New Video Object Segmentation Technique Based on Flow-Thread Features for MPEG-4 and Multimedia Systems

Ho-Chao Huang[a], Yung-Chieh Lin[b], Yi-Ping Hung[a,b], Chiou-Shann Fuh[b]

[a]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[b]Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan

ABSTRACT

In this paper, we present a novel technique for video object (VO) segmentation. Compared to the existing VO segmentation methods, our method has the advantage that it does not decompose the VO segmentation problem into an initial image segmentation problem (segmenting a single image frame) followed by a temporal tracking problem. Instead, motion information contained in a finite duration of the image sequence is considered simultaneously. Given a video sequence, our method first estimates motion vectors between consecutive images, and then constructs the flow-thread for each pixel based on the estimated motion vectors. Here, a flow-thread is a series of pixels obtained by tracing the motion vectors along the image sequence. Next, we extract a set of flow-thread features (ft-features) from each flow-thread, which is then used to classify the associated pixel into the VO it belongs to. The segmentation results obtained by our unsupervised method look promising and the processing speed is fast enough for practical uses.

Keywords: MPEG-4, Video Object Segmentation, Flow-Thread, Fourier Descriptor, Pattern Recognition

1. INTRODUCTION

In MPEG-4 visual coding standard, the video stream can be divided into several video objects (VOs) based on the demands of interactive multimedia applications. Those VOs can be separately encoded, stored, or transmitted. The MPEG-4 based multimedia system can reassemble, remove, or replace some VOs in the video stream as necessary. Since VO is the basic interactive unit in MPEG-4 video stream, how to automatically or semi-automatically classify VOs from an image sequence has become one of the important issues for a MPEG-4 authoring system.

Recently, many research results[1-9] have been proposed for segmenting an image sequence into several VOs. Some of these methods are semi-automatic,[7,8] and some are unsupervised.[4-6] Some representatives of previous research will be reviewed in section 2. In general, the segmentation results of semi-automatic methods are much better than those unsupervised methods. However, human assistance in semi-automatic methods is less desired because it may limit the applications.

In this paper, we present a novel technique for VO segmentation. Our method can be divided into two stages: the analysis stage and the classification stage. In the analysis stage, a finite duration of image sequence are converted into a dense field of flow-threads, and then a set of flow-thread features (ft-feature) are extracted from each flow-thread. Here, a flow-thread is a series of pixels obtained by tracing the motion vectors along the image sequence. In the classification stage, ft-features are used to classify the VOs either by unsupervised approach or by supervised approach. When using unsupervised approach, users first tell the system the number of desired VOs, and then the VOs are extracted automatically by unsupervised pattern recognition technique. When using supervised approach, users must manually assist the system by choosing some training samples of desired VOs, and then the VOs are segmented by supervised pattern recognition technique. Another possibility is to use the unsupervised segmentation results as the training samples. Figure 1 illustrates the block diagram of our segmentation system, where the dashed boxes indicate the two stages. Details of the segmentation algorithm are presented in section 4.

Correspondence: Y.-P. Hung, E-Mail: hung@iis.sinica.edu.tw

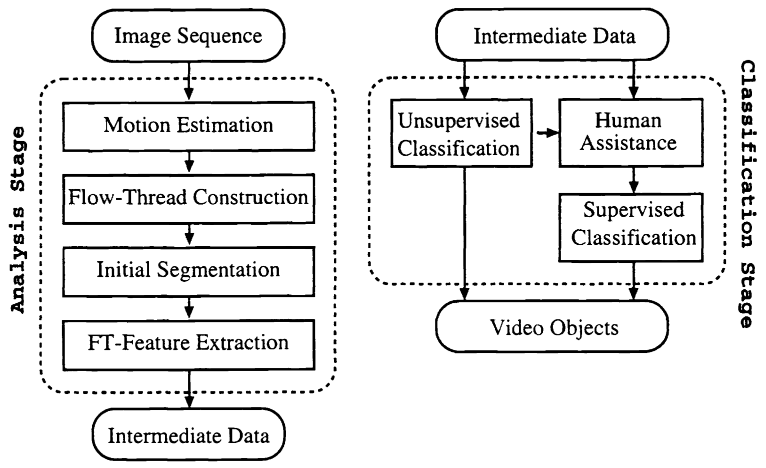

Figure 1. The block diagram of our video object segmentation method.

2. REVIEW

This section briefly reviews several representative research results on VO segmentation. Most of the VO segmentation methods[5,7,8] are based on a system architecture that combines an image segmentation procedure with a motion tracking procedure. For example, in Ref. 8, the VO segmentation is done in two stages. The first stage is the segmentation stage which is performed on I-frames (intra frames) of the video stream. The initial contour of the desired object is obtained by image processing technique with user assistance. The second stage is the motion estimation and object tracking process, which is performed on the P-frames (predicted frames) using the initial contour obtained from the first stage performed on I-frames. The method proposed in Ref. 7 is similar to that in Ref. 8, and its segmentation procedure also needs user assistance. Although the experimental results shown in Refs. 7 and 8 look good, it is not easy for the object tracking process to obtain good segmentation result when tracking long period of P-frames. Thus, the human editing process for I-frames segmentation becomes the bottleneck and is a tedious work for the users. Refs. 4 and 5, the proposed methods are unsupervised. The segmentation method proposed in Ref. 4 is based on motion cue and its application is restricted to those videos containing fast moving foreground or background objects. The method proposed in Ref. 5 is based on watersheds and tracking techniques, which segments images into homogeneous regions, and may require further merging to form VOs.

Some other researches on VO segmentation use different approaches. In Ref. 6, the VOs are extracted by using normalized cuts. The idea is interesting, but the cutting process is time-consuming and its results do not seem to be good enough for practical uses. Ref. 1 focuses on segmenting moving objects from static background, which restricts its application. Ref. 9 proposes a spatio-temporal algorithm based on the combination of temporal edges and asymmetric fuzzy-C-mean on spatial region classification. Ref. 10 uses morphological operators, but only show some preliminary results.

3. MOTIVATION

In traditional image segmentation, a single image is partitioned into homogeneous regions based on colors or textures. For video object segmentation, additional information extracted from motion should be utilized. In many cases, motion consistency is even more important than color consistency or texture consistency for extracting meaningful VOs. Hence, in this paper, VO segmentation is formulated to be a problem of clustering motion trajectories into groups having consistent characteristics. Most of the previously proposed methods only use the motion vectors between two consecutive images at a time, and the segmentation result is usually more sensitive to the noise. Instead, we use the longer-term motion information which is contained in the flow-thread. Here, a flow-thread is a series of pixels obtained by tracing the motion vectors along the image sequence. Our experiments show that flow-threads of longer duration are usually more classifiable. Another benefit of using flow-threads is that we can suppress the effects of noisy motion estimation by smoothing the flow-threads. This is performed by applying a feature-extraction function to the flow-threads to generate a set of features, and then choosing the most important features to be the representatives.

4. SEGMENTATION TECHNIQUE

Our VO segmentation method is divided into two stages: the analysis stage (i.e., a feature extraction stage), and the classification stage. The analysis stage consists of four steps: motion estimation, flow-thread construction, ft-feature extraction, and initial segmentation. In the classification stage, we implement an interactive system to provide two modes of segmentation: supervised classification and unsupervised classification. In the following subsections, we shall describe the four steps in the first stage and the two segmentation modes in the second stage.

4.1. Motion Estimation

There are many previous works available for motion estimation. It is worth to mention that the accuracy of the estimated motion vectors determines the quality of flow-threads, which will further affects the result of VO segmentation. For better result, we should estimate motion vectors to subpixel precision in order to avoid large error accumulation in flow-thread construction. Currently, we use a gradient-descent hierarchical algorithm[11] to first generate block-wise motion vectors, and then further refine these vectors by using a regularization method[12] to obtain pixel-wise motion vectors. Let a video segment V contain a sequence of images, as expressed in the following

$$V = \{I_t | 1 \leq t \leq T\}, \tag{1}$$

where I_t is the t-th image in V, and T is the total number of images in V. It is convenient to use $I_t(x)$ to represent a pixel at I_t, or the color of that pixel, where x is a two-tuple indicating the coordinates in an image. For each image point x in I_t, we estimate both the forward motion vector (using I_{t+1}) and the backward motion vector (using I_{t-1}). If a good motion vector can not be found for a pixel, its motion vector will be assigned as (∞, ∞). The estimated forward and backward motion vectors of the $I_t(x)$ are denoted as $v_t^{(f)}(x)$ and $v_t^{(b)}(x)$, where

$$\begin{cases} I_t(x) \sim I_{t+1}(x + v_t^{(f)}(x)) & \text{if } v_t^{(f)} \neq (\infty, \infty); \\ I_t(x) \sim I_{t-1}(x + v_t^{(b)}(x)) & \text{if } v_t^{(b)} \neq (\infty, \infty). \end{cases} \tag{2}$$

4.2. Flow-Thread Construction

The total number of images in a video segment is usually very large. If we are going to extract the flow-threads along the whole video segment, it will require a great amount of memory. Therefore, when I_{t_s} is to be segmented, a temporal window $W = [t_s - \lceil L/2 \rceil + 1, t_s + \lfloor L/2 \rfloor]$ is created to restrict the length of the flow-thread, where L is the desired length of the flow-thread, and it must be greater than 1. If it is necessary, the temporal window W is shifted δ units to keep $[t_s - \lceil L/2 \rceil + 1 + \delta, t_s + \lfloor L/2 \rfloor + \delta]$ inside $[1, T]$. After the temporal window is determined, the forward and backward motion vectors are traced along the image sequence to generate flow-thread. Let $\Gamma_{t_s}(x)$ be the flow-thread associated with the pixel $I_{t_s}(x)$. That is,

$$\Gamma_{t_s}(x) = \{\gamma_{t_s}(x, t) | t_b \leq t \leq t_e\} \tag{3}$$

where, t_b and t_e are the indices of the beginning frame and the ending frame in the range of W, respectively, and $\gamma_{t_s}(x, t)$ is a two-tuple indicating the image coordinates of the pixel in I_t corresponding to the reference pixel $I_{t_s}(x)$, or more precisely,

$$\begin{cases} \gamma_{t_s}(x, t) = x & \text{if } t = t_s; \\ \gamma_{t_s}(x, t) = \gamma_{t_s}(x, t-1) + v_{t-1}^{(f)}(\gamma_{t_s}(x, t-1)) & \text{if } t > t_s; \\ \gamma_{t_s}(x, t) = \gamma_{t_s}(x, t+1) + v_{t+1}^{(b)}(\gamma_{t_s}(x, t+1)) & \text{if } t < t_s. \end{cases} \tag{4}$$

If during the construction of a flow-thread Γ there exists an $\gamma_t \in \Gamma$ such that $v_t^{(f)}(\gamma_t) = (\infty, \infty)$ or $v_t^{(b)}(\gamma_t) = (\infty, \infty)$, this flow-thread will be considered to be partially occluded. In our current implementation, a partially occluded flow-thread does not considered to be classifiable. However, if some local features can be extracted for classification, then a partially occluded flow-thread can be considered to be classifiable, which is our future work. If a flow-thread is not partially occluded, it is called a complete flow-thread. See Fig. 2 for an illustration of the flow-thread associated with $I_{t_s}(x)$.

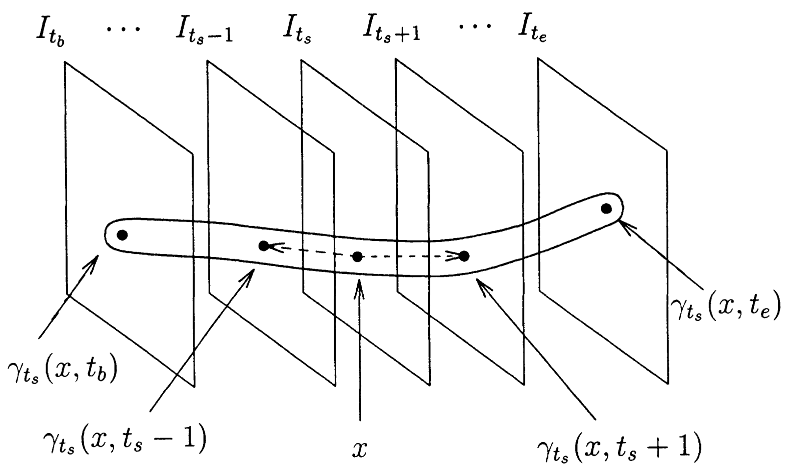
I_{t_b} $\cdots$ I_{t_s-1} I_{t_s} I_{t_s+1} $\cdots$ I_{t_e}
$\gamma_{t_s}(x, t_e)$
$\gamma_{t_s}(x, t_b)$
$\gamma_{t_s}(x, t_s - 1)$
x
$\gamma_{t_s}(x, t_s + 1)$

Figure 2. The flow-thread associated with $I_{t_s}(x)$, where $\gamma_{t_s}(x, t_s + 1) - x = v_{t_s}^{(f)}(x)$, and $\gamma_{t_s}(x, t_s - 1) - x = v_{t_s}^{(b)}(x)$.

4.3. Feature Extraction

After the flow-thread for each pixel in image I_{t_s} has been constructed, a set of features, referred to as the ft-features, will be extracted from each flow-thread. Equation (5) gives an example of the feature-extraction function which we have tried.

$$F(\Gamma_{t_s}(x)) = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 \end{pmatrix}^t, \tag{5}$$

where α_1 is the net flow distance of the flow-thread, α_2 is the net flow direction of the flow-thread, α_3 is the mean time of the flow-thread movement, α_4 is the total flow distance of the flow-thread, and α_5 is the total flow acceleration of the flow-thread. That is,

$$\begin{aligned}
\alpha_1 &= ||\gamma_{t_s}(x, t_e) - \gamma_{t_s}(x, t_b)||, && (6)\\
\alpha_2 &= \Theta(\gamma_{t_s}(x, t_e) - \gamma_{t_s}(x, t_b)), && (7)\\
\alpha_3 &= \frac{\sum_{t_b \le t < t_e} t \times ||v(t)||}{\sum_{t_b \le t < t_e} ||v(t)||}, && (8)\\
\alpha_4 &= \sum_{t_b \le t < t_e} ||v(t)||, && (9)\\
\alpha_5 &= \sum_{t_b \le t < t_e - 1} ||v(t) - v(t+1)||, && (10)
\end{aligned}$$

where $v(t) = \gamma_{t_s}(x, t + 1) - \gamma_{t_s}(x, t)$, and $\Theta(\gamma)$ is the polar angle of γ. This feature-extraction function is quite straightforward, but is hard to generalized if more than five features are desired.

Another example of the feature-extraction function is to use the Discrete Fourier Transform (DFT). The DFT of a flow-thread $\Gamma_{t_s}(x)$ is defined to be

$$d_k = \sum_{t_b \le t \le t_e} \gamma_{t_s}^*(x, t) \times e^{-j\omega_k (t - t_b)}, \tag{11}$$

where d_k is a complex number indicating the Fourier Descriptor (FD) at frequency ω_k, $\omega_k = 2\pi k/L$, and $\gamma_{t_s}^*(x, t)$ is the complex form of $\gamma_{t_s}(x, t)$, i.e., $\gamma_{t_s}^*(x, t) = a + jb$ if $\gamma_{t_s}(x, t) = (a, b)$. After performing the DFT, each flow-thread is transformed into L FDs. Here, the feature-extraction function can be written as

$$\mathcal{F}(\Gamma_{t_s}(x)) = \begin{pmatrix} d_0 & d_1 & \cdots & d_{L-1} \end{pmatrix}^t. \tag{12}$$

The set of FDs represents a flow-thread in the frequency domain, and from the inverse transform we have

$$\gamma^*_{t_s}(x,t) = \frac{1}{L} \sum_{0 \le k < L} d_k \times e^{j\omega_k (t-t_b)}. \tag{13}$$

According to our experience, the FD d_0, which is the centroid of the projected trajectory (i.e., the trajectory projected on image I_{t_s}), should be used carefully. Consider Equation (14), the first derivative of Equation (13), which represents the velocity of the reference pixel $I_{t_s}(x)$. It can be seen that the velocity of the reference pixel is independent of d_0 because $\omega_0 = 0$. Hence, it is reasonable not to select d_0 as a representative feature, since our purpose is to cluster the motion trajectories.

$$\frac{\partial \gamma^*_{t_s}(x,t)}{\partial t} = \frac{1}{L} \sum_{0 \le k < L} (j\omega_k d_k) \times e^{j\omega_k (t-t_b)} \tag{14}$$

4.4. Initial Segmentation

There are two approaches to use the ft-features for VO segmentation. One is the pixel-wise segmentation, and the other one is the region-based segmentation. These two approaches are different in the primitives to be classified. The pixel-wise segmentation directly classifies the ft-features of each pixel. On the other hand, the region-based segmentation first partitions the images into a set of primitive regions by applying an initial image segmentation to I_{t_s}, and then classifies the representative ft-features of each region. The major advantage of using pixel-wise segmentation is its capability of separating VOs sharing weak edges on the object boundary. However, its segmentation results are noisy if the motion estimation is not accurate enough, and it requires more computation time for classification due to the huge number of the primitive units. Our current implementation uses the approach of region-based segmentation, and the initial regions are obtained by a toboggan image segmentation algorithm.[13] After the image is initially segmented, the representative ft-features for each region are computed from pixel-wise flow-threads.

To compute the representative ft-feature for a primitive region, the reliability of the flow-threads should be considered, especially when the motion estimation is not accurate. Here, we use a confidence measure which is a function of image Hessian and color variance. We assume the motion vector is more accurate at some pixel if its Hessian is larger. That is, the flow-thread $\Gamma_{t_s}(x)$ is more reliable if $h^2_{t_s}(x)$ is large, where $h_{t_s}(x) = \det\nabla^2 I_{t_s}(x)$. Besides, the color variance $\sigma^2_{t_s}(\mathrm{x})$ is evaluated as a measure of the accuracy of motion estimation, i.e. $\sigma^2_{t_s}(x) = E\left(I^2_t(\gamma_{t_s}(x,t)), t_b \le t \le t_e\right) - E^2\left(I_t(\gamma_{t_s}(x,t)), t_b \le t \le t_e\right)$. Then, the confidence measure $\mathcal{C}$ is defined to be the product of two sigmoid functions:

$$\mathcal{C}(\Gamma_{t_s}(x)) = \begin{cases} 0 & \text{if } \Gamma_{t_s}(x) \text{ is partially occluded;} \\ \epsilon + \left(1 + \exp(+s_1(\sigma^2_{t_s}(x) - \sigma^2_0))\right)^{-1} & \\ \quad \times \left(1 + \exp(-s_2(h^2_{t_s}(x) - h^2_0))\right)^{-1} & \text{if } \Gamma_{t_s}(x) \text{ is complete,} \end{cases} \tag{15}$$

where s_1, s_2, σ_0, and h_0 are parameters of the sigmoid functions, and ϵ is a small positive number to avoid zero confidence. Then the representative ft-features of a primitive region R_{t_s} are defined to be the weighted sum of the pixel-wise ft-features:

$$\mathcal{F}(R_{t_s}) = \frac{\sum_{x \in R_{t_s}} \mathcal{C}(\Gamma_{t_s}(x)) \times \mathcal{F}(\Gamma_{t_s}(x))}{\sum_{x \in R_{t_s}} \mathcal{C}(\Gamma_{t_s}(x))}. \tag{16}$$

If a primitive region R_{t_s} has at least one pixel whose flow-thread is complete, then the region R_{t_s} is called a ft-region, and can be classified in the following classification stage. In our current implementation, only ft-region will be classified. Because we use the FDs as ft-features, the representative ft-features have an interesting property which can be used for reducing some computation. The representative flow-thread $\Gamma(R_{t_s})$ of the region R_{t_s} is defined as

$$\Gamma(R_{t_s}) = \left\{ \gamma(R_{t_s}, t) \middle| \gamma(R_{t_s}, t) = \frac{\sum_{x \in R_{t_s}} \mathcal{C}(\Gamma_{t_s}(x)) \times \gamma_{t_s}(x,t)}{\sum_{x \in R_{t_s}} \mathcal{C}(\Gamma_{t_s}(x))}, t_b \le t \le t_e \right\}. \tag{17}$$

It is easy to verify that the we can obtain representative features alternately by applying DFT to the representative flow-thread directly, i.e., $d_k = \sum_{t_b \le t \le t_e} \gamma^*(R_{t_s}, t) \times e^{-j\omega_k (t-t_s)}$. Therefore, we can first construct the representative flow-threads, and then extract the representative ft-features. This avoids performing DFT on each pixel-wise flow-thread, which is much more time-consuming.

4.5. Video Object Segmentation

After initial segmentation, image I_{t_s} is partitioned into several homogeneous regions and each region has a set of representative ft-features. When the DFT is used as the feature-extraction function, there are L FDs in each set of representative ft-features. If L is large, it is very time-consuming to classify the VOs using all of the features. Therefore, we select a subset of L FDs for classification, as described in the following two subsections. Let N_U and N_S be the numbers of features selected for unsupervised classification and supervised classification, respectively. The major difference between the feature selection methods for the two kinds of classification is on the availability of the training samples. Because the training samples are available for supervised classification, they can also be used to measure the separability for feature selection. For unsupervised classification, there is no training sample, and hence we select features which make the samples have larger variance. Since the FDs are complex numbers, the actual dimensions of the feature spaces for unsupervised and supervised classification are $2N_U$ and $2N_S$, respectively.

4.5.1. Unsupervised Classification

Before applying the unsupervised classification technique to the representative ft-features of all primitive regions, a feature-selection procedure is taken to reduce the dimension of the feature space. In our implementation, the variance of each feature is used as the measure of the separability. For example, given n primitive regions, $R_{t_s}^{(1)}, \ldots, R_{t_s}^{(n)}$, obtained from the last section, the separability measure of the L ft-features is

$$\mathcal{M}_U = \frac{1}{n}\left(\sum_{1\le i\le n} \mathcal{F}(R_{t_s}^{(i)})\mathcal{F}(R_{t_s}^{(i)})^*\right) - \left(\frac{1}{n}\sum_{1\le i\le n} \mathcal{F}(R_{t_s}^{(i)})\right)\left(\frac{1}{n}\sum_{1\le i\le n} \mathcal{F}(R_{t_s}^{(i)})\right)^*, \quad (18)$$

where A^* is the complex conjugate of A^t, and $\mathcal{M}_U$ is a $L \times L$ matrix. Since our goal is to select a subset of features which make the samples have larger variance, the N_U largest values in the diagonal $\mathcal{M}_U$ decide which features are used in the following unsupervised classification. Next, an unsupervised clustering algorithm, such as K-means algorithm or LBG algorithm, is used to cluster the ft-regions into K representative classes using the N_U selected ft-features. Notice that the FD d_0 is not used in classification for the reason explained in section 4.3.

4.5.2. Supervised Classification

For supervised classification, users can simply assign some training samples by a user interface, or modify the results of unsupervised classification to generate more precise training samples. In our implementation, a multi-layer perceptron neural network is used to learn from the training samples and classify the remaining unlabeled ft-regions. To reduce the number of nodes of the input layer, a feature selection procedure is taken before training the neural network. Since training data are provided, we can choose the features where the feature means of VOs are most separable. For example, given m training VOs, $O_{t_s}^{(1)}, \ldots, O_{t_s}^{(m)}$, each of which is a set of ft-regions in I_{t_s}, the sum of square distance between FD means is used to measure the separability, i.e.,

$$\mathcal{M}_S = \sum_{1\le i_1<i_2\le m} \left(E(\mathcal{F}(R), R\in O_{t_s}^{(i_1)}) - E(\mathcal{F}(R), R\in O_{t_s}^{(i_2)})\right)\left(E(\mathcal{F}(R), R\in O_{t_s}^{(i_1)}) - E(\mathcal{F}(R), R\in O_{t_s}^{(i_2)})\right)^*, \quad (19)$$

where $\mathcal{M}_S$ is a $L \times L$ matrix. Since there are N_S features wanted, the N_S largest values in the diagonal of $\mathcal{M}_S$ decide which features will be used as the inputs of the neural network.

5. EXPERIMENTS

In our previous work, we have implemented a system using pixel-wise segmentation approach with the feature-extraction function $F(\Gamma)$ in Equation (5). This system allows user to adjust the coefficients of a linear discriminant function manually, and obtains some good results as shown in Fig. 3.

Currently, we have a new system incorporating the unsupervised and supervised classification, making the interface much easier to use, and allowing users to decide the number of features selected to be used in classification. This system uses region-based segmentation with Fourier descriptors as ft-features. The unsupervised classification can use either the K-means algorithm or the LBG algorithm provided by Intel's Recognition Library.[14] The supervised classification uses the neural network simulator provided by Nevada backPropagation[15] version 4. See Fig. 4 for some results of MPEG-4 test sequence using supervised classification. In Fig. 5, results of the supervised and unsupervised

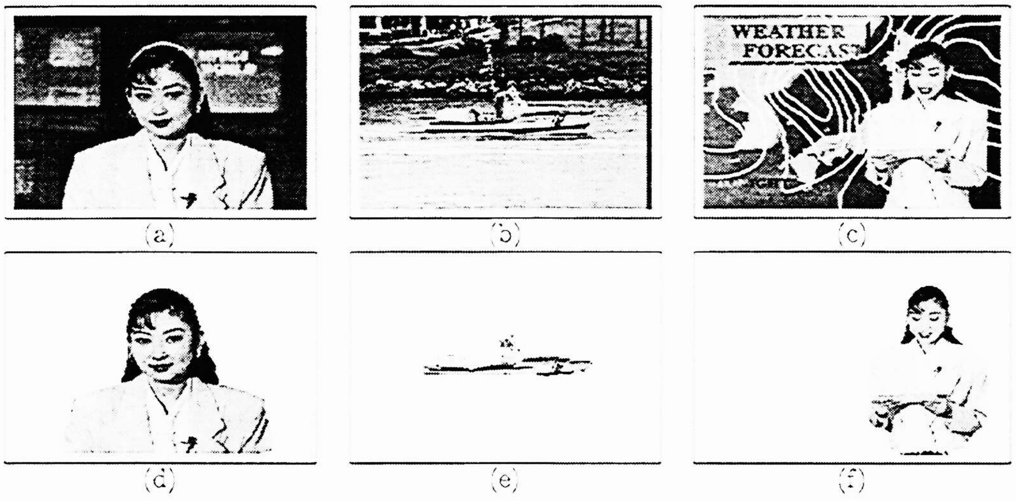

Figure 3. Results of pixel-wise segmentation using the feature-extraction function F shown in Equation (5). (a), (b), and (c) are some images of the videos, `Akiyo`, `Coastguard`, and `Weather`, respectively. (d), (e), and (f) are the extracted VOs from (a), (b), and (c).

classification are compared. The results of supervised classification are slightly better than the results of unsupervised classification as we can see that the central part of the boat body is broken in Fig. 5(c). The segmentation results in Figs. (3), (4), and (5) are filtered by morphological operators for noise removal. All experiments use the same parameters: $L = 16$, $s_1 = s_2 = 10^{-5}$, $N_U = 2$, and $N_S = 4$. The image resolutions of the three sequences, `Akiyo`, `Coastguard`, and `Weather`, are 360×242, 360×240, and 360×242, respectively.

The processing time of the analysis stage for each frame varies from 5 to 20 seconds on Pentium-II PCs, depending on the complexity of the video stream. The processing time of unsupervised classification is one to two seconds, and the processing time of supervised classification varies according to the number of training samples, normally is less than one minute.

In Figs. 3(d) and 4(c), the `Akiyo` sequence is segmented by pixel-based and region-based methods, respectively. The pixel-based segmentation produces a better result in Fig. 3(d) because it uses longer flow-threads (larger L) and its discriminant function is tuned manually. Without manual assistance, it is hard to separate the desired VO, i.e. the reporter, from a short duration of the `Akiyo` sequence because the VO is nonrigid. After observing the whole sequence, we can find that the head of the reporter has the most significant motion and the body of the reporter only has a little motion. Therefore, in order to accomplish a highly automatic segmentation system, it requires a method to adaptively determine the length of flow-thread used for VO segmentation which will be long enough to include significant motion information. However, it is still possible to segment a nonrigid VO into different parts, which can be merged into high-level VOs in post-processing.

In Fig. 4(f), some parts of background are miss-classified to the VO in the foreground. This may be caused by inaccurate motion estimation. For further improvement, we plan to use the multi-frame flow estimation algorithm.

6. CONCLUSION

This paper has proposed a new technique for video object (VO) segmentation. Different from previous research results on VO segmentation, our method uses the flow-thread features, which are more global in the temporal dimension. The proposed technique first estimates a dense field of bi-directional motion vectors. Then it traces flow-threads for each pixel in the reference frames, and extracts a set of ft-features for each flow-thread. We have also developed an interactive system for helping the user to perform unsupervised and/or supervised classification, and examine the segmented VOs. The proposed technique is efficient and highly automatic, and the result can be used by MPEG-4 and other multimedia authoring systems to segment and manipulate VOs. The processing speed of the system is fast enough for many applications.

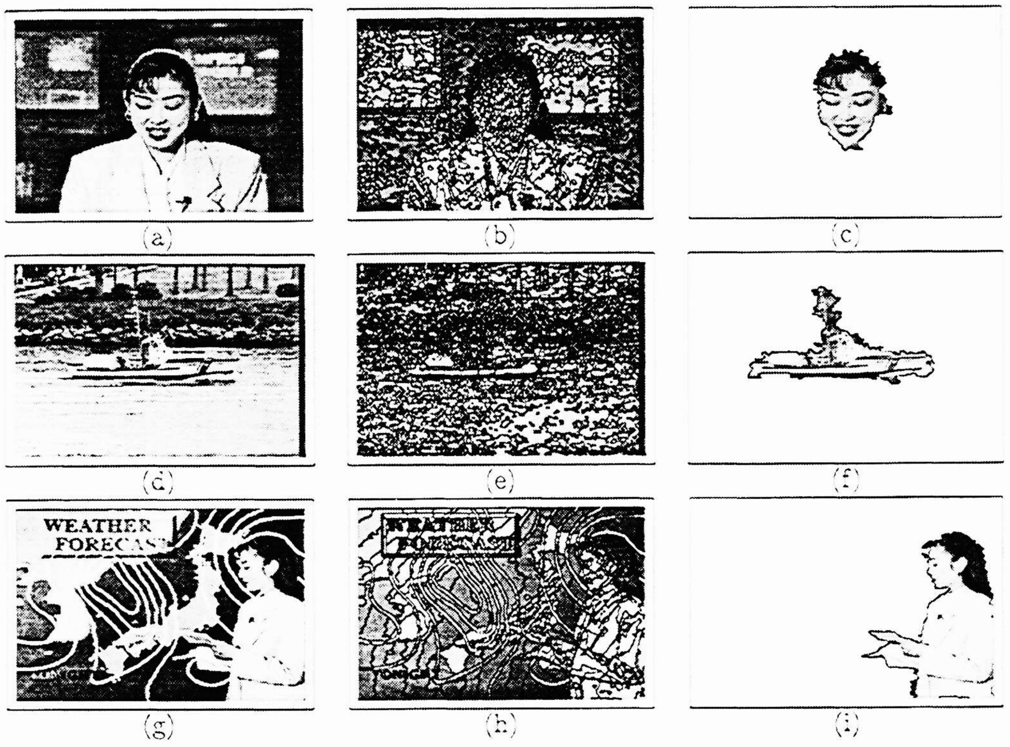

Figure 4. Results of region-based segmentation using Fourier descriptors as features. (a), (d), and (g) are some images of the videos, `Akiyo`, `Coastguard`, and `Weather`, respectively. (b), (e), and (h) are the intermediate results of initially segmented primitive regions using a toboggan image segmentation algorithm. (c), (f), and (i) are the extracted VOs by unsupervised classification.

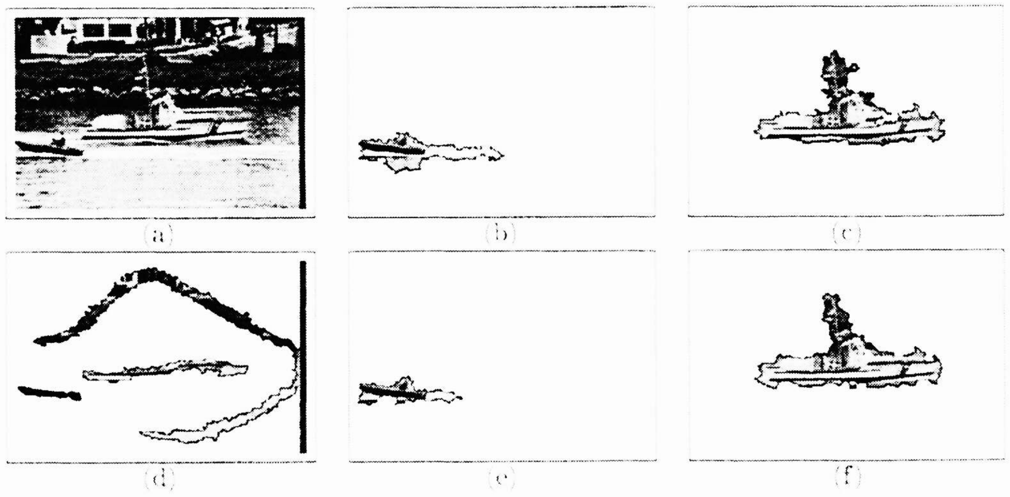

Figure 5. (a) is an image of the video, `Coastguard`, containing two desired VOs. (b) and (c) are the VOs extracted by the unsupervised classification. (e) and (f) are the VOs extracted by supervised classification using (d) as the training samples, where the three strips indicate two desired VOs and the background.

REFERENCES

1. A. Neri, S. Colonnese, G. Russo, and P. Talone, "Automatic moving object and background separation," *Signal Processing* **66**(2), pp. 219–232, 1998.
2. R. Mech and M. Wollborn, "A noise robust method for segmentation of moving objects in video sequences," in *IEEE Intl. Conf. Acoust., Speech, Signal Processing*, vol. 4, pp. 2657–2660, (Munich, Germany), April 1997.
3. J. G. Choi, S. W. Lee, and S. D. Kim, "Spatio-temporal video segmentation using a joint similarity measure," *IEEE Trans. on Circuits and Systems for Video Technology* **7**, pp. 279–286, April 1997.
4. T. Meier and K. N. Ngan, "Automatic segmentation of moving objects for video object plane generation," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, pp. 525–538, September 1998.
5. D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, pp. 539–546, September 1998.
6. J. Shi and J. Malik, "Motion segmentation and tracking using normalized cuts," in *Proc. of IEEE Intl. Conf. on Computer Vision*, pp. 1154–1160, 1998.
7. R. Castagno, T. Ebrahimi, and M. Kunt, "Video segmentation based on multiple features for interactive multimedia applications," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, pp. 562–571, September 1998.
8. C. Gu and M.-C. Lee, "Semiautomatic segmentation and tracking of semantic video objects," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, pp. 572–584, Septemeber 1998.
9. Y.-R. Choo, P.-C. Chung, and et al., "Temporal edges and spatial classification for video object segmentation," in *Proceedings of International Symposium on Multimedia Information Processing December*, pp. 245–250, (Taipei, Taiwan), 1999.
10. P. Salembier, P. Brigger, J. R. Casas, and M. Pardas, "Morphological operators for image and video compression," *IEEE Trans. on Image Processing* **5**, pp. 881–898, June 1996.
11. R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. on Communications* **33**, pp. 888–896, August 1985.
12. H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic, Dordrecht, 1996.
13. Y.-P. Hung and X. Yao, "Keep-sliding toboggan image segmentation," in *Proceedings of National Computer Symposium*, vol. 2, pp. 392–397, (Taiwan), December 1991.
14. *Intel Recognition Primitives Library Reference Manual*, Intel Corporation, 1999.
15. P. Goodman, *NevProp software, version 4*, University of Nevada, Reno, NV (http://www.scs.unr.edu/nevprop), 1996.