A Mathematical Model for Shape Coding with B-Splines

Fabian W. Meier, Guido M. Schuster and Aggelos K. Katsaggelos

Abstract—
A major problem in object oriented video coding is the efficient encoding of the shape information of arbitrarily shaped objects. Efficient shape coding schemes are also needed in encoding the shape information of Video Object (VO) in the upcoming MPEG-4 standard. Furthermore, there are many applications where only the shape needs to be encoded, such as CAD, 3D modeling and signature encoding. In this paper, we present an efficient method for the lossy encoding of object shapes which are given as 8-connect chain codes using a mathematical model. We approximate a boundary by a second order B-spline curve and consider the problem of finding the curve with the lowest bit rate for a given distortion. The presented scheme is optimal, efficient and offers complete control over the trade-off between bit-rate and distortion. It is an extension of our previous research where we used polygons to approximate a boundary. The main reason for using curves rather than polygons is that curves have a more natural appearance than polygons and can give better coding efficencies. We present results of the proposed scheme using objects boundaries in different shapes and sizes as well as a MPEG-4 test sequence.

I. Introduction

A. Motivation for Shape Coding

This research is partly motivated by the importance of shape coding within the MPEG-4 standard, but also by object oriented video coding [1], [2], where the encoding of shape information of objects is an important problem. Furthermore, shape coding algorithms can also be used for other applications, such as CAD, object recognition and image description in video databases. In this paper we refer to the shape information of a single object as a *boundary* (sometimes also referred as contours), which can be represented with a chain code. Usually a boundary is defined in pixel resolution. In that case the boundary consists of a number adjacant pixels which we call *boundary points* (Figure 4 shows a simple boundary). A shape represented in a binary alpha plane has the same values for all the pixels that are inside the shape. A boundary is the contour around the the shape in the alpha plane. The new multimedia standard MPEG-4 [3], [4], makes it possible to encode several layers of overlapping video information at the same time, where each layer contains one or more arbitrarily shaped objects. In MPEG-4 terminology these layers are called Video Objects (VO). A Video Object Plane (VOP) is a snapshot of a VO at a given moment of time. For example, one VO contains a moving person as a foreground object - the area outside the person's shape would be transparent - and the second VO contains background frames (see Figure 1). A VOP is described by texture information (the actual pixel values within the object shape) and the shape information. The texture information is encoded with a conventional block based method (as intra or inter blocks), the difference to a conventional block based code is that only blocks that are partly or fully covered by the object shape will be encoded. The shape information is described by an alpha-plane and can be either in a gray-scale format or a binary format. In the binary alpha plane, all pixels within the object shape have the value 1 assigned, whereas all remaining pixels have the value 0 and are supposed to be completely transparent. In this paper[1] we are only concerned by shape information in binary format. Even though the encoding of the shape information and the encoding of the texture information can be viewed as two seperate steps, these two processes are related: for example in MPEG-4 a macroblock that lays on a shape boundary needs to be *padded* before encoding the texture. In [5] we proposed an extension which includes texture information for the boundary encoding.

B. Shape Coding Approaches

The performance of a shape coding scheme can be measured either with the absolute rate in bits or with a relative measure $e = R/N_B$ in *bits per boundary point (bbp)*, where R is the rate to encode the boundary and N_B the number of boundary points. For lossy encoding, using a coding efficiency measure is only meaningful if a certain distortion describes the degree of the lossy boundary approximation. In the following paragraphs we discuss some shape coding approaches.

Chain Coding: Freeman [6] originally proposed the use of chain coding for boundary quantization and lossless boundary encoding, which has attracted considerable attention over the last thirty years [7], [8], [9]. The most common chain code is the 8-connect chain code which is based on a rectangular grid superimposed on a planar curve. Since the planar curve is assumed to be continuous, the increments between grid points are limited to the 8 grid neighbors, and hence an increment can be represented by 3 bbp. There have been many extensions to this basic scheme such as generalized chain codes [7], where the coding efficiency has been improved with the use of links of different length and different angular resolution. There has also been interest in the theoretical performance of chain codes. In [8] the performance of different quantization schemes is compared,

A. Katsaggelos is with Northwestern University, Evanston IL 60208-3118, USA. Email: aggk@ece.nwu.edu

F. Meier is with Silicon Graphics, Mountain View, CA 94043, USA. Email: fabian@engr.sgi.com

G. Schuster is with U.S. Robotics, Advanced Technologies Research Center, Mount Prospect, IL 60056, USA. Email: gschuste@usr.com

[1] A preliminary version of this paper was first published in Proceedings of the 2nd Erlangen Symposium *Advances in Digital Image Communication* (Erlangen, Germany), pp. 75-84, April 1997.

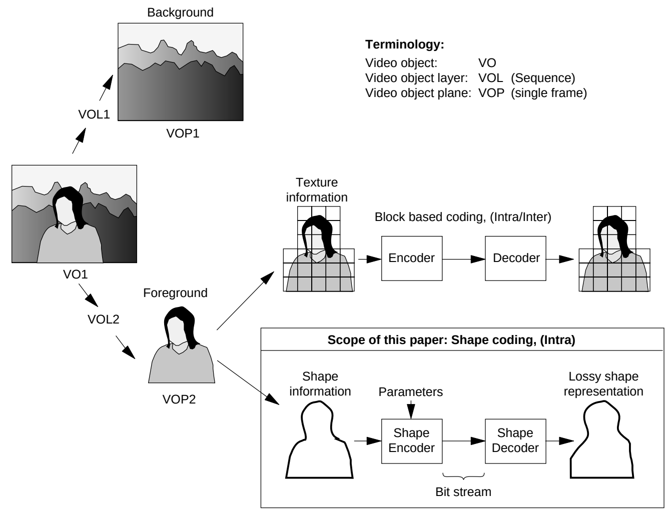

Fig. 1. The scope of this paper is an efficient shape coding scheme.

whereas in [9] the rate distortion characteristics of certain chain codes are studied. A lossy simplified chain code was presented in [10] and used in [11] for shape coding with bit-rates of 1.47 bbp.

Polygons: In [12], [13], [14] we found vertices in an optimal way to approximate boundaries with polygons given a certain distortion. The current paper is based on this work. A polygonal shape code was introduced in [15] with an improved vertex encoding scheme. Bit rates of 1.3 bbp for a maximal distortion of 1.0 pixels and 1.0 bbp for a maximal distortion of 2.0 pixels were achieved. An improved version of this method (INTRA and INTER coding) was evaluated in for the MPEG-4 shape coding part [16]. In an iterative scheme [17] vertices are forward predicted from the last frame, where new vertices are added and existing ones are dropped in order to represent the object shape with the required accuracy. The vertices locations are then encoded with a DPCM Huffman coding scheme.

Splines: Average bit-rates of 0.88 bbp were achieved in [18] with third order B-spline curves that approximated boundaries with a maximum distortion of 1.0 pixels. Because of the iterative approach the results are not unique nor optimal and depend on the initial curve. In [19] regions in a frame that contain motion were first segmented, and then lossy encoded with either macro-blocks for a coarse representation or with Catmull-Romm splines for a more accurate object shape representation. In this paper, we approximate boundaries with *quadratic B-splines* with a method that is based on a shortest path algorithm for a weighted directed acyclic graph (DAG) [13]. Note that this is achieved in an optimal fashion, with a complete control over the tradeoff between distortion and bit rate, resulting in an efficient encoding scheme. The Motivation for using B-splines are better coding efficiencies for objects in natural images. Such objects or shapes tend to have few straight lines and narrow corners. Furthermore, straight lines can be represented with second order B-splines. [20] reviews and compares our research on different approaches of optimal shape coding usiing polygons and B-splines.

The shape coding method chosen for MPEG-4, called *Contex-based Arithmetic Encoding* [21], uses an approach that encodes the black and white bitmap efficiently rather than a shape itself. The advantage of this approach is the robustness of the algorithm for arbitrary images and possible easy hardware implementation. The other major advantage of this approach is high coding efficiency, especially for lossless and quasi-lossless shape coding (intercoded frames). In [16] the different methods that were being considered for MPEG-4 shape coding are reviewed.

This paper is organized as follows. The B-spline curve is briefly reviewed in section II. In section III we define the problem and introduce the required notation. The definition of the set of admissible control points for the spline approximation is discussed in section IV. In section V we consider the problem of finding the B-spline curve which requires the smallest bit rate for a given distortion. We then consider the dual problem in section VI, that of finding the B-spline curve with the smallest distortion for a given bit rate. In section VII we introduce a control point encoding scheme. Finally, in section VIII we present simulation results of the proposed algorithm.

II. Review of B-spline curves

A B-spline is a specific curve type from the family of parametric curves [22]. A parametric curve consists of one ore more curve segments. Each curve segment is defined by $(n+1)$ *control points* where n defines the degree of the curve. The control points are located around the curve segment and together with a constant base matrix M the control points solely define the shape of the curve. A two dimensional curve segment Q_u with control points (p_{u-1}, p_u, p_{u+1}) is defined as follows:

$$Q_u(p_{u-1}, p_u, p_{u+1}, t) = \begin{bmatrix} x(t), & y(t) \end{bmatrix}, \quad \text{for } 0 \leq t \leq 1 \text{ and } 0 \text{ otherwise} \tag{1}$$

The points at the beginning and the end of a curve segment are called knots and can be found by setting $t = 0$ and $t = 1$. The following is the definition for a second degree curve segment, with u as index for the different curve segments, and $p_{u,x}$ and $p_{u,y}$, respectively, as the horizontal and vertical coordinates of control point P_u,

$$\begin{aligned} Q_u(p_{u-1}, p_u, p_{u+1}, t) &= T \cdot M \cdot P \\ &= \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix} \\ &\cdot \begin{bmatrix} p_{u-1,x} & p_{u-1,y} \\ p_{u,x} & p_{u,y} \\ p_{u+1,x} & p_{u+1,y} \end{bmatrix}. \end{aligned} \tag{2}$$

Both, the base matrix M, with specific constant parameters for each specific type of parametric curve, and the control point matrix P, with $(n+1)$ control points, define the shape of Q_u in a two dimensional plane. Every point of the curve segment can be calculated with Eq. (2) by letting t vary from 0 to 1. Every curve segment can be

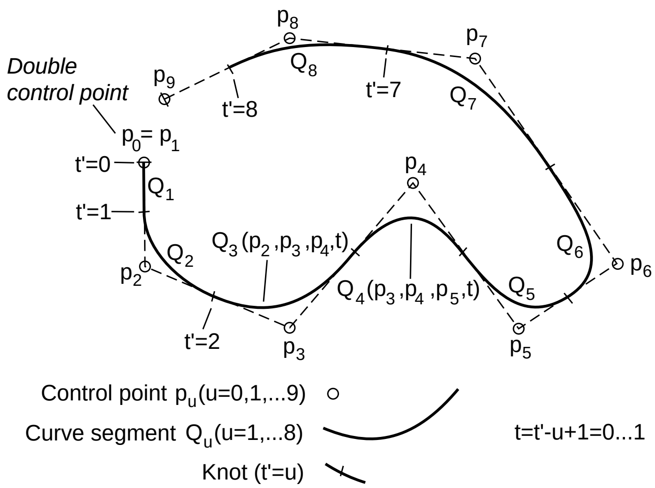

Fig. 2. A second degree B-spline curve with 8 curve segments Q_u and a double control point at the beginning of the curve.

calculated independently in order to calculate the entire curve Q, consisting of N_P curve segments, which is of the following form,

$$Q(t') = \sum_{u=1}^{N_P} Q_u(p_{u-1}, p_u, p_{u+1}, t' - u + 1), \quad \text{with } 0 \leq t' \leq N_P + 1. \tag{3}$$

Among common parametric curves are the Bezier curve and the B-spline curve. For our boundary approximation algorithm we chose a second order (quadratic) basis uniform non-rational B-spline curve [22] with the following base matrix,

$$M = \begin{bmatrix} 0.5 & -1.0 & 0.5 \\ -1.0 & 1.0 & 0.0 \\ 0.5 & 0.5 & 0.0 \end{bmatrix}. \tag{4}$$

Figure 2 shows such a second order B-spline curve. The reason for choosing the B-spline with the lowest possible order (n=2) was to keep the complexity of the curve, and the proposed algorithm, minimal. Note that a first degree B-spline is a polygon. The shape coding method presented in this paper is independent of the matrix M and degree n, that is, parametric curves of higher order can be used.

Double Control Points: The beginning and the end of the boundary approximation have to be treated as special cases, if the first curve segment should start exactly from the first boundary point and the last curve segment should end exactly at the last boundary point. When we use a double control point such as ($p_{u-1} = p_u$) the curve segment Q_u will begin exactly from the double control point (see Figure 2). We apply this property to the beginning and to the end of the curve, so that $p_0 = p_1$ and $p_{N_P} = p_{N_P+1}$. These two special cases can easily be incorporated into the boundary approximation algorithm.

III. Problem Formulation

The goal of the proposed approach is to approximate a given boundary by a second order B-spline curve: we have to find a set of control points which define a B-spline with the lowest possible rate given a certain distortion.

A. Notation

The following notation will be used. Let $B = \{b_0, \ldots, b_{N_B-1}\}$ denote the connected boundary which is an ordered set, where b_i is the i-th point of B and N_B is the total number of points in B. Note that in the case of a closed boundary, $b_0 = b_{N_B-1}$. Let $P = \{p_0, \ldots, p_{N_P+1}\}$ denote the set of control points of the B-spline curve, which is also an ordered set, with N_P the total number of curve segments. Every second order B-spline curve segment is defined by three control points p_{u-1}, p_u and p_{u+1}, henceforth denoted by $Q_u(p_{u-1}, p_u, p_{u+1})$ without the use of t as in Eq. (1), or simply by Q_u. Note that every curve segment shares control points with its neighboring curve segments. Since P is an ordered set, the ordering rule and the set of control points uniquely define the curve.

In general, the B-spline curve could be permitted to place its control points anywhere on the image plane. We restrict the possible locations of control points to a set A, which we will specify in detail in section IV.

We assume that the locations of the control points of the curve are encoded using a predictive scheme. This is an efficient method for natural boundaries since the locations of the control points are highly correlated. We denote the required bit rate for the encoding of control point p_{u+1}, given control point p_{u-1} and p_u, by $r(p_{u-1}, p_u, p_{u+1})$. Hence the total bit rate $R(p_0, \ldots, p_{N_P+1})$ for the entire B-spline is equal to

$$\sum_{u=0}^{N_P} r(p_{u-1}, p_u, p_{u+1}), \tag{5}$$

where $r(p_{-1}, p_0, p_1)$ is set equal to the number of bits needed to encode the absolute position of the first control point p_0. However, since this number depends on the size of the image plane, we neglect the rate for encoding the absolute position $r(p_{-1}, p_0, p_1)$ when we calculate the bit-rate e. Note that the rate $r(p_{u-1}, p_u, p_{u+1})$ depends on a specific control point encoding scheme, which will be discussed in section VII.

B. Distortion Measure

Besides the rate, we also need the curve segment distortion for our proposed curve approximation scheme, which we define as $d(p_{u-1}, p_u, p_{u+1})$. One popular distortion measure for curve approximation is the *maximum absolute distance*, which has also been employed in [8], [9], [2], [12], [15]. Because of its perceptual relevance we also use the maximum absolute distance in this paper. Other distortion measures are evaluating the difference area between the original and the approximated boundary shape [23], as well as, building a sum of all the segment maximum distortion measurements [14].

So far we have only discussed the segment distortion measures, i.e., the measures which judge the approximation of one part of the boundary by a given curve segment. In general we are interested in a curve distortion measure

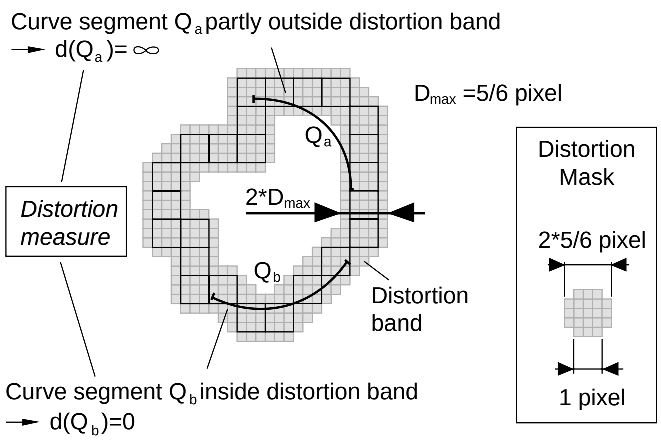

Fig. 3. The distortion band of width $2 \cdot D_{max}$ along the object boundary B. The band is constructed by moving a mask with radius r=D_{max} along the boundary of the object. The resolution of the distortion band is 1/3 pixel.

which can be used to determine the approximation quality of an entire curve. As mentioned above, we are using the maximum absolute distance distortion measure. The distortion measure for the whole curve using $d(p_{u-1}, p_u, p_{u+1})$ can be expressed as follows,

$$D(p_0, \ldots, p_{N_P+1}) = \max_{u \in [1,\ldots,N_P]} d(p_{u-1}, p_u, p_{u+1}). \quad (6)$$

Whereas the maximum absolute distance between a boundary and an edge of a polygon approximation can be calculated exactly [12], there is no unique method to measure the maximum absolute distance between a boundary and a curve approximation. However, the curve segment distortion can be defined as the distance of every boundary point to the closest point of its approximated representation. If we imagine a *distortion-band* with the width $2 \cdot D_{max}$ along the boundary B, a B-spline approximation must always be inside the band in order to satisfy the distortion requirement:

$$d(p_{u-1}, p_u, p_{u+1}) = \begin{cases} 0: & \text{all points of } Q_u(p_{u-1}, p_u, p_{u+1}) \\ & \text{are inside the distortion band} \\ & \text{of width } 2 \cdot D_{max} \\ \infty: & \text{any point of } Q_u(p_{u-1}, p_u, p_{u+1}) \\ & \text{is outside the distortion band} \\ & \text{of width } 2 \cdot D_{max} \end{cases} \quad (7)$$

This distortion measure takes a curve segment $Q_u(p_{u-1}, p_u, p_{u+1})$ as input and checks if the curve segment is fully inside the distortion band with width $2 \cdot D_{max}$.

Implementation of the Distortion Measure: The following method illustrates an efficient implementation of Eq. (7). A distortion band can be drawn by assigning all pixels to the band that are within a certain distance D_{max} from every boundary pixel. A given curve segment is quantized to pixel resolution, and then every curve pixel is tested whether it is inside or outside the distortion band. The disadvantage of this procedure is that values of D_{max} can only be multiples of one pixel. It can be overcome if we introduce a distortion band with sub-pixel resolution. For our experiments we chose a distortion band with a sub-pixel resolution of 1/3 pixel (see Figure 3). Clearly, the distortion band does not need to be symmetric . Without loss of generality a symmetric band is primarily considered in the following.

C. Optimization Problems

In the reminder of the paper we introduce a fast and efficient algorithm which solves the following constrained optimization problem,

$$\begin{array}{ll} \min_{p_0,\ldots,p_{N_P+1}} R(p_0, \ldots, p_{N_P+1}), & \\ \text{subject to:} \quad D(p_0, \ldots, p_{N_P+1}) \le D_{max}, & \end{array} \quad (8)$$

where D_{max} is the maximum distortion permitted. Note that there is an inherent tradeoff between the rate and the distortion in the sense that a small distortion requires a high rate, whereas a small rate results in a high distortion. Using the optimal solution of Eq. (8) iteratively we will also write a solution to the dual problem, that of finding a B-spline curve approximation with the lowest possible distortion given a maximum rate R_{max}, that is,

$$\begin{array}{ll} \min_{p_0,\ldots,p_{N_P+1}} D(p_0, \ldots, p_{N_P+1}), & \\ \text{subject to:} \quad R(p_0, \ldots, p_{N_P+1}) \le R_{max}. & \end{array} \quad (9)$$

IV. Admissible Control Point Set

From a theoretical point of view, the set of admissible control points for a B-spline boundary approximation should contain all pixels in the image plane. In order to keep the algorithm efficient, we restrict the control points to a set of relevant locations. We call this the set of admissible control points A and define it as a band along the boundary B, where the band is determined by W_{max} (see Figure 4). W_{max} is measured from the center of the boundary pixel to the center of the admissible control point pixel. A must be an ordered set to employ the presented boundary approximation algorithm. We therefore propose to order set A by assigning all points of A to their nearest boundary point and then imposing the order of the boundary onto the set A. The assigning algorithm [14] is explained in the next paragraph. Every admissible control point a_{i,i_b} has two indices: i and i_b. Index i has the same number as the index of its closest boundary point b_i. The second index i_b enumerates all admissible control points that have the same index i. Index i_b starts always at 0 and every point $a_{i,0}$ is by definition equal to b_i. Therefore, the simplest case is when A is equal B with $W_{max} = 0$. Clearly, for values of W_{max} larger than zero, a varying number of admissible control points are associated with every boundary point (compare the sets A and B in Figure 4). A good value for W_{max} is 1.0 pixel, which results in an admissible control point band of thickness equal to 3 pixels

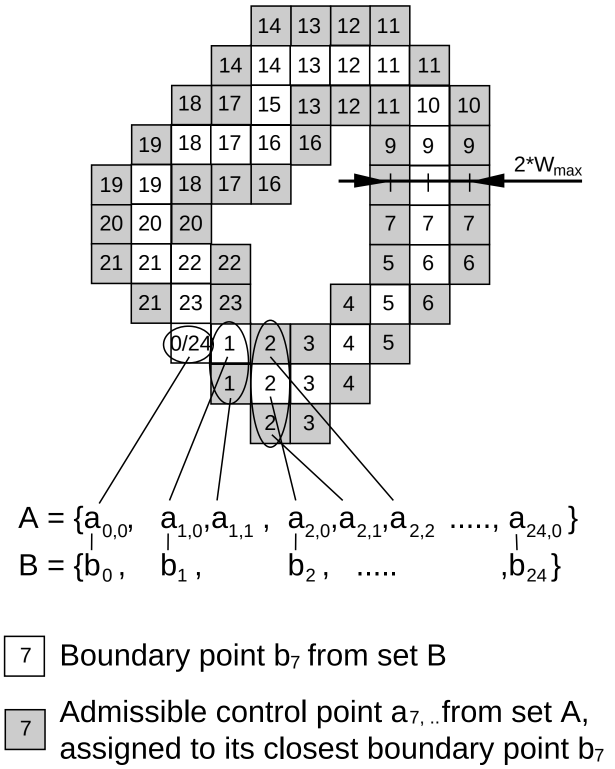

Fig. 4. The admissible control point set A is a band determined by W_{max}. Note that $b_0/b_{24} = a_{0,0}/a_{24,0}$ have no additional points assigned.

along boundary B. We further define that the first and the last boundary points have no additional admissible control points assigned, so that the B-spline will begin exactly at $b_0 = a_{0,0}$ and will end at exactly $b_{N_B-1} = a_{N_B-1,0}$.

Assigning Algorithm: The assigning algorithm of admissible control points consists of two loops, where the outer loop draws spirals around the boundary points. The radius W for drawing the spirals (see Table I) are values with discrete increments of range $0 \dots W_{max}$. The following algorithm assigns pixels from the image plane to set A as a function of W_{max}:

1. Set initial counter variable j equal to zero.
2. Pick a vector in Table I with the current variable j as index.
3. Add vector v with to the boundary point b_i. Assign the new point $(b_i + v)$ to A if this point is not yet an element of A.
4. Repeat line 3 for all boundary points.
5. Increase j by 1. If $W \leq W_{max}$ go back to line 2, else exit the algorithm.

W denotes the length of vector v: $W = \sqrt{v_x^2 + v_y^2}$.

V. Finding a Curve with the Lowest Rate

In this section we first show how to describe a mathematical model of the given boundary using a graph. In a

j	v_x	v_y	W	j	v_x	v_y	W
0	0	0	0.00	5	1	1	1.41
1	1	0	1.00	6	-1	1	1.41
2	0	1	1.00	7	-1	-1	1.41
3	-1	0	1.00	8	1	-1	1.41
4	0	-1	1.00	9	2	0	2.00
				10	0	2	2.00
				11	-2	0	2.00
				12	0	-2	2.00

TABLE I

Vector v with components v_x, v_y and length W as a function of index j.

second step we will find the B-spline curve with the lowest bitrate using a shortest path algorithm.

The B-spline curve that will be found by the proposed algorithm has the two following requirements:

- The distortion of the curve is smaller than or equal to the maximum distortion D_{max} as stated in Eq. (8).
- The control points must be selected from a specified admissible control point set A.

The key observation for deriving an efficient search is the fact that, given two control points p_{u-1} and p_u of a B-spline curve and the minimum rate which is required to code the curve up to and including these two control points, $R^*_u(p_{u-1}, p_u)$, the selection of the next control point p_{u+1} is independent of the selection of the previous control points $p_0, \dots, p_{u-2}$. This is true since the smallest total rate R^* can be expressed *recursively* as a function of the control point rates $r(p_{u-1}, p_u, p_{u+1})$, that is,

$$R^*_{u+1}(p_u, p_{u+1}) = \min_{p_{u-1}} \{R^*_u(p_{u-1}, p_u) + r(p_{u-1}, p_u, p_{u+1})\} \tag{10}$$

Recall that the distortion of the curve segment Q_u depends also on the three control points p_{u-1}, p_u and p_{u+1}. The segment distortion can be combined with the segment rate by defining a weight function w as follows,

$$\begin{aligned} &w(p_{u-1}, p_u, p_{u+1}) \\ &= r(p_{u-1}, p_u, p_{u+1}) + d(p_{u-1}, p_u, p_{u+1}) \end{aligned} \tag{11}$$

Note that w is equal to the rate for all the curve segments which satisfy the distortion constraint of Eq. (7), but infinite for those which do not. Hence by replacing r in Eq. (10) by w, the rate $R^*_{u+1}(p_u, p_{u+1})$ is infinite if a curve segment is used which violates the distortion constraint and equal to the required rate otherwise. The recursion of Eq. (10) needs to be initialized by setting $R^*_0(p_{-1}, p_0)$ equal to zero. Clearly $R^*_{N_P+1}(p_{N_P}, p_{N_P+1}) = \min_{p_0,\dots,p_{N_P+1}}\{R(p_0, \dots, p_{N_P+1})\}$, is the rate for the entire curve.

Because of the recursive relationship in Eq. (10), the problem in Eq. (8) can be formulated as a shortest path problem in a *weighted directed graph*. A vector $\vec{E}$ starts at control point $p_u = a_{i,i_b}$ and ends at control point

$p_{u+1} = a_{k,k_b}$ with the condition that both admissible control points cannot be assigned to the same boundary point $(\vec{E} = (a_{k,k_b} - a_{i,i_b}) \in A^2; \forall i \neq k)^2$. A path of order K from control point p_0 to control point p_K is an ordered set $\{p_0, \ldots, p_K\}$. The total length or total weight of the path is redefined as follows,

$$\sum_{u=1}^{K-1} w(p_{u-1}, p_u, p_{u+1}) \tag{12}$$

Again, note that the definition of the weight function w leads to a length of infinity for every path that includes a curve segment with an approximation error larger than D_{max}. Therefore a shortest path algorithm will not select a path with one or more distorted curve segments.

The classical algorithm for solving such a single-source shortest-path problem, where all the weights are non-negative, is Dijkstra's algorithm [24]. This is a significant reduction compared to the time complexity of the exhaustive search. We can further simplify the algorithm by observing that it is very unlikely for the optimal path to select a control point $p_{u+1} = a_{k,k_b}$ where the last selected control point was $p_u = a_{i,i_b}$ and $i > k$. This restriction results in that the selected curve approximation has to follow the original boundary without rapid direction changes, and more important, the resulting graph is now a *weighted directed acyclic graph (DAG)*. Hence the vector set is redefined in the following way, $(\vec{E} = (a_{k,k_b} - a_{i,i_b}) \in A^2; \forall i < k)^2$. A DAG-shortest-path algorithm [24] finds a single-source shortest-path and is faster than Dijkstra's algorithm. As can be seen from the DAG algorithm, the time complexity is cubic in the number of admissible control points, which very low for the resulting optimal selection.

Let $R^*(a_{j,j_b}, a_{i,i_b})$ represent the *minimum total rate* to reach the control point $p_u = a_{i,i_b}$ from the source control point $p_0 = a_{0,0}$ via a B-spline curve approximation. $p_u = a_{i,i_b}$ and $p_{u-1} = a_{j,j_b}$ are the two last control points of that curve. Clearly $R^*(a_{N_B-1,0}, a_{N_B-1,0})$ is the solution to problem (8), since it represents the minimum total rate to reach the last control point, which is by definition equal to the last boundary point.

The Sliding Window: With the formulation of the boundary approximation problem the solution of the DAG shortest path algorithm results in a trivial solution which is illustrated in Figure 5. This is a correct solution under the current conditions, but not a useful boundary approximation. We need a way to force the algorithm along the boundary in order to find a curve. With the introduction of a *sliding window* we not only avoid trivial solutions but also are able to control the speed of the algorithm. The sliding window restricts the selection of control point p_{u+1} to all the admissible control points within the sliding window. The length of the sliding window l_{sw} in pixel is measured along the boundary, starting at control point p_u. l_{sw} can be any value below $N_B/2$ for closed boundaries and below N_B for open boundaries. Changing l_{sw} gives us also the possibility to influence the smoothness of the B-spline curve where larger values of l_{sw} result in smoother curves.

[2] The following exception is necessary to allow double control points at the beginning and the end of the curve :
$i = k$ if $\{i = 0, i = N_B - 1\}$

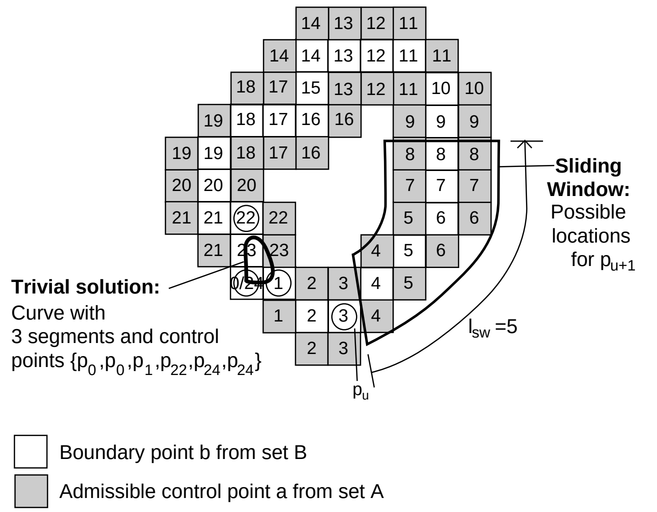

Fig. 5. The sliding window restricts the selection of control point p_{u+1} to all the admissible control points within the sliding window. The introduction of a sliding window prevents trivial solutions.

Example: Figure 6 shows an example how a DAG is derived from a boundary $(N_B = 7)$ and how the shortest path solution leads to a lossy shape approximation. The admissible control point set A is equal to B; the only valid control point locations are all boundary pixels. The reason of A being very small is to keep the complexity of this example as low as possible. The DAG in Figure 6 B) consists of states and vectors, in graph theory commonly called vertices and edges. Several states are associated with a single admissible control point a_i. Every state is uniquely described by two admissible control points (a_j, a_i), where a_j refers to the state's connecting previous state, with $(a_j \in A; \forall j \leq i)^1$. The total number of states that are associated with each admissible control point is $l_{sw} = N_B - i$, but in our example we restrict l_{sw} to 3. The sliding window starts at $p_u = a_{i+1}$ and includes all points of A up to $a_{i+l_{sw}}$. The length or weight of a vector is assigned with Eq. (11) in the unit bits. Note that function Eq. (11) has three input variables; the first two variables represent the two admissible control points associated by the state where the vector begins and the third variable is the admissible control point associated with the state where the vector ends. Every vector represents now a B-spline curve segment, so that any path in the DAG from state (a_0, a_0) to state (a_6, a_6) is a possible curve approximation of boundary B. Let $R^*_{(a_i,a_j)}$ be the best total rate of the path from the first state (a_0, a_0) to state (a_i, a_j); and let $Ptr(a_i, a_j)$ be a back pointer that is used to remember that path. $R^*_{(a_i,a_j)}$ is the sum of all the weights of that path.

The task of a shortest path algorithm is to find a path from state (a_0, a_0) to state (a_6, a_6) with the lowest total rate, which is clearly $R^*_{(a_6,a_6)}$. Because we interpret the length of a vector as the number of bits necessary to encode that vector, the shortest path is the path with the lowest total bit-rate. Once a shortest path is found, the control points for the B-spline approximation are all the admissible control points assigned to the states of this path (Fig. 6 D).

Pseudo Code for the Example: The following algorithm outlines the DAG-shortest path algorithm used to find the minimum rate of the previous example. The input variable to the algorithm is the admissible control point set A, with the parameters maximum distortion D_{max} and length of the sliding window l_{SW}. The output of the algorithm consists of an optimal path and its rate R^*. Then the proposed algorithm works as follows:

DAG shortest path algorithm:

1) $R^*(a_j, a_i)=\infty$; $i, j \in \{0, \ldots, N_B - 1\}$
2) $R^*(a_0, a_0) = 0$;
3)
4) for $i = 0, \ldots, N_B - 1$; Loop 1
5) {
6) for $j = i - l_{sw}, \ldots, i - 1$; Loop 2
7) {
8) for $k = i + 1, \ldots, i + l_{sw}$; Loop 3
9) {
10) $p_{u-1} = a_j$; $p_u = a_i$; $p_{u+1} = a_k$;
11) distortion of Q_u: $d_u = d(p_{u-1}, p_u, p_{u+1})$;
12) rate of $\vec{E_u}$: $r_u = r(p_{u-1}, p_u, p_{u+1})$;
13) weight: $w_u = d_u + r_u$;
14) if $R^*(p_{u-1}, p_u) + w_u < R^*(p_u, p_{u+1})$;
15) {
16) store the new lowest rate:
17) $R^*(p_u, p_{u+1}) = R^*(p_{u-1}, p_u) + w_u$;
18) assign back pointer to previous
19) state value:
20) $Ptr(p_u, p_{u+1}) = (p_{u-1}, p_u)$;
21) }
22) }}}

The optimal path with the lowest rate R^* ends at state (a_{N_B-1}, a_{N_B-1}). Now the complete optimal path $\{p_0^*, \ldots, p_{N_P+1}^*\}$ can be found by back tracking the pointers $Ptr(\cdot)$ in the following recursive fashion,

$$(p_{u-1}^*, p_u^*) = Ptr(p_u^*, p_{u+1}^*), \quad u = N_P, \ldots, 1 \tag{13}$$

with $Ptr(p_{N_P}^*, p_{N_P+1}^*) = Ptr(a_{N_B-1}, a_{N_B-1})$ as initial value. The formal proof of the correctness of the DAG-shortest-path algorithm, on which the above scheme is based, can be found in [24].

Detailed Explanation of the Pseudo Code: We will reason more intuitively how this approach works. In lines (1) and (2) the initial control conditions for the graph are defined. Loop 1 with the counter variable i selects all points in A in sequence. Loop 3 with the counter variable k selects all points from a_{i+1} up to the end of the boundary. Hence the loops 1 and 3 select each possible vector $\vec{E}$ in the DAG exactly once. The number of repetitions of the loops 2 and 3 can be restricted with the length of the sliding window l_{sw} in order to speed the algorithm up. A DAG shortest path algorithm for a polygon [12] needs only two loops because an edge segment is defined by two vertices. Lines (11) to (13) are used to calculate the weight $w(p_{u-1}, p_u, p_{u+1})$ of every curve segment. The most important part of this algorithm is the comparison on line (14). Here we test if the new bit rate, $R^*(p_{u-1}, p_u) + w(p_{u-1}, p_u, p_{u+1})$, to reach the admissible control point p_{u+1} – given that the last two control points were p_{u-1} and p_u – is smaller than the smallest bit rate used so far to reach the p_{u+1}. If the bit rate is indeed smaller, then it is assigned as the new smallest bit rate to reach the control point p_{u+1} on line (17). We also assign the back pointer of state (p_u, p_{u+1}) to state (p_{u-1}, p_u).

This algorithm leads to the optimal solution because, as stated earlier, if the rate $R^*(p_{u-1}, p_u)$ of two control points p_{u-1} and p_u is given, then the selection of any future control point p_{u+1} is independent of the selection of the past control points.

VI. Finding a Curve with for a Given Bit-rate

We now consider the minimum distortion case which is stated in Eq. (9). In the minimum distortion case we have a given rate budget R_{budget}, and we would like to find a curve with the lowest possible distortion where the corresponding rate must be equal to or smaller than R_{budget}. We propose an iterative solution to this problem which is based on the fact that we can solve the dual problem stated in Eq. (8) optimally. Consider D_{max} in Eq. (8) to be a variable. We derived in the previous section an algorithm which finds the curve approximation which results in the minimum rate for any D_{max}. We denote this optimal rate by $R^*(D_{max})$. Since $R^*(D_{max})$ is a non-increasing function (for a proof see [12]), we can use bisection to find the optimal D_{max}^* such that $R^*(D_{max}^*) = R_{max}$.

VII. Control Point Encoding Scheme

So far, any control point encoding scheme which satisfies the assumption that the control points are encoded differentially, i.e., the rate to encode point p_{u+1} depends only on the previous two points, p_u and p_{u-1}, could have been used. In this section we present a specific control point encoding scheme to encode the vector $\vec{E_u}$ between the control points p_u and p_{u+1}. The encoding scheme can be considered as a combination of a modified 8-connect chain code and a run-length encoding scheme [12]. The chain code and the run-length encoding can be combined by representing the vector between two control points by an angle α and a run β, which form the symbol (α,β). For each of the possible symbols (α,β) we encode the angle and the run independently. In this paper, we employ a entropy code for encoding the runs β, which is displayed in Table II. Note that the longest possible run is 15 so that any vector with length larger than 15 cannot be encoded. In consequence, setting the length of the sliding window l_{sw} to values above 15 does not result in lower bit-rates with this run-length encoding scheme.

Clearly it takes 3 bits to encode 8 equally probable directions with the 8-connected chain code. The eight angles have 45 degree increments, therefore there are only 8 available directions to encode the angle of the vector. This scheme together with Huffman coding for the run length of the vector was used in [12]. In natural boundaries, the arrival direction of a vector is highly correlated with the departure direction of the following vector. This implies that the arrival direction should be used to predict the departure direction (see Figure 8 (A)). We propose to use only 2 bits for α for the four most probable directions. These four relative directions of α are $\{-90^o, -45^o, +45^o, +90^o\}$, where 0^o is the direction of the previous vector (see Figure 8 (B)). The rate function $r(p_{u-1}, p_u, p_{u+1})$ must consider the case when a vector cannot be encoded. If this happens, the rate is considered infinite. This can be expressed as follows,

$$r(p_{u-1}, p_u, p_{u+1}) = \begin{cases} \text{rate}(\alpha) + \text{rate}(\beta): \\ \vec{E}_u \text{ is codeable} \\ \\ \infty: \\ \vec{E}_u \text{ is not codeable} \end{cases} \tag{14}$$

$$\text{rate}(\alpha) = \text{ 2 bit, rate}(\beta) = 2 \text{ to } 5 \text{ bit}$$

With this scheme we clearly restrict the set of codeable vectors. Experiments showed that using this encoding scheme with four relative angles results in boundary approximations with lower rates than when we used a scheme with 8 absolute angles.

Special Cases:

- Since we use relative angles to encode the vectors $\vec{E}$ it is necessary to encode the first vector $\vec{E}_0$ with an absolute angle; therefore we need 3 bits for the angle of the first vector.
- As mentioned before, the first two control points p_0 and p_1 are identical, so are the last two control points, p_{N_P} and p_{N_P+1}. These double control points make sure the curve starts exactly at the first boundary point and ends exactly at the last boundary point. Consequently $r(p_0, p_1, p_2)$ (rate of the first curve segment) and $r(p_{N_P-1}, p_{N_P}, p_{N_P+1})$ (rate of the last curve segment) are assigned to zero.
- In case of a closed boundary, the first two control points are identical to the last two control points. The vector from control point p_{N_P-1} to control point p_{N_P} is therefore redundant and does not need to be encoded; rate $r(p_{N_P-2}, p_{N_P-1}, p_{N_P})$ can be set to zero.

The proposed shape coding algorithm is not restricted to the just described control point encoding method. Other control point schemes might achieve lower rates for some shapes, the performance of different encoding schemes are shape dependent. The control point encoding scheme can be viewed as a *replacable module* to the proposed shape coding algorithm. For example an alternative run-length encoding scheme (Table III) could be used.

Run	Rate	Run	Rate	Run	Rate
1	2 bit	6	4 bit	11	5 bit
2	3 bit	7	4 bit	12	5 bit
3	3 bit	8	5 bit	13	5 bit
4	4 bit	9	5 bit	14	5 bit
5	4 bit	10	5 bit	15	5 bit

TABLE II

Run length encoding table. The bit rate of control point vector $\vec{E}$ as a function of its run length β.

Run	Rate	Run	Rate	Run	Rate
1	3 bit	6	∞	11	∞
2	2 bit	7	∞	12	3 bit
3	∞	8	3 bit	13	∞
4	2 bit	9	∞	14	3 bit
5	∞	10	∞	≤ 15	∞

TABLE III

Alternative run length encoding table.

VIII. Simulations

A. Simulation Conditions

To demonstrate the proposed shape coding scheme we encoded three different objects boundaries with 70 (Shape 1), 158 (Shape 2) and 257 (Shape 3) boundary points, the objects are shown in Figure 9. For the encoding simulations we varied the maximum distortion D_{max} from 0.4 to 4.0 pixels. We chose the following constant parameters for all simulations:

- W_{max}=1.0 (width of the admissible control point band A).
- l_{sw}=15 (length of the sliding window)
- The rate to encode the absolute position of the first control point is neglected since it is not exactly known.
- Use Table II for run length encoding.

The following parameters were used for 100 frames of the MPEG-4 test sequence kids:

- W_{max}=2.0 (width of the admissible control point band A).
- l_{sw}=14 (length of the sliding window)
- Include the rate to encode the absolute position of the first control point.
- Use Table III for run lenght encoding.

B. Results

Figure 10 shows the performance of the shape coding algorithm in function of the distortion. Average encoding rates in our experiences in the range of 0.70 ...0.84 bbp were achieved with distortion values of D_{max}=1.0 and e=0.57 ...0.64 with D_{max}=2.0. The bit-rates e for all three encoded boundaries go into a saturation for distortion values of 2.0 and larger. Figure 11 illustrate how better bit-rates are achieved on cost of the shape distortion. Note that a B-spline curve approximation with $D_{max} \approx 0.5$ has a distortion band with width of about 1.0 pixel, which is a lossless representation of the boundary. In this case an encoding scheme that incorporates 8 encodable angles rather than 4 will achieve better encoding rates.

Figure 12 shows the B-spline curve approximation for Shape 1 given $D_{max} = 1.0$. The curve is, according to the algorithms conditions, always inside the distortion band, the distortion band consists of gray squares of size 1/3 by 1/3 pixel.

The results of encoding the MPEG-4 test sequence kids (352 by 240 pixels) are illustrated with a rate-distortion chart in Figure 13. Instead of the distortion measure D_{max} in the x-axis the distortion Dn is used in this figure. Dn represents the number of incorrectly coded pixels divided by the number of pixels inside the shape (white pixels). The bit rate is averaged over 100 frames of the kids sequence. Figure 14 illustrates frame 14 of the QCIF kids sequence resolution (176 x 144 pixels) that was encoded with distortion $D_{max} = 1.0$, Figure 15 shows frame 150 of the fish sequence.

IX. Discussion

We presented an efficient method for the lossy encoding of object shapes which are given as 8-connect chain codes. The object shape is approximated by a second order B-spline curve which leads to the smallest bit rate for a given distortion. The B-spline curve is found by a shortest path algorithm since the problem can be described as a single-source directed acyclic graph (DAG). We introduced an encoding scheme that describes a control point vector using 2 bits for the angle information and 2 to 5 bits for the length information. Bit rates of 0.7 bbp and 0.57 bbp for a maximum distortion of 1.0 pixels and 2.0 pixels, respectively, were achieved experimentally.

A more sophisticated encoding scheme for the control point vectors might result in lower bit-rates. Encoding of the angle information with 2 bits seems to be very efficient, but the run-length encoding scheme could be potentially improved. As was also mentioned earlier higher order curves as well as distortion bands of variable width can be incorporated into the proposed algorithm in a straightforward way.

Taking advantage of the temporal similarites of adjacent frames could reduce the overall bitrate in a shape coding scheme. The INTER coding method used for shape coding with vertices [16] uses motion vectors and reduces the INTRA bitrate by about 50 percent. Such an approach could be used with the proposed B-Spline coding scheme, but the INTER coding steps would not be rate-distortion optimal. The extension of our current model from optimal INTRA coding to optimal INTER coding is subject of our current research [25].

Acknowledgment The authors would like to thank Gerry Melnikov, Northwestern University, for his help in the simulations section.

References

[1] H. Musmann, M. Hötter, and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images," *Signal Processing: Image Communication*, vol. 1, pp. 117–138, Oct. 1989.

[2] M. Hötter, "Object oriented analysis/ synthesis coding based on moving two-dimensional objects," *Signal Processing: Image Communication*, vol. 2, pp. 409–428, Dec. 1990.

[3] M. R. Banham and J. C. Brailean, "An overview of the MPEG-4 standard: Enabling digital multimedia compression," in *International conference on advanced science and technology*, (Schaumburg, IL), pp. 2–19, 1997.

[4] T. Sikora, "The MPEG-4 video standard verification model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 19–31, Feb. 1997.

[5] L. P. Kondi, F. W. Meier, G. M. Schuster, and A. K. Katsaggelos, "Joint optimal object shape estimation and encoding," in *Proceedings SPIE Conf. on Visual Comm. and Image Proc.*, pp. 784–795, SPIE, Jan. 1998.

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

[7] J. Saghri and H. Freeman, "Analysis of the precision of generalized chain codes for the representation of planar curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, pp. 533–539, Sept. 1981.

[8] J. Koplowitz, "On the performance of chain codes for quantization of line drawings," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, pp. 180–185, Mar. 1981.

[9] D. Neuhoff and K. Castor, "A rate and distortion analysis of chain codes for line drawings," *IEEE Transactions on Information Theory*, vol. IT-31, pp. 53–68, Jan. 1985.

[10] C. Gu and M. Kunt, "Contour simplification and motion compensated coding," *Signal Processing: Image Commun.*, vol. 7, pp. 279–296, Nov. 1995.

[11] M.-C. Lee, W.-G. Chen, C.-L. B. Lin, C. Gu, T. Markoc, S. I. Zabinsky, and R. Szeliski, "A layered video object coding system using sprite and affine motion model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 130–145, Feb. 1997.

[12] G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression, Optimal Video frame compression and Object boundary encoding*. Kluwer Academic Press, 1997.

[13] G. M. Schuster and A. K. Katsaggelos, "An optimal polygonal boundary encoding scheme in the rate distortion sense," *IEEE Transactions on Image Processing*, pp. 13–26, Jan. 1998.

[14] G. M. Schuster and A. K. Katsaggelos, "An efficient boundary encoding scheme which is optimal in the rate distortion sense," in *Proceedings of the International Conference on Image Processing*, vol. II, (Lausanne, Switzerland), pp. 77–80, Sept. 1996.

[15] K. J. O'Connell, "Object-adaptive vertex-based shape coding method," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 251–255, Feb. 1997.

[16] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann, and G. M. Schuster, "Mpeg-4 and rate-distortion-based shape-coding techniques," *pieee,special issue on* Multimedia Signal Processing, vol. 86, pp. 1126–1154, June 1998.

[17] P. Kauff, B. Makai, S. Rauthenberg, U. Goelz, J. L. Lameillieure, and T. Sikora, "Functional coding of video using a shape-adaptive DCT algorithm and an object-based motion prediction tool box," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 181–195, Feb. 97.

[18] R. Lagendijk and J. Biemond, "Low bit-rate coding for mobile multimedia communications," in *Proceedings of the European Signal Processing Conference*, pp. 435–438, 1996.

[19] R. Talluri, K. Oehler, T. Bannon, J. D. Courtrey, A. Das, and J. Liao, "A robust, scalable, object-based video compression technique for very low bit-rate coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 221–233, Feb. 1997.

[20] G. M. Schuster, G. Melnikov, and A. K. Katsaggelos, "Operationally optimal vertex-based shape coding," *IEEE Signal Processing Magazine*, vol. 15, pp. 91–108, Nov. 1998.

[21] N. Brady, F. Bossen, and N. Murphy, "Contex-based arithmetic encoding of 2d shape sequences," in *Proceedings of the International Conference on Image Processing*, (Santa Barbara, CA), 1997.

[22] Foley, vanDam, Feiner, and Hughes, *Computer Graphics: Principles and Practice*, pp. 478–516. Addison-Wesley, 1990.

[23] "Common test conditions, core experiments on MPEG-4 video shape coding," tech. rep., Video Subgroup/Adhoc Group on Shape Coding, MPEG-4, 1996.

[24] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to algorithms.* McGraw-Hill Book Company, 1991.

[25] G. Melnikov, G. M. Schuster, and A. K. Katsaggelos, "Exploiting temporal correlation in shape coding," in *Asilomar Conf. on Signals, Systems, and Computers*, (Pacific Grove, CA), pp. 1601–1605, Nov. 1998. invited paper.

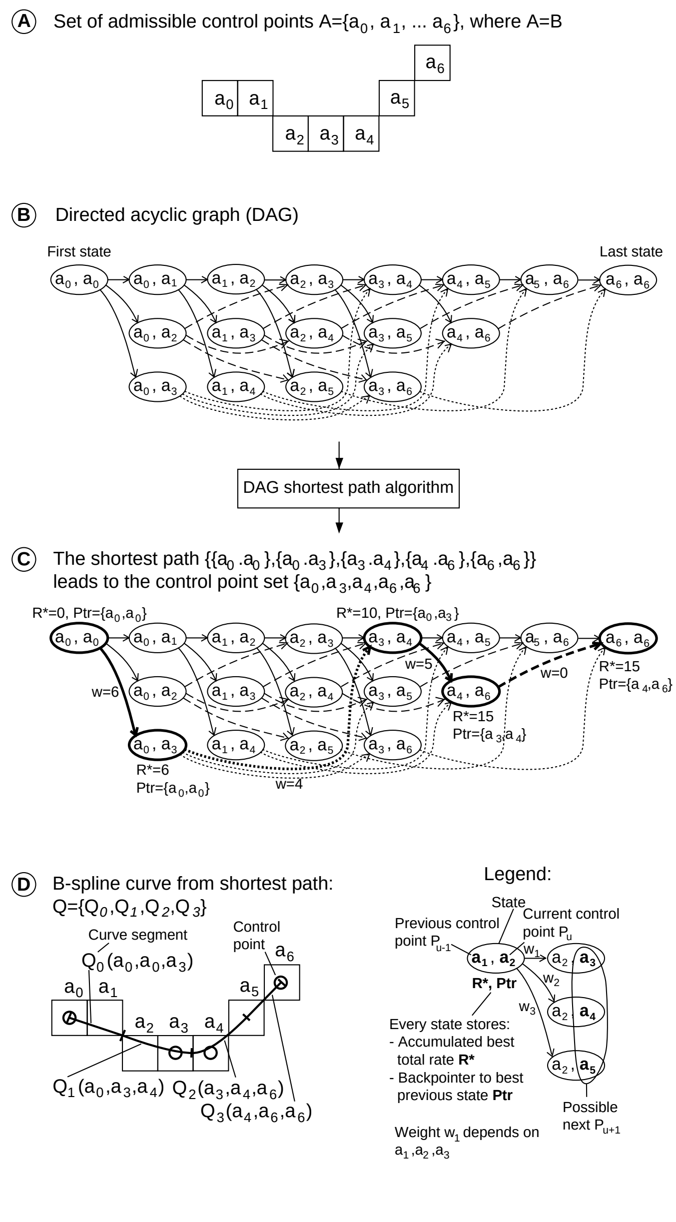

Fig. 6. The boundary in (A) can be modeled by a graph (B). Once a set of admissible control points A is defined (A), a DAG can be defined (B). The shortest path algorithm finds a set of control points of the shortest path (C) from state (a_0, a_0) to state (a_6, a_6). The control point set defines the B-spline curve approximation (D) of the original boundary.

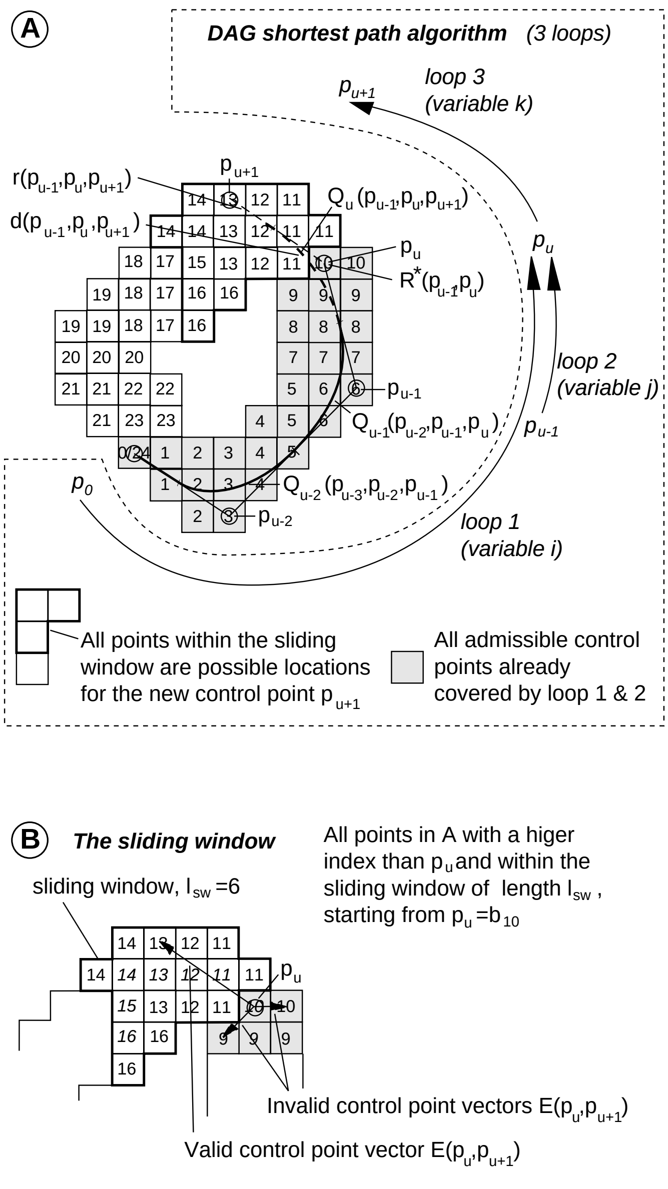

Fig. 7. (A) The DAG shortest path algorithm.(B) Sliding window at its current location.

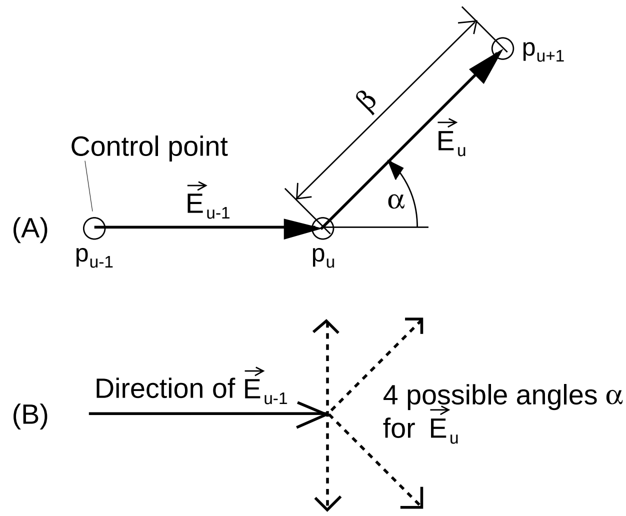

Fig. 8. (A) The control point vector $\vec{E_i}$ is encoded with its run length β and the relative angle α. (B) Possible direction for α, encoded with 2 bits.

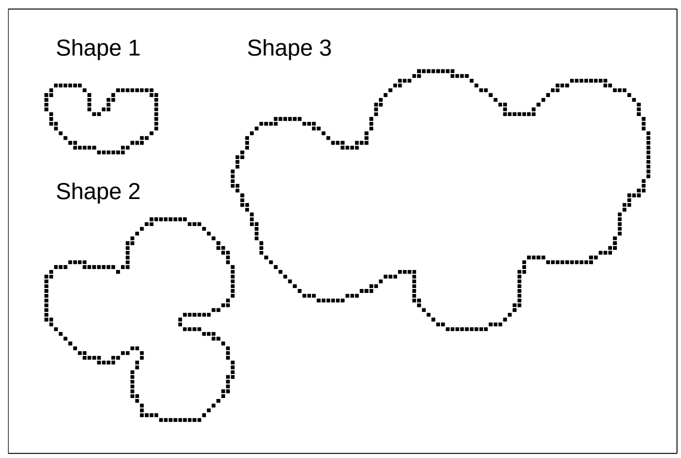

Fig. 9. The three object shapes used for the experiments with 70, 158 and 257 boundary points.

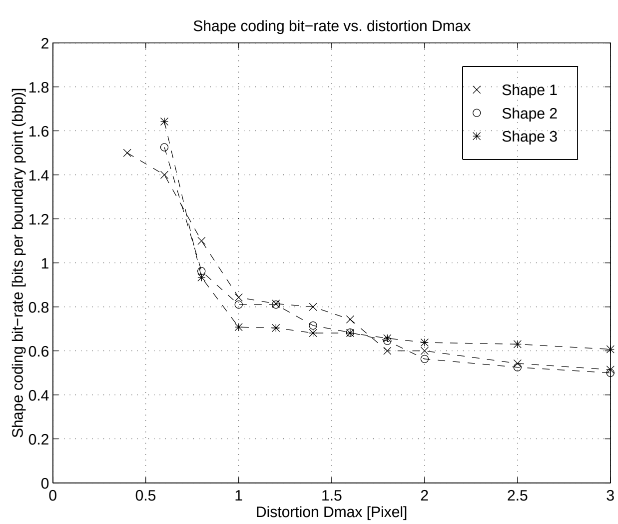

Fig. 10. Results of the Simulation: Shape encoding bit-rate e in bits per boundary point (bbp) vs. the maximal distortion D_{max}.

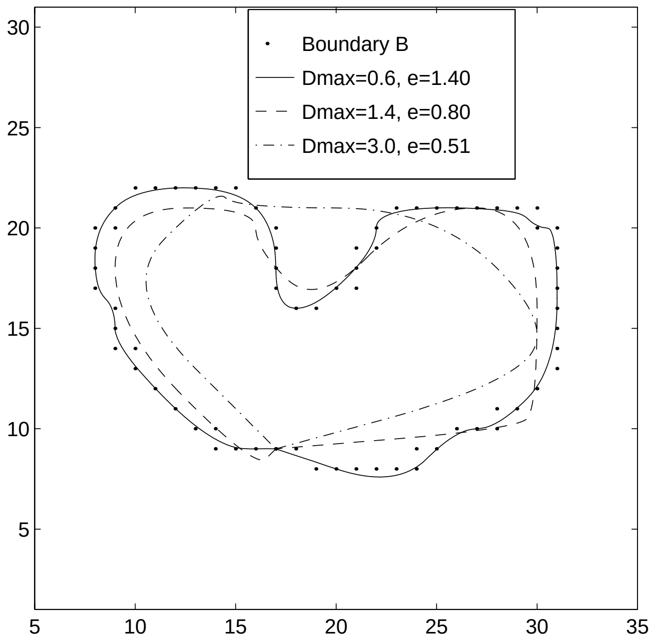

Fig. 11. Object shape approximations of shape 1.

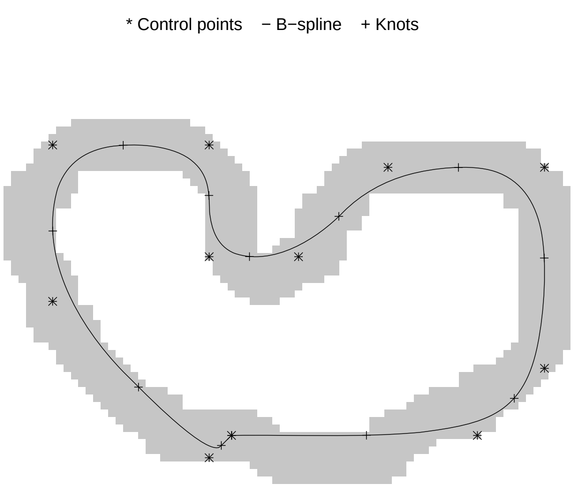

Fig. 12. Distortion band of width $2 \cdot D_{max} = 2 \cdot 1.0$ of Shape 1 results in an encoding efficiency of e=0.84 (Rate=59 bits). The resolution of the distortion band is 1/3 pixel, whereas the resolution of the location of the control points is 1 pixel.

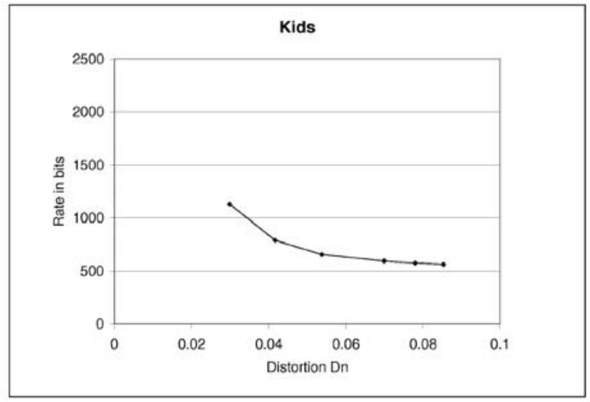

Fig. 13. The average bit rate per frame (average of 100 frames) vs. the distortion Dn of the Kids sequence (352 x 240 pixels).

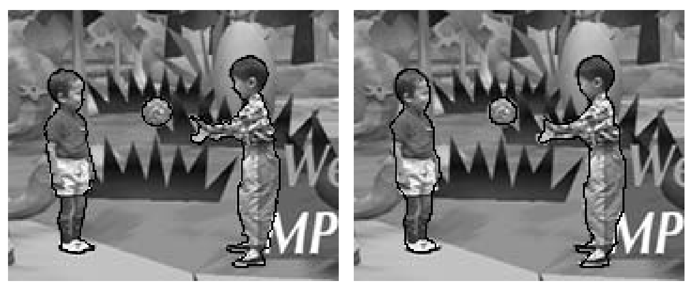

Fig. 14. Frame 14 of the QCIF kids sequence (176 x 144 pixels), the shape information is encoded with $D_{max} = 1.0$ resulting in a rate of 487 bits and a $Dn = 0.071$. Frame on the left: original shape (black lines), frame on the right: encoded shape information.

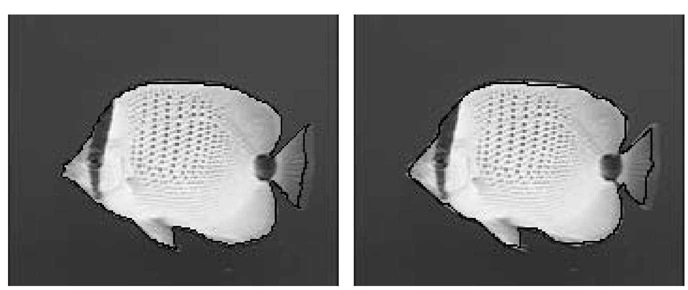

Fig. 15. Frame 150 of the fish sequence (176 x 144 pixels), encoded with $D_{max} = 1.0$ resulting in a rate of 304 bits and a $Dn = 0.024$. Frame on the left: original shape (black lines), frame on the right: encoded shape information.