SPATIAL INTERPOLATION ALGORITHM FOR ERROR CONCEALMENT

Hamid Gharavi, and Shaoshuai Gao

National Institute of Standards and Technology
100 Bureau Drive, Gaithersburg, MD 20899 USA

ABSTRACT

In this paper, we propose a new spatial interpolation algorithm for Intra-Frame error concealment. The method aims at interpolating areas in the image, which have been affected by packet loss. We have proposed an edge detection technique to aid the bilinear interpolation. The edge-detection scheme is based on designing a robust Hough transform-based technique that is capable of systematically connecting edges irrespective of the number of edge points surrounding missing areas. The connected edges are used to divide the missing areas into different regions for interpolation along the directions of each detected line. Simulation results demonstrate that the proposed algorithm can recover the missing areas with a greater accuracy, when compared with the bilinear interpolation technique.

***Index Terms*—** Error concealment, directional interpolation, Hough transform, region growing

1. INTRODUCTION

In order to obtain a higher compression ratio many image and video coding standards are based on block, prediction coding and variable length coding (VLC). However, when the coded streams are transmitted over error-prone wireless channels, the effect of errors may result in false decoding and a loss of synchronization at the decoder. As a result, a number of blocks will be lost and the decoded image quality will be severely degraded. To alleviate this effect, one method is error concealment [1], which aims at minimizing the distortion caused by loss of packets at the decoder. It takes advantage of the spatial and/or temporal correlation in the received video to interpolate the missing data. For image coding and INTRA coded pictures (I-pictures) in video coding, only the spatial correlation will be exploited.

A simple and typical spatial error concealment method is bilinear interpolation, which is to interpolate each pixel in the lost areas from intact neighboring pixels [2]. In [3]-[7], more advanced techniques were proposed to perform the interpolation adaptively to achieve maximum smoothness and/or keep the edge information. Hough transform [10], which has been used for many applications in computer vision such as edge detection and connection [11-13], can also be used for edge preserving error concealment. For example, in [8] the application of the Hough transform for spatial error concealment was suggested using directional interpolation of a broken edge in an isolated lost block. However, the main problem with the Hough transform approach is a lack of robustness in connecting multiple broken edges, particularly when they are in close proximity.

Thus, in this paper we present a robust Hough-based edge-connection algorithm. The scheme is capable of automatically connecting all the crossing edges broken by a loss of packet which may consist of multiple blocks (or macroblocks). To signify the effectiveness of the proposed algorithm, we used a simple bilinear interpolation scheme.

The paper is organized as follows. In Section 2, the proposed algorithm is described. Simulation results and discussions are then presented in Section 3, followed by the conclusion in Section 4.

2. PROPOSED ALGORITHM

In this section, we present our Hough transform-based scheme, which has been used to connect broken edges in the image. Before applying the algorithm, dominant edge points around the lost areas are first detected using the Sobel operator. Then, all the detected edge points are refined through a thinning algorithm. Subsequently, the Hough transform is applied to systematically connect the dominant edges that are broken as a result of data loss. The connected edges are then used to segment the missing areas into different regions prior to the interpolation process.

2.1. Edge Detection

In order to get information of edges around the lost areas, one of the efficient and popular methods is using gradient measures by the Sobel mask operator.

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \; S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \tag{1}$$

That is, for every pixel *p(x,y)* around the lost areas, the Sobel mask is applied as follows:

$$\begin{aligned} g_x = {} & p(x+1, y-1) - p(x-1, y-1) + 2p(x+1, y) \\ & - 2p(x-1, y) + p(x+1, y+1) - p(x-1, y+1) \end{aligned}, \tag{2}$$

$$\begin{aligned} g_y = {} & p(x-1, y+1) - p(x-1, y-1) + 2p(x, y+1) \\ & - 2p(x, y-1) + p(x+1, y+1) - p(x+1, y-1) \end{aligned}. \tag{3}$$

The magnitude of the gradient at *p(x, y)* is:

$$G = \sqrt{g_x^2 + g_y^2}\,. \tag{4}$$

If G is larger than a predefined threshold, the corresponding pixel *p(x, y)* is then considered an edge point. However, this may result in the detection of too many edge points around the lost areas, as illustrated in Fig. 1(c), which shows an example of edge detection for a portion of the image “Lena” (the original image is shown in Fig. 5a). Consequently, it is hard to distinguish between dominant and insignificant edges, which are crucial for directional interpolation. To solve this problem, a thinning algorithm is used to make the detected edges more distinct.

Thinning [9] is a pixel reduction process that erodes an object until it is one-pixel wide, producing a skeleton of the object. It is usually applied in recognizing objects such as letters or silhouettes by looking at their bare bones. Here, we apply the thinning algorithm to obtain the one-pixel wide edges. To do this, thinning erodes clusters of edge points over and over (without breaking it) until they are narrowed down to one-pixel wide, as can be observed in Fig. 1(d). This will simplify the edge connection process that is described next.

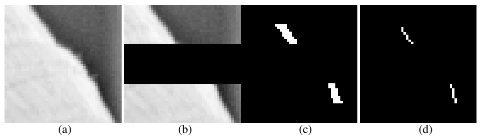

Fig. 1. An example of edge detection for a portion of the image “Lena”; From left to right then above to bottom are: (a) Original image; (b) The image with packet loss; (c) The result of the Sobel operator. (d) The result after Thinning.

2.2. Hough Transform and Edge Points Connection

To obtain the direction of the edges and connect some separated edge points, a well-known technique called Hough transform is considered [10]. The basic idea of this technique is to find curves that can be parameterized like straight lines, polynomials, circles, etc., in a suitable parameter space. Here, we consider the Hough transform to detect and connect separated line segments formed by a co-linear set of points. We assume lines are parameterized in the form,

$$\rho = x\cos\theta + y\sin\theta , \tag{5}$$

where ρ is the perpendicular distance from the origin and θ the angle with the normal. Co-linear points (x_i, y_i), with $i = 1,..., N$, are transformed into N sinusoidal curves $\rho = x_i \cos\theta + y_i \sin\theta$ in the (ρ,θ) plane, which intersect in the point (ρ,θ).

The transform is implemented by quantizing the Hough parameter space into finite intervals or accumulator cells. As the algorithm runs, each (x_i, y_i) is transformed into a discretized (ρ,θ) curve and the accumulator cells that lie along this curve are incremented. Since co-linear edge points in the image would accumulate in the same cell, a peak in the accumulator array would indicate the existence of a straight line in the image. Bear in mind that co-linear edge points in the image intersect in the Hough transform space. However, for image applications, not all the lines are straight. In addition, there may be too many edges that don’t pass through a missing region.

Connecting edges through the Hough transform depends on the size of the accumulator cell by which the highest peaks are identified. For a straight line consisting of *n* points, the size of the maximum cell should ideally be *n*. However, in the image domain this depends on the image’s vertical and horizontal resolutions. For example, every pair of pixels along the straight line in an image may not have exactly the same Hough parameters; hence they may not be accumulated in the same cell. However, by appropriately quantizing the Hough parameters it is possible to merge these pixels into the same location. Also, coarser quantization could help to connect edges with slight curvatures.

In order to apply Hough transform to connect broken edges, the most straightforward approach would be to detect the crossing points of a passing through edge. As shown in Fig. 2, these points (e.g., point 1 and point 2) can then be connected if the two pieces of a broken edge (edge 1 and edge 2) have similar ρ and θ parameters. This approach however, is rather tedious, especially when there are a large number of passing through edges around the lost area.

In this paper we have developed a robust technique that is capable of connecting edges systematically, regardless of the number of edges surrounding a missing region. As shown in Fig. 3, a fixed number of pixels surrounding a missing region are considered for measuring peaks in the accumulator cells in the Hough domain. Under these conditions, the highest peak corresponds to the most dominant edge (close to a straight line), which can then be connected automatically via inverse Hough transformation. However, when there are many crossing edges, we should expect multiple peaks within the array of accumulator cells. We should point out however, that even in the case of a single crossing edge (in the image domain) not all its co-linear points in the Hough domain, intersect exactly at the same point (i.e., ρ and θ) and thus, may not be accumulated in the same cell. Under these conditions, the neighboring cells with high peaks should merge into one cell prior to inverse transformation. Inverse transformation, in this case, would be based on the average values of ρ and θ for all the neighboring cells that were merged together. Note that the cells are considered to be neighbors if their corresponding ρ and θ are within close range of each other (e.g., second stage quantization).

In this arrangement lower peaks (second, third, and so on) may also be considered to detect less dominant edges. Lower peaks may include edges with slight curvatures or lines which may not be long enough to pass through all the neighboring edge pixels surrounding the lost packet. Furthermore, as indicated in Fig. 3 there may be some near horizontal lines in the vicinity of the missing area (non-passing through edges), which cannot be effectively used for interpolation. These lines are eliminated from the accumulator cell (θ close to $\pi/2$).

The following steps summarize the edge connection process:

Step 1. Select n neighboring pixels surrounding the missing packet (e.g., *n* lines above and below the lost area);

Step 2. Find all the cells whose peaks are larger than a predefined threshold t_p.

Step 3. Merge the neighboring cells whose Hough parameters are close to each other, into one cell.

Step 4. Perform inverse Hough Transform to connect the broken edges.

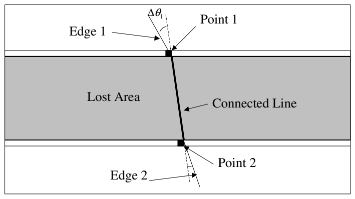

Fig. 2. Illustration about connetion of the two broken edges detected by Hough Transform.

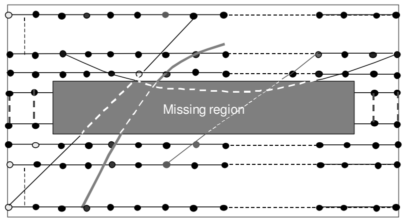

Fig. 3. A systematic edge connection scheme using neighboring edge points surrounding the missing region.

2.3. Region-based Interpolation

Having connected the broken edges, the next step is to segment the lost areas into different regions to aid the interpolation process. This is achieved by a pixel labeling method known as region growing. Region growing is a segmentation process that puts pixels that have the same property into one group. As shown in Fig. 4, for interpolating every pixel in the lost area, only the closest pixels in the same zone (permissible zone) are used. The permissible zones are those that have the same property as the region in which its pixel will be interpolated (without any edges between them). The neighboring points in the forbidden zones will not be used for interpolation. This approach can preserve the edge information in interpolating the lost areas, thereby preventing a significant blurring effect.

We have used bilinear interpolation using the closest reference pixels within the same zone. The weight used for interpolation can be shown as,

$$p_{x,y} = \frac{D_b}{D_a + D_b} p_{x+k1,y-l} + \frac{D_a}{D_a + D_b} p_{x+k2,y+N-l} \qquad (6)$$

Where, D_a and D_b are the distances between the interpolating pixel ($p_{x,y}$) and its same-zone bilinear reference pixels. These pixels are represented as $p_{x+k1,y-l}$ and $p_{x+k2,\ y+(N-l)}$.They are the nearest pixels to $p_{x,y}$ that are located above and below the missing area at the horizontal distances of $k1$ and $k2$, and the vertical distances of l and $(N-l)$, respectively. N is the height of the missing area (e.g., macroblock).

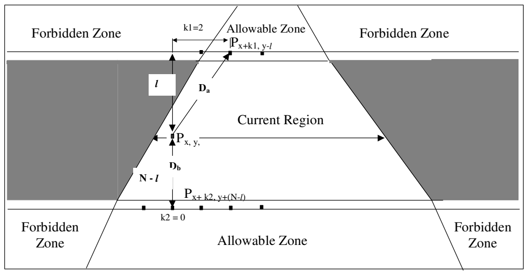

Fig. 4. Region-based interpolation

3. SIMULATION RESULTS

The performance of the proposed algorithm is tested on several still images of 512 × 512 with the size of the corrupted blocks being 16×16. In addition, for the sake of comparisons, we use the PSNR, which includes all the pixels in the lost areas. We have compared the performance of the proposed method with other well-known algorithms [2], [3], [4], and [5]. Table I shows the results of these comparisons in terms of PSNR.

Table 1. PSNR values (in dB) of test images reconstructed by several algorithms

PSNR	Salama's	Park's	Wang's	Sun's	Proposed
Lena	19.90	20.04	20.92	20.14	23.06
Pepper	20.34	20.63	20.87	19.12	24.79
Zelda	22.39	22.23	23.37	22.17	25.12

From this table we can clearly observe that that the proposed algorithm can significantly improve the PSNR performance compared with other schemes. For the "Lena" and "Zelda" images, a PSNR gain of about 2 dB can be achieved using the proposed algorithm. For the "Pepper" image, the improvement would be even greater.

Fig. 5 and Fig. 6 show the subjective comparison results of two different spatial interpolation algorithms. One is the bilinear interpolation algorithm [2] and the other is our proposed interpolation algorithm. From both figures we can observed that the bilinear algorithm (Fig. 5 (c) and Fig 6 (c)) restores the missing areas, but at the expense of significant blurring effects around the edges of the recovered areas. In Fig. 5(d) and Fig. 6(d) however, with the proposed scheme, it can be observed that many dominant edges have been reconstructed gracefully using the proposed algorithm. Thanks to the consideration of lower peaks in the Hough domain, we can clearly observe the recovery of some of the curved edges in the picture of 'Lena' (around the hat and arm). The reconstruction quality of the first frame of 352×288 video sequence "Foreman" that contains more straight edges, is distinctly noticeable. For this image, Fig. 7 shows the recovery stages in more detail.

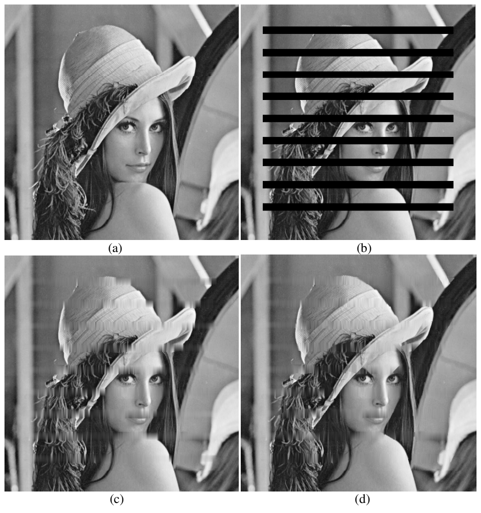

Fig. 5. Subjective comparison of different algorithms for the image "Lena". (a) The original image, (b) The corrupted image, (c) The reconstructed image by the bilinear interpolation, (d) The reconstructed image by the proposed algorithm

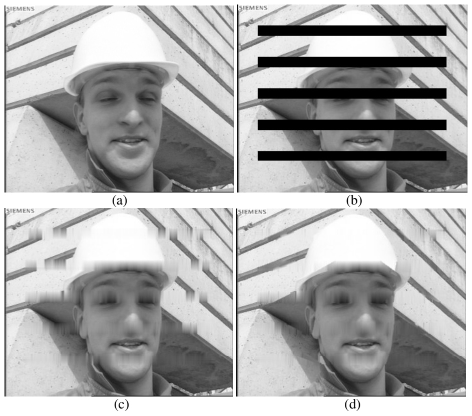

Fig. 6. Subjective comparison of different algorithms for the first frame of video sequence "Foreman". (a) The original image. (b) The corrupted image. (c) The reconstructed image by the bilinear interpolation. (d) The reconstructed image by the proposed algorithm.

4. CONCLUSION

Many video coding standards, such as H.264 and MPEG-4, are based on block prediction coding techniques. When a coded stream is transmitted over packet-loss networks, some blocks may not reach their destination. Consequently, this would not only distort the current frame, but also its effect may propagate to the following frames. In this paper our objective has been to develop an error concealment scheme for Intra-frames that utilizes information from neighboring macroblocks (MBs) in order to conceal the effect of missing pixels. Our approach was based on classifying regions in the neighborhood of the missing blocks. This was accomplished by developing a Hough transform-based algorithm capable of systematically connecting crossing edges, which have then been used to segment the missing areas prior to bilinear interpolation. This method shows a significant improvement in the quality of the concealed frame, especially around the dominant edges.

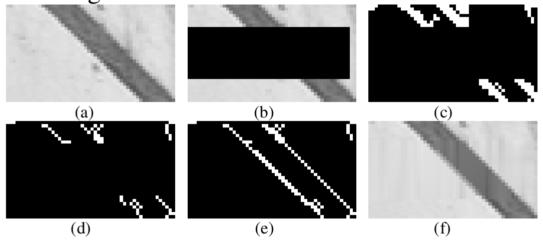

Fig. 7. An example of the proposed error concealment for a portion of the video image "Foreman"; (a) the original image; (b) the image with packet loss; (c) the edge map after Sobel operator; (d) the edge map after thinning; (e) the edge map after Hough transform; (f) the concealed image after region-based interpolation.

5. REFERENCES

[1] Y. Wang, and Q. F. Zhu, "Error control and concealment for video communication: A review," Proceedings of the IEEE, vol. 86, no. 5, pp. 974-997, May 1998.

[2] P. Salama, N. B. Shroff, E. J. Coyle, and E. J. Delp, "Error concealment techniques for encoded video streams," in Proc. ICIP'95, Washington, DC, Oct. 1995, pp. I 9–12.

[3] J. W. Park, J. W. Kim, and S. U. Lee, "DCT coefficients recovery based error concealment technique and its application to the MPEG-2 bit stream error," IEEE Trans. Circuits Syst. Video Technol., vol. 7, pp. 845–854, Dec. 1997.

[4] Y. Wang, and Q. Zhu, "Signal loss recovery in DCT-based image and video codecs," in Proc. SPIE Visual Communications and Image Processing'91, Boston, MA, Nov. 1991, pp. 667–678.

[5] H. Sun, and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," IEEE Trans. Image Processing, vol. 4, pp. 470–477, Apr. 1995.

[6] J. W. Park, and S. U. Lee, "Recovery of corrupted image data based on the NURBS interpolation," IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 1003–1008, Oct. 1999.

[7] T. P.-C. Chen, and T. Chen, "Second-generation error concealment for video transport over error prone channels," Wireless Communications and Mobile Computing, Special Issue on Multimedia over Mobile IP, vol. 2, pp. 607-624, Oct. 2002.

[8] D. L. Robie, and R. M. Mersereau, "The use of Hough transforms in spatial error concealment," in Proc. of IEEE ICASSP 2000, Istanbul, Turkey, pp. 2131-2134, 2000.

[9] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning Methodologies – A Comprehensive Survery," IEEE Trans. Patt. Anal. Machine Intell., vol. 14, no. 9, pp.869-885, Sept. 1992.

[10] P.V.C. Hough, "Machine Analysis of Bubble Chamber Pictures," International Conference on High Energy Accelerators and Instrumentation, CERN, 1959.

[11] R. O. Duda, and P. E. Hart, "Use of Hough transform to detect curves and lines in pictures," Commun. ACM, vol. 15, pp. 11–15, Jan. 1972.

[12] J. Illingworth, and J. Kittler, "A survey of the Hough transform", Computer Vision, Graphics, and Image Processing, vol. 44, pp.87-116, Oct. 1988.

[13] N. Aggarwal, and W.C. Karl, "Line detection in images through regularized Hough transform," IEEE Trans. Image Proc., vol. 15, pp. 582-591, Mar. 2006.