Real-Time Motion Estimation by Object-Matching for High-Level Video Representation

Aishy Amer Amar Mitiche Eric Dubois

INRS-Télécommunications Université d'Ottawa
Montréal, H5A 1C6 Canada Ottawa, K1N 6N5 Canada
E-mail: amer@inrs-telecom.uquebec.ca

Abstract

Motion estimation plays a key role in many video applications, such as frame-rate video conversion, video retrieval, video surveillance, and video compression. The key issue in these applications is to define appropriate representations that can efficiently support motion estimation with the required accuracy. In this paper, a low-complexity object motion estimation technique is proposed that is designed to fit the needs of high-level video representation such as in video surveillance or retrieval applications. In these applications, a representation of object motion in a way meaningful for high-level interpretation, such as event detection and classification, foregoes precision of estimation. The proposed method relies on the estimation of the displacements of the minimum bounding box (MBB) sides of an object. Two motion estimation steps are proposed: initial coarse estimation to find a single displacement for an object using the four sides of the MBB between two successive images and detection of non-translational motion and its estimation. The result is the detection of the type of object motion and the subsequent estimation of one or more motion values per object depending on the detected motion type. Special consideration is given to object motion in interlaced video and at image margin. Various simulations show that the proposed method provides a response in real-time and gives good estimates to use for object tracking, event detection, and high-level video representation. The proposed object motion estimation method is insensitive to inaccurate segmentation in these applications.

1 Introduction

Objects can be classified into three major categories: rigid, articulated, and non-rigid [7]. The motion of a rigid object is a composition of a translation and a rotation. An articulated object consists of rigid parts linked by joints. Most video applications, such as entertainment, surveillance, or retrieval, assume rigid objects.

An image acquisition system projects a 3-D world scene onto a 2-D image plane. When an object moves its projection is animated by a 2-D motion, to be estimated from the space-time image variation. These variations can be divided into global and local. Global variations can be a result of camera motion or global illumination change. Local variations can be due to object motion, local illumination change and noise. Motion estimation techniques estimate apparent motion which is due to true motion or to various artifacts, such as noise and illumination change. The goal of a motion estimation technique is to assign a motion vector (displacement or velocity) to each pixel in an image. Motion estimation relies on hypotheses about the nature of the image or object motion, and is often tailored to applications needs. Difficulties in motion estimation arise from unwanted camera motion, occlusion, noise, lack of image texture, and illumination changes. Motion estimation is an ill-posed problem which requires regularization. A problem is ill-posed if no unique solution exists or the solution does not continuously depend on the input data.

The choice of a motion estimation approach strongly depends on the application and on the nature of the processes that will interpret the estimated motion. A key issue when designing a motion estimation technique is its degree of efficiency with enough accuracy to serve the purpose of intended application. For instance, in video surveillance and retrieval, a tradeoff is required between computation cost and quality of results. This paper proposes a real-time method to estimate 2-D object motion from a video using segmented object. The method aims at representing object motion in a way meaningful for high-level applications, e.g., event classification, that foregoes precision of estimation.

The paper is organized into five additional sections. Section 2 discusses related work, Section 3 discusses models for object-based motion estimation, Section 4 describes a real-time motion estimation method based on extracted object, Section 5 presents experimental results, and Section 6 concludes the paper.

2 Review of methods and motivation

Motion estimation methods can be classified into two broad categories: gradient-based and matching methods [10]. Motion estimation by matching is most frequent used in video applications. Most representative is block-matching which can be implemented in hardware for real-time execution [4, 5]. The motion field is assumed to be constant over rectangular blocks and represented by a single motion vector in each block. Several refinements of this basic idea have been proposed [7]. Advantages of block-matching algorithms are: 1) easy implementation, 2) better quality of the resulting motion vector fields compared to other methods such as phase correlation and gradient methods in the presence of large motion, and 3) they can be implemented by regular VLSI architectures. An additional important advantage of block-matching is that it does not break down totally. Block matching has, however, some drawbacks. This is particularly true at object boundaries where an incorrect model is assumed and result in erroneous motion vectors leading to discontinuity in the motion vector fields causing *ripped boundaries* artifacts. Another drawback is that the resulting motion vectors inside objects or object regions with a single motion are not homogeneous producing *ripped region* artifacts. Additionally, using block-based algorithms result in block patterns in the motion vector field causing *block patterns* or *blocking* artifacts. These patterns often result in block motion artifacts in subsequently processed images. The human visual system is very sensitive to such artifacts (especially abrupt changes).

Block-based and pixel-based motion estimation methods have been widely used in the field of coding and image interpolation. The focus in these applications is on accurate motion and less on meaningful representation of object motion. In high-level representation such as in video surveillance, the focus is on reliable estimation without high precision, but flexible and stable throughout an image sequence. Content-based video representation and processing call for motion estimation based on objects. In an object-based motion estimation algorithm, motion vectors are estimated using information about the shape or structure of segmented objects. This causes the motion vectors to be consistent within the objects and at object boundaries. In addition, since the number of objects is significantly less than the number of blocks in an image, object-based motion estimation has lower complexity.

Various object-based motion estimation methods have been proposed that, in general, require large amounts of computations [12, 8, 6, 13]. This complexity is mainly due to image segmentation. Also, although they generally give good motion estimates, they can fail to interpret object motion correctly or can simply break down. This is due to their dependence on accurate segmentation. Several methods use region growing for segmentation, or try to minimize a global energy function. Furthermore, these methods include several levels of refinement. In this paper, a low-complexity object motion estimation technique is introduced that is aimed at meaningful and real-time object representation.

3 Object motion model

Two broad categories of motion models are defined: non-parametric and parametric. Non-parametric models are based on a dense local motion field where one motion vector is estimated for each pixel of the image. Parametric models describe the motion of a region in the image by a set of parameters. The motion of rigid objects, for example, can be described by a parameter motion model. Various simplifications of parametric motion models exist [11, 8, 10]. Models have different complexity and accuracy. As a compromise between complexity and accuracy, 2-D affine motion models or 'simplified linear' models are used [11, 8].

Assuming a static camera or a camera-motion compensated video, local object motion can be described adequately as the composition of translation, rotation, and scaling. Changes in object scale occur when the object moves towards or away from the camera. This paper uses the so-called 'simplified linear' models [11] to describe objects motion. Let (p_x, p_y) and (q_x, q_y) be the initial, respectively the final, position of a point p of an object undergoing motion.

Translation Translation of p by (d_x, d_y) is given by

$$\begin{aligned} q_x &= p_x + d_x \\ q_y &= p_y + d_y \end{aligned} \tag{1}$$

Scaling The scale change transformation of p is defined by

$$\begin{aligned} q_x &= s \cdot (p_x - c_x) + c_x \\ q_y &= s \cdot (p_y - c_y) + c_y \end{aligned} \tag{2}$$

where s is the scaling factor and (c_x, c_y) is the center of scaling.

Rotation The rotational transformation of p is defined by

$$\begin{aligned} q_x &= c_x + (p_x - c_x)\cos\phi - (p_y - c_y)\sin\phi \\ q_y &= c_y + (p_x - c_x)\sin\phi + (p_y - c_y)\cos\phi \end{aligned} \tag{3}$$

where (c_x, c_y) is the center of rotation and ϕ the rotation angle.

Composition If an object O_i is scaled, rotated, and displaced then the final position of $p \in O_i$ is defined by (assume a small-angle rotation which gives $\sin\phi \simeq \phi$ and $\cos\phi \simeq 1$)

$$\begin{aligned} q_x &= p_x + d_x + s \cdot (p_x - c_x) - \phi \cdot (p_y - c_y) \\ q_y &= p_y + d_y + s \cdot (p_y - c_y) + \phi \cdot (p_x - c_x) \end{aligned} \tag{4}$$

4 Object-based motion estimation

The proposed method estimates object motion based on the displacements of the object MBB. MBB-based motion estimation is not a new concept. Usually, MBB-based methods use the displacement of the centroid of the MBB. This is sensitive to noise and other image artifacts such as occlusion. Besides, most MBB motion estimators assume translational motion when motion type can be important information as in object-based video retrieval, for instance. The contribution of this paper is in the detection of the type of object motion: translation, scaling, composition, and the subsequent estimation of one or more motion values per object depending on the detected motion type. Special consideration is given to object motion in interlaced video and at image margin. Also, proposed analysis of displacements of the MBB-sides allows the estimation of complex motion as when objects move towards or away from the camera.

The proposed non-parametric method is based on four steps (Fig. 1): object segmentation, object matching, MBB-based displacement estimation, and motion analysis and update. Object segmentation and object matching are shortly discussed in Sec. 4.1 and are studied with more details in [1]. Note that the proposed motion estimation does not require accurate segmentation. In the third step (Section 4.2), an initial object motion is estimated by considering the displacements of the sides of the MBBs of two corresponding objects (Fig. 2) accounting for possible segmentation inaccuracies due to occlusion and splitting of object regions. In the fourth step (Section 4.3), the type of the object motion is determined. If the motion is a translation, a single motion vector is estimated. Otherwise different motion vectors are assigned to the different object regions. The proposed motion estimation scheme assumes that the shape of moving objects does not change drastically between successive images and that the displacements are within a predefined range. These two assumptions are realistic for most video applications and motion of real objects.

4.1 Object segmentation and matching

Segmentation Segmentation is realized in four steps [1, 2]: binarization of the input gray-level images, morphological edge detection, contour analysis, and object labeling. The critical task are the binarization and the contour selection which must stay reliable throughout the video. Assuming a static camera or a camera-motion compensated input video, motion detection-based binarization is used where the algorithm memorizes previously detected motion to adapt the current motion detection [1]. 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 [1]. 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, e.g., [9]).

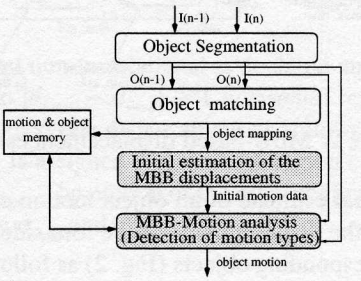


Figure 1: The proposed motion estimation method.

Matching Object matching is achieved by matching single object features and then combining the matches based on a voting scheme. Such feature-based solutions need to answer some questions concerning feature selection, monitoring, correction, integration, and filtering [1]. Feature selection schemes define good features to match. Feature monitoring aims at detecting errors and at adapting the matching process to these errors. Feature correction aims at compensating for segmentation errors during matching, especially during occlusion. Feature integration defines ways to efficiently and effectively combine features. Feature filtering is concerned with ways to monitor and eventually filter noisy features during matching over time.

Matching is activated once an object enters the scene. An entering object is immediately detected by the segmentation module. The segmentation and motion estimation modules extract the relevant features for the matching module. While matching objects, the segmentation module keeps looking for new objects entering the scene. In the case of multiple object occlusion, an occlusion detection module first detects occluded and occluding objects, and then continues to track both types of objects even if objects are completely invisible, i.e., their area is zero since no pixel can be assigned to them. This is because objects may reappear. No prior knowledge is assumed and plausibility rules for consistency, error allowance and monitoring are used for accurate matching [1].

4.2 Initial estimation

Let $I(n)$ be the observed image at time instant n, defined on an $X \times Y$ lattice where the starting pixel, $I(1,1,n)$, is at the upper-left corner of the lattice. If the motion is estimated

forward, between $I(n-1)$ and $I(n)$, then the direction of object motion is defined as follows: horizontal motion to the left is negative and positive to the right; vertical motion down is positive and negative up.

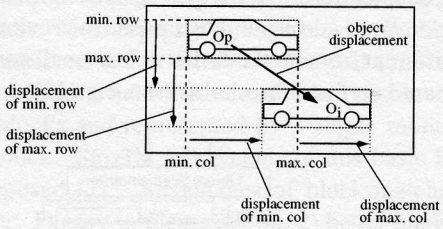


Figure 2: MBB-based displacement estimation.

The initial estimate of an object motion comes from the analysis of the displacements of the four sides of the MBBs of two corresponding objects (Fig. 2) as follows. Let

- $M_i : O_p \rightarrow O_i$ a function that assigns to an object O_p at time $n-1$ an object O_i at time n [1].
- $w = (w_x, w_y)$ the current displacement of O_i, between $I(n-2)$ and $I(n-1)$.
- $(r_{\min_p}, c_{\min_p})$, $(r_{\max_p}, c_{\min_p})$ $(r_{\min_p}, c_{\max_p})$, and $(r_{\max_p}, c_{\max_p})$ the four corners, upper left, lower left, upper right, and lower right, of the MBB of O_p (cf. Fig. 2).
- $r_{\min_p}$ and $r_{\max_p}$ the upper and lower row of O_p.
- $c_{\min_p}$ and $c_{\max_p}$ the left and right column of O_p.
- $r_{\min_i}$ and $r_{\max_i}$ the upper and lower row of O_i. If upper occlusion or splitting is detected then $r_{\min_i} = r_{\min_p} + w_x$. If lower occlusion or splitting is detected then $r_{\max_i} = r_{\max_p} + w_x$.
- $c_{\min_i}$ and $c_{\max_i}$ the left and right column of O_i. If left occlusion or splitting is detected then $c_{\min_i} = c_{\min_p} + w_y$. If right occlusion or splitting is detected then $c_{\max_i} = c_{\max_p} + w_y$.
- $d_{r_{\min}} = r_{\min_i} - r_{\min_p}$ the vertical displacement of the point $(r_{\min_p}, c_{\min_p})$.
- $d_{c_{\min}} = c_{\min_i} - c_{\min_p}$ the horizontal displacement of the point $(r_{\min_p}, c_{\min_p})$.
- $d_{r_{\max}} = r_{\max_i} - r_{\max_p}$ the vertical displacement of the point $(r_{\max_p}, c_{\max_p})$.
- $d_{c_{\max}} = c_{\max_i} - c_{\max_p}$ the horizontal displacement of the point $(r_{\max_p}, c_{\max_p})$.
- $d_r = d_{r_{\max}} - d_{r_{\min}}$ the difference of the vertical displacements.
- $d_c = d_{c_{\max}} - d_{c_{\min}}$ the difference of the horizontal disp.

The initial displacement, $w_i^1 = (w_{x_i}^1, w_{y_i}^1)$, of an object is the mean of the displacements of the horizontal and vertical MBB-sides (see the first part of Eqs. 5 and 6). In case of segmentation errors, the displacements of parallel sides can deviate significantly, i.e, $|d_c| > t_d$ or $|d_r| > t_d$. So the method detects these deviations and corrects the estimate based on previous estimates of (w_x, w_y). This is given in the second and third part of Eqs. 5 and 6.

$$w_{x_i}^1 = \begin{cases} \frac{(d_{c_{\max}} + d_{c_{\min}})}{2} : & |d_c| \leq t_d \\ \frac{(d_{c_{\min}} + w_x)}{2} : & \begin{array}{l}(|d_c| > t_d) \quad \wedge \\ [((d_{c_{\max}} d_{c_{\min}} > 0) \quad \wedge \\ (d_{c_{\max}} > d_{c_{\min}})) \quad \vee \\ ((d_{c_{\max}} d_{c_{\min}} < 0) \quad \wedge \\ (d_{c_{\max}} w_x < 0))] \end{array} \\ \frac{(d_{c_{\max}} + w_x)}{2} : & \begin{array}{l}(|d_c| > t_d) \quad \wedge \\ [((d_{c_{\max}} d_{c_{\min}} > 0) \quad \wedge \\ (d_{c_{\max}} \leq d_{c_{\min}})) \quad \vee \\ ((d_{c_{\max}} d_{c_{\min}} < 0) \quad \wedge \\ (d_{c_{\max}} w_x > 0))] \end{array} \end{cases} \tag{5}$$

$$w_{y_i}^1 = \begin{cases} \frac{(d_{r_{\max}} + d_{r_{\min}})}{2} : & |d_r| \leq t_d \\ \frac{(d_{r_{\min}} + w_y)}{2} : & \begin{array}{l}(|d_r| > t_d) \quad \wedge \\ [((d_{r_{\max}} d_{r_{\min}} > 0) \quad \wedge \\ (d_{r_{\max}} > d_{r_{\min}})) \quad \vee \\ ((d_{r_{\max}} d_{r_{\min}} < 0) \quad \wedge \\ (d_{r_{\max}} w_y < 0))] \end{array} \\ \frac{(d_{r_{\max}} + w_y)}{2} : & \begin{array}{l}(|d_r| > t_d) \quad \wedge \\ [((d_{r_{\max}} d_{r_{\min}} > 0) \quad \wedge \\ (d_{r_{\max}} \leq d_{r_{\min}})) \quad \vee \\ ((d_{r_{\max}} d_{r_{\min}} < 0) \quad \wedge \\ (d_{r_{\max}} w_y > 0))] \end{array} \end{cases} \tag{6}$$

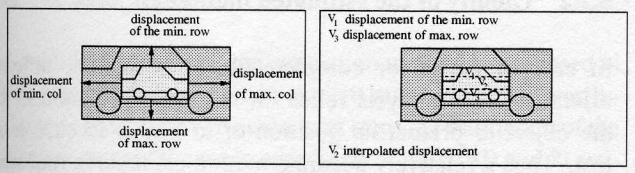
This estimated displacement may deviate from the correct value due to inaccurate shape estimation. To stabilize the estimation throughout the image sequence, the first initial estimate $w_i^1 = (w_{x_i}^1, w_{y_i}^1)$ is compared to the current estimate $w = (w_x, w_y)$. If they deviate significantly, i.e., the difference $|w_x - w_{x_i}^1| > t_m$ or $|w_y - w_{y_i}^1| > t_m$ for a threshold t_m, acceleration is assumed and the estimate w_i is adapted to the current estimate as given in Eq. 7, where a represents the maximal allowable acceleration. This way, the estimated displacement is adapted to the previous displacement to provide stability to inaccuracies in the estimation of the object shape by the object segmentation module.

$$w_{x_i} = \begin{cases} w_{x_i}^1 & : \quad |w_{x_i}^1 - w_x| \leq t_m \\ w_x + a & : \quad |w_{x_i}^1 - w_x| > t_m \quad \wedge \quad w_{x_i}^1 > w_x \\ w_x - a & : \quad |w_{x_i}^1 - w_x| > t_m \quad \wedge \quad w_{x_i}^1 < w_x \end{cases}$$

$$w_{y_i} = \begin{cases} w_{y_i}^1 & : \quad |w_{y_i}^1 - w_y| \leq t_m \\ w_y + a & : \quad |w_{y_i}^1 - w_y| > t_m \quad \wedge \quad w_{y_i}^1 > w_y \\ w_y - a & : \quad |w_{y_i}^1 - w_y| > t_m \quad \wedge \quad w_{y_i}^1 < w_y \end{cases} \tag{7}$$

4.3 Motion analysis and update

Often, objects correspond to a large area of an image and a translational model for object matching is not appropri-



(a) Detection of scale change. (b) Vertical scaling estimation.

Figure 3: Scaling: symmetrical, (nearly) identical displacements of all MBB-sides.

ate; a more complex motion model must be introduced. To achieve this, the motion of the sides of the MBB is analyzed and motion types are detected based on plausibility rules. This analysis detects four states of object motion changes: translation, scaling, composition, and acceleration. If non-translational motion is estimated, an object is divided into several partitions that are assigned different motion vectors. The number of regions depends on the magnitude of the estimated non-translation. Usually motion in objects does not contain fine details and motion vectors are spatially consistent so that large object regions have identical motion vectors. Therefore, the number of regions need not to be high.

Detection of translation This paper assumes translational object motion if the displacements of the horizontal and vertical sides of the object MBB are nearly identical, i.e.,

$$Translation \quad : \quad \begin{array}{l} |d_r| < t_d \quad \wedge \\ |d_c| < t_d \end{array} \tag{8}$$

In this case one motion vector (Eq. 7) is assigned to the whole object.

Detection of scaling This paper assumes the scaling center as the centroid of the object and assumes object scaling if the displacements of the parallel sides of the MBB are symmetrical and nearly identical. This means

$$Scaling \quad : \quad \begin{array}{llll} ((|d_r| < t_s) & \wedge & (d_{r_{\min}} \cdot d_{r_{\max}}) < 0) & \wedge \\ ((|d_c| < t_s) & \wedge & (d_{c_{\min}} \cdot d_{c_{\max}}) > 0) & \end{array} \tag{9}$$

with a small threshold t_s. For example, if one side is displaced to the right by three pixels, the parallel side is displaced by three pixels to the left (Fig. 3(a)). If scale change is detected the object is divided into sub-regions where the number of regions depends on $|d_r|$. Each region is then assigned one displacement as follows: the region closest to $r_{\max}$ is assigned $d_{r_{\max}}$ and the region closest to $r_{\min}$ is assigned $d_{r_{\min}}$. For in between regions motion is interpolated by increasing or decreasing $d_{r_{\min}}$ and $d_{r_{\max}}$ (Fig. 3)

Detection of general motion In case of composition of motion types, three types of motion are considered: translational, non-translational, and acceleration: If

$$|w_{y_i} - d_{r_{\min}}| > a \quad \vee \quad |w_{y_i} - d_{r_{\max}}| > a \tag{10}$$

where a is the maximal possible acceleration, then it is a *vertical* non-translation and the object is divided into $|d_{r_{\max}} - d_{r_{\min}}| + 1$ *vertical* regions. Each region is assigned one displacement as follows: the region closest to $r_{\max}$ is assigned $d_{r_{\max}}$ and the region closest to $r_{\min}$ is assigned $d_{r_{\min}}$. The motion of in-between regions is interpolated by increasing or decreasing $d_{r_{\min}}$ and $d_{r_{\max}}$ (Fig. 4). If

$$|w_{x_i} - d_{c_{\min}}| > a \quad \vee \quad |w_{x_i} - d_{c_{\max}}| > a \tag{11}$$

then *horizontal* non-translational is declared and the object is divided into $|d_{c_{\max}} - d_{c_{\min}}| + 1$ *horizontal* regions. Each region is assigned one displacement as follows: the region closest to $c_{\max}$ is assigned $d_{c_{\max}}$ and the region closest to $c_{\min}$ is assigned $d_{c_{\min}}$. The motion of the other regions is interpolated based on $d_{c_{\min}}$ and $d_{c_{\max}}$.

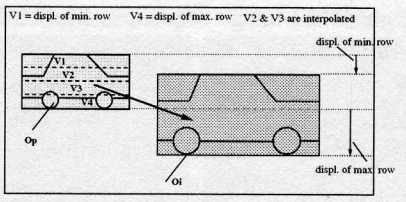


Figure 4: Vertical non-translational motion estimation.

Detection of motion at image margin MBB-based motion estimation will be affected by objects entering or leaving the visual field. Therefore, this situation has to be explicitly detected to adapt the estimation. Motion at image borders is detected by small motion of the MBB-side that is closer to the image border. The object motion is defined based on the motion of the MBB-side that is far from the the image border. This consideration is important for event-based representation of video. It enables monitoring object activity as soon as the objects enter or leave the image.

Compensation of interlaced artifacts Video is either interlaced or non-interlaced. The interlacing often disturbs image edges aligned vertically. In interlaced video, vertical motion estimation can be distorted because of aliasing where two successive fields have different rasters. The effect is that the vertical motion vector will fluctuate by ± 1 between two fields. To compensate for this fluctuation, the current and previous vertical displacements are compared; if they deviate only by one pixel, then the minimal displacement of the two is selected. Another (computationally more expensive) method to compensate for the effect of interlaced video is to interpolate the missing line of the raster so that both fields becomes on the same raster. This interpolation results in shifting each line of the field; therefore, it must be done differently for different fields. Such an approach has been investigated in [4] which shows that the effect of the interlaced alias can be significantly reduced.

5 Results and discussion

5.1 Evaluation criteria

Evaluation criteria for motion estimation techniques can be divided into:

1) Accuracy criteria: Two subjective evaluation criteria to evaluate the accuracy of the estimated motion vectors are used. The first criterion is to display the estimated vector fields and the original image side by side (Fig. 5). The second criterion is based on motion compensation. Motion compensation is a non-linear prediction technique where the current image $I(n)$ is predicted from $I(n-1)$ using the motion estimated between these images (Fig. 7).
2) Consistency criteria: The second category of evaluation criteria is consistency of the motion vectors throughout the image sequence. Motion-based object tracking is one way to measure the consistency of estimated motion vector.
3) Implementation criteria: An important implementation-oriented evaluation criterion is the cost of computing the motion vectors. The real-time aspect is an important evaluation aspect when using motion estimation in critical real-time applications, such as video surveillance, retrieval, or frame-rate conversion. It is important to evaluate proposed methods based on these criteria if these are intended for usage in real-time environment.

There are also objective criteria, such as the Mean Square Error (MSE), the Root Mean Square Error (RMSE) and the PSNR to evaluate the accuracy of motion estimation. The selection of an appropriate evaluation criterion depends on the application. In this paper, the applications are real-time object tracking, video surveillance, and object-based video retrieval. Objective evaluation criteria are not as appropriate as in the case of coding or noise reduction applications.

5.2 Evaluation and discussion

5.2.1 Computational costs

Simulation results show that the computational cost for the proposed object-based motion estimation (including object segmentation and object matching) is about $\frac{1}{15}$ of the computation cost of fast block matching [5]. This block-based method has a complexity about forty times lower than that of a Full-search block matching algorithm which is used in various MPEG-2 encoders. Furthermore, regular (i.e., the same operations are applied for each object) MBB-based object partition and motion estimation are used. Because of its low computational cost and regular operations, this method is suitable for real-time applications, such as video retrieval.

5.2.2 Quality of the estimated motion

In case of partial or complete object occlusion, object-oriented video analysis relies on the estimated motion of the object to predict its position or to find it in case it is lost. Thus a relatively accurate motion estimate is required which needs also to be fast. Block matching is one of the fastest and relatively reliable motion estimation techniques. It is used in many applications. In the video analysis system presented in this paper faster techniques are, however, needed. In addition, block matching gives motion estimation for blocks of the image and not for objects and it can fail if there is insufficient structure. The proposed method provides a response in real-time and gives good estimates to use for tracking, event detection, and high-level video representation [1] for video surveillance and retrieval.

Fig. 5 shows the horizontal and vertical components of the motion field between two frames of the sequence 'Highway', estimated by block matching [5] and proposed object matching methods. Both components are encoded separately and the magnitudes are scaled for display purposes.

It is likely that, in many applications, block-matching techniques will be used. A drawback of block-matching is that they deliver non-homogeneous motion vectors inside objects which affect motion-based video processing techniques, such as noise reduction. On the other hand, the proposed cost-effective object-based motion estimation provide homogeneous motion vectors inside objects. Using this motion information, the block-matching motion field can be enhanced [3]. Fig. 6 displays an example of the incomplete block-matching motion compensation [5]. It also shows the better performance of the proposed object-matching motion compensation. The integration block and object motion information is an interesting research topic for applications, such as motion-adaptive noise reduction or image interpolation.

Another quality measure is given by comparing the object predictions using block matching and object matching. Fig. 7 shows that, despite being a simple technique, the proposed method gives good results compared to more sophisticated block matching techniques. In [2] it is shown that the proposed method produces acceptable results also in complex scenes such as when the camera is not static.

To illustrate the reliability of the proposed object-matching-based algorithm throughout image sequences (video shots) the estimated trajectory of each object in three video shots is displayed in Fig. 8. The trajectories are represented by the position of the center of the object-MBB at each time n. As can be seen, the proposed method is reliable even in the presence of multiple small occluding objects ('Urbicande' shot) and in outdoor environments with occlusion, shadows and other artifacts ('Highway' and 'Survey' shots).

6 Conclusion

A real-time object motion estimation method is proposed in this paper. It is based on an explicit matching of arbitrarily-shaped objects and two estimation steps: 1) estimation of the displacement of the mean coordinate of the MBB and 2) estimation of the displacements of the four MBB-sides. If these estimates differ significantly, a non-translation motion is assumed and different motions are assigned to different image regions. Our method does not require accurate segmentation for applications such as motion and event classification.

In the proposed method, extracted object data (e.g., size, MBB, position, motion direction) is used in rule-based steps: object correspondence, estimation of the MBB motion based on the displacement of the sides of the MBB, i.e., the estimation process is independent of the intensity signal and detecting motion types (e.g., scaling, translation) by analyzing the displacements of the four MBB sides and assigning different motion vectors to different object regions.

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.

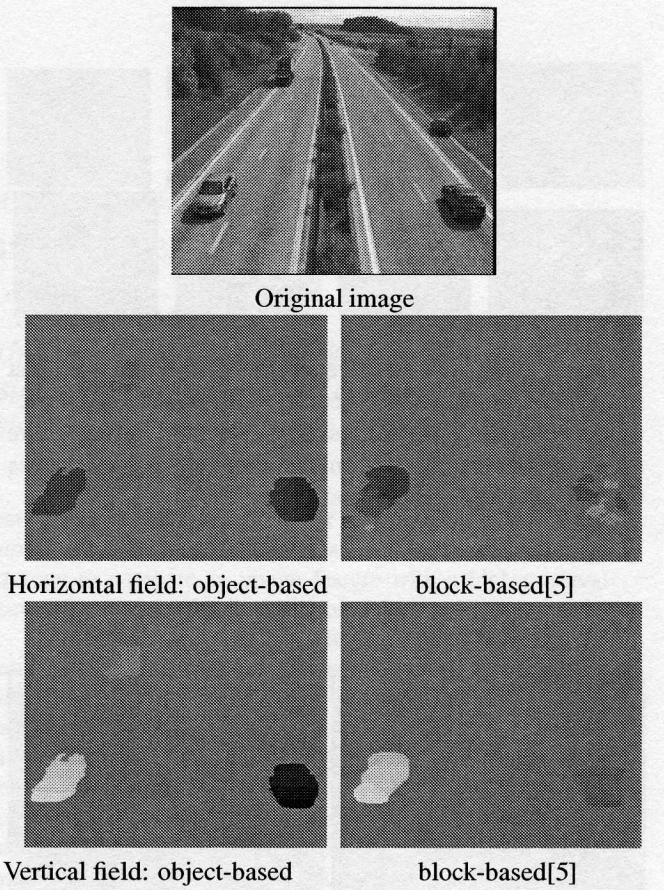


Figure 5: Object versus block motion. Note the non-translation motion of the left car. Motion is coded as gray-levels where strong dark indicates fast motion to the left or up and strong bright indicates fast motion to the right or down.

References

[1] A. Amer. *Object and Event Extraction for Video Processing and Representation in On-Line Video Applications*. PhD thesis, INRS-Télécommunications, Dec. 2001. www.inrs-telecom.uquebec.ca/users/amer/phd/.

[2] A. Amer and E. Dubois. Segmentation-based motion estimation for video processing using object-based detection of motion types. In *Proc. SPIE Visual Communications and Image Process.*, volume 3653, pages 1475–1486, San Jose, CA, Jan. 1999.

[3] A. Amer and E. Dubois. Image segmentation by robust binarization and fast morphological edge detection. In *Proc. Vision Interface*, pages 357–364, Montréal, Canada, May 2000.

[4] H. Blume. Bewegungsschätzung in videosignalen mit parallelen örtlich zeitlichen prädiktoren. In *Proc. 5. Dortmunder Fernsehseminar*, volume 0393, pages 220–231, Dortmund, Germany, 29 Sep.- 1 Oct. 1993. In German.

[5] G. de Haan, T. Kwaaitaal-Spassova, M. Larragy, and O. Ojo. IC for motion compensated 100 Hz TV with smooth movie motion mode. *IEEE Trans. Consum. Electron.*, 42(2):165–174, May 1996.

[6] N. Diehl. Object-oriented motion estimation and segmentation in image sequence. *Signal Process., Image Commun.*, 3(1):23–56, Feb. 1991.

[7] E. Dubois and T. Huang. Motion estimation. In R. Chellappa, B. Girod, D. Munson, and M. V. M. Tekalp, editors, *The past, present, and future of image and multidimensional signal processing*, pages 35–38. IEEE Signal Processing Magazine, Mar. 1998.

[8] F. Dufaux and F. Moscheni. Segmentation-based motion estimation for second generation video coding techniques. In L. Torres and M. Kunt, editors, *Video coding: Second generation approach*, pages 219–263. Kluwer Academic Publishers, 1996.

[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):203–217, 1998.

[10] A. Mitiche and P. Bouthemy. Computation and analysis of image motion: a synopsis of current problems and methods. *Intern. J. Comput. Vis.*, 19(1):29–55, 1996.

[11] H. Nicolas and C. Labit. Motion and illumination variation estimation using a hierarchy of models: Application to image sequence coding. Technical Report 742, IRISA, July 1993.

[12] 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):1147–1169, 1999.

[13] C. Stiller. Object-based estimation of dense motion fields. *IEEE Trans. Image Process.*, 6(2):234–150, Feb. 1997.

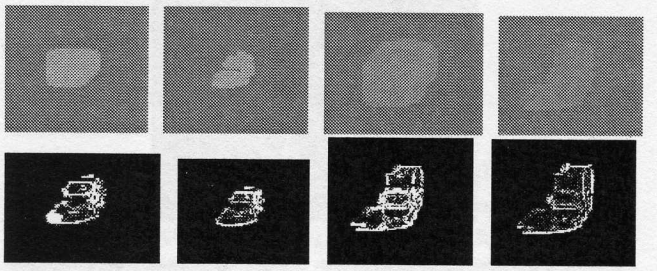
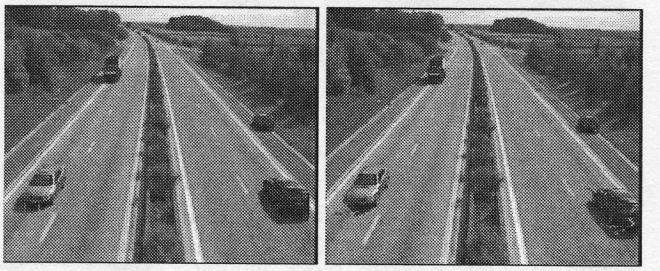
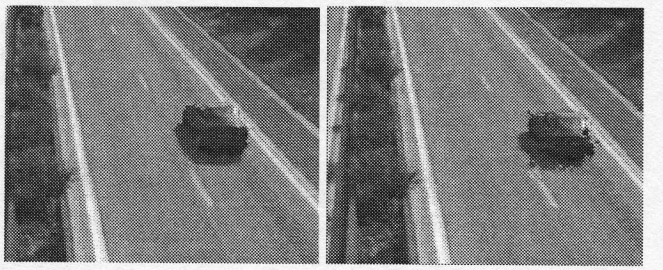


Figure 6: Object-matching versus block-matching: the first row shows a block motion vectors [5] and object motion vectors in sequence. The second row shows the mean-square error between the motion compensated and original images using block vectors and object vectors.



(a) Object-based prediction $I(196)$. The objects are correctly predicted.

(b) Block-based prediction $I(196)$. Note the artifacts introduced at object boundaries.



(c) Object-based prediction $I(6)$ (zoomed in).

(d) Block-based prediction $I(6)$ (zoomed in).

Figure 7: Prediction of objects: block-based [5] prediction introduce various artifacts while proposed block-based prediction gives smoother results inside objects and at boundaries. Similar performance is obtained also in complex scenes (e.g., with camera motion) [2].

Figure 8: Estimated trajectories of the objects in the shots 'Highway', 'Survey', and 'Urbicande'. The original sequence of 'Urbicande' is rotated here by 90° to the right to comply with the CIF format.