4.10

Video Segmentation

A. Murat Tekalp
University of Rochester

1 Introduction

Video segmentation is an integral part of many video analysis and coding problems, including: (i) video summarization, indexing and retrieval, (ii) advanced video coding, (iii) video authoring and editing, (iv) improved motion (optical flow) estimation, and (v) 3D motion and structure estimation with multiple moving objects [1–3]. The first three applications concern multimedia services which require temporal segmentation of video into shots or groups of pictures (GoP). They may also benefit from spatio-temporal segmentation of video into objects for better content description, object-based video editing, and rate allocation. The latter two are computer vision applications, where spatio-temporal segmentation helps to identify optical flow boundaries (motion edges) and occlusion regions where the smoothness constraint should be turned off for better optical flow estimation, and distinct 3D motion and structure parameters are needed to model the flow vectors associated with each independently moving object, respectively.

Video segmentation refers to partitioning video into spatial, temporal or spatio-temporal regions that are homogeneous in some feature space [4]. As with any segmentation problem, effective video segmentation requires proper feature selection and an appropriate distance measure. Different features and homogeneity criteria generally lead to different segmentations of the same video, e.g., color, texture or motion segmentation. Furthermore, there is no guarantee that any of the resulting automatic segmentations will be semantically meaningful, since a semantically meaningful region may have multiple colors, multiple textures and/or multiple motion. Although semantic objects can be computed automatically in some well-constrained settings, e.g., in video surveillance systems [16], where objects can be extracted by simple change detection and background subtraction methods; semantic object segmentation requires specialized capture methods (chroma-keying) or user interaction. Specific video segmentation methods should be considered in the context of the requirements of the application in which they are used. Factors that affect the choice of a specific segmentation method include [5]:

- Real-time performance: If segmentation must be performed in real-time, e.g., for rate-control in video-telephony, then simple algorithms which are fully automatic must be used. On the other hand, one can employ semi-automatic, interactive algorithms for off-line applications such as video indexing or off-line

video coding to obtain semantically meaningful segmentations [6].

- Precision of segmentation: If segmentation is employed to improve the compression efficiency or rate control, then certain misalignment between segmentation results and actual object borders may not be of concern. On the other hand, if segmentation is needed for object-based video authoring/editing or shape similarity matching, then it is of utmost importance that the estimated boundaries align with actual object boundaries perfectly, where even a single pixel error may not be tolerable.
- Scene complexity: Complexity of video content can be modeled in terms of amount of camera motion, color and texture uniformity within objects, contrast between objects, smoothness of motion of objects, objects entering and leaving the scene, regularity of object shape along the temporal dimension, frequency of cuts and special effects, etc. Clearly, more complex scenes require more sophisticated segmentation algorithms. For example, it is easier to detect cuts than special effects such as wipes or fades.

This chapter provides an overview of some video segmentation methods ranging from simple shot boundary detection and change detection techniques to more sophisticated motion segmentation and interactive video object segmentation and tracking methods. Although multimodal signal processing methods have been shown to be effective in specific applications [7], we cover only video modality in this chapter. We start our discussion with temporal video segmentation methods in Section 2. Change detection methods, where we study both two-frame methods and methods with memory are covered in Section 3. Motion segmentation methods are discussed in Section 4. We first introduce the "dominant motion" approach, which aims to label independently moving regions sequentially (one at a time). We then present the alternative "multiple motion segmentation" approach, including clustering in the motion parameter space, maximum likelihood segmentation, maximum a posteriori probability segmentation, and region labeling methods. Section 5 addresses simultaneous optical flow estimation and segmentation method, since the accuracy of segmentation results depends on the accuracy of the estimated motion field and vice versa. Finally, Section 6 deals with semantically meaningful object segmentation with emphasis on chroma-keying and semi-automatic (interactive) object tracking methods.

2 Scene Change Detection

Scene change or shot boundary detection is a relatively easy segmentation problem since it is one-dimensional, along the temporal dimension. Shot boundary detection methods locate temporal discontinuities, i.e., frames across which large differences are observed in some feature space, usually a combination of color and motion [8–13]. Temporal discontinuities may be abrupt (cuts) or gradual (special effects, such as wipes and fades). It is easier to detect cuts than special effects. The simplest approach for detecting temporal discontinuities is to quantify frame differences in the pixel intensity domain. If a pre-determined number of pixels exhibit differences larger than a threshold value, then a cut can be declared. Clearly, this method is sensitive to presence of noise, camera motion, and compression artifacts in the video. A slightly more robust approach may be to divide each frame into rectangular blocks, compute statistics of each block such as the mean and variance independently, and then check the count of blocks with changing statistics against a set threshold. Applying low-pass filtering to each frame prior to computing frame differences or block statistics should also improve robustness.

A more robust alternative is to consider frame histogram differences instead of pixelwise or blockwise frame intensity differences. To this effect, we compute n-bin color histogram, $h_k(i)$, $i = 1, \ldots, n$, for each frame k. Various measures and tests have been developed to quantify similarity or dissimilarity of histograms. These include the histogram intersection measure, χ^2 test, and Kolmogorov-Smirnov test [11]. A closely related approach is to detect changes in the counts of edge-pixels in successive frames, i.e., similarity of edge histograms. Although they are effective to detect cuts and fades, neither histogram differences nor intensity differences can usually differentiate between wipes and camera motion, such as pans and zooms. Detection of these special effects requires combination of histogram difference and camera motion estimation. Global motion can be estimated and frames are motion compensated before computation of the features [14]. Another approach to detect gradual changes is the so-called twin comparison method [15], which can be used with different features. A lower threshold is used to detect abrupt scene changes, while a higher threshold is used to detect the actual position of gradual ones.

There also exist shot boundary detection algorithms for specific domains, such as surveillance video [16], sports video [17], and movies [18, 19]. Sports video is arguably one of the most challenging domains for robust shot boundary detection due to: (1) existence of a strong color correlation between successive shots, since a single dominant color background, such as the soccer field, may be seen in successive shots; (2) existence of large camera and object motions; (3) existence of many gradual transitions, such as wipes and dissolves. Ekin et al. [17] observed that gradual transitions in sports video are not accurately detected by generic algorithms, and proposed using two features, the absolute difference between two frames of the ratio of dominant colored pixels to total number of pixels, and color histogram dissimilarity, measured by histogram intersection, for reliable shot boundary detection.

Videos are almost always stored and transmitted in compressed form. Detection of scene changes in real-time poses a challenge in many applications since decompressing and processing video data sequentially requires significant computational resources. Hence, the need for scene segmentation algorithms in the compressed domain (without completely decoding the bitstream) [21]. DC images which are spatially reduced versions of the original video frames can be constructed from the DC coefficient of each 8×8 block [20]. Successful results have been obtained for detection of both abrupt and gradual scene changes using DC images [2].

3 Spatio-Temporal Change Detection

Change detection methods segment each frame into two regions, namely changed and unchanged regions in the case of a static camera or global and local motion regions in the case of a moving camera [22]. This section deals only with the former case, where unchanged regions correspond to the background (null hypothesis) and changed regions to the foreground object(s) or uncovered (occlusion) areas. The case of moving camera is identical to the former, once the global motion between successive frames due to camera motion is estimated and compensated [23]. However, accurate estimation of the camera motion may require scene segmentation resulting in a chicken-egg problem. Fortunately, the dominant motion segmentation approach, presented in the next section, offers a solution to the estimation of the camera motion without prior scene segmentation. Hence, the discussion of the case of moving camera is deferred until Section 4.1.

Various change detection methods in the literature differ according to: (i) what features and background model are used, (ii) what distance metrics are used, and (iii) what kind of threshold and background model adaptation rules are used. In the following, we first discuss change detection using two frames. Temporal integration (using more than two frames) and combination of spatial and temporal segmentation are also studied to obtain spatially and temporally coherent regions.

3.1 Spatial Change Detection Using Two Frames

The simplest method to detect changes between two registered frames would be to analyze the frame difference (FD) image, which is given by

$$\mathrm{FD}_{k,r}(\mathbf{x}) = s(\mathbf{x}, k) - s(\mathbf{x}, r) \tag{1}$$

where $\mathbf{x} = (x_1, x_2)$ denotes pixel location and $s(\mathbf{x}, k)$ stands for the intensity value at pixel $\mathbf{x}$ in frame k. FD image shows the pixel-by-pixel difference between the current image k and the reference image r. The reference image r may be taken as the previous image $k-1$ (successive frame difference) or an image at a fixed time. Methods using a fixed reference frame, which may be updated for global illumination changes, are called background subtraction methods [26]. For example, if we are interested in monitoring a hallway using a fixed camera, an image of the hallway when it is empty may be used as a fixed reference image. Assuming that we have a static camera and the illumination remains more or less constant between the frames, the pixel locations where $\mathrm{FD}_{k,r}(\mathbf{x})$ differs from zero indicate regions "changed" due to local motion. In order to distinguish the nonzero differences that are due to noise from those that are due to local motion, segmentation can be achieved by thresholding the FD as

$$z_{k,r}(\mathbf{x}) = \begin{cases} 1 & \text{if } |\mathrm{FD}_{k,r}(\mathbf{x})| > T \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where T is an appropriate threshold. Here, $z_{k,r}(\mathbf{x})$ is called a segmentation label field, which is equal to "1" for changed regions and "0" otherwise. The value of the threshold T can be chosen by an optimal threshold determination algorithm (see Chapter 2.2). This pixelwise thresholding is generally followed by one or more post-processing steps to eliminate isolated labels. Postprocessing operations include forming 4- or 8-connected regions and discarding labels with less than a predetermined number of entries, and morphologic filtering of the changed and unchanged region masks.

In practice, a simple FD image analysis is not satisfactory for two reasons: First, a uniform intensity region may be interpreted as stationary even if it is moving (aperture problem). It may be possible to avoid the aperture problem using a multi-resolution decision procedure, since uniform intensity regions are smaller at lower resolution levels. Second, the intensity difference due to motion is affected by the magnitude of the spatial gradient in the direction of motion. This problem can be addressed by considering a locally normalized frame difference function [24], or locally adaptive thresholding [25]. An improved change detection algorithm that addresses both concerns can be summarized as:

(i) Construct a Gaussian pyramid where each frame is represented in multiple resolutions. Start processing at the lowest resolution level.

(ii) For each pixel at the present resolution level, compute the normalized frame difference given by [24]

$$\mathrm{FDN}_{k,r}(\mathbf{x}) = \frac{\sum_{\mathbf{x}\in\mathcal{N}} |s(\mathbf{x}, k) - s(\mathbf{x}, r)||\nabla s(\mathbf{x}, r)|}{\sum_{\mathbf{x}\in\mathcal{N}} |\nabla s(\mathbf{x}, r)|^2 + c} \tag{3}$$

where $\mathcal{N}$ denotes a local neighborhood of the pixel $\mathbf{x}$, $\nabla s(\mathbf{x}, r)$ denotes the gradient of image intensity at pixel $\mathbf{x}$, and c is a constant to avoid numeric instability. If the normalized difference is high (indicating that

the pixel is moving), replace the normalized difference from the previous resolution level at that pixel with the new value. Otherwise, retain the value from the previous resolution level.

(iii) Repeat step (ii) for all resolution levels.

Finally, we threshold the normalized motion detection function at the highest resolution level.

3.2 Temporal Integration

An important consideration is to add memory to the motion detection process in order to ensure both spatial and temporal continuity of the changed regions at each frame. This can be achieved in a number of different ways, including temporal filtering (integration) of the intensity values across multiple frames before thresholding and post-processing of labels after thresholding.

A variation of the successive frame difference and normalized frame difference is the frame difference with memory $\mathrm{FDM}_k(\mathbf{x})$, that is defined as the difference between the present frame $s(\mathbf{x}, k)$ and a weighted average of past frames $\bar{s}(\mathbf{x}, k)$ given by [24]

$$\mathrm{FDM}_k(\mathbf{x}) = s(\mathbf{x}, k) - \bar{s}(\mathbf{x}, k) \tag{4}$$

where

$$\bar{s}(\mathbf{x}, k) = (1 - \alpha)s(\mathbf{x}, k) + \alpha\bar{s}(\mathbf{x}, k-1), \quad k = 1, \ldots \tag{5}$$

and

$$\bar{s}(\mathbf{x}, 0) = s(\mathbf{x}, 0)$$

Here $0 < \alpha < 1$ is a constant. After processing a few frames, the unchanged regions in $\bar{s}(\mathbf{x}, k)$ maintain their sharpness with a reduced level of noise, while the changed regions are blurred. The function $\mathrm{FDM}_k(\mathbf{x})$ is thresholded either by a global or a spatially adaptive threshold as in the case of two frame methods. The temporal integration increases the likelihood of eliminating spurious labels; thus, resulting in spatially contiguous regions.

Accumulative differences can be employed when detecting changes between a sequence of images and a fixed reference image (as opposed to successive frame differences). Let $s(\mathbf{x}, k)$, $s(\mathbf{x}, k-1), \ldots, s(\mathbf{x}, k-N)$ be a sequence of N frames, and let $s(\mathbf{x}, r)$ be a reference image. An accumulative difference image is formed by comparing every frame in the sequence with this reference image. For every pixel location, the accumulative image is incremented if the difference between the reference image and the current image in the sequence at that pixel location is bigger than a threshold. Thus, pixels with higher counter values are more likely to correspond to changed regions.

An alternative procedure that was adopted by MPEG-4 as a non-normative tool considers post-processing of labels [25]. First, scene changes are detected. Within each scene (shot), an initial change detection mask is estimated between successive pairs of frames by global thresholding of the frame difference function. Next, the boundary of the changed regions are smoothed by a relaxation method using local adaptive thresholds [22]. Then, memory is incorporated by re-labeling unchanged pixels which correspond to changed locations in one of the last L frames. This step ensures temporal continuity of changed regions from frame to frame. The depth of the memory L may be adapted to scene content to limit error propagation. Finally, post-processing to obtain the final changed and unchanged masks eliminates small regions.

3.3 Combination with Spatial Segmentation

Another consideration is to enforce consistency of the boundaries of the changed regions with spatial edge locations at each frame. This may be accomplished by first segmenting each frame into uniform color and/or texture regions. Next, each region resulting from the spatial segmentation is labeled as changed or unchanged as a whole as opposed to labeling each pixel independently. Region labeling decisions may be based on the number of changed and unchanged pixels within each region or thresholding the average value of the frame differences within each region [27].

4 Motion Segmentation

Motion segmentation (also known as optical flow segmentation) methods label pixels (or optical flow vectors) at each frame that are associated with independently moving part of a scene. The region boundaries may or may not be pixel-accurate or semantically meaningful. For example, a single object with articulated motion may be segmented into multiple regions. The occlusion and aperture problems are mainly responsible for misalignment of motion and actual object boundaries. Furthermore, model misfit possibly due to deviation of the surface structure from a plane generally leads to oversegmentation of the motion field. While it is possible to achieve fully automatic motion segmentation with some limited accuracy for certain content domains, semantically meaningful video object segmentation generally requires user interaction to define the object of interest in at least some key frames as discussed in Section 6.

Motion segmentation is closely related to two other problems, motion (change) detection and motion estimation. Change detection is a special case of motion segmentation with only two regions, namely changed and unchanged regions (in the case of a static camera) or global and local motion regions (in the case of a moving camera). An important distinction between change detection and motion segmentation is that the former can be achieved without motion

estimation if the scene is recorded with a static camera. Change detection in the case of a moving camera and general motion segmentation, on the other hand, require some sort of global and/or local motion estimation either explicitly or implicitly. Motion detection and segmentation are also plagued with the same two fundamental limitations associated with motion estimation: occlusion and aperture problems [4]. For example, pixels in a flat image region may appear stationary even if they are moving due to the aperture problem (hence the need for hierarchic methods); and/or erroneous labels may be assigned to pixels in covered or uncovered image regions due to the occlusion problem.

In general, application of standard image segmentation methods directly to estimated optical flow vectors may not yield meaningful results, since an object moving in 3D usually generates a spatially varying optical flow field [28]. For example, in the case of a rotating object, there is no flow at the center of the rotation, and the magnitude of the flow vectors grows as we move away from the center of rotation. Therefore, a parametric model-based approach, where we assume that the motion field can be described by a set of K parametric models, is usually adopted. In parametric motion segmentation, the model parameters are the motion features. Then, motion segmentation algorithms aim to determine the number of motion models that can adequately describe a scene, type/complexity of these motion models, and the spatial support of each motion model. Most commonly used types of parametric models are affine, perspective, and quadratic mappings, which assume a 3D planar surface in motion. In the case of a nonplanar object, the resulting optical flow can be modeled by a piecewise affine, perspective or quadratic flow field if we approximate the object surface by a union of a small number of planar patches. Since each independently moving object and/or planar patch will best fit a different parametric model, the parametric approach may lead to oversegmentation of motion in the case of nonplanar objects.

4.1 Dominant Motion Segmentation

Segmentation by dominant motion analysis refers to extracting one object (with the dominant motion) from the scene at a time [24, 29, 30, 38]. Dominant motion segmentation can be considered as a hierarchically structured top-down approach, that starts by fitting a single parametric motion model to the entire frame, and then partitions the frame into two regions, those pixels which are well represented by this dominant motion model and those that are not. The process converges to the dominant motion model in a few iterations, each time fitting a new model to only those pixels that are well represented by the motion model in the previous iteration. The dominant motion may correspond to the camera (background) motion or a foreground object motion, whichever occupies a larger area in the frame. The dominant motion approach may also handle separation of individually moving objects. Once the first dominant object is segmented, it is excluded from the region of analysis, and the entire process is repeated to define the next dominant object. This is unlike the multiple motion segmentation approaches that are discussed in the next section, which start with an initial segmentation mask (usually with many small regions) and refine them according to some criterion function to form the final mask. It is worth noting that the dominant motion approach is a direct method that is based on spatio-temporal image intensity gradient information. This is in contrast to first estimating the optical flow field between two frames and then segmenting the image based on the estimated optical flow field.

4.1.1 Segmentation Using Two Frames

Motion estimation in the presence of more than one moving objects with unknown supports is a difficult problem. It was Burt et al. [29] who first showed that the motion of a 2D translating object can be accurately estimated using a multiresolution iterative approach even in the presence of other independently moving objects without prior knowledge of their supports. This is, however, not always possible with more sophisticated motion models (e.g., affine and perspective), which are more sensitive to presence of other moving objects in the region of analysis.

To this effect, Irani et al. [24] proposed multi-stage parametric modeling of dominant motion. In this approach, first a translational motion model is employed over the whole image to obtain a rough estimate of the support of the dominant motion. The complexity of the model is then gradually increased to affine and projective models with refinement of the support of the object in between. The parameters of each model are estimated only over the support of the object based on the previously used model. The procedure can be summarized as follows:

(i) Compute the dominant 2D translation vector (d_x, d_y) over the whole frame as the solution of

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix} \tag{6}$$

where I_x, I_y and I_t denote partials of image intensity with respect to x, y, and t. In case the dominant motion is not a translation, the estimated translation becomes a first order approximation of the dominant motion.

(ii) Label all pixels that correspond to the estimated dominant motion as follows:

(a) Register the two images using the estimated dominant motion model. The dominant object appears stationary between the registered images, while other parts of the image are not.

(b) Then, the problem reduces to labeling stationary regions between the registered images, which can be solved by the multiresolution change detection algorithm given in Section 3.1.

(c) Here, in addition to the normalized frame difference (3), define a motion reliability measure as the reciprocal of the condition number of the coefficient matrix in (6), given by [24]

$$R(\mathbf{x}, k) = \frac{\lambda_{min}}{\lambda_{max}} \tag{7}$$

where λ_{min} and λ_{max} are the smallest and largest eigenvalue of the coefficient matrix. A pixel is classified as stationary at a resolution level if its normalized frame difference is low, and its motion reliability is high. This step defines the new region of analysis.

(iii) Estimate the parameters of a higher order motion model (affine, perspective, or quadratic) over the new region of analysis as in [24]. Iterate over steps (ii) and (iii) until a satisfactory segmentation is attained.

4.1.2 Temporal Integration

Temporal continuity of the estimated dominant objects can be facilitated by extending the temporal integration scheme introduced in Section 3.2. To this effect, we define an internal representation image [24]

$$\bar{s}(\mathbf{x}, k) = (1 - \alpha)s(\mathbf{x}, k) + \alpha\ warp(\bar{s}(\mathbf{x}, k-1), s(\mathbf{x}, k)), k = 1, \ldots \tag{8}$$

where

$$\bar{s}(\mathbf{x}, 0) = s(\mathbf{x}, 0)$$

and *warp*(A, B) denotes warping image A towards image B according to the dominant motion parameters estimated between images A and B, and $0 < \alpha < 1$. As in the case of change detection, the unchanged regions in $\bar{s}(\mathbf{x}, k)$ maintain their sharpness with a reduced level of noise, while the changed regions are blurred after processing a few frames.

The algorithm to track the dominant object across multiple frames can be summarized as follows [24]: For each frame

(i) Compute the dominant motion parameters between the internal representation image $\bar{s}(\mathbf{x}, k)$ and the new frame $s(\mathbf{x}, k)$ within the support M_{k-1} of the dominant object at the previous frame.

(ii) Warp the internal representation image at frame $k-1$ towards the new frame according to the computed motion parameters.

(iii) Detect the stationary regions between the registered images as described in Section 4.1.1 using M_{k-1} as an initial estimate to compute the new mask M_k.

(iv) Update the internal representation image using (8).

Comparing each new frame with the internal representation image as opposed to the previous frame allows the method to track the same object. This is because the noise is significantly filtered in the internal representation image of the tracked object, and the image gradients outside the tracked object are lowered due to blurring. Note that there is no temporal motion constancy assumption in this tracking scheme.

4.1.3 Multiple Motions

Multiple object segmentation can be achieved by repeating the same procedure on the residual image after each object is extracted. Once the first dominant object is segmented and tracked, the procedure can be repeated recursively to segment and track the next dominant object after excluding all pixels belonging to the first object from the region of analysis. Hence, the method is capable of segmenting multiple moving objects in a top-down fashion if a dominant motion exists at each stage.

Some difficulties with the dominant motion approach were reported when there's no overwhelmingly dominant motion. Then, in the absence of competing motion models, the dominant motion approach can lead to arbitrary decisions (relying upon absolute threshold values) that are irrevocable, especially when the motion measure indicates unreliable motion vectors (in low spatial gradient regions). Sawhney et al. [32] proposed using robust estimators to partially alleviate this problem.

4.2 Multiple Motion Segmentation

Multiple motion segmentation methods let multiple motion models compete against each other at each decision site. They consist of three basic steps, which are strongly interrelated: estimation of the number K of independent motions, estimation of model parameters for each motion, and determination of support of each model (segmentation labels). If we assume that we know the number K of motions and the K sets of motion parameters, then we can determine the support of each model. The segmentation procedure then assigns the label of the parametric motion vector that is closest to the estimated flow vector at each site. Alternatively, if we assume that we know the value of K and a segmentation map consisting of K regions, the parameters for each model can be computed in the least squares sense (either from estimated flow vectors or from spatio-temporal intensity values) over the support of the respective region. But since both the parameters and supports are unknown in reality, we have

a chicken-egg problem; that is, we need to know the motion model parameters to find the segmentation labels, and the segmentation labels are needed to find the motion model parameters.

Various approaches exist in the literature for solving this problem by iterative procedures. They may be grouped as: segmentation by clustering in the motion parameter space [28, 37, 44], maximum likelihood (ML) segmentation [38, 40, 42], and maximum a posteriori probability (MAP) segmentation [34], which are covered in Sections 4.2.1–4.2.3, respectively. Pixel-based segmentation methods suffer from the drawback that the resulting segmentation maps may contain isolated labels. Spatial continuity constraints in the form of Gibbs random field (GRF) models have been introduced to overcome this problem within the MAP formulation [34]. However, the computational cost of these algorithms may be prohibitive. Furthermore, they do not guarantee that the estimated motion boundaries coincide with spatial color edges (object boundaries). Section 4.2.4 presents an alternative region labeling approach to address this problem.

4.2.1 Clustering in the Motion Parameter Space

A simple segmentation strategy is to first determine the number K of models (motion hypotheses) that are likely to be observed in a sequence, and then perform clustering in the model parameter space (e.g., a six dimensional space for the case of affine models) to find K models representing the motion. In the following, we study two distinct approaches in this class: the K-means method and the Hough transform method.

K-Means Method. Wang and Adelson (W-A) [37] employed K-means clustering for segmentation in their layered video representation. W-A method starts by partitioning the image into non-overlapping blocks uniformly distributed over the image, and fits an affine model to the estimated motion field (optical flow) within each block. In order to determine the reliability of the parameter estimates at each block, the sum of squared distances between the synthesized and estimated flow vectors is computed as

$$\bar{\eta}^2 = \sum_{\mathbf{x}\in\mathcal{B}} ||\mathbf{v}(\mathbf{x}) - \tilde{\mathbf{v}}(\mathbf{x})||^2 \tag{9}$$

where $\mathcal{B}$ refers to a block of pixels. Obviously, if the flow within the block complies with a single affine model, the residual will be small. On the other hand, if the block falls on the boundary between two distinct motions, the residual will be large. The motion parameters for blocks with acceptably small residuals are selected as the seed models. Then, the seed model parameter vectors are clustered to find the K representative affine motion models. The clustering procedure can be described as: Given N seed affine parameter vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N$, where

$$\mathbf{A}_n = \begin{bmatrix} a_{n,1} \\ a_{n,2} \\ a_{n,3} \\ a_{n,4} \\ a_{n,5} \\ a_{n,6} \end{bmatrix}, n = 1, \ldots, N, \tag{10}$$

find K cluster centers $\bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \ldots, \bar{\mathbf{A}}_K$, where $K \ll N$, and the label k, $k = 1, \ldots, K$, assigned to each affine parameter vector $\mathbf{A}_n$ which minimizes

$$\sum_{n=1}^{N} \mathcal{D}(\mathbf{A}_n, \bar{\mathbf{A}}_k) \tag{11}$$

The distance measure $\mathcal{D}$ between two affine parameter vectors $\mathbf{A}_n$ and $\mathbf{A}_k$ is given by

$$\mathcal{D}(\mathbf{A}_n, \mathbf{A}_k) = \mathbf{A}_n^T \mathbf{M} \mathbf{A}_k \tag{12}$$

where $\mathbf{M}$ is a 6×6 scaling matrix.

The solution to this problem is given by the well-known K-means algorithm, which consists of the following iteration:

(i) Initialize $\bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \ldots, \bar{\mathbf{A}}_K$ arbitrarily.

(ii) For each seed block n, $n = 1, \ldots, N$, find k given by

$$k = Arg \min_s \mathcal{D}(\mathbf{A}_n, \bar{\mathbf{A}}_s) \tag{13}$$

where s takes values from the set $\{1, 2, \ldots, K\}$. It should be noted that if the minimum distance exceeds a threshold, then the site is not labeled, and the corresponding flow vector is ignored in the parameter update that follows.

(iii) Define $\mathcal{S}_k$ as the set of seed blocks whose affine parameter vector is closest to $\bar{\mathbf{A}}_k$, $k = 1, \ldots, K$. Then, update the class means

$$\bar{\mathbf{A}}_k = \frac{\sum_{n\in\mathcal{S}_k} \mathbf{A}_n}{\sum_{n\in\mathcal{S}_k} 1} \tag{14}$$

(iv) Repeat steps (ii) and (iii) until the class means $\bar{\mathbf{A}}_k$ do not change by more than a predefined amount between successive iterations.

Statistical tests can be applied to eliminate some parameter vectors that are deemed as outliers.

Once the K cluster centers are determined, a label assignment procedure is employed to assign a segmentation label $z(\mathbf{x})$ to each pixel $\mathbf{x}$ as

$$z(\mathbf{x}) = Arg \min_k \| \mathbf{v}(\mathbf{x}) - \mathcal{P}(\bar{\mathbf{A}}_k; (\mathbf{x})) \|^2 \quad (15)$$

where k is from the set $\{1, 2, \ldots, K\}$, the operator $\mathcal{P}$ is defined as

$$\mathcal{P}(\bar{\mathbf{A}}_k; (\mathbf{x})) = \begin{bmatrix} (\bar{a}_{k,1} - 1)x_1 + \bar{a}_{k,2}x_2 + \bar{a}_{k,3} \\ \bar{a}_{k,4}x_1 + (\bar{a}_{k,5} - 1)x_2 + \bar{a}_{k,6} \end{bmatrix} \quad (16)$$

and $\mathbf{v}(\mathbf{x})$ is the dense motion vector at pixel $\mathbf{x}$ given by

$$\mathbf{v}(\mathbf{x}) = \begin{bmatrix} v_1(\mathbf{x}) \\ v_2(\mathbf{x}) \end{bmatrix} \quad (17)$$

where v_1 and v_2 denote the horizontal and vertical components, respectively. All sites without labels are assigned one according to the motion compensation criterion, which assigns the label of the parameter vector that gives the best motion compensation at that site. This feature ensures more robust parameter estimation by eliminating the outlier vectors. Several post-processing operations may be employed to improve the accuracy of the segmentation map. The procedure can be repeated by estimating new seed model parameters over the regions estimated in the previous iteration. Furthermore, the number of clusters can be varied by splitting or merging of clusters between iterations. The K-means method requires a good initial estimate of the number of classes K. The Hough transform methods do not require this information but are more expensive.

Hough Transform Methods. The Hough transform is a well-known clustering technique where the data samples "vote" for the most representative feature values in a quantized feature space. In a straightforward application of the Hough transform method to optical flow segmentation using the six-parameter affine flow model (16), the six-dimensional feature space $a_1, \ldots, a_6$ would be quantized to certain parameter states after the minimal and maximal values for each parameter are determined. Then, each flow vector $\mathbf{v}(\mathbf{x}) = [v_1(\mathbf{x})v_2(\mathbf{x})]^T$ votes for a set of quantized parameters which minimizes

$$\eta^2(\mathbf{x}) \doteq \eta_1^2(\mathbf{x}) + \eta_2^2(\mathbf{x}) \quad (18)$$

where $\eta_1(\mathbf{x}) = v_1(\mathbf{x}) - a_1 - a_2x_1 - a_3x_2$ and $\eta_2(\mathbf{x}) = v_2(\mathbf{x}) - a_4 - a_5x_1 - a_6x_2$. The parameter sets that receive at least a predetermined amount of votes are likely to represent candidate motions. The number of classes K and the corresponding parameter sets to be used in labeling individual flow vectors are hence determined. The drawback of this scheme is the significant amount of computation and memory requirements involved.

In order to keep the computational burden at a reasonable level, several modified Hough methods have been presented. Proposed simplifications to ease the computational load include: [28] (i) decomposition of the parameters space into two disjoint subsets $\{a_1, a_2, a_3\} \times \{a_4, a_5, a_6\}$ to perform two 3D Hough transforms, (ii) a multiresolution Hough transform, where at each resolution level the parameter space is quantized around the estimates obtained at the previous level, and (iii) a multipass Hough technique, where the flow vectors which are most consistent with the candidate parameters are grouped first. In the second stage, those components formed in the first stage which are consistent with the same flow model in the least squares sense are merged together to form segments. Several merging criteria have been proposed. In the third and final stage, ungrouped flow vectors are assimilated into one of their neighboring segments. Other simplifications that are proposed include the probabilistic Hough transform [43] and randomized Hough transform [44].

Clustering in the parameter space has some drawbacks: (i) both methods rely on pre-computed optical flow as an input representation, which is generally blurred at motion boundaries and may contain outliers, (ii) clustering based on distances in the parameter space can lead to clustered parameters which are not physically meaningful and the results are sensitive to the choice of the weight matrix $\mathbf{M}$ and small errors in the estimation of affine parameters, and (iii) parameter clustering and label assignment procedures are decoupled; hence, ad-hoc post-processing operations which depend on some threshold values are needed to clean up the final segmentation map. The following section proposes a maximum likelihood segmentation method, which addresses all of these shortcomings.

4.2.2 Maximum-Likelihood Segmentation

Motion segmentation approaches in general are classified as optical flow segmentation methods, which operate on pre-computed optical flow estimates as an input representation, and direct methods, which operate on spatio-temporal intensity values. We present here a unified formulation that covers both cases. The maximum likelihood (ML) method finds the segmentation labels that maximize the likelihood function, which models the deviation of the observations (estimated dense motion vectors or observed intensity values) from a parametric description of them (parametric motion vectors or motion compensated intensity values, respectively) for a given motion model.

We start by defining the log-likelihood function as

$$L(\mathbf{o}|\mathbf{z}) = \log(p(\mathbf{o}|\mathbf{z})) \quad (19)$$

where $\mathbf{z}$ denotes the lexicographic ordering of the segmentation labels $\mathbf{z}(\mathbf{x})$, which takes values from the set $1, 2, \ldots, K$ at each pixel $\mathbf{x}$. The vector $\mathbf{o}$ stands for the lexicographic ordering of the observations, which are either estimated dense motion (optical flow) vectors or image intensity values. The conditional probability $p(\mathbf{o}|\mathbf{z})$ quantifies how well piecewise parametric motion modeling fits the observations $\mathbf{o}$ given the segmentation labels $\mathbf{z}$. If we model the mismatch between the observations $\mathbf{o}(\mathbf{x})$ and their parametric representations computed by the operator $\mathcal{O}(\mathbf{A}_{z(\mathbf{x})}; \mathbf{x})$,

$$\eta = \mathbf{o}(\mathbf{x}) - \mathcal{O}(\mathbf{A}_{z(\mathbf{x})}; \mathbf{x}) \tag{20}$$

where $\mathbf{A}_k$ denotes the kth parametric motion model, by white Gaussian noise with zero mean and variance σ^2, then the conditional pdf of the observations given the segmentation labels can be expressed as

$$p(\mathbf{o}|\mathbf{z}) = \frac{1}{(2\pi\sigma^2)^{M/2}} \exp\left\{ -\sum_{i=1}^{M} \eta^2(\mathbf{x}_i)/2\sigma^2 \right\} \tag{21}$$

where M is the number of observations available at the sites $\mathbf{x}_i$. Assuming that the parametric flow model is more or less accurate, this deviation is due to presence of observation noise (given correct segmentation labels). Then, the problem is to find K motion models $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_K$, and a label field $z(\mathbf{x})$ to maximize the log-likelihood function $L(\mathbf{o}|\mathbf{z})$.

We consider two cases:

I. Pre-computed optical flow segmentation: The observation $\mathbf{o}(\mathbf{x})$ stands for the estimated dense motion vectors $\mathbf{v}(\mathbf{x})$, and the operator $\mathcal{O}$ stands for the parametric motion operator $\mathcal{P}$ given by Eq. (16) or a higher-order model given by

$$\begin{aligned} \tilde{v}_1(\mathbf{x}) &= a_1 x_1 + a_2 x_2 - a_3 + a_7 x_1^2 + a_8 x_1 x_2 \\ \tilde{v}_2(\mathbf{x}) &= a_4 x_1 + a_5 x_2 - a_6 + a_7 x_1 x_2 + a_8 x_2^2 \end{aligned} \tag{22}$$

Then,

$$\eta^2(\mathbf{x}_i) = (v_1(\mathbf{x}_i) - \tilde{v}_1(\mathbf{x}_i))^2 + (v_2(\mathbf{x}_i) - \tilde{v}_2(\mathbf{x}_i))^2 \tag{23}$$

is the norm-squared deviation of the actual flow vectors from what is predicted by the quadratic flow model. This case concerns motion segmentation by motion vector matching.

II. Direct segmentation: The observation $\mathbf{o}(\mathbf{x})$ stands for the scalar pixel intensities $I_t(\mathbf{x})$ at frame t, and the operator $\mathcal{O}$ is the motion-compensation operator $\mathcal{Q}$, defined by

$$\mathcal{Q}(\mathbf{A}_{z(\mathbf{x})}; \mathbf{x}) = \mathbf{I}_{t-1}(\mathbf{x}') \tag{24}$$

where

$$\mathbf{x}' = [x_1' \; x_2']^T = [x_1 \; x_2]^T + \mathcal{P}(\mathbf{A}_{z(\mathbf{x})}; \mathbf{x}) \tag{25}$$

Then,

$$\eta^2(\mathbf{x}_i) = (I_t(\mathbf{x}) - I_{t-1}(\mathbf{x}'))^2 \tag{26}$$

This case corresponds to motion segmentation by motion-compensated intensity matching. The motion parameters $\mathbf{A}_k$ are estimated over the support of model k using direct methods (see step (iii) below).

In either case, assuming that the variances for all classes are the same, maximization of the log-likelihood function is equivalent to minimization of the cost function

$$\sum_{x_1, x_2} \| \mathbf{o}(\mathbf{x}) - \mathcal{O}(\mathbf{A}_{z(\mathbf{x})}; \mathbf{x}) \|^2 \tag{27}$$

or equivalently

$$\sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{Z}_k} \| \mathbf{o}(\mathbf{x}) - \mathcal{O}_k(\mathbf{x}) \|^2 \tag{28}$$

where $\mathcal{Z}_k$ is the set of pixels $\mathbf{x}$ with the motion label $z(\mathbf{x}) = k$, and $\mathcal{O}_k(\mathbf{x}) \doteq \mathcal{O}(\mathbf{A}_k; \mathbf{x})$.

A two-step iterative solution to this problem is given by

(i) Initialize $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_K$.

(ii) Assign a motion label $z(\mathbf{x})$ to each pixel $\mathbf{x}$ as

$$z(\mathbf{x}) = Arg \min_k \| \mathbf{o}(\mathbf{x}) - \mathcal{O}(\mathbf{A}_k; \mathbf{x}) \|^2 \tag{29}$$

where k takes values from the set $\{1, 2, \ldots, K\}$.

(iii) Update $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_K$ as

$$\mathbf{A}_k = Arg \min_{\mathbf{A}} \sum_{\mathbf{x} \in \mathcal{Z}_k} \| \mathbf{v}(\mathbf{x}) - \mathcal{P}(\mathbf{A}; \mathbf{x}) \|^2 \tag{30}$$

This minimization is equivalent to least squares estimation of the affine motion model fit to the motion vectors with the label $z(\mathbf{x}) = k$. A closed form solution to this problem can be expressed in terms of a linear matrix equation

$$\begin{bmatrix} x_1 & x_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & x_2 & 1 \end{bmatrix} \begin{bmatrix} a_{k,1} \\ a_{k,2} \\ a_{k,3} \\ a_{k,4} \\ a_{k,5} \\ a_{k,6} \end{bmatrix} = \begin{bmatrix} x_1 + v_1(\mathbf{x}) \\ x_2 + v_2(\mathbf{x}) \end{bmatrix} \tag{31}$$

for all $\mathbf{x}$ such that $z(\mathbf{x}) = k$.

(iv) Repeat steps (ii) and (iii) until the class means $\mathbf{A}_k$ do not change by more than a predefined amount between successive iterations.

This method does not require gradient-based optimization or other numeric search procedures for optimization of a cost function. Thus, it is robust and computationally efficient. Extensions of this formulation using mixture modeling and robust estimators, that require gradient-based optimization, have also been proposed [38].

Motion vector matching is a good motion segmentation criterion when the estimated motion field is accurate; that is, all outlier motion estimates are properly eliminated. Motion-compensated intensity matching is a more suitable criterion when spatial intensity (color) variations are sufficient and/or a multiresolution labeling procedure is employed. A possible limitation of ML segmentation framework is that it lacks constraints to enforce spatial and temporal continuity of the segmentation labels. Thus, rather ad-hoc steps are needed to eliminate small, isolated regions in the segmentation label field. The maximum a posteriori probability (MAP) segmentation strategy promises to impose continuity constraints in an optimization framework.

4.2.3 Maximum *a Posteriori* Probability Segmentation

The MAP method is a ***Bayesian*** approach that searches for the maximum of the *a posteriori* pdf of the segmentation labels given the observations (either pre-computed optical flow or spatio-temporal intensity data). This pdf is not only a measure of how well the segmentation labels explain the observed data, but also how well they conform with our prior expectations. The MAP formulation differs from the maximum likelihood approach in that it includes smoothness terms to enforce spatial continuity of the output motion segmentation map.

The *a posteriori* pdf $p(\mathbf{z}|\mathbf{o})$ of the segmentation label field $\mathbf{z}$ given the observed data $\mathbf{o}$ can be expressed, using the Bayes theorem, as

$$p(\mathbf{z}|\mathbf{o}) = \frac{p(\mathbf{o}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{o})} \tag{32}$$

where $p(\mathbf{o}|\mathbf{z})$ is the conditional pdf of the optical flow data given the segmentation $\mathbf{z}$, and $p(\mathbf{z})$ is the *a priori* pdf of the segmentation. Observe that, (i) $\mathbf{z}$ is a discrete-valued random vector with a finite sample space Ω, and (ii) $p(\mathbf{o})$ is constant with respect to the segmentation labels, and hence can be ignored for the purpose of computing $\mathbf{z}$. The MAP estimate, then, maximizes the numerator of (32) over all possible realizations of the segmentation field $\mathbf{z} = \omega$, $\omega \in \Omega$.

Modeling of the conditional pdf $p(\mathbf{o}|\mathbf{z})$ has been discussed in detail in Section 4.2.2 through (21) and (23) or (26). The prior pdf is modeled by a Gibbs distribution, which effectively introduces local constraints on the segmentation. It is given by

$$p(\mathbf{z}) = \frac{1}{Q}\sum_{\omega\in\Omega} \exp\{-U(\mathbf{z})\}\delta(\mathbf{z}-\omega) \tag{33}$$

where Ω denotes the sample space of the discrete-valued random vector $\mathbf{z}$, Q is the partition function (normalization constant) given by

$$Q = \sum_{\omega\in\Omega} \exp\{-U(\omega)\} \tag{34}$$

$U(\mathbf{z})$ is the potential function given by

$$U(\mathbf{z}) = \sum_{\mathbf{x}_i}\sum_{\mathbf{x}_j\in\mathcal{N}_{\mathbf{x}_i}} V_C(z(\mathbf{x}_i), z(\mathbf{x}_j)) \tag{35}$$

which can be expressed as a sum of local clique potential functions, such as

$$V_C(z(\mathbf{x}_i), z(\mathbf{x}_j)) = \begin{cases} -\gamma & \text{if } z(\mathbf{x}_i) = z(\mathbf{x}_j) \\ +\gamma & \text{otherwise} \end{cases} \tag{36}$$

and $\mathcal{N}_{\mathbf{x}_i}$ denotes the neighborhood system for the label field. Prior constraints on the structure of the segmentation labels, such as spatial smoothness, can be specified in terms of the clique potential function. Temporal continuity of the labels can similarly be modeled [34].

Substituting (21) and (33) into the criterion (32) and taking the logarithm of the resulting expression, maximization of the *a posteriori* probability distribution can be performed by minimizing the cost function

$$E = \frac{1}{2\sigma^2}\sum_{i=1}^{M} \eta^2(\mathbf{x}_i) + U(\omega) \tag{37}$$

The first term describes how well the predicted data fit the actual measurements (estimated optical flow vectors or image intensity values), and the second term measures how well the segmentation conforms to our prior expectations.

Because the motion model parameters corresponding to each label are not known *a priori*, the MAP segmentation must alternate between estimation of the model parameters and assignment of the segmentation labels to optimize the cost function (37). Murray and Buxton [34] were the first to propose a MAP segmentation method where the optical flow was modeled by a piecewise quadratic flow field (22), and the segmentation labels were assigned based on a simulated annealing (SA) procedure. Given the estimated flow field $\mathbf{v}$ and the number of independent motion models K, the MAP

segmentation using the Metropolis algorithm can be summarized as follows:

(i) Start with an initial labeling $\mathbf{z}$ of the optical flow vectors. Calculate the model parameters $\mathbf{a} = [a_1 \ \ldots \ a_8]^T$ for each region using least squares fitting (similar to that in Section 4.2.2). Set the initial temperature for SA.

(ii) Update the segmentation labels at each site $\mathbf{x}_i$ as follows:

(a) Perturb the label $z_i = z(\mathbf{x}_i)$ randomly.

(b) Decide whether to accept or reject this perturbation, based on the change ΔE in the cost function (37),

$$\Delta E = \frac{1}{2\sigma^2}\Delta\eta^2(\mathbf{x}_i) + \sum_{\mathbf{x}_j \in \mathcal{N}_{\mathbf{x}_i}} \Delta V_C(z(\mathbf{x}_i), z(\mathbf{x}_j)) \qquad (38)$$

where $\mathcal{N}_{\mathbf{x}_i}$ denotes a neighborhood of the site $\mathbf{x}_i$ and $V_C(z(\mathbf{x}_i), z(\mathbf{x}_j))$ is given by Equation (36). The first term indicates whether or not the perturbed label is more consistent with the given flow field determined by the residual (23), and the second term reflects whether or not it is in agreement with the prior segmentation field model.

Because the update at each site is dependent on the labels of the neighboring sites, the order in which the sites are visited affects the result of this step.

(iii) After all pixel sites are visited once, re-estimate the mapping parameters for each region based on the new segmentation label configuration.

(iv) Exit if a stopping criterion is satisfied. Otherwise, lower the temperature according to a predefined temperature schedule, and go to step (ii).

We can make the following observations: (i) The MAP method carries a high computational cost. (ii) The procedure proposed by Murray-Buxton suggests performing step (iii) above, the model parameter update, after each and every perturbation. We did not notice a significant difference in performance if motion parameter updates are done after all sites are visited once. (iii) The method can be applied with any parametric motion model, although the original formulation has been developed on the basis of the eight-parameter model.

4.2.4 Region-based Label Assignment

In this section, we extend the ML approach (Section 4.2.2) to region-based motion segmentation, where the image is first divided into predefined homogeneous regions, and then, at every iteration, each region is assigned a single motion label. This region-based label assignment strategy facilitates obtaining spatially continuous segmentation maps that are closely related to actual object boundaries, without the heavy computational burden of statistical Markov random field (MRF) model-based approaches. The predefined regions

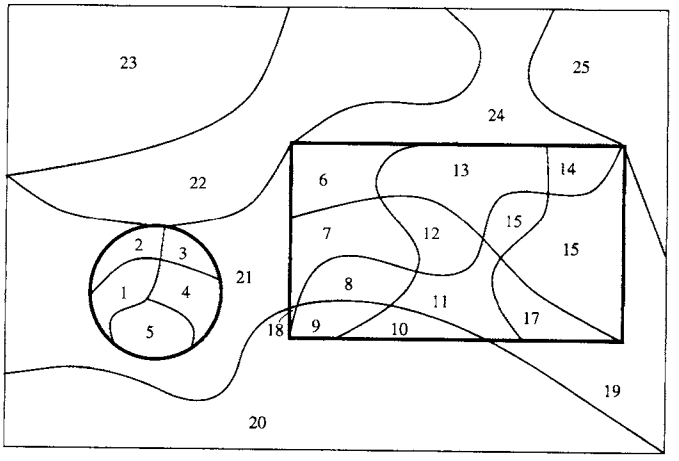

FIGURE 1 Illustration of the observation that color segments are generally subsets of motion segments. Here, the bold lines indicate motion segment boundaries, and each motion segment is composed of many color regions.

should be such that each region has a single motion. It is generally true that motion boundaries coincide with color segment boundaries, but not vice versa; i.e., color segments are almost always a subset of motion segments as illustrated in Fig. 1. Therefore, one can first perform a color segmentation to obtain a set of candidate motion segments. Other approaches to region definition include mesh-based partitioning of the scene [4] and macro pixels ($N \times N$ blocks) to improve the robustness of the ML motion segmentation. Here, we assume that each frame of video has been subject to a region formation procedure. We let $C(\mathbf{x})$ denote the region map of a frame consisting of M mutually exclusive and exhaustive regions, and define $\mathcal{C}_m$ as the set of pixels $\mathbf{x}$ with the region label $C(\mathbf{x}) = m$, $m = 1, \ldots, M$.

We wish to find the motion segmentation map $\mathbf{z}$ (a vector formed by lexicographic ordering of $z(\mathbf{x})$) and the corresponding affine parameter vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_K$ which best fit the dense motion-vector field, such that [42]

$$\sum_{m=1}^{M} \sum_{\mathbf{x} \in \mathcal{C}_m} \| \mathbf{v}(\mathbf{x}) - \mathcal{P}(\mathbf{A}_{z(m)}; \mathbf{x}) \|^2 \qquad (39)$$

is minimized. Here $z(m)$ refers to the motion label of all pixels within $\mathcal{C}_m$ and takes one of the values $1, 2, \ldots, K$; $\mathcal{P}$ is an operator defined by Eq. (16), and $\mathbf{v}(\mathbf{x})$ is the dense motion vector at pixel $\mathbf{x}$ as defined by Eq. (17). The procedure is given by:

(i) Initialize the motion segmentation map $\mathbf{z}$ by assigning a single motion label k, $k = 1, \ldots, K$ to each $\mathcal{C}_m$.

(ii) Update the parameter vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_K$ as

$$\mathbf{A}_k = Arg \min_{\mathbf{A}} \sum_{\mathbf{x} \in \mathcal{Z}_k} \| \mathbf{v}(\mathbf{x}) - \mathcal{P}(\mathbf{A}; \mathbf{x}) \|^2 \qquad (40)$$

where $\mathcal{Z}_k$ is the set of pixels $\mathbf{x}$ with the label $z(\mathbf{x}) = k$. This minimization can be achieved by solving the linear matrix equation

$$\begin{bmatrix} x_1 & x_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & x_2 & 1 \end{bmatrix} \begin{bmatrix} a_{k,1} \\ a_{k,2} \\ a_{k,3} \\ a_{k,4} \\ a_{k,5} \\ a_{k,6} \end{bmatrix} = \begin{bmatrix} v_1(\mathbf{x}) \\ v_2(\mathbf{x}) \end{bmatrix} \tag{41}$$

for all $\mathbf{x}$ in $\mathcal{Z}_k$.

(iii) Assign a motion label to each region $\mathcal{C}_m$, $m = 1, 2, \ldots, M$, such that

$$z(\mathcal{C}_m) = Arg \min_k \sum_{\mathbf{x} \in \mathcal{C}_m} \| \mathbf{o}(\mathbf{x}) - \mathcal{O}(\mathbf{A}_k; \mathbf{x}) \|^2 \tag{42}$$

where $k = 1, 2, \ldots K$ and $\mathbf{o}(\mathbf{x})$ and $\mathcal{O}(\mathbf{x})$ are as defined in Section 4.2.2. This allows region-based affine motion segmentation with pixel-based motion-vector or intensity matching.

(iv) Repeat steps (ii) and (iii) until the class means $\mathbf{A}_k$ do not change by more than a predefined amount between successive iterations.

We note that the pixel-based ML motion segmentation method presented in Section 4.2.2 is a special case of this region-based framework. If each region $\mathcal{C}_m$ contains a single pixel, then the iterations are carried over individual pixels, and the motion label assignment is performed at each pixel independently.

We conclude this section by observing that the methods discussed here that used pre-computed optical flow as an input representation are limited by the accuracy of the available optical flow estimates. Next, we introduce a framework, in which optical flow estimation and segmentation interact in a mutually beneficial manner.

5 Simultaneous Motion Estimation and Segmentation

Up to now, we discussed methods to compute the segmentation labels from either pre-computed optical flow or directly from intensity values, but did not address how to compute an improved dense motion field along with the segmentation map. It is clear that the success of optical flow segmentation is closely related to the accuracy of the estimated optical flow field (in the case of using pre-computed flow values), and vice versa. It follows that optical flow estimation and segmentation have to be addressed simultaneously for best results. Here, we present a simultaneous Bayesian approach based on a representation of the motion field as the sum of a parametric field and a residual field. The interdependence of optical flow and segmentation fields is expressed in terms of a Gibbs distribution within the MAP framework. The resulting optimization problem, to find estimates of a dense set of motion vectors, a set of segmentation labels, and a set of mapping parameters, is solved using the highest confidence first (HCF) and iterated conditional mode (ICM) algorithms.

5.1 Modeling

We model the optical flow field $\mathbf{v}(\mathbf{x})$ as the sum of a parametric flow field $\tilde{\mathbf{v}}(\mathbf{x})$ and a nonparametric residual field $\mathbf{v}_r(\mathbf{x})$, which accounts for local motion and other modeling errors; that is,

$$\mathbf{v}(\mathbf{x}) = \tilde{\mathbf{v}}(\mathbf{x}) + \mathbf{v}_r(\mathbf{x}) \tag{43}$$

The parametric component of the motion field clearly depends on the segmentation label $z(\mathbf{x})$, which takes on the values $1, \ldots, K$.

The simultaneous MAP framework aims at maximizing the *a posteriori* pdf

$$p(\mathbf{v}_1, \mathbf{v}_2, \mathbf{z} \mid \mathbf{g}_k, \mathbf{g}_{k+1}) = \frac{p(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z}) p(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z}, \mathbf{g}_k) p(\mathbf{z} \mid \mathbf{g}_k)}{p(\mathbf{g}_{k+1} \mid \mathbf{g}_k)} \tag{44}$$

with respect to the optical flow $\mathbf{v}_1$, $\mathbf{v}_2$ and the segmentation labels $\mathbf{z}$, where $\mathbf{v}_1$ and $\mathbf{v}_2$ denote the lexicographic ordering of the first and second components of the flow vectors $\mathbf{v}(\mathbf{x}) = [v_1(\mathbf{x}) v_2(\mathbf{x})]^T$ at each pixel $\mathbf{x}$. Through careful modeling of these pdfs, we can express an interrelated set of constraints that help improve both optical flow and segmentation estimates.

The first conditional pdf $p(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z})$ provides a measure of how well the present displacement and segmentation estimates conform with the observed frame $k+1$ given frame k. It is modeled by a Gibbs distribution as

$$p(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z}) = \frac{1}{Q_1} \exp\{-U_1(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z})\} \tag{45}$$

where Q_1 is the partition function (normalizing constant), and

$$U_1(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z}) = \sum_{\mathbf{x}} [g_k(\mathbf{x}) - g_{k+1}(\mathbf{x} + \mathbf{v}(\mathbf{x}) \Delta t)]^2 \tag{46}$$

is called the Gibbs potential. Here, the Gibbs potential corresponds to the norm-square of the displaced frame difference (DFD) between the frames $\mathbf{g}_k$ and $\mathbf{g}_{k+1}$. Thus, maximization of (45) imposes the constraint that $\mathbf{v}(\mathbf{x})$ minimizes the DFD.

The second term in the numerator in (44) is the conditional pdf of the displacement field given the motion segmentation and the search image. It is also modeled by a Gibbs distribution

$$p(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z}, \mathbf{g}_k) = p(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z}) = \frac{1}{Q_2} \exp\{-U_2(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z})\} \tag{47}$$

where Q_2 is a constant, and

$$U_2(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z}) = \alpha \sum_{\mathbf{x}} ||\mathbf{v}(\mathbf{x}) - \tilde{\mathbf{v}}(\mathbf{x})||^2 + \beta \sum_{\mathbf{x}_i} \sum_{\mathbf{x}_j \in \mathcal{N}_{\mathbf{x}_i}} ||\mathbf{v}(\mathbf{x}_i) - \mathbf{v}(\mathbf{x}_j)||^2 \delta(z(\mathbf{x}_i) - z(\mathbf{x}_j)) \tag{48}$$

is the corresponding Gibbs potential, $||\cdot||$ denotes the Euclidian distance, and $\mathcal{N}_{\mathbf{x}}$ is the set of neighbors of site $\mathbf{x}$. The first term in (48) enforces a minimum norm estimate of the residual motion field $\mathbf{v}_r(\mathbf{x})$; that is, it aims to minimize the deviation of the motion field $\mathbf{v}(\mathbf{x})$ from the parametric motion field $\tilde{\mathbf{v}}(\mathbf{x})$ while minimizing the DFD. Note that the parametric motion field $\tilde{\mathbf{v}}(\mathbf{x})$ is calculated from the set of model parameters $\mathbf{a}_i$, $i = 1, \ldots, K$, which in turn is a function of $\mathbf{v}(\mathbf{x})$ and $z(\mathbf{x})$. The second term in (48) imposes a piecewise local smoothness constraint on the optical flow estimates without introducing any extra variables such as line fields. Observe that this term is active only for those pixels in the neighborhood $\mathcal{N}_{\mathbf{x}}$ which share the same segmentation label with the site $\mathbf{x}$. Thus, spatial smoothness is enforced only on the flow vectors generated by a single object. The parameters α and β allow for relative scaling of the two terms.

The third term in (44) models the *a priori* probability of the segmentation field in a manner similar to that in MAP segmentation. It is given by

$$p(\mathbf{z} \mid \mathbf{g}_k) = p(\mathbf{z}) = \frac{1}{Q_3} \sum_{\omega \in \Omega} \exp\{-U_3(\mathbf{z})\} \delta(\mathbf{z} - \omega) \tag{49}$$

where Ω denotes the sample space of the discrete-valued random vector $\mathbf{z}$, and Q_3 and $U_3(\mathbf{z})$ are as defined in (34) and (35), respectively. The dependence of the labels on the image intensity is usually neglected, although region boundaries generally coincide with intensity edges.

5.2 An Algorithm

Maximizing the *a posteriori* pdf (44) is equivalent to minimizing the cost function,

$$E = U_1(\mathbf{g}_{k+1} \mid \mathbf{g}_k, \mathbf{v}_1, \mathbf{v}_2, \mathbf{z}) + U_2(\mathbf{v}_1, \mathbf{v}_2 \mid \mathbf{z}) + U_3(\mathbf{z}) \tag{50}$$

that is composed of the potential functions in Equations (45), (47), and (49). Direct minimization of (50) with respect to all unknowns is an exceedingly difficult problem, because the motion and segmentation fields constitute a large set of unknowns. To this effect, we perform the minimization of (50) through the following two-steps iterations [48]:

(i) Given the best available estimates of the parameters $\mathbf{a}_i$, $i = 1, \ldots, K$, and $\mathbf{z}$, update the optical flow field $\mathbf{v}_1$, $\mathbf{v}_2$. This step involves the minimization of a modified cost function

$$E_1 = \sum_{\mathbf{x}} [g_k(\mathbf{x}) - g_{k+1}(\mathbf{x} + \mathbf{v}(\mathbf{x})\Delta t)]^2 + \alpha \sum_{\mathbf{x}} ||\mathbf{v}(\mathbf{x}) - \tilde{\mathbf{v}}(\mathbf{x})||^2 + \beta \sum_{\mathbf{x}_i} \sum_{\mathbf{x}_j \in \mathcal{N}_{\mathbf{x}_i}} ||\mathbf{v}(\mathbf{x}_i) - \mathbf{v}(\mathbf{x}_j)||^2 \delta(z(\mathbf{x}_i) - z(\mathbf{x}_j)) \tag{51}$$

which is composed of all terms in (50) that contain $\mathbf{v}(\mathbf{x})$. While the first term indicates how well $\mathbf{v}(\mathbf{x})$ explains our observations, the second and third terms impose prior constraints on the motion estimates that they should conform with the parametric flow model, and that they should vary smoothly within each region. To minimize this energy function, we employ the HCF method recently proposed by Chou and Brown [49]. HCF is a deterministic method designed to efficiently handle the optimization of multivariable problems with neighborhood interactions.

(ii) Update the segmentation field $\mathbf{z}$, assuming that the optical flow field $\mathbf{v}(\mathbf{x})$ is known. This step involves the minimization of all the terms in (50) which contain $\mathbf{z}$ as well as $\tilde{\mathbf{v}}(\mathbf{x})$, given by

$$E_2 = \alpha \sum_{\mathbf{x}} ||\mathbf{v}(\mathbf{x}) - \tilde{\mathbf{v}}(\mathbf{x})||^2 + \sum_{\mathbf{x}_i} \sum_{\mathbf{x}_j \in \mathcal{N}_{\mathbf{x}_i}} V_C(z(\mathbf{x}_i), z(\mathbf{x}_j)) \tag{52}$$

The first term in (52) quantifies the consistency of $\tilde{\mathbf{v}}(\mathbf{x})$ and $\mathbf{v}(\mathbf{x})$. The second term is related to the *a priori* probability of the present configuration of the segmentation labels. We use an ICM procedure to optimize E_2 [48]. The mapping parameters $\mathbf{a}_i$ are updated by least squares estimation within each region.

An initial estimate of the optical flow field can be found by using the Bayesian approach with a global smoothness constraint. Given this estimate, the segmentation labels can be initialized by a procedure similar to Wang and Adelson's [37]. The determination of the free parameters α, β, and γ is

a design problem. One strategy is to choose them to provide a dynamic range correction so that each term in the cost function (50) has equal emphasis. However, because the optimization is implemented in two steps, the ratio α/γ also becomes of consequence. We recommend to select $1 \leq \alpha/\gamma \leq 5$, depending on how well the motion field can be represented by a piecewise-parametric model and whether we have a sufficient number of classes.

A hierarchic implementation of this algorithm is also possible by forming successive low-pass filtered versions of the images $\mathbf{g}_k$ and $\mathbf{g}_{k+1}$. Thus, the quantities $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{z}$ can be estimated at different resolutions. The results of each hierarchy are used to initialize the next lower level. Note that the Gibbsian model for the segmentation labels has been extended to include neighbors in scale by Kato et al. [53].

Several other motion analysis approaches can be formulated as special cases of this framework. If we retain only the first and the third terms in (50), and assume that all sites possess the same segmentation label, then we have Bayesian motion estimation with a global smoothness constraint. The motion estimation algorithm proposed by Iu [47] utilizes the same two terms, but replaces the $\delta(\cdot)$ function by a local outlier rejection function. The motion estimation and region labeling algorithm proposed by Stiller [52] involves all terms in (50), except the first term in (48). Furthermore, the segmentation labels in Stiller's algorithm are used merely as tokens to allow for a piecewise smoothness constraint on the flow field, and do not attempt to enforce consistency of the flow vectors with a parametric component. We also note that the motion estimation method of Konrad and Dubois [51] which uses line fields are fundamentally different in that they model discontinuities in the motion field, rather than modeling regions that correspond to different parametric motions. On the other hand, the motion segmentation algorithm of Murray and Buxton [34] (Section 4.2.3) employs only the second term in (48) and third term in (50) to model the conditional and prior pdf, respectively. Wang and Adelson [37] relies on the first term in (48) to compute the motion segmentation (Section 4.2.2). However, they also take the DFD of the parametric motion vectors into consideration when the closest match between the estimated and parametric motion vectors, represented by the second term, exceeds a threshold.

6 Semantic Video Object Segmentation

So far we discussed methods for automatic motion segmentation. However, it is difficult to achieve semantically meaningful object segmentation using fully automatic methods based on low level features such as motion, color, and texture. This is because a semantic object may contain multiple motions, colors, textures, and so on, and definition of semantic objects may depend on the context which may not be possible to capture by low level features. Thus, in this section, we present two approaches that can extract semantically meaningful objects using capture-specific information or user-interaction.

6.1 Chroma-Keying

Chroma-keying is an object-based video capture technology where each video object is recorded individually in a special studio against a key color. The key color is selected such that it does not appear on the object to be captured. Then, the problem of extracting the object from each frame of video becomes one of color segmentation. Chroma-keyed video capture requires special attention to avoid shadows and other non-uniformity in the key color within a frame; otherwise, segmentation of key color may become a nontrivial problem.

6.2 Semi-Automatic Segmentation

Because chroma-keying requires special studios and/or equipment to capture video objects, an alternative approach is interactive segmentation using automated tools to aid a human operator. To this effect, we assume that the contour of the first occurrence of the semantic object of interest is marked interactively by a human operator. While detection of moving regions (by change detection methods) may result in semantically meaningful objects in well-contsrained settings, in an unconstrained environment, user interaction is indeed the only way to define a semantically meaningful object unambiguously because only the user can know what is semantically meaningful in the context of an application. For example, if we have the video clip of a person carrying a ball, whether the ball and the person are two separate objects or a single object may depend on the application. Once the boundary of the object of interest is interactively determined in one or more keyframes, its boundary in all other frames can be automatically computed by 2D motion tracking until the object exits the field of view.

2D object tracking is closely related to the problem of spatio-temporal segmentation in the sense it provides temporally linked spatial segmentation maps. The general approach can be summarized as projecting the current segmentation map into the next frame using 2D motion information. The projected region can be updated by morphological or other operators using the color and edge information in the next frame. This update step allows fine tuning of the segmentation map to alleviate motion estimation errors as well as including newly uncovered regions in the segmentation map. Object tracking methods can be classified as feature-point-based, contour-based, and region-based tracking methods [4]. Feature points are points on the object (current segmentation map) that can be used as markers, such as corner points. Motion of these points can be found by gradient-based (e.g., Lukas-Kanade) or matching-based (e.g., block matching) methods. Goodness of tracking results at each feature point can be evaluated at each frame, and some feature points

can be removed and others may be added [54]. In contour-based methods, the tracking step defines a polygonal or spline approximation of the boundary of the video object, which may be further refined automatically or interactively using appropriate software tools [55, 56]. In region-based methods, the object region is repartitioned into color- and/or motion-homogeneous subregions, and each subpartition is projected into the next frame individually with or without using subregion connectivity constraints [57, 58].

7 Examples

Examples are shown for automatic motion segmentation using the pixel-based and region-based ML methods on two MPEG-4 test sequences: "Mother & Daughter" (frames 1-2) and "Mobile & Calendar"(frames 136-137). The former is an example of a slowly moving object against a still background, where mother's head is rotating while her body, the background and the child are stationary. The latter is a challenging sequence, with several distinctly moving objects such as a rotating ball, a moving train and a vertically translating calendar against a background that moves due to camera pan. Figure 2 (a) and (b) show the first and second frames of the "Mother & Daughter" sequence, and Fig. 2 (c) shows the estimated motion field between these frames. Figures 4 (a), (b), (c) show the corresponding pictures for frames 136 and 137 of the "Mobile & Calendar" sequence. Motion estimation was performed by using the hierarchical version of the Lucas-Kanade method [4] with three levels of hierarchy. In both cases, region definition by color segmentation is performed on the temporally second frame using the fuzzy c-means technique [41]. Each spatially disconnected piece of the color segmentation map was defined as an individual region. The resulting region maps are shown in Figs. 2 (d) and 4 (d), respectively.

Figure 3 demonstrates the performance of the ML method for foreground/background separation (i.e., $K=2$) with two different initializations. Figures 3 (e) and (f) show two possible initial segmentation maps, where the segmentation map is divided into two horizontal and vertical parts, respectively. Figures 3 (c) and (d) show the segmentation maps using

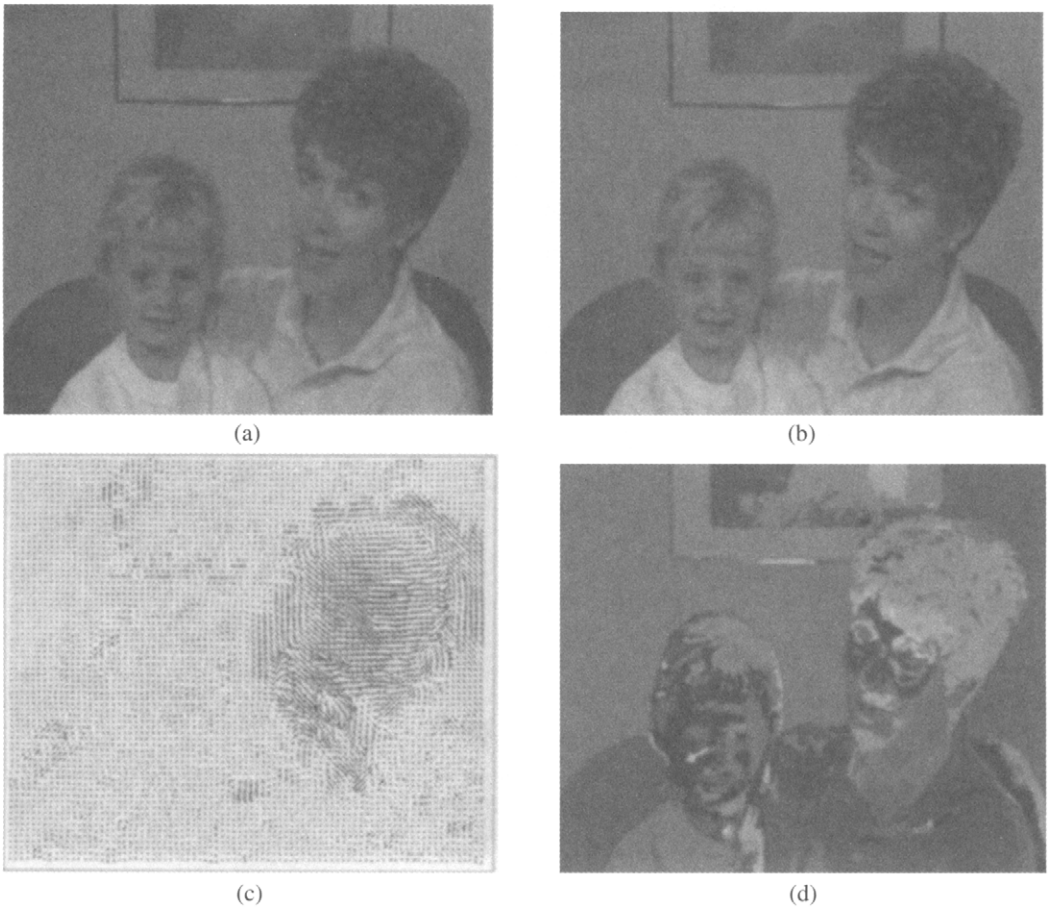

FIGURE 2 (a) The first and (b) second frames of the Mother and Daughter sequence; (c) 2D dense motion field from second frame to first frame; (d) region map obtained by color segmentation.

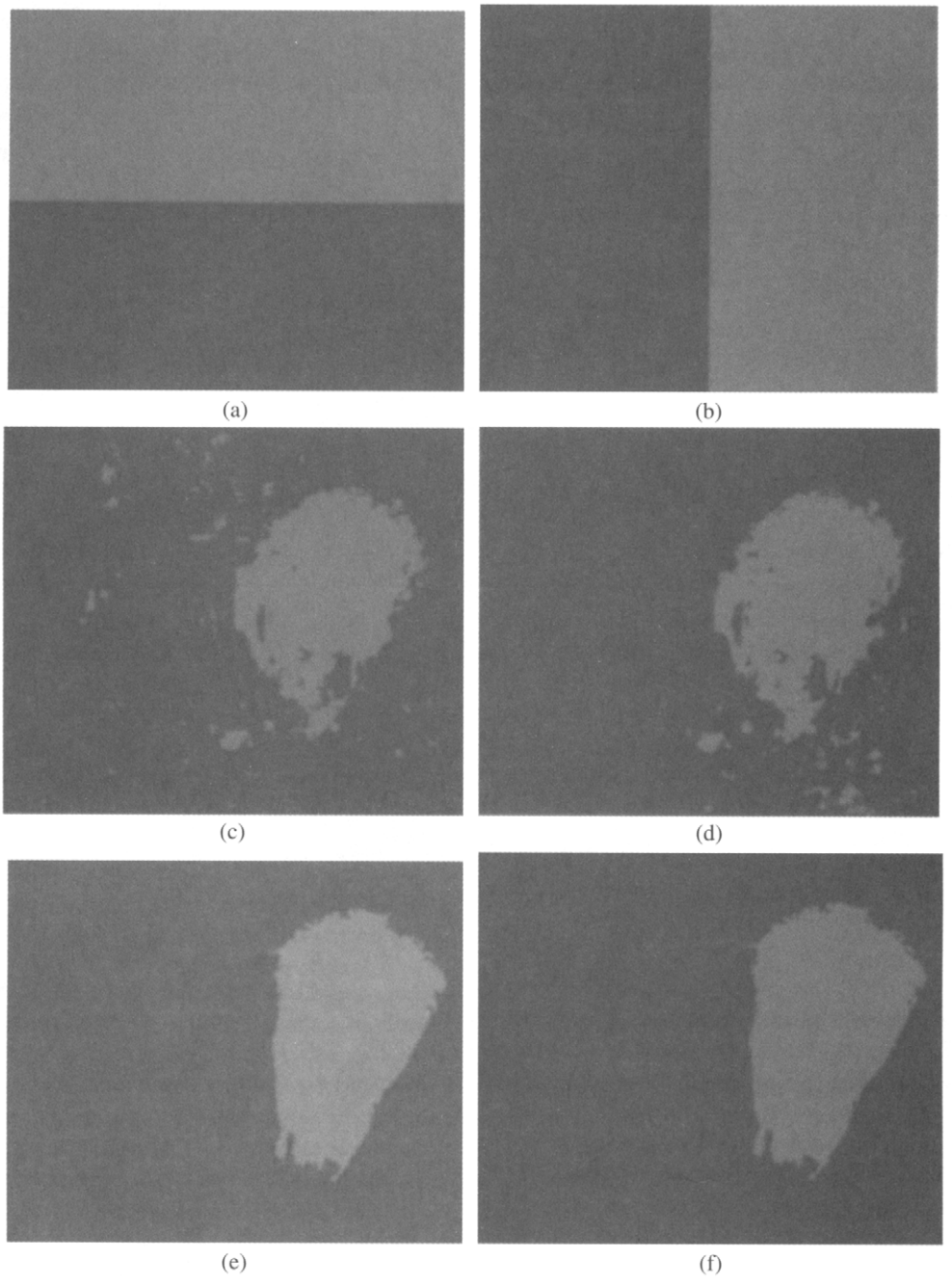

FIGURE 3 Results of the ML method with two different initializations: (a), (b) initial map; (c), (d) pixel-based motion-vector matching; (e), (f) region-based motion-vector matching.

pixel-based labeling by motion vector matching after 10 iterations starting from Figs. 3 (a) and (b), respectively. Figures 3 (e) and (f) show the results of region-labeling by motion vector matching starting with the affine parameter sets obtained from the maps Figs. 3 (c) and (d), respectively. Observe that the segmentation maps obtained by pixel-labeling contain many misclassified pixels, while the maps obtained by color-region-labeling are more coherent with the moving object in the scene.

Figure 5 illustrates the performance of the ML method with different number of initial segments, K. Figure 5 (a) and (b) show two initial segmentation maps with $K=4$ and $K=6$, respectively. The results of pixel-based labeling by motion vector-matching after 10 iterations for both initializations are depicted in Figs. 5 (c) and (d), respectively. Figure 5 (e) shows the result of region-based labeling using the color regions depicted in Fig. 4 (d) and the affine model parameters initialized by those computed from the

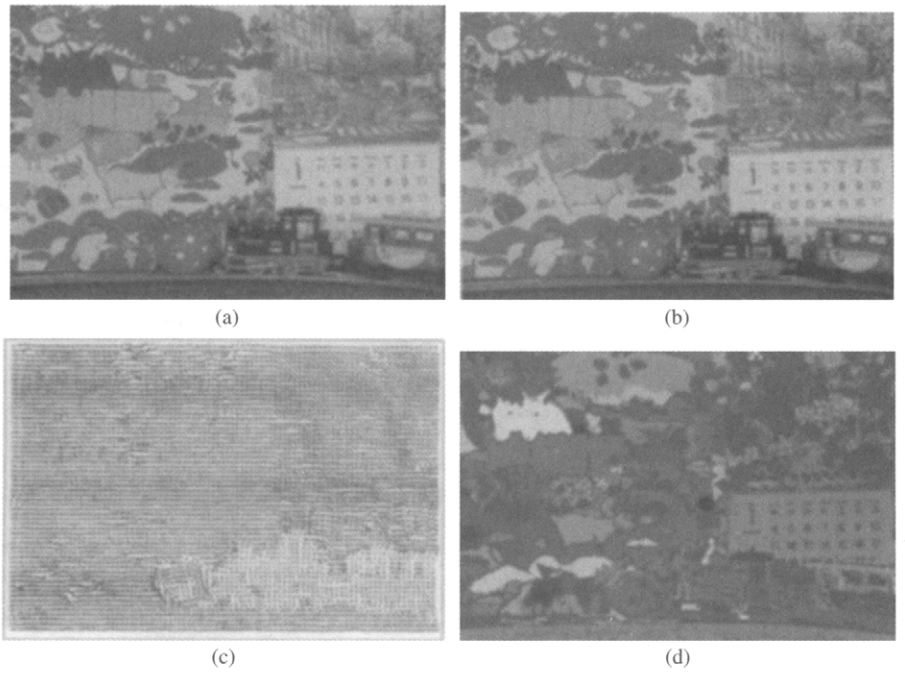

FIGURE 4 (a) The 136th and (b) 137th frames of the Mobile and Calendar sequence; (c) 2D dense motion field from 137th to 136th frame; (d) region map obtained by color segmentation.

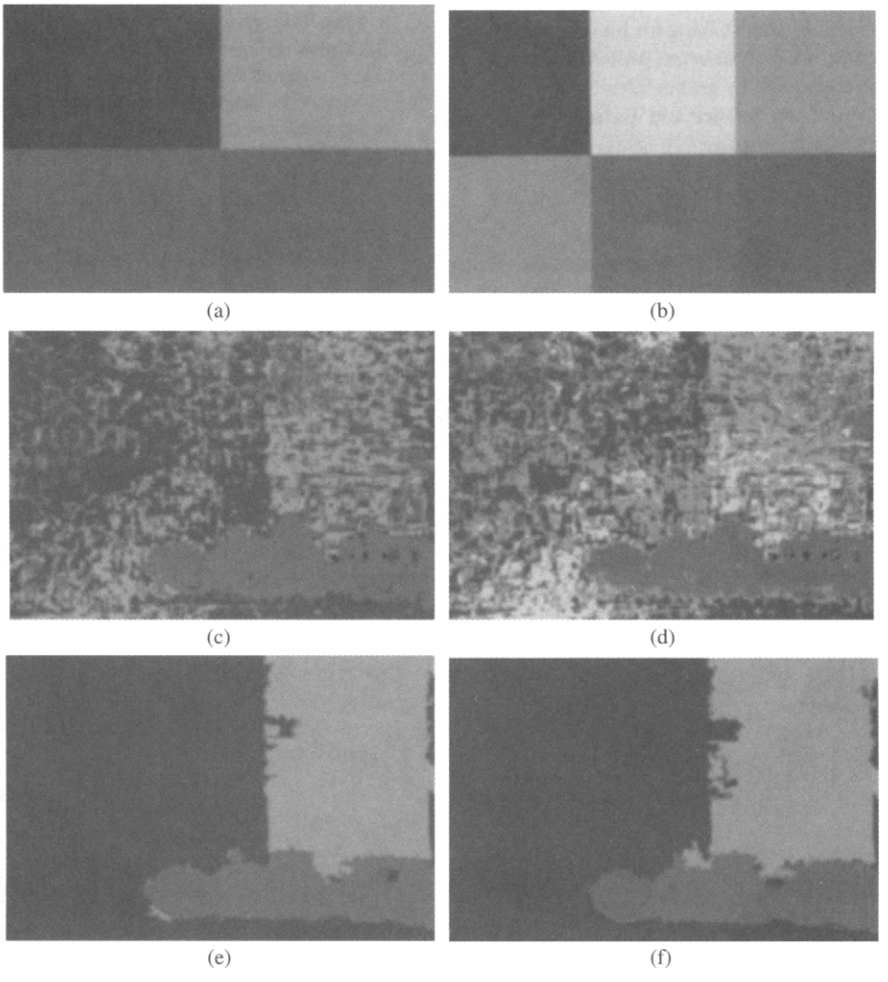

FIGURE 5 Results of the ML method: initial map (a) $k=4$, (b) $k=6$; pixel-based labeling (c) $k=4$, (d) $k=6$; region-based labeling (e) $k=4$, (f) $k=6$.

map in Fig. 5 (c) with $K=4$. We observed that this procedure results in oversegmentation when repeated with $K=6$. Therefore, we employ motion-compensated intensity matching and region merging to reduce the number of the motion classes if necessary. In this step, a region is merged with another if the latter set of affine parameters gives a comparable DFD as the former. The result of this final step is depicted in Fig. 5 (f) for $K=6$, where two of the six classes are eliminated by motion-compensated intensity matching. The ML segmentation method is computationally efficient since it does not require gradient-based optimization or any numeric search. It converges within approximately 10 iterations, and each iteration involves solution of only two 3×3 matrix equations. The complete procedure takes less than a minute to implement on a SparcStation 20.

It is difficult to associate a generic figure of merit with a video segmentation result. However, some measures have recently been proposed with or without ground-truth for automatic assesment of segmentation results [59, 60].

Acknowledgments

The author acknowledges Yucel Altunbasak and P. Erhan Eren for their contributions to the section on the maximum likelihood segmentation and Michael Chang for his contributions to the section on the maximum *a posteriori* probability segmentation. This work was supported by grants from National Science Foundation, New York State Science and Technology Foundation, and Eastman Kodak Company.

References

[1] N. Dimitrova, H. Zhang, B. Shahraray, I. Sezan, T, Huang, and A. Zakhor, "Applications of video content analysis and retrieval," *IEEE Multimedia*, 9, 42–55, July-Sept. 2002.

[2] P. Salembier and F. Marques, "Region-based representations of images and video: Segmentation tools for multimedia services," *IEEE Trans. Circ. Syst. for Video Tech.*, 9, 8, 1147–1169, Dec. 1999.

[3] R. Castagno, T. Ebrahimi, and M. Kunt, "Video segmentation based on multiple features for interactive multimedia applications," *IEEE Trans. Circ. Syst. for Video Tech.*, 8, 5, 562–571, Sept. 1998.

[4] A. M. Tekalp, *Digital Video Processing*, Prentice Hall, New Jersey, 1995.

[5] P. L. Correia and F. Pereira, "Classification of video segmentation application scenarios," *IEEE Trans. Circ. Syst. for Video Tech.*, 14, 5, 735–741, May 2004.

[6] E. Izquierdo and M. Ghanbari, "Key components for an advanced segmentation system," *IEEE Trans. Multimedia*, 4, 1, 97–113, March 2002.

[7] Y. Wang, Z. Liu, and J.-C. Huang, "Multimedia content analysis using both audio and video clues," *IEEE Signal Processing Magazine*, 17, 12–36, Nov. 2000.

[8] G. Ahanger and T. D. C. Little, "A survey of technologies for parsing and indexing digital video," *J. Visual Comm. and Image Repres.*, 7, 28–43, Mar. 1996.

[9] H. Jiang, A. Helal, A. K. Elmagarmid, and A. Joshi, "Scene change detection techniques for video databases," *Multimedia Systems*, 6, 186–195, 1998.

[10] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video-shot change detection methods," *IEEE Trans. Circ. Syst. for Video Tech.*, 10, 1–13, Feb. 2000.

[11] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," *Signal Proc.: Image Communication*, 16, 5, 477–500, Jan. 2001.

[12] R. Lienhart, "Reliable transition detection in videos: A survey and practitioners guide," *Int. J. Image Graph.*, 1, 469–486, Aug. 2001.

[13] A. Hanjalic, "Shot-boundary detection: Unraveled and resolved?" *IEEE Trans. Circ. Syst. for Video Tech.*, 12, 90–105, Feb. 2002.

[14] P. Bouthemy, M. Gelgon, and F. Ganansia, "A unified approach to shot change detection and camera motion characterization," *IEEE Trans. Circ. Syst. Video Tech.*, 9, 1030–1044, Oct. 1999.

[15] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," Multimedia Systems, 1, 10–28, 1993.

[16] E. Stringa and C. S. Regazzoni, "Real-time video shot detection for scene surveillance applications," *IEEE Trans. Image Proc.*, 9, 1, 69–79, Jan. 2000.

[17] A. Ekin, A. M. Tekalp, and R. Mehrotra, "Automatic soccer video analysis and summarization," *IEEE Trans. on Image Proc.*, 12, 7, 796–807, July 2003.

[18] A. Hampapur, R. Jain, and T. E. Weymouth, "Production model based digital video segmentation," *Multimedia Tools Appl.*, 1, 9–46, 1995.

[19] H. Sundaram and S.-F. Chang, "Computable scenes and structures in films," *IEEE Trans. Multimedia*, 4, 482–491, Dec. 2002.

[20] B. L. Yeo and B. Liu, "Rapid scene analysis on compressed videos," *IEEE Trans. Circ. Syst. Video Tech.*, 5, 6, Dec. 1995.

[21] D. Lelescu and D. Schonfeld, "Statistical sequential analysis for real-time video scene change detection on compressed multimedia stream," *IEEE Trans. Multimedia*, 5, 1, 106–117, March 2003.

[22] T. Aach, A. Kaup, and R. Mester, "Statistical model-based change detection in moving video," *Signal Proc.*, 31, 2, 165–180, March 1993.

[23] R. Mech and M. Wollborn, "A noise robust method for 2D shape estimation of moving objects in video sequences considering a moving camera," *Signal Processing* (special issue), 66, 2, 203–217, April 1998.

[24] M. Irani, B. Rousso, and S. Peleg, "Computing occluding and transparent motions," *Int. J. Computer Vision*, 12, 1, 5–16, 1994.

[25] A. Neri, S. Colonnese, G. Russo, and P. Talone, "Automatic moving object and background separation," *Signal Processing* (special issue), 66, 2, 219–232, April 1998.

[26] S.-Y. Chien, S. Y. Ma, and L.-G. Chen, "Efficient moving object segmentation algorithm using background registration

technique," *IEEE Trans. Circ. Syst. for Video Tech.*, 12, 7, 577–586, July 2002.

[27] C. Gu, T. Ebrahimi, and M. Kunt, "Morphological moving object segmentation and tracking for content-based video coding," *Proc. Int. Symp. Multimedia Comm. and Video Coding*, New York, NY, Oct. 1995.

[28] G. Adiv, "Determining three-dimensional motion and structure from optical flow generated by several moving objects," *IEEE Trans. Patt. Anal. Mach. Intell.*, 7, 384–401, 1985.

[29] P. J. Burt, R. Hingorani, and R. Kolczynski, "Mechanisms for isolating component patterns in the sequential analysis of multiple motion," in *IEEE Workshop on Visual Motion*, 187–193, Princeton, New Jersey, Oct. 1991.

[30] J. R. Bergen, P. J. Burt, K. Hanna, R. Hingorani, P. Jeanne, and S. Peleg, "Dynamic multiple-motion computation," in *Artificial Intelligence and Computer Vision* (Y. A. Feldman and A. Bruckstein, Eds.), Holland: Elsevier, 1991, 147–156.

[31] J. R. Bergen, P. J. Burt, R. Hingorani, and S. Peleg, "A three-frame algorithm for estimating two-component image motion," *IEEE Trans. Patt. Anal. Mach. Intell.*, 14, 886–896, Sep. 1992.

[32] H. Sawhney, S. Ayer, and M. Gorkani, "Model-based 2D and 3D dominant motion estimation for mosaicing and video representation," *IEEE Int. Conf. Computer Vision*, Cambridge, MA, June 1995.

[33] W. B. Thompson, "Combining motion and contrast for segmentation," *IEEE Trans. Pattern Anal. Mach. Intel.*, 2, 543–549, 1980.

[34] D. W. Murray and B. F. Buxton, "Scene segmentation from visual motion using global optimization," *IEEE Trans. Patt. Anal. Mach. Intel.*, 9, 2, 220–228, March 1987.

[35] M. Hoetter and R. Thoma, "Image segmentation based on object oriented mapping parameter estimation," *Signal Proc.*, 15, 315–334, 1988.

[36] P. Schroeter and S. Ayer, "Multi-frame based segmentation of moving objects by combining luminance and motion," *Signal Processing VII, Theories and Applications: Proc. of Seventh European Signal Proc. Conf.*, Sept. 1994.

[37] J. Y. A. Wang and E. Adelson, "Representing moving images with layers," *IEEE Trans. on Image Proc.*, 3, 625–638, Sep. 1994.

[38] S. Ayer and H. Sawhney, "Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL coding," *IEEE Int. Conf. Comp. Vision*, Cambridge, MA, June 1995.

[39] J.-M. Odobez and P. Bouthemy, "Direct model-based image motion segmentation for dynamic scene analysis," *Proc. Second Asian Conf. on Computer Vision (ACCV)*, Dec. 1995.

[40] Y. Weiss and E. H. Adelson, "A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models," *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, June 1996.

[41] Y. W. Lim and S. U. Lee, "On the color image segmentation algorithm based on the thresholding and the fuzzy c-means techniques," *Pattern Recognition*, 23, 9, 935–952, 1990.

[42] Y. Altunbasak, E. Eren, and A. M. Tekalp, "Region-based affine motion segmentation using color information," *Graphical Models and Image Processing*, 60, 1, 13–23, Jan. 1998.

[43] N. Kiryati et al., "A probabilistic Hough transform," *Patt. Recog.*, 24, 4, 303–316, 1991.

[44] S.-M. Kruse, "Scene segmentation from dense displacement vector fields using randomized Hough transform," *Signal Proc.: Image Comm.*, 9, 29–41, 1996.

[45] N. Diehl, "Object-oriented motion estimation and segmentation in image sequences," *Signal Processing: Image Comm.*, 3, 23–56, 1991.

[46] S. F. Wu and J. Kittler, "A gradient-based method for general motion estimation and segmentation," *J. Vis. Comm. Image Rep.*, 4, 1, 25–38, March 1993.

[47] S.-L. Iu, "Robust estimation of motion vector fields with discontinuity and occlusion using local outliers rejection," *SPIE*, 2094, 588–599, 1993.

[48] M. M. Chang, A. M. Tekalp, and M. I. Sezan, "Simultaneous motion estimation and segmentation," *IEEE Trans. Image Processing*, 6, 9, 1326–1333, Sep. 1997. (also in Proc. ICASSP'94, Adelaide, Australia.)

[49] P. B. Chou and C. M. Brown, "The theory and practice of Bayesian image labeling," *Int. J. Comp. Vision*, 4, 185–210, 1990.

[50] S. Hsu, P. Anandan, and S. Peleg, "Accurate computation of optical flow by using layered motion representations," *Proc. Int. Conf. Patt. Recog.*, Jerusalem, Israel, 743–746, Oct. 1994.

[51] E. Dubois and J. Konrad, "Estimation of 2-D motion fields from image sequences with application to motion-compensated processing," in *Motion Analysis and Image Sequence Processing* (M. I. Sezan and R. L. Lagendijk, Eds.), Norwell, MA: Kluwer, 1993.

[52] C. Stiller, "Object-oriented video coding employing dense motion fields," *Proc. Int. Conf. ASSP*, Adelaide, Australia, April 1994.

[53] Z. Kato, M. Berthod, and J. Zerubia, "Parallel image classification using multiscale Markov random fields," *Proc. IEEE Int. Conf. ASSP*, Minneapolis, MN, V137–140, April 1993.

[54] J. Shi and C. Tomasi, "Good features to track," *IEEE Conf. Comp. Vision and Patt. Recog.* (CVPR), Seattle, WA, June 1994.

[55] Y. Fu, A. T. Erdem, and A. M. Tekalp, "Tracking visible boundary of objects using occlusion adaptive motion snake," *IEEE Trans. Image Processing*, 9, 12, 2051–2060, Dec. 2000.

[56] H. W. Park, T. Schoepflin, and Y. Kim, "Active contour model with gradient directional information: Directional snake," *IEEE Trans. Circ. Syst. Video Tech.*, 11, 2, Feb. 2001.

[57] P. Salembier, "Morphological multiscale segmentation for image coding," *Signal Proc.*, 38, 339–386, 1994.

[58] A. M. Tekalp, P. J. L. van Beek, C. Toklu, and B. Gunsel, "2D mesh-based visual object representation for interactive synthetic/natural video," *Proc. IEEE* (special issue), 86, 6, 1029–1051, June 1998.

[59] P. L. Correia and F. Pereira, "Objective evaluation of video segmentation quality," *IEEE Trans. Image Proc.*, 12, 2, 186–200, Feb. 2003.

[60] C. Erdem, B. Sankur, and A. M. Tekalp, "Preformance measures for video object segmentation and tracking," *IEEE Trans. on Image Processing*, 13, 7, 937–951, July 2004.