Voting-based simultaneous tracking of multiple video objects

Aishy Amer

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

ABSTRACT

In the context of content-oriented applications such as video surveillance and video retrieval this paper proposes a stable object tracking method based on both object segmentation and motion estimation. The method focuses on the issues of speed of execution and reliability in the presence of noise, coding artifacts, shadows, occlusion, and object split.

Objects are tracked based on the similarity of their features in successive images. This is done in three steps: object segmentation and motion estimation, object matching, and feature monitoring and correction. In the first step, objects are segmented and their spatial and temporal features are computed. In the second step, using a non-linear voting strategy, each object of the previous image is matched with an object of the current image creating a unique correspondence. In the third step, object segmentation errors, such as when objects occlude or split, are detected and corrected. These new data are then used to update the results of previous steps, i.e., object segmentation and motion estimation. The contributions in this paper are the multi-voting strategy and the monitoring and correction of segmentation errors.

Extensive experiments on indoor and outdoor video shots containing over 6000 images, including images with multi-object occlusion, noise, and coding artifacts have demonstrated the reliability and real-time response of the proposed method.

Keywords: Video object, tracking, non-linear voting, occlusion, region merging, object prediction.

1. INTRODUCTION

Object tracking can be used in many video applications. For example, it facilitates the interpretation of video content and extraction of high-level description of temporal object behavior.[1] High-level descriptions are needed in content-oriented video applications such as surveillance or retrieval.[2–6] Tracking can also be used to assist estimation of coherent motion trajectories and to support object segmentation (cf.[7]).

Tracking of objects throughout a video is possible under the assumptions that object motion is smooth and objects do not disappear or change direction suddenly. Tracking of objects in real scenes is a difficult task because of 1) image changes, such as noise, shadows, light changes, reflection, and clutter, that can obscure object features to mislead tracking, 2) the presence of multiple moving objects, especially when objects have similar features, when their paths cross, or when they occlude each other, 3) the presence of non-rigid and articulated objects and their non-uniform features, 4) inaccurate of preceded object segmentation, 5) changing object features, e.g., due to object deformation or scale change, and of 6) application related requirements, such as real-time processing.

While object tracking has been extensively studied, limited work has been done to develop fast but reliable methods to be used in real-time applications such as video surveillance. The goal of this paper is to develop a method of tracking that addresses these difficulties in the context of content-oriented applications such as video surveillance and video retrieval. The proposed method aimed at assisting the extraction of high-level video content such as events.[8] We propose a fast stable object tracking method that is based on a non-linear object feature voting scheme that particularly accounts for object occlusion and splitting.

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

The remainder of the paper is organized as follows: Section 2 gives an overview of related work; Section 3 presents an overview of our tracking method; Section 4 presents our choice of features used for object matching and Section 5 proposes a non-linear voting strategy for tracking. Section 6 gives an overview of our approach to filter faulty object features (6.1), proposes a method to handle multi-object occlusion (6.2), and presents an approach to merge object regions in the case of object splitting (6.3). Section 7 describe experiments on real image sequences and Section 8 contains a conclusion.

2. RELATED WORK

Techniques for object tracking are numerous.[3, 7, 9–16] Two strategies can be distinguished: one uses correspondence to match objects between successive images (e.g.,[11, 15, 16]) and the other performs explicit tracking using a position prediction strategy or motion estimation (e.g.,[7, 11, 14, 17]). Explicit tracking approaches model occlusion implicitly but have difficulty detecting entering objects without delay and to track multiple objects simultaneously. Furthermore, they assume that object features remain invariant in time.[14] Most of these methods have high computational costs and are not suitable for real-time applications. Tracking based on correspondence tracks object, either by estimating their trajectory or by matching their features. In both cases some form of object prediction is used to handle, for instance, occlusion of objects. Prediction techniques can be based on Kalman filters or on motion estimation and compensation. The use of a Kalman filter[9, 12, 15, 16] relies on an explicit trajectory model. In complex scenes, the definition of an explicit trajectory model is difficult and can not be easily generalized.[11] Kalman filtering is noise sensitive and does not, usually, recover its target when lost.[11] Extended Kalman filters can estimate tracks in some occlusion cases but have difficulty when the number of objects and artifacts increases.

Few methods have considered real environments with multiple rigid or/and articulated objects and limited solutions to the occlusion problem exist (examples are[3, 15]). These methods track objects after and not during occlusion. In addition, many methods are designed for specific applications[13, 14, 16] (e.g., tracking based on body-part models or vehicle models) or impose constraints regarding camera or object motion (e.g., upright motion).[3, 15] Many object tracking approaches based on feature extraction assume that the object topology is fixed throughout the image sequence. In this paper, the object to be tracked can be of arbitrary shape and no prior knowledge or object models are assumed.

3. PROPOSED APPROACH - AN OVERVIEW

In this paper, a method to track multiple moving objects in the presence of occlusion is proposed. The method is able to handle objects crossing paths. No constraints are imposed on the motion of objects or on the camera position. The method is developed for applications such as video surveillance and video retrieval.

In this method, objects are tracked based on the similarity of their features in successive images. This is done in three steps: object segmentation and motion estimation, object matching, and feature monitoring and correction (Fig. 1). In the first step, the object segmentation and motion estimation modules segment objects and compute their spatial and temporal features.[18, 19]

In the second step, using a voting-based feature integration, *each* object O_p of the previous image $I(n-1)$ is matched with an object O_i of the current image $I(n)$ creating a unique correspondence $M_i : O_p \rightarrow O_i$. This means that all objects in $I(n-1)$ are matched with objects in $I(n)$. In this step, each tracked object is assigned an *identity* throughout the image sequence. M_i provides a temporal link between objects to determine the trajectory of each object throughout the video and allows a semantic-based interpretation of the input video.[8]

Solving the correspondence problem, i.e., a unique correspondence M_i, in ambiguous conditions is the challenge of object tracking. The important goal is not to lose any objects while tracking. Ambiguities arise in the case of multiple matches, when one object corresponds to several objects or in the case of zero match $M_0 : O_p \dashv$ when an object O_p cannot be matched to any object in $I(n)$. This can happen, for example, when objects split, merge, or are occluded. Further ambiguity arises when the appearance of an object varies from one image to the next. This can be a result of erroneous segmentation, changes in lighting conditions or in viewpoint.

In the third step, object segmentation errors, such as when object occlude or are split, are detected and corrected. These new data are then used to *update* the results of previous steps, i.e., object segmentation and

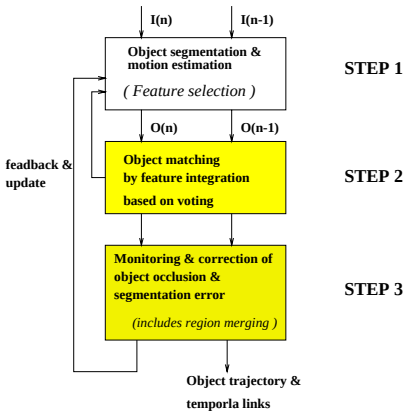

Figure 1. Framework of the proposed tracking method.

motion estimation (observe the feedback loops in Fig. 1). For example, the error correction steps can produce new objects after detecting occlusion. Motion estimation and tracking need to be performed for these new objects.

Tracking is activated once an object enters the scene. An entering object is immediately detected by the change detection module. The segmentation and motion estimation modules extract then the relevant features for the correspondence module. While tracking objects, the segmentation module keeps looking for new objects entering the scene. Once an object is in the scene, it is assigned a new trajectory. Objects that have no correspondence are assumed to be new, entering or appearing, and are assigned new trajectory. In the case of multiple object occlusion, the occlusion detection module first detects occluded and occluding objects and then continues to track both types of objects even if objects are completely occluded. This is important in the case objects reappear.

Object tracking is achieved by matching single object features and then combining the matches based on a voting scheme. Such a multi-feature based solution has to address the problems of feature selection, integration, monitoring, correction, and filtering. Feature selection defines good features for matching. Feature integration defines ways to efficiently combine features. Feature monitoring detects errors and adapts the tracking process to these errors. Feature correction compensates for segmentation errors during tracking, particularly during occlusion. Feature filtering is concerned with ways to monitor and eventually filter out harmful features during tracking. In the following sections, the three steps of our tracking methods are proposed: feature selection, feature integration, and feature monitoring and correction.

4. FEATURES FOR TRACKING

In the first step, spatio-temporal object features are extracted and selected. Object features can be extracted using any object segmentation and motion estimation method. Here, we have used the method in.[19–21] In this paper, we propose feature descriptions that balance the requirements of being effective and efficient for a real-time application. The proposed descriptors are simple but efficient when combined. In the following, let O_i represent an object of the current image $I(n)$ and O_p an object in the previous image $I(n-1)$.

- Size: the size is described by the area A_i of the 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).
- Shape: we use the following descriptors: 1) Minimum bounding box (MBB) B_{O_i}: the MBB of an object is the smallest rectangle that includes the object; 2) Extent ratio: $e_i = \frac{H_i}{W_i}$; 3) Compactness: $c_i = \frac{A_i}{H_i W_i}$; 4) Irregularity (elongation): $r_i = P_i^2/(4\pi A_i)$. This ratio increases when the shape becomes irregular or when its boundaries become jerky. The perimeter is squared to make the ratio independent of the object size. r_i is invariant to various transformations.[22]

- Motion: object motion is described by 1) the current displacement vector $w_i = (w_x, w_y)$ of O_i and 2) the horizontal and vertical direction of the object $\delta_i = (\delta_x, \delta_y)$).
- Center-of-gravity: the center-of-gravity of O_i is defined as the center of B_{O_i}.
- Distance: the Euclidean distance between the centroid of an object $O_i \in I(n)$ and an object $O_p \in I(n-1)$.

5. FEATURE INTEGRATION BY VOTING

When matching two objects using several features, a question is how to combine these features for stable tracking. Many methods combine features linearly using a weighting function. A linear combination does not, however, 1) take into account the non-linear properties of the human visual system (HVS) when tracking objects, 2) consider the distinguishing power of single feature and 3) monitor the effectiveness, that can vary in time, of a feature.

Here, we combine spatial and temporal features using a non-linear voting scheme consisting of two steps: voting for object features of two objects (object voting) and voting for features of two correspondences in the case one object is matched to two objects (correspondence voting). Each voting step is first divided into m sub-votes with m object features. Since features can become harmful or occluded, the value m varies spatially (objects) and temporally (throughout the image sequence) depending on a spatial and temporal filtering. Then each sub-vote, m_i, is performed separately using an appropriate voting function. When the voting function is applied either a similarity variable s or a non-similarity variable d is increased. Depending on the number of features in a sub-vote, m_i, s or d may increase by one or more. Finally, a majority rule compares the two variables and decides about the final vote. The simplicity of the two-step non-linear feature combination which uses distinctive features based on properties of the HVS provides a good basis for fast and efficient matching which is illustrated in the results sections.

In the case of zero match $\dashv O_i$, i.e., no object in $I(n-1)$ can be matched to an object in $I(n)$, a new object is declared entering or appearing into the scene depending on its location. In the case of reverse zero match $O_p \dashv$, i.e., no object in $I(n)$ can be matched to an object in $I(n-1)$, O_p is declared disappearing or exiting the scene which depends on its location. Note that the voting system requires the definition of some thresholds. These thresholds are important to allow variations due to feature estimation errors. The thresholds are adapted to the image and object size as will be shown in the next two sections (see also Section 6.1).

5.1. Object voting

In this step, three main feature votes are used: shape, size, and motion vote. The use of multiple votes aims at avoiding cases where one feature fails and the tracking module loses the object (especially in the case of occlusion). Define 1) O_p, an object of the previous image $I(n-1)$; 2) O_i, the i^{th} object of the current image $I(n)$; 3) $M_i : O_p \rightarrow O_i$, a correspondence and $\bar{M}_i = O_p \nrightarrow O_i$ a non-correspondence between O_p and O_i; 4) d_i, the distance between the centroids of O_p and O_i; 5) t_r, the radius of a search area around O_p; 6) $w_i = (w_{x_i}, w_{y_i})$, the estimated displacement of O_p relative to O_i; 7) $w_{\max}$, the maximal possible object displacement (typically $15 < w_{\max} < 32$); 8) s, the variable to count the similarity between O_p and O_i; 10)d, the variable to count the dissimilarity between O_p and O_i; 11) s_{++}, an increase of s by one vote; and 12) d_{++}, an increase of d by one vote.

Two objects, O_p and O_i, match if, where t_m is a threshold,

$$\begin{aligned} M_i &: \quad (d_i < t_r) \wedge (w_{x_i} < w_{x_{\max}}) \wedge (w_{y_i} < w_{y_{\max}}) \wedge (\zeta > t_m) \\ \bar{M}_i &: \text{ otherwise} \end{aligned} \tag{1}$$

with the vote confidence $\zeta = \frac{s}{d}$. M_i is accepted if O_i lays within a search area of O_p, its displacements is not larger than a maximal displacement, and if both objects are similar, i.e., $\frac{s}{d} > t_m$. The use of this rule instead of the majority rule (i.e., $s > d$) is to allow the acceptance of M_i even if $s < d$. This is important in the case objects are occluded, where some features are significantly dissimilar and might cause the rejection of a good

correspondence. Note that this step is followed by a correspondence step and no error can be introduced because of accepting correspondences with eventually dissimilar objects.

For each correspondence $M_i : O_p \rightarrow O_i$ a confidence measure ζ_i that measures the degree of certainty of M_i is used, defined as follows:

$$\zeta_i = \begin{cases} \frac{d-s}{v} & : \quad \frac{s}{d} < t_m \\ \frac{s-d}{v} & : \quad \frac{s}{d} > t_m \end{cases} \tag{2}$$

where v is the total number of feature votes.

To compute the similarity variable s and the dissimilarity variable d between two objects O_i and O_j, three feature vote functions are applied as defined in Appendix Section A. The three functions are based on the features shape, size, and motion (direction).

5.2. Correspondence voting

Recall that *all* objects $I(n-1)$ are matched against *all* objects of $I(n)$. Each object $O_p \in I(n-1)$ is matched to each object $O_i \in I(n)$. This may result in multiple matches for one object, for example, ($M_{pi} : O_p \rightarrow O_i$ and $M_{pj} : O_p \rightarrow O_j$) or ($M_{pi} : O_p \rightarrow O_i$ and $M_{qi} : O_q \rightarrow O_i$) with $O_p, O_q \in I(n-1)$ and $O_i, O_j \in I(n)$. If the final correspondence voting results in $s_i \approx s_j$, i.e, two objects of $I(n)$ are matched with the same object in $I(n-1)$, or $s_p \approx s_q$, i.e., two objects of $I(n-1)$ are matched with the same object in $I(n)$ plausibility rules are applied to resolve the ambiguity, as follows: Let s_i (s_j) be the variable that describes if M_i (M_j) is the better correspondence. Then

$$\begin{aligned} M_i &: (s_i > s_j) \wedge (\zeta_i > \zeta_j) \\ M_j &: s_i \leq s_j \end{aligned} \tag{3}$$

A simple majority voting rule is applied here.

To compute the similarity variable s_i and the dissimilarity variable s_j between two correspondences, M_i and M_j five vote functions are applied as defined in Appendix Section B. The five functions are based on the features distance, confidence, size, shape, and motion (direction and displacement).

6. FEATURE MONITORING AND CORRECTION

A good tracking technique must account for errors of previous steps. Object segmentation is likely to output erroneous object masks and features. Such errors are recovered by plausibility rules and prediction strategies to filter faulty object features, to monitor occlusion, and to merge divided objects. Analysis of displacements of the four minimum bounding box (MBB) sides allows the detection and correction of object occlusion and splitting.

6.1. Feature filtering

Due to various artifacts, errors are likely in feature extraction (cf. Section 1). These errors are recovered by ignoring features that are erroneous or occluded using error tolerance, error monitoring, and matching consistency principles. Examples are given in the following paragraphs.

Error tolerance For example, in small objects the difference of few percent in the number of pixels is significant while in large objects a small deviation may not be as significant. Therefore, thresholds of the feature votes used (Eq. 8-12) are adapted to the size of matched objects. This adaptation to the object size allows a better distinction at smaller sizes and a stronger matching at larger sizes. The adaptation of the thresholds to the object size is done as follows:

$$t_s = \begin{cases} 0.15 & : \quad A \leq A_{\min} \\ \text{linearly interpolated} & : \quad A_{\min} < A \leq A_{\max} \\ 0.5 & : \quad A > A_{\max} \end{cases} \tag{4}$$

Error monitoring For example, if the feature votes of two correspondence (M_i and M_j) of the same object O_p are equal, then this feature is excluded from the voting process. For example, shape irregularity $d_r = |r_i - r_j| < t_r$ with $r_i = \frac{r_p}{r_i}$ and $r_j = \frac{r_p}{r_j}$.

Matching consistency For example, objects are tracked once they enter the scene and also *during* occlusion; object correspondence is performed only if the estimated motion directions are consistent; if, after applying the correspondence voting scheme, two objects of $I(n-1)$ are matched with the same object in $I(n)$, the match with the oldest object (i.e, with the longer trajectory) is selected.

6.2. Monitoring erroneous object fusion and occlusion

Detection of fusion and occlusion Define

- $O_{p_1}, O_{p_2} \in I(n-1)$;
- $M_i : O_{p_1} \rightarrow O_i$ where O_i results from the occlusion of O_{p_1} and O_{p_2} in $I(n)$;
- $d_{p_{12}}$, the distance of the centroids of O_{p_1} and of O_{p_2};
- $w = (w_x, w_y)$, the current displacement of O_{p_1}, i.e., between $I(n-2)$ and $I(n-1)$.
- $d_{r_{\max}}$ ($d_{r_{\min}}$), the vertical displacement of the lower (upper) row and
- $d_{c_{\max}}$ ($d_{c_{\min}}$), the horizontal displacement of the right (left) column of O_{p_1}.

Object occlusion is declared if

$$\begin{array}{l} ((|w_y - d_{r_{\max}}| > t_1) \wedge (d_{r_{\max}} > 0) \wedge (d_{i_{12}} < t_2)) \vee \\ ((|w_y - d_{r_{\min}}| > t_1) \wedge (d_{r_{\min}} > 0) \wedge (d_{i_{12}} < t_2)) \vee \\ ((|w_x - d_{c_{\max}}| > t_1) \wedge (d_{c_{\max}} > 0) \wedge (d_{i_{12}} < t_2)) \vee \\ ((|w_x - d_{c_{\min}}| > t_1) \wedge (d_{c_{\min}} > 0) \wedge (d_{i_{12}} < t_2)) \end{array} \quad (5)$$

where t_1 and t_2 are thresholds. If occlusion is detected then both the occluding and the occluded objects are labeled for subsequent tracking. This labeling enables the system to continue tracking both objects in following images even if the occluded object is completely non-visible. Tracking non-visible objects is important since they might reappear. The labeling is further important to help detect occlusion even if the occlusion conditions in Eq. 5 are not met.

Correction of occlusion by object prediction If occlusion is detected, the occluded object O_i is split into two objects. This is done by predicting both object O_{p_2} and O_{p_1} onto $I(n)$ using the following displacement estimate:

$$\begin{array}{l} d_{p_1} = (\text{MED}(d^1_{x_c}, d^1_{x_p}, d^1_{u_x}), \text{MED}(d^1_{y_c}, d^1_{y_p}, d^1_{u_y})) \\ d_{p_2} = (\text{MED}(d^2_{x_c}, d^2_{x_p}, d^2_{u_x}), \text{MED}(d^2_{y_c}, d^2_{y_p}, d^2_{u_y})) \end{array} \quad (6)$$

where MED represents a 3-tap median filter, $d^1_{x_c}(d^1_{y_c})$, $d^1_{x_p}(d^1_{y_p})$,$d^1_{u_x}(d^1_{u_y})$ are the current, previous, and past-previous horizontal (vertical) displacement of O_{p_1}, and $d^2_{x_c}(d^2_{y_c})$, $d^2_{x_p}(d^2_{y_p})$,$d^2_{u_x}(d^2_{u_y})$ are the current, previous and past-previous horizontal (vertical) displacement of O_{p_2}. After splitting occluded and occluding objects, the lists of objects in $I(n)$ and $I(n-1)$ are updated, for example, by adding O_{p_2} to the list of objects in $I(n-1)$. If new objects are added to $I(n)$ or $I(n-1)$ matching is applied recursively for these objects (Fig. 1).

Two examples of object occlusion detection and correction are shown in Fig. 2. The scene shows two objects moving before they occlude. The change detection module provides one segment for both objects but the tracking module is able to correct the error and track the two objects also during occlusion. Note that in the original images of these examples, the objects appear very small and pixels are missing or misclassified due to non-accuracy of the object segmentation used. However, most pixels of the two objects are correctly classified and tracked.

Figure 2. Two examples of tracking two objects during occlusion.

6.3. Monitoring erroneous object splitting

Detection of splitting Assume $O_p \in I(n-1)$ is split in $I(n)$ into two objects O_{i_1} and O_{i_2}.Let $M_i : O_p \rightarrow O_{i_1}$, $d_{i_{12}}$ be the distance between the centroids of O_{i_1} and of O_{i_2}, and $w = (w_x, w_y)$ the current displacement of O_p between $I(n-2)$ and $I(n-1)$. Then object splitting is declared if

$$\begin{array}{l} (|w_y - d_{r_{\max}}| > t_1) \wedge d_{r_{\max}} < 0 \wedge d_{i_{12}} < t_2 \quad \vee \\ (|w_y - d_{r_{\min}}| > t_1) \wedge d_{r_{\min}} < 0 \wedge d_{i_{12}} < t_2 \quad \vee \\ (|w_x - d_{c_{\max}}| > t_1) \wedge d_{c_{\max}} < 0 \wedge d_{i_{12}} < t_2 \quad \vee \\ (|w_x - d_{c_{\min}}| > t_1) \wedge d_{c_{\min}} < 0 \wedge d_{i_{12}} < t_2 \end{array} \tag{7}$$

This means if 1) the difference between the current object (O_1) displacement and the displacement of *one* of the four sides of the MBB is larger than a threshold, 2) the displacement of that MBB side is inwards (i.e., towards the center of the object), and 3) there is an object O_2 close to O_1 then object splitting or separation close to the MBB-side with the large displacement is assumed, i.e., large inward displacement of an MBB-side. If splitting is detected, then the two object regions O_{i_1} and O_{i_2} are merged into one object O_i. After merging the features of O_i and the match $M_i : O_p \rightarrow O_i$ are updated (Fig. 1).

Correction of splitting by region merging Segmentation methods may divide an object into several regions. Regions can be merged either based on i) spatial homogeneity features such as texture or color, ii) temporal features such as motion, or iii) spatial relationships such as inclusion and size ratio (if a region is contained in another region and its size is significantly smaller, it maybe merged if the two objects show similar characteristics such as motion).

This paper develops a merging strategy that is based on spatial relationships, temporal coherence, and matching of objects as follows: assume 1) equation 7 is true, i.e., an object $O_p \in I(n-1)$ is split in $I(n)$ into two sub-regions O_{i1}, and O_{i2} and 2) the matching process matches O_p with O_{i1}. Then O_{i2} and O_{i1} are merged to be O_i if *all* the following conditions are met:

- Object voting gives $M_i : O_p \rightarrow O_i$ with a low vote of confidence ζ, i.e., $\zeta > t_{m_{\text{merge}}}$ with $t_{m_{\text{merge}}} < t_m$ (Eq. 1).
- If a split is found on one side of the MBB (based on Eq. 7), then all the displacements of the three other MBB sides of O_p should not change significantly when the two objects are merged.
- O_{i1} is spatially close to O_{i2} and O_{i2} to O_p, for example, in the case of down split, all the distances d, d_{nc}, d_{xc}, and d_{xr} are small.
- The size, hight, and width, of the merged object $O_i = O_{i1} + O_{i2}$ matches those of O_p. For example, $t_{\min} < \frac{A_p}{A_i} < t_{\max}$, with thresholds $t_{\min}, t_{\max}$.
- The motion direction of O_p does not significantly change if matched to O_i.

This merging strategy has proven to be powerful in various simulations. The good performance is due to the cooperation between the matching and merging processes. Each process supports the other based on rules that aim at limiting erroneous merging. The advantage of the proposed merging strategy compared to known merging techniques (cf.[23, 24]) is that it is based on temporal coherence throughout the tracking process.

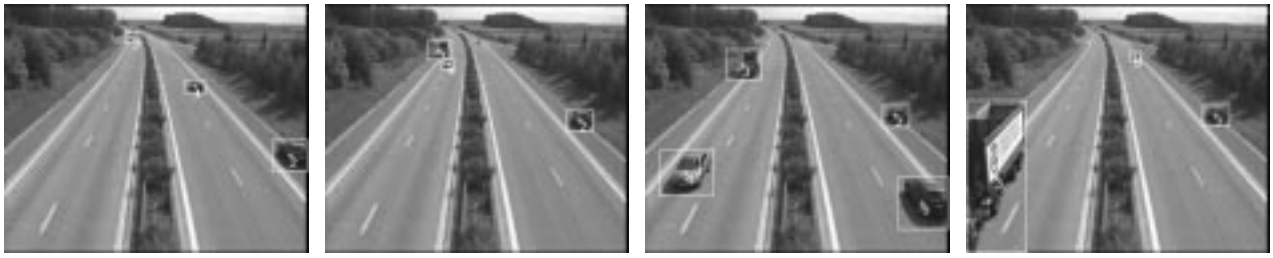

Figure 3. Tracking results of the sequence 'Highway'. To show the reliability of the tracking algorithm only one in every five images has been used. This shows, for example, that the proposed method can track objects that move fast.

7. RESULTS

Extensive experimentation on more than 10 indoor and outdoor video shots containing a total of 6371 images, including images with multi-object occlusion, noise, and coding artifacts have demonstrated the reliability and real-time response of the proposed technique. This reliability is due to the non-linear voting scheme and due to the use of plausibility rules for temporal stability and for detection of occlusion and segmentation errors.

The reliability of the proposed tracking method can be demonstrated when tracking objects in sequences with skipped images. As can be seen in Fig. 3, the objects are reliably tracked even when five images have been skipped.

An object trajectory is approximated by that of its centroid. To illustrate the temporal stability of the proposed algorithm, the estimated trajectory of each object is plotted as a function of the image number. Such a plot illustrates the reliability of both the motion estimation and tracking methods and allows the analysis and interpretation of the behavior of an object throughout the video shot. For example, the trajectories in Fig. 4 show that objects enter the scene at different times. Two objects (O_4 and O_2) are moving fast (note that the trajectory curve increases rapidly). In Fig. 5, the video analysis extracts three objects. Two objects enter the scene in the first image while the third object enters around the 70^{th} image. O_1 moves horizontally to the left and vertically down, O_2 moves horizontally right and vertically up, and O_5 moves fast to left. While the interpretation of objects going straight-forward motion (for example, not stopping or depositing something) is easy to follow and interpret, motion and behavior of persons that perform action is not easy to follow.

Fig. 5 shows a sample of tracking results. The proposed method is reliable in the case of occlusion, object scale variations, local illumination changes and noise (Fig. 3).

8. CONCLUSION

This paper proposes a method for tracking multiple moving objects reliably in the presence of shadows, noise, and occlusion. The proposed algorithm has been developed for content-based video application such as video surveillance and retrieval. The method is based on a non-linear voting system that solves the problem of occlusion. Objects are tracked once they enter the scene and also during occlusion. This is important for high-level video content extraction. Plausibility rules for consistency, error tolerance and monitoring are proposed for accurate tracking over long periods of time. Another important contribution of the proposed tracking method is the reliable region merging which improves significantly the performance of the whole algorithm.

The proposed algorithm is able to handle several objects simultaneously and to adapt to their occlusion or crossing. A confidence measure is maintained over time until the system is confident about the correct matching. Our tracking procedure is independent of how objects are segmented. No template or model matching is used but rather rules that are largely independent of object appearance are used. Finally, no constraints regarding object motion and camera position are imposed. Furtehr research is planed to enhance the performance of the algorithm in case of shadow and object occlusion.

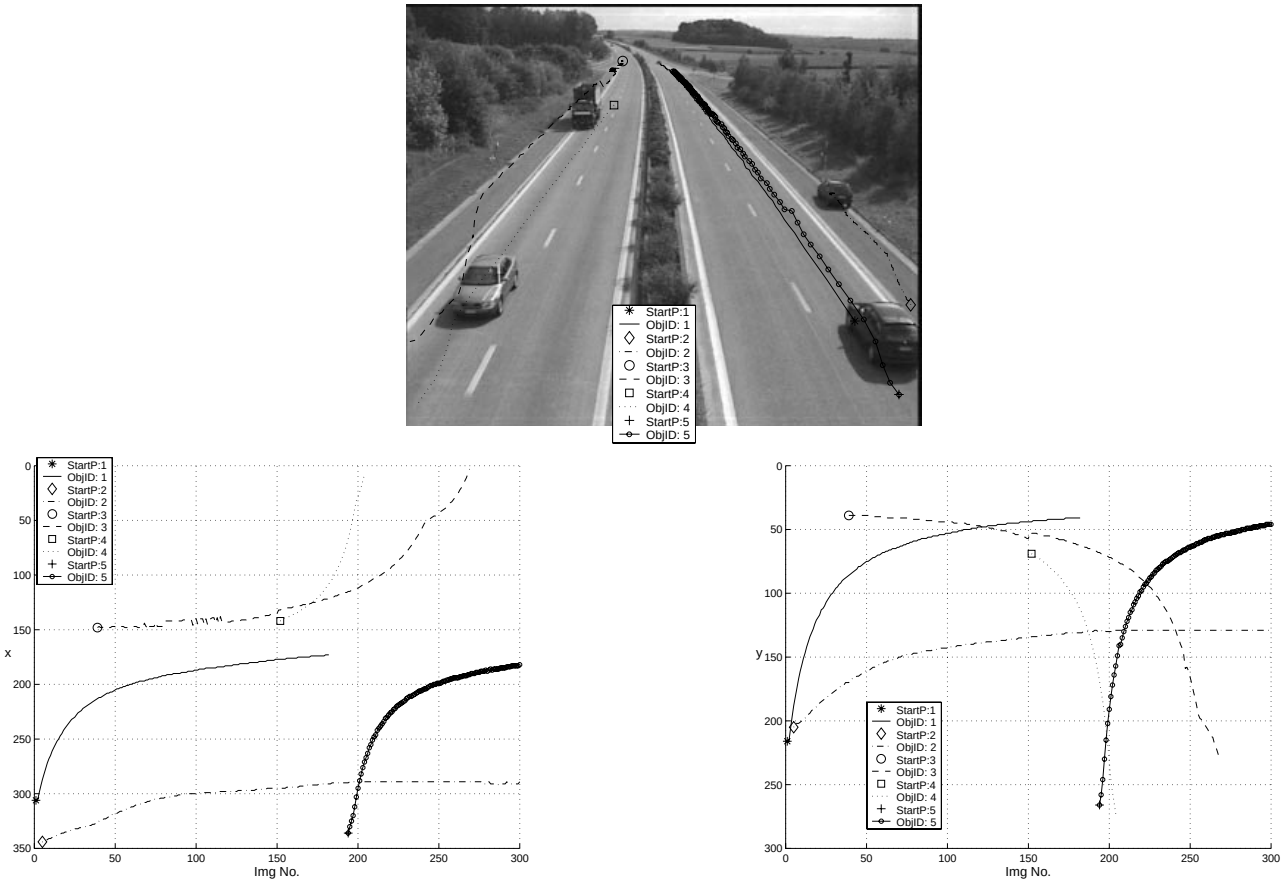

Figure 4. The trajectories of the objects in the sequence 'Highway' where 'StartP' represents the starting point of a trajectory. The upper figure gives the trajectory of the objects in the image plane while the two other figures give the trajectories for vertical and horizontal directions separately. This allows an interpretation of the object motion behavior throughout the sequence. The system tracks all objects reliably.

Figure 5. Tracking results of the 'Survey' sequence. Each object is marked by an ID-number and enclosed in its minimum bounding box. Despite the multi-object occlusion, light changes, and reflections, the algorithm stays stable. Note how the method recovered properly after the static traffic sign.

APPENDIX A. OBJECT VOTING

To compute the similarity variable s and the dissimilarity variable d (see Section 5.1) between two objects O_i and O_j, the following three feature (shape, size, and motion) votes are applied. $t_z < 1$ and $t_s < 1$ are functions of the image and object sizes (see Eq. 4):

1. Size vote: Let $r_{a_i} = \begin{cases} A_p/A_i & : \quad A_p \leq A_i \\ A_i/A_p & : \quad A_p > A_i \end{cases}$, $r_{h_i} = \begin{cases} H_p/H_i & : \quad H_p \leq H_i \\ H_i/H_p & : \quad H_p > H_i \end{cases}$, and $r_{w_i} = \begin{cases} W_p/W_i & : \quad W_p \leq W_i \\ W_i/W_p & : \quad W_p > W_i \end{cases}$, where A_i, H_i, and W_i are the area, height, and width of object O_i. Then

$$\begin{aligned} s_{++} &: \quad r_{a_i} > t_z \quad \vee \quad r_{h_i} > t_z \quad \vee \quad r_{w_i} > t_z \\ d_{++} &: \quad r_{a_i} \leq t_z \quad \vee \quad r_{h_i} \leq t_z \quad \vee \quad r_{w_i} \leq t_z \end{aligned} \tag{8}$$

2. Shape vote: Let $e_p(e_i)$, $c_p(c_i)$, $r_p(r_i)$ be the extent ratio, compactness and irregularity of the shape of $O_p(O_i)$, $d_{e_i} = |e_p - e_i|$, $d_{c_i} = |c_p - c_i|$, and $d_{r_i} = |r_p - r_i|$. Then

$$\begin{aligned} s_{++} &: \quad d_{e_i} \leq t_s \quad \vee \quad d_{c_i} \leq t_s \qquad \vee d_{r_i} \leq t_s \\ d_{++} &: \quad d_{e_i} > t_s \quad \vee \quad d_{c_i} > t_s \qquad \vee d_{r_i} > t_s \end{aligned} \tag{9}$$

3. Motion vote: Let the previous horizontal and vertical direction of the object be $\delta_p = (\delta_{x_p}, \delta_{y_p})$ and its current direction be $\delta_c = (\delta_{x_c}, \delta_{y_c})$. Then

$$\begin{aligned} s_{++} &: \quad \delta_{x_c} = \delta_{x_p} \quad \vee \quad \delta_{y_c} = \delta_{y_p} \\ d_{++} &: \quad \delta_{x_c} \neq \delta_{x_p} \quad \vee \quad \delta_{y_c} \neq \delta_{y_p} \end{aligned} \tag{10}$$

APPENDIX B. CORRESPONDENCE VOTING

To compute the similarity variable s_i and the dissimilarity variable s_j (see Section 5.2) between two correspondences M_i and M_j, the following five feature (distance, confidence, size, shape, and motion) votes are applied. $t_z^k < 1$, $t_s^k < 1$, and $t_d^k > 1$ are function of the image and object sizes (see equation 4). In the following, the index k denotes a vote for a correspondence M_k.

1. Distance vote: Let d_i be the distance between O_p and O_i and d_j the distance between O_p and O_j. Let $d_d^k = |d_i - d_j|$. Then

$$\begin{aligned} s_{i++} &: \quad d_d^k > t_d^k \quad \wedge \quad d_i < d_j \\ s_{j++} &: \quad d_d^k > t_d^k \quad \wedge \quad d_i > d_j \end{aligned} \tag{11}$$

The aim of the condition $d_d^k > t_d^k$ is to ensure that only if the two features differ significantly can the vote be applied; if the features do not differ significantly then neither s_i nor s_j are increased.

2. Confidence vote: Let $d_\zeta = |\zeta_i - \zeta_j|$. Then

$$\begin{aligned} s_{i++} &: \quad (d_\zeta > t_\zeta) \quad \wedge \quad (\zeta_i > \zeta_j) \\ s_{j++} &: \quad (d_\zeta > t_\zeta) \quad \wedge \quad (\zeta_i < \zeta_j) \end{aligned} \tag{12}$$

The condition $d_\zeta > t_\zeta$ ensures that only if the two features differ significantly can the vote be applied.

3. Size vote: Let $d_a^k = |r_{a_i} - r_{a_j}|$, $d_h^k = |r_{h_i} - r_{h_j}|$, and $d_w^k = |r_{w_i} - r_{w_j}|$. Then

$$\begin{aligned} s_{i++} &: \quad (d_a^k > t_z^k \quad \wedge \quad r_{a_i} < r_{a_j}) \quad \vee (d_h^k > t_z^k \quad \wedge \quad r_{h_i} < r_{h_j}) \quad \vee (d_w^k > t_z^k \quad \wedge \quad r_{w_i} < r_{w_j}) \\ s_{j++} &: \quad (d_a^k > t_z^k \quad \wedge \quad r_{a_i} > r_{a_j}) \quad \vee (d_h^k > t_z^k \quad \wedge \quad r_{h_i} > r_{h_j}) \quad \vee (d_w^k > t_z^k \quad \wedge \quad r_{w_i} > r_{w_j}) \end{aligned} \tag{13}$$

If the features do not differ significantly then neither s_i nor s_j are increased.

4. Shape vote: Let $d_e^k = |d_{e_i} - d_{e_j}|$, $d_c^k = |d_{c_i} - d_{c_j}|$, and $d_r^k = |d_{r_i} - d_{r_j}|$. Then

$$\begin{array}{lcl} s_{i++} & : & (d_e^k > t_s^k \quad \wedge \quad r_{e_i} < r_{e_j}) \quad \vee (d_c^k > t_s^k \quad \wedge \quad r_{c_i} < r_{c_j}) \quad \vee (d_r^k > t_s^k \quad \wedge \quad r_{r_i} < r_{r_j}) \\ s_{j++} & : & (d_e^k > t_s^k \quad \wedge \quad r_{e_i} > r_{e_j}) \quad \vee (d_c^k > t_s^k \quad \wedge \quad r_{c_i} > r_{c_j}) \quad \vee (d_r^k > t_s^k \quad \wedge \quad r_{r_i} > r_{r_j}) \end{array} \tag{14}$$

If the features do not differ significantly then neither s_i nor s_j are increased.

5. Motion vote:

- Direction vote: let $\delta_c = (\delta_{x_c}, \delta_{y_c})$, $\delta_p = (\delta_{x_p}, \delta_{y_p})$, $\delta_u = (\delta_{x_u}, \delta_{y_u})$ be the current, previous, and past-previous motion direction of O_p. Let $\delta_i = (\delta_{x_i}, \delta_{y_i})$ be the motion direction of O_p if it is matched to O_i and $\delta_j = (\delta_{x_j}, \delta_{y_j})$ if matched to O_j.

$$\begin{array}{ll} s_{i++}: & (\delta_{x_i} = \delta_{x_c} \wedge \delta_{x_i} = \delta_{x_p} \wedge \delta_{x_i} = \delta_{x_u}) \quad \vee (\delta_{y_i} = \delta_{y_c} \wedge \delta_{y_i} = \delta_{y_p} \wedge \delta_{y_i} = \delta_{y_u}) \\ s_{j++}: & (\delta_{x_j} = \delta_{x_c} \wedge \delta_{x_j} = \delta_{x_p} \wedge \delta_{x_j} = \delta_{x_u}) \quad \vee (\delta_{y_j} = \delta_{y_c} \wedge \delta_{y_j} = \delta_{y_p} \wedge \delta_{y_j} = \delta_{y_u}) \end{array} \tag{15}$$

- Displacement vote: let d_{m_i} (d_{m_j}) be the displacement of O_p relative to O_i (O_j) and $d_m^k = |d_{m_i} - d_{m_j}|$. Then

$$\begin{array}{lcll} s_{i++} & : & (d_m^k > t_m^k) & \wedge \quad (d_{m_i} < d_{m_j}) \\ s_{j++} & : & (d_m^k > t_m^k) & \wedge \quad (d_{m_i} > d_{m_j}) \end{array} \tag{16}$$

  Here $d_m^k > t_m^k$ means that the displacements have to differ significantly to be considered for voting. t_m^k is adapted to detected segmentation error. For example, in the case of occlusion, it is increased and is a function of the image and object size. The motion magnitude vote can contribute more than one vote to the matching process if $s_i = s_j$ and the difference d_m^k is large then s_i or s_j are increased by 1,2, or 3 as follows:

$$\begin{array}{ll} s_{i+1}: (d_m^k < t_{m_{\text{min}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} < d_{m_j}) \\ s_{i+2}: (t_{m_{\text{min}}}^k < d_m^k < t_{m_{\text{max}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} < d_{m_j}) \\ s_{i+3}: (d_m^k > t_{m_{\text{max}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} < d_{m_j}) \\ & \\ s_{j+1}: (d_m^k < t_{m_{\text{min}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} > d_{m_j}) \\ s_{j+2}: (t_{m_{\text{min}}}^k < d_m^k < t_{m_{\text{max}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} > d_{m_j}) \\ s_{j+3}: (d_m^k > t_{m_{\text{max}}}^k) & \wedge (d_m^k > t_m^k) \wedge (d_{m_i} > d_{m_j}) \end{array} \tag{17}$$

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. A. Amer, E. Dubois, and A. Mitiche, "A real-time system for high-level representation of video shots," in *Proc. SPIE Int. Conf. on Image and Video Communications and Processing*, (Santa Clara, California, USA), Jan. 2003. to appear.
2. J. Boyd, J. Meloche, and Y. Vardi, "Statistical tracking in video traffic surveillance," in *Proc. IEEE Int. Conf. Computer Vision*, **1**, pp. 163–168, (Corfu, Greece), Sept. 1999.
3. 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.

4. R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa, "A system for video surveillance and monitoring," Tech. Rep. CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2000.
5. E. Stringa and C. Regazzoni, "Content-based retrieval and real time detection from video sequences acquired by surveillance systems," in *Proc. IEEE Int. Conf. Image Processing*, pp. 138–142, (Chicago, IL), Oct. 1998.
6. "Special section on video surveillance." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, Aug. 2000.
7. F. Dufaux and F. Moscheni, "Segmentation-based motion estimation for second generation video coding techniques," in *Video coding: Second generation approach*, L. Torres and M. Kunt, eds., pp. 219–263, Kluwer Academic Publishers, 1996.
8. A. Amer, E. Dubois, and A. Mitiche, "Context-independent real-time event recognition: Application to key-image extraction," in *Proc. IEEE Int. Conf. Pattern Recognition*, pp. 000–000, (Québec, Canada), Aug. 2002.
9. G. Legters and T. Young, "A mathematical model for computer image tracking," *IEEE Trans. Pattern Anal. Machine Intell.* **4**, pp. 583–594, Nov. 1982.
10. K. Daniilidis, C. Krauss, M. Hansen, and G. Sommer, "Real time tracking of moving objects with an active camera," *J. Real-Time Imaging* **4**, pp. 3–20, February 1998.
11. M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. European Conf. Computer Vision*, **A** , pp. 343–356, 1996.
12. B. Bascle, P. Bouthemy, R. Deriche, and F. Meyer, "Tracking complex primitives in an image sequence," in *Proc. IEEE Int. Conf. Pattern Recognition*, pp. 426–431, (Jerusalem), Oct. 1994.
13. S. Gil, R. Milanese, and T. Pun, "Feature selection for object tracking in traffic scenes," in *Proc. SPIE Int. Symposium on Smart Highways*, **2344**, pp. 253–266, (Boston, MA), Oct. 1994.
14. S. Khan and M. Shah, "Tracking people in presence of occlusion," in *Proc. Asian Conf. on Computer Vision*, pp. 1132–1137, (Taipei, Taiwan), Jan. 2000.
15. S. Dockstader and A. Tekalp, "On the tracking of articulated and occluded video object motion," *J. Real-Time Imaging* **7**, pp. 415–432, Oct. 2001.
16. A. Crétual, F. Chaumette, and P. Bouthemy, "Complex object tracking by visual servoing based on 2-D image motion," in *Proc. IEEE Int. Conf. Pattern Recognition*, **2**, pp. 1251–1254, (Brisbane, IL), Aug. 1998.
17. A. Azarbayejani, C. Wren, and A. Pentland, "Real-time 3-D tracking of the human body," in *Proc. IMAGE'COM*, pp. 19–24, (Bordeaux, France), May 1996. M.I.T. TR No. 374.
18. 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.
19. 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.
20. 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.
21. A. Amer, "Memory-based spatio-temporal real-time object segmentation," in *Proc. SPIE Int. Conf. on Real-Time Imaging*, (Santa Clara, California, USA), Jan. 2003. to appear.
22. A. Rosenfeld and C. Kak, *Digital Picture Processing*, vol. 2, Academic Press, INC., Orlando, 1982.
23. L. Garrido, P. Salembier, and D. Garcia, "Extensive operators in partition lattices for image sequence analysis," *Signal Process.* **66**, pp. 157–180, 1998.
24. 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.