DCT-Based Object Tracking in Compressed Video

Lan Dong, Stuart C. Schwartz

Dept. of Electrical Engineering, Princeton University
E-Quad, Olden St., Princeton 08544, U.S.A.
E-mail: {ldong, stuart}@ princeton.edu

ABSTRACT

This paper presents a novel real-time DCT-based object tracking algorithm that operates on I frames in compressed MPEG video. Discrete Cosine Transform (DCT) coefficients that are provided from video sequences are exploited to find motion and color cues. The temporal motion and spatial color information are then fused by means of a posterior probability framework which allows the information from different measurement sources to be fused in a principled manner. The DCT-based method and probabilistic framework ensure robustness with respect to noise, occlusion, local scene changes, global illumination changes and inconsistent movement. The tracker is also able to recognize the object when it re-appears after it has left the scene. After tracker selection, the background and object model are selectively updated to adapt to environmental changes. Experimental results show the effectiveness and efficacy of the proposed method.

1. INTRODUCTION

With the dramatic advances in communications and multimedia computing technology, there is growing interest to support the capability to interact with a movie or video sequence by retrieving data of any kind related to a particular object. Among all salient object features, motion is probably the most effective feature used to provide a global understanding and description of the dynamic content of a video sequence.

Several approaches have been proposed in the literature to track objects in a video sequence to obtain motion information. The majority of the object trackers deal with decoded pixel data [1]-[3]. Some of these are clearly not suitable for real-time applications as they are computationally intensive. Since a great deal of video data is available in compressed form, research has also been directed at tracking in the compressed domain [4]. Among the cues available in the compressed domain, macroblock motion vectors are usually used [5], [6], since they are readily available and easy to use. But there are two main drawbacks of motion vectors: motion vectors available in the compressed data are purely for coding purposes so that their use for tracking is not always feasible [7]; motion vectors are only present in P and B frames but not in I frames. Consequently, error from motion vectors due to absence of any verification for the object being tracked (i.e., ground truth) is easily accumulated and results in loss of tracker. In order to guarantee robust tracking, the I frame, the reference for each GOP (Group of Pictures) is always used as an "anchor" frame to determine the position of the tracked object [5]-[7]. We call this I-rectification. Its validity is important since future tracking will follow from this reference frame. Most previous papers use template matching on the decoded I frame for I-rectification. However, it is a potentially computationally intensive task. In addition, a simple template matching method is not good enough in complex situations where there is noise, illumination change, occlusion or objects moving out of the scene.

In this paper, we explore the I-rectification as part of the compressed video tracking system. Our goal is to develop an I-rectification algorithm in the compressed domain, rather than utilize template matching in the pixel domain, which is robust under various environments and fast enough to run in real-time. The proposed DCT-based tracking algorithm uses posterior probability estimation which fuses temporal motion with spatial color information. It brings all the advantages of working in the transform domain: less coefficients to manipulate, more robust to illumination changes, more robust to noise. Combined with motion vector based tracking in P, B frames [5], robust and fast tracking in compressed video can be achieved.

Our algorithm is also applicable for videos with I frame only. For retrieving and indexing applications in extremely large databases of video, it is often feasible to perform tasks only on selected frames. Our algorithm is able to perform fast and robust tracking on selected I frames to retrieve motion information.

The rest of the paper is organized as follows. The description of the approach is discussed in Sec.2. The experimental results are presented in Sec.3, followed by Sec.4, with discussion and conclusions.

2. DESCRIPTION OF THE APPROACH

2.1. Overview

Given a MPEG-2 video clip as input, we perform tracking of an object which is manually selected at the first frame of the video. The space selected is called a tile and the object selected is called a reference. For each new I frame, by comparing, based only on DCT coefficients, the current frame with a background model, we can detect foreground blocks, which are then used to decide candidate object tiles. Among all object candidates, we find the position which maximizes the posterior probability based on color and motion and label it as the object we are tracking. Before beginning to process the next new frame, we do selective color adaptation for both the background and reference models. This is essential in case the background and reference are changing. The overall procedure is illustrated in Figure 1.

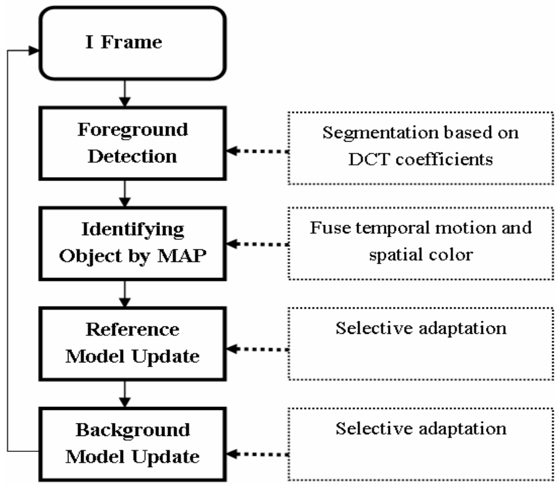


Figure 1. Block diagram of the system
(Right hand side is the method used for the task in the block)

2.2. Background Modelling and Foreground Detection

The first important step is to find the foreground and then candidate object tiles. Since we assume we are using a stationary camera, the detection of object foreground can be achieved by comparing each new frame with a representation of the scene background. The scene representation is called the background modelling. Block based, rather than pixel based modelling can take advantage of the spatial correlation of the pixels and is more robust to illumination change and noise [8]. As the I frame consists of 8x8 DCT coefficient blocks, adapting the transform domain segmentation approach developed in [8], two features are used for each block:

$$f_{DC} = DCT(0,0) \quad ,$$
$$f_{AC} = DCT(0,1)^2 + DCT(1,0)^2 + DCT(1,1)^2 \quad (1)$$

These two features are extracted from the DCT coefficients of the luminance component of each block. The DC feature gives information of the average intensity and the AC feature, the square sum of low frequency AC coefficients, gives information of the boundary or energy of the block. The high frequency coefficients of the block are not used since most of them are zero after quantization and they are also quite sensitive to noise. Manipulating only 4 DCT coefficients which are readily available in compressed video enables the algorithm to run fast.

We obtain a background feature from the first several static scenes by averaging their DCT coefficients. In the new frame, for each block, we will declare a foreground block by comparing with background model, if:

$$|\Delta f_{DC}| > th1 \text{ or } |\Delta f_{AC}| > th2 \quad (2)$$

If either threshold test is met, a block is declared in the foreground because of the fact that a foreground object is often different from the background either in intensity or energy or both. This test avoids labelling rippling water or waving leaves in the background scene since neither the DC nor AC feature, i.e. average intensity or energy, will change much in that block. This threshold test has good performance for being invariant to noise and small scene changes. A foreground label will also be used as a confidence measure when updating the reference model.

Then we develop a confidence level β to say that for a tile that does not contain enough foreground blocks, it is not likely to include the object. More specifically, we will declare a candidate tile if and only if that tile contains more than β percentage foreground blocks. In our experiments, we chose β as 50, which means if more than half of the blocks in a tile are foreground, we consider that tile a candidate tile which may contain the object. In addition, the tile located at the predicted position of the object is always considered as a candidate no matter how many foreground blocks it contains. This avoids the tracker getting lost when the object is occluded by the background.

2.3. Identifying Object by Data Fusion and MAP

In tracking problems, statistical methods, such as search window methods [7] and probabilistic methods, such as particle filter [2] can be used. Probabilistic methods are more reliable and robust in cases of clutter and partial occlusions. Even when the tracker gets lost, probabilistic methods are able to guide the tracker to the correct object. Another important advantage of using the probability is that it allows the information from different measurement sources to be fused in a principled manner. More specifically, we will introduce color as one cue and fuse it with motion activity cues.

Denote by $\mathbf{x}_i$ the tracking result of the i-th frame, $\mathbf{y}_i$ the measurements (i.e., color) at the i-th frame, and $\mathbf{c}_i$ the candidate state of i-th frame.

For tracking purposes, the probability of interest is the posterior probability $p(\mathbf{c}_i | \mathbf{y}_i, \mathbf{x}_{i-1})$, i.e., the probability of a particular object being in this candidate tile, given the position of the tracker in the last frame and the data (color) of the current frame. Once we find the posterior probability for all the candidate tiles, we perform the maximum a posteriori (MAP) estimate $\mathbf{x}_i = \arg\max_{\mathbf{c}_i} p(\mathbf{c}_i | \mathbf{y}_i, \mathbf{x}_{i-1})$ to determine the position of the object. We point out here that MAP, like the Kalman Filter, is suboptimal as it estimates the new state conditioned upon the correctness of the previous decision. Using Bayes rule, we have:

$$p(\mathbf{c}_i | \mathbf{y}_i, \mathbf{x}_{i-1}) = p(\mathbf{c}_i | \mathbf{x}_{i-1}) p(\mathbf{y}_i | \mathbf{c}_i, \mathbf{x}_{i-1}) / p(\mathbf{y}_i | \mathbf{x}_{i-1}) . (3)$$

The denominator is same for all candidates. Conditioned on $\mathbf{c}_i$, color $\mathbf{y}_i$ is independent of previous result $\mathbf{x}_{i-1}$. So the problem now is to find candidate position $\mathbf{c}_i$ such that $p(\mathbf{c}_i | \mathbf{x}_{i-1}) p(\mathbf{y}_i | \mathbf{c}_i)$ is maximized.

$p(\mathbf{y}_i | \mathbf{c}_i)$ is the "color likelihood" given the object is located at $\mathbf{c}_i$ and will be discussed in detail in the following section. $p(\mathbf{c}_i | \mathbf{x}_{i-1})$ is the "motion likelihood" designed to favor regions where the object is likely to appear. It is clear now the MAP method has combined the motion and color information together in a principled manner

In the majority of cases, the motion of the object in the image sequence will satisfy some temporal motion continuity constraints, so we can assume $p(\mathbf{c}_i | \mathbf{x}_{i-1})$ monotonically decreasing with the distance $\mathbf{x}_i^p - \mathbf{c}_i$, where $\mathbf{x}_i^p$ is a predicted position (linear prediction used in our experiment). A convenient density is the Gaussian with centre located at $\mathbf{x}_i^p$:

$$p(\mathbf{c}_i | \mathbf{x}_{i-1}) \propto \exp\left(-\frac{(\mathbf{c}_i - \mathbf{x}_i^p)}{2\sigma_m^2}\right) . \quad (4)$$

2.4. Spatial Color Information in Posterior Probability

Color information is a powerful feature to characterize the appearance of tracked objects since the color feature of an object or a region of interest often forms a very persistent localization cue [9].

The color of the reference tile is compared to the color of the candidate tile. The smaller the discrepancy between the candidate and reference models, the higher the probability that the object is located in the corresponding image region. This is quantified by "color likelihood" $p(\mathbf{y}_i \mid \mathbf{c}_i)$.

In MPEG-2, YUV (601) color space is used. Let $\mathbf{y}_i(\mathbf{p})$ denote the observed YUV color vector of pixel $\mathbf{p}$ in the candidate tile $\mathbf{c}_i$ and $\mathbf{g}(\mathbf{p})$ the reference YUV color vector of corresponding pixel $\mathbf{p}$. The residual vector between the template value and the observed data is $\mathbf{r}(\mathbf{p}) = \mathbf{y}_i(\mathbf{p}) - \mathbf{g}(\mathbf{p})$. The matching error at pixel $\mathbf{p}$ can be defined by the Mahalanobis distance: $\varepsilon(\mathbf{p}) = \sqrt{\mathbf{r}(\mathbf{p})^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{p})}$ where $\mathbf{R}$ is a covariance matrix of the residue. Furthermore, in order to make the matching robust against occlusion, we should down-weight residuals that are too large. This is achieved by using a robust error norm $\rho(\varepsilon)$ which is a Huber's function defined as:

$$\rho(\varepsilon) = \begin{cases} \varepsilon^2 / 2 & if \quad |\varepsilon| < c \\ c(|\varepsilon| - c/2) & otherwise \end{cases} \tag{5}$$

where c is the cut-off threshold.

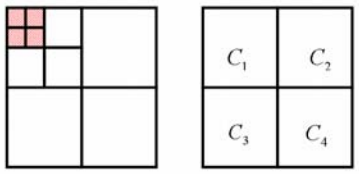
Having defined the matching error for each pixel, we define the color probability as:

$$p(\mathbf{y}_i \mid \mathbf{c}_i) = \exp(-k_r \sum_{\mathbf{p} \in Tile} \rho(\sqrt{\mathbf{r}(\mathbf{p})^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{p})})) \cdot \tag{6}$$

where coefficient k_r is used to balance the importance of color likelihood and motion likelihood.

To obtain pixel-wise color information from MPEG-2 video sequences is difficult. For DCT-based video compression, full decoding from DCT to the pixel domain will result in a large computation load. Sometimes, the video processing does not necessarily require full resolution pixel data. Our goal is to maintain good tracking performance and at the same time achieve computing efficiency so as to realize real time processing.

Partial decoding by Norhashimah, Fang and Jiang[10] is used in our algorithm to extract 2x2 "Super-Pixels" for a 8x8 block as color information. For each DCT block, upper left 4 DCT coefficients (Fig. 2 (a)) are manipulated to obtain 4 average color values of the "Super-Pixels" (Fig. 2 (b)). The algorithm in [10] helps to improve the speed of our algorithm since it uses only 8 additions, 4 right-shifting operations and no multiplications while full IDCT requires 96 multiplications and 466 additions for decompression of one single block with 8x8 pixels using the FFT.



(a) (b)

Figure 2. (a)DCT domain 8x8 block
(b)Pixel domain 8x8 block (2x2 Super-Pixels block)

The MAP method actually incorporates both prediction of the search window technique by utilizing $p(\mathbf{c}_i \mid \mathbf{x}_{i-1})$ and a color template matching technique by utilizing $p(\mathbf{y}_i \mid \mathbf{c}_i)$. But the probabilistic framework is superior in the case when the track has been temporarily lost, as may be caused by occlusion or the object leaving the scene. With this mechanism, the tracker can go back to the objects once it re-appears.

2.5. Selective Update of Reference and Background Model

Due to movement of the object and noise in the environment, some colors may be changed. To have robust tracking, both the reference model and the background must be updated.

For each block, the DCT coefficients of the reference and background are updated according to:

$$\mathbf{C}_i = (1 - \alpha_{R/B})\mathbf{C}_{i-1} + \alpha_{R/B}\mathbf{y}_i, \tag{7}$$

$\mathbf{C}_i$ is the coefficient used in the reference or background model. $\mathbf{y}_i$ denotes the block data of i-th frame. $\alpha_{R/B}$, either α_R or α_B, is the learning parameters controlling the learning speed. For stability reasons, we choose a smaller learning parameter α_B compared to α_R.

For the reference model, we do not update with the blocks where no foreground is found. That avoids the reference model being corrupted by background color, which is a common "drift" problem in tracking processes. For the background model, we do not update with the blocks where the tracker of this frame is located. This avoids a stationary object being considered as background and the tracker getting lost after several frames.

2.6. Sudden Change in Illumination

The above algorithm is robust to noise, occlusion, small scene changes and gradual illuminance change. Special attention must be paid to the occurrence of sudden illumination change.

A sudden global illumination change would result in a significant increase of blocks initially labelled as foreground. After we detect a great increase of blocks, in addition to f_{DC}, f_{AC}, we add one more feature $f_r = f_{DC} / \sqrt{f_{AC}}$, which is illumination invariant under the assumption that the illumination color is white and the diffuse refection is Lambertian, to segment the foreground.

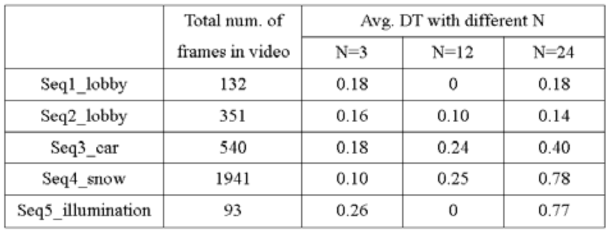
The observation that change in illumination under a white light assumption results in same scale of intensity of RGB color component was used in various tracking methods [11], [12]. Together with the fact that the DCT transform is linear, it is easy to see that the feature f_r is illumination invariant. Comparing this new feature value of blocks which are pre-labelled foreground with this feature value in the background model, we declare a true foreground block only if $|\Delta f_r| > th3$.

Under global illumination change, the algorithm is essentially the same with this new feature. There are two other minor differences: One is the color $\mathbf{y}_i(\mathbf{p})$ used for calculating "color likelihood" is now an illumination invariant one: U/Y, V/Y rather than the YUV value themself. The other one is we increase the learning parameter to speed up the adaptation of both object and background model.

3. EXPERIMENTAL RESULTS

To test the I-rectification algorithm, the tracker has been applied to track objects only in I frames on a variety of video sequences representing different difficulties in tracking tasks. The initial simulations are more than satisfactory and have been very encouraging.

We test our algorithm on different settings and the values for the parameters are fine-tuned to optimize the performance. For each setting, the video is encoded with 3 different numbers of frames in the GOP, i.e. the videos have different number of I frames. The tracking result is shown in Table 1 where the average distance (DT) of ground-truth object to tracker-submitted object is used as an indicator of the performance. (DT = 1 is 8 pixels as the tracker is block based. N = number of frames in a GOP.)



	Total num. of	Avg. DT with different N		
	frames in video	N=3	N=12	N=24
Seq1_lobby	132	0.18	0	0.18
Seq2_lobby	351	0.16	0.10	0.14
Seq3_car	540	0.18	0.24	0.40
Seq4_snow	1941	0.10	0.25	0.78
Seq5_illumination	93	0.26	0	0.77

Table 1. Tracking performance of different videos

Setting 1 is an indoor one with two people walking towards each other and then passing by one another. We track the occluded person. See figure 3 (a). Setting 2 has two people walk towards each other and stop for some time and then turn around. When the object stays still, it does not become part of the background because of our selective updating. See figure 3 (b). Setting 3 contains a moving car which disappears for about 100 frames and appears again at a place far from where it disappeared. The tracker jumps to the new location and keeps tracking from then on. There are also trees on the road. The tracker does not update the reference model with tree color when the object is occluded by the tree. Setting 4 has a river with running water. Though the color of the water is very similar to the object, we still succeed in tracking the object because our algorithm is robust to noise and small scene changes when segmenting the foreground. Setting 5 has global illumination change. During the tracking process, the new feature f_r plays an important role in segmenting the true foreground. After several frames, the object and background adapt to the new light condition. Because of space limits, only two results are shown here. More results can be accessed at:

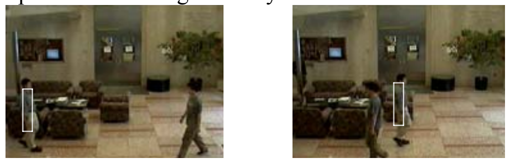
www.princeton.edu/~ldong/research/I_tracking/

On a 1 GHz PIII processor, our implementation can process about 35 I frames per second with the frame size of 240x320 pixels. The C++ program is not optimized in terms of speed.

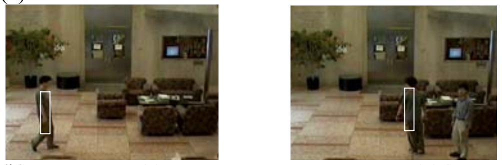
4. CONCLUSION AND FUTURE WORK

This paper presented a new DCT-based I frame tracking approach. After locating foreground and candidate tiles, we use color and fuse it with motion activity to estimate a posterior probability. Then a MAP estimate is performed to determine the position of the object. The algorithm also selectively updates the background and reference model, avoiding false updates. The algorithm in its present form achieves very good performance and can be implemented for real time operation.

There are problems to be considered. Objects can rarely be tracked sharply and with precision along their exact boundaries in the compressed domain due to the blocky nature of the data. Moreover, objects too small are hard to track. Among our future work are to use smaller sub-blocks in the tracking process in order to improve the tracking accuracy.



(a) Frame 12 Frame 72



(b) Frame 12 Frame 216

Figure 3. Two sample video sequence with tracking results

REFERENCE

[1] D. Comaniciu, V. Ramesh, P. Meer, "Real-Time Tracking of Non- Rigid Objects using Mean Shift", IEEE Proc. of CVPR,Vol. 2 ,pp.142 – 149,2000

[2] P.Perez, J.Vermaak, A.Blake, "Data Fusion for Visual Tracking with Particles", Proc. of the IEEE, Vol. 92-3, pp.495 – 513, 2004

[3] Y. Ren, C. Chua, Y. Ho, "Color based tracking by adaptive modeling", Proc. of ICARCV, Vol. 3, pp. 1597 - 1602, 2002

[4] S.F. Chang, "Compositing and manipulation of video signals for multimedia network video services", Ph.D. dissertation, Dept. EECS, Univ. of California Berkeley, 1993

[5] L.Favalli, A Mecocci, F. Moshetti,"Object tracking for retrieval applications in MPEG-2", IEEE Trans. Circuits and Systems for Video Technology,Vol.10, Issue 3,pp.427 – 432, 2000

[6] D. Schonfeld, D. Lelescu, "VORTEX: video retrieval and tracking from compressed multimedia databases", Image Processing Proceedings. International Conference on, Vol.3, pp.123 – 127, 1998

[7] R. Achanta, M. Kankanhalli, P. Mulhem, "Compressed domain object tracking for automatic indexing of objects in MPEG home video", Proc of ICME, Vol. 2, pp. 61 – 64, 2002

[8] J. Zhu, S. Schwartz, B. Liu, "A Transform Domain Approach to Real-Time Foreground Segmentation in Video Sequences", Proceedings. (ICASSP '05). IEEE International Conference on, Vol.2, pp.685 – 688, 2005

[9] M.J.Swain, D.H.Ballard, "Color indexing", Int'l J. of Computer Vision, Vol.7-1, pp.11-32, 1991

[10] P. Norhashimah, H. Fang, J. Jiang, "Video extraction in compressed domain", IEEE Proc. of Conference on Advanced Video and Signal Based Surveillance, pp.321 – 326, 2003

[11] T. Gevers, A.W.M. Smeulders, "A Comparative Study of Several Color Models for Color Image Invariant Retrieval", Proc. Intern. Workshop on Image Database and Multimedia Search, pp. 17-23, 1996

[12] E. Durucan, T. Ebrahimi, "Change detection and background extraction by linear algebra", Proc. of IEEE, Vol.89-10, pp. 1368-1381, 2001