

Object-Based Affine Motion Estimation

A. Gahlot¹

S. Arya²

D. Ghosh³

Department of Electronics and Communication Engineering
Indian Institute of Technology Guwahati
Guwahati, 781 039 INDIA

{¹anil_gahlot, ²saurabh_iitg}@yahoo.com, ³ghosh@iitg.ernet.in

Abstract—Motion estimation and compensation form a major part in any video coding scheme. The video coding standard adopted for multimedia applications is the MPEG-4 wherein object-based motion estimation comes into play. However, MPEG-4 uses conventional motion estimation technique that can detect only pure translational motion along the image plane and fails to consider any complex motion of the objects that arises due to rotation, zooming, etc. An efficient way of detecting complex motion is by using the affine motion model. In this paper, we develop a scheme for estimating affine motion of objects suitable for MPEG-4 video coding thereby achieving high quality. Simulation results demonstrate the efficiency of the proposed scheme in comparison to conventional object-based motion estimation techniques.

1. INTRODUCTION

Motion estimation is the process of estimating the motion of moving objects in a video scene. This is accomplished by determining the motion by which an object moves from one frame to the next. A given frame in a video sequence may, hence, be predicted from its previous frame by displacing all the moving objects in the previous frame by the estimated motion. This is motion compensation. The motion compensated frame difference is then coded and transmitted. Thus, motion estimation/compensation effectively removes the temporal redundancy present in the video signal while subsequent low bit-rate coding of the prediction error (motion compensated frame difference) brings about significant amount of compression. This has motivated to incorporate motion estimation/compensation in the basic coding structures of most low bit-rate video coding standards, viz., H.26x standards and MPEG standards.

The MPEG (Moving Picture Experts Group) is a working group under ISO/IEC in charge of the development of international standards for compression, decompression, processing and coded representation of moving pictures, audio and their combination [1]. So far MPEG has produced the following:

- **MPEG-1:** A standard for storage and retrieval of moving pictures and associated audio on storage media.

- **MPEG-2:** A standard for digital television.
- **MPEG-4:** A standard for multimedia applications.
- **MPEG-7:** A content representation standard for information search.
- **MPEG-21:** This is at present in the developmental stage.

Anticipating the rapid convergence of telecommunication, computer and TV/film industries, the MPEG group officially initiated MPEG-4 standardization phase in 1994 with the mandate to standardize algorithms and tools for coding and flexible representation of audiovisual data to meet the challenges of future multimedia applications and application requirements. In particular, MPEG-4 addresses the need for

- universal accessibility and robustness in error prone environments,
- high interactive functionality,
- coding of natural and synthetic data, and
- compression efficiency.

Bit rates targeted for the MPEG-4 video standard are between 5 to 64 kbits/s for mobile or PSTN video applications and up to 2 Mbits/s for TV/film applications. Some new key video coding functionalities have been defined which support MPEG-4 and provide the requirements related to “Content-Based Interactivity”, “Compression” and “Universal Access”.

As with most other video coding standards, the basic coding structure in MPEG-4 also involves motion estimation and compensation. However, while H.261, H.263, MPEG-1 and MPEG-2 use block-based motion estimation techniques, MPEG-4 employs object-based motion estimation. Here the moving foreground objects are first segmented out. The background information is transmitted to the receiver only once as first frame of the sequence and in each consecutive frame only the camera parameters relevant for the background are transmitted to the receiver. The arbitrary-shaped moving foreground objects are coded (in terms of motion vectors and prediction errors), transmitted and decoded as separate video object planes (VOPs) [2].

Unfortunately, MPEG-4 results in high quality coding only when the objects in a scene exhibit pure transla-

tional motion. This is because MPEG-4 uses conventional motion estimation techniques that can detect only translational motion along the image plane. But, in real world, the relative motion between the objects and the camera may be very complex due to rotation, flipping, shearing, zooming, panning, etc. in addition to translational motion. In such cases, the quality of coded video is not always satisfying. This means, for high quality coding it is necessary to estimate complex motion that an object in the scene may undergo. An efficient way of detecting complex motion that arises due to rotation, translation and zooming is the affine motion estimation. Incorporation of affine transformation in H.263 video coding standard has been proposed recently [3]. In this paper, we develop an object-based affine motion estimation scheme that may be incorporated in the currently ongoing MPEG-4 coding standard.

Several algorithms for estimating complex affine motion have been proposed in the literature. Some of these techniques are triangle-based motion compensation [4], [5], mesh-based motion compensation [6], [7] and grid interpolation [8]. In all these methods, the image is divided into triangular patches or quadrangular patches. The motion of a patch is considered as patch deformation that is measured in terms of the displacements of the vertices of the patch. Once the displacements of the vertices of a patch are exactly known, the motion of all points inside the patch are determined. The motion, thus obtained, is the estimated motion of the patch.

Recently, a block-based affine motion estimation algorithm has been proposed in [9]. In this, the affine motion model is combined with the conventional block-matching algorithm (BMA). So, while the conventional BMA can detect translational motion only, the proposed method makes BMA capable of capturing affine motion. On the other hand, this scheme offers significantly lower computational overhead compared to other affine motion estimation techniques and hence is well suited for real-time video coding. But, the algorithm is suitable for application only where block-based motion estimation is used, as in H.261, H.263, MPEG-1 and MPEG-2 codings. In this paper, we propose to extend this method so as to make it applicable in case of video coding schemes that involve object-based motion estimation, e.g., MPEG-4 coding.

2. OBJECT MOTION MODELS

Motion estimation techniques may be either pixel-based or region-based. In pixel-based motion estimation, viz., gradient technique [10] and pel recursive technique [11], motion is estimated for every individual pixel in the frame. This demands for high computational cost. The region-based motion estimation takes care of this computational cost where motion is estimated for a region

as a whole in the frame. A region may be a total object in the scene or may be a part of it. For a rigid object, all pixels in the object undergo same frame-to-frame displacements which correspond to the motion of the object itself. Hence, under the assumption that the objects are rigid, it makes sense to compute the motion of a region as a whole instead of pixel-wise motion.

The motion of 2D objects can be described by several models of increasing complexity. The simplest case is to allow only translational motion along the image plane. In the next levels, parameters for planar rotation, zooming (translation along the direction perpendicular to the image plane), etc. can be added. The effect of zooming manifests as scaling of objects in the scene. All these transformations can be described by a linear set of equations, as described below.

Translational motion model

If we assume that the motion of objects in a video scene is pure translation along the image plane, the coordinate of each pixel \mathbf{p}^t of an object in the current frame t is related to its corresponding position $\hat{\mathbf{p}}^{t-1}$ in the previous frame as

$$\mathbf{p}^t = \hat{\mathbf{p}}^{t-1} + \vec{v} \quad (1)$$

where \vec{v} is the motion vector by which the concerned object moves from frame $t - 1$ to frame t . The aim of motion estimation algorithm is to estimate this motion vector \vec{v} . One popular approach for this is the block-matching algorithm (BMA) [12]. In BMA, the current video frame is split into non-overlapping blocks; each block is considered to be an object as a whole or a part of it. BMA attempts to find the best match block in the previous frame. The relative position between the block under consideration in the current frame and the best match block in the previous frame gives the vector \vec{v} . The matching criterion that is used mostly is the mean absolute difference (MAD). The advantage of using MAD is its computational simplicity and easy hardware implementation. It is given by

$$MAD(m, n) = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left| U^t(a+i, b+j) - U^{t-1}(a+i-m, b+j-n) \right| \quad (2)$$

where $U^t(x, y)$ is the pixel intensity at the position (x, y) in frame t , $M \times N$ is the block size, (a, b) is the position of the upper-left hand corner pixel in the block, (m, n) is the search location with respect to the center of the search window. The estimated motion vector, therefore, is given as

$$\vec{v} = (v_x, v_y) = (m, n) \left| \min_{(m,n) \in R} MAD(m, n) \right. \quad (3)$$

where R is the search window in the previous frame. Ideally, the search window R should be whole of the previous frame. However, a real world object can move only by a small amount in one frame time. Assuming that the displacement of an object from one frame to another can be at most p in either direction, i.e., $-p \leq v_x, v_y \leq +p$, the search window R is taken to be a square of size $(p+1) \times (p+1)$ with the current block position as its center. So, the number of MAD calculations is reduced significantly. Further reduction in the number of MAD calculations may be obtained by considering only a limited number of search locations in the search window. Some of these fast BMAs are 2D Logarithmic Search (2DLS) [13], 3-Step Hierarchical Search (3SHS) [14], One-at-a-Time Search (OTS) [15], 4-Step Hierarchical Search (4SHS) [16], 1D Hierarchical Search (1DHS) [17], Diamond Search [18], etc.

Affine motion model

Affine transformation may be incorporated along with the search schemes to predict the motion of an object more accurately. Affine motion model incorporates planar rotation and scaling (due to zooming) in addition to translation. In this model, the coordinate of each pixel \mathbf{p}^t of an object in the current frame t is related to its corresponding position $\hat{\mathbf{p}}^{t-1}$ in the previous frame as

$$\mathbf{p}^t = \mathbf{A}\hat{\mathbf{p}}^{t-1} + \vec{v} \quad (4)$$

where matrix \mathbf{A} is the *deformation matrix* that includes scaling and/or rotation. Combination of deformation and translation gives *affine transformation* and the pair (\mathbf{A}, \vec{v}) denotes the *affine motion*.

3. PROPOSED ALGORITHM

The aim of our proposed method of motion estimation is to estimate the affine motion by which an object is transformed from one frame to next. This may be accomplished by estimating the deformation matrix \mathbf{A} and the translation vector \vec{v} in eqn. (4).

Eqn. (4) may be rewritten as

$$\begin{aligned} \hat{\mathbf{p}}^{t-1} &= \mathbf{A}^{-1}(\mathbf{p}^t - \vec{v}) \\ &= \mathbf{B}\mathbf{p}^t - \vec{u} \end{aligned} \quad (5-a)$$

$$\text{or} \quad \mathbf{B}\mathbf{p}^t = \hat{\mathbf{p}}^{t-1} + \vec{u} \quad (5-b)$$

where $\mathbf{B} = \mathbf{A}^{-1}$ and $\vec{u} = \mathbf{A}^{-1}\vec{v}$. The pair $(\mathbf{B}, -\vec{u})$ may be interpreted as *backward affine motion* by which an object in the current frame is transformed back to the corresponding object in the previous frame. As we see, the necessary affine motion (\mathbf{A}, \vec{v}) can be easily com-

puted from the backward affine motion $(\mathbf{B}, -\vec{u})$ as

$$\mathbf{A} = \mathbf{B}^{-1} \quad (6-a)$$

$$\vec{v} = \mathbf{B}^{-1}\vec{u} = \mathbf{A}\vec{u} \quad (6-b)$$

In our proposed method, we aim at estimating this backward affine motion. Comparing eqn. (5-b) with eqn. (1), we see that for a given deformation \mathbf{B} applied to a block in the current frame, the problem of finding \vec{u} is nothing but the conventional block-matching algorithm. It follows that object-based affine motion estimation can be accomplished by deforming the object (or a part of the object) under consideration in the present frame by some deformation matrix \mathbf{B} and then matched with some region (of same dimension) in the previous frame. This is discussed in the subsection to follow.

Steps of the algorithm

Step 1: For a given video sequence, take the segmented objects and form VOPs. Thus, we have different VOP sequences corresponding to different objects in the scene.

Step 2: Construct BAPs (Binary Alpha Planes) from the VOPs by placing 1s for object pixels and 0s for background pixels.

Step 3: Each of the sequences corresponding to a video object is coded using a motion coder. The motion coder uses macro-block and block motion estimation similar to H.263 and MPEG-1/2 but modified to work with arbitrary shapes. In our algorithm, we propose to incorporate the affine motion model in the existing motion coder. This is described below.

Step 4: Apply some known deformation \mathbf{B}' on a macro-block.

Step 5: Search for a block in the previous frame that best matches the current block deformed by \mathbf{B}' . This may be accomplished by the conventional BMA and using MAD as the matching criterion. The relative location of the matched block gives the backward translation vector $-\vec{u}'$ corresponding to the deformation \mathbf{B}' .

Step 6: Repeat the above two steps for different possible combinations of deformation.

Step 7: The estimated backward affine motion $(\mathbf{B}, -\vec{u})$ is the deformation and its corresponding translation vector for which the MAD is the least.

Step 8: Compute the required affine motion parameters using eqns. (6-a) and (6-b).

Fast affine motion search

As with the conventional BMA, the number of searches in Step 5 above may also be restricted to some fixed locations in the search region. Accordingly, the search points

corresponding to any fast BMA, e.g., 2DLS, 3SHS, OTS, 4SHS, 1DHS, Diamond Search, etc., may only be considered. Similarly, we may apply only a few convenient deformations on the blocks out of the unlimited number of possible deformations. In our algorithm, we propose to use a set of such deformations which are suitable for application on a block; they are such that a square block is oriented in a manner so as to result in a square block only. These are rotations by 0° , 90° , 180° and 270° , horizontal flip, vertical flip and diagonal-flips. Also, assuming that the object can zoom only by a small amount in one frame time, a scaling factor of small value (both up-scaling and down-scaling) is enough. Accordingly, we have considered scaling factors as 0.5, 1 and 2 only.

4. EXPERIMENTAL RESULTS

In our experiments we demonstrate the performance of the proposed object-based affine motion estimation algorithm in comparison to the conventional BMA that is used in the standard MPEG-4 object-based video coding. Three video sequences used for the experiments are ‘‘Stefan’’, ‘‘Child’’ and ‘‘Football’’, each of frame-size $352 \text{ pixels} \times 240 \text{ lines}$. The performance index that we use is the prediction quality, that is how good is the current frame predicted from the previous frame. It is expected that for more accurate motion estimation the frame prediction will be better resulting in high quality motion compensated video frames. The quality of the motion compensated sequence is measured in terms of PSNR (dB) averaged over 50 frames of the sequences.

In a first set of experiments, the performance of our proposed algorithm is compared to that of the conventional Full-search (exhaustive search) block-matching algorithm. The search window taken is of size 15×15 . The search points considered in our algorithm are also same as that of the exhaustive search, that is all the 225 search locations in the search window are considered for matching. The macro-block size is taken to be 8×8 . The comparative results are given in Table 1. In a second set of experiments, the above is repeated with macro-block size 16×16 and the results are tabulated in Table 2.

In the third and fourth sets of experiments, the performance of our proposed algorithm is compared to that of the popular 3SHS block-matching algorithm. The search window is again taken to be of size 15×15 . In both these cases, the search points considered in our algorithm are the same 25 search locations as that of the 3SHS algorithm. The macro-block size are taken to be 8×8 and 16×16 and the results are given in Tables 3 and 4, respectively.

We see from the results obtained in our experiments that the reconstructed video sequences after motion compensation is always better with the application of our pro-

Table 1. Performance comparison between Full-search and our algorithm with 8×8 macro-blocks

Video Sequence	Exhaustive Search	Proposed Algorithm
Stefan	23.9317	31.5068
Child	30.6618	38.0876
Football	24.9019	26.6002

Table 2. Performance comparison between Full-search and our algorithm with 16×16 macro-blocks

Video Sequence	Exhaustive Search	Proposed Algorithm
Stefan	23.0689	25.3241
Child	28.8861	30.5036
Football	22.8410	23.8723

Table 3. Performance comparison between 3SHS and our algorithm with 8×8 macro-blocks

Video Sequence	Three Step Search	Proposed Algorithm
Stefan	18.6063	22.5978
Child	26.6904	28.6410
Football	20.8133	23.1791

Table 4. Performance comparison between 3SHS and our algorithm with 16×16 macro-blocks

Video Sequence	Three Step Search	Proposed Algorithm
Stefan	18.0982	20.0084
Child	26.4592	26.9407
Football	19.7765	21.2162

posed affine motion estimation algorithm than with the application of the two conventional block-matching techniques, viz., Full-search and 3SHS. Since in each set of experiments the search points and other parameters are same, this may be attributed to our motion estimation technique only which means our affine motion estimation algorithm is capable of tracking object motion better than the conventional motion estimation techniques. Therefore, we can safely conclude that our proposed object-based video coding scheme is superior to the standard object-based coding as used in MPEG-4.

5. CONCLUSION

An efficient affine motion estimation algorithm suitable for object-based motion estimation is presented. In se-

quences where complex motion is involved, frame prediction on the basis of affine motion estimation is more suitable than that involving estimation of translational motion only. As we see from the experimental results, our proposed algorithm performs better than the standard MPEG-4 object-based video coding that uses conventional block-matching algorithms and hence proves to be effective in estimating affine motion. The proposed motion estimation algorithm takes care of object motion due to translation, rotation and zooming. This in turn results in high quality video coding. Therefore, the object-based affine motion estimation algorithm proposed in this paper may be used efficiently in MPEG-4 video coding to bring about better quality. However, the computational complexity involved in this proposed algorithm is comparatively very high and so may not be good for real-time applications. We may overcome this limitation if a dedicated architecture is developed for the purpose.

REFERENCES

- [1] R. Schafer, and T. Sikora, "Digital Video Coding Standards and Their Role in Video Communications," *Proc. IEEE*, **83**(6), 907-923, 1995.
- [2] T. Sikora, "MPEG-4 Very Low Bit Rate Video," *Proc. IEEE ISCAS Conf.*, 1997.
- [3] S.-H. Lee, D.-H. Choi, and C.-S. Hwang, "Error Concealment Using Affine Transformation for H.263 Coded Video Transmission," *Electronics Letters*, **37**(4), 218-220, 2001.
- [4] H. Brusewitz, "Motion Compensation with Triangles," *Proc. Third Intl. Workshop on 64kbits/s Coding of Moving Video*, 1990.
- [5] H. Li, and R. Forchheimer, "A New Motion-compensated Technique for Video Compression," *Proc. IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP'93)*, **5**, 441-446, 1993.
- [6] X. Marrichal, and B. Macq, "Active Mesh Reconstruction of Block Based Motion Information," *Proc. IEEE Intl. Conf. Acoustics, Speech and Signal Processing (ICASSP'99)*, 2605-2208, 1999.
- [7] Y. Wang, and O. Lee, "Active Mesh – A Feature Seeking and Tracking Image Sequence Representation Scheme," *IEEE Trans. Image Processing*, **3**, 610-624, 1994.
- [8] C. L. Huang, and C. Y. Hsu, "A New Motion Compensation Method for Image Sequence Coding Using Hierarchical Grid Interpolation," *IEEE Trans. Circuits and Systems on Video Technology*, **4**, 42-52, 1994.
- [9] U. V. Raghavendranadh Reddy, D. Ghosh, and I. Chakrabarti, "Block-based Affine Motion Estimation," *Proc. Intl. Conf. Imaging Science, Systems, and Technology (CISST'02)*, **2**, 819-824, 2002.
- [10] J. O. Limb, and J. A. Murphy, "Measuring The Speed of Moving Objects from Television Signals," *IEEE Trans. Communications*, **23**(4), 474-478, 1975.
- [11] A. N. Netravali, and J. D. Robbins, "Motion-compensated Television Coding: Part I," *Bell Syst. Tech. J.*, **58**(3), 631-670, 1979.
- [12] B. Furht, J. Greenberg, and R. Westwater, *Motion Estimation Algorithms for Video Compression* Kluwer Academic, 1997.
- [13] J. R. Jain, and A. K. Jain, "Displacement Measurement and Its Applications in Interframe Image Coding," *IEEE Trans. Communication*, **29**(12), 1799-1808, 1981.
- [14] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated Interframe Coding for Video Conferencing," *Proc. Natl. Telecommunication Conf.*, G5.3.1-G5.3.5, 1981.
- [15] R. Srinivasan, and K. R. Rao, "Predictive Coding Based on Efficient Motion Estimation," *IEEE Trans. Communication*, **33**(8), 888-896, 1985.
- [16] M. L. Po, and W. C. Ma, "A Novel Four-step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. Circuits and Systems on Video Technology*, **6**(3), 313-317, 1996.
- [17] D. Ghosh, and A. P. Shivaprasad, "One Dimensional Hierarchical Search for Block Matching Algorithm in Motion Estimation," *Proc. Third Intl. Electronic Engineering Congress (INTERCON'96)*, 1996.
- [18] S. Zhu, and K. K. Ma, "A New Diamond Search Algorithm for Fast Block Matching Motion Estimation," *Proc. Intl. Conf. Information, Communications and Signal Processing*, **1**, 292-296, 1997.