AN EFFICIENT BOUNDARY ENCODING SCHEME WHICH IS OPTIMAL IN THE RATE DISTORTION SENSE

*Guido M. Schuster and †Aggelos K. Katsaggelos

Northwestern University, Department of Electrical and Computer Engineering,
Evanston, IL 60208-3118, Email: *gmschust@ece.nwu.edu, †aggk@ece.nwu.edu

ABSTRACT

In this paper, we present a fast and optimal method for the lossy encoding of object boundaries which are given as 8-connect chain codes. We approximate the boundary by a polygon and consider the problem of finding the polygon which can be encoded with the smallest number of bits for a given maximum distortion. The presented scheme is an extension of the approaches introduced in [1, 2], in that the admissible polygon vertices belong to a band around the original boundary and a new vertex encoding scheme is proposed.

1. INTRODUCTION

A major problem in object oriented video coding [3] is the efficient encoding of object boundaries. There are two common approaches for encoding the segmentation information. A lossy approach, which is based on a spline approximation of the boundary [4], and a lossless approach which is based on chain codes [5]. The proposed boundary encoding scheme is a lossy scheme which extends the schemes we presented in [1, 2]. It can be considered a combination of the spline and the chain code approaches, since the boundary is approximated by a polygon and its vertices are encoded relatively to each other. The main feature of the proposed approach is that it is very efficient and the vertices of the polygon are selected optimally in the rate distortion sense. This means that the bit rate required to encode the approximation polygon is minimized for a given maximum distortion, or vice versa, the maximum distortion is minimized for a given bit rate.

2. PROBLEM FORMULATION

Since we assume that the original boundary is represented with pixel accuracy, it can be losslessly encoded by an 8-connect chain-code. We propose to approximate the boundary with a low order polygon which can be encoded efficiently and also reduces some of the noise along the object boundary which is a result of the segmentation algorithm. 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_j is the j-th point of B and N_B is the total number of points in B. Let $P = \{p_0, \ldots, p_{N_P-1}\}$ denote the polygon used to approximate B which is an ordered set, where p_k is the k-th vertex of P and N_P is the total number of vertices in P.

We assume that the vertices of the polygon are encoded differentially. In other words, only the difference between two consecutive vertices is encoded, where we denote the required bit rate as $r(p_{k-1}, p_k)$. Therefore the rate to encode the entire polygon, $R(p_0, \ldots, p_{N_P-1})$, can be expressed as the sum of the edge rates $r(p_{k-1}, p_k)$,

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

where $r(p_{-1}, p_0)$ is set equal to the number of bits needed to encode the absolute position of the first vertex. In the case of a closed boundary, i.e., the first vertex is identical to the last one, the rate $r(p_{N_P-2}, p_{N_P-1})$ is set to zero since the last vertex does not need to be encoded.

In general the polygon which is used to approximate the boundary should be permitted to place its vertices anywhere in the plane. In section 4 we come back to this point. In [1], we restricted the location of the vertices to belong to the original boundary ($p_k \in B$). This restriction results in the following fact, which we employed to derive a low complexity optimization algorithm.

The straight line connecting two consecutive vertices p_{k-1} and p_k is an approximation to the partial boundary $\{b_j = p_{k-1}, b_{j+1}, \ldots, b_{j+l} = p_k\}$ which contains $l+1$ boundary points. Therefore, we can measure the quality of this approximation by an edge distortion measure which we denote by $d(p_{k-1}, p_k)$. In this paper we select the maximum absolute distance between a boundary point and the approximation as the distortion measure.

The polygon distortion $D(p_0, \ldots, p_{N_P-1})$ can be expressed as the maximum of the edge distortions $d(p_{k-1}, p_k)$, that is,

$$D(p_0, \ldots, p_{N_P-1}) = \max_{k \in [0, \ldots, N_P-1]} d(p_{k-1}, p_k), \qquad (2)$$

where $d(p_{-1}, p_0)$ is defined to be equal to zero.

3. REVIEW

In [1] we have derived an efficient algorithm for selecting the optimal polygonal approximation for a given boundary in the sense that the encoding of this polygon requires the fewest number of bits for a given maximum distortion. We call this the minimum rate case which can be stated as follows,

$$\min R(p_0, \ldots, p_{N_P-1}), \quad \text{s.t.:} \quad D(p_0, \ldots, p_{N_P-1}) \leq D_{max}, \qquad (3)$$

where D_{max} is the maximum permissible distortion.

The key observation for deriving an efficient search is based on the fact that given a certain vertex of a polygon, the rate which is required to code the polygon up to and including this vertex, and the distortion of this polygonal approximation up to and including this vertex, the selection of the next vertex is independent of the selection of the previous vertices. The algorithm which we proposed in [1] is based on a shortest path algorithm through a directed acyclic graph (DAG) and its time complexity is quadratic in the number of admissible vertices.

In [1] we also introduced an algorithm which finds the optimal solution to the minimum distortion case, which can be formulated as follows,

$$\min D(p_0, \ldots, p_{N_P-1}), \quad \text{s.t.:} \quad R(p_0, \ldots, p_{N_P-1}) \leq R_{max}, \tag{4}$$

where R_{max} is the maximum bit rate available. We proposed an iterative solution to this problem which is based on the fact that we can solve the minimum rate case optimally.

4. EXTENDED ADMISSIBLE VERTEX SET

One of the main contributions of this paper is that we relax the restriction that the admissible vertices for the polygon belong to the original boundary. The main drawback of this restriction is that one can easily construct an example where for a given maximum polygon distortion, the polygon with the smallest bit rate uses vertices which do not fall onto boundary points. The problem with using vertices which do not belong to the boundary is that for a given polygon edge no direct correspondence exists between the edge and a subset of boundary points. Hence the polygon distortion cannot be formulated as the maximum of the edge distortions since the edge distortions are not defined.

4.1. Definition of the admissible vertex set

From a theoretical point of view, the set of admissible vertices should contain all the pixels in the image plane. On the other hand, the DAG shortest path algorithm has a time complexity which is quadratic in the number of admissible vertices and hence we would like to keep that number as small as possible, without sacrificing coding efficiency. In the proposed approach, the set of all admissible vertices is defined as all the pixels which are within a given maximum distance DM from a boundary point (see Fig. 1). Hence the set of admissible vertices forms a "band" of width $2 * DM$ around the original boundary.

4.2. Definition of the edge distortion measure

The basic idea behind adapting the DAG shortest path algorithm for the case where off-boundary vertices are permissible, is to re-define the edge distortion measure. Since the DAG shortest path algorithm requires edge distortions, we would like to define edge distortions in such a way that the maximum of these edge distortions is smaller than or equal to the polygon distortion. Hence the optimal polygon found using these edge distortions will always satisfy the maximum polygon distortion and we can still employ the fast graph algorithm for the optimization procedure.

To achieve this, every admissible vertex (for example p_b in Fig. 1) is associated with the closest boundary point (which is b_b in Fig. 1). The distortion of two admissible vertices (the edge distortion $d(p_a, p_b)$) is then defined as the maximum distance between the polygon edge ((p_a, p_b) in Fig. 1) and the partial boundary ($\{b_a = p_a, \ldots, b_b = p_b\}$ in Fig. 1). This definition of edge distortion leads to the required independence between edge distortions and hence the fast DAG algorithm can be used to find the optimal polygon, where optimality is defined with respect to these edge distortions.

4.3. Ordering of the admissible vertices

As discussed in [1], the set of all admissible vertices should be an ordered set, otherwise Dijkstra's algorithm, which is slower than the DAG algorithm, needs to be employed. The ordering of the admissible vertices is straightforward when they belong to the boundary. We just use the order of the boundary set and impose it on the vertex set. Since the vertex set is now bigger than the boundary set, this ordering is not obvious anymore. We propose to order the admissible vertices by assigning them to the nearest boundary point and then use the order of the boundary points to impose an order on the set of admissible vertices. The closest boundary point of a given admissible vertex is not necessarily unique. We propose the following procedure to define the set of admissible vertices V and to assign the vertices to boundary points.

```
V=B; i=1;
while ( d(v_0, v_i) ≤ DM )
    for j = 1, ..., N_B − 2;
        if ( {b_j + v_i} ∉ V )
            V ← V ∪ {b_j + v_i};
            assign vertex b_j+v_i to boundary point b_j;
    i=i+1;
```

The $v_i s$ in the above algorithm are offset vectors, which are displayed in Fig. 2. In Fig. 2, the vector v_j starts at 0 and ends at j. Note that the distance $d(v_0, v_j)$ is a non-decreasing function of j. The basic idea behind the proposed algorithm is to draw spirals around the boundary points, where the spirals are drawn in discrete increments. For each increment all boundary points are visited. For each boundary point an admissible vertex is assigned to it, if the vertex does not already belong to the admissible vertex set. Note that no additional admissible vertices are associated with the first or last boundary point. This results in the fact that b_0 is selected as the first and b_{N_B-1} as the last vertex, or in other words, the polygonal approximation starts and ends at the same points as the original boundary.

The proposed algorithm results in a set of admissible vertices, which forms a band around the original boundary of width $2 * DM$, and an ordering of the admissible vertices which enables us to use the fast DAG shortest path algorithm. The imposed ordering is such that we only admit

directed edges (from,to) where the boundary point associated with the "from" vertex is of lower order than the boundary point associated with the "to" vertex.

Note that using the above definition of the edge distortion and the admissible set of vertices, the algorithms introduced in [1], where the admissible vertices belong to the original boundary, can still be applied. Clearly since the time complexity of all proposed algorithms is quadratic in the number of admissible vertices, we should select the size of the band DM to be as small as possible.

5. VERTEX ENCODING SCHEME

In this section we present the second major contribution of this paper, a vertex encoding scheme which can be considered a combination of an 8-connect chain code and a run-length encoding scheme. The chain code and the run-length encoding can be combined by representing the increment between two vertices by an angle α and a run β, which form the symbol (α,β). Therefore for a run of 1, the 8 closest neighbors of a given point P are:

$$\begin{array}{ccc} (3,1) & (2,1) & (1,1) \\ (4,1) & P & (0,1) \\ (-3,1) & (-2,1) & (-1,1). \end{array} \tag{5}$$

As an example, $(3,4)$ represents a straight line of 4 increments in the $3 * \pi/4$ direction. 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 (βs), which is displayed in Table 1. Note that the longest possible run is 15.

The formulation of the optimization problem requires that the edge rate from vertex p_i to vertex p_j is only a function of these two vertices. Hence every direction of departure from vertex p_i must have the same probability, which is 1/8. Clearly it takes 3 bits to encode 8 equally probable directions. On the other hand, in natural boundaries, the arrival direction of the edge (p_k, p_i) is highly correlated with the departure direction of the edge (p_i, p_j). This implies that the arrival direction should be used to predict the departure direction.

The main problem of using such a prediction is that the DAG shortest path algorithm cannot directly be applied to find the optimal polygon, since this edge direction prediction results in a higher order dependency. This problem can still be solved but it is computationally more expensive. Therefore we propose the following approximation to this edge prediction concept, which uses the same DAG shortest path algorithm we have been employing throughout this paper.

We propose to use only 2 bits for 4 equally probable directions. Clearly this is in conflict with the fact that there are 8 directions. The underling assumption is that 4 out of these 8 directions will never be used by the optimal solution and hence, we do not need a code word for them, even though for the optimization, they also use 2 bits.

Our experiments have shown, that the "jack-knife" directions (see Fig. 3a) are very unlikely to occur in an optimal solution. Therefore, even though we assume that they cost 2 bits to encode, we do not have a code word for them. If the optimal solution uses a "jack-knife" direction, then this direction can be obtained by adding an artificial turn, which will require a few additional bits and the distortion might change slightly (see Fig. 3a). In that case, the solution is not optimal, but still very close to optimal. In almost all cases however, the optimal solution will not include a "jack-knife" turn and therefore it will truly be an optimal solution.

Furthermore, an optimal solution will not append two edges which are co-linear since the compounded edge can be encoded more efficiently (see Fig. 3b) . Hence the two co-linear directions will never occur in an optimal polygon and therefore we do not need a code word for them. This does not hold if the compounded edge is longer than the maximum run (15 in the logarithmic code). Then it might be optimal to append a co-linear edge. This case can be recognized by the encoder and decoder and treated by sending 3 bits for the next direction which allows for the next edge to be co-linear. Therefore, in most cases, the two co-linear departure directions are not needed which leaves us with 4 equally probable departure directions which in turn can be encoded by only 2 bits instead of the original 3 bits (see Fig. 3c). Note that for the first vertex, all 8 departure directions are admissible since no arrival direction exists for this vertex. Therefore, 3 bits are sent for the first vertex.

6. EXPERIMENTAL RESULTS

We present the minimum rate approach where we set D_{max} to one pixel and optimally encode an object of the "Miss America" sequence. For the band of admissible vertices we select $DM = 1$ pixel, which is the same as the maximum distortion. Our experiments have shown that this is a good tradeoff between speed and coding efficiency. In Fig. 4 the original boundary ((*), 267 bits), the approximation proposed in [1] ((:), 137 bits), and the proposed approximation ((-), 72 bits) are displayed. Note that the proposed approximation uses only 27% of the bits required to encode the original boundary. Clearly this demonstrates the efficiency of the proposed approach.

7. REFERENCES

[1] G. M. Schuster and A. K. Katsaggelos, "An optimal lossy segmentation scheme," in *Proceedings of the Conference on Visual Communications and Image Processing*, pp. 1050–1061, SPIE, Mar. 1996.

[2] 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, (Atlanta, GA), pp. 640–643, May 1996.

[3] 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.

[4] A. K. Jain, *Fundamentals of digital image processing.* Prentice-Hall, 1989.

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

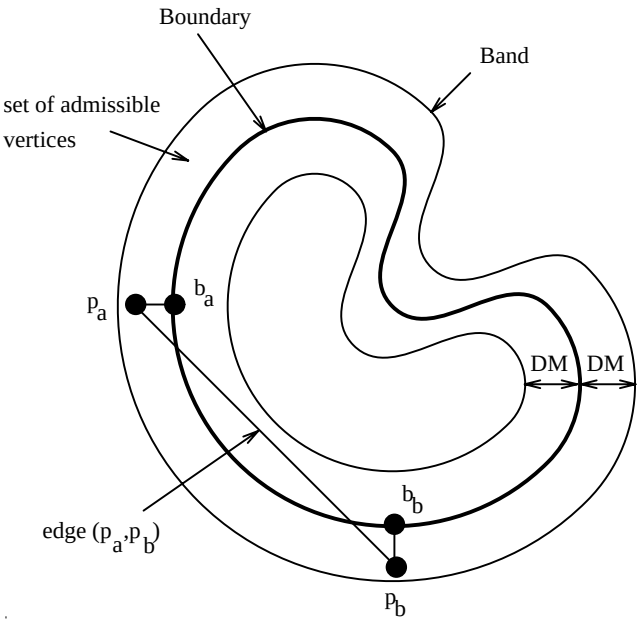

Figure 1: The "band" concept.

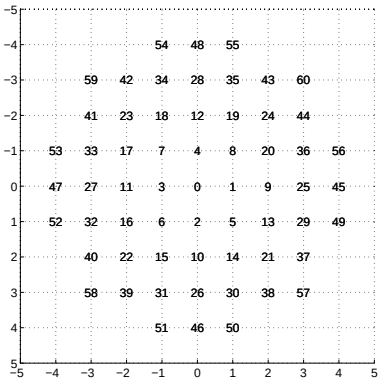

Figure 2: Vector increments for the ordering of the admissible vertices. Vector v_j starts at 0 and ends at j.

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

Table 1: Code word assignment for the runs

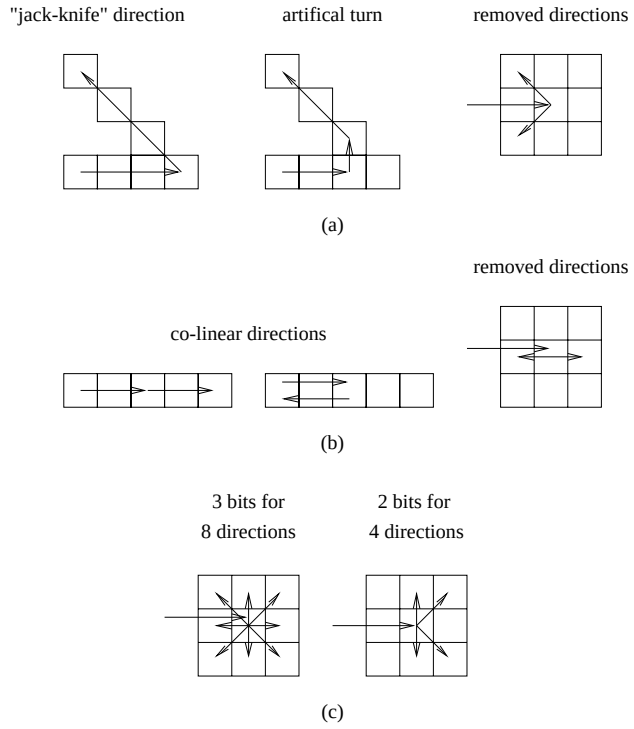

Figure 3: Improved orientation encoding. No code word exists for the "jack-knife" directions in Fig. (a). If a "jack-knife" direction needs to be encoded for the optimal polygon, it is approximated with an artificial turn. No optimal polygon will select the two co-linear directions displayed in Fig. (b). In Fig. (c), the original 8 directions are now encode using only 2 bits, with only four code words available.

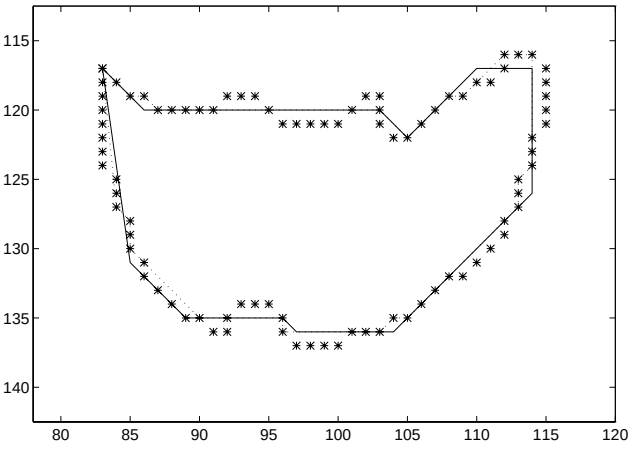

Figure 4: (*) Original boundary, 239 bits, using 8-connect chain code. (:) Optimal polygon for $D_{max} = 1$ pixel, 137 bits, using the algorithm proposed in [1]. (-) Optimal polygon for $DM = D_{max} = 1$ pixel, 72 bits, using the proposed algorithm.