A generic shape descriptor using Bezier curves

Conference or Workshop Item

A Generic Shape Descriptor using Bezier Curves

Ferdous Ahmed Sohel, Gour C. Karmakar, Laurence S. Dooley
Gippsland School of Computing and Information Technology
Monash University, Churchill, Victoria – 3842, Australia.
{Ferdous.Sohel, Gour.Karmakar, Laurence.Dooley}@infotech.monash.edu.au

Abstract

Bezier curves are robust tool for a wide array of applications ranging from computer-aided design to calligraphic character outlining and object shape description. In terms of the control point generation process, existing shape descriptor techniques that employ Bezier curves do not distinguish between regions where an object's shape changes rapidly and those where the change is more gradual or flat. This can lead to an erroneous shape description, particularly where there are significantly sharp changes in shape, such as at sharp corners. This paper presents a novel shape description algorithm called a generic shape descriptor using Bezier curves (SDBC) *which defines a new strategy for Bezier control point generation by integrating domain specific information about the shape of an object in a particular region. The strategy also includes an improved* dynamic fixed length coding *scheme for control points. The SDBC framework has been rigorously tested upon a number of arbitrary shapes, and both quantitative and qualitative analyses have confirmed its superior performance in comparison with existing algorithms.*

Keywords: *Shape description, Bezier curve, control points, significant points, supplementary points.*

1. Introduction

Bezier curves (BC) were independently developed by P. de Casteljau [1] and P. E. Bézier [2] in the previous century, and while their origin can be traced to the design of car body shapes in the automotive industry, their use is no longer confined to this field. Indeed, their robustness in curve and surface representation means they pervade many areas of multimedia technology including shape description of characters [3-4] and objects [5], shape error concealment for MPEG-4 objects [6] and surface mapping [7-8]. The classical BC is defined by a set of *control points* with the number and orientation of these points governing the shape of the curve. The actual computation of the control points that are derived from the shape points, are thus crucial to the efficacy of any BC application.

In [3-4], BC was used to describe the outline of Arabic and Chinese calligraphic characters respectively. In both cases, shapes were described by composite Bezier [12] curves comprising multiple segments, each of which was described by an individual curve. In each segment, the start and end control points determine the respective segment end points. All other control points were generated in such that they are located mainly outside the shape region, so the size of the shape increases, as does the number of bits for shape description. Major drawbacks of the technique include the computational expensive and that it applies a trial-and-error approach to generating control points.

In [5], cubic Bezier curves were applied to each shape segment, however the control points were evenly distributed irrespective of a shape's complexity, so the same consideration was given to a region where the shape changed little as it did if it changed very rapidly. As a consequence the decoded representation using this shape descriptor had a larger distortion in the areas with rapid shape changes. In addition, [5] adopted a parametric control point definition for each shape segment. The first and last points were defined using their coordinate values, while the second and third control points were subsequently defined using the magnitude and gradient of the tangent vector of the first and last control points respectively. This meant a large shape descriptor definition, comprising the absolute location of the two end-control points and the

corresponding magnitude and gradients to form two new control points for a particular segment.

Clearly the approach used in [5] results in a very high communications overhead and given the innate bandwidth restrictions of current communications technology, applications such as video streaming over the Internet, video-on-demand and mobile video transmission for handheld devices could all benefit significantly from faster and more efficient shape coding strategies. This was the motivation behind this research, namely to develop shape description schemes that minimize the descriptor size and hence reduce communication cost.

This paper presents *a generic shape descriptor using Bezier curves (SDBC)* which seeks to reduce both the distortion and shape descriptor size by simultaneously defining a new strategy for control point generation by incorporating domain specific information upon an object's shape and also by integrating an improved coding scheme. The performance of the SDBC framework has been extensively analysed and both quantitative and qualitative results clearly confirm its superiority compared with [5].

The rest of the paper is organised as follows: Section 2 provides a short overview of the classical BC, while Section 3 explores the existing BC-based shape description scheme [5] along with its limitations. Section 4 presents the new SDBC shape description framework including a novel dynamic fixed length coding scheme which improves the coding efficiency. Experimental results are presented in Section 5, confirming the improved performance of the SDBC model relative to the existing shape description technique using BC, with some conclusions given in Section 6.

2. Overview of the classical Bezier curve

The Bezier curve is a recursive linear weighted sub-division of the edges of the generated polygon starting with a set of points to form the initial polygon and ending when the final point is generated for a particular weight t. The set of $N+1$ starting points is referred to as the *control points* which govern the characteristics of the Bezier curve of degree N. The polygon connecting the control points is known as the *control polygon*. The Casteljau form of the Bezier curve for an ordered set of control points $V=\{v_0, v_1, \ldots, v_N\}$ is defined as:-

$$v_i^r(t)=\begin{cases} i^{th}\ memberof\ V, v_i; & if\ r=0; \\ (1-t)v_i^{r-1}(t)+tv_{i+1}^{r-1}(t); & r=1,\cdots,N;\ \ i=0,\cdots,N-r;\ \ 0\le u\le 1 \end{cases} \quad (1)$$

where t is the weight of subdivision which determines the number of points on the Bezier curve. The *final* generation $v_0^N(t)$ is the Bezier curve. The Casteljau form (1) algorithmically develops into a Bezier curve. An explicit functional form of the BC in the Bernstein form is given by:-

$$v(t)=\sum_{k=0}^{N} v_k B_k^N(t) \quad (2)$$

$$\text{where } B_k^N(t)=\binom{N}{k}(1-t)^{N-k}t^k \quad (3)$$

$\binom{N}{k}$ is the combination function. Either BC form will generate exactly the same curve for the same set of control points.

3. Existing Shape Description Techniques using Bezier Curves

The shape descriptor in [5] uses a cubic BC, with the entire shape subdivided into a pre-defined number (SR=segment rate) of segments, each having an equal number of consecutive shape points. For each segment, four control points are calculated and these describe he actual object shape. For instance, if a given shape $P=\{p_0, p_1, \ldots, p_{M-1}\}$ of M shape point is defined with SR number of curve segments, then each segment will describe the shape of $m=\frac{M}{SR}$ shape points. The control points for a particular segment covering shape points from p_i to p_{i+m} are heuristically calculated as:-

$$v_0 = p_i;\ v_1 = p_{i+\lceil \frac{m}{4} \rceil};\ v_2 = p_{i+\lceil \frac{3\times m}{4} \rceil};\ v_3 = p_{i+m} \quad (4)$$

As mentioned above, the first and last control points respectively define the end-points of the segment and the intermediate (the second and third) control points are subsequently defined from these points. The distance between the second and first control points is equal to the distance between the third and the last control points. A parametric description of each BC segment has been incorporated in [5] including the position of the two endpoints, plus the magnitude and gradient of the tangent vectors at those endpoints, approximated by the intermediate control points. Each segment is then joined sequentially to form a composite Bezier which describes the entire shape.

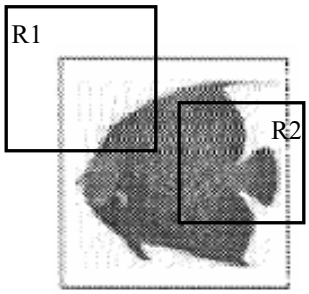


Figure 1: Object with flat (R1) and sharply changing (R2) regions on the shape.

The even spacing of the control points in [5] however, means the shape descriptor do not always properly represent sharply changing curvatures. As the example in Figure 1 highlights, region R1 has a comparatively smooth (regular) shape contour, while R2 possess a number of rapidly changing curvatures. Since both shape segments and the number of control points per segment are evenly distributed, it is clear that certain segments will give disproportionate emphasis to the relatively smooth part of the shape R1 and insufficient attention to the curved regions of R2. In addition, the coding scheme of [5] generates a large number of bits which means it is unsuitable for very low bit rate video applications, so alternative methods to reduce the bit rate must be explored.

To address these two fundamental issues, a new generic *shape descriptor using Bezier curve (SDBC)* is presented which concomitantly defines an improved strategy for BC control point generation by incorporating domain specific shape information, as well as a novel *dynamic fixed length coding* scheme for the control points.

4. Generic Shape Description Technique

The proposed generic shape description framework comprises two major elements. Firstly, the definition of the control points for each of the shape segments and secondly, a new coding strategy for these control points based on a combination of chain [9] and run-length coding (RLC). These are respectively detailed in the following two subsections.

4.1. Control point determination

In the SDBC framework, control points are selected from the shape points, however rather than considering all shape points in calculating the control points, the concept of *significant points* is introduced, so that regions having more rapidly changing shape features require more significant points than flatter regions. Formally, s*ignificant points* are those ordered shape points which are least in number and can generate the original shape entirely with zero error. Thus the significant points are those shape points at which there is a change of direction over the shape. This takes account of domain specific shape information during the control point generation process and also means that consecutive significant points will not necessarily be separated by 1 *pel* as with shape points. The larger the distance between consecutive significant points, the greater will be their influence upon the shape. In these situations, shape descriptions based only on significant points can produce higher distortion because many influential significant points may be excluded from being control points. To reduce the likelihood of losing potential significant points as control points, *supplementary points* are inserted at equal distances between the significant points. Note, the greater the number of supplementary points the more the approximating shape tends towards the original, while fewer supplementary points may not be able to represent any potential significant points adequately. To balance these two extreme scenarios, the average distance between consecutive significant points is used as the metric to insert the supplementary points. This can be mathematically explained as follows:-

Let $S = \{s_0, s_1, \cdots, s_{l-1}\}$ be the set of significant points for a given shape P where l is the number of significant points. If $d(s_{k-1}, s_k)$ denotes the distance between two consecutive significant points s_{k-1} and s_k, the average distance between them is:-

$$d_{avg} = \frac{1}{l}\sum_{k=1}^{l-1} d(s_{k-1}, s_k) \quad (5)$$

So when $d(s_{k-1}, s_k) > d_{avg}$, supplementary points are inserted between s_{k-1} and s_k. The first point (sp_1) is placed a distance d_{avg} from s_{k-1} and if $d(sp_1, s_k) > d_{avg}$, then a further supplementary point is inserted a distance d_{avg} from (sp_1). This process continues until the distance between two consecutive significant (one of which may be a supplementary point) points is $\le d_{avg}$. These significant shape points are called *approximated boundary points* and are defined as $B = \{b_0, b_1, \cdots, b_{B-1}\}$ and are used to calculate the control points for the BC segments. In this paper, the cubic BC is used for shape description as lower-order curves are too inflexible in controlling the shape of the

curve, while higher degree curves introduce unwanted oscillations and higher computation overheads. The four control points for the cubic BC to represent a particular segment covering approximated boundary points from b_i to b_{i+z-1}, where $z = \frac{B}{SR}$ are calculated as:-

$$v_0 = b_i; v_1 = b_{i+\lceil \frac{z}{4} \rceil}; v_2 = b_{i+\lceil \frac{3\times z}{4} \rceil}; v_3 = b_{i+z-1} \quad (6)$$

This implies that the control points are generated in terms of the number of the approximated boundary points. It also denotes that the number of control points is taken in regular interval of the approximated boundary points. The number of approximated boundary points between v_0 and v_1 is equal to the number of approximated boundary points between v_2 and v_3, while the number between v_1 and v_2 is always double that between v_0 and v_1.

4.2. Control point coding

As the shape points are a ranked order set, both the set of significant and approximated boundary points are also ordered, which means it is feasible to encode the control points differentially. A combination of chain code (CC) [9] and RLC is used to encode the control points. The direction of a control point from its previous (immediate) control point is coded using the CC, with the distance between them being the length of the code. A 6-bit coding scheme has 64-different directions and a maximum $error \le 2.82^o$, which is acceptable for shape description. To encode the length of a run in a direction for a particular point, a novel *dynamic fixed length coding* (DFLC) technique is proposed as follows:-

4.2.1. Dynamic fixed length coding

For a given object shape, the size of a segment depends on the number of segments used to describe the shape, and the distance between control points depends on the segment size. Thus, the distance between control points depends on the number of segments used to describe that particular shape. As noted earlier, z is the number of approximated boundary points in each shape segment, so the maximum shape segment length is $z \times d_{avg}$. The maximum distance between control points is thus $\frac{z \times d_{avg}}{2}$ and this is encoded to $L_1 = \left\lceil \lg\left(\frac{z \times d_{avg}}{2}\right)\right\rceil bits$.

Similarly, to encode all the other distances between the control points requires $L_2 = \left\lceil \lg\left(\frac{z \times d_{avg}}{4}\right)\right\rceil$ bits. From the definition, it is clear that the length of L_1 is always one bit longer than that of L_2, so if the length of L_2 is transmitted to the decoder, it will be able to generate the length of L_1. Normally 4 bits are sufficient to denote the length of L_2, since for example, in the object shape in Figure 1, the total number of shape points is 213 and the number of approximated boundary points is 174 at a maximum of $d_{avg} = 1.8 pel$ apart. With SR=5, $L_2 = \left\lceil \lg\left(\frac{174 \times 1.8}{5 \times 4}\right)\right\rceil = \lceil 3.95 \rceil = 4$ bits, so $\lceil \lg 4 \rceil = 2 bits$ is required to represent the length of L_2. Considering $L_2 > 0$, 4 bits to represent the length of L_2 is more than sufficient to represent a large shape, for instance, using this length for L_2 a distance of up to 2^{16} pel between the first and second control points for a segment can be represented. Thus the segment length could be up to $4 * 2^{16}$ pel.

BC control point encoding uses the periodic pattern shown in Table 1, where **Dir** is the direction of a particular control point. The first segment comprises the absolute location of the first approximated boundary point followed by L_2 bits and its direction after which comes L_1 bits with direction and finally L_2 bits with direction. Each subsequent segment will not require a starting point and so they are defined in ordered sequence of L_2, L_1, L_2 as shown in Table 1. Since this pattern is periodic, the decoder will be easily able to detect and separately collect each data pattern so the transmitted sequence of the encoded data will be:-

Table 1: Complete encoded data table.

4- bit for length L_2	Start-ing point	Dir + L_2	Dir + L_1	Dir + L_2	Dir + L_2	Dir + L_1	Dir + L_2	…	Dir + L_2	Dir + L_1
	First Segment				Next segment			…	Last Segment	

Note, the leading $4 bits$ in the encoded sequence reserved to represent the length of L_2. The complete generic *Shape Description using Bezier Curve* (SDBC) algorithm is formalised as follows:-

Algorithm 1: Generic Shape Description using Bezier Curve (SDBC) Algorithm.

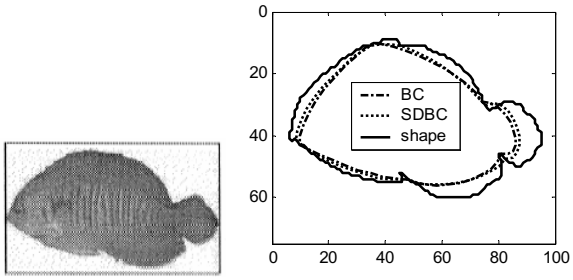
1. Find the significant points from the shape points (Section 4.1).
2. Determine the approximated boundary points by adding supplementary points with the significant points by applying the strategy in Section 4.1.
3. Calculate the control points for each of the segments using (6)
4. Encode the control points using directional coding along with the dynamic fixed length coding as (Section 4.2).

4.3. Decoding of the shape information

Due to the parametric representation of the encoded information and the periodic nature there-in the decoder knows the length and thus the delimiter of each parameter and so can correctly parse these parameters and hence reconstruct the shape using them.

5. Experimental Results and Analysis

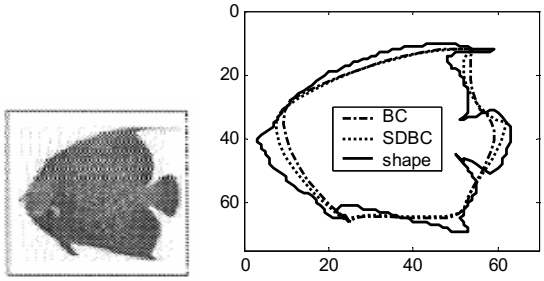
The widely-used *shape distortion measurement* metrics $D_{\max}$ and $D_{overall}$ were respectively used for the *class one* (maximum) and *class two* (overall) distortion measurements [10] for the numerical analysis, along with the distortion measure techniques in [11]. Figure 2b shows the various shape descriptions for the first test object (*Fish1*) in Figure 2a for a segment rate (SR=5). BC generated a peak distortion of 9.5*pel* in the tail region of the object, where there is considerable rapid change in the shape, while the SDBC framework had a corresponding class one distortion of 8.1*pel*.



(a) **(b)**

Figure 2: a) Test object (*Fish1*); b) Shape described by 5 segments.

When the entire object is considered, the SDBC framework provided a better shape description in comparison to BC as confirmed by the numerical results in Table 2, for the maximum and overall (Ovl) distortion values, for various segment numbers. For example, for SR=5, the BC and SDBC representations had overall distortions of 14 and 10.2 pel^2 respectively, which highlights that by considering domain specific shape information in selecting the control points for the BC, better performance is achieved.



(a) **(b)**

Figure 3: a) Test object (*Fish2*); b) Shape described by 5 segments.

The next series of experiments used the shape (*Fish2*) in Figure 3a, with the corresponding approximations given in Figure 3b for SR=5. Table 2 includes the results for a range of different segment rates and both the perceptual and empirical analysis confirms the SDBC model again provided better performance in comparison with the classical BC approximation.

Table 2: Distortion metrics in shape representation (Max distortion in pels; Ovl distortion in pel^2).

Fish		SR = 5		SR = 6		SR = 7		SR =8	
		Max	Ovl	Max	Ovl	Max	Ovl	Max	Ovl
1	BC	9.5	14	7.0	6.7	6.4	4.1	5.1	2.9
	SDBC	8.1	10.2	6.3	6	5.8	3.1	4.5	1.8
2	BC	7.6	9.1	7.0	5.8	6.0	4.3	5.2	2.8
	SDBC	6	6.6	5.6	3.5	5.4	3.6	4.8	1.8

In terms of the size of the shape descriptor, if one byte is respectively assigned to represent both the magnitude and gradient of the tangent of each control point, then on average 6 bytes are needed to represent each segment in the existing system [5]. It should be noted that in the original definitions, both the magnitude and gradient values were assumed to be floating point numbers and this commensurately increased the descriptor size. The absolute location of

the first control point requires 2 bytes and since the shape is closed, the final control point is not included, so only 4 bytes are used to define the final segment. Hence, with an average of 6 bytes descriptor per segment and SR=6, the existing BC descriptor will be 36 bytes (288 bits) long.

In the SDBC approach, each additional segment requires a maximum of only 31 bits (3 directional components of 6 bits each, two L_2 size each of 4 bits and one L_1 of 5 bits), with the final segment requiring only 21 bits. An overhead of 4 bits is incurred to define the length of L_2 and 2 bytes in defining the absolute position of the first control point of the first segment, so for SR=6, SDBC requires 196 bits which is over a 30% reduction compared with the BC representations. The BC system also requires 6 bytes (48 bits) for each additional segment of additional information while the SDBC needs only 31 bits, so SDBC provides an overall improvement of over 35% in the bit requirement for each additional segment using the *dynamic fixed length coding* technique detailed in Section 4.2.1.

The bit requirements for a number of different segment numbers for the two test shapes are summarised in Table . *Fish1* required 213 shape points and the cardinality of approximated boundary point set = 174 with $d_{avg} = 1.8 pel$. Hence for SR=6, L_2 is 4 bits; L_1 is 5 bits giving a total number of (4+16+31*5+ 21) = 196 bits.

Table 3: Bit requirements in shape representation.

Fish		SR = 5	SR = 6	SR = 7	SR =8
1	BC	240	288	336	384
	SDBC	165	196	227	258
2	BC	240	288	336	384
	SDBC	165	196	227	258

6. Conclusions

Cubic Bezier curves have been applied in many applications including describing object shapes. The critical aspect in using the Bezier curve is the proper selection of the control points. This paper has presented *a generic shape descriptor using Bezier curve (SDBC)* which provides a novel strategy in defining the control points by considering domain specific information about the shape of an object. Both qualitative and quantitative results have proven the improved performance of SDBC. The paper has also presented an improved *dynamic fixed length coding* strategy for more efficiently encoding the shape's control points.

7. References

[1] P. de Casteljau, *Outillage Méthodes Calcul*, Citroën, 1959.

[2] P.E. Bézier, *Employ des Machines á Commande Numérique*, Mason et Cie, Paris, 1970. Translated by Forrest, A. R., and A.F. Pankhurst as P. Bézier, *Numerical Control– Mathematics and Applications*, Wiley, London, 1972.

[3] M. Sarfraz, and M.A. Khan, "Automatic outline capture of Arabic fonts," *Information Sciences,* Elsevier Science Inc., pp. 269-281, 2002.

[4] H.-M. Yang, J.-J. Lu, and H.-J. Lee, "A Bezier curve-based approach to shape description for Chinese calligraphy characters," Proceedings of Sixth International Conference on Document Analysis and Recognition, pp. 276-280, 2001.

[5] L. Cinque, S. Levialdi, and A. Malizia, "Shape description using cubic polynomial Bezier curves," *Pattern Recognition Letters,* Elsevier Science Inc., pp. 821-828, 1998.

[6] L.D. Soares, and F. Pereira, "Spatial shape error concealment for object-based image and video coding," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 586-599, 2004.

[7] P. Direckx, *Curves and Surface Fitting with Splines*, Oxford Science Publications, 1993.

[8] R. Zhang, and G. Wang, "Some estimates of the height of rational Bernstein-Bezier triangular surfaces," *Proceedings of the Geometric Modeling and Processing*, pp.79-84, 2004.

[9] H. Freeman, "On the encoding of arbitrary geometric configurations," *IRE Trans. Electron. Computing*, vol. EC-10, pp. 260-268, June 1961.

[10] G.M. Schuster, and A.K. Katsaggelos, *Rate-Distortion Based Video Compression-Optimal Video Frame Compression and Object Boundary Encoding,* Kluwer Academic Publishers, 1997.

[11] F.A. Sohel, L.S. Dooley, and G.C. Karmakar, "A modified distortion measurement algorithm for shape coding," Proceedings of the 3rd Workshop on the Internet, Telecommunication and Signal Processing – WITSP04, Dec. 20-22, 2004.

[12] R.H. Bartels, J.C. Beatty, and B.A. Barsky, *An Introduction to Splines for use in Computer Graphics & Geometric Modeling,* Morgan Kaufmann Publishers, 1987.