i}{2}s^2\alpha$$

$$P_i=P_0+\sum_{j=1}^{i}\frac{2\sin\frac{\kappa_j s}{2}}{\kappa_j}T(\phi_j+\theta_0)$$

where $\phi_j=\frac{\theta_{j-1}+\theta_j}{2}-\theta_0=\frac{s^2(j-1)^2\alpha+(2j-1)\kappa_1 s}{2}$.

Moreover, there are constraints on s,α,κ_1,n to ensure that each arc is not longer than a full circle, which gives $-2\pi\leq\Delta\theta_i\leq 2\pi$. Substituting κ_i into this constraint, we obtain

$$-2\pi\leq s^2\alpha(i-1)+\kappa_1 s\leq 2\pi,\quad i=1,2,\ldots,n. \tag{3}$$

Proposition 3. *For an Euler arc spline with each arc segment not longer than a full circle, its initial curvature* κ_1 *and curvature slope* α *satisfy*:

$$-\frac{2\pi}{s}\leq\kappa_1\leq\frac{2\pi}{s}, \tag{4}$$

$$-\frac{4\pi}{(n-1)s^2}\leq-\frac{2\pi+\kappa_1 s}{(n-1)s^2}\leq\alpha\leq\frac{2\pi-\kappa_1 s}{(n-1)s^2}\leq\frac{4\pi}{(n-1)s^2}. \tag{5}$$

Proof. The proof is straightforward just by letting $i=1$ and $i=n$ in (3), respectively. □

Proposition 3 shows that when n increases, the range of α decreases if s remains unchanged. If α remains unchanged, then when n increases, s must decrease. In particular, when n goes to infinity, s tends to zero, which implies that an Euler arc spline converges to an Euler curve when the number of arcs goes to infinity.

Furthermore, we have

Proposition 4. *Given constants* κ_1, α, $s(>0)$, *and a natural number* n *satisfying* $|\kappa_1 s|\leq 2\pi$ *and* $|s^2\alpha(n-1)+\kappa_1 s|\leq 2\pi$, *there exists an Euler arc spline with* n *arc segments having a total arc-length of* n s *and curvature* $\kappa_i=\alpha s(i-1)+\kappa_1$ *varying linearly in arc index* i. *The Euler arc spline is unique up to a rigid transformation.*

Proof. When $\alpha\geq 0$, for all $i=1,2,\ldots,n$, we have

$$-2\pi\leq\kappa_1 s\leq\alpha(i-1)s^2+\kappa_1 s\leq\alpha(n-1)s^2+\kappa_1 s\leq 2\pi.$$

Similarly, when $\alpha<0$, we have

$$-2\pi\leq\alpha(n-1)s^2+\kappa_1 s\leq\alpha(i-1)s^2+\kappa_1 s\leq\kappa_1 s\leq 2\pi.$$

We let the central angles $\Delta\theta_i=\kappa_i s$. Then $|\Delta\theta_i|\leq 2\pi$. If we arbitrarily choose a starting point P_0 and an orientation θ_0, Proposition 4 gives the expression of an Euler arc spline satisfying the requirements. It also shows that such an Euler arc spline is unique up to a translation dependent of P_0 and a rotation dependent of θ_0. □

The shape of an Euler arc spline curve can be classified into four types depending on the signs of α and κ_1, which are illustrated in Fig. 5.

In addition to having curvature change linearly from one arc to another, Euler arc splines also have other nice properties as listed below. These properties include or are similar to similarity transformation invariance, symmetry, extensibility, smoothness, and roundedness, which are required by the aesthetics of curves.

- An Euler arc spline is invariant to translation, rotation, and scaling.
  In fact, suppose an Euler arc spline curve $\mathcal{A}$ is defined by P_0,θ_0,n,s,α and κ_1. A translation applied to $\mathcal{A}$ is achieved by just applying the translation to P_0. If we rotate $\mathcal{A}$ around the origin by an angle ϕ, this is achieved by applying the rotation to P_0 and meanwhile updating θ_0 to $\theta_0+\phi$. If $\mathcal{A}$ is multiplied by c, the result is a new Euler arc spline defined by $cP_0,\theta_0,n,cs,\alpha/c^2$ and κ_1/c.
- The Euler arc spline defined by parameters P_0,θ_0,n,s,α and κ_1 coincides with the Euler arc spline defined by $P_n,\theta_n+\pi,n,s,\alpha$ and $-\kappa_n$, where $\kappa_n=\alpha s(n-1)+\kappa_1$, $\theta_n=\theta_0+n\kappa_1 s+(n(n-1)/2)s^2\alpha$ and $P_n=P_0+\sum_{j=1}^{n}(2\sin(\kappa_j s/2)/\kappa_j)T((s^2(j-1)^2\alpha+(2j-1)\kappa_1 s)/2+\theta_0)$.
  To prove this, we denote the first curve by $r(t),t\in[0,ns]$ and the second one by $\overline{r}(t),t\in[0,ns]$ and show that $r(hs+l)=\overline{r}((n-h-1)s+s-l)$ for $h=0,1,\ldots,n-1$ and $l\in[0,s]$. In fact,

$$r(hs+l)=P_0+\sum_{j=1}^{h}\frac{2\sin\frac{\kappa_j s}{2}}{\kappa_j}T(\phi_j+\theta_0)+\frac{2\sin\frac{\kappa_{h+1}l}{2}}{\kappa_{h+1}}T\left(\theta_h+\frac{\kappa_{h+1}l}{2}\right)$$

and

$$\overline{r}((n-h-1)s+s-l)=P_n+\sum_{j=1}^{n-h-1}\frac{2\sin\frac{\overline{\kappa}_j s}{2}}{\overline{\kappa}_j}T(\overline{\phi}_j+\theta_n+\pi)+\frac{2\sin\frac{\overline{\kappa}_{n-h}(s-l)}{2}}{\overline{\kappa}_{n-h}}T\left(\overline{\theta}_{n-h-1}+\frac{\overline{\kappa}_{n-h}(s-l)}{2}\right)$$

where $\overline{\kappa}_j=\alpha s(j-1)-\kappa_n=-\alpha s(n-j)-\kappa_1=-\kappa_{n-j+1}$, $\overline{\theta}_j=\pi+\theta_{n-j}$, and $\overline{\phi}_j=\alpha s^2(j-1)^2-(2j-1)k_n s/2$. Thus

$$\overline{r}((n-h-1)s+s-l)=P_0+\sum_{j=1}^{n}\frac{2\sin\frac{\kappa_j s}{2}}{\kappa_j}T(\phi_j+\theta_0)$$

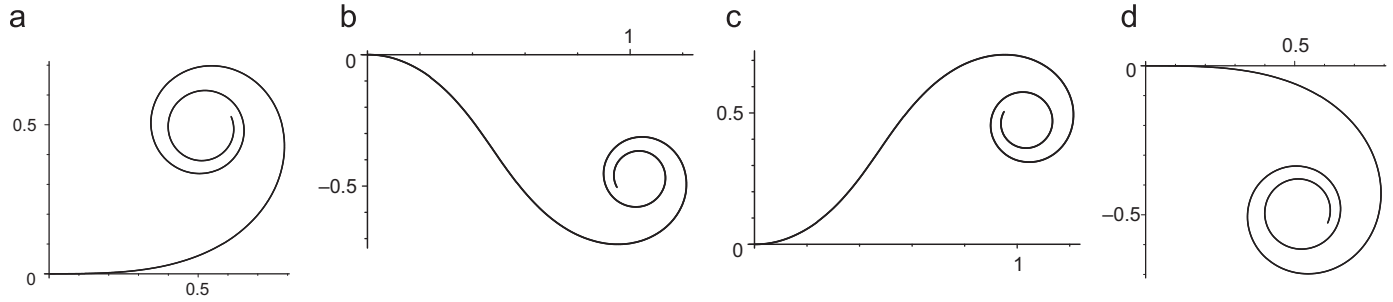

Fig. 5. Four types of shape of an Euler arc spline: (a) $\alpha>0,\kappa_1>0$, (b) $\alpha>0,\kappa_1<0$, (c) $\alpha<0,\kappa_1>0$, (d) $\alpha<0,\kappa_1<0$.

$$+\sum_{j=1}^{n-h-1}\frac{2\sin\frac{\kappa_{n+1-j}s}{2}}{\kappa_{n+1-j}}T(\phi_{n+1-j}+\theta_0+\pi)$$
$$+\frac{2\sin\frac{\kappa_{h+1}(s-l)}{2}}{\kappa_{h+1}}T\left(\pi+\theta_{h+1}-\frac{\kappa_{h+1}(s-l)}{2}\right)$$
$$=P_0+\sum_{j=1}^{h}\frac{2\sin\frac{\kappa_j s}{2}}{\kappa_j}T(\phi_j+\theta_0)+\frac{2\sin\frac{\kappa_{h+1}l}{2}}{\kappa_{h+1}}T\left(\theta_h+\frac{\kappa_{h+1}l}{2}\right)$$
$$=r(hs+l).$$

- Two Euler arc curves defined by $P_0,\theta_0,m,s,\alpha,\kappa_1$ and $P_m,\theta_m,n-m,s,\alpha,\alpha sm+\kappa_1$ coincide with the Euler arc curve defined by $P_0,\theta_0,n,\ s,\ \alpha,\kappa_1$, each in its own section.
- An Euler arc spline curve is at least G^1 continuous.
- If the two point-orientation pairs lie on a circle, then there exists an EAS interpolating the point-orientation pairs, which coincides with the circle. This is because a circle is a special case of an Euler arc spline with $\alpha=0$.

3. Curve completion algorithm

This section presents an algorithm that attempts to construct an EAS curve to interpolate two given point-orientation pairs for shape completion. The problem of curve completion using an EAS can be stated as follows: Given two points $P_A=(x_A,y_A)$ and $P_B=(x_B,y_B)$ with orientations θ_A and θ_B, find an EAS curve consisting of n arcs that interpolates them, which is illustrated in Fig. 6.

As discussed in Section 2.2, an Euler arc curve can be defined by $P_0,\theta_0,n,\ s,\ \alpha$ and κ_1. For our problem, we obtain the first three parameters immediately from the interpolation condition at one given point: $P_0=P_A,\theta_0=\theta_A$. When n is specified, only s,α and κ_1 are to be determined. By Proposition 4, $\theta_n=\theta_0+n\kappa_1 s+((n-1)n)/2s^2\alpha$. Letting $\theta_n=\theta_B$ gives

$$\alpha=\frac{2(\theta_B-\theta_A-n\kappa_1 s)}{(n-1)ns^2} \tag{6}$$

We further need to let the last point of the Euler arc spline curve interpolate the second point P_B. Thus

$$P_B=P_A+\sum_{j=1}^{n}\frac{2\sin\frac{\kappa_j s}{2}}{\kappa_j}T(\phi_j+\theta_A) \tag{7}$$

which gives two equations for determining s and κ_1. Unfortunately, these equations are highly nonlinear, which makes them difficult to be solved. Kimia et al. [1] propose an numerical method by sampling s and κ_1 values to find the solution. Since usually an approximate solution is obtained, it sometimes happens that the approximate curve does not touch P_B. However, human perception is sensitive to the gap.

In this section, we present a new approach to find s and κ_1. Our basic idea is that we perturb the arc-length of each arc by an δ_i to ensure the interpolation, which is somewhat equivalent to distributing the deficiency between the last point of the Euler arc spline and P_B to all the arcs, and we want to find the best s and κ_1

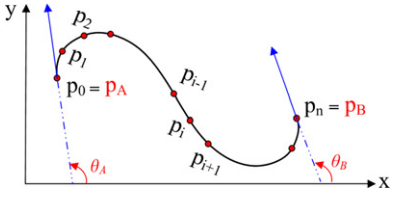

Fig. 6. Shape completion with an EAS curve.

so that the perturbation amount is minimized. One advantage of our approach is that the constructed curve always interpolates the point-orientation pairs. Thus instead of solving (7), we search for a solution to an optimization problem:

$$(s,\kappa_1)=\arg\min_{s,\kappa_1}\left(\min_{\substack{\delta_i\\ \text{s.t. } P_n^\delta=P_B}}\sum_{i=1}^{n}\delta_i^2\right) \tag{8}$$

where P_n^δ is the last point of the perturbed curve. The next two sub-sections explain how to solve this optimization problem.

3.1. Perturbation for interpolation

Suppose we have already had an EAS that may not interpolate P_B. Now we fix the central angles, and change the arc-length of each arc to $s_i=s+\delta_i$ with a perturbation δ_i. There are many choices for δ_i to achieve interpolation. We determine δ_i by solving the following minimization problem:

$$\begin{aligned}&\min\quad \sum_{i=1}^{n}\delta_i^2\\ &\text{s.t.}\quad P_n^\delta=P_B\end{aligned} \tag{9}$$

This is a constrained optimization problem. Based on Proposition 1,

$$P_n^\delta=P_A+\sum_{j=1}^{n}(s+\delta_j)\frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j}T(\theta_A+\phi_j).$$

Let

$$V_j=\frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j}T(\theta_A+\phi_j)=\frac{2\sin\frac{\Delta\theta_j}{2}}{\Delta\theta_j}\begin{bmatrix}\cos(\theta_A+\phi_j)\\ \sin(\theta_A+\phi_j)\end{bmatrix}.$$

By introducing Lagrangian multipliers λ_1 and λ_2, we convert the minimization problem (9) into an unconstrained one:

$$\min G(\delta_1,\ldots,\delta_n,\lambda_1,\lambda_2)=\min\left\{\sum_{i=1}^{n}\delta_i^2+\lambda_1\left(P_A^x-P_B^x+\sum_{j=1}^{n}sV_j^x+\sum_{j=1}^{n}\delta_j V_j^x\right)\right.$$
$$\left.+\lambda_2\left(P_A^y-P_B^y+\sum_{j=1}^{n}sV_j^y+\sum_{j=1}^{n}\delta_j V_j^y\right)\right\}$$

where the superscripts "x" and "y" stand for the x- and y-components of vectors.

To solve the optimization problem, we take the partial derivatives of $G(\delta_1,\ldots,\delta_n,\lambda_1,\lambda_2)$ with respect to δ_k, and λ_1,λ_2, respectively, and set the results to zero. This results in the following equations:

$$\frac{\partial G}{\partial\delta_k}=2\delta_k+\lambda_1V_k^x+\lambda_2V_k^y=0,\quad k=1,\ldots,n \tag{10}$$

and

$$\frac{\partial G}{\partial\lambda_1}=P_A^x-P_B^x+\sum_{j=1}^{n}sV_j^x+\sum_{j=1}^{n}\delta_jV_j^x=0$$

$$\frac{\partial G}{\partial\lambda_2}=P_A^y-P_B^y+\sum_{j=1}^{n}sV_j^y+\sum_{j=1}^{n}\delta_jV_j^y=0 \tag{11}$$

From Eq. (10), we obtain

$$\delta_k=-\frac{\lambda_1V_k^x+\lambda_2V_k^y}{2}. \tag{12}$$

Substituting them into Eq. (11) arrives at two linear equations in λ_1 and λ_2:

$$\sum_{j=1}^{n}(V_j^x)^2\lambda_1+\sum_{j=1}^{n}(V_j^xV_j^y)\lambda_2=2\left(P_A^x-P_B^x+s\sum_{j=1}^{n}V_j^x\right)$$

$$\sum_{j=1}^{n}(V_j^xV_j^y)\lambda_1+\sum_{j=1}^{n}(V_j^y)^2\lambda_2=2\left(P_A^y-P_B^y+s\sum_{j=1}^{n}V_j^y\right)$$

The solutions of the two linear equations are

$$\lambda_1=\frac{2\sum_{j=1}^{n}(V_j^y)^2\left(P_A^x-P_B^x+s\sum_{j=1}^{n}V_j^x\right)-2\sum_{j=1}^{n}(V_j^xV_j^y)\left(P_A^y-P_B^y+s\sum_{j=1}^{n}V_j^y\right)}{\sum_{j=1}^{n}(V_j^x)^2\sum_{j=1}^{n}(V_j^y)^2-\left(\sum_{j=1}^{n}V_j^xV_j^y\right)^2}$$

$$\lambda_2=\frac{2\sum_{j=1}^{n}(V_j^x)^2\left(P_A^y-P_B^y+s\sum_{j=1}^{n}V_j^y\right)-2\sum_{j=1}^{n}(V_j^xV_j^y)\left(P_A^x-P_B^x+s\sum_{j=1}^{n}V_j^x\right)}{\sum_{j=1}^{n}(V_j^x)^2\sum_{j=1}^{n}(V_j^y)^2-\left(\sum_{j=1}^{n}V_j^xV_j^y\right)^2} \tag{13}$$

Thus we can conclude

Proposition 5. *Given two point-orientation pairs (P_A,θ_A) and (P_B,θ_B), and also n,s,α,κ_1, let $\Delta\theta_i=\alpha(i-1)s^2+\kappa_i s$ and δ_i be defined by (12). Then the G^1 continuous arc spline defined by P_A,θ_A, $s+\delta_1,\Delta\theta_1,\ldots,s+\delta_n,\Delta\theta_n$ interpolates the two point-orientation pairs.*

3.2. Optimal arc-length and initial curvature

We have seen that the perturbations are actually functions of s and κ_1. Now we want to find the optimal s and κ_1 such that the sum of squares of δ_i, $\sum_{i=1}^{n}\delta_i^2$, is minimized. In the ideal situation, all $\delta_i=0$ and the curve is an EAS.

3.2.1. Bound estimation on parameters

First, the length of the arc spline curve must be greater than the length of line segment connecting P_A and P_B, which implies

$$s\geq\frac{\|P_AP_B\|}{n}. \tag{14}$$

Second, substituting (6) into (5) gives

$$-2\pi\leq\frac{\theta_B-\theta_A}{n}-\kappa_1 s\leq 2\pi. \tag{15}$$

Combining this with inequality (4) leads to the following bounds on $\kappa_1 s$:

$$\max\left(-2\pi,-2\pi+\frac{\theta_B-\theta_A}{n}\right)\leq\kappa_1 s\leq\min\left(2\pi,2\pi+\frac{\theta_B-\theta_A}{n}\right). \tag{16}$$

Third, the combination of (15) with (4) also gives $-4\pi\leq\theta_B-\theta_A/n\leq 4\pi$. Thus

$$n\geq\frac{|\theta_B-\theta_A|}{4\pi}. \tag{17}$$

3.2.2. Levenberg–Marquardt solver

We introduce variable transformation, $\eta=\kappa_1 s$. Without ambiguity, we denote δ_i of (12) by $\delta_i(s,\eta)$. Let $\mathbf{f}(s,\eta)=(\delta_1(s,\eta),\ldots,\delta_n(s,\eta))^T:\mathcal{R}^2\to\mathcal{R}^n$ be a vector function. Then we formulate our problem as a minimization problem:

$$\min_{(s,\eta)\in\Omega}\|\mathbf{f}(s,\eta)\|^2=\min_{(s,\eta)\in\Omega}\sum_{i=1}^{n}\delta_i^2(s,\eta)$$

where $\Omega=\{(s,\eta):s\geq\|P_AP_B\|/n,\max(-2\pi,-2\pi+(\theta_B-\theta_A)/n)\leq\eta\leq\min(2\pi,2\pi+(\theta_B-\theta_A)/n)\}$. This is a nonlinear least square problem. We use the Levenberg–Marquardt (LM) algorithm to find the solution. The LM algorithm is one of the most widely used optimization algorithms [24,25]. It uses an effective damping strategy that makes it be able to converge quickly from a wider range of initial guesses.

The LM algorithm is an iterative procedure. Starting from an initial guess for (s,η), each iteration step replaces the current (s,η) by a new estimate $(s,\eta)+(\Delta s,\Delta\eta)$, where $(\Delta s,\Delta\eta)$ are the solution to a linear system

$$(\mathbf{J}^T\mathbf{J}+\mu\,\mathrm{diag}(\mathbf{J}^T\mathbf{J}))\begin{pmatrix}\Delta s\\ \Delta\eta\end{pmatrix}=-\mathbf{J}^T\mathbf{f}(s,\eta)$$

with the Jacobian matrix

$$\mathbf{J}=\begin{pmatrix}\frac{\partial\delta_1}{\partial s} & \frac{\partial\delta_1}{\partial\eta}\\ \vdots & \vdots\\ \frac{\partial\delta_n}{\partial s} & \frac{\partial\delta_n}{\partial\eta}\end{pmatrix},$$

the diagonal matrix $\mathrm{diag}(\mathbf{J}^T\mathbf{J})$ consisting of the diagonal elements of $\mathbf{J}^T\mathbf{J}$, and the damping factor μ. The damping factor is adjusted at each iteration. If the reduction of the objective function is rapid, a smaller value is used for μ; Otherwise, if an iteration gives insufficient reduction, μ is increased [26]. The pseudocode of this optimization is given in Algorithm 1, which is adapted from [26]. Here we heuristically set the initial estimation: $s=1.5\|P_AP_B\|/n$ and $\eta=0$.

Algorithm 1. (LM-Solver).

Input: A vector function $\mathbf{f}(s,\eta)=(\delta_1(s,\eta),\ldots,\delta_n(s,\eta))$ and an initial parameter estimation (s,η).
Output: Optimal (s,η).
Algorithm:

```
ν ← 2, μ ← 10^-3
ε1 ← 10^-10, ε2 ← 10^-10, k_max = 100
stop ← false
k ← 0
A ← J^T J + μ diag(J^T J), G ← −J^T f
while ((stop = = false) or (k < k_max)) do
  k ← k + 1
  repeat
    Solve the equation: (A + μ diag(J^T J))(Δs,Δη)^T = G
    if (‖(Δs,Δη)‖^2 ≤ ε1 ‖(s,η)‖^2) then
      stop ← true
    else
      Find a positive h such that (s,η) + h(Δs,Δη) ∈ Ω
      (s_new,η_new) ← (s,η) + min(1,h)(Δs,Δη),
      f_new ← f(s_new,η_new),
      J_new ← J(s_new,η_new)
      d = (‖f‖^2 − ‖f_new‖^2) / ((Δs,Δη)(μ diag(J^T J)(Δs,Δη)^T + G))
      if (‖f_new‖^2 < ‖f‖^2) then
        (s,η) ← (s_new,η_new), f ← f_new, J ← J_new
        A ← (J^T J), G ← (−J^T f)
        stop ← ((‖G‖_∞ ≤ ε1) or (‖f‖^2 ≤ ε2))
        μ ← μ max {1/3, 1−(2d−1)^3}
        ν ← 2
      else
        μ ← μν
        ν ← 2ν
      endif
    endif
  until (d > 0) or (stop)
end while
```

3.3. Algorithm

Now we are ready to summarize the curve completion algorithm. Given two point-orientation pairs (P_A,θ_A) and (P_B,θ_B) as input, the algorithm proceeds as follows:

Step 1. Use Algorithm 1 to find the optimal arc-length s and initial curvature κ_1.
Step 2. Use Eq. (6) to compute α.
Step 3. Use Eq. (12) to compute δ_k.
Step 4. If all δ_k are zero, we obtain an Euler arc curve defined by $P_A,\theta_A,n,s,\alpha,\kappa_1$. In case not all δ_k are zero (probably due to the behavior of the numerical solver), we modify the Euler arc curve to a G^1 arc spline, which is defined by P_A,θ_A, $s+\delta_1,\Delta\theta_1,\ldots,s+\delta_n,\Delta\theta_n$ where $\Delta\theta_i=\alpha s^2(i-1)+\kappa_1 s$. In this way, the output curve is guaranteed to interpolate the two point-orientation pairs.

Note that in the above procedure, the user is required to specify the number of arcs. In practice, we can let the algorithm automatically determine it. The basic idea is as follows: First, an initial number is given as the number of arcs. We compute the Euler arc spline interpolating the given point-orientation pairs. Second, we increase the number of arcs by a constant and recompute the Euler arc spline. Third, we compare the distance between the two Euler arc spline curves. Since the two curves may have different arc-lengths, we evenly sample both curves, compute the distance between two corresponding points, and use the maximum distance as an upper bound on the distance of the two curves. If the distance bound is greater than a given tolerance, we go back to the second step to increase the number of arcs. This process continues until the distance bound is smaller than the tolerance. Then we output the curve with the larger number of arcs.

4. Experimental results

This section provides experimental results to validate the algorithm. We have implemented our algorithm using C++. The test was conducted on HP xw6600 Workstation with 2.00 GHz CPU and 3.00 GB of RAM. The testing results show that the proposed algorithm is fast enough for realtime applications. The running time for all the examples given in this section is less than 0.02 s using the proposed algorithm. The rest of the section then reports the performance of the proposed algorithm in other aspects.

We first show that when the number of arcs increases, Euler arc curves converge to the Euler curves, which complies the results found in [21]. Fig. 7 shows three Euler curves (in pink color) and their approximation by Euler arc splines (in gray) with different numbers of arcs. All the Euler arc spline curves interpolate the boundary conditions. The statistics of the length of the Euler curves and the approximation errors is given in Table 1. The approximation error is computed by evenly sampling the curves and computing the distance between the corresponding points, and we use the maximum distance as the error. It can be seen that the errors drop quickly as n increases. We would also like to point out that the approximation error may better be measured by Hausdorff distances, but Hausdorff distances are much more expensive and difficult to compute in our case due to complicated transcendental function representation of Euler curves.

Paper [1] has revealed that completion curves using Euler curves are intuitive and natural. Thus our Euler arc splines are also expected to provide intuitive and natural curve completion. Fig. 1(b) and (d) show the results of shape completion using our proposed method. Fig. 8 demonstrates our method with a variety of point-orientation configurations. In Fig. 8(a), the start angle is $\theta_A=-\pi/2$ and the end angle θ_B changes around from π to $-\pi$. In Fig. 8(b), $\theta_A=-\pi/2$ with θ_B varying around from $-\pi$ to -2π.

We also implement the methods of [1,23] for comparison. In the following experiments, the starting point and its orientation are shown in red colors and the ending point and its the orientation are shown in blue. As pointed out in Section 1.2, due to complicated nonlinear equations in Kimia et al.'s method, it is not rare that the numerical procedure fails to find the exact solution, and thus the resulting Euler curve would not achieve the interpolation condition. Fig. 9 shows the results of shape completion using Kimia et al.'s method and our method. It is observed that Kimia et al.'s method does not produce an Euler curve that interpolates the ending point while our algorithm guarantees the interpolation of both endpoints. Walton and Meek's method [23] produces the results which are not different visually from the results produced by our method. However, Walton and Meek's

Table 1
Statistics for approximate errors.

Curve	Length of the Euler curve	Maximum error between the Euler curve and the Euler arc spline of n arcs			
		$n=10$	$n=20$	$n=30$	$n=40$
Fig. 7(a)	597	37.7	12.1	3.9	0.88
Fig. 7(b)	569	33.1	8.6	1.7	0.75
Fig. 7(c)	445	26.5	10.9	4.9	0.82

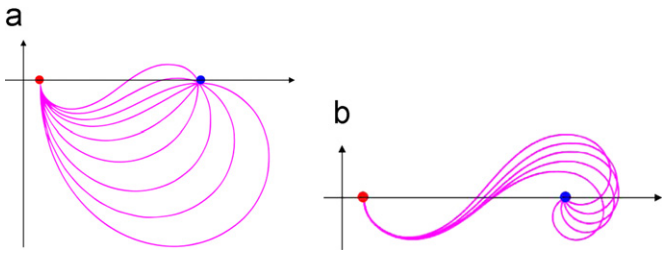

Fig. 8. Curve completion for various point-orientation configurations.

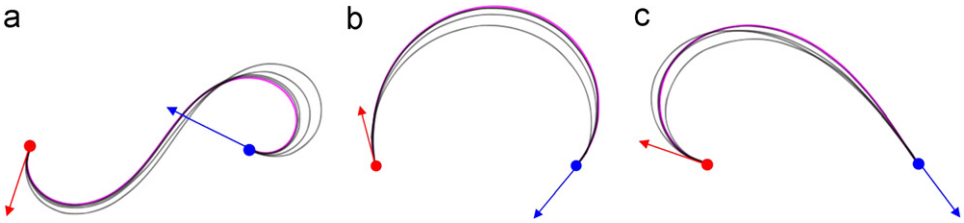

Fig. 7. Examples of Euler curves in pink color and a series of Euler arc splines shown in gray whose numbers of arcs are 10, 20, 30 and 40. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

method [23] has a limitation on the angle range while Kimia et al.'s method and ours can accept arbitrary orientation angles. Fig. 10 shows that with the same tangent vectors at the two endpoints we can set different orientation angle values to achieve different completions using our proposed method, which may be useful in practice. Due to the limitation on the angle range, Walton and Meek's method cannot produce Fig. 10(c) or (d).

Finally, the use of Euler arc splines for shape completion is illustrated in Fig. 11, where the constructed Euler arc splines are overlayed on synthetic occluders. In these examples, we manually specify the two points and the corresponding orientations, which are the inputs for the algorithm. It is also possible to extract the two points and the corresponding orientations by computing the intersection between the contour and the area to be completed, and the tangent directions of the contour. With the presence of completion curves, better image inpainting can be achieved. Fig. 12 shows such examples. Fig. 12(a) are three images with occluders and their corresponding completed images by conventional patch-based image inpainting. In Fig. 12(b), we perform the shape completion first. The completion curves separate the objects from the backgrounds and split the occluders into two sub-regions. Then we apply image inpainting on two sub-regions separately to allow for proper filling-in of the sub-regions. In this way, the structure of the objects is well preserved and unnecessary diffusion is avoided in image inpainting, which is depicted in Fig. 12(b).

5. Conclusion

We have described a new solution to curve completion. The main contribution of the paper is twofold. First, we introduce Euler arc splines as a new completion curve representation. Euler arc spline curves can be considered as an extension of Euler curves. Similar to Euler curves, Euler arc splines have several nice properties desired by the aesthetics of curves and meanwhile they can be represented by NURBS curves, making them easier to use in conjunction with existing commercial software systems. Second, we propose an algorithm to construct an Euler arc spline curve to interpolate two given point-orientation pairs for shape completion. The construction is converted into a problem of minimizing perturbations applied to each arc segment, which can be solved by the Levenberg-Marquardt algorithm. Compared to previous methods, our method has an obvious advantage that it guarantees the interpolation of point-orientation pairs. The use of the proposed algorithm has been demonstrated in image inpainting application. We believe that Euler arc splines and the associate curve completion algorithm can find other applications involving shape design and processing.

Though we always find a solution for all the inputs we tried in our experiments, theoretical analysis of existence of an Euler arc spline for given point-orientation pairs remains an open question,

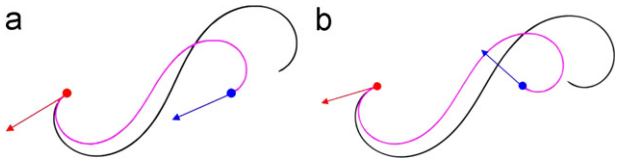

Fig. 9. Comparison of our method and Kimia et al.'s method. The curves generated by our method is shown in pink color and the curves generated by Kimia et al.'s method is shown in black: (a) $P_A=(400,320),\theta_A=3.932;P_B=(650,320),\theta_B=3.494$, (b) $P_A=(400,320),\theta_A=3.433;P_B=(650,320),\theta_B=2.295$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

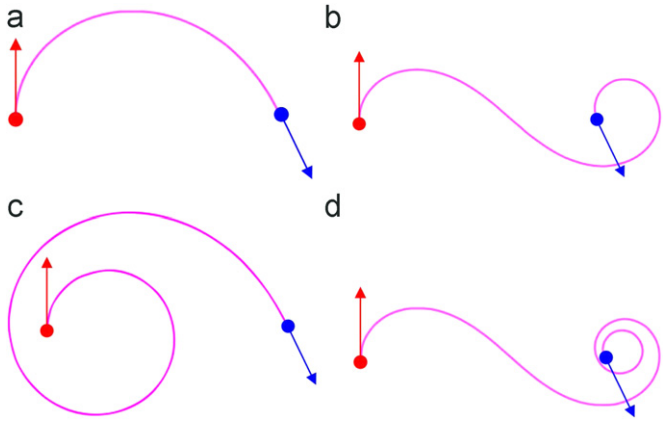

Fig. 10. Curve completion for the same tangent vectors but different angle values: (a) $\theta_A=90^\circ,\theta_B=-66^\circ$, (b) $\theta_A=90^\circ,\theta_B=-66^\circ+360^\circ$, (c) $\theta_A=90^\circ+360^\circ,\theta_B=-66^\circ$, (d) $\theta_A=90^\circ,\theta_B=-66^\circ+720^\circ$.

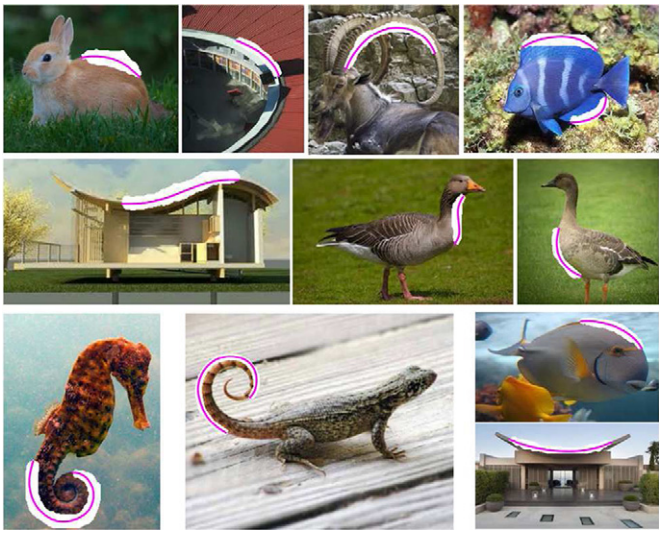

Fig. 11. Shape completion using Euler arc spline curves.

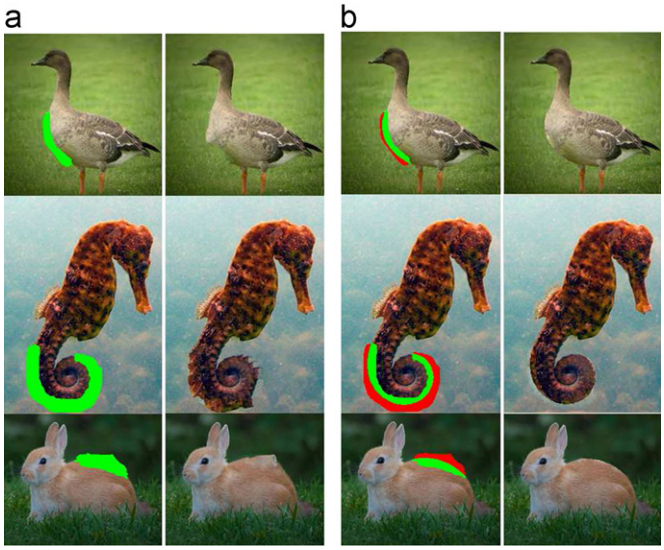

Fig. 12. Comparison of image inpainting: (a) without shape completion, (b) with shape completion.

which is worth further investigation. Also, the analysis of uniqueness of the solution is interesting as future work.

Acknowledgments

This work is supported by ARC 9/09 Grant (MOE2008-T2-1-075) from Singapore.

References

[1] Kimia BB, Frankel I, Popescu AM. Euler spiral for shape completion. Int J Comput Vision 2003;54(1–3):159–182.
[2] Ben-Yosef G, Ben-Shahar O. Minimum length in the tangent bundle as a model for curve completion. In: Proceeding of IEEE conference on computer vision and pattern recognition; 2010. p. 2384–2391.
[3] Harary G, Tal A. 3D Euler spirals for 3D curve completion. In: Proceeding of the symposium on computational geometry, ACM; 2010. p. 393–402.
[4] Farin G. Curves and surfaces for computer aided geometric design. Academic Press; 2002.
[5] Nitzberg M, Mumford D, Shiota T. Filetring, segmentation and depth. New York: Springer-Verlag; 1993.
[6] Yong J-H, Cheng F. Geometric hermite curves with minimum strain energy. Comput Aided Geom D 2004;21(3):281–301.
[7] Ullman S. Filling-in the gaps: the shape of subjective contours and a model for their generation. Biol Cybern 1976;25(1):1–6.
[8] Rutkowski WS. Shape completion. Comput Graph Image Process 1979; 9(1):89–101.
[9] Brady M, Grimson W. Shape encoding and subjective contours. In: Proceedings of the 1st annual national conference on artificial intelligence; 1980. p. 15–17.
[10] Knuth DE. Mathematical typography. Bull Am Math Soc 1979;1(2):337–372.
[11] Singh M, Fulvio J. Visual extrapolation of contour geometry. Proc Natl Acad Sci USA 2005;102(3):939–944.
[12] Horn BKP. The curve of least energy. ACM Trans Math Softw 1983; 9(4):442–460.
[13] Levien R. The Elastica: a mathematical history. Technical report UCB/EECS-2008-103. Berkeley: EECS Department, University of California; 2008.
[14] Mehlum E. Nonlinear splines. Comput Aided Geom D 1974:173–207.
[15] Mumford D. Elastica and computer vision. In: Algebraic geometry and its applications; 1994. p. 491–506.
[16] Harary G, Tal A. The natural 3d spiral. Comput Graph Forum 2011; 30:237–246.
[17] Walton D, Meek D. A controlled clothoid spline. Comput Graph 2005;29(3): 353–363.
[18] Montes N, Herraez A, Armesto L, Tornero J. Real-time clothoid approximation by rational Bézier curves, in: Proceeding of international conference on robotics and automation; 2008. p. 2246–2251.
[19] Mielenz KD. Computation of Fresnel integrals II. J Res NIST 2000;105(4):589.
[20] Wang LZ, Miura KT, Nakamae E, Yamamoto T, Wang TJ. An approximation approach of the clothoid curve defined in interval $[0,\pi/2]$ and its offset by free-form curves. Comput Aided Des 2001;33(4):1049–1058.
[21] Meek DS, Walton DJ. An arc spline approximation to a clothoid. J Comput Appl Math 2004;170(1):59–77.
[22] Sanchez-Reyes J, Chacon JM. Polinomial approximation to clothoids via s-power series. Comput Aided Des 2003;35(14):1305–1319.
[23] Walton DJ, Meek DS. An improved euler spiral algorithm for shape completion. In: Proceeding of canadian conference on computer and robot vision; 2008. p. 237–244.
[24] Levenberg K. A method for the solution of certain non-linear problems in least squares. Q Appl Math 1944;2(2):164–168.
[25] Marquardt DW. An algorithm for the least-squares estimation of nonlinear parameters. SIAM J Appl Math 1963;11(2):431–441.
[26] Lourakis MIA. A brief description of the Levenberg-Marquardt algorithm implemented by levmar, Institute of Computer Science Foundation for Research and Technology-Hellas; 2005. Available at 〈http://www.ics.forth.gr/lourakis/levmar/levmar.pdf〉.