Joint optimal object shape estimation and encoding

Lisimachos P. Kondi[a], Fabian W. Meier[b], Guido M. Schuster[c] and Aggelos K. Katsaggelos[a]

[a]Northwestern University
Dept. of Electrical and Computer Engineering
Evanston, IL 60208 USA

[b]Silicon Graphics
Advanced Entertainment Systens
Mountain View CA 94043-1389, USA

[c]Advanced Technologies Research Center
3COM, Mt. Prospect, IL 60016, USA

ABSTRACT

A major problem in object oriented video coding and MPEG-4 is the encoding of object boundaries. In our previous work, we presented efficient methods for the lossy encoding of object boundaries which were optimal in the rate distortion sense. In this paper, we extend our work to utilize both the original image and an initial segmentation to obtain the optimal shape representation based on the criteria we impose. The boundary detection and encoding problems are considered simultaneously. If there is low confidence on the location of the boundary, a large approximation error is allowed when encoding the boundary and vide versa. Experimental results demonstrate the effectiveness of the proposed algorithm.

Keywords: Shape coding, rate-distortion, segmentation.

1. INTRODUCTION

In our previous work[1–5] we have introduced efficient object boundary encoding schemes, (polygon and/or B-spline based), which are optimal in the operational rate distortion sense. In other words, for a given maximum approximation error, we found the boundary approximation, which required the fewest bits to encode. Vice versa, for a given bit budget, we found the boundary approximation, which resulted in the minimum maximum approximation error. In both cases we proposed a very fast search for the optimal approximation, which is based on a directed acyclic graph (DAG) formulation of the problem. While these schemes are optimal and very efficient, we noticed that the resulting boundaries are not always visually pleasing, since small, but distinct features in the original boundary might disappear (be "cut off") in its approximation.

The reason for this "cut off" effect is that traditionally the estimation of the object boundary and its encoding are two distinct operations. While the estimation scheme might be aware of the importance of certain features, this is not communicated to the lossy encoding scheme. In contrast to these traditional approaches, in this paper we propose to *jointly* estimate and encode object boundaries. We do this in the framework of the optimal boundary encoding schemes we previously proposed.

In our previous work, we proposed a lossy shape coding scheme using polygons and/or B-spline curves to approximate the object shape. Towards this end, we defined a fixed-width tolerance band around the object boundary and any curve which stayed within the band was considered a valid approximation. As it is common in the field of shape encoding, we considered the given boundary as perfectly estimated. Clearly though, any segmentation might be flawed. In other words, we might be spending bits to *exactly* encode a certain part of the boundary which is *incorrectly* estimated. In general, boundary estimation errors occur in areas where the image gradient is small. In areas where the gradient is large, the detection of the boundary is quite reliable, independently of the specific detection technique used. As a consequence, the magnitude of the gradient can be used as a measure of confidence for the estimation of the boundary. For example, fingers in front of a dark background result in boundary estimates which are probably correct, whereas a dark microphone attached to a dark shirt results in estimates which are probably wrong.

In this paper, we incorporate a measure of the confidence in the boundary estimate into the boundary approximation by introducing a variable-width distortion band into our optimal boundary encoding algorithm. This is achieved by setting the admissible distortion band width around a given initial boundary point inversely proportional to the magnitude of the image gradient at that point. In other words, in high gradient areas, the initial segmentation is probably correct and the approximation is forced to follow the estimated boundary closely, whereas in low gradient areas, the approximation is allowed to deviate from the estimation and hence save bits by doing so. The overall algorithm then estimates the boundary approximation which is accurate where the gradient is high and efficient where the gradient is low. The proposed jointly optimal boundary estimation and encoding scheme trades accuracy for bits in areas where we are not sure about the boundary and bits for accuracy in areas where the boundary is well established. Our initial experiments show that, as expected, the resulting boundary approximations are very accurate around well defined features and very efficient in ill defined boundary areas. In other words, we have now a boundary estimation/encoding scheme which is fast, optimal, efficient and results in visually pleasing boundaries.

The paper is organized as follows: In Section 2, we present the formulation of the problem. In Section 3, we discuss the concept of the tolerance band. In Section 4, we present a review of B-spline curves. In Section 5, we discuss the admissible control point set. In Section 6, we present the shape coding algorithm and in Section 7 we explain the control point encoding scheme. In Section 8 we present the experimental results and in Section 9 we present our conclusions.

2. PROBLEM FORMULATION

The main idea behind the proposed approach is to approximate a given initial boundary by a second order B-spline curve, and to encode the control points that define the B-spline curve. This must be done by also taking into account the gradient information of the original image and defining a tolerance band, as described later. Note that the efficiency of a boundary encoding scheme can be measured either with the absolute rate in bits or with a relative measure of the rate per boundary point. In this paper we are using the latter relative measure $e = R/N_B$ with the unit bits per boundary point, where R is the rate and N_B is the number of boundary points.

The following notation will be used. Let $B = \{b_0, \dots, b_{N_B-1}\}$ denote the connected boundary which is an ordered set, where b_j is the j-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 $P = \{p_0, \dots, p_{N_P+1}\}$ 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 Q_k is defined by three control points p_{k-1}, p_k, p_{k+1}. 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 for control points to a set A, which we will specify in detail in section 5.

We assume that the control points of the curve are encoded differentially, which 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 differential encoding of control point p_{k+1}, given control point p_{k-1} and p_k, by $r(p_{k-1}, p_k, p_{k+1})$. Hence the bit rate $R(p_0, \dots, p_{N_P+1})$ for the entire approximation curve is equal to

$$R(p_0, \dots, p_{N_P+1}) = \sum_{k=0}^{N_P} r(p_{k-1}, p_k, p_{k+1}), \tag{1}$$

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 two control points p_0 and p_1, which are identical. However, since this number depends on the size of the image plane, we neglect the rate of the absolute position when we calculate the encoding efficiency e. As mentioned before, the first two control points p_0 and p_1 are identical and so are the last two control points, p_{N_P} and p_{N_P+1}. This results in the fact that the B-spline approximation starts at p_0 and ends at p_{N_P+1}. Hence p_{N_P+1} does not need to be encoded and therefore $r(p_{N_P-1}, p_{N_P}, p_{N_P+1})$ is always zero. For a closed boundary, the first two control points are identical to the last two, hence the rate $r(p_{N_P-2}, p_{N_P-1}, p_{N_P})$ is also set to zero since the control point p_{N_P} does not need to be encoded. Note that the rate $r(p_{k-1}, p_k, p_{k+1})$ depends on the specific control point encoding scheme, which will be discussed in section 7.

3. THE TOLERANCE BAND

Besides the rate, we also need the curve segment distortion for our proposed curve approximation scheme, which we define as $d(p_{k-1}, p_k, p_{k+1})$. One popular distortion measure for curve approximation is the maximum absolute distance, which has also been employed in.[6–9] Other distortion measures are evaluating the difference area between the original and the approximated boundary shape,[10] as well as, adding all the segment distortion measures.[11]

So far we have only discussed the segment distortion measures, i.e., the measures which judge the approximation of a certain partial boundary by a given curve segment. In general we are interested in a curve distortion measure $D(p_0, \ldots, p_{N_P+1})$ which can be used to determine the quality of approximation of an entire curve. As mentioned above, we are using the maximum absolute distance distortion measure. This can be expressed as follows, using the segment distortion measures defined above,

$$D(p_0, \ldots, p_{N_P+1}) = \max_{k \in [1,\ldots,N_P]} d(p_{k-1}, p_k, p_{k+1}). \tag{2}$$

In our previous work, we defined a "distortion band" with width $2 \cdot D_{max}$ along the boundary B. The B-spline approximation must lie within the distortion band. In this paper, we allow the distortion band to have variable width along the boundary. We call this new band a *tolerance* band. The definition of the tolerance band requires a D_{max} for every boundary point. We denote this as $D_{max}[i], i = 0, \ldots, N_B - 1$. In order to construct the tolerance band, we draw circles from each boundary point b_i with radius $D_{max}[i]$. The tolerance band consists of the set of all point that lie inside the circles.

We can now define the segment distortion measure which we will use by

$$d(p_{k-1}, p_k, p_{k+1}) = \begin{cases} 0: & \text{all points of } Q_k(p_{k-1}, p_k, p_{k+1}) \\ & \text{are inside the tolerance band} \\ \\ \infty: & \text{any point of } Q_k(p_{k-1}, p_k, p_{k+1}) \\ & \text{is outside the tolerance band} \end{cases} \tag{3}$$

This distortion measure takes a curve segment Q_k, given by the three control points p_{k-1}, p_k and p_{k+1}, as input and checks if the curve segment is inside the tolerance band.

As mentioned earlier, we define $D_{max}[i]$ in a way that is inversely proportional to the image gradient. The algorithm proceeds as follows: The gradient is first calculated for the whole image; that is, for for an image $f(x, y)$ is defined as:

$$\nabla f(x, y) = [\partial f/\partial x \ \partial f/\partial y]^T = [f_x \ f_y]^T. \tag{4}$$

The Sobel edge detector masks are used to calculate the gradient.[12] The magnitude of the gradient is then computed:

$$|\nabla f(x, y)| = \sqrt{f_x^2(x, y) + f_y^2(x, y)}. \tag{5}$$

The minimum and maximum of the magnitude of the image gradient for the whole image are then computed. Let us denote these as *gradmin* and *gradmax*, respectively. Let us also denote the desired minimum and maximum values of $D_{max}[i]$ as T_{min} and T_{max}, respectively. Then, a linear mapping is performed between the gradient value of each boundary point and the width of the distortion band. If the magnitude of the gradient at the boundary point b_i is grad[i], then the width of the tolerance band at this point is given by:

$$D_{max}[i] = T_{min} + \lambda(grad[i] - gradmax) \tag{6}$$

where

$$\lambda = \frac{T_{max} - Tmin}{gradmin - gradmax}. \tag{7}$$

In practice, we need to define a threshold for the gradient magnitude. The boundary points whose gradient magnitude exceeds the threshold should have the minimum possible $D_{max}[i]$. Clearly, *gradmax* is equal to the threshold in that case.

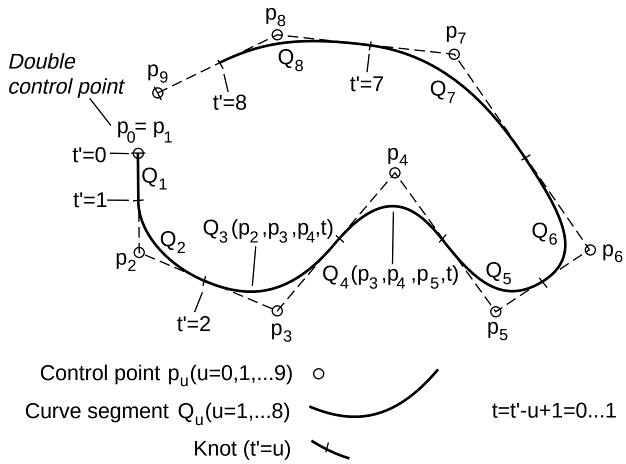

Figure 1. A second degree B-spline curve with 8 curve segments Q_k.

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

$$\min_{p_0,\ldots,p_{N_P+1}} R(p_0,\ldots,p_{N_P+1}), \tag{8}$$

provided that the approximation lies within the tolerance band.

4. REVIEW OF B-SPLINE CURVES

A B-spline is a specific curve type from the family of parametric curves. A parametric curve of degree n consists of at least one curve segment where each curve segment is defined by $(n+1)$ control points. The control points are located around the curve segment and define its shape. A segment Q_k is a curve in a two dimensional plane. The following is the definition of a second degree curve segment,

$$Q_k(p_{k-1},p_k,p_{k+1},t) = T\cdot M\cdot P = \begin{bmatrix} t^2 & t & 1 \end{bmatrix}\cdot\begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix}\cdot\begin{bmatrix} p_{k-1,x} & p_{k-1,y} \\ p_{k,x} & p_{k,y} \\ p_{k+1,x} & p_{k+1,y} \end{bmatrix}. \tag{9}$$

The beginning and the end of a curve segment are called knots. They can be found by setting $t=0$ and $t=1$. 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_k in a two dimensional plane. Every point of the curve segment can be calculated with Eq.(9) by letting t go from 0 to 1. Every curve segment can be calculated independently in order to calculate the entire curve Q which is of the following form,

$$Q(t') = \textstyle\sum_{k=1}^{N_P} Q_k(p_{k-1},p_k,p_{k+1},t'-k+1), \text{with } 0\le t'\le N_P+1. \tag{10}$$

where $Q_k(p_{k-1},p_k,p_{k+1},t)$ is zero when $t\notin[0,1]$. 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.[13] Figure 1 shows such a second order B-spline curve. We chose a B-spline with the lowest possible order to keep the complexity of the curve, and the proposed algorithm, minimal. Note that a first degree B-spline is a polygon.

5. 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

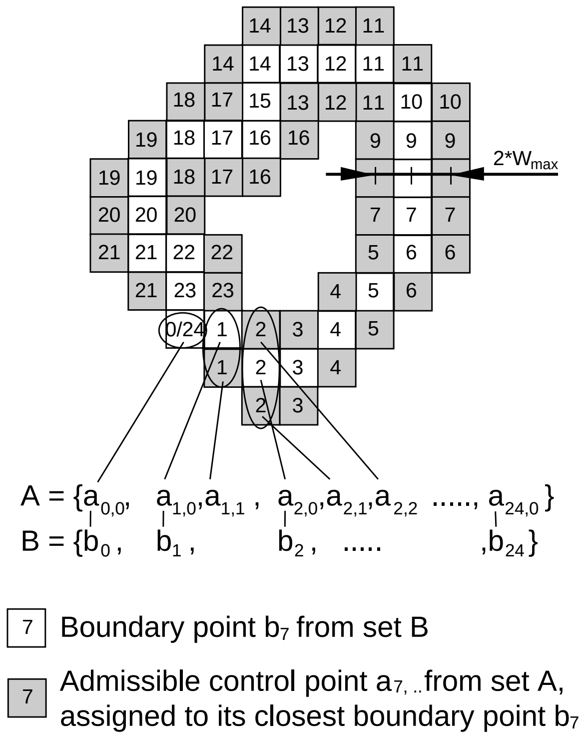

Figure 2. The admissible control point set A is a band of width $2 \cdot W_{max}$ along B. Note that $b_0/b_{24} = a_{0,0}/a_{24,0}$ have no additional points assigned.

with width $2 \cdot W_{max}$ (see Figure 2). Set 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. Details of the assigning algorithm can be found in.[1,2] Every admissible control point a_{i,i_b} has two indexes: 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 numerates 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 $W_{max} = 0$, since then $A = B$. Clearly, not every boundary point has the same number of admissible control points. A good value for W_{max} is one pixel, which results in a band of three pixels thickness. We further define that the first and the last boundary points have no additional admissible control points assigned.

6. THE SHAPE CODING ALGORITHM

The goal of the proposed algorithm is to find the B-spline curve whose control points can be encoded with the smallest number of bits under two conditions: 1) the distortion of the curve is smaller than or equal to the maximum distortion $D_{max}[i]$), and 2) the control points must be selected from the admissible control point set A. The key observation for deriving an efficient search is the fact that, given two control points p_{k-1} and p_k of a B-spline curve and the rate which is required to code the curve up to and including these two control points, $R_k(p_{k-1}, p_k)$, the selection of the next control point p_{k+1} is independent of the selection of the previous control points $p_0, \ldots, p_{k-2}$. This is true since the total rate R can be expressed recursively as a function of the control point rates $r(p_{k-1}, p_k, p_{k+1})$, that is,

$$R_{k+1}(p_k, p_{k+1}) = R_k(p_{k-1}, p_k) + r(p_{k-1}, p_k, p_{k+1}). \tag{11}$$

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

$$w(p_{k-1}, p_k, p_{k+1}) = r(p_{k-1}, p_k, p_{k+1}) + d(p_{k-1}, p_k, p_{k+1}). \tag{12}$$

Note that w is equal to the rate for curve segments which satisfy the distortion constraint, but infinite for those which do not. Hence by replacing r in Eq.(11) by w, the rate $R_{k+1}(p_k, p_{k+1})$ is infinite if a curve segment is used which violates the distortion constraint and equal to the required rate otherwise. The recursion of Eq.(11) needs to be initialized by setting $R_0(p_{-1}, p_0)$ to zero. Clearly $R_{N_P+1}(p_{N_P}, p_{N_P+1}) = R(p_0, \ldots, p_{N_P+1})$, is the rate for the entire curve.

Because of the recursive relationship in Eq.(11), the problem in Eq.(8) can be formulated as a shortest path problem in a weighted directed graph (see Fig. 3 (A)). 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 ($\vec{E} = a_{i,i_b} - a_{k,k_b} \in A^2; \forall i \neq k\}$ (see Figure 3 (B)). 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 length of the path is defined as follows,

$$\sum_{k=1}^{K-1} w(p_{k-1}, p_k, p_{k+1}), \tag{13}$$

Again, note the above definition of the weight function leads to a length of infinity for every path which includes a curve segment which has a part that lies outside the tolerance band. Therefore a shortest path algorithm will not select these paths.

The classical algorithm for solving such a single-source shortest-path problem, where all the weights are non-negative, is Dijkstra's algorithm,[14] which results in 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 $i > k$. Hence the vector set is redefined in the following way, ($\vec{E} = a_{i,i_b} - a_{k,k_b} \in A^2; \forall i < k$). This restriction results in the selected curve approximation having to follow the original boundary without rapid direction changes (see Fig. 3 (B)), and more important, the resulting graph is a weighted directed acyclic graph (DAG). For a DAG, the DAG-shortest-path algorithm[14] finds a single-source shortest-path and is even faster than Dijkstra's algorithm.

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. $p_u = a_{i,i_b}$ is the last and $p_{u-1} = a_{j,j_b}$ is the next to last control point of the last curve segment. 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 two control points, which are by definition equal to the last boundary point.

The sliding window

With this formulation of the boundary approximation problem, the solution of the DAG shortest path algorithm may result in a trivial solution (see Fig. 4). 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 contol the speed of the algorithm. The sliding window indicates the admissible selections for the next control point p_u (see Fig. 4). Thus, trivial solutions are eliminated and the computational complexity of the algorithm is decreased.

7. 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 a combination of a modified 8-connect chain code and a run-length encoding scheme.[1] 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 logarithmic code for encoding the runs β, which is displayed in Table 1. Note that the longest possible run is 15.

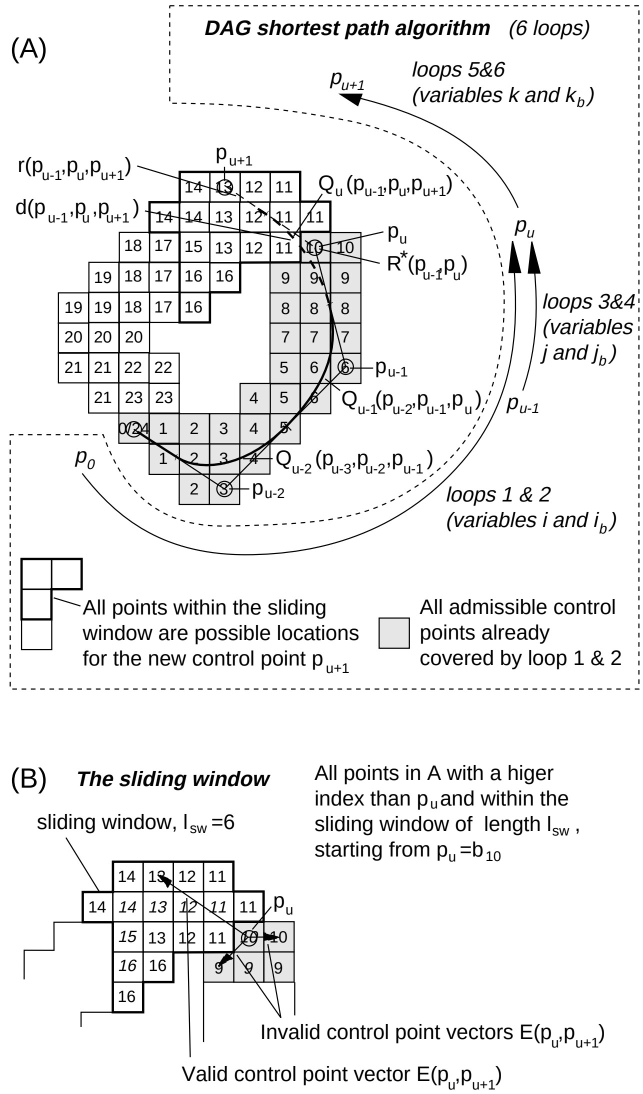

Figure 3. (A)DAG shortest path algorithm. (B) Sliding window at its current location. Valid and invalid control point vectors $\vec{E}$ in the DAG.

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 for the vector. This scheme together with Huffman coding for the run length of the vector was used in.[9] 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 5 (A)). We propose to use only 2 bits for α for the four most probable directions. These four relative directions of α are $\{+45^o, +90^o, -45^o, -90^o\}$, where 0^o is the direction of the previous vector (see Figure 5 (B)). With this scheme we restrict the set of codeable vectors slightly compared to a scheme with 8 codeable absolute directions, but on the other hand, we achieve lower encoding rates for the control point vectors. The curve segment rate function $r(p_{u-1}, p_u, p_{u+1})$ must consider the case when

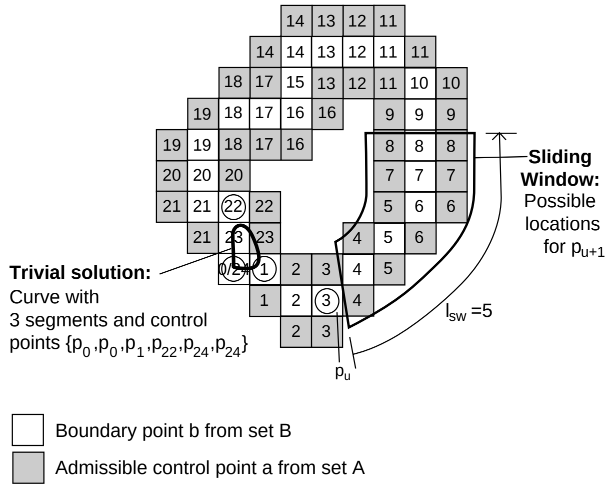

Figure 4. The sliding window restricts the selection of control point p_{u+1} to all the admissible control points within the sliding window. The introduction of a sliding window prevents trivial solutions.

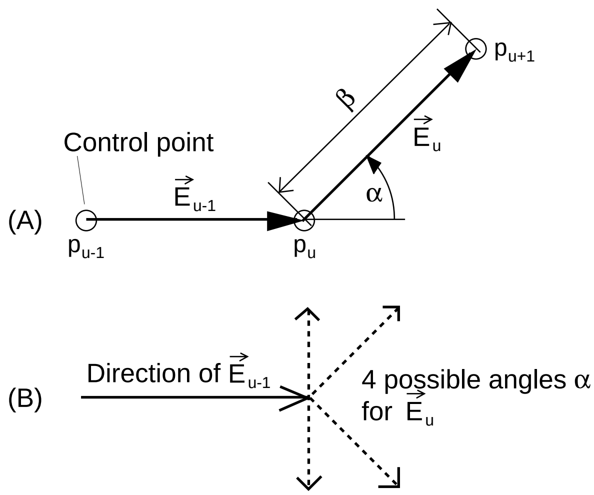

Figure 5. (A) The control point vector $\vec{E}_i$ is encoded with its run length β and the relative angle α. (B) Possible direction for α, encoded with 2 bits.

a vector cannot be encoded. If this happens, the rate is considered infinite. This can be expressed as follows,

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

$$\text{rate of } \alpha : 2 \text{ bits, rate of } \beta : 2 \text{ to } 5 \text{ bits}$$

Experiments showed that using an encoding scheme with four relative angles with 2 bits results in boundary approximations with lower rates than when we used the scheme with 8 absolute angles.

run	CW	rate	run	CW	rate
1	00	2 bit	8	11000	5 bit
2	010	3 bit	9	11001	5 bit
3	011	3 bit	10	11010	5 bit
4	1000	4 bit	11	11011	5 bit
5	1001	4 bit	12	11100	5 bit
6	1010	4 bit	13	11101	5 bit
7	1011	4 bit	14	11110	5 bit
			15	11111	5 bit

Table 1. Code word (CW) assignment for the run length β of control point vector $\vec{E}$.

Figure 6. Frame 0 of the "Kids" sequence

8. EXPERIMENTAL RESULTS

For our experimental results, frames from the MPEG-4 "Kids" sequence were used. Figure 6 shows the original frame 0 of the sequence and figure 7 its segmentation. It should be noted that, for this paper, we used the segmentation provided with the test sequence.

The parameters used for the simulation were $T_{min} = 0.8$ and $T_{max} = 3.0$. The magnitude of the gradient was thresholded with a value of 255. The tolerance band was computed using 1/3 pixel accuracy. A sliding window of size 15 was used. The magnitude of the image gradient is shown in figure 8.

Figure 9 shows the result of the proposed boundary encoding algorithm. Figure 10 shows the tolerance band and

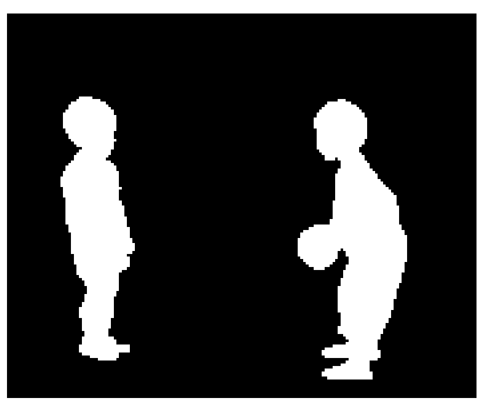

Figure 7. Segmentation of frame 0 of the "Kids" sequence

Algorithm	Kid 1 bits	Kid 2 bits	Total bits	e (bits/boundary point)
Variable width tolerance band	207	249	456	0.77
Fixed width tolerance band (1.5 pixels)	175	240	415	0.70
Fixed width tolerance band (0.8 pixels)	267	318	585	0.98

Table 2. Performance of the different algorithms

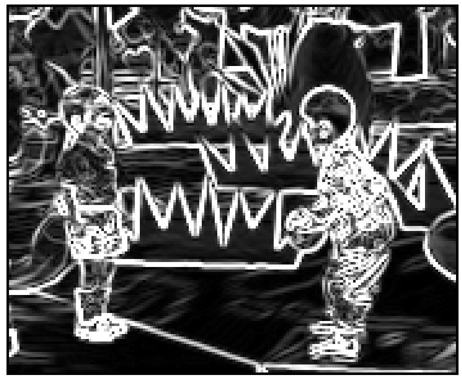

Figure 8. Magnitude of the image gradient (frame 0 of the "Kids" sequence)

the boundary approximation for the kid on the left. Figure 11 shows the result using a tolerance band with a fixed width $D_{max} = 1.5$. Figure 12 shows the result using a tolerance band with a fixed width $D_{max} = 0.8$. The original segmentation has 596 boundary points. We can see from table 8 that the variable width tolerance band algorithm requires much fewer bits than the algorithm with $D_{max} = 0.8$ but more bits than the algorithm with $D_{max} = 1.5$. The variable width tolerance band algorithm preserves some of the details of the segmentation that the algorithm with $D_{max} = 1.5$ does not, such as, the feet of the kid on the left, for comparable bit rates. As we can see in figure 8, the gradient in the feet area is large, thus, the corresponding tolerance band width is equal to $T_{min} = 0.8$. In the third experiment, using $D_{max} = 0.8$, we have better approximation throughout the boundary at the expense of a much higher bit rate. Thus, the proposed algorithm saves bits by imposing a narrow tolerance band only in areas where the segmentation ambiguous.

9. CONCLUSIONS

In this paper, we presented a boundary encoding scheme which is optimal in the rate distortion sense and also takes into account information from the original intensity image rather than using only the segmentation information. The

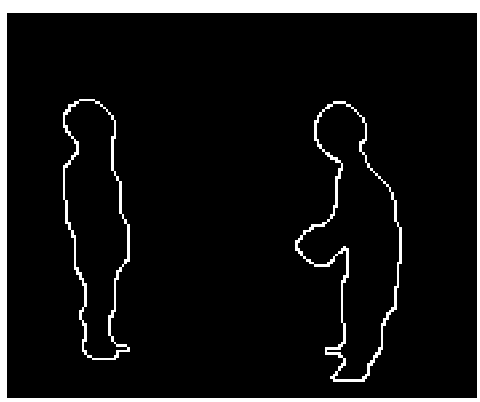

Figure 9. Result of the proposed algorithm

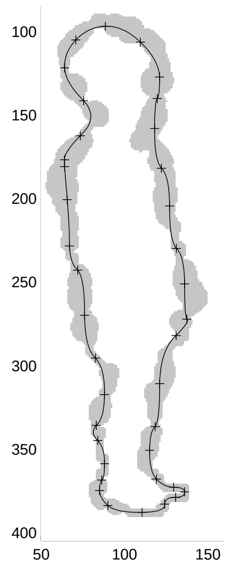

Figure 10. The tolerance band and the boundary approximation for the kid on the left

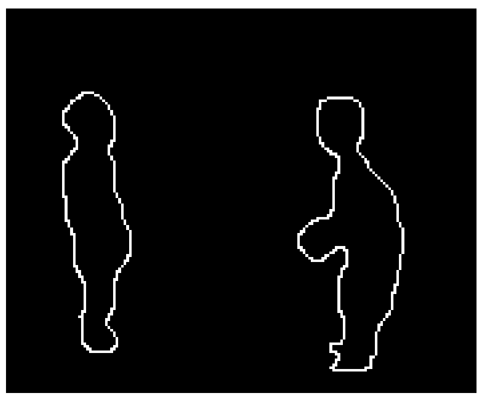

Figure 11. Result of the fixed width tolerance band algorithm ($D_{max} = 1.5$)

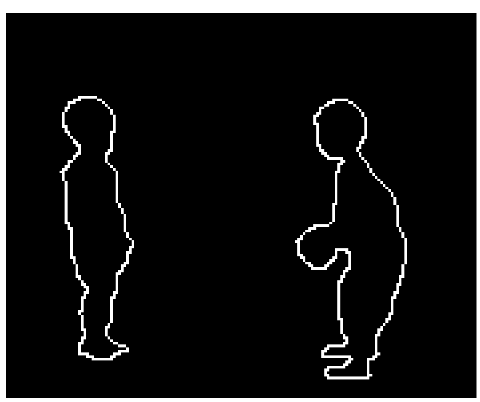

Figure 12. Result of the fixed width tolerance band algorithm ($D_{max} = 0.8$)

success of the basic algorithm, as described in our previous work, is that we formulate the problem in a way that it can be solved optimally using a fast method from graph theory. Here, we extend this algorithm to "reallocate" the bits used in the encoding along the boundary according to an importance measure that we define. There is no point in using many bits to encode parts of the boundary where there is low confidence. These bits can be used in other parts of the boundary where there is a higher level of confidence. The importance measure used here is proportional to the image gradient.

Future work in this area will fully integrate the boundary detection and encoding. For the scope of this paper, these two procedures are linked using the image gradient. The segmentation problem was not addressed directly, but the segmentation provided with the test sequence was used. Although this segmentation was not obtained using a gradient-based method, the gradient is still a useful importance measure. However, the algorithm is general and any other importance measure can be used.

REFERENCES

1. 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, pp. 77–80, (Lausanne, Switzerland), Sept. 1996.
2. G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression, Optimal Video frame compression and Object boundary encoding*, Kluwer Academic Press, 1997.
3. F. W. Meier, G. M. Schuster, and A. K. Katsaggelos, "An efficient boundary encoding scheme using B-spline curves which is optimal in the rate-distortion sense," in *2nd Erlangen Symposium, Advances in Digital Image Communication*, pp. 75–84, (Erlangen, Germany), Apr. 1997.
4. F. W. Meier, G. M. Schuster, and A. K. Katsaggelos, "An efficient boundary encoding scheme using B-spline curves which is optimal in the rate-distortion sense," in *Proceedings of the International Conference on Image Processing*, (Santa Barbara, CA), 1997.
5. G. M. Schuster and A. K. Katsaggelos, "An optimal boundary encoding scheme in the rate distortion sense," *IEEE Transactions on Image Processing* . To appear January 1998.
6. J. Koplowitz, "On the performance of chain codes for quantization of line drawings," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-3**, pp. 180–185, Mar. 1981.
7. D. Neuhoff and K. Castor, "A rate and distortion analysis of chain codes for line drawings," *IEEE Transactions on Information Theory* **IT-31**, pp. 53–68, Jan. 1985.
8. M. Hötter, "Object oriented analysis/ synthesis coding based on moving two-dimensional objects," *Signal Processing: Image Communication* **2**, pp. 409–428, Dec. 1990.
9. G. M. Schuster and A. K. Katsaggelos, "An optimal lossy segmentation encoding scheme," in *Proceedings SPIE Conf. on Visual Comm. and Image Proc.*, pp. 1050–1061, Mar. 1996.
10. "Common test conditions, core experiments on MPEG-4 video shape coding," tech. rep., Video Subgroup/Adhoc Group on Shape Coding, MPEG-4, 1996.
11. G. M. Schuster and A. K. Katsaggelos, "An optimal segmentation encoding scheme in the rate-distortion sense," in *Proceedings of the International Symposium on Circuits and Systems*, vol. 2, pp. 640–643, (Atlanta, GA), May 1996.
12. A. K. Jain, *Fundamentals of digital image processing*, Prentice-Hall, 1989.
13. Foley, vanDam, Feiner, and Hughes, *Computer Graphics: Principles and Practice*, pp. 478–516. Addison-Wesley, 1990.
14. T. Cormen, C. Leiserson, and R. Rivest, *Introduction to algorithms*, McGraw-Hill Book Company, 1991.