Object-driven block-based algorithm for the compression of stereo image pairs

E.A.Edirisinghe [*a], J.Jiang [b], H.E.Bez [a]

[a] Department of Computer Science, Loughborough University, UK
[b] School of Computing, Glamorgan University, UK

ABSTRACT

In this paper, we propose a novel object driven, block based algorithm for the compression of stereo image pairs. The algorithm effectively combines the simplicity and adaptability of the existing block based stereo image compression techniques [1-6] with an edge/contour based object extraction technique to determine appropriate compression strategy for various areas of the right image. Extensive experiments carried out support that significant improvements of up to 20% in compression ratio can be achieved by the proposed algorithm, compared with the existing stereo image compression techniques. Yet the reconstructed image quality is maintained at an equivalent level in terms of PSNR values. In terms of visual quality, the right image reconstructed by the proposed algorithm does not incur any noticeable effect compared with the outputs of the best algorithms.

The proposed algorithm performs object extraction and matching between the reconstructed left frame and the original right frame to identify those objects that match but are displaced by varying amounts due to binocular parallax. Different coding strategies are then applied separately to internal areas and the bounding areas for each identified object. Based on the mean squared matching error of the internal blocks and a selected threshold, a decision is made whether or not to encode the predictive errors inside these objects. The output bit stream includes entropy coding of object disparity, block disparity and possibly some errors, which fail to meet the threshold requirement in the proposed algorithm

Keywords: Stereo image compression, video compression, contour analysis, algorithm development.

1. INTRODUCTION

Although a series of JPEG [16] and MPEG [17,18] standards represent a global effort for developing image compression technology, they are not efficient for the compression of stereo image pairs. However, the applications requiring stereo images for better information access and interpretation are huge, which makes stereo image compression technology an important area for further research and development. Stereo images are constructed by simulating human's eyesight effect upon observing objects through two horizontally separated positions. Correspondingly, every stereo image gives rise to two frames labelled as *left frame* and *right frame*. If these two frames were to be transmitted independently, we would need double the bandwidth required for monocular image transmission. However, due to the fact that the left and right frames are produced by observing the same scene from two slightly different positions, they contain significant redundant information in the form of *inter-frame* redundancy. Since this is in addition to the *intra-frame* redundancy present in the two constituent images, significant data compression can be achieved by exploiting both types of redundancies when data compression algorithms are developed.

If we superimpose one of the stereo image pair frames on top of the other, we would observe that the matching points on the images do not coincide, but are apparently displaced from one another. This is called *disparity* or *binocular parallax*. The disparity of a point depends on the distance of it from the camera image plane. Fundamentally, there are two different ways to estimate the *disparity field* of a stereo image pair: the *intensity based* method and the *object/feature based* method. The first looks for correspondence between luminance values [1-6], and the second determines a set of objects/features in both images and seeks correspondence between the two sets [10,15].

The simplest intensity based method is *block matching*, in which the image frames are divided into sub-blocks of an appropriate size and are matched under a certain matching criterion for maximum correspondence. Several authors [1-6]

Part of the SPIE Conference on Applications of Digital Image Processing XXII
Denver, Colorado • July 1999
SPIE Vol. 3808 • 0277-786X/99/$10.00

have proposed *block-based* stereo image compression algorithms. These algorithms have the advantage of being simple and being easily adaptable for being used in association with standard image compression techniques (JPEGs and MPEGs) [16-18]. However, they suffer from inefficient exploitation of the inter-frame redundancy that is unique to stereo images in the sense that disparity is determined via fixed blocks rather than objects. Yet human visual perception of depth in stereo images is exactly interpreted in terms of objects rather than the rigid blocks. Therefore, it is unlikely that block-based techniques could achieve any further improvement in comparison with the existing state-of-the-art developments [1-6]. In addition, it also fails to address the compression strategy in an object-based perspective, which is the core in the development of modern and future interactive communication systems. Thus, in this paper, we propose a novel block-object hybrid approach as a solution. The basic idea behind the development of this algorithm is the efficient exploitation of redundancy in smoothly textured areas that are present in both frames, but are relatively displaced from each other due to binocular parallax. The algorithm has been designed based an object-layered platform, in a similar spirit to that of MPEG-4.

The rest of the paper is organised as follows. Section 2 describes an overview of the proposed algorithm. Section 3 discusses contour analysis of a stereo image pair, which is the major technique used to extract objects between the left and right frames. In this section, special emphasis is given to contour *extraction, matching, blocking* and *filling* strategies. Section 4 presents the coding techniques used for various types of objects and areas identified as a result of contour analysis. Section 5 gives the experimental results and their analysis. Finally, conclusions and possible further research are presented and identified in Section 6.

2. AN OVERVIEW OF THE PROPOSED ALGORITHM

The block diagram of the proposed stereo image encoder is shown in Figure 1. The left image, which is selected as the reference, is independently compressed using a JPEG encoder [16]. It is then reconstructed locally at the encoding end in order to achieve a guaranteed correct decoding of all right frames, which requires that both encoder and decoder use the same reference frame during the entire process of matching and encoding.

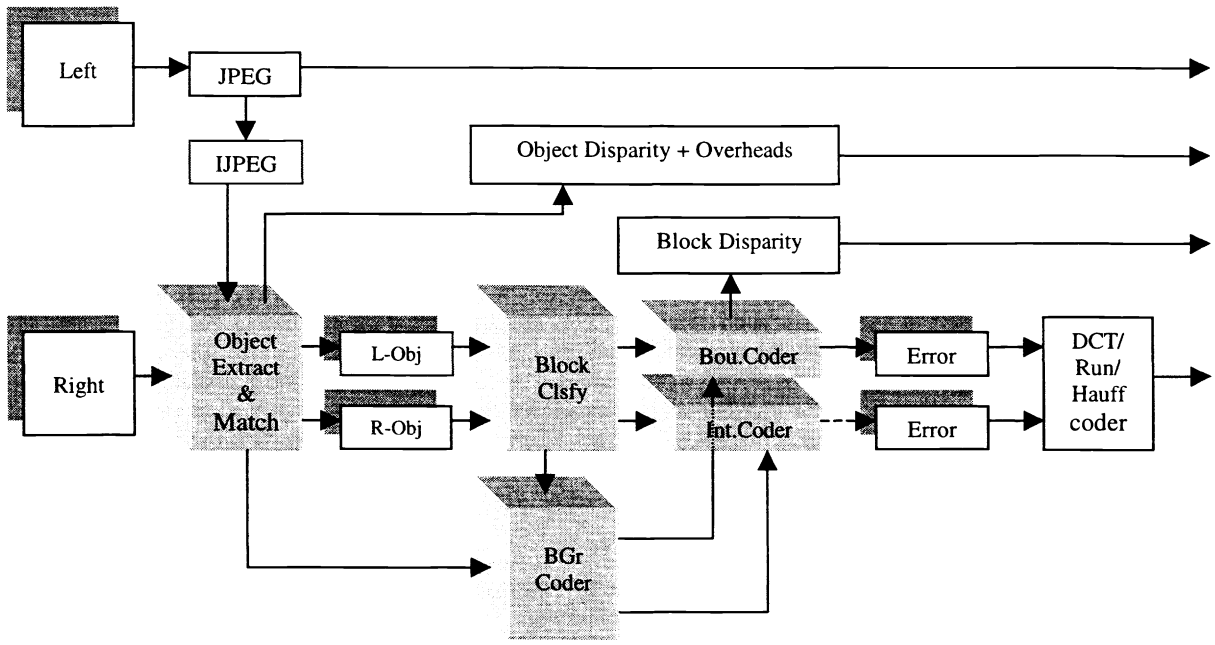


Figure 1. Block Diagram of Object Based Stereo Image Encoder

The proposed algorithm performs object extraction and matching between the reconstructed left frame and the original right frame to identify those objects or areas that match but are displaced by varying amounts due to the binocular parallax. The first step towards achieving this is to use a contour extraction process (3.1) [19,23]. Depending on one or more pre-selected intensity thresholds, an edge detection step [22] followed by a contour tracing strategy [19] is adapted on both frames to

identify objects or areas that have similar texture. The resulting contours are matched based on their shape information [19] to find those objects/areas, which are similar in shape (3.2). The matching object pairs are then included in their tightest possible bounding rectangles (3.3). These rectangles are suitably extended in order to be divided into 8×8 sub-blocks. The right bounding rectangle is then adjusted to the size of the left frame bounding rectangle which would act as the reference. The resulting rectangles are identified as *object planes*. Note that the centres of areas (centroids) of matching objects are calculated and used when the bounding rectangles are defined and aligned. The next step is to classify the constituent 8×8 sub-blocks on both object planes into three types namely, *interior blocks, boundary blocks* and *exterior blocks* (3.4). The *boundary* blocks are those that have at least one of the contour points within them and thus are easily identified. A parity based contour filling algorithm [20,21] is used to identifty the interior and exterior pixels (3.5). At the end of this step, the shapes of the left and right objects are also identified as binary object planes. These planes are used as an aid in the coding process.

Before the objects are coded the identified objects are further categorised into three groups: (i) matching object pairs that are enclosed by closed contours; (ii) matching object pairs enclosed by contours that terminate at the image frame boundaries; and (iii) unidentified areas that are treated as the background. The first two groups of objects are treated in a similar way but with slight differences in the way that bounding rectangles are found and aligned. After the above matching objects have been identified, the reference frame background becomes the area that remains unidentified within the left frame. However, the right frame background is the area that remains non-coded after all the matched objects have been decoded at the encoder end. In order to prevent the transmission of shape information of the right frame objects and improve the compression efficiency, we propose a special shape coding strategy (4) in which the shapes of the right frame objects are determined by their matching left frame objects.

The coding strategy for a right frame object, which has been enclosed within its bounding rectangle, can be described as follows. We first pad the left object plane pixels which are outside the object area using a special *extrapolated average (EA) padding* technique followed by a *extended padding* technique (4.1). The interior blocks of the right frame objects are then encoded using a fixed disparity value, which is defined as the *object disparity*. This is found by calculating the average disparity of all internal blocks. If the *mean squared error* (*MSE*) of the internal blocks, calculated as the intensity difference between pixels, is lower than a certain threshold, we decide not to encode the errors. In such a situation, only the object disparity is transmitted to the decoder end. However, if the *MSE* is above the threshold, a further object extraction process is performed with new thresholds to identify more objects within the current object. The process is continued until the *MSE* for all identified objects fall below the threshold or the maximum number of iterations is achieved. In the latter case, the internal area errors are encoded using a pioneering block based prediction technique [3-4]. The arbitrary shaped boundary blocks of the right object plane are encoded using a special shape adaptive coding scheme (4.2). As a result of this process, the shape of the encoded right object would be exactly the same as the matching left object. At the decoding end, however, the shape of the right object will be correctly reconstructed since the difference information will be encoded by other neighbouring objects or background. To secure a high quality reconstruction at the decoding end, the disparity values and the predictive errors are always transmitted for all the boundary blocks. This also exploits the feature that human visual perception is more sensitive to boundary shape of objects than their internal areas where smoother texture is likely embedded. Finally, the external blocks are not encoded as they fall into the interior of some other object or the background.

In comparison with block based compression techniques, one of the advantages is that *end-of-block* codes (4 bits for each block usually) need not to be transmitted for every block in the right frame, as most blocks will fall into the interiors of areas that need not be corrected. More details will be discussed in latter sections.

3. CONTOUR ANALYSIS OF STEREO IMAGES

This section proposes a detailed contour analysis strategy for object extraction in stereo image pairs. It results in the development of a framework that enables matching objects to be identified in such image pairs. These objects are later used to exploit the redundancy present in stereo image pairs and achieve data compression.

3.1. Contour extraction

Contour extraction is performed by a two-step process [19]. Firstly, the image is convolved with a Laplacian-of-Gaussian (LoG) operator [22]. The LoG of a continuous function f(x,y) is defined as follows:

$$LoG(f) = \left(\frac{r^2 - \sigma^2}{\sigma^4}\right)\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{1}$$

where $r^2 = x^2 + y^2$ and σ is the standard deviation. To facilitate its application, equation (1) can be discretized in various ways [22] to form a LoG-mask.

After the image is convoluted with a suitable LoG mask (note that the smoothness of the contours extracted will depend on σ), the edges are detected at the zero-crossing points (e.g., patterns such as "++--" and "--++" along both vertical and horizontal directions). In the second step, the slopes of the LoG of the image along both x and y directions, denoted by $\mathbf{S}_x$ and $\mathbf{S}_y$, are used to compute the edge strength at each zero-crossing point. An edge strength at a point (x,y) is defined as follows:

$$S(x,y) = \begin{cases} \sqrt{S_x^2 + S_y^2} & \text{if } (x,y) \text{ is a zero crossing point,} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The contour points are chosen based on the following criteria: the edge strength at each point along the contour is greater than T_l and at least one point on the contour has an edge strength greater than T_u, where T_l and T_u are pre-set thresholds and $T_l < T_u$. Generally, T_l is set sufficiently low to preserve the whole contour around the region boundary and T_u is chosen large enough to avoid spurious edges. Contour search is initiated whenever one point with a value greater than T_u is scanned. The search is conducted in both directions of the contour and the neighbouring pixels with values greater than T_l are accepted as contour points. The search is terminated when no neighbouring pixels are found to satisfy this condition. Then all edge strength values along the detected contour are set to zero so that these points will not be visited again. The same search operation continues until the whole edge strength array has been scanned.

The result of this stage would be the identification of a number of object contours from the image frames. Figure 2, shows an example of the identified object contours in the image pair '*cans*'. Note that the short contours that cannot be used reliably in the matching process have been discarded.

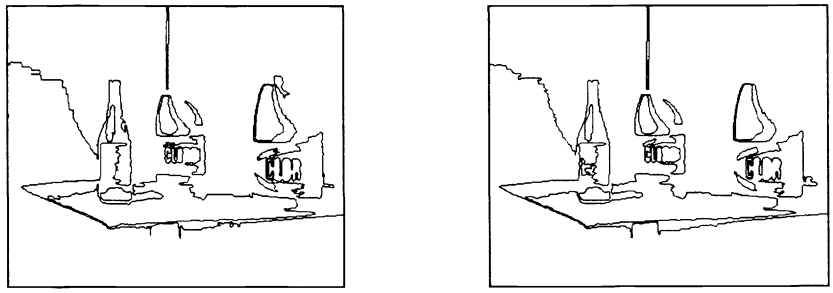


Figure 2. Left and right contour plots of image pair 'cans'

3.2. Contour matching

After the above contour extraction process, the resulting contours are categorised into two types: those which are closed and those which are open, which terminate at the image boundaries.

3.2.1. Closed contour matching

For every closed contour, three shape attributes are computed: (i) the number of pixels representing the perimeter of the contour, n; (ii) the y co-ordinate of the centroid (x_c, y_c); and (iii) the first invariant moment, h. Note that y_c is

considered as a good matching attribute as we assume that the vertical stereo disparity between points in the image frames are negligible. This is because that parallel axis geometry is used in obtaining all the test stereo image pairs. If x_i and y_i represent the x and y co-ordinates of the points along the contour, the first invariant moment, h is defined as follows:

$$h = \frac{1}{n^2}\sum_{i=1}^{n}\left[(x_i - x_c)^2 + (y_i - y_c)^2\right] \tag{3}$$

where, $x_c = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $y_c = \frac{1}{n}\sum_{i=1}^{n} y_i$. (4)

Every closed contour of the right frame is matched with every closed contour of the left frame. A contour from the left frame is accepted as an initial candidate match for the right frame contour, if the differences between each of their shape attributes fall below some pre-set thresholds (e.g. 20% for h, and 10% for y_0 and n). At the end of this procedure, closed contours in right frames may have multiple candidate matches and a further step is necessary to find the best match among them. This is performed as follows.

Firstly, the contours are converted into their chain codes $\{a_i \in \{0,1,2,\ldots 7\}\}$ [19]. Let $\{l_i\}$ and $\{r_i\}$ be the chain code representations of two contours L and R, corresponding to the left and right frames respectively, and let N_L and N_R be their lengths. We can define a measure of correlation MC_{kl} between two n-point segments, one starting at index k of contour L and the other starting at index l of contour R. The specific definition of MC_{kl} is given below:

$$MC_{kl} = \frac{1}{n}\sum_{j=0}^{n-1}\cos\frac{\pi}{4}(l'_{k+j} - r'_{l+j}) \tag{5}$$

where, $l'_{k+i} = l_{(k+i) \bmod N_L} - \frac{1}{n}\sum_{j=0}^{n-1} l_{(k+j) \bmod N_L}$, $0 \le i < n$, and r'_{l+i} defined similarly.

In order to identify the location of the best fit between the two contours, an n-point segment of R, starting at index k, is slid over contour L. The similarity function, $F_{LR} = \max\{MC_{kl}\}_{l \in M}$, where M specifies the search range, is then used to locate the best fit. Contour R in the right image and contour L in the left image are selected as a matched pair, if the following two conditions are satisfied.

1. $F_{LR} \ge F_{L'R}$, where L' represents all the contours with similar shapes to contour R.
2. $F_{LR} > T$, where T is a pre-set threshold that eliminates matches with poor correlation.

In the case that multiple contours get matched to the same contour, the pair with the highest F_{LR} is selected.

3.2.2. Open contour matching

For matching of open contours, the salient segments along the contours are used as the matching primitives [19]. Salient segments such as corners can be detected from the chain code representation. For a contour of length n with chain code $\{l_i\}$, we define a measure of curvature at the i^{th} point as,

$$C_i = \max_{1 \le j \le 3\sigma}\left\{\max\left\{|l_{i-j} - l_{i+j}|, |l_{i-j} - l_{i+j-1}|\right\}\right\} \tag{6}$$

where, σ is the standard deviation of the LoG operator used for the contour extraction. The i^{th} point along the contour is chosen as a salient point if both of the following conditions are satisfied,

1. $c_i \geq T_s$, and

2. $c_i \geq c_k$ for all $k \in [i-p, i+p]$.

where p is a constant that determines the minimum distance between the salient points, and T_s is a threshold specifying the minimum acceptable curvature. The contour segments surrounding the salient points are then used as 1-D templates in finding the corresponding matches in the other image.

3.3. Contour blocking

After a matching pair has been found, the next step is to enclose each contour within its smallest possible bounding rectangle [Figure 3]. These rectangles are then extended in all four directions to lengths of integer multiples of 8 as such that they are centred at a modified centroid of the corresponding contour area. To facilitate the matching process, the size of the rectangle bounding the right contour is changed appropriately so that it is identical to the one bounding the left contour.

Let $C = \{(x_i, y_i)\}$, where $i = 1,2,\ldots n$, represents an n-point contour. The centroid of this contour (x_c, y_c) is given by equation (4). We modify the y co-ordinate of this centroid to $y_{cm} = \frac{1}{2}(y_{cl} + y_{cr})$, where y_{cl} and y_{cr} are the y co-ordinates of the left and right contour centroids respectively. As we assume that parallel axis camera geometry and zero vertical disparity are used in obtaining the stereo image pairs, it is justifiable to align the bounding rectangles horizontally. Note that for the left contour we have: $x_c = x_{cl}$, and for the right contour, $x_c = x_{cr}$, where x_{cl} and x_{cr} represent the x co-ordinates of the left and right contour centroids. Let us denote the top, bottom, left and right sides of the bounding rectangle by R_top, R_bottom, R_left and R_right. Thus we have the following relationships:

$$R\_top = \max\{y_i\} + [8 - (\max\{y_i\} - y_{cm}) \bmod 8] \quad (7)$$

$$R\_bottom = \min\{y_i\} - [8 - (y_{cm} - \min\{y_i\}) \bmod 8] \quad (8)$$

$$R\_right = \max\{x_i\} + [8 - (\max\{x_i\} - x_c) \bmod 8] \quad (9)$$

$$R\_left = \min\{x_i\} - [8 - (x_c - \min\{x_i\}) \bmod 8] \quad (10)$$

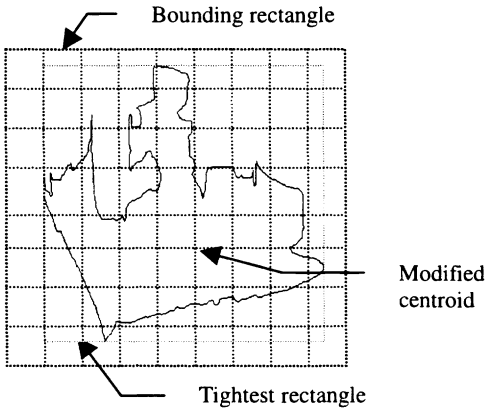


Figure 3. Contour blocking procedure

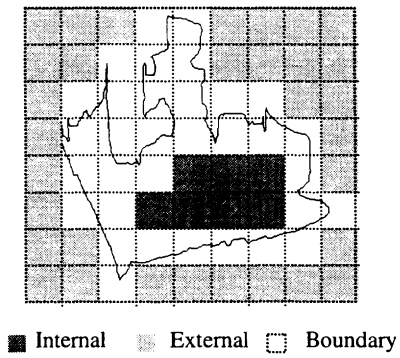


Figure 4. Block classification

3.4. Block classification

After enclosing the contour in the rectangle defined by equations 7-10, it is then divided into 8×8 sub-blocks. The resulting sub-blocks can be classified into three types: blocks that lie fully inside the contour, blocks fully outside the contour and blocks on the boundary of the contour. One example is shown in Figure 4. The classification of blocks is essential as they are treated differently in the coding process.

We propose the following strategy to differentiate the sub-blocks into the corresponding types. Let $(x_{TL}, y_{TL}), (x_{TR}, y_{TR}), (x_{BL}, y_{BL})$ and (x_{BR}, y_{BR}) respectively be the co-ordinates of top-left, top-right, bottom-left and bottom-right vertices of a sub-block B. Note that $y_{TL} = y_{TR} (= y_U)$, $y_{BL} = y_{BR} (= y_L)$, $x_{TL} = x_{BL} (= x_L)$ and $x_{TR} = x_{BR} (= x_U)$. Let (x_k, y_k) be a point on a given contour C. We select B as a block which contains at least one point of C, if for some k (=1,2...n), at least one of the following four conditions are satisfied.

I. $y_k = y_U \rightarrow x_L < x_k < x_U$

II. $y_k = y_L \rightarrow x_L < x_k < x_U$

III. $x_k = x_U \rightarrow y_L < y_k < y_U$

IV. $x_k = x_L \rightarrow y_L < y_k < y_U$

Once the above blocks are marked the rest fall into the two remaining categories: those that are completely inside the contour and those that are completely outside. If we find all four vertices of a sub-block to be inside the contour, we can guarantee that the block itself (i.e. all points belonging to the sub-block) would lie completely inside the contour. Similarly, if we find all four vertices to lie outside the contour, we can guarantee that the sub-block lies outside the contour.

3.5. Contour filling algorithm

Before encoding all those classified blocks, it is necessary to identify those pixel locations, which belong to the inside of the two contours. This is achieved by following a pixel based, parity check contour filling algorithm [20-21]. Figure 5 illustrates the flowchart of the algorithm. It should be self-explanatory except the procedure LINK, for which a pseudo-code can be presented as below.

Procedure *LINK*

0. Input $\{x,y\} \in$ C. Outputs, *above* and *below*.
1. *above=0, below=0*
2. **If** $\{x\text{-}1, y\text{+}1\} \in$ C, **then** *above* ++.
3. **If** $\{x\text{-}1, y\text{-}1\} \in$ C, **then** *below* ++.
4. **While** $\{x,y\} \in$ C **do** steps 5-7.
   Begin.
5. **If** $\{x,y\text{+}1\} \in$ C & $\{x\text{-}1, y\text{+}1\} \notin$ C, **then** *above* ++.
6. **If** $\{x, y\text{-}1\} \in$ C & $\{x\text{-}1, y\text{-}1\} \notin$ C, **then** *below* ++.
7. x ++.
   End.
8. **If** $\{x\text{-}1, y\text{+}1\} \notin$ C & $\{x, y\text{+}1\} \in$ C, **then** *above* ++.
9. **If** $\{x\text{-}1, y\text{-}1\} \notin$ C & $\{x, y\text{-}1\} \in$ C, **then** *below* ++.
10. **Return** location of pixel $\{x,y\}$ and values of counters *above* and *below*.
11. **End of procedure.**

The above algorithm is proved to work well [20-21] for all those contours, which enclose full regions, i.e. the contour has no multiple points (no boundary doubles upon itself). However, there may be cases when the extracted contour does not enclose a full region. In such situations we either break the enclosed region into its constituent full regions or define each of them as a separate contour or disregard the non-full regions if they are insignificant.

Thus, if all four vertices $(x_{TL}, y_{TL}), (x_{TR}, y_{TR}), (x_{BL}, y_{BL})$ and (x_{BR}, y_{BR}) are inside the contour, we deduce that the sub-block defined by them lies completely inside the contour. Otherwise, they would be taken as being completely outside the contour. Note that, at this stage, there cannot be sub-blocks in which some of the vertices lie inside and the rest lie outside the contour. This is because we have already separated these sub-blocks using the algorithm proposed in section 3.4.

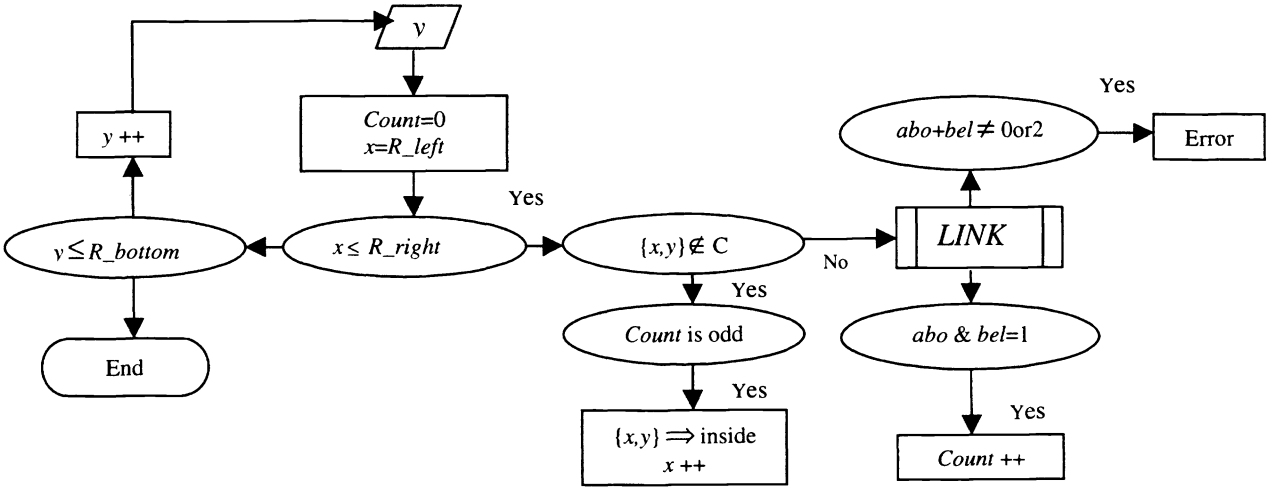


Figure 5. Flowchart of contour filling algorithm

4. OBJECT ENCODING

The stereo image pair is encoded on an object basis by identifying three types of objects within the images. These include: objects that are bounded within closed contours, objects that have open contours and both ends of the open contour terminate at image boundaries, and the rest which fall into the background. Although there exist differences among the three types and their encoding requires individual co-ordination, the major proposed encoding techniques can be described as: (i) padding of *left object bounding rectangles* (hereafter referred to as OBR); (ii) disparity-based prediction for both arbitrary shaped boundary blocks and internal blocks; and (iii) background encoding.

4.1. Padding of left OBR

To match the arbitrary shaped objects inside the right frame, the left OBR need to be padded to ensure that all pixels inside the right object are covered and a full square error can be calculated to produce the best match. Specifically, each arbitrary shaped boundary block of the left OBR is padded using an extrapolated average (EA) padding technique. Firstly, the arithmetic mean value A of all the block pixels $p(i,j)$ situated within the object region L is calculated using the following formula :

$$A = \frac{1}{N} \sum_{(i,j) \in L} p(i, j) \tag{11}$$

Where, $(1 \le i, j \le 8)$, N is the number of pixels situated within the object region L. Note that the division by N is done by rounding to the nearest integer. The next step is to assign A to each block pixel situated outside of the object region L, i.e.

$$p(i, j) = A \qquad for\ all \quad (i, j) \notin L \tag{12}$$

After all the boundary blocks are padded according to the EA padding technique, the exterior blocks immediately next to the boundary blocks are filled by replicating the samples at the border of boundary blocks. As each of these exterior blocks may

have four connected boundary blocks, the padding value is chosen based on a prioritised manner. The priority order is selected as starting from the boundary block to its left and then follow a clock-wise route to determine the priority. The remaining exterior blocks are filled with an intensity value of 128. At the end of this extended padding procedure all the pixels within the left OBR are filled and it is ready to be used as the reference plane.

4.2. Disparity-based prediction

For boundary blocks, the best match of the right OBR is found by searching within a horizontaly spread window, w=[-7,+7], centered at the corresponding block location in the padded left OBR. Note that the search needs to be done only in a horizontal direction as we asuume parallel axis camera geometry [1]. This assumption also implies that the objects in the left frame only displaces to the left of the right frame objects. However, in the above search procedure it is necessary to do the search on either side of the corresponding location, due to the fact that the search is done on an already displaced left OBR. For the same reason, the search range can also be limited to low values. The window size is appropriately clipped when the search is close to the bourndaries of left OBR, so that search is always carried out only inside the left OBR.

Let L and R represent the regions within the left (unpadded) and right objects, respectively. We define two binary object planes $Alpha_L$ and $Alpha_R$ as :

$$Alpha_R(r,s) = \begin{cases} 1 & if\ (r,s) \in R \\ 0 & otherwise \end{cases} \quad \text{and} \quad Alpha_L(k,l) = \begin{cases} 1 & if (k,l) \in L \\ 0 & otherwise \end{cases} \tag{13}$$

Where values of (k,l) and (r,s) are limited inside the left and right OBRs. Let $R\_block=\{x_{ij}\}$ represent a right block to be encoded, and $L\_block=\{y_{ij}\}$ be a block from the padded left OBR, a search inside the window $(1 \le i, j \le 8)$ would enable us to calculate the squared error (SE) between the two blocks as follows.

$$SE(R\_block, L\_block) = \sum_{i,j=1}^{8} (x_{ij} - y_{ij})^2 \times [Alpha_R \neq 0] \tag{14}$$

As explained earlier, $Alpha_R$ is used to ensure that only the pixels within the right objects contribute to the calculation of the squared error. Hence, the best match, $L\_block_{match}=\{m_{ij}\}$, can be found by :

$$L\_block_{match} = \min_{L\_block \in w} SE(R\_block, L\_block) \tag{15}$$

Since the right OBR is not padded, each boundary block taken from the right OBR actually contains all the pixels whether they are inside the right object or not. If we represent this block by $S=\{s_{ij}\}$, the error block between the left and the right can then be produced as follows:

$$e(i,j) = (s(i,j) \times Alpha_L) - (m(i,j) \times Alpha_L) \tag{16}$$

In this equation, $Alpha_L$ is used to determine the shape of those right objects. Consequently, the shape of the right object encoded in this way would be the same as that of its matching left object. Otherwise, we would need extra bits to encode the shape information for the right objects. However, this does not mean that we would have the shape of all right objects be distorted. As a matter of fact, the difference between the shapes of left and right objects would not disappear. They would be encoded by other neighbouring objects or background.

As illustrated in Figure 1, all the operations explained above in object extraction and encoding are carried out in reconstructed left frame for both the encoder and decoder. Hence, correct decoding is guaranteed by using the JPEG decoded left frame as a reference to reconstruct the right frame. In addition, the disparity values and the prediction errors for each arbitrary shaped boundary block is transmitted on an object by object basis, thus enabling the error blocks and disparity values to be properly identified and used to reconstruct the right objects at the decoding end. Considering the feature of object-based encoding for those arbitrary shaped boundary blocks and the fact that internal areas are often of smooth

texture, two different techniques are applied. They are: (i) only one single disparity is encoded for the entire internal area of each object; (ii) whenever the internal texture is detected to be smooth enough, no error block is encoded.

Specifically, an *MSE*-based block matching technique is firstly used to find the disparity of all internal blocks. The average disparity is then calculated and used to determine the single disparity for the entire internal area of each object. With this disparity, matching blocks are selected to produce error blocks. The internal texture is detected by comparing the error blocks with a selected threshold. If the errors fall below the threshold, a decision would be taken not to encode the internal error blocks. The average disparity value, however, is always transmitted along with other object parameters, in order to decode the internal right blocks. If the internal texture is not smooth, the decision would be either to go into further levels of object extraction or to use the pioneering block based technique [3-4] to encode those internal blocks. In this way, only error blocks are encoded without sending any disparity values.

The blocks which do not fall into either of the above categories, i.e. those that are in the exterior of the right object but within the right OBR are not encoded. They would be categorised as either background or a part of another extracted object, which would be encoded accordingly.

5. EXPERIMENTAL RESULTS AND ANALYSIS

To assess the proposed algorithm, extensive experiments were carried out by using the Disparity Compensated Predictive Coding (DCPC) algorithm proposed in [1] as our benchmark. The software simulation was implemented in MATLAB [24]. In order to make the experiments verifiable by other researchers, we established the test data set from a group of seven stereo image pairs downloaded from the Internet [25-26]. All the experimental results for the compression of all those right frames inside the data set are illustrated in Table 1.

	DCPC Algorithm		Proposed Algorithm	
Image Pair	ECR%	PSNR - dB	ECR%	PSNR - dB
Cans	38.88	39.52	44.05	38.59
Lamp	35.17	41.38	56.02	39.26
Lab	40.02	33.19	44.97	31.55
Slanted	45.30	34.68	52.19	33.61
Packs	43.82	35.11	45.62	32.56
Texture	44.44	34.72	57.29	33.48
RISC	20.85	39.94	35.50	37.89

Table 1. Experimental results

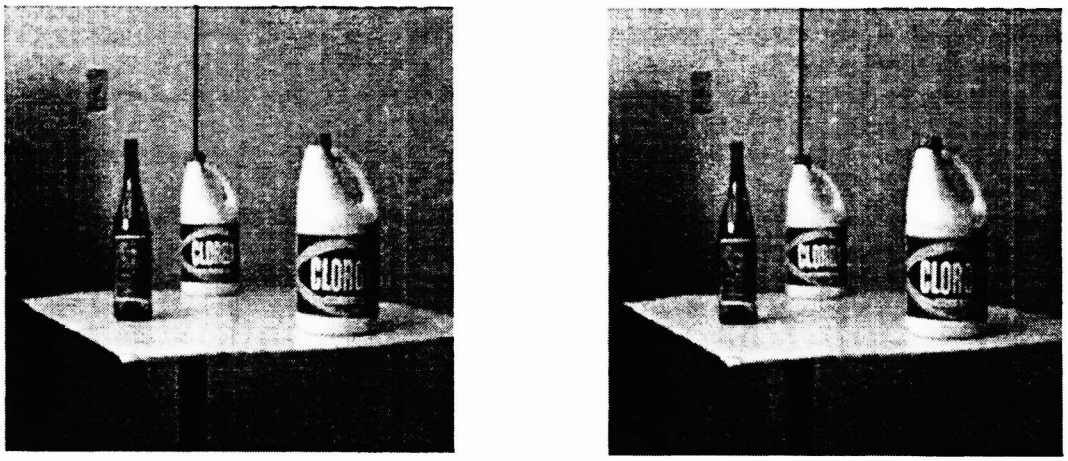
Since the right frame is compressed in addition to the conventional JPEG, we use a so-called *extra compression ratio* to measure the performance for all the algorithms tested. The Extra Compression Ratio (*E.C.R.*) is defined as,

$$E.C.R = \frac{bit\_No_{JPEG} - bit\_No}{bit\_No_{JPEG}} \times 100\ \% \tag{17}$$

where, bit_No_{JPEG} and bit_No stands for the total number of bits produced in compressing right frames by JPEG and by the proposed algorithm respectively. Thus, *E.C.R.* represents the additional data compression for the right frames in comparison with JPEG. For the left frames, the compression performance is entirely dependent on JPEG, which is used as a baseline technique in this occasion.

The total number of bits which are necesssary to reconstruct the right frame contains overhead bits in the form of disparity values of boundary blocks, object disparity values, location information for the centroid of those right frame objects (only the x-coordinate value) and the threshold values. The results shown in Table 1 for the proposed algorithm are produced by limiting the further thresholding of matched object internal areas to one additional step. In other words, if the *sub-objects* formed by one more step of thresholding for the internal areas do not satisfy the matching criteria, the pioneering block based prediction scheme [3,4] is used to encode the errors. A search window size of [16 (horizontal), 0 (vertical)] has been used for both algorithms. The PSNR values quoted are calculated with respect to the original images.

The experimental results in Table 1 clearly show a significant improvement on the compression performance for the proposed algorithm against the benchmark. For all the test images, the proposed algorithm overwhelmingly outperforms the benchmark in terms of E.C.R values. From the algorithm design presented in Sections 2-4, it can be seen that two major factors contribute to this improvement. The first one is the fact that the predictive errors for the internal areas of those identified objects are not encoded if the matching texture is detected to be smooth and within our quality threshold. The second factor can be analysed to include the following: (i) Although the internal area of each object is encoded by a block-based technique, the disparity is estimated in terms of objects rather than blocks; (ii) due to the object based coding strategy, no bits are used for the *end of block* as that in both JPEG and the DCPC scheme. This is because, for any block based algorithm, variable length of DCT error blocks require a special code-word (usually 4 bits) to indicate the end of each block, even if the predictive errors are all zeros.



(a) (b)

Figure 6. Reconstructed right image samples: (a) by the benchmark (b) by the proposed algorithm

Table 1 also illustrates that the PSNR values obtained for reconstructed right frames using the proposed algorithm are competitive in comparison with those achieved by the benchmark, although theoretically the PSNR values are smaller. This is caused by the selective coding strategy for the internal areas of objects. Figure 6 illustrates an example of the right frame for *Cans*, in which part (a) is reconstructed by the benchmark algorithm and part (b) by the proposed algorithm. Since we are using different coding strategies to compress those internal areas and boundary areas, visual comparisons between these two samples reveal that our reconstructed image has no noticeable difference compared to that of the benchmark. This is due to the fact that the proposed algorithm preserves the quality of reconstruction around the object boundaries, and would compensate for some loss of image quality in smoother internal areas of objects, as compared to the DCPC scheme [1]. This quality preservation near object boundaries has been experimentally proved to be a basic necessity for the visual comfort of the viewers [1,7,8]. Essentially, the division of a boundary block into two blocks, which have smoother texture variation, together with the left object padding technique results in a more accurate boundary block being reconstructed.

6. CONCLUSIONS

In this paper we have proposed an object-based algorithm for the compression of stereo image pairs. The basic idea behind the development of this algorithm is the efficient exploitation of redundancy in smoothly textured areas that are present in both frames, but are relatively displaced from each other due to binocular parallax. The identification and separation of such areas into matching object pairs improve the flexibility and efficiency when the coding strategy is decided. A special shape coding technique has been used for the boundary blocks in order to keep the coding efficiency to a maximum, yet enabling the right frame object shapes to be decoded using the shape of the matching left frame object. Experimental results show that significant compression improvement has been achieved in comparison with the existing block based coding techniques, while the reconstructed image quality remains competitive in PSNR measurement and visual inspection.

As the algorithm has been developed on an object-based platform, which is similar to that of MPEG-4, it can be transferable to the compression of stereo-videos or stereo image sequences. Under this circumstance, motion information in left frame sequence would need to be similarly exploited to determine the shape of object in the right frame sequence as well as the matching between each object pair.

REFERENCES

1. M.G.Perkins, "Data compression of stereo-pairs", IEEE Trans.on Comms., Vol.40, No.4, pp 684-696, 1992.
2. M.Ziegler, "Digital stereoscopic imaging and applications, a way towards new dimensions, the RACE II project DISTIMA", IEE Colloquium on Stereoscopic Television, London, 1992.
3. J.Jiang, E.A.Edirisinghe, H.Schroder, "Algorithm for compression of stereo image pairs", Electronics Letters, Vol. 33, No. 12, pp 1034-1035, 1997.
4. J.Jiang, E.A.Edirisinghe, H.Schroder, "A novel predictive coding algorithm for 3-D image compression", IEEE Trans. Consumer Electronics, 1997.
5. M.Forman, A.Aggoun, M.Mc Cormick, "Compression of integral 3DTV pictures", Proceedings of the International conference in Image Processing, IEE, 1995.
6. S.T.Barnard, W.B.Thompson, "Disparity analysis of images", IEEE Trans. Pattern Analysis Machine Intelligence, Vol. PAMI-2, pp. 333-340, July 1980.
7. I.Dinstein et.al., "Compression of stereo images and the evaluation of it's effects on 3-D perception", Proceedings of IEE Applications of Digital Image Processing XII, 1989.
8. T.Mitsuhashi, "Subjective image position in stereoscopic TV systems – considerations on comfortable stereoscopic images", SPIE Vol. 2179, pp. 229-265, 1994.
9. S.Sethuramam, M.W.Siegel, A.G.Jordan, "A multi-resolution framework for stereoscopic image sequence compression", Proceedings of ICIP-95, Vol. 2, pp 361-365, 1995.
10. D.Tzovaras, N.Grammalidis, M.G.Strintzis, "Object-based coding of stereo image sequences using joint 3D motion/disparity compensation", IEEE trans. on circuits and systems for video technology, Vol. 7, No.2, pp 312-327, 1997.
11. J.L.Dugelay, D.Pele, "Motion and disparity analysis of a stereoscopic sequence, application to 3DTV coding", EUSIPCO'92, pp 1295-1298, 1992.
12. N.Grammalidis, S.Malassiotis, D.Tzovaras, M.G.Strintzis, "3D motion estimation and compensation for stereoscopic image sequence coding.", Proceedings of the 4th Euro workshop on 3DTV, Rome, 1993.
13. D.V.Papadimitriou, T.J.Dennis, "3D parameter estimation from stereo image sequences for model-based image coding", Signal Proceedines of Image Communication, Vol. 7, pp 471-487, 1995.
14. B.Choquet, J.L.Dugelay, D.Pele, "A coding scheme for stereoscopic television sequences based on motion estimation-compensation using a 3D approach", IEE conf. in image processing and its applications. (IPA), pp 188-192, July 1995.
15. D.Q.Huynh, R.A.Owens, "Line labelling and region segmentation in stereo image pairs", Image and vision computing, Vol. 12, No.4, pp 213-225, 1994.
16. G.K.Wallace, "The JPEG still picture compression standard", IEEE Trans. on Consumer Electronics, Dec 1991.
17. MPEG-2, test model 5,*ISO/IEC/JTC1/SC29/WG11/93-225B*, Test model editing committee, 1993.
18. MPEG-4 standards, *MPEG97/N1796 – ISO/IEC JTC1/SC29/WG11, 1997.*
19. Hu Li, B.S.Manjunath, Sanjit K Mitra, "A contour based approach to multi-sensor image registration", IEEE Transactions in Image Processing, Vol.4, No 3, pp 321-324, 1995.
20. T.Pavlidis, "Filling algorithms for raster graphics", CPIG 10, pp. 126-141, 1979.
21. T.Pavlidis, *Algorithms for graphics and image processing*, computer science press, ISBN 0-914894-65-X, pp 167-195.
22. J.Chen, A.Huertas, G.Medioni, "Fast convolution with Laplacian-of-Gaussian masks", IEEE Trans., Pattern Analysis, Machine Intelligence, Vol. PAMI-9, pp 584-590, 1987.
23. E.A.Edirisinghe, J.Jiang, "A contour analysis based technique to extract objects for MPEG-4", Accepted for Int. Conference in Multimedia Computing and systems 1999, Florence, Italy, June 1999.
24. MATLAB, *Image Processing Toolbox*, The Math Works, Inc. USA.
25. http://www.vision.stanford.edu/~birch/research/pro.
26. http://www.cs.cmu.edu/afs/cs.cmu/project/cil/ftp/html/cil-ster.html *

* Correspondence: Email: E.A.Edirisinghe@lboro.ac.uk ; Telephone: +44 (0)1443 483291; Fax: +44 (0)1443 482715