Automatic Segmentation of Moving Objects for Video Object Plane Generation

Thomas Meier and King N. Ngan, *Senior Member, IEEE*

(Invited Paper)

***Abstract*—The new video coding standard MPEG-4 is enabling content-based functionalities. It takes advantage of a prior decomposition of sequences into video object planes (VOP's) so that each VOP represents one moving object. A comprehensive review summarizes some of the most important motion segmentation and VOP generation techniques that have been proposed. Then, a new automatic video sequence segmentation algorithm that extracts moving objects is presented. The core of this algorithm is an object tracker that matches a two-dimensional (2-D) binary model of the object against subsequent frames using the Hausdorff distance. The best match found indicates the translation the object has undergone, and the model is updated every frame to accommodate for rotation and changes in shape. The initial model is derived automatically, and a new model update method based on the concept of moving connected components allows for comparatively large changes in shape. The proposed algorithm is improved by a filtering technique that removes stationary background. Finally, the binary model sequence guides the extraction of the VOP's from the sequence. Experimental results demonstrate the performance of our algorithm.**

***Index Terms*— Content-based functionalities, MPEG-4, object tracking, video object planes, video sequence segmentation.**

I. Introduction

TRADITIONAL video standards such as MPEG-1, MPEG-2, H.261, or H.263 are low-level techniques in the sense that no segmentation or analysis of the scene is required. They can achieve high compression ratios and are suitable for a wide range of applications. However, with the increasing popularity of multimedia applications and content-based interactivity, new video coding schemes are necessary.

The standard MPEG-4 [1], [2], which is currently being developed, enables content-based functionalities by introducing the concept of video object planes (VOP's). Each frame of the input sequence is segmented into arbitrarily shaped image regions (VOP's) such that each VOP describes one semantically meaningful object or video content of interest. A video object layer is assigned to each VOP containing shape, motion, and texture information.

Decomposing a video sequence into VOP's is very difficult, and comparatively little research has been undertaken in this field. An intrinsic problem of VOP generation is that objects of interest are not homogeneous with respect to low-level features such as color, intensity, or optical flow. Thus, conventional segmentation algorithms will fail to obtain meaningful partitions.

This paper addresses video object plane generation and presents a new algorithm that can automatically extract moving objects from a sequence. Since these objects are characterized by a different motion from that of the background, some type of motion information must be incorporated into the segmentation algorithm.

Optical flow or motion fields could theoretically be used, but they are extremely noise sensitive, and their accuracy is limited due to the aperture and occlusion problem. Change detectors or difference images, on the other hand, mark occlusion areas as changed, while the objects themselves are unchanged unless they contain sufficient texture. This makes exact boundary location difficult, and an additional mechanism is necessary to fill the holes inside objects.

Our proposed algorithm is based on pattern recognition and object tracking principles, and thereby avoids many of the problems associated with motion estimation. The concept of moving connected components is introduced to enable automatic detection of moving objects, and a novel model update method allows for relatively large changes in shape. To improve the stability of the segmentation and to reduce the computational complexity, a filter is presented that removes stationary background. The VOP's obtained by our algorithm are more accurate than those of other techniques examined.

The rest of this paper is organized as follows. Section II covers motion estimation and points out some limitations of motion fields for segmentation. A comprehensive review of motion segmentation and VOP generation techniques is given in Section III. Our new algorithm is then described in Section IV, and results are shown in Section V. Finally, Section VI concludes this paper and outlines some extensions of the proposed algorithm for future research.

II. Motion

A. Motion as Cue for Segmentation

Moving objects are often characterized by a coherent motion that is distinct from that of the background. This makes motion a very useful feature for segmenting video sequences. It can complement other features such as color, intensity, or edges that are commonly used for segmentation of still images,

Manuscript received October 31, 1997; revised April 8, 1998. This paper was recommended by Associate Editor S. Panchanathan.

The authors are with the Visual Communications Research Group, Department of Electrical and Electronic Engineering, University of Western Australia, Nedlands, Perth, WA 6907 Australia.

Publisher Item Identifier S 1051-8215(98)06323-X.

and some motion segmentation algorithms are even based on motion only.

Let us start by defining the rather vague term motion. We denote by $I(x, y; k)$ the intensity or luminance of pixel (x, y) in frame k. Following the definitions in [3], we have to distinguish between two-dimensional (2-D) motion and apparent motion. The projection of the three-dimensional (3-D) motion onto the image plane is referred to as 2-D motion. It is the true motion that we would like to know. On the other hand, apparent motion is what we perceive as motion and is induced by temporal changes in the image intensity $I(x, y; k)$. Apparent motion can be characterized by a correspondence vector field or by an optical flow field. A correspondence vector describes the displacement of a pixel between two frames, whereas the optical flow (u, v) at pixel $(x, y; k)$ refers to a velocity and is defined as

$$(u, v) = \left(\frac{dx}{dt}, \frac{dy}{dt}\right). \tag{1}$$

It is easy to see that optical flow and correspondence vectors are related.

Note that apparent motion and 2-D motion are not equivalent. Consider a static scene with varying illumination. The 2-D motion is obviously zero because no 3-D motion is present; however, the change in illumination induces optical flow, and therefore apparent motion. From (1), it can also be seen that apparent motion is highly sensitive to noise, which can cause largely incorrect results. Further, moving objects or regions must contain sufficient texture to generate optical flow, because the luminance in the interior of moving regions with uniform intensity remains constant. Unfortunately, we can only observe apparent motion, making motion estimation a very challenging task.

B. Motion Estimation

Besides the difficulties already mentioned, motion estimation algorithms have to solve the so-called occlusion and aperture problem. The occlusion problem refers to the fact that no correspondence vectors exist for covered and uncovered background. To illustrate the aperture problem, we first introduce the optical flow constraint (OFC). It is generally assumed that the intensity remains constant along the motion trajectory [3], i.e.,

$$\begin{aligned}\frac{d}{dt}I(x, y; t) &= \frac{\partial I}{\partial x}\cdot\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\cdot\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t}\\ &= \langle \nabla I, (u, v)\rangle + \frac{\partial I}{\partial t} = 0\end{aligned} \tag{2}$$

where $\langle\cdot,\cdot\rangle$ denotes the vector inner product. The aperture problem states that the number of unknowns is larger than the number of observations. From the optical flow constraint (2) follows that only the flow component in the direction of the gradient ∇I, the so-called normal flow, can be estimated. The orthogonal component can take on any value without changing the inner product, and is therefore not defined. Thus, additional assumptions are necessary to obtain a unique solution. These usually impose some smoothness constraints on the optical flow field to achieve continuity.

Two ways of describing motion fields are possible. In the nonparametric representation, a dense field is estimated where each pixel is assigned a correspondence or flow vector. Block matching and variants thereof are among the most popular nonparametric approaches due to their simplicity. The current frame is subdivided into blocks of equal size, and for each block the best match in the next (or previous) frame is computed. All pixels of a block are assumed to undergo the same translation, and are assigned the same correspondence vector. The selection of the block size is crucial. Large windows might contain more than one motion and cannot accurately locate motion boundaries, whereas small windows often result in wrong matches within uniform regions in the presence of noise. A weakness of block-matching algorithms is their inability to cope with rotations and deformations. Nevertheless, their simplicity and relative robustness make it a popular technique.

Nonparametric dense field representations are generally not suitable for segmentation because an object moving in the 3-D space generates a spatially varying 2-D motion field even within the same region, except for the simple case of pure translation. That is the reason why parametric models are commonly used in segmentation algorithms. However, dense field estimation is often the first step in calculating the model parameters.

Parametric models require a segmentation of the scene, which is our ultimate goal, and describe the motion of each region by a set of a few parameters. The motion vectors can then be synthesized from these model parameters. A parametric representation is more compact than a dense field description and less sensitive to noise, because many pixels are treated jointly to estimate a few parameters.

In order to derive a model or transformation that describes the motion of pixels between successive frames, assumptions on the scene and objects have to be made. Let (X, Y, Z) and (X', Y', Z') denote the 3-D coordinates of an object point in frame k and $k+1$, respectively. The corresponding image plane coordinates are (x, y) and (x', y'). If a 3-D object undergoes translation, rotation, and linear deformation, the 3-D displacement of a point on the object is given by [4]

$$\begin{bmatrix}X'\\Y'\\Z'\end{bmatrix} = \begin{bmatrix}s_{11} & s_{12} & s_{13}\\ s_{21} & s_{22} & s_{23}\\ s_{31} & s_{32} & s_{33}\end{bmatrix}\cdot\begin{bmatrix}X\\Y\\Z\end{bmatrix} + \begin{bmatrix}t_1\\t_2\\t_3\end{bmatrix}. \tag{3}$$

It is very common to model 3-D objects by (piecewise) planar patches whose points satisfy

$$aX + bY + cZ = 1. \tag{4}$$

If such a planar object is moving according to (3), the affine motion model is obtained under orthographic projection and the eight-parameter model under perspective projection.

As can be seen from Fig. 1, the 3-D coordinates are related to the image plane coordinates under the orthographic (parallel) projection by

$$(x, y) = (X, Y) \quad \text{and} \quad (x', y') = (X', Y'). \tag{5}$$

This projection is computationally efficient and a good approximation if the distance between the objects and the camera is

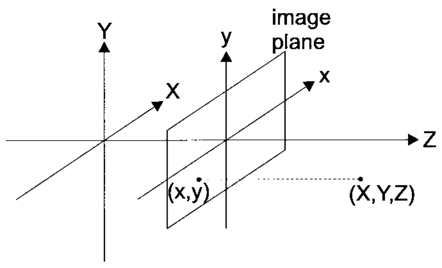


Fig. 1. Projection of pixel (X, Y, Z) onto image plane (x, y) under orthographic (parallel) projection.

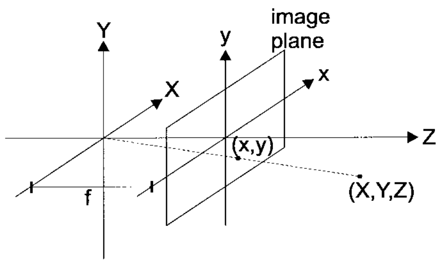


Fig. 2. Projection of pixel (X, Y, Z) onto image plane (x, y) under perspective (central) projection.

large compared to the depth of the objects. From (3)–(5), it follows that

$$\begin{aligned} x' &= a_1 x + a_2 y + a_3 \\ y' &= a_4 x + a_5 y + a_6 \end{aligned} \tag{6}$$

which is known as the affine model. In the case of the more realistic perspective (central) projection, we can see from Fig. 2 that

$$(x, y) = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right) \quad \text{and} \quad (x', y') = \left(f\frac{X'}{Z'}, f\frac{Y'}{Z'}\right). \tag{7}$$

Together with (3) and (4), this results in the eight-parameter model

$$\begin{aligned} x' &= \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1} \\ y' &= \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1}. \end{aligned} \tag{8}$$

Both the affine and eight-parameter model are very popular, but many other transformations exist depending on the assumptions made.

Parametric models describe each region by one set of parameters that is either estimated by fitting a model in the least squares sense to a dense motion field obtained by a nonparametric method or directly from the luminance signal $I(x, y; k)$ as in [5] and [6]. Although parametric representations are less noise sensitive, they still suffer from the intrinsic problems of motion estimation. It should be noticed that one has to be careful when interpreting an estimated flow field. Most likely, it is necessary to include additional information such as color or intensity to accurately and reliably detect boundaries of moving objects.

III. Motion Segmentation

A classical approach to motion segmentation is to estimate a dense motion field followed by a segmentation of the scene based only on this motion information [7]–[10]. In his early work, Adiv [7] proposed a hierarchically structured two-stage algorithm. The flow field is first segmented into connected components using the Hough transform such that the motion of each component can be modeled by an affine transformation (6). Adjacent components are then merged into segments if they obey the same eight-parameter quadratic motion model. In the second stage, neighboring segments that are consistent with the same 3-D motion (3) are combined, resulting in the final segmentation.

The Bayesian framework provides an elegant formalism and is among the most popular approaches to motion segmentation [8]–[13]. The key idea is to find the maximum *a posteriori* (MAP) estimate of the segmentation X for some given observation O, i.e., to maximize $P(X|O) \propto P(O|X)P(X)$.

Murray and Buxton [8] used an estimated flow field as observation O. As it is common, the segmentation or prior model X is assumed to be a sample of a Markov random field (MRF) to enforce continuity of the segmentation labels, and thus, $P(X)$ is a Gibbs distribution [14]. The energy function of the MRF consists of a spatial smoothness term, a temporal continuity term, and a line field as in [15] to allow for motion discontinuities. To define the observation model $P(O|X)$, the parameters of a quadratic flow model [7] are calculated for each region by linear regression. The mismatch between this synthesized flow and the flow field given in O is assumed to be zero-mean white Gaussian noise. The resulting probability function $P(O|X)P(X)$ is maximized by simulated annealing [15]. Major drawbacks of this proposal are the computational complexity and that the number of objects likely to be found has to be specified.

A similar approach was taken by Bouthemy and François [9]. The energy function of their MRF consists only of a spatial smoothness term. The observation O contains the temporal and spatial gradients of the intensity function, which is essentially the same information as the optical flow due to the OFC (2). For each region, the affine motion parameters (6) are computed in the least-squares sense, and $P(O|X)$ models the deviation of this synthesized flow from the optical flow constraint (2) by zero-mean white Gaussian noise. The optimization is performed by iterated conditional modes (ICM) [16], which is faster than simulated annealing, but likely to get trapped in a local minimum. To achieve temporal continuity, the segmentation result of the previous frame is used as an initial estimate for the current frame. The algorithm then alternates between updating the segmentation labels X, estimating the affine motion parameters, and updating the number of regions in the scene.

The techniques [7]–[9] include only optical flow data into the segmentation decision, and hence, their performance is limited by the accuracy of the estimated flow field. This means that they inevitably suffer from the problems described in Section II such as noise sensitivity and inaccuracy at motion and therefore object boundaries. In contrast, Chang *et al.* [10]

incorporated intensity information into the observation O. The energy function of the MRF includes a spatial continuity term and a motion-compensated temporal term to enforce temporal continuity. Two methods were proposed to generate a synthesized flow field for each region: the eight-parameter quadratic model [7] or the mean flow vector of the region calculated from the given field in O. For the conditional probability $P(O|X)$, it is assumed that both the deviation of the observed flow from the synthesized flow and the difference between the gray level of a pixel and the mean gray level of the region it belongs to obey zero-mean Gaussian distributions. By controlling the variances of these two distributions, more weight is put on the flow data in the case where it is reliable, i.e., for small values of the displaced frame difference (DFD), and more weight on the intensity in areas with unreliable flow data. The optimization is then performed by ICM as in [9].

It is possible to treat motion estimation and segmentation jointly in the Bayesian framework [11]–[13]. In this case, the observation O consists only of the gray-level intensity, and both the segmentation and the motion field have to be estimated. Chang *et al.* [11] used both a parametric and a dense correspondence field representation of the motion, with the parameters of the eight-parameter model (8) being obtained in the least squares sense from the dense field. The objective function resulting from the MAP criterion consists of three terms, each derived from an MRF. The first term is maximized when both the synthesized and dense motion field minimize the DFD, and the second term is maximized if the dense field is smooth and the parametric representation is consistent with the dense field. However, smoothness is only enforced for pixels having the same segmentation label, i.e., it is not enforced across region boundaries. The last term is a standard spatial continuity term to describe the prior expectation on the segmentation. Since the number of unknowns is three times higher when the motion field has to be estimated as well, the computational complexity is significantly larger. Chang *et al.* decomposed the objective function into two terms so that the motion estimates and the segmentation labels can be maximized alternating using highest confidence first (HCF) [17] and ICM.

The technique proposed by Stiller in [12] and extended in [13] is similar, but no parametric motion field representation is necessary. In [12], the objective function consists of two terms. The DFD generated by the dense motion field is modeled by a zero-mean generalized Gaussian distribution, and an MRF ensures segmentwise smoothness of the motion field and spatial continuity of the segmentation. In [13], the DFD is also assumed to obey a zero-mean generalized Gaussian distribution; however, occluded regions are detected, and no correspondence is required for them. The MRF modeling the motion field and segmentation is made up of four terms enforcing spatial and temporal continuity of the segmentation, segmentwise spatial smoothness of the motion field, and temporal continuity of motion vectors along the motion trajectories. Although ICM is used to obtain the MAP estimate, the computational burden of this algorithm is enormous.

Techniques that make use of Bayesian inference and model images by Markov random fields are more plausible than some rather ad hoc methods. They can also easily incorporate mechanisms to achieve spatial and temporal continuity. On the other hand, these approaches suffer from high computational complexity, and many algorithms need the number of objects or regions in the scene as an input parameter.

Hierarchically structured segmentation algorithms were proposed by Hötter and Thoma [5], Musmann *et al.* [6], and Diehl [18]. A change detector divides the current frame into changed and unchanged regions, and each connected changed region is assumed to correspond to one object. Starting from the largest changed region, the motion parameters for this object are estimated directly from the spatiotemporal image intensity and gradient. If the prediction error after motion compensation is too large, this object is further subdivided and analyzed in subsequent levels of hierarchy. The algorithm sequentially refines the segmentation and motion estimation until all changed regions are accurately compensated. Because these techniques alternate between analyzing the image and synthesizing, they have been described as object-oriented analysis–synthesis algorithms. In [5] and [6], the eight-parameter motion model (8) is used, and the parameters are obtained by a direct method. The luminance function is approximated by a Taylor series expansion so that the frame difference can be expressed in terms of spatial intensity gradients and the unknown parameters. Both frame difference and gradients are easy to compute, and the model parameters are obtained by linear regression. A 12-parameter quadratic motion model that describes a parabolic surface undergoing the 3-D motion (3) under parallel projection is proposed in [18]. An iterative technique that is similar to the Newton–Raphson algorithm estimates the parameters by minimizing the MSE between the motion-compensated and the current frame. Edge information is incorporated into the segmentation algorithm to improve the accuracy of boundaries.

Morphological tools such as the watershed algorithm and simplification filters are becoming increasingly popular for segmentation and coding [19]–[23]. An introduction, discussion of potential problems, and several applications to segmentation are presented by Meyer and Beucher [19]. Salembier and Pardàs [20] described a segmentation algorithm that has a typical structure for morphological approaches. In a first step, the image is simplified by the morphological filter "open–close by reconstruction" to remove small dark and bright patches. The size of these patches depends on the structuring element used. The color or intensity of the resulting simplified images is relatively homogeneous. An attractive property of these filters is that they do not blur or change contours like low-pass or median filters. The following marker extraction step detects the presence of homogeneous areas, for example, by identifying large regions of constant color or luminance. This step often contains most of the knowhow of the algorithm. Each extracted marker is then the seed for a region in the final segmentation. Undecided pixels are assigned a label in the decision step, the so-called watershed algorithm, which is a technique similar to region growing. The watershed algorithm is well defined and can be efficiently implemented by hierarchical FIFO queues. A quality estimation is performed in [20] as a last step to determine which regions require resegmentation. The proposed segmentation by Salembier *et al.* in [21] is very similar, but an additional projection step is incorporated that warps the previous partition onto the current frame. This projection, which is also computed by the watershed algorithm, ensures temporal continuity and linking of the segmentation.

The segmentation algorithms in [19]–[21] are not true video segmentation techniques. They consider video sequences to be 3-D signals and extend conventional 2-D methods, although the time axis does not play the same role as the two spatial axes. A morphological video segmentation algorithm was proposed by Choi *et al.* [22]. Their marker extraction step detects areas that are not only homogeneous in luminance, but also in motion, so-called joint markers. For that, intensity markers are extracted as in [20], and affine motion parameters (6) are calculated for each marker by linear regression from a dense flow field. Intensity markers for which the affine model is not accurate enough are split into smaller markers that are homogeneous. As a result, multiple joint markers might be obtained from a single intensity marker. The watershed algorithm also uses a joint similarity measure that incorporates luminance and motion. In a last stage, the segmentation is simplified by merging regions with similar affine motions. A drawback of this technique is the lack of temporal correspondence to enforce continuity in time.

Morphological segmentation techniques are computationally efficient, and there is no need to specify the number of objects as with some Bayesian approaches, because this is determined automatically by the marker or feature extraction step. However, due to its nature, the watershed algorithm suffers from the problems associated with region-growing techniques.

The algorithms described so far are mainly focused on coding. They segment video sequences into regions that are homogeneous with respect to motion and possibly color or luminance. For content-based functionalities as in MPEG-4, we would like to partition the frames into objects that are semantically meaningful to the human observer. Thus, the above techniques will fail in many practical situations where objects do not correspond to partitions based on simple features like motion or color. Segmentation algorithms that specifically address video object plane (VOP) generation have been proposed, many of them just recently with the development of the new video coding standard MPEG-4 [24]–[26], [23], [27].

Wang and Adelson [24] proposed a layered representation of image sequences that corresponds to the VOP technique used by MPEG-4. The current frame is segmented based on motion with each object or layer being modeled by an affine transformation (6). The algorithm starts by estimating the optical flow field, and then subdivides the frame into square blocks. The affine motion parameters are computed for each block by linear regression to get an initial set of motion hypotheses. The pixels are then grouped by an iterative adaptive K-means clustering algorithm. Pixel (x, y) is assigned to hypothesis or layer i if the difference between the optical flow at (x, y) and the flow vector synthesized from the affine parameters of layer i is smaller than for any other hypothesis. To construct the layers, the information of a longer sequence is necessary. The frames are warped according to the affine motion of the layers such that coherently moving objects are aligned. A temporal median filter is then applied to obtain a single representative image for each object. This proposal has several disadvantages. If in a sequence different views of the same object are shown, it is not possible to represent that object by a single image that is warped from frame to frame. Further, the affine transformation (6) might not be able to describe the motion of a complete layer in the presence of strongly nonrigid motion such as a person walking. The algorithm also depends completely on the accuracy of the optical flow estimates since no color or intensity information is used. Finally, the layer construction process makes real-time execution impossible, because a longer sequence of frames is required.

A double-partition approach based on morphology was suggested by Marqués and Molina [23]. Initially, objects of interest have to be selected interactively, leading to a partition at object level that corresponds to a decomposition into video object planes. These objects are normally not homogeneous in color or motion and are resegmented to obtain a fine partition that is spatially homogeneous. After estimating a dense motion field by block matching, the fine partition is projected onto the next frame using motion compensation. These projected regions are used to extract the markers for the next frame, which is then segmented by the watershed algorithm based on luminance. To improve the temporal stability, the segmentation process is guided by a change detection masks that prevents markers of static areas to overgrow moving areas and vice versa. Finally, the new object level partition is computed from the projected and segmented fine partition, whereby the algorithm must keep track of the labels of each region to know the correspondence between fine regions and objects.

Automatic segmentation is formulated by Neri *et al.* [25] as the problem of separating moving objects from a static background. In a preliminary stage, potential foreground regions are detected by applying a higher order statistics (HOS) test to a group of interframe differences. The nonzero values in the difference frames are either due to noise or moving objects, with the noise being assumed to be Gaussian in contrast to the moving objects, which are highly structured. In the case of moving background, the frames must first be aligned by motion compensation. For all difference frames, the zero-lag fourth-order moments are calculated because of their capability to suppress Gaussian noise. These moments are then thresholded, resulting in a preliminary segmentation map containing moving objects and uncovered background. To identify uncovered background, the motion analysis stage calculates the displacement of pixels that are marked as changed. The displacement is estimated at different lags from the fourth-order moment maps by block matching. If the displacement of a pixel is zero for all lags, it is classified as background and as foreground otherwise. Finally, the regularization phase applies morphological opening and closing operators to achieve spatial continuity and to remove small holes inside moving objects of the segmentation map. The resulting segmented foreground objects are slightly too large, because the boundary location is not directly determined from the gray level or edge image. A version of this technique is currently under investigation in the ISO MPEG-4 N2 Core Experiment on Automatic Segmentation Techniques [28]. It has a postprocessor incorporated to improve the boundary location by adjusting the boundaries to spatial edges.

Mech and Wollborn [26] generate the video object plane or object mask from an estimated change detection mask (CDM). Initially, a change detection mask is generated by taking the difference between two successive frames using a global threshold. This CDM is then refined in an iterative relaxation that uses a locally adaptive threshold to enforce spatial continuity. Temporal stability is increased by incorporating a memory such that each pixel is labeled as changed if it belonged to an object at least once in the last L change detection masks. The simplification step includes a morphological close and removes small regions to obtain the final CDM. The object mask is calculated from the CDM by eliminating uncovered background and adapting to gray-level edges to improve the location of boundaries. A version of this algorithm is also part of the ISO MPEG-4 N2 Core Experiment [29]. It contains an additional scene change or cut detector, a global motion estimation and compensation step based on the eight-parameter model (8), and the memory length L has been made adaptive.

While the two proposals [28], [29] to the ISO MPEG-4 N2 Core Experiment perform segmentation mainly based on temporal information, Choi *et al.* [30] presented a spatial morphological segmentation technique. It starts with a global motion estimation and compensation step. The global affine motion parameters (6) are calculated from the correspondence field, which is obtained by a block-matching algorithm. After that, the presence of a scene cut is examined. Then, the actual segmentation commences by simplifying the frame with a morphological open–close by reconstruction filter. The thresholded morphological gradient image, calculated from the luminance and chrominance components of the frame, serves as input for the watershed algorithm which detects the location of the object boundaries. To avoid oversegmentation, regions smaller than a threshold are merged with their neighbors. Finally, a foreground/background decision is made to create the video object planes. Every region for which more than half of its pixels are marked as changed in a change detection mask is assigned to the foreground. To enforce temporal continuity, the segmentation is aligned with that of the previous frame, and those regions for which a majority of pixels belonged to the foreground before are added to the foreground too. This allows tracking an object even when it stops moving for an arbitrary time. In contrast, the techniques [25] and [26] will lose track after a certain number of frames, depending on the size of the group of frames and memory length, respectively.

A combination of the two temporal segmentation techniques [28], [29] with the spatial segmentation method [30] to form one algorithm is currently under investigation [31], [32].

In the next section, we will describe a new video object plane segmentation algorithm based on Hausdorff object tracking. It is an extension of the technique by Meier and Ngan submitted to the ISO MPEG-4 N2 Core Experiment [27].

IV. Video Object Plane Segmentation Algorithm

In this section, a new segmentation algorithm is presented that automatically extracts moving objects from a video sequence. It extends the techniques proposed in [27], [33] to scenes with a moving camera or background. The resulting video object planes (VOP's) can be used as input for content-based coding schemes such as MPEG-4.

As explained in Sections II and III, motion estimation is a very difficult task, and motion fields are often not reliable enough. In fact, many of these estimation techniques were developed for coding purposes and not for segmentation. If a correspondence vector points to a completely wrong pixel in the sense of motion, it hardly affects the coding result as long as the pixel the vector points to has similar color. In contrast, errorneous motion vectors have a visible, negative effect on segmentation results.

Some sort of motion information is, of course, necessary because our ultimate goal is to segment objects that are moving relative to the background. Several methods employ a change detection mask instead of a motion field. Although easy to compute, this approach has two drawbacks. First, unless moving objects contain sufficient texture, only occlusion areas will be marked as changed, while the interior of objects will be unchanged [see Fig. 5(a)]. Second, objects or parts of objects that stop moving for a certain period of time will be lost, which is not acceptable in content-based applications. To prevent this, a memory would have to be incorporated. Unfortunately, this would cause background that is becoming uncovered to remain classified as an object for the length of the memory, and the resulting VOP's would be larger than the actual objects, depending on the speed of movement and length of memory.

Nevertheless, change detection masks are sometimes more useful than motion fields. For example, in head-and-shoulder sequences, there is only little movement of the person, and the occlusion regions are very small. Thus, a relatively long memory can be attached without getting VOP's that are significantly larger than the object. Estimating a motion field would be more difficult because the motion is simply too small.

In our algorithm, we focus on applications comprising outdoor scenes or objects with strongly nonrigid motion where change detection masks have been shown to be ineffective [25], [28], [29].

The core of our proposed technique is an object tracker that naturally establishes the temporal correspondence of objects throughout the video sequence. This is important for content-based functionalities and allows us to keep track of objects even when they stop moving for an arbitrarily long time.

After a binary model for the object of interest has been derived from the edge image, the tracker matches the model against subsequent frames in the sequence and updates the model every frame to accommodate for rotation and changes in shape of the object. The output of the tracker is a sequence of binary models that will guide the extraction of the VOP's.

Finding the best match for the binary model is very reliable even when the background or camera is moving. The main difficulty is to obtain an initial model and to update the model of a nonrigid object with considerable changes in shape in the presence of cluttered background.

The location of object boundaries is determined based on the binary model, which in turn is derived from the edge image. Hence, the problem of occlusion areas associated with motion fields is avoided. In fact, our algorithm only requires motion estimates in the case of a moving camera or background to align the frames. This global motion estimation is normally very robust if appropriate techniques are applied [see Section IV-B2].

A. Hausdorff Object Tracker

Gray scale images are normally not suitable for template or object matching because they are too sensitive to changes in illumination. Instead, it is common to use binary edge images, which also involve fewer computations. The edge points of the models are not restricted to object boundaries, but can also be in the interior such as eyes and mouth in a face. In our algorithm, the edge images are obtained by the Canny operator [34].

A robust matching method must be able to detect objects that are undergoing translation, rotation, and changes in shape. This excludes basic template matching where the new position is determined by the highest correlation between the model and subsequent frames. The generalized Hough transform, which is used in [35], has been successfully applied to the detection of arbitrarily shaped 2-D binary objects. However, it comes at a high computational cost, especially for a multidimensional Hough accumulator space that includes translation, rotation, and scaling.

The Hausdorff distance was proposed by Huttenlocher *et al.* [36], and an object tracker was described in [37] where the model of the video object of interest was matched against subsequent frames by minimizing the Hausdorff distance. This approach is computationally efficient and robust to noise and changes in shape.

1) The Hausdorff Distance: The Hausdorff distance was proposed in [36] as a measure to compare binary images or portions thereof. The edge pixels that form the model of the object to track are considered as a set of feature points. The same applies to the edge image in which we have to search for the object. Let these sets of feature points be denoted by $O = \{o_1, \cdots, o_m\}$ for the object and $I = \{i_1, \cdots, i_n\}$ for the image where m and n are the number of object and image points, respectively. Then, the Hausdorff distance is defined as

$$H(O, I) = \max\{h(O, I), h(I, O)\} \tag{9}$$

with

$$h(O, I) = \max_{o \in O} \min_{i \in I} \|o - i\| \tag{10}$$

and

$$h(I, O) = \max_{i \in I} \min_{o \in O} \|i - o\|. \tag{11}$$

Thus, for every model point o, the distance to the nearest image point i has to be calculated, and the maximum value is assigned to $h(O, I)$. Then, for each image point i, the distance to the nearest model point o is computed, and $h(I, O)$ is set to the maximum distance. The Hausdorff distance is the larger of the two maxima (see Fig. 3). It is easy to see that for $h(O, I) = d$, every model point must be within distance d of some point in I.

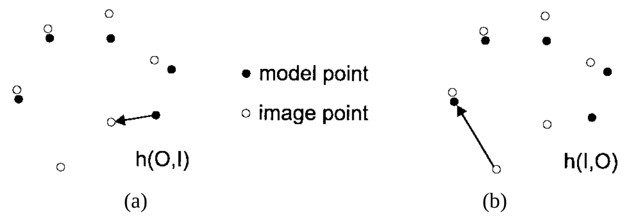


Fig. 3. Calculation of Hausdorff distance. (a) $h(O, I)$ measures the maximum distance of an object point to the nearest image point and (b) $h(I, O)$ the maximum distance of an image point to the nearest object point. In this example, $h(O, I)$ is smaller than $h(I, O)$, and therefore the Hausdorff distance $H(O, I)$ is equal to $h(I, O)$.

The definitions in (10) and (11) can cause some problems, as can be seen in Fig. 3. If one model or image point is outlying, the resulting Hausdorff distance will be very large, even if all other points perfectly match. Therefore, it is preferable to use the generalized Hausdorff distance [36]. Instead of using the maximum value in (10) and (11), the distances are sorted in ascending order and the kth value is chosen, i.e.,

$$h_k(O, I) = k\text{th}_{o \in O} \min_{i \in I} \|o - i\|. \tag{12}$$

Equation (12) is equivalent to (10) for $k = m$. For $k < m, m - k$ points may be outlying without increasing the Hausdorff distance. This is a very useful property when dealing with objects that are partially occluded or rapidly changing their shape. Similarly, $h_l(I, O)$ is defined as the lth value of the ordered distances. With the parameters k and l, we can essentially choose how many model points have to be near image points and vice versa.

There are no point correspondences between model and image points required, because the Hausdorff distance automatically selects the k (or l) best matching points. This is helpful when objects are changing their shape. The best match is found by minimizing the Hausdorff distance between the image and the model for all translations of the model relative to the image. Fig. 4 shows the best match according to the Hausdorff distance with the model of the previous frame shifted to the position of the best match and superimposed on the current frame. Despite the change in shape, the match is very accurate.

2) Implementation of Hausdorff Distance: Several suggestions for efficient implementation are presented in [36]. The main idea is to assume that the Hausdorff distance is smaller than a threshold T so that bad matches can be ruled out early. Obviously, matches can only be found if the Hausdorff distance is indeed smaller than T. In our algorithm, early scan termination has been implemented.

The Hausdorff distance can be computed using the distance transformation (see part A of the Appendix). First, the distance transform is calculated for the edge image so that for each pixel the distance to the nearest edge pixel is known. Then, for all translations $t = (t_x, t_y)$, we calculate $h_{k,t}(O, I)$, where the

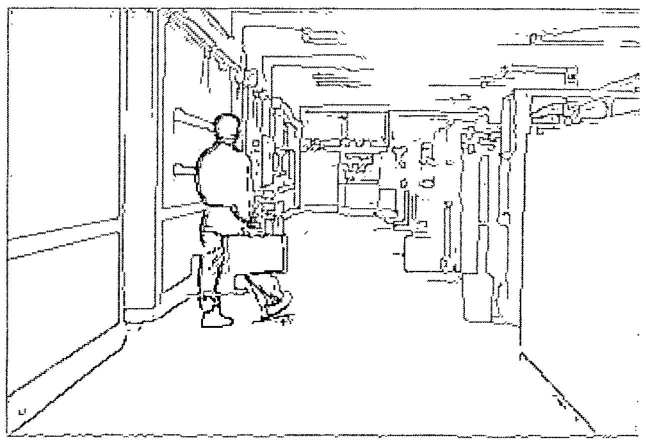


Fig. 4. Model of the person for frame 40 (black) of the sequence *hall monitor* is shifted to the position of best match with respect to the Hausdorff distance in frame 41. The match is accurate except for the right leg, which is moving differently from the rest of the person.

index t indicates that $h_k(O,I)$ depends on the translation t. For that, the object O is translated by the vector t, and the distance transform at the location of model points o directly gives the distance between o and the nearest edge pixel. These distances are then sorted in ascending order, and the kth value is selected to get $h_{k,t}(O,I)$. Because of early scan termination, $h_{k,t}(O,I)$ can only be found for translations where $h_{k,t}(O,I) \le T$. For these translations, $h_{l,t}(I,O)$ is calculated in a similar way to finally obtain $H_t(O,I)$. The smallest Hausdorff distance $H_t(O,I)$ indicates the new position, i.e., the translation t that the model has undergone.

To further accelerate the matching process, the search area has been restricted to translations of up to a specified number of pixels in all directions relative to the position of the object in the previous frame.

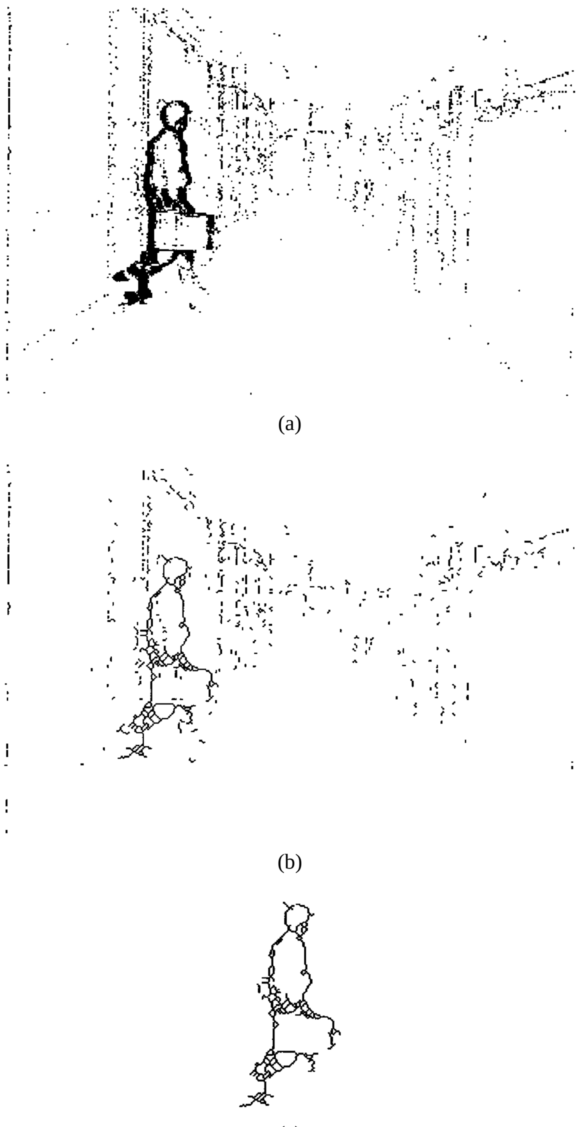
B. Initialization of Moving Objects

Initially, the position of objects is unknown and has to be determined based on motion since we would like to segment moving objects. For scenes with nonstationary background or moving camera, the global motion must first be compensated. Therefore, this will be treated as a separate case.

1) Stationary Background: Let us first assume that the background is stationary and that there is only one moving object in the scene. The extension to multiple objects is straightforward as long as they do not overlap.

Taking the color or intensity difference between two frames is one of the most efficient ways to detect changed areas. High values indicate objects that are moving or changing their shape, as can be seen in Fig. 5(a) where the difference image was thresholded. Unless the objects are highly textured, only the boundaries of moving objects can be observed, and not the objects themselves. However, this is exactly what we need to derive a model for the tracker.

The threshold that is required for the difference image [Fig. 5(a)] depends on the characteristics of the sequence like speed of motion, changes in overall illumination, and noise. Currently, this threshold must be specified as an input parameter to our algorithm, and further study is needed to



(a)

(b)

(c)

Fig. 5. Gray-level difference image after (a) thresholding, (b) thinning, and (c) moving connected component labeling for frame 31 of sequence *hall monitor*.

make the threshold estimation fully automatic. The estimation might be performed similarly to the method in [25], which involves statistical tests and measurement of background activity. The segmentation result is expected to be relatively robust to variations of the threshold for two reasons. First, if the threshold is chosen too low, the following thinning step ensures that all components in the difference image are only one pixel wide. Second, the model is updated every frame and can pick up components of the object that were initially missing in the case where the threshold was too high.

Occlusion areas are normally more than one pixel wide and are thinned by eroding the difference image. It is important that connected components are not split during this thinning process. The algorithm described in part B of the Appendix does not alter connectedness and also preserves geometry well by alternately eroding from north, south, west, and east. In the same step, isolated noisy pixels are eliminated because they are unlikely to indicate an object.

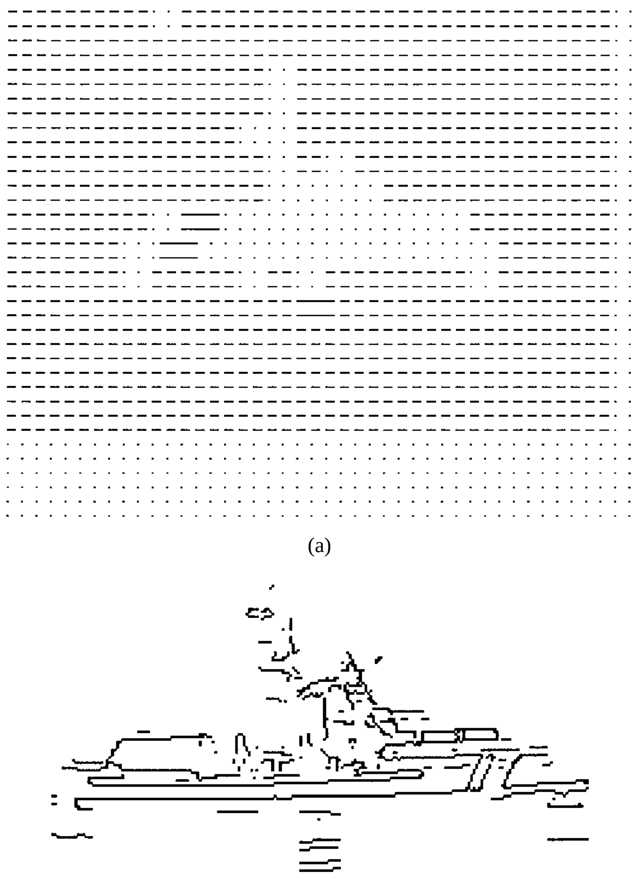


Fig. 6. Model initialization. (a) Displacement field obtained by hierarchical block matching [39]. (b) Model obtained for frame 121 of sequence *coastguard*.

The result after thinning in Fig. 5(b) demonstrates that the pixels belonging to the object are connected, whereas noisy pixels form isolated clusters. A simple algorithm to find connected components in binary images is connected component labeling [38]. Components larger than a specified threshold are then assumed to belong to a moving object, and we refer to them as moving connected components (MCC). It should be possible to determine this threshold automatically, because noisy components are significantly smaller than those belonging to objects. Fig. 5(c) shows an MCC for the sequence *hall monitor*.

Thus, moving objects are detected by finding moving connected components. It remains the task of deriving an initial model for the object tracker by choosing all pixels in the binary edge image that are within a small distance (one–two pixels) of the MCC. This can easily be implemented using the distance transform (see part A of the Appendix).

2) Moving Background: In the case of nonstationary background, it is necessary to compensate the global motion. For that, the correspondence vector field is calculated by hierarchical block matching [39]. Fig. 6(a) shows the motion field obtained for the sequence *coastguard*. The vectors at the bottom and on the right are all zero due to padding. In Sections II and III, difficulties with motion estimation and segmentation were outlined. Global motion estimation is normally an easier task for two reasons. First, global motion is relatively simple and consists only of translation, panning, and possibly zooming. Second, in many applications the background area is large compared to the independently moving objects. Therefore, the effect of occluded regions or errors in the correspondence field is minimal, and robust parameter estimation techniques are likely to yield good results. We suggest using the affine model (6) and estimating the parameters by least median of squares [40] instead of linear regression.

A straightforward approach to extract an initial model would be to calculate the difference between the aligned frames, but due to inaccuracies of the motion model and the estimated correspondence field, the difference image is too noisy. Instead, we propose a different approach.

The block motion estimation algorithm is used to partition the frame into square blocks. Blocks that are moving differently from the global motion can be identified by comparing the estimated correspondence vectors with those synthesized from the affine global motion model. Connected blocks of coherent motion that is distinct from the global motion indicate moving objects, and the initial model consists of the edge pixels inside these blocks [see Fig. 6(b)].

C. Model Update

As a tracked object moves through a video sequence, it might rotate or change its shape. To allow for this, the model must be updated every frame. This stage can be difficult in the presence of cluttered background or moving camera.

In [37], it was assumed that the model changes only slowly between subsequent frames. However, there are often situations where parts of an object change or move more rapidly than the rest of the object. For a walking person, for example, legs and arms move faster than the body. Therefore, we would like to relax this assumption and propose a new update technique that consists of two components: one for slowly changing parts, and the other for parts that change or move rapidly compared to the overall motion of the object. The combination of these two components yields a robust updating mechanism.

The first component updates quasi-rigid parts. The model of the previous frame is shifted to the new position of best match, and it is assumed that pixels close to this shifted old model are part of the object. Thus, all edge pixels within a specified distance T_S of the shifted old model, typically about one–three pixels, are assigned to the new model. This is accomplished by calculating the distance transform of the old model and finding all points in the edge image that have a value for the distance transform smaller or equal to T_S. Fig. 7(a) shows that the slowly changing component can update the object very well, except for the left leg and right arm, which are moving differently from the overall motion.

The larger T_S is chosen, the more likely the whole object will be included into the new model. However, it also increases the possibility of background becoming part of the model. To avoid picking up cluttered background, it is often preferable to filter stationary background beforehand (see Section IV-D).

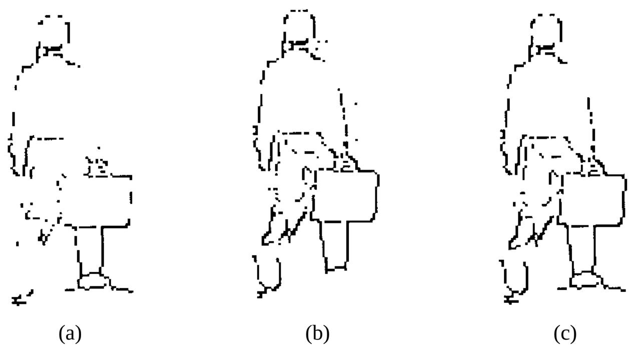


Fig. 7. Model update. (a) Slowly changing component. (b) Fast-changing component. (c) Updated model (combination of slowly and fast-changing component) for frame 48 of sequence *hall monitor*.

The second component picks up nonrigid motion. As the initialization process, it is based on the concept of moving connected components. Components that are connected to the tracked object are used to update the corresponding model by adding all edge pixels that are within a specified distance of these MCC's. The result in Fig. 7(b) shows that the left foot and right arm were picked up in contrast to the right foot, which was not moving.

The combination of both components results in an updated model that extracts slowly and fast-changing or moving components very well [Fig. 7(c)].

D. Filtering Stationary Cluttered Background

Object tracking would be fairly easy if all pixels in the edge image belonged to objects, but unfortunately, many sequences contain cluttered background [see Fig. 8(a)]. This can be a problem, and it is desirable to remove cluttered background prior to model matching and updating. Otherwise, the model update might pick up background edge points if they are close enough to the model. Note that Hausdorff matching could handle cluttered background quite well, but it is preferable to use the edge image after filtering to reduce the number of image points and therefore the computation time.

To eliminate stationary background, one could remove all edge pixels that were already edge pixels in the previous frame. However, this simple binary differencing is very sensitive to noise and would remove objects that stop moving as well. In Fig. 8(b), for example, the person's left leg, which was stationary between two frames, has been removed.

We propose a filtering technique that counts for each pixel how often it has been classified as edge. If this counter exceeds a threshold, the pixel is assumed to be part of the background and is removed. The counter is updated only for pixels that are not occluded by an object. This is achieved by updating the counter after processing a frame when the position of all objects is known. Hence, we only collect information on pixels that are really classified as background. For the first few frames, the counter cannot give reliable results and simple binary differencing of consecutive frames is applied until enough data have been collected.

Our proposed filter preserves the edge pixels belonging to objects much better than simple differencing [Fig. 8(c)]. It is less sensitive to noise, and more importantly, it works even

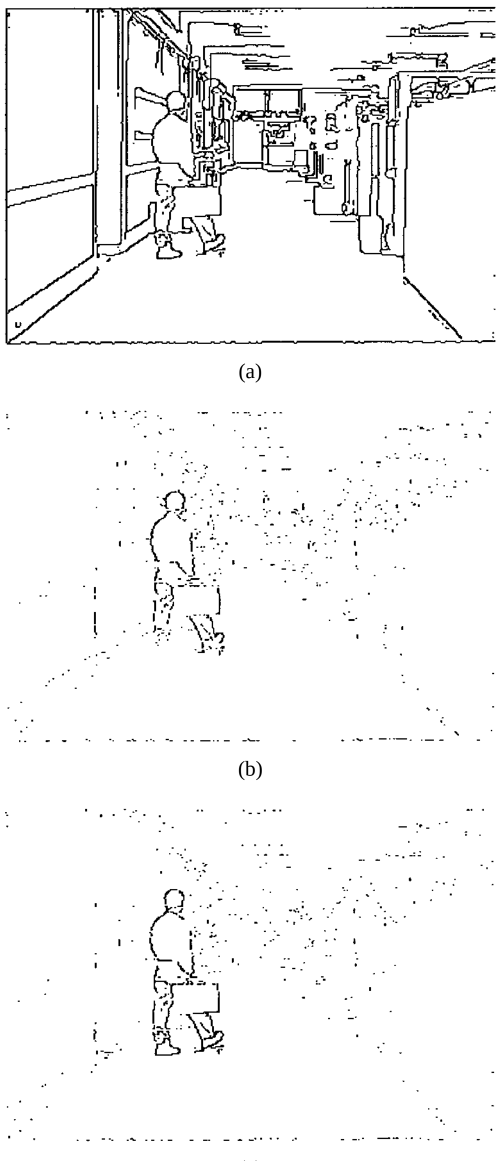


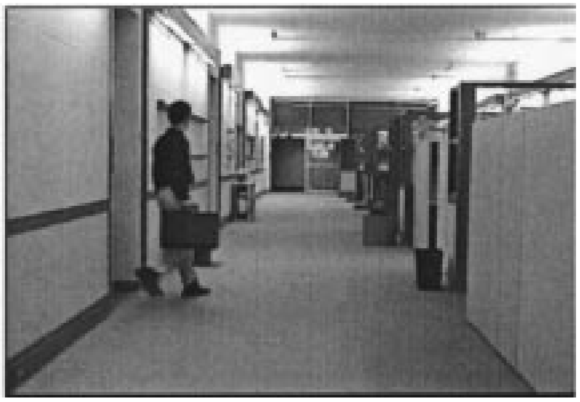
Fig. 8. (a) Binary edge image of frame 40 of sequence *hall monitor* obtained by Canny operator. (b) Simple binary difference image. (c) Binary feature image obtained by proposed method.

when an object stops for an arbitrarily long time. Since the counter is not increased at the location of objects, it will never exceed the threshold for removal.

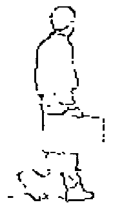
The assumption of stationary background is valid for many applications. Unfortunately, an extension to filtering moving background is not simple because global motion estimation and compensation cannot perfectly align edge images. A different approach is necessary and is currently under investigation.

E. Extraction of VOP's

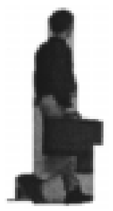
The output of the tracker is a sequence of binary edge images modeling the object of interest. The remaining step is to extract the corresponding object from the video sequence, i.e., we have to create the VOP of the object. This is done by finding the first and last model points for each row. The pixels



(a)

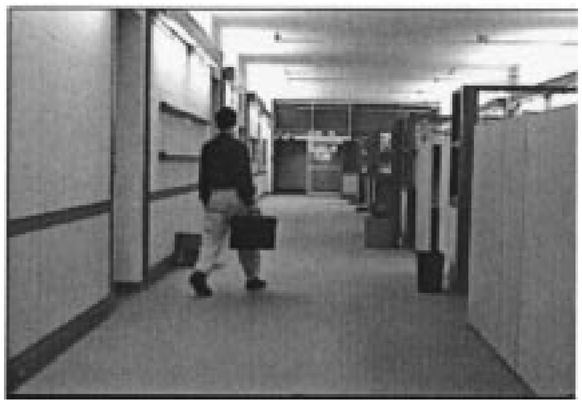


(b)

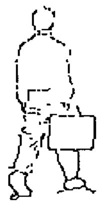


(c)

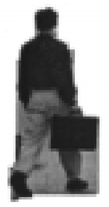
Fig. 9. (a) Original frame 32, (b) binary model, and (c) resulting VOP for sequence *hall monitor*.



(a)



(b)



(c)

Fig. 10. (a) Original frame 46, (b) binary model, and (c) resulting VOP for sequence *hall monitor*.

in between are assigned to the VOP, and the same procedure is repeated for each column. The following results section will demonstrate the performance of our proposed video sequence segmentation algorithm.

V. Results

In this section, the results of our new algorithm are given for the two test sequences *hall monitor* and *coastguard*. In *hall monitor*, the background is not moving, but very cluttered, and there is also a high level of noise present. The original frame, the binary model of the person, and the VOP for frames 32, 46, and 98 are shown in Figs. 9–11, respectively.

As can be seen, the model adapted very well to the large changes in shape. The resulting video object planes are clearly more accurate than those reported for other VOP segmentation algorithms in [25], [28], or [29] and at least as good as in [26] and [30]. The slightly jagged look of the VOP's is caused by the simple extraction technique described in Section IV-E, and improvements are possible.

The camera in the sequence *coastguard* is following the boat so that the background appears to be moving. The results in Fig. 12 show that the boat was quite well segmented. Most problems were caused by the waves below the boat, because they were temporally varying and close to the tracked object, which made the tracker include the waves into the model. The results on the same sequence reported in [28] and [29] are not as good as the ones of our proposed method.

VI. Conclusions and Future Work

A new video sequence segmentation algorithm based on object tracking was presented. A model of the object was automatically derived and matched against subsequent frames using the Hausdorff distance. To accommodate for rotation and changes in shape, the model was updated every frame by a novel update technique that consists of two components for slowly and rapidly changing or moving parts. Further, a new filtering method to improve the performance in the case of stationary background was described. Experimental results showed that the algorithm can extract video object planes from sequences with stationary and moving backgrounds.

Matching the binary model using the Hausdorff distance is remarkably robust. The new position is accurately detected even when the objects undergo large changes in shape or the background is moving. The most difficult task is to distinguish between background and objects in the initialization and update stage because the models must not pick up background. This is particularly difficult in the presence of cluttered or moving background. It has to be further investigated in filters

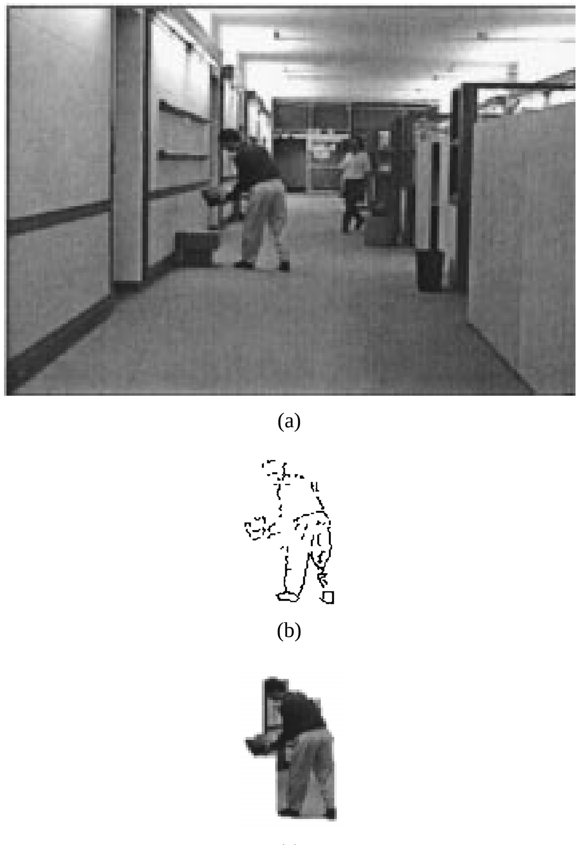


Fig. 11. (a) Original frame 98, (b) binary model, and (c) resulting VOP for sequence *hall monitor*.

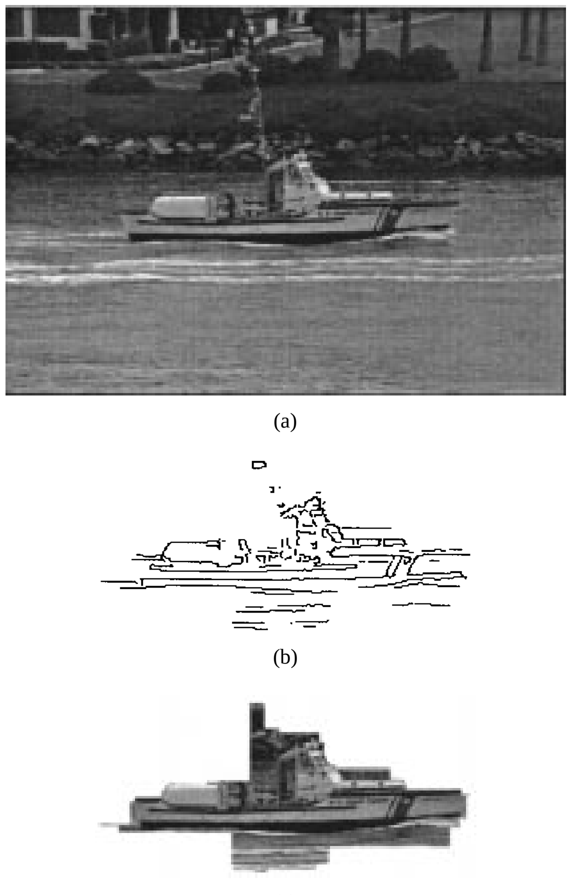


Fig. 12. (a) Original frame 150, (b) binary model, and (c) resulting VOP for sequence *coastguard*.

that can recognize background, possibly by analyzing the color or intensity of successive frames using temporal filters.

Extracting the VOP based on the binary object model is not trivial because the boundaries are not closed. In this paper, a very simple technique is used that leads to a slightly jagged look. We are currently developing a postprocessing method that can correct the boundary location of the extracted VOP's. Most VOP boundary pixels coincide with binary model points and are assumed to be correct. However, some parts of the VOP boundaries do not correspond to model points because they were artificially created by our simple extraction technique. These wrong boundaries can easily be detected by comparing the extracted VOP boundary with the binary model. Each wrong boundary is then removed, and the correct boundary between the two endpoints is determined by analyzing the binary model points. Preliminary results indicate a potential to significantly improve the boundary location.

The results of our combined segmentation and tracking algorithm are very promising when compared to those of other techniques. Nevertheless, the human visual system is still much more accurate at locating the boundaries of moving objects. At the moment, it is not possible to extract objects and to place them into other sequences. For example, the floor that is visible between the legs of the person in Fig. 10(c) does not allow that VOP to be copied into a scene with different background.

There exist some techniques such as chroma keying that achieve precise extraction of moving objects, but their applications are limited. Otherwise, we are not aware of any automatic segmentation algorithm that can accurately locate the boundaries of moving objects in generic video sequences. More research, and probably the inclusion of higher level concepts from artificial intelligence, image understanding, and *a priori* knowledge, are necessary to successfully perform segmentation of real video sequences.

APPENDIX

A. Distance Transformation

To calculate the Hausdorff distance for object matching, it is necessary to know for each pixel the distance to the nearest edge pixel. Edge pixels obviously have a distance of 0, while there horizontal and vertical neighbors, if not edge pixels themselves, have a distance of 1. For diagonal neighbors, the corresponding distance is $\sqrt{2}$ unless they or their horizontal or vertical neighbors are edge pixels.

Unfortunately, computing these distances is a global operation and computationally expensive. An algorithm that operates locally and approximates the Euclidean distance well enough is described in [41]. This so-called distance transformation (DT) defines small masks containing integer

4	3	4
3	0	3
4	3	4

(a)

	11		11	
11	7	5	7	11
	5	0	5	
11	7	5	7	11
	11		11	

(b)

Fig. 13. (a) Mask Chamfer 3-4 and (b) Chamfer 5-7-11 suggested as integer approximations of Euclidean distance for the distance transformation.

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	1	0	0	0	0	1	1
0	0	1	1	1	1	1	1
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0

(a)

7	6	7	8	8	7	4	3
4	3	4	6	6	4	3	0
3	0	3	3	3	3	0	0
4	3	0	0	0	0	0	0
7	4	3	3	3	0	3	3
8	7	6	6	3	0	3	6
11	10	9	6	3	0	3	6
14	13	10	7	4	3	4	7

(b)

Fig. 14. (a) Binary image with edge pixels having value of 1 and nonedge pixels being 0. (b) Corresponding distance transform using Chamfer 3-4 indicates for each pixel the distance to the nearest edge pixel.

approximations of distances in a small neighborhood. There are two such masks suggested, Chamfer 3-4 and Chamfer 5-7-11 (see Fig. 13). The horizontal and vertical distances for Chamfer 3-4 are 3 and the diagonal is 4. This gives a ratio of 1.333 compared to 1.414 for Euclidean distances.

The DT is initialized by assigning zero to edge pixels and infinity or a suitably large number to nonedge pixels. In two iterations, the distances are calculated by centering the mask at each pixel in turn and updating the distance of this pixel. A binary image and its distance transform using Chamfer 3-4 are given in Fig. 14. Note that the distances are about three times higher than the corresponding Euclidean distances because of the approximation made by Chamfer 3-4.

In our algorithm, the metric Chamfer 5-7-11 is used because of its higher accuracy.

B. Thinning

Thinning or erosion is a very popular tool in image processing. The idea is to erode an object until a topological skeleton of one pixel width is obtained. Many algorithms, however, do not ensure that connected components remain connected during the thinning procedure. An algorithm that does preserve connectedness by examining 3×3 neighborhoods is described in [42]. We implemented that algorithm by using an adjacency code (AC) combined with a lookup table. The adjacency code (AC) at pixel (x, y) is defined by [see Fig. 15(a)]

$$\mathrm{AC}(x, y) = \sum_{i=0}^{7} p_i \cdot 2^i \qquad (13)$$

p_0	p_1	p_2
p_7	x,y	p_3
p_6	p_5	p_4

(a)

●	○	○
○	x,y	○
○	○	●

● foreground
○ background

(b)

Fig. 15. (a) Labeling of neighbor pixels for adjacency code. (b) Example of a configuration where the pixel in question (x, y) must not be removed.

where p_i is one for foreground pixels and zero for background. Thus, there are 256 different values for $\mathrm{AC}(x, y)$, and for each one it can be determined in advance whether the pixel (x, y) should be removed or not, resulting in a very efficient lookup table. In Fig. 15(b), we have $\mathrm{AC}(x, y) = 2^0 + 2^4 = 17$, and the pixel may not be removed because otherwise, the connectedness of the top left and bottom right pixels would not be guaranteed anymore. Thus, the lookup table entry for $\mathrm{AC} = 17$ is "do not remove."

The same lookup table can be employed to remove isolated noise pixels. The corresponding adjacency code is $\mathrm{AC} = 0$, and thus, pixels with adjacency code zero are removed as well.

REFERENCES

[1] T. Sikora, "The MPEG-4 video standard verification model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 19–31, Feb. 1997.

[2] T. Ebrahimi, "MPEG-4 video verification model: A video encoding/decoding algorithm based on content representation," *Signal Processing: Image Commun.*, vol. 9, pp. 367–384, 1997.

[3] A. M. Tekalp, Ed., *Digital Video Processing.* Upper Saddle River, NJ: Prentice-Hall, 1995.

[4] A. Summerfeld, Ed., *Mechanics of Deformable Bodies.* New York: Academic, 1964.

[5] M. Hötter and R. Thoma, "Image segmentation based on object oriented mapping parameter estimation," *Signal Processing*, vol. 15, no. 3, pp. 315–334, 1988.

[6] H. G. Musmann, M. Hötter, and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images," *Signal Processing: Image Commun.*, vol. 1, pp. 117–138, 1989.

[7] G. Adiv, "Determining three-dimensional motion and structure from optical flow generated by several moving objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 384–401, July 1985.

[8] D. W. Murray and B. F. Buxton, "Scene segmentation from visual motion using global optimization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9, pp. 220–228, Mar. 1987.

[9] P. Bouthemy and E. François, "Motion segmentation and qualitative dynamic scene analysis from an image sequence," *Int. J. Comput. Vision*, vol. 10, no. 2, pp. 157–182, 1993.

[10] M. M. Chang, A. M. Tekalp, and M. I. Sezan, "Motion-field segmentation using an adaptive MAP criterion," in *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'93*, Minneapolis, MN, Apr. 1993, vol. V, pp. 33–36.

[11] M. M. Chang, M. I. Sezan, and A. M. Tekalp, "An algorithm for simultaneous motion estimation and scene segmentation," in *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'94*, Adelaide, Australia, Apr. 1994, vol. V, pp. 221–224.

[12] C. Stiller, "A statistical image model for motion estimation," in *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'93*, Minneapolis, MN, Apr. 1993, vol. V, pp. 193–196.

[13] ——, "Object-based estimation of dense motion fields," *IEEE Trans. Image Processing*, vol. 6, pp. 234–250, Feb. 1997.

[14] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. Roy. Statist. Soc. B*, vol. 36, no. 2, pp. 192–236, 1974.

[15] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 721–741, Nov. 1984.

[16] J. Besag, "On the statistical analysis of dirty pictures," *J. Roy. Statist. Soc. B*, vol. 48, no. 3, pp. 259–279, 1986.

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

[18] N. Diehl, “Object-oriented motion estimation and segmentation in image sequences,” *Signal Processing: Image Commun.*, vol. 3, pp. 23–56, 1991.

[19] F. Meyer and S. Beucher, “Morphological segmentation,” *J. Visual Commun. Image Representation*, vol. 1, pp. 21–46, Sept. 1990.

[20] P. Salembier and M. Pardàs, “Hierarchical morphological segmentation for image sequence coding,” *IEEE Trans. Image Processing*, vol. 3, pp. 639–651, Sept. 1994.

[21] P. Salembier, P. Brigger, J. R. Casas, and M. Pardàs, “Morphological operators for image and video compression,” *IEEE Trans. Image Processing*, vol. 5, pp. 881–898, June 1996.

[22] J. G. Choi, S. W. Lee, and S. D. Kim, “Spatio-temporal video segmentation using a joint similarity measure,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 279–286, Apr. 1997.

[23] F. Marqués and C. Molina, “Object tracking for content-based functionalities,” in *SPIE Visual Commun. Image Processing, VCIP’97*, San Jose, CA, Feb. 1997, vol. 3024, pp. 190–199.

[24] J. Y. A. Wang and E. H. Adelson, “Representing moving images with layers,” *IEEE Trans. Image Processing*, vol. 3, pp. 625–638, Sept. 1994.

[25] A. Neri, S. Colonnese, G. Russo, and P. Talone, “Automatic moving object and background separation,” *Signal Processing*, vol. 66, no. 2, pp. 219–232, 1998.

[26] R. Mech and M. Wollborn, “A noise robust method for segmentation of moving objects in video sequences,” in *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP’97*, Munich, Germany, Apr. 1997, vol. 4, pp. 2657–2660.

[27] T. Meier and K. N. Ngan, “Automatic segmentation based on Hausdorff object tracking,” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m2238*, Stockholm, Sweden, July 1997.

[28] S. Colonnese, U. Mascia, G. Russo, and P. Talone, “Core experiment N2: Preliminary FUB results on combination of automatic segmentation techniques,” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m2365*, Stockholm, Sweden, July 1997.

[29] R. Mech and P. Gerken, “Automatic segmentation of moving objects (partial results of core experiment n2),” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m1949*, Bristol, U.K., Apr. 1997.

[30] J. G. Choi, M. Kim, M. H. Lee, and C. Ahn, “Automatic segmentation based on spatio-temporal information,” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m2091*, Bristol, U.K., Apr. 1997.

[31] P. Gerken, R. Mech, G. Russo, S. Colonnese, C. Ahn, and M. H. Lee, “Merging of temporal and spatial segmentation,” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m1948*, Bristol, U.K., Apr. 1997.

[32] J. G. Choi, M. Kim, M. H. Lee, C. Ahn, S. Colonnese, U. Mascia, G. Russo, P. Talone, R. Mech, and M. Wollborn, “Combined algorithm of ETRI, FUB and UH on core experiment N2 for automatic segmentation algorithm of moving objects,” in *ISO/IEC JTC1/SC29/WG11 MPEG97/m2383*, Stockholm, Sweden, July 1997.

[33] T. Meier and K. N. Ngan, “Automatic video sequence segmentation using object tracking,” in *IEEE Tencon’97*, Brisbane, Australia, Dec. 1997, vol. 1, pp. 283–286.

[34] J. Canny, “A computational approach to edge detection,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 679–698, Nov. 1986.

[35] M. R. Dobie and P. H. Lewis, “Object tracking in multimedia systems,” in *Int. Conf. Image Processing Appl.*, Maastricht, The Netherlands, 1992, pp. 41–44.

[36] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, “Comparing images using the Hausdorff distance,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 850–863, Sept. 1993.

[37] D. P. Huttenlocher, J. J. Noh, and W. J. Rucklidge, “Tracking nonrigid objects in complex scenes,” in *4th Int. Conf. Comput. Vision*, Berlin, Germany, May 1993, pp. 93–101.

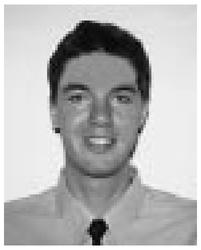
[38] A. Rosenfeld and J. L. Pfaltz, “Sequential operations in digital picture processing,” *J. ACM*, vol. 13, pp. 471–494, Oct. 1966.

[39] M. Bierling, “Displacement estimation by hierarchical blockmatching,” in *SPIE Visual Commun. Image Processing, VCIP’88*, Cambridge, MA, Nov. 1988, vol. 1001, pp. 942–951.

[40] P. J. Rousseeuw and A. M. Leroy, Eds., *Robust Regression and Outlier Detection*. New York: Wiley, 1987.

[41] G. Borgefors, “Distance transformations in digital images,” *Comput. Vision, Graphics, Image Processing*, vol. 34, pp. 344–371, 1986.

[42] E. R. Davies, Ed., *Machine Vision: Theory, Algorithms, Practicalities*. London, U.K.: Academic, 1990.



Thomas Meier received the Dipl. El.-Ing. degree in electrical engineering from the Swiss Federal Institute of Technology (ETH), Zurich, in 1993. For his excellent honors thesis, he was awarded the ETH medal.

After two years of industrial experience, he joined the Visual Communications Research Group at the University of Western Australia in 1995, where he is currently working toward the Ph.D. degree. His research interests are in video sequence analysis and segmentation, object tracking, computer vision, and reduction of coding artifacts.

King N. Ngan (M’79–SM’91), for a photograph and biography, see this issue, p. 522.