# Memory-based Spatio-Temporal Real-Time Object Segmentation for Video Surveillance

Aishy Amer

Concordia University, Electrical and Computer Engineering,
Montréal, Québec, Canada

## ABSTRACT

In real-time content-oriented video applications, fast unsupervised object segmentation is required. This paper proposes a real-time unsupervised object segmentation that is stable throughout large video shots. It trades precise segmentation at object boundaries for speed of execution and reliability in varying image conditions. This interpretation is most appropriate to applications such as surveillance and video retrieval where speed and temporal reliability are of more concern than accurate object boundaries. Both objective and subjective evaluations, and comparisons to other methods show the robustness of the proposed methods while being of reduced complexity. The proposed algorithm needs on average 0.15 seconds per image.

The proposed segmentation consists of four steps: motion detection, morphological edge detection, contour analysis, and object labeling. The contributions in this paper are: a segmentation process of simple but effective tasks avoiding complex operations, a reliable memory-based noise-adaptive motion detection, and a memory-based contour tracing and analysis method. The proposed contour tracing aims 1) at finding contours with complex structure such as those containing dead or inner branches and 2) at spatial and temporal adaptive selection of contours. The motion detection is spatio-temporal adaptive as it uses estimated intra-image noise variance and detected inter-image motion.

**Keywords:** Object segmentation, motion detection, edge detection, contour tracing, noise, video surveillance

## 1. MOTIVATION

Many video applications require the extraction and manipulation of image objects and their features. Examples are object-based motion estimation, video coding, video retrieval, and video surveillance. In applications such as surveillance, a background image is available to assist the task of segmentation.

Object segmentation is, therefore, an active field of research that has produced a large variety of segmentation methods.[1–5] Each method has emphasis on different issues. Some methods are computationally expensive but give, in general, accurate results and others have low computation but fail to provide reliable segmentation. Few of the methods are adequately tested, particularly on a large number of video shots, and are evaluated throughout large shots. Furthermore, many methods work with fine tuned parameters by experts. A drawback common to most methods is that they are not tested on noisy images and images with artifacts.

An object segmentation algorithm classifies the pixels of a video image into a certain number of classes that are homogeneous with respect to some characteristic (e.g., texture or motion). It aggregates image pixels into objects. Some methods focus on color features and others on motion features. Some methods combine various features aiming at better results. The use of more features does not, however, guarantee better result since some features can become erroneous or noisy and complicate the achievement of a good solution.

The objective in this paper is to propose an automated modular object segmentation method that stays stable throughout an image sequence. This method uses a small number of features for segmentation, but focuses on their robustness to varying image conditions such as noise. This foregoes precise segmentation such as at object boundaries. This interpretation of segmentation is most appropriate to applications such as surveillance and video database retrieval. In surveillance applications, robustness with respect to varying image and object conditions is of more concern than accurate segmentation. In object-based retrieval, the detailed outline of objects is often not necessary but the semantic meaning of these objects is important.

Send correspondence to Aishy Amer, E-mail: amer@ece.concordia.ca, Tel.: 1 514 848-4081, Fax. 514 848-2802, Address: Concordia University, 1455 de Maisonneuve, West, H-961; Montréal, Québec H3G 1M8 Canada

## 2. RELATED WORK

The MPEG-4 oriented method proposed in[6] operates on binary edge images generated with the Canny operator and tracks objects using a generalized Hausdorff distance. It uses enhancements, such as adaptive maintenance of a background image over time and improvement of boundary localization.

The AVI system[7] uses motion detection information to extract objects based on a background image. Results show that the motion detection method used in AVI is sensitive to noise and artifact. The system can operate in simple environments where one human is tracked and translational motion is assumed. It is limited to applications of indoor environments and cannot deal with occlusion.

In,[8] a segmentation method is proposed for video surveillance that is based on spatial regularization of a statistical model-based change detection. This method has difficulties in images with noise, shadow, and local illumination change. In addition, it has high computational cost.

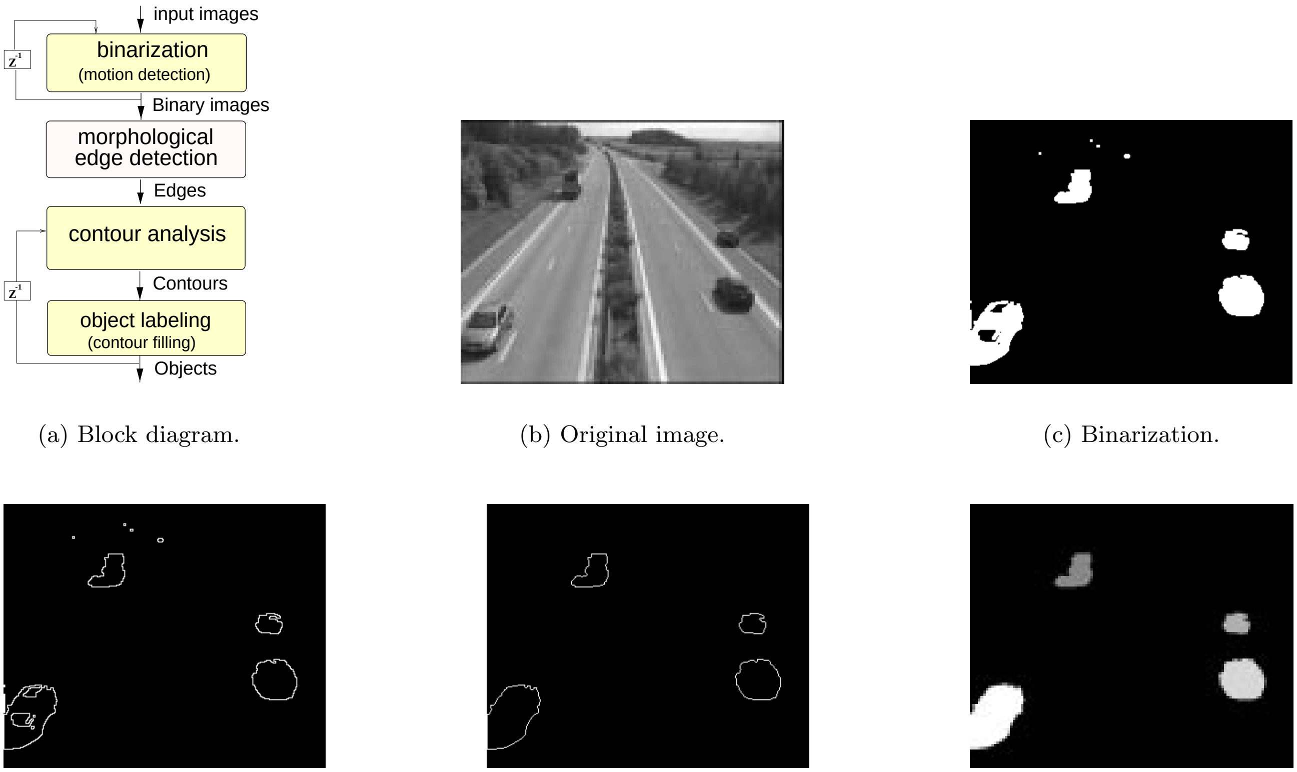
Within the framework of the video analysis model of the COST-211-Group, a new object segmentation scheme, the COST-AM scheme, has been introduced.[9–11] The basic steps of its current implementation are camera-motion compensation, color segmentation, and motion detection. Subjective and objective evaluation suggests that this method produces good object-oriented analysis of input video (see[5]). Difficulties arise when the combination of features (here motion and color) fails and strong artifacts are introduced in the resulting object masks (Fig. 9). Moreover, the method produces outliers at object boundaries where large areas of the background are estimated as belonging to objects. Big portions of the background are added to the object masks in some cases. In addition, the algorithm fails in some cases to produce temporally reliable results. For example, the method loses some objects and no object mask can be generated. This can be critical in tracking and event-based interpretation of video. One way to solve this is to use tracking to stabilize analysis. Tracking ensures coherent motion trajectories through time[12]).

Few of the methods are adequately tested particularly on video shots or image sequences with noise, artifact, and illumination changes. Many are tested in limited set of video shots.

## 3. PROPOSED APPROACH - AN OVERVIEW

This proposed segmentation method consists of simple but effective tasks, some of which are based on motion and object contour information. Segmentation is realized in four steps (Fig. 1): motion-detection-based binarization of the input gray-level images (Sec. 4), morphological edge detection,[3,13] contour analysis and tracking (Sec. 5), and object labeling.

The critical task is the motion-based binarization which must stay reliable throughout the video shot. Here, the algorithm memorizes previously detected motion to adapt current motion detection. Edge detection is performed by novel morphological operations[13] with a significantly reduced number of computations compared to traditional morphological operations. The advantage of morphological detection is that it produces gap-free and single-pixel-wide edges without need for post-processing. Contour analysis transforms edges into contours and uses data from previous frames to adaptively eliminate noisy and small contours. Small contours are only eliminated if they cannot be matched to previously extracted contours, i.e., if a small contour has no corresponding contour in the previous image. Small contours lying completely inside a large contour are merged with that large contour according to a spatial homogeneity criterion, as will be shown in Section 5. The elimination of small contours is spatially adapted to the homogeneity criterion of an object and temporally to corresponding objects in previous images. This is different from methods that delete small contours and objects based on fixed thresholds (see, for example,[1,9,14–16]). Object labeling attempts to recreate each point of the original object given its contour data.[17,18] Finding the interior of a region when its contour is given is one of the most common tasks in image analysis. Several methods for contour filling exist. The two most used are the *seed-based* method and the *scan-line* method.[17–19] In the seed-based method an interior contour point is needed as a start point, then the contour is filled in a recursive way. Since this method is not automated it is not suitable for on-line video applications. The Scan-line method is an automated technique that fills a contour line by line. This paper uses an enhanced efficient version of the Scan-Line method as described in.[20]



(a) Block diagram.

(b) Original image.

(c) Binarization.

(d) Edge detection: gap-free edges.

(e) Contour analysis: small contours and noise reduction.

(f) Object labeling: objects with unique labels.

**Figure 1**. Four-step object segmentation.

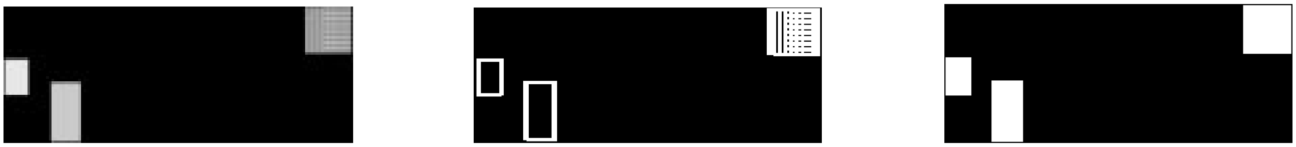
This object segmentation method is evaluated in the presence of MPEG-2-coding artifacts, white and impulsive noise, and illumination changes. Its robustness to these artifacts is shown in various simulations (Section 6). The computational cost is low and results are reliable. Few parameters are used; these are adjusted automatically to the amount of noise and to the local image content.

The result of the segmentation process is a *list of objects* with spatial descriptors to be used for further object-based video processing. To reduce storage space, object and contour points are compressed using a differential run-length code. The following spatial descriptors are used: the size defined by the area $A_i$ of an object $O_i$, its perimeter $P_i$, width $W_i$ (i.e., the maximum horizontal extent of $O_i$), and height $H_i$ (i.e., the maximum vertical extent of $O_i$); the shape defined by the minimum bounding box (MBB) $B_{O_i}$ of an object which is the smallest rectangle that includes the object, the extent ratio, $e_i = \frac{H_i}{W_i}$, the compactness, $c_i = \frac{A_i}{H_i W_i}$, the irregularity (elongation), $r_i = P_i^2/(4\pi A_i)$; the Center-of-gravity of $O_i$ is defined as the center of $B_{O_i}$; and the Euclidean distance between the centroid of an object $O_i \in I(n)$ and an object $O_p \in I(n-1)$. Note that temporal descriptors are added using a motion estimation and tracking procedures.[12, 21]

## 4. A MEMORY-BASED MOTION DETECTION METHOD

### 4.1. Introduction

Motion can be detected either between successive images or between an image and a background image. A major difficulty with techniques using consecutive images is that they depend on inter-image motion being present between every image pair. Any moving object (or part of an object in the case of articulated objects)



(a) Original image.

(b) Motion detection using a successive image.

(c) Motion detection using a background image.

**Figure 2**. Motion detection schemes.

that becomes stationary or uncovered is erroneously merged with the background. Furthermore, temporal changes between two successive images may be detected in areas that do not belong to objects but are close to object boundaries and in uncovered background as shown in Fig. 2(b). In addition, removed or deposited objects cannot be correctly detected using successive images. This paper develops an effective fast motion detection method based on image differencing with respect to a background image. The background image can be updated using a background updating technique. The disadvantages in using a background image is that shadows and reflections of moving objects can be highlighted in the difference image. The proposed method uses a thresholding technique to reduce to a minimum the typical errors of motion detection using a background image, for instance, errors associated with shadows and reflections. As will be shown, simulations show that the proposed approach successfully reduces the effect of both artifacts.

Global illumination changes can be additive and multiplicative. Assuming the images of a video shot are affected by global illumination change and by Gaussian noise. Then two successive images of the shot are modeled by:

$$\begin{aligned} I(n) &= S(n) + \eta(n) \\ I(n+1) &= S(n+1) + \eta(n+1) + \xi(n+1) \end{aligned} \tag{1}$$

where $S(n)$ and $S(n+1)$ are the projections of the scene into the image plane. $\eta(n)$ and $\eta(n+1)$ are additive noise. $\xi(n+1) = a + bS(n+1)$ represents the additive and multiplicative illumination changes. The constants $a$ and $b$ describe the strength of the global illumination changes. Thus an image difference may include artifacts due to noise and illumination changes.

## 4.2. Motion detection

The basic concept to detect motion between the current image $I(n)$ and the background image $R(n)$ is shown in Fig. 3. The method comprises spatial filtering of the difference image using a $3 \times 3$ average filter $LP$, a $3 \times 3$ maximum filter $max$ (Eq. 2), and spatio-temporal adaptation based on thresholding.

$$D(n) = max(LP(|I(n) - R(n)|)) \tag{2}$$

In real images, the difference $|I(n) - R(n)|$ includes artifacts due to noise and illumination changes. To increase spatial accuracy of detection, an average and a maximum filter are used. Averaging causes a linear addition of the correlated true image data, whereas the noise is uncorrelated and is reduced by averaging. Hence, motion detection becomes less sensitive to noise and the difference image becomes smoother. The maximum operator

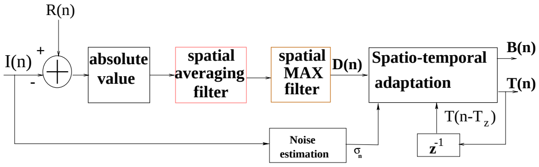


**Figure 3**. Proposed motion detection.

limits motion detection to a neighborhood of the current pixel, causing stability around object boundaries and reducing granular noisy points.

To partially compensate for global illumination changes, a binarization method based on adaptation to the difference image is used.[3] To increase temporal stability of detection throughout the video, a memory component is added to the motion detection process as shown next.

### 4.3. Spatio-temporal adaptation

Typical difficulties of motion detection based on differencing are 1) it does not distinguish between object motion and other changes, for example, due to illumination changes and 2) it does not account for changes occurring throughout a long video shot. Usually a fixed threshold is used for all images of the shot to decide on moving and non-moving image parts. A fixed threshold method fails, e.g., when the amount of moving regions changes significantly. To answer these difficulties, this paper proposes a three step thresholding method.

**1. Adaptation to noise:** First, a spatial threshold, $T_g$, is estimated using a robust thresholding method.[3] The threshold $T_g$ is adapted to the amount of image noise using the following positive-quadratic weighting:

$$T_n = T_g + c \cdot \sigma_n^2, \quad c < 1 \tag{3}$$

This weighting is a function of the noise standard deviation, $\sigma_n$, taking into consideration that low sensitivity to small $\sigma_n$ is needed. $\sigma_n$ is estimated using a new fast block-based noise estimation technique.[22]

**2. Quantization:** To further stabilize thresholding, $T_n$ is quantized to $T_q$ into $m$ values to compensate for background and local illumination changes. In our implementation, $m$ was set to 3 and the quantization is:

$$T_q = \begin{cases} T_{\text{min}} & : \; T_n \leq T_{\text{min}} \\ T_{\text{mid}} & : \; T_{\text{min}} < T_n \leq T_{\text{mid}} \\ T_{\text{max}} & : \; \text{otherwise} \end{cases} \tag{4}$$

**3. Adding memory:** To adapt detection to temporal changes throughout a video shot the following memory function is used:

$$T(n) = \begin{cases} T_{\text{min}} & : \; T_q \leq T_{\text{min}} \\ T(n-1) & : \; T_q < T(n-1) \\ T_q & : \; \text{otherwise} \end{cases} \tag{5}$$

This function examines if there has been a significant change, i.e., $T_q > T(n-1)$, in the current image and, if so, the current threshold $T_q$ is selected. If no significant temporal change is detected, i.e., $T_q < T(n-1)$, the previous threshold $T(n-1)$ is selected. When no or little motion is detected, $T_q \leq T_{\text{min}}$, $T_{\text{min}}$ is selected.

As can be seen in Fig. 4, the proposed method displays better robustness compared to a statistical motion detection method,[8] especially in images with local illumination change (for example, when the sun shines or doors are opened as in ‘Stair’ and objects enter the scene as in ‘Survey’ sequence). Also, the proposed method remains reliable in the presence of noise because it compensates for noise by adapting its parameter automatically to the amount of noise estimated in the image. Another important factor is the introduction of the temporal adaptation which makes the procedure reliable throughout the whole video shot. An additional advantage of the proposed method is that it has a low computational cost. For example, it requires an average of 0.1 seconds compared to 0.25 seconds for the reference method on a SUN-SPARC-5 360 MHz.

## 5. ADAPTIVE CONTOUR TRACING

### 5.1. Introduction

The morphological edge detection used[3, 13] gives an edge image, $E(n)$, where edges are marked white and points inside the object or of the background are black. The important advantage of morphological edge detection techniques is that they never produce edges with gaps. This facilitates contour tracing techniques.

To identify the object boundaries in $E(n)$, the white points belonging to a boundary have to be grouped in one contour, $C$. An algorithm that groups the points in a contour is called contour tracing. The result of tracing

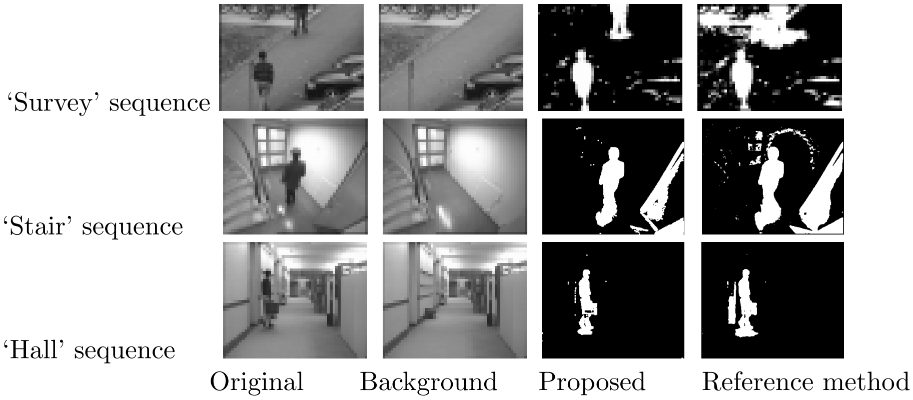


**Figure 4**. Motion detection comparison.

edges in $E(n)$ is a list, $C(n)$, of contours and their features, such as starting point and perimeter. A contour, $C \in C(n)$, is a finite set of points, $\{p_1, \cdots, p_n\}$, where for every pair of points $p_i$ and $p_j$ in $C$ there exists a sequence of points $s = \{p_i, \cdots, p_j\}$ such that i) $s$ is contained in $C$ and ii) every pair of successive points in $s$ are *neighbors* of each other. In an image defined on a rectangular sampling lattice, two types of neighborhoods are distinguished: 8-neighborhood and 4-neighborhood. In an 8-neighborhood all the eight neighboring points around a point are considered. In a 4-neighborhood only the four neighboring points, right, left, up, and down, are considered.

A contour can be represented by the point coordinates or by a chain code. A list of point coordinates is needed for on-line object analysis. A chain code requires less memory than coordinate representation and is desirable in applications that require storing the contours for later use.[17]

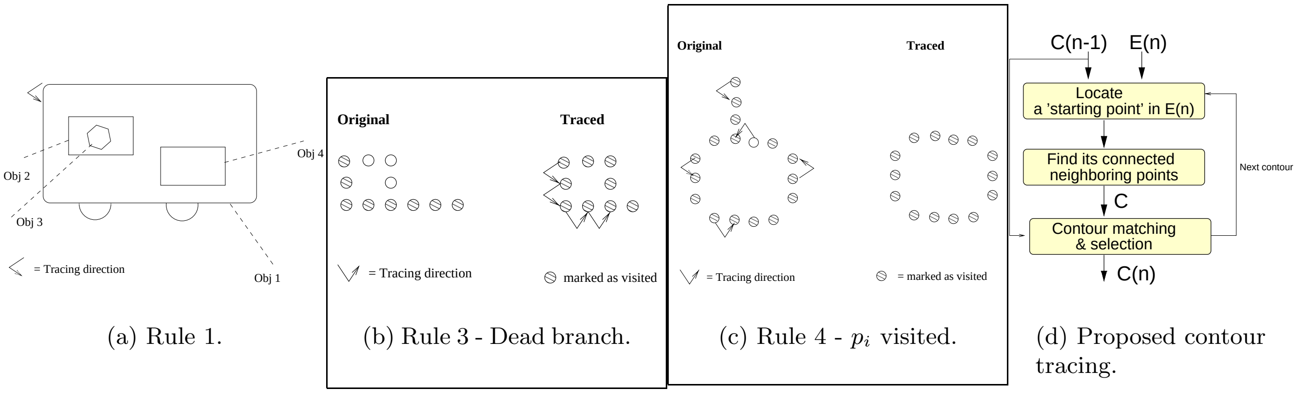
Different contour tracing techniques are reported in the literature ([17, 20]). Many methods are developed for specific applications such as pattern recognition. Some are defined to trace contours of simple structure. A commonly used technique is described in.[18] A drawback of this method is that it ignores contours inside other contours, fails to extract contours containing some 8-neighborhoods, and fails in case of contours with dead or inner branches.

### 5.2. Tracing complex contours in real images

This proposed procedure aims at tracing contours of complex structure such as those containing dead or inner branches as with contours illustrated in Fig. 5. The proposed tracing algorithm (Fig. 5(d)) uses plausibility rules i) to locate the starting point of a contour, ii) to find neighboring points, and iii) to decide whether to select a contour for subsequent processing. The tracing is in a clockwise manner and the algorithm looks for a neighboring point of a current point in the 8-neighborhood starting at the rightmost neighbors. This rule forces the algorithm to move around the object by looking for the rightmost point and never inside the object (see **Rule 2**). The algorithm records both the contour chain code and the point coordinates.

In the following, let $E(n)$ be the edge image of the original image $I(n)$, $C(n-1)$ the list of contours of the objects of the original image $I(n-1)$, $C(n)$ the list of contours of the objects of the original image $I(n)$, $C_c \in C(n)$ the current contour with starting point $p_s$, $P_c$ the length of $C_c$, i.e., the number of points of $C_c$, $p_c$ the current point, $p_i$ an 8-neighbor of it, and $p_p$ its previous neighbor.

**Rule 1 - Locating a starting point** Scan the edge image, $E(n)$, from left to right and from top to bottom until a white point $p_w$ is found. $p_w$ must be *unvisited* (i.e., not reached before) and has *at least one unvisited* neighbor. If such a point is found, i) set $p_s = p_w$, ii) set $p_c = p_s$, and iii) **perform Rule 2**. If no starting point is found, end tracing. In case objects contain other objects, the given scanning direction forces the algorithm to trace first the outwards and then the inwards contours (Fig. 5(a)). Note that due to the image scanning direction (from left to right and from top to bottom) the object boundaries lay always left of the tracing direction.



(a) Rule 1.

(b) Rule 3 - Dead branch.

(c) Rule 4 - $p_i$ visited.

(d) Proposed contour tracing.

**Figure 5**. Illustrating effects of proposed tracing rules.

**Rule 2 - Finding a neighboring point** The basic idea is to locate the rightmost neighbor of the current point. This ensures that object contours are traced from outward and tracing never enters branches inside the object. The definition of the rightmost neighbor of a current point depends on the current direction of tracing as defined by the previous and the current points. If, for example, the previous point lays to the left of the current point then the algorithm looks for a neighboring point, $p_i$, within the five neighbors of $p_c$ displayed at the upper left of Fig. 6. The other neighbors of $p_c$ were neighbors of $p_p$ and were already *visited* and there is no need to consider them. Based on the position of $p_p$ eight search neighborhoods are defined (Fig. 6). Note that the remaining neighbors of $p_c$, that are not considered are already visited when tracing $p_p$. Since the algorithm is designed to close contours when a visited point is reached (see **Rule 4**), these points should not be considered. Depending on the position of $p_p$, look for the next neighboring point $p_i$ of $p_c$ in the respective neighborhood as given in Fig. 6. If a $p_i$ is found i) mark $p_c$ as *visited* if it is not marked visited, ii) set $p_p = p_c$, iii) set $p_c = p_i$, and **perform Rule 4**. If no $p_i$ is found **perform Rule 3**, i.e., delete a dead branch.

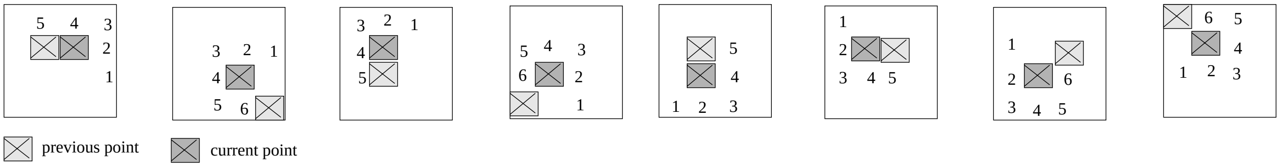


**Figure 6**. Neighborhoods of the current point.

**Rule 3 - Deleting dead branches** This rule is activated only if $p_c$ has no neighbor except $p_p$. In this case, $p_c$ is assumed to be at the end of a dead branch of the contour and the following steps are performed: i) eliminate $p_c$ from $E(n)$, ii) $p_c = p_p$, iii) $p_p$ is set to its previous neighbor (which can be easily derived from the stored chain code), and iv) **perform Rule 2**. Dead branches are points at the end of a contour and are not connected to the remaining contour (an example is given in Fig. 5(b)). In some rare cases these single-point-wide branches are part of the original object contour. In applications where reliable object segmentation foregoes the need for precision, these points provide no important information and can be deleted. Note that only single-point-wide dead branches are deleted using this rule. The elimination of dead branches facilitates subsequent steps of object-oriented video analysis.

**Rule 4 - Closing a contour** Close $C_c$, eliminate its points from $E(n)$, and **perform Rule 5** if $p_c = p_s$ or $p_c$ is marked visited. If $p_c$ is marked visited, eliminate the remaining dead points of $C_c$ from $E(n)$. If $C_c$ is not closed i) store the coordinate and chain code of $p_c$ and ii) look for the next point, i.e., **perform Rule 2**. Note that this rule closes a contour even if the starting point is not reached. This is important in case of errors in previous segmentation steps that produce, for instance, dead branches (Fig. 5(c)).

**Rule 5 - Selecting a contour** Do not add $C_c$ to $C(n)$ if

1) $P_c$ is too small, i.e., $P_c < t_{p_{\min}}$ where $t_{p_{\min}}$ is a threshold,
2) $P_c$ is small (i.e., $t_{p_{\min1}} < P_c < t_{p_{\min2}}$ where $t_{p_{\min2}}$ is a threshold) and $C_c$ has no corresponding contour in $C(n-1)$, or
3) $C_c$ is inside a previously traced contour $C_p$ so that the spatial homogeneity[23] of the object of $C_p$ is low. A spatial homogeneity measure of an object $O_i$ describes the connectivity of its pixels.

Otherwise add $C_c$ to $C(n)$. In both cases, **perform Rule 1**.
With rule 5, small contours are assumed to be the result of noise or erroneous thresholding and are, therefore, eliminated if they have no corresponding contours in the previous image. The elimination of small contours (representing small objects) is spatially adapted to the homogeneity criterion of an object and temporally to corresponding objects in previous images. This is different from many methods that delete small objects based on a fixed threshold (see, for example,[1, 9, 14–16]).

# 6. EVALUATION OF THE PROPOSED SEGMENTATION

## 6.1. Evaluation criteria

Evaluation criteria for object segmentation can be distinguished into two groups:

i) Criteria based on implementation and architecture efficiency: implementation efficiency is measured by *memory use* and *computational cost*, i.e., the time needed to segment an image. Important parameters are image size, frame rate and computing system (e.g., multitasking computers or computers with specialized hardware). Architectural performance is evaluated by the level of human *supervision*, level of *parallelism*, and *regularity*, which means that similar operations are performed at each pixel.
ii) Criteria based on the quality of the segmentation results, e.g., *spatial accuracy* and *temporal stability.*

It is important to be able to evaluate segmentation results using some objective numerical measures similar to PSNR in comparing coding and enhancement algorithms. Recently, an objective evaluation measure has been introduced.[24] It measures the spatial accuracy, temporal stability, and temporal coherence of the estimated object masks relative to reference masks. The spatial accuracy ($sQM(dB)$) is measured in terms of the number and location of the differences between estimated and reference masks. The $sQM$ is 0 for an estimated segmentation identical to the reference and grows with deviation from the reference, indicating lower quality of segmentation. The temporal stability ($vQM(dB)$) is measured in terms of fluctuating spatial accuracy with respect to the reference segmentation. The temporal coherency ($vGC(dB)$) is measured by the relative variation of the gravity centers of both the reference and estimated masks. Both $vQM$ and $vGC$ are zero for perfect segmentation throughout the video. The higher the temporal stability values $vQM$ and $vGC$, the less stable the estimated segmentation over time. If all values are zero, the segmentation is perfect with respect to a reference.

## 6.2. Evaluation and comparison

In this section, simulation results using commonly referenced shots are given and discussed. These results are compared with the current state-of-the-art segmentation method,[9–11] the COST-AM method, based on both objective and subjective evaluation criteria. The reference method is described in Section 2.

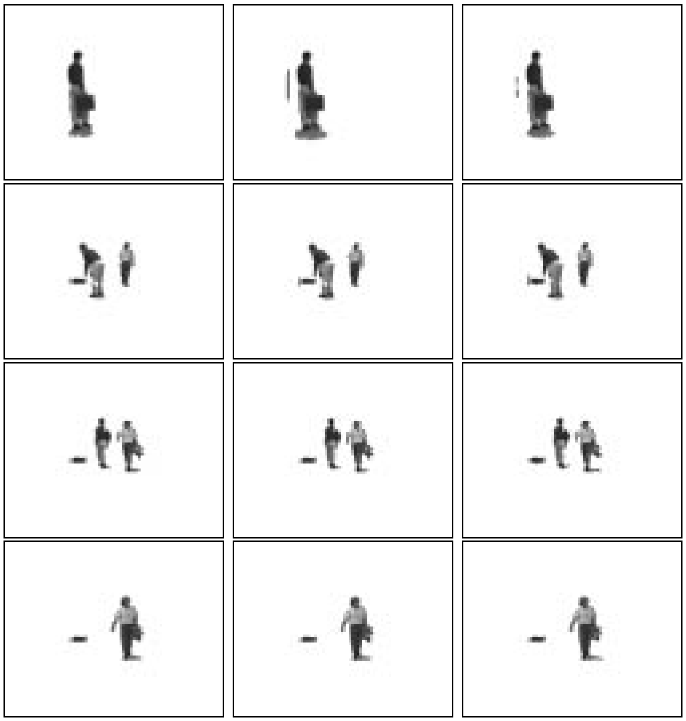
**Automatic operation** The proposed method does not use a priori knowledge, and significant low-level parameters and thresholds are adaptively calculated.

**Parallelism and regularity** All the elements of the algorithm have regular structure: motion detection with filtering and thresholding, morphological edge detection, contour tracing, and filling. A rough analysis shows that the process of motion detection and thresholding can be performed using parallel processing units. Edge detection, contour tracing, and filling are sequential methods.

**Computational cost** The proposed algorithm needs on average 0.15 seconds on a SUN-SPARC-5 360 MHz per image. Most of the computational cost is needed for motion detection and thresholding (Table 1). The the reference method, COST-AM, needs roughly 95 seconds (without global motion estimation). As reported in,[5] the COST-AM method needs on average 45 seconds on a PC-Pentium-II 333 MHz.

| | |
|---|---|
| Binarization | 0.1 - 0.15 |
| Morphological edge detection | 0.01 |
| Contour analysis | 0.01 |
| Contour filling | 0.001-.01 |

**Table 1**. Segmentation time in seconds for the proposed method.



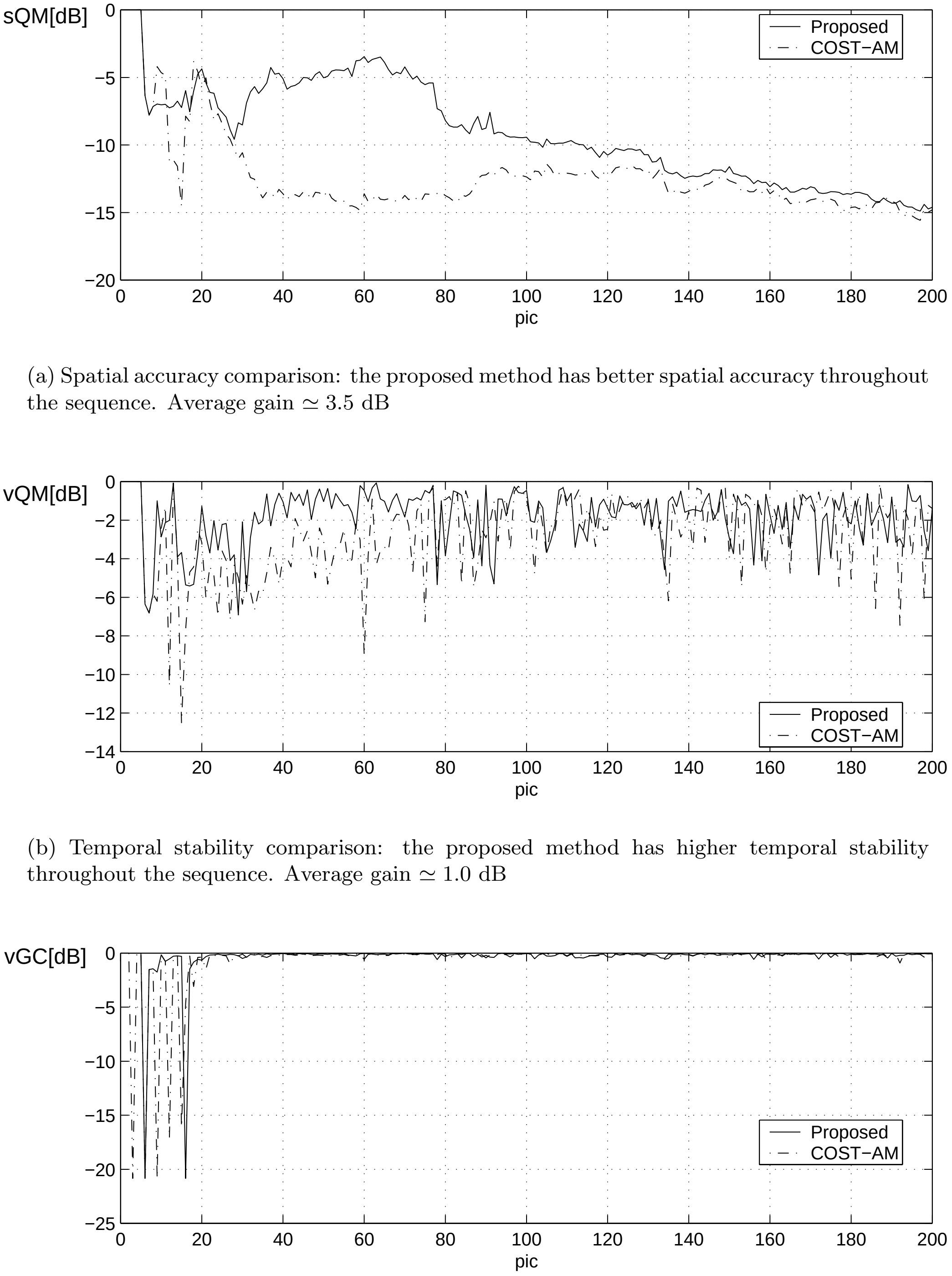
Original objects MPEG-2 objects Noisy objects

**Figure 7.** Object segmentation comparison for the 'Hall' test sequence in case of MPEG-2 decoded (25 dB) and noisy (30 dB) sequences. The proposed method is robust with respect to noise and artifacts.

**Quality of results** In our evaluation, both indoor and outdoor test sequences and small and large objects are considered. All simulations are performed with the same parameter settings. Performance of the proposed segmentation is evaluated in the presence of MPEG-2 artifacts and noise. In Fig. 7 the robustness of the proposed method in such environments is demonstrated. Note that the MPEG-2 images have an average of 25.90 dB PSNR, which means that the MPEG-2 images are strongly compressed and include many artifacts, while the noisy images have a Gaussian white noise of 30 dB.

In addition, the proposed segmentation is objectively compared to the current version (4.x) of the COST-AM method. (The evaluation software and the reference masks are courtesy of the COST-211-group, http://www.tele.ucl.ac.be/EXCHANGE/.) Fig. 8 shows the proposed segmentation is better with respect to all three criteria; especially it yields higher spatial accuracy.

In Fig. 9 subjective results are given for sample sequences and compared to the reference algorithm COST-AM. The segmented object masks using the proposed method are more accurate than those of the reference method. In the masks generated by the COST-AM method, parts of moving objects are often not detected, or large background areas are added to the object masks. Both spatial and temporal coherence of the estimated object masks are better for the proposed method than the COST-AM method. In case of small objects and outdoor scenes, the proposed method stays stable and segments all objects of the scene. As shown in results of the 'Hall' and 'Urbicande' sequences, the reference method loses objects in some images, which is critical for object tracking applications. In few cases, the COST-AM method results in more accurate object boundaries than the proposed method. This is a result of using color image segmentation in the COST-AM method.



(a) Spatial accuracy comparison: the proposed method has better spatial accuracy throughout the sequence. Average gain ≃ 3.5 dB

(b) Temporal stability comparison: the proposed method has higher temporal stability throughout the sequence. Average gain ≃ 1.0 dB

(c) Temporal coherency comparison: after the first object enters the scene, the proposed method has higher temporal coherency. Average gain ≃ 0.5 dB

**Figure 8.** Objective evaluation obtained for the 'Hall' test sequence. The proposed segmentation is better with respect to all three criteria.

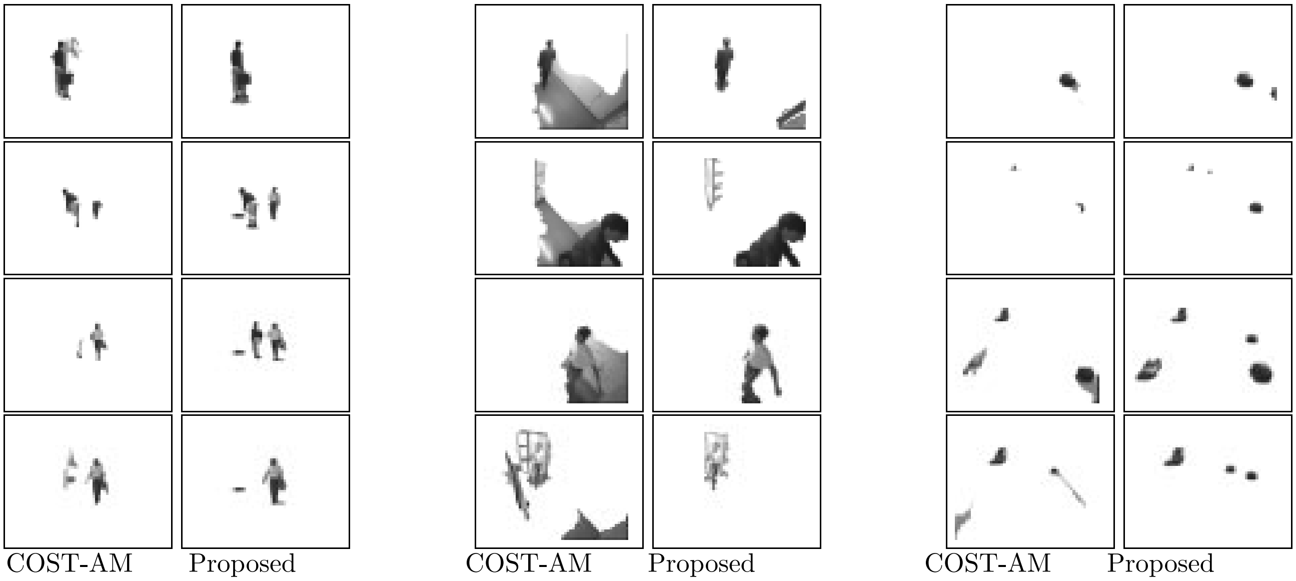


**Figure 9.** Comparison of results of the indoor 'Hall' and 'Stair', and the outdoor 'Highway' test sequences. In all tests, the reference method has lower temporal and spatial stability compared to the proposed method.

## 7. CONCLUSION

This paper has proposed a fast automated object segmentation which consists of four steps: motion-detection based binarization, morphological edge detection, contour analysis, and object labeling. The originality of our approach is : 1) the segmentation process is divided into simple but effective tasks so that complex operations are avoided, 2) a fast robust motion detection is proposed which uses a novel memory-based thresholding technique, 3) new morphological operations are introduced that show significantly reduced computations and equal performance compared to standard morphological operations, and 4) a new contour analysis method that effectively traces complex contours and adapts contour selection to spatial and temporal criteria.

Our method focuses on robustness to varying image conditions that foregoes precision (at object boundaries). This interpretation is most appropriate to applications such as video surveillance. Both objective and subjective comparisons show the robustness of the proposed methods in noisy images and in images with illumination changes while being of reduced complexity. The segmentation method uses few parameters, and these are automatically adjusted to noise and temporal changes within a video shot.

Further research is planed to enhance the performance of the proposed method in the presence of shadows. Some systems apply strategies to reduce shadows.[25,26] This might increase, however, the computational cost. We propose to compensate for the shadow artifacts in higher-level processing, e.g., at the tracking level.[12] We also plan to introduce motion detection information from successive images in addition to background data.

## ACKNOWLEDGMENTS

This work was supported, in part, by the the Natural Sciences and Engineering Research Council of Canada under Strategic Grant SRT224122 and Research Grant OGP0004234. The author thanks Prof. Eric Dubois and Prof. Amar Mitiche for their valuable reading and comments.

## REFERENCES

1. P. Salembier and F. Marqués, "Region-based representations of image and video: Segmentation tools for multimedia services," *IEEE Trans. Circuits Syst. Video Technol.* **9**(8), pp. 1147–1169, 1999.
2. "Special issue on segmentation, description, and retrieval of video content." *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 5, Sept. 1998.

3. A. Amer and E. Dubois, "Image segmentation by robust binarization and fast morphological edge detection," in *Proc. IAPR/CIPPRS Int. Conf. on Vision Interface*, pp. 357–364, (Montréal, Canada), May 2000.
4. "Workshop on image analysis for multimedia interactive services." *Proc. COST211ter*, Louvain-la-Neuve, Belgium, June 1997.
5. "Call for analysis model comparisons." *On-line COST211ter, http://www.tele.ucl.ac.be/EXCHANGE/.*
6. T. Meier and K. Ngan, "Automatic segmentation of moving objects for video object plane generation," *IEEE Trans. Circuits Syst. Video Technol.* **8**, pp. 525–538, Sept. 1998. Invited paper.
7. J. Courtney, "Automatic video indexing via object motion analysis," *Pattern Recognit.* **30**(4), pp. 607–625, 1997.
8. F. Ziliani and A. Cavallaro, "Image analysis for video surveillance based on spatial regularization of a statistical model-based change detection," in *Proc. Int. Conf. on Image Analysis and Processing*, pp. 1108–1111, (Venice, Italy), Sept. 1999.
9. R. Mech and M. Wollborn, "A noise robust method for 2-D shape estimation of moving objects in video sequences considering a moving camera," *Signal Process.* **66**(2), pp. 203–217, 1998.
10. A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tunceland, and T. Sikora, "Image sequence(1) analysis for emerging interactive multimedia services - the European COST 211 Framework," *IEEE Trans. Circuits Syst. Video Technol.* **8**, pp. 802–813, Nov. 1998.
11. M. Gabbouj, G. Morrison, F. Alaya-Cheikh, and R. Mech, "Redundancy reduction techniques and content analysis for multimedia services - the European COST 211quat Action," in *Proc. Workshop on Image Analysis for Multimedia Interactive Services*, pp. 1251–1255, (Berlin, Germany), May 1999.
12. A. Amer, "Voting-based simultaneous tracking of multiple video objects," in *Proc. SPIE Int. Conf. on Image and Video Communications and Processing*, (Santa Clara, California, USA), Jan. 2003. to appear.
13. A. Amer, "New binary morphological operations for low-cost boundary detection," *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)* , Mar. 2003. in press, World Scientific Publishers.
14. L. Garrido, P. Salembier, and D. Garcia, "Extensive operators in partition lattices for image sequence analysis," *Signal Process.* **66**, pp. 157–180, 1998.
15. P. Salembier, L. Garrido, and D. Garcia, "Image sequence analysis and merging algorithm," in *Proc. Int. Workshop on Very Low Bit-rate Video*, pp. 1–8, (Linkoping, Sweden), July 1997. Invited paper.
16. I. Haritaoglu, D. Harwood, and L. S. Davis, "$W^4$: Real-time surveillance of people and their activities," *IEEE Trans. Pattern Anal. Machine Intell.* **22**, pp. 809–830, Aug. 2000.
17. T. Pavlidis, "Contour filling in raster graphics," in *Proc. SIGGRAPH*, pp. 29–36, (Dallas, Texas), Aug. 1981.
18. T. Pavlidis, *Algorithms for Graphics and Image Processing*, Computer Science Press, Maryland, 1982.
19. J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, Reading, MA, 1990. Second edition.
20. S. Reichert, "Comparison of contour tracing and filling methods," Master's thesis, Dept. Elect. Eng., Univ. Dortmund, Feb. 1995. In German.
21. A. Amer and E. Dubois, "Real-time motion estimation by object-matching for high-level video representation," in *Proc. IAPR/CIPPRS Int. Conf. on Vision Interface*, pp. 31–38, (Calgary, Canada), May 2002.
22. A. Amer, *Object and Event Extraction for Video Processing and Representation in On-Line Video Applications.* PhD thesis, INRS-Télécommunications, Univ. du Québec, Dec. 2001.
23. J. Flack, *On the Interpretation of Remotely Sensed Data Using Guided Techniques for Land Cover Analysis.* PhD thesis, EEUWIN Center for Remote Sensing Technologies, Feb. 1996.
24. P. Villegas, X. Marichal, and A. Salcedo, "Objective evaluation of segmentation masks in video sequences," in *Proc. Workshop on Image Analysis for Multimedia Interactive Services*, pp. 85–88, (Berlin, Germany), May 1999.
25. J. Stauder, R. Mech, and J. Ostermann, "Detection of moving cast shadows for object segmentation," *IEEE Trans. on Multimedia* **1**(1), pp. 65–76, 1999.
26. P. Rosin and T. Ellis, "Image difference threshold strategies and shadow detection," in *Proc. British Machine Vision Conf.*, pp. 347–356, (Birmingham, UK), 1995.