Moving object tracking in H.264/AVC bitstream

Wonsang You, Houari Sabirin, Munchurl Kim

Moving Object Tracking in H.264/AVC Bitstream

Wonsang You, M.S. Houari Sabirin, and Munchurl Kim

School of Engineering, Information and Communications University,
Munjiro 119, Daejeon, 305732, Republic of Korea
{wsyou, houari, mkim}@icu.ac.kr

Abstract. Data broadcasting services are required to provide user interactivity through connecting additional contents such as object information to audio-visual contents. H.264/AVC-based metadata authoring tools include functions which identify and track position and motion of objects. In this work, we propose a method for tracking the target object by using partially decoded texture data and motion vectors extracted directly from H.264/AVC bitstream. This method achieves low computational complexity and high performance through the dissimilarity energy minimization algorithm which tracks feature points adaptively according to these characteristics. The experiment has shown that the proposed method had high performance with fast processing time.

Keywords: Object Tracking, H.264/AVC, Dynamic Programming, Neural Network.

1 Introduction

H.264/AVC does not handle video objects directly while MPEG-4 contains object-based encoding scheme. However, the interactive broadcasting service should provide additional object information as the form of MPEG-7 metadata to support user interactivity. The metadata authoring tool includes object tracking function which generates the position information of the predefined target object in all frames.

Unlike pixel-based object tracking approaches, object tracking algorithms for H.264/AVC videos can achieve lower computational complexity by using block-based motion vectors or residual data extracted directly from encoded bitstream; these are called compressed domain approaches. One difficulty in these approaches is that motion vectors do not always coincide with the true motion or optical flow.

To overcome the above difficulty, many researchers have proposed a variety of object tracking algorithms for MPEG videos. These can be classified as two categories as the motion-based method and the residual-based method. The motion-based methods rely on the probabilistic properties of the motion vector field. Babu et al. [1] predicted the motion of objects corresponding to affine motion parameters which are computed by the expectation maximization (EM) algorithm. Treetasanatavorn et al. [2] applied the Bayesian method to separate the significant foreground object from the background given the motion vector field. Zeng et al. [3] assigned the object label to blocks with homogeneous motion through the Markovian labeling

N. Sebe, Y. Liu, and Y. Zhuang (Eds.): MCAM 2007, LNCS 4577, pp. 483–492, 2007.

procedure. On the other hand, the residual-based methods use the statistical properties of DCT coefficients such as histogram. Aggarwal et al. [4] found the target object by histogram matching and motion interpolation. However, these algorithms have great tracking error in long image sequences due to block-based information. Moreover, motion-based methods tend to have high computational complexity or low performance in cases of deformable objects. Also, the residual-based methods are not applicable for H.264/AVC videos since residual data of intra-coded blocks is transformed from spatially intra-predicted values instead of original pixel values.

In this paper, we propose the dissimilarity energy minimization algorithm which uses motion vectors and partially decoded luminance signals to perform tracking adaptively according to properties of the target object in H.264/AVC videos. It is one of the feature-based approaches that tracks some feature points selected by a user. First, it roughly predicts the position of each feature point using motion vectors extracted from H.264/AVC bitstream. Then, it finds out the best position inside the given search region by considering three clues such as texture, form, and motion dissimilarity energies. Since just neighborhood regions of feature points are partially decoded to compute this energy, the computational complexity is scarcely increased. The set of the best positions of feature points in each frame is selected to minimize the total dissimilarity energy by dynamic programming. Also, weight factors for dissimilarity energies are adaptively updated by the neural network.

This paper is organized as follows. First, we describe the proposed object tracking algorithm at Section 2 and 3. Then, experimental results are presented in Section 4 and finally the conclusion is drawn in Section 5.

2 Forward Mapping of Backward Motion Vectors

The motion vectors extracted directly from H.264/AVC bitstream can be used to predict roughly the motion of feature points. Since all motion vectors in P-frames have backward direction, it should be changed to have forward direction. Following Porikli and Sun [5], the forward motion field is built by the region-matching method. First, motion vectors of blocks with various sizes are dispersed to 4x4 unit blocks. After each block is projected to the previous frame, the set of overlapping blocks is extracted as shown at Fig. 1.

Forward motion vectors of overlapped blocks in the previous frame are updated with respect to the ratio of the overlapping area to the whole block area. Assuming

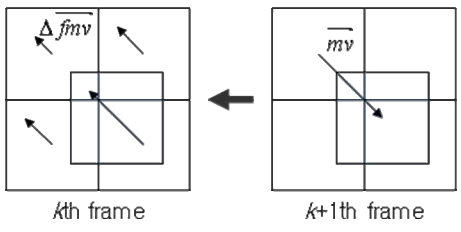

Fig. 1. The region-matching method for constructing the forward motion field

that the *j*th 4x4 block $b_{k,j}$ in the *k*th frame is overlapped with the *i*th 4x4 block $b_{k-1,i}$ in the *k-1*th frame, the forward motion vector $fmv_{k-1}(b_{k-1,i})$ is given by

$$\overrightarrow{fmv}_{k-1}\left(b_{k-1,i}\right) = -\sum_{j=1}^{N}\left(\frac{S_{k-1}(i,j)}{16}\cdot\overrightarrow{mv}_k\left(b_{k,j}\right)\right) \tag{1}$$

where $S_{k-1}(i,j)$ stands for the overlapping area between $b_{k,j}$ and $b_{k-1,i}$, and $mv_k(b_{k,j})$ denotes the backward motion vector of $b_{k,j}$ with $i,j=1,2,\ldots,N$. We assume that H.264/AVC videos are encoded in the baseline profile which each GOP contains just one I-frame and several P-frames. It should be noticed that the above region-matching method cannot be applied in the last P-frame in one GOP since the next I-frame does not have backward motion vectors. Assuming that the motion of each block is approximately constant within a small time interval, the forward motion vector of any block in the last P-frame can be assigned as a vector with the reverse direction of the backward motion vector as expressed by

$$\overrightarrow{fmv}_{k-1}\left(b_{k-1,i}\right) = -\overrightarrow{mv}_{k-1}\left(b_{k-1,i}\right). \tag{2}$$

Thereafter, positions of feature points in the next frame are predicted using forward motion vectors. If the *n*th feature point in the *k-1*th frame has the displacement vector $f_{k-1,n}=(fx_{k-1,n},fy_{k-1,n})$ and is included in the *i*th block $b_{k-1,i}$, the predicted displacement vector $p_{k,n}=(px_{k,n},py_{k,n})$ in the *k*th frame is defined as

$$\vec{p}_{k,n} = \vec{f}_{k-1,n} + \overrightarrow{fmv}_{k-1}\left(b_{k-1,i}\right). \tag{3}$$

3 Moving Object Tracking in H.264/AVC Bitstream

Since the predicted position of any feature point is not precise, we need the process of searching the best position of any feature point inside the search region centered at the predicted position $p_{k,n}= (px_{k,n},py_{k,n})$. It is checked whether each candidate point inside the search region is the best position using the dissimilarity energies related to texture, form, and motion. The set of candidate points with the minimum total dissimilarity energy is selected as the optimal configuration of feature points.

3.1 Texture Dissimilarity Energy

The similarity of texture means how the luminance property in neighborhood of a candidate point is similar with that in the previous frame. The set of candidate points inside the square search region is denoted as $C_{k,n}=\{c_{k,n}(1), c_{k,n}(2),\ldots, c_{k,n}(L)\}$ with $L=(2M+1)\times(2M+1)$ in the case of the *n*th feature point in the *k*th frame. Then, the texture dissimilarity energy E_C for the *i*th candidate point $c_{k,n}(i)=(cx_{k,n}(i),cy_{k,n}(i))$ is defined as

$$E_C\left(k;n,i\right) = \frac{1}{(2W+1)^2}\sum_{x=-W}^{W}\sum_{y=-W}^{W}\left|s_k\left(x+cx_{k,n}(i),y+cy_{k,n}(i)\right)-s_{k-1}\left(x+cx_{k,n}(i),y+cy_{k,n}(i)\right)\right| \tag{4}$$

where $s_k(x,y)$ stands for the luminance value in a pixel (x,y) of the *k*th frame, and *W* is the maximum half interval of neighborhood. The smaller E_C is, the more the texture of its neighborhood is similar with that of the corresponding feature point in the previous

frame. This energy forces the best point to be decided as the position with the most plausible neighbor texture as far as possible. Fig. 2 shows how the search region and the neighborhood of a candidate point are applied to calculate E_C.

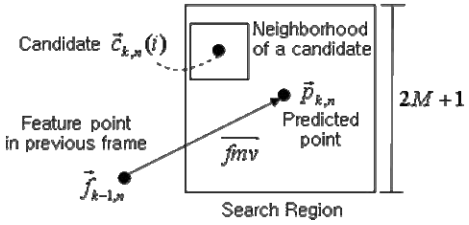

Fig. 2. The search region is centered at the predicted point located by a forward motion vector. A candidate point inside the search region has its neighborhood of square form to compute E_C.

Only necessary blocks can be partially decoded in P-frames to reduce the computational complexity. On the other hand, intra-coded blocks are impossible to be partially decoded since these are spatially intra-coded from these neighbor blocks.

General partial decoding takes long time since decoding particular blocks in P-frames requires many reference blocks to be decoded in the previous frames. We can predict decoded blocks to reduce the computation time. To predict decoded blocks in the *k*th P-frame, we assume that the velocity inside one GOP is as uniform as the forward motion vector of the *k-2*th frame. For the *i*th frame with $i=k,k+1,\ldots,K$, the predicted search region $P_{k,n}(i)$ is defined as the set of pixels which are necessary to calculate the texture dissimilarity energies of all possible candidate points for the *n*th feature point. Then, the half maximum interval $T_{k,i}$ of $P_{k,n}(i)$ is $T_{k,i}=(i-k+1)\times M+W+\gamma$ where γ denotes the prediction error. Then, $P_{k,n}(i)$ is given as follows:

$$P_{k,n}(i)=\left\{\vec{p}\,\middle|\,\vec{p}=(i-k+1)\overrightarrow{fmv}_{k-2}\left(b\left(\vec{f}_{k-2,n}\right)\right)+\vec{m}+\vec{f}_{k-1,n},\vec{m}=(x_m,y_m);x_m,y_m=-T_{k,i},\ldots,T_{k,i}\right\} \tag{5}$$

where $b(f_{k-2,n})$ stands for the block which includes the *n*th feature point $f_{k-2,n}$. The decoded block set $D_{k,n}(i)$ is defined as the set of blocks which should be decoded to reconstruct $P_{k,n}(i)$. Using the motion vector of the *k-1*th frame, $D_{k,n}(i)$ is given by

$$D_{k,n}(i)=\left\{b\left(\vec{d}\right)\middle|\,\vec{d}=(i-k)\overrightarrow{mv}_{k-1}\left(b\left(\vec{f}_{k-1,n}\right)\right)+\vec{p},\vec{p}\in P_{k,n}(i)\right\}. \tag{6}$$

Assuming that there exist *F* feature points, the total decoded block set D_k in the *k*th frame can be finally computed as

$$D_k=\bigcup_{n=1}^{F}\bigcup_{i=k}^{K}D_{k,n}(i). \tag{7}$$

Fig. 3 shows how partial decoding is performed in the first P-frame of one GOP which contains one I-frame and three P-frames. It should be noticed that the time for calculating the total decoded block set is proportional to the GOP size.

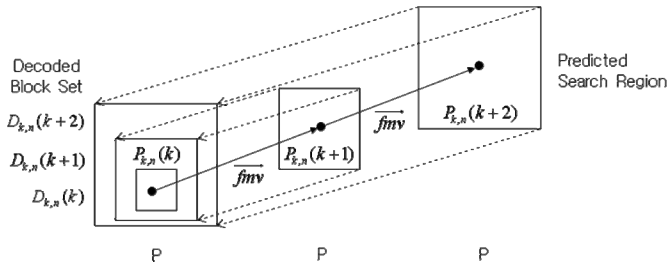

Fig. 3. The structure of partial decoding in the first P-frame of a GOP which contains one I-frame and three P-frames. Two decoded block sets $D_{k,n}(k+1)$ and $D_{k,n}(k+2)$ in the first P-frame are projected from two predicted search regions $P_{k,n}(k+1)$ and $P_{k,n}(k+2)$.

3.2 Form Dissimilarity Energy

The similarity of form means how the network of candidate points is similar with the network of feature points in the previous frame. Each feature point is jointly linked by a straight line like Fig. 4. After a feature point is initially selected, it is connected to the closest one among non-linked feature points. In this way, the feature network in the first frame is built by connecting all feature points successively.

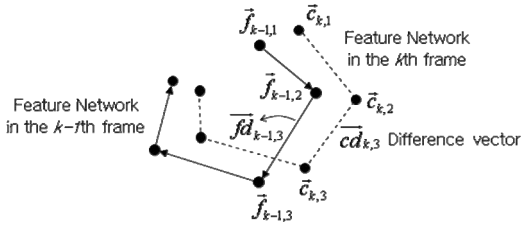

Fig. 4. The network of feature points in the previous frame and the network of candidate points in the current frame

To calculate the form dissimilarity energy of each candidate point, we assume that each feature point is arranged in the order named at the first frame. The feature point $f_{k-1,n}$ in the k-1th frame has its difference vector $fd_{k-1,n}(i)=f_{k-1,n}(i)-f_{k-1,n-1}(i)$ as shown at Fig. 4. Likewise, the ith candidate point of the nth feature point in the kth frame has its difference vector $cd_{k,n}(i)=c_{k,n}(i)-c_{k,n-1}(j)$. Then, the form dissimilarity energy E_F for the ith candidate point of the nth feature point (n>0) is defined as follows:

$$E_F(k;n,i)=\left\|\overrightarrow{cd}_{k,n}(i)-\overrightarrow{fd}_{k-1,n}\right\|^{1/2}. \tag{8}$$

All candidate points of the first feature point (n=0) have zero form dissimilarity energy $E_F(k;0,i)=0$. The smaller E_F is, the less the form of the feature network will be transformed. The form dissimilarity energy forces the best position of a candidate point to be decided as the position where the form of the feature network is less changed as far as possible.

3.3 Motion Dissimilarity Energy

The reliability of a forward motion vector means how it is similar with true motion enough to get a predicted point as exactly as possible. Following Fu et al. [6], if the predicted point $p_{k,n}$ which has located by the forward motion vector fmv_{k-1} returns to its original location in the previous frame by the backward motion vector mv_k, fmv_{k-1} is highly reliable. Assuming that $p_{k,n}$ is included to the jth block $b_{k,j}$, the reliability R can be given as follows:

$$R(\vec{p}_{k,n})=\exp\left(-\frac{\left\|\overrightarrow{fmv}_{k-1}(b_{k-1,i})+\overrightarrow{mv}_k(b_{k,j})\right\|^2}{2\sigma^2}\right) \tag{9}$$

where σis the variance of reliability. Fig. 5 shows forward motion vectors with high and low reliability. In a similar way of Fu's definition [6], the motion dissimilarity energy E_M for the ith candidate point is defined as follows:

$$E_M(k;n,i)=R(\vec{p}_{k,n})\left\|\vec{c}_{k,n}(i)-\vec{p}_{k,n}\right\|. \tag{10}$$

With high reliability R, E_M has greater effect on finding the best point than E_C or E_F since it is sharply varying according to the distance between a predicted point and a candidate point.

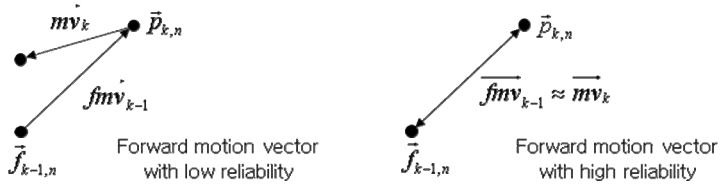

Fig. 5. The reliability of forward motion vectors. The great gap between a forward motion vector and a backward motion vector results in low reliability.

3.4 Energy Minimization

The dissimilarity energy $E_{k,n}(i)$ for the ith candidate point of the nth feature point is defined as follows:

$$E_{k,n}(i)=\omega_C(k)E_C(k;n,i)+\omega_F(k)E_F(k;n,i)+\omega_M(k)E_M(k;n,i) \tag{11}$$

where $w_C(k)$, $w_F(k)$, and w_M(k) are weight factors for texture, form, and motion dissimilarity energy. If the configuration of candidate points is denoted as $I=\{c_{k,1}(i_1), c_{k,2}(i_2),\ldots,c_{k,F}(i_F)\}$, the optimal configuration $I_{opt}(k)$ in the kth frame is selected as what minimizes the total dissimilarity energy $E_k(I)$ expressed by

$$E_k(I)=\sum_{n=1}^{F} E_{k,n}(i_n). \tag{12}$$

When all possible configurations of candidate points are considered, it takes so much time $\Theta((2M+1)^{2F})$ that causes high computation complexity especially in cases of large search region or many feature points. We can reduce the amount of computations by $\Theta(F)$ using the discrete multistage decision process called the dynamic programming which corresponds to two steps [7]:

1) The accumulated dissimilarity energy (ADE) $E_{local}(n,i)$ for the ith candidate point of the nth feature point ($n>0$) is calculated as follows:

$$E_{local}(n,i)=\min_{j}\left[E_{k,n}(i,j)+E_{local}(n-1,j)\right]. \tag{13}$$

The ADE for the first feature point is $E_{local}(0,i)=E_{k,0}(i)$. Then, the point which minimizes the ADE is selected among candidate points of the n-1th feature point; the index of this point is saved as

$$s(n,i)=\arg\min_{j}\left[E_{k,n}(i,j)+E_{local}(n-1,j)\right]. \tag{14}$$

2) In the last feature point, the candidate point with the smallest ADE is selected as the best point o_F. Then, the best point o_n for the nth feature point is heuristically decided as follows:

$$o_F=\arg\min_{i}\left[E_{local}(F,i)\right] \text{ and } o_n=s(n+1,o_{n+1}). \tag{15}$$

The best position for nth feature point $f_{k,n}$ is $f_{k,n}=c_{k,n}(o_n)$.

3.5 Adaptive Weight Factors

The arbitrarily assigned weight factors for texture, form, and motion dissimilarity energy can give rise to tracking error since the target object can have various properties. In this reason, weight factors need to be decided adaptively according to properties of the target object. For instance, for an object which texture is scarcely changing, the weight factor w_C should be automatically set up as high value.

Weight factors can be automatically updated in each frame by using the neural network as shown in Fig. 6. The dissimilarity energy E_k is transformed to its output value $\dot{E}_k$ by the nonlinear activation function ξ. The update of weight factors is performed by the backpropagation algorithm which minimizes the square output error ε_k defined as follows:

$$\varepsilon_k = \frac{1}{2}\left(E_d-\dot{E}_k\right)^2. \tag{16}$$

where E_d denotes the ideal output value. If the activation function ξ is the unipolar sigmoidal function ($\xi(x)=1/(1+e^{-x})$), the gradient of a weight factor is calculated as

$$\Delta\omega_x(k) = \eta\left(E_d-\dot{E}_k\right)\dot{E}_k\left(1-\dot{E}_k\right)E_x(k) \tag{17}$$

where x can be T (texture), F (form), or M (motion), and η is the learning constant [8].

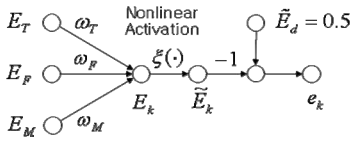

Fig. 6. The neural network for updating weight factors

4 Experimental Results

To demonstrate the performance of the proposed method, the tracking results of various objects have extracted from videos such as "Stefan", "Coastguard" and "Lovers" with CIF size. Each video was encoded as the GOP structure of 'IPPP' in the baseline profile, and included P-frames whose previous frame only can be a reference frame. Fig. 7(a) shows the tracking results of a rigid object with slow motion in "Coastguard". Four feature points were well tracked in the uniform form of feature network. Fig. 9(b) also shows the tracking result of a deformable object with fast motion in "Stefan". We can observe that tracking is successful even though the form of feature network is greatly changing due to fast three-dimensional motion.

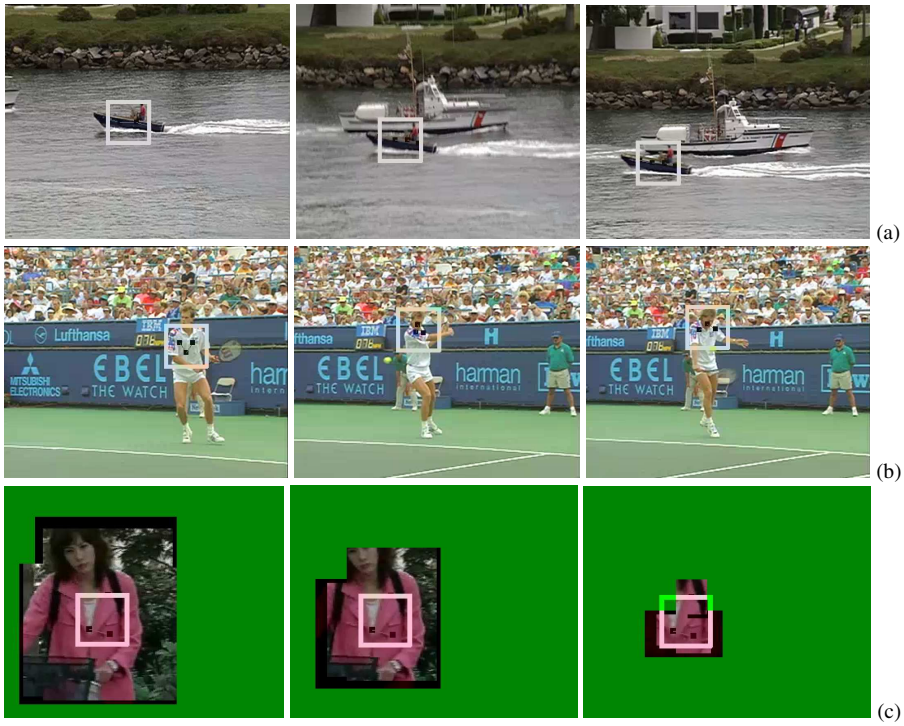

Fig. 7. The object tracking in (a) "Coastguard", (b) "Stefan" with 100 frames, and (c) "Lovers" with 300 frames. Partially decoded regions are shown in "Lovers".

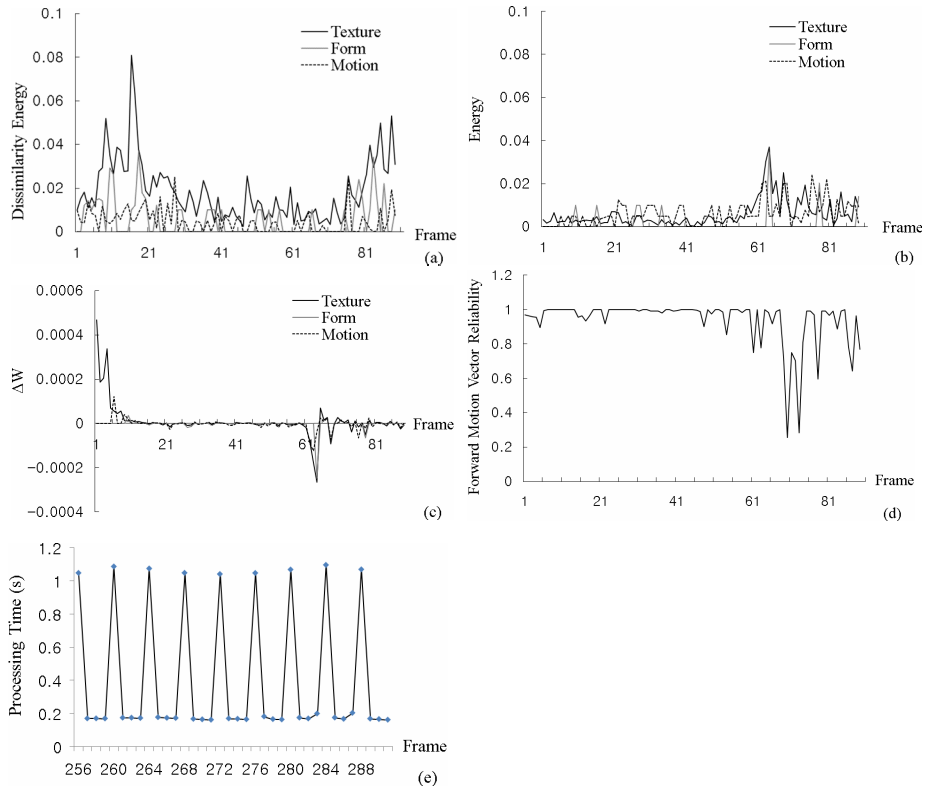

Fig. 8. (a) Dissimilarity energies in "Stefan" and (b) "Coastguard", (c) the variation of weight factors, (d) the average reliability of *fmv* in "Coastguard", (e) the processing time in "Lovers"

Fig. 7(c) represents the visual results of partial decoding in P-frames of "Lovers" when the search half interval M and the neighborhood half interval W are assigned as 10 and 5. Only the neighborhood region of three feature points was partially decoded. Even in a sequence "Lovers" with 300 frames, no tracking errors were found.

Numerical data of tracking from two video samples is shown at Fig. 8. In Fig. 8(a) and (b), dissimilarity energies in "Coastguard" are lower than those in "Stefan". We can see from this result that the variation of texture, form, and motion in "Coastguard" is smaller than "Stefan". Fig. 8(d) shows a plot for the average reliability of forward motion vectors in "Coastguard". The average percentage of reliabilities in "Coastguard" is 93.9% higher than 12.2% in "Stefan"; it indicates that the motion dissimilarity energy is the good measure for the motion property of the target object.

Through the neural network, the square error of dissimilarity energy is minimized. When the learning constant was equal to 5, this error had approximately zero value after the 15th frame. Moreover, weight factors converge on optimal values as shown at Fig. 8(c). We can observe that weight factor variations and dissimilarity energies increase greatly from the 61th frame to the 66th frame in "Coastguard"; it illustrates that weight factors are adaptively controlled when another ship is approaching.

When the JM reference software was used to read H.264/AVC bitstream, the computation time was about 430ms/frame (Intel Pentium 4 CPU 3.2GHz 1GB RAM). As shown at Fig. 8(e), most computations are performed at I-frames which are fully decoded. In this reason, if faster decoder is used, we can reduce the computation time by 230ms/frame maximally. The computation time is nearly similar with Zeng's algorithm [3] which computation time is roughly 450ms/frame in "PIE" sequence. However, Zeng's algorithm cannot track a target object identified by a user since it extracts all moving objects. In order to select the target object from all segmented objects, it requires more computation. Therefore, the proposed algorithm can track the target object in real-time applications with faster speed than Zeng's algorithm.

5 Conclusion

We have proposed a novel object tracking algorithm with low computational complexity and high performance. It finds the best positions of feature points which have high similarity in texture, form, and motion. Moreover, the computational complexity can be reduced by the partial decoding and the dynamic programming for optimal energy minimization. Also, main parameters are adaptively optimized according to properties of the target object. We demonstrated that the proposed algorithm can track precisely deformable objects or fast-moving objects in small computation time. It can be applied to the metadata authoring tool which generates the position information of the target object. In future work, we will study the automatic extraction of feature points using motion vectors in H.264/AVC bitstream.

References

1. Babu, R.V., Ramakrishnan, K.R.: Video Object Segmentation: A Compressed Domain Approach. IEEE Trans. Circuits Syst. Video Technol. 14, 462–474 (2004)
2. Treetasanatavorn, S., Rauschenbach, U., Heuer, J., Kaup, A.: Bayesian Method for Motion Segmentation and Tracking in Compressed Videos. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 277–284. Springer, Heidelberg (2005)
3. Zeng, W., Du, J., Gao, W., Huang, Q.: Robust moving object segmentation on H.264/AVC compressed video using the block-based MRF model. Real-Time Imaging 11(4), 290–299 (2005)
4. Aggarwal, A., Biswas, S., Singh, S., Sural, S., Majumdar, A.K.: Object Tracking Using Background Subtraction and Motion Estimation in MPEG Videos. In: Narayanan, P.J., Nayar, S.K., Shum, H.-Y. (eds.) ACCV 2006. LNCS, vol. 3852, pp. 121–130. Springer, Heidelberg (2006)
5. Porikli, F., Sun, H.: Compressed Domain Video Object Segmentation. Technical Report TR2005-040 of Mitsubishi Electric Research Lab (2005)
6. Fu, Y., Erdem, T., Tekalp, A.M.: Tracking Visible Boundary of Objects Using Occlusion Adaptive Motion Snake. IEEE Trans. Image Processing 9, 2051–2060 (2000)
7. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. MIT Press, Cambridge, MA (2001)
8. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons, New York (2001)