Motion Segmentation Using an Occlusion Detector*

Doron Feldman **Daphna Weinshall**
Computer Science Department and the center for neural computation
The Hebrew University of Jerusalem
Jerusalem, Israel 91904

Abstract

We present a novel method for the detection of motion boundaries in a video sequence based on differential properties of the spatio-temporal domain. Regarding the video sequence as a 3D spatio-temporal function, we consider the second moment matrix of its gradients (averaged over a local window), and show that the eigenvalues of this matrix can be used to detect occlusions and motion discontinuities. Since these cannot always be determined locally (due to false corners and the aperture problem), a scale-space approach is used for extracting the location of motion boundaries. A closed contour is then constructed from the most salient boundary fragments, to provide the final segmentation. The method is shown to give good results on pairs of real images taken in general motion. We use synthetic data to show its robustness to high levels of noise and illumination changes; we also include cases where no intensity edge exists at the location of the motion boundary, or when no parametric motion model can describe the data.

1 Introduction

Motion-based segmentation involves the partitioning of images in a video sequence into segments of coherent motion. There are two main approaches to motion segmentation: one may assume a global parametric motion model and segment the image according to the parameters of the model (e.g., [5, 14, 15, 20]), or one may assume piecewise smooth motion and identify the boundaries along motion discontinuities (e.g., [1, 6, 13, 19]).

In this work we focus on the extraction of motion boundaries, which are defined *locally* as boundaries between different motions (since many real video sequences do not obey any global motion model). In addition, we restrict ourselves to solutions which do not rely on the existence of color or texture boundaries between the moving object and the background while computing motion boundaries (but see, for example, [2, 6, 18]). This is motivated by humans' ability to segment objects from motion alone (e.g., in random dot stereograms), and by the need to avoid over-segmentation of objects whose appearance includes varying color and textures. Finally, we only consider local properties of the motion profile, in order to be able to deal with pairs of frames or stereo pairs (but see, for example, [17]).

Motion boundaries can be computed by clustering a previously computed motion field (e.g., [15,20]). The problem is that motion discontinuities are found on exactly those locations where the motion field computation is least reliable: since all optical flow algorithms rely on the analysis of a region around a point (even if only to compute first order derivatives), the optical flow must be continuous within the region to support reliable computation. This chicken-and-egg problem, which is characteristic (though to a lesser extent) of the computation of intensity edges and some related problems, makes motion segmentation particularly challenging. On the other hand, the successful computation of motion discontinuities can be useful for a number of applications, including motion computation (by highlighting those areas where the computation should be considered unreliable) and object segmentation from multiple cues.

In our approach we start by considering the video sequence as a spatio-temporal intensity function, where the goal is to extract information from this spatio-temporal structure. Video sequences have highly regular temporal structure, with regions of coherent motion forming continuous tube-like structures. These structures break where there

*This research was supported by the EU under the DIRAC integrated project IST-027787

is occlusion, creating spatio-temporal corner-like features. Using a differential operator that detects such features, we develop an algorithm that extracts motion boundaries.

Specifically, our algorithm is based on the occlusion detector described in Section 2. This operator is used to extract a motion boundary at any given scale, as described in Section 3. Since different scales may be appropriate for different parts of the image, a cross-scale optimal boundary is computed, based on the response of the detector. Finally, a closed contour is built along the most salient boundary fragments to provide the final segmentation. In Section 4 we analyze the behavior of the detector. Some experimental results are described in Section 5 using two challenging sequences of real images (see, e.g., Fig. 8). We include a number of synthetic examples which are particularly difficult for some commonly used algorithms, in order to demonstrate the robustness of our method. Results from other algorithms, whose implementation was made available by the authors, are provided for comparison.

2 Occlusion Detector

Regarding the video sequence as a spatio-temporal intensity function, let $I(x, y, t)$ denote the intensity at pixel (x, y) in frame t. We refer to the average of the second moment matrix over a neighborhood ω around a pixel as the *Gradient Structure Tensor*

$$\mathbf{G}(x, y, t) \equiv \sum_{\omega} \nabla I \, (\nabla I)^T = \sum_{\omega} \begin{bmatrix} I_x^2 & I_x I_y & I_x I_t \\ I_x I_y & I_y^2 & I_y I_t \\ I_x I_t & I_y I_t & I_t^2 \end{bmatrix} \tag{1}$$

This matrix has been invoked before in the analysis of local structure properties. In [7], eigenvalues of $\mathbf{G}$ were used for detecting spatio-temporal interest points. In [12] it was suggested that the eigenvalues of $\mathbf{G}$ can indicate spatio-temporal properties of the video sequence and can be used for motion segmentation. The idea behind this is reminiscent of the Harris corner detector [3], as it detects 3D "corners" and "edges" in the spatio-temporal domain. Here we take a closer look and develop this idea into a motion segmentation algorithm.

Specifically, if the optical flow in ω is (v_x, v_y) and the brightness constancy assumption [4] holds, then

$$\mathbf{G} \cdot (v_x, v_y, 1)^T = 0 \tag{2}$$

Hence, 0 is an eigenvalue of $\mathbf{G}$. Since $\mathbf{G}$ is positive-semidefinite, we can use the smallest eigenvalue of $\mathbf{G}$ as a measure of deviation from the assumptions above, which leads to the following definition:

Definition 2.1. *Let $\lambda(x, y, t)$ denote the smallest eigenvalue of the* Gradient Structure Tensor $\mathbf{G}(x, y, t)$. *The operator λ is the* occlusion detector.[1]

We do not normalize λ with respect to the other eigenvalues of $\mathbf{G}$ (as in [12]), since it may amplify noise.

In order to provide rotational symmetry and avoid aliasing due to the summation over the neighborhood ω, we define ω to denote a Gaussian window, and the operation $\sum_\omega$ in (1) stands for the convolution with a Gaussian. Since we do not assume temporal coherence of motion, the Gaussian window is restricted to the spatial domain, as explained in Section 3.

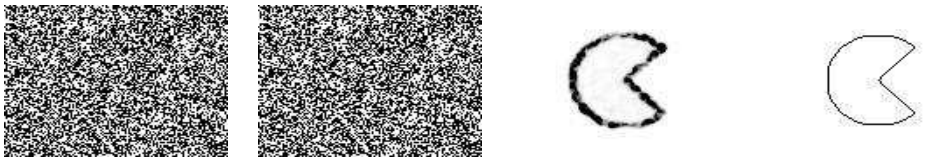
Figure 1 demonstrates the detector results on a simple synthetic example. In this example there are no intensity or texture cues to indicate the boundaries of the moving object, and it can only be detected using motion cues. The value of λ, shown in Fig. 1c, is low in regions of smooth motion and high values of λ describe the boundary of the moving object accurately.

2.1 Velocity-Adapted Detector

The values of ∇I, and hence of λ, are invariant to translation transformations on I. Additionally, for any rotation matrix $\mathbf{R}$,

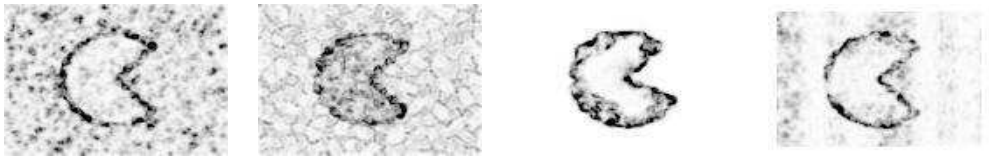
$$|\lambda \mathbf{I} - \mathbf{G}| = |\mathbf{R}||\lambda \mathbf{I} - \mathbf{G}||\mathbf{R}^T| = |\lambda \mathbf{I} - \sum_{\omega} (\mathbf{R}\nabla I)(\mathbf{R}\nabla I)^T|$$

[1] Note that the values of λ at each pixel can be evaluated directly using Cardano's formula.



(a) (b) (c) (d)

Figure 1: Random dots example. A shape is moving sideways, where both the shape and the background are covered by a random pattern of black and white dots. It is impossible to identify the moving object from each of the two frames (a) and (b) (a stereo pair) alone. The occlusion detector (c) (higher values of λ are darker) shows the outline of the object very clearly. Compare to the ground truth (d).



(a) (b) (c) (d)

Figure 2: False λ response. The same example as in Fig. 1: (a) with 20% white noise; (b) with illumination change of 5%; (c) with the object rotating by 20°; (d) with both object and background patterns deformed smoothly.

($\mathbf{I}$ is the identity matrix) and therefore the values of λ are also invariant to the rotation of I. The issue of scale invariance will be discussed in Section 3.

While rotational invariance is desirable in the spatial domain, non-spatial rotations in the spatio-temporal domain have no physical meaning. It is preferable to have invariance to spatially-fixed shear transformations, which correspond to 2D relative translational motion between the camera and the scene. As suggested in [9] by the reference of *Galilean diagonalization*, one can use the velocity-adapted matrix $\tilde{\mathbf{G}}$ given by

$$\tilde{\mathbf{G}} = \begin{bmatrix} G_{11} & G_{12} & 0 \\ G_{21} & G_{22} & 0 \\ 0 & 0 & \lambda_T \end{bmatrix} \quad \text{where} \quad \lambda_T = \frac{\det(\mathbf{G})}{\det(\mathbf{G}^*)} \tag{3}$$

(G_{ij} denote the entries of $\mathbf{G}$, and $\mathbf{G}^*$ denotes the 2×2 upper-left submatrix of $\mathbf{G}$ containing only spatial information).

Definition 2.2. *The operator λ_T is the* velocity-adapted occlusion detector.

To justify this definition, observe that $\tilde{\mathbf{G}}$ is also invariant to translation and spatial rotation. The entry λ_T is an eigenvalue of $\tilde{\mathbf{G}}$, and it has been suggested that it encodes the temporal variation, being the "residue" unexplained by pure-spatial information.

In practice, λ_T gives results similar to λ, though it has certain advantages, as discussed in Section 4. In the remainder of the paper we use λ to denote either operator, unless stated otherwise.

2.2 Detector Effectiveness

High values of λ indicate significant deviation from (2), which is often due to the existence of a motion boundary. Other sources of large deviations include changes in illumination (violation of the brightness constancy assumption), or when the motion varies spatially (motion is not constant in ω). However, often these events lead to smaller λ values as compared to motion boundaries (see Fig. 2), in which case the boundary response can be distinguished from a false response by thresholding.

Low values of λ do not necessarily indicate that the motion in ω is uniform. The rank of $\mathbf{G}$ is affected by spatial structure as well as temporal structure, so λ may be low even at motion boundaries, when certain spatial degeneracies exist. Specifically, this occurs when there is local ambiguity, i.e., when the existence of a motion boundary cannot be

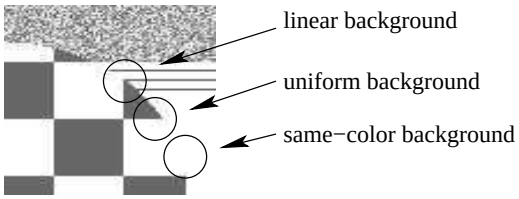


Figure 3: Areas where the λ detector is likely to give low values despite the existence of a local motion boundary.

determined locally. This includes areas where the occluding object and its background are of the same color, areas where the background is of uniform color, and areas where the background texture is uniform in the direction of the motion (Fig. 3). In the first case the rank of $\mathbf{G}$ is 0, and in the other cases the rank of $\mathbf{G}$ may be 1 or 2, depending on the appearance of the occluding object (recall that the λ detector is high when the rank of $\mathbf{G}$ is 3). In these cases, the background may be interpreted as part of the moving object, since no features in the background appear to vanish due to occlusion.

The response of λ to occlusion occurs only where some background features become occluded. Clearly boundary location cannot always be inferred based on local information alone, and it is therefore necessary to integrate information across larger areas of the image. This is done using scale-space techniques, as discussed in Section 3.

2.3 Temporal Aliasing

Since real video data is discrete, the partial derivatives in the definition of λ must be estimated. This is done by convolving I with the partial derivatives of a 3-dimensional Gaussian. Rotational invariance implies that the spatial variance in the X and Y directions should be the same, and the kernel is therefore an ellipsoidal Gaussian with spatial variance s_{xy} and temporal variance s_t. Due to the distortion introduced by the convolution, it is desirable that these values be small.

Estimating the temporal partial derivative from video presents a severe aliasing problem. Since video frames represent data accumulated during short and sparse exposure periods, and since a feature may move several pixels between two consecutive frames, data is aliased in the temporal domain significantly more than in the spatial domain. We overcome this problem by taking advantage of the spatio-temporal structure of video, as described next.

Suppose that the velocity in a certain region is $v = (v_x, v_y)$, and therefore

$$I(x, y, t) = I(x - v_x t, y - v_y t, 0) \tag{4}$$

The temporal derivative in $t = 0$ is given by

$$I_t = -v_x I_x - v_y I_y \tag{5}$$

In discrete video, I_t can be estimated by convolution in the T direction, which, due to (4), is the same as convolution in the v direction of a subsample of $I(x, y, 0)$ at intervals of size $|v|$. In order to avoid aliasing due to undersampling while estimating I_t, the Sampling Theorem requires I to be band-limited, so that its Fourier transform vanishes beyond $\pm\frac{1}{2|v|}$. This can be achieved by smoothing with a spatial Gaussian. However, smoothing poses a notable drawback, as it distorts the image data, causing features to disappear, merge and blur.

An alternative approach, closely related to the concept of "warping" (e.g., [10]), would be to take advantage of prior estimates of the optical flow. If a point is estimated to move at velocity $u = (u_x, u_y)$, we can use the convolution of I in the direction of $(u_x, u_y, 1)$ to estimate the directional derivative I_u and apply

$$I_t = I_u - u_x I_x - u_y I_y \tag{6}$$

The convolution that yields I_u is equivalent to subsampling in the direction of $v - u$, and thus the estimate of I_t is unaliased if the Fourier transform vanishes beyond $\pm\frac{1}{2|v-u|}$. This occurs when either the estimated velocity u is close to the real velocity v, or the region is smooth. This is particularly important, as the estimation of optical flow in smooth regions is often inaccurate. Also note that the spatial smoothness of u is not required.

(a) (b) (c) (d)

Figure 4: Checkerboard example: (a) A frame from the sequence; (b) and (c) show the response of λ at fine ($s_{xy} = 1$) and coarse ($s_{xy} = 10$) scales respectively. At the fine scale, λ only responds at intensity edges (which appear like discrete "bursts"), while the entire contour is visible at the coarse scale, alas with considerable distortion. (d) shows the final contour selected by integrating over scales.

Note that temporal smoothing has no effect on the aliasing problem, and it is desirable to have as little temporal smoothing as possible.

3 Extraction of Motion Boundaries and Scale-Space Structure

Recall from Section 2.2 that λ does not respond to motion boundaries when the boundary cannot be inferred locally (e.g., when the object and the background are of the same color locally). While there may be no cues to indicate the location of the boundary in a fine scale, in a coarser scale (i.e., in a larger neighborhood) there may be enough information and λ may respond. Thus we incorporate multi-scale component in our algorithm, in order to detect motion boundaries that are not detectable at fine scales.

In order to define the notion of scale in our algorithm, note that the evaluation of λ involves Gaussian convolutions in two different stages – during the estimation of the partial derivatives, and when taking the average over the neighborhood ω. In both cases, larger Gaussians lead to coarser structures, and we shall refer to the size of the Gaussian as the *scale*. In this work we will only consider the spatial scale.

The notion of scale has been studied extensively for features such as edges and blobs. As with these features, different structures can be found at different scales. The response of λ to noise, which can occur in finer scales, is suppressed in coarser scales. On the other hand, localization is poor at coarse scales and motion boundaries may break and merge.

Figure 4 illustrates this idea – at fine scale (Fig. 4b), λ responds only at discrete locations, because the background consists of regions with constant color, and the occlusion can be detected only where there are color variations in the background. In the coarser scale (Fig. 4c), the neighborhood of every boundary point contains gradients in several directions and the boundary is detected continuously. In Section 3.2 we describe a method to combine data from multiple scales.

Image features, such as edges, typically shift and become distorted at coarse scales. The scale space structure of motion boundary edges (and in particular our *occlusion detector*) has its own particular biases in coarse scales. As discussed in Section 4, motion boundaries at coarse scales are shifted towards the occluded side, i.e., the occluding objects becomes "thicker". In addition, it can be shown that the bias is stronger when there is a large intensity difference between the object and the background, and it increases with scale.

3.1 Scale Normalization

One problem with multi-scale analysis is that derivatives decrease with scale. Indeed, if $0 \leq I \leq 1$, then

$$|I_x|, |I_y| \leq \frac{1}{\sqrt{2\pi s_{xy}}} \tag{7}$$

when smoothing with a Gaussian of variance s_{xy}. This well-known problem can be handled by scale normalization, as proposed in [8]. Scale normalization is done by defining the *scale-normalized* partial derivatives

$$I_x^{(s_{xy})} = \sqrt{s_{xy}} \cdot \frac{\partial}{\partial x}(g_{s_{xy}} * I) \quad \text{and} \quad I_y^{(s_{xy})} = \sqrt{s_{xy}} \cdot \frac{\partial}{\partial y}(g_{s_{xy}} * I) \tag{8}$$

where $g_{s_{xy}} *$ stands for convolution with a Gaussian with variance s_{xy}. Thus $I_x^{(s_{xy})}$ and $I_y^{(s_{xy})}$ are used in the evaluation of λ instead of I_x and I_y. Note that scale normalization does not violate the assumptions leading to the definition of λ in Section 2.

One important property of scale normalization is that λ becomes invariant to spatial scaling of I. This means that λ gives comparable values for a video sequence in different resolutions.

To see this, let us scale I by α, and define

$$J(x, y, t) = I(x/\alpha, y/\alpha, t) \tag{9}$$

Substituting (9) into (8) yields

$$\begin{array}{rcl} J_x^{(\alpha^2 s_{xy})}(\alpha x, \alpha y, t) & = & I_x^{(s_{xy})}(x, y, t) \\ J_y^{(\alpha^2 s_{xy})}(\alpha x, \alpha y, t) & = & I_y^{(s_{xy})}(x, y, t) \end{array} \tag{10}$$

Let s_ω denote the variance of the Gaussian window ω, and let $\mathbf{G}^{(s_{xy}, s_\omega)}[I]$ denote the second moment matrix defined in (1), with the scales of differentiation and averaging s_{xy} and s_ω, respectively. From (10) it follows that

$$\left(\mathbf{G}^{(s_{xy}, s_\omega)}[I]\right)(x, y, t) \quad = \quad \left(\mathbf{G}^{(\alpha^2 s_{xy}, \alpha^2 s_\omega)}[J]\right)(\alpha x, \alpha y, t) \tag{11}$$

That is to say, if J is a scaling by α of I, then the value of λ at (x, y, t) in I at scales s_{xy}, s_ω will be the same as at the corresponding point in J at scales $\alpha^2 s_{xy}, \alpha^2 s_\omega$.

For our purpose of computing a good *occlusion detector*, it follows from (11) that as long as our computation scans all scales in scale space, the result does not depend on the image resolution.

Note that in order for λ to be scale-invariant, it follows from (11) that s_ω must be proportional to s_{xy}, as in [7]. In our implementation we use $s \equiv s_{xy} = s_\omega$, which defines a single scale s. We denote the λ evaluated at scale s as $\lambda^{(s)}$.

3.2 Boundary Extraction in Scale-Space

Since λ is computed by taking the average over a neighborhood, its response is diffuse. We wish to extract a ridge curve where λ is strongest. This can be defined locally as points where λ is maximal in the direction of the maximal principal curvature, which can be expressed as

$$\left\{ \begin{array}{rcl} \lambda_{xy}(\lambda_x^2 - \lambda_y^2) - \lambda_x \lambda_y (\lambda_{xx} - \lambda_{yy}) & = & 0 \\ (\lambda_{xx} + \lambda_{yy}) \cdot \left((\lambda_{xx} - \lambda_{yy})(\lambda_x^2 - \lambda_y^2) + 4\lambda_x \lambda_y \lambda_{xy}\right) & < & 0 \\ \lambda_x^2 \lambda_{yy} - 2\lambda_x \lambda_y \lambda_{xy} + \lambda_y^2 \lambda_{xx} & < & 0 \end{array} \right. \tag{12}$$

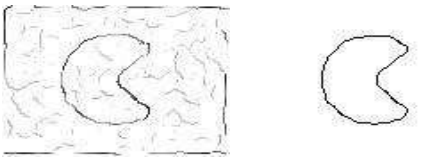
Thus, at every scale s, the values of λ and its derivatives are computed, and the ridge can be extracted. For reasons of numerical stability, at each scale s the derivatives of $\lambda^{(s)}$ are computed with the same Gaussian smoothing s.

Different boundaries are extracted at different scales, as fine-scale boundaries may often split because of the absence of local information, and coarse-scale boundaries may disappear or merge. Since these may occur at different parts of the image at different scales, we wish to select different scales for boundary extraction at different localities (as in [8]). Considering the multi-scale boundary surface as the union of all ridges in $\lambda^{(s)}$ for $s \in (0, \infty)$, we wish to find a cross-scale boundary where $\lambda^{(s)}$ is maximal. This can be expressed as

$$\left\{ \begin{array}{rcl} \lambda_s & = & 0 \\ \lambda_{ss} & < & 0 \end{array} \right. \tag{13}$$

using the scale-derivatives of λ.

Combining (12) and (13) defines the final *cross-scale motion boundary*. It is a curve in the three-dimensional space $X - Y - S$, defined by the intersection of the two surfaces defined respectively by these 2 sets of equations.



(a) (b)

Figure 5: Saliency measure. (a) All boundaries extracted from the random dots example with illumination changes (Fig. 2b); intensity codes λ response. (b) The most salient closed contour.

3.3 Segmentation

As stated above, λ also has some false responses which lead to the selection of false boundary fragments. It is therefore necessary to define a saliency criterion, which is used to select the most interesting boundaries. Since we regard λ as a measure of local boundary strength, for each connected set of boundary points we define the *saliency measure* to be the sum of the value of λ along the boundary, as in [8]. This measure may be sensitive to fragmentation of the boundary, so in our implementation we tolerate small gaps.

Finally, segmentation is achieved by searching for closed contours with high saliency and small gaps. We employ a simple greedy heuristic to connect the motion boundary fragments into a continuous boundary with maximal saliency and minimal gaps. Since the extracted boundaries are usually almost complete, this heuristic gives good results (see Fig. 5).

4 Analysis

In order to analyze the performance of the proposed technique, we consider a video of two moving layers l^1, l^2, where w.l.o.g. l^2 partially occludes l^1. A frame in the video sequence can be written as

$$I = l^1 \cdot (1 - \alpha) + l^2 \cdot \alpha \tag{14}$$

where α is the *matting map*.

We assume w.l.o.g. that the occlusion edge is perpendicular to the X axis and that at frame $t = 0$ it is at $x = 0$. We further assume that the occlusion edge is a Gaussian-smoothed line, so α is of the form $\alpha_{s_0}(x) = \int_{-\infty}^{x} g_{s_0}(u)du$ (we denote the Gaussian function with variance s as g_s).

If the motions of l^1 and l^2 are (v_x^1, v_y^1) and (v_x^2, v_y^2) respectively, then the video volume is given by

$$I(x,y,t) \quad = \quad l^1(x–v_x^1 t, y–v_y^1 t)\cdot(1–\alpha(x–v_x^2 t)) + l^2(x–v_x^2 t, y–v_y^2 t)\cdot\alpha(x–v_x^2 t) \tag{15}$$

Note that the motion of α is the same as the motion of l^2, since it is the occluding layer.

Denoting the video volume of each layer as $I^k(x,y,t) = l^k(x - v_x^k t, y - v_y^k t)$, the gradient of the video volume is given by

$$\nabla I \quad = \quad (1-\alpha)\cdot\nabla I^1 + \alpha\cdot\nabla I^2 + (I^2 - I^1)\cdot g_{s_0}\cdot\mathbf{n} \tag{16}$$

where $\mathbf{n} = (1, 0, -v_x^2)^T$. Note that $\mathbf{n}$ is perpendicular in space-time to the occlusion edge $(0,1,0)^T$ and to the motion vector $\mathbf{v}^2 = (v_x^2, v_y^2, 1)^T$, i.e., $\mathbf{n}$ is the normal of the plane in the video space formed by the motion of the occlusion edge.

Therefore, ∇I is composed of the matting of ∇I^1, ∇I^2, and a component that depends on $I^2 - I^1$. Note that ∇I^1 is perpendicular to $\mathbf{v}^1$, while both ∇I^2 and $\mathbf{n}$ are perpendicular to $\mathbf{v}^2$. This means that ∇I is composed of two components that are related to the occluding layer and only one that is related to the occluded layer.

For scale-space analysis we use the approximation

$$g * (f \cdot \alpha) \approx (g * f)\cdot(g * \alpha) \tag{17}$$

where g is a Gaussian function and α is an integral of a Gaussian as defined above. Eq. (17) is an equality when f is constant, and it provides a good approximation when f does not change rapidly near $x = 0$ (in each layer separately).

Applying (17), the gradient estimated at scale s, denoted by $\nabla I^{(s)} = \nabla(g_s * I)$, is

$$\nabla I^{(s)} \approx (1-\alpha_{s_0+s}) \cdot \nabla I^{1(s)} + \alpha_{s_0+s} \cdot \nabla I^{2(s)} + (I^{2(s)} - I^{1(s)}) \cdot g_{s_0+s} \cdot \mathbf{n} \tag{18}$$

4.1 Velocity-Adapted Occlusion Detector λ_T

We assume the 2D gradients in each layer are distributed isotropically, in the sense that the mean gradient is 0. Furthermore, we assume that they are uncorrelated. Thus, using (16) and (17), we can write the gradient structure tensor defined in (1) as

$$\begin{aligned} \mathbf{G}^{(s)} &= g_{s_\omega} * \left((1-\alpha)^2 \nabla I^1 (\nabla I^1)^T + \alpha^2 \nabla I^2 (\nabla I^2)^T + (I^2 - I^1)^2 \cdot g_{s_0}^2 \cdot \mathbf{n}\mathbf{n}^T\right) \\ &\approx h_1 \cdot \mathbf{M}^1 + h_2 \cdot \mathbf{M}^2 + h_3 \cdot \mathbf{n}\mathbf{n}^T \end{aligned} \tag{19}$$

where

$$\mathbf{M}^k \equiv \begin{bmatrix} 1 & 0 & -v_x^k \\ 0 & 1 & -v_y^k \\ -v_x^k & -v_y^k & (v_x^k)^2 + (v_y^k)^2 \end{bmatrix} \quad \text{and} \quad \begin{aligned} h_1 &= c^1 \cdot (1 - \alpha_{s+s_0+s_\omega})^2 \\ h_2 &= c^2 \cdot \alpha_{s+s_0+s_\omega}^2 \\ h_3 &= c \cdot g_{s_\omega + (s+s_0)/2} \end{aligned} \tag{20}$$

The constants $c^k = \mathrm{var}(\|\nabla l^k\|)/2$ and $c = \mathrm{var}(l^2 - l^1)/\sqrt{4\pi(s+s_0)}$ describe the distribution of intensities in the layers.

Then, the velocity-adapted occlusion detector from (3) can be shown to be

$$\lambda_T = \frac{(v_x^1 - v_x^2)^2}{1/h_1 + 1/(h_2+h_3)} + \frac{(v_y^1 - v_y^2)^2}{1/h_1 + 1/h_2} \tag{21}$$

From the expression above, we can draw the following conclusions:

- λ_T has a single local maximum.
- In the special case where $c^1 = c^2$ (i.e., both layers have the same intensity variance) and $c \to 0$ (i.e., both layers have similar intensities), λ_T is maximal at $x = 0$.
- In the limit $c \to 0$, λ_T is maximal when $\alpha(x) = \sqrt[3]{c^1}/(\sqrt[3]{c^1} + \sqrt[3]{c^2})$, which means that the detected edge location is biased towards the layer with lower intensity variance. The magnitude of the bias is proportional to $\sqrt{s + s_0 + s_\omega}$.
- If only $c^1 = c^2$ is assumed, then $\frac{d\lambda_T}{dx}(x=0) < 0$, therefore λ_T is maximal at a negative x, which means that the detected edge location is biased towards the occluded layer.

4.2 Occlusion Detector λ

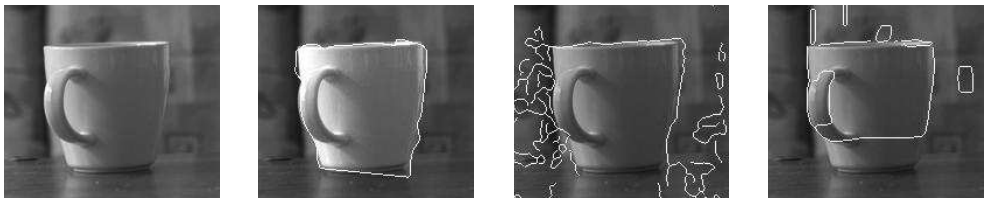
Behavior analysis of the smallest eigenvalue λ is harder. Thus we make the further assumption that $l^1 = l^2$ along the edge. Then we can omit the last term in (19) and get

$$\mathbf{G} = c^1(1-\alpha)^2 \mathbf{M}^1 + c^2 \alpha^2 \mathbf{M}^2 \tag{22}$$

Calculating the eigenvalue of (22), the following can be shown:

- The smallest eigenvalue of $\mathbf{G}$ is given by

$$\lambda = \frac{1}{2}\left(a - \sqrt{a^2 - 4b}\right) \quad \text{where} \quad \begin{aligned} a &= (1-\alpha)^2 c^1 \|\mathbf{v}^1\|^2 + \alpha^2 c^2 \|\mathbf{v}^2\|^2 \\ b &= (1-\alpha)^2 \alpha^2 c^1 c^2 \|\mathbf{v}^1 - \mathbf{v}^2\|^2 \end{aligned} \tag{23}$$



(a) (b) (c) (d)

Figure 6: Cup example. (a) The left image of a stereo pair. (b) Most salient edge detected by our algorithm. (c) Edges in the horizontal component of the optical flow. (d) Edges from a graph cuts segmentation algorithm [6].

(a) (b) (c) (d)

Figure 7: Flower example. (a) The left image of a stereo pair. (b) Most salient closed contour detected by our algorithm. (c) Edges in the optical flow. (d) Edges from a graph cuts segmentation algorithm [6].

- λ has a single local maximum.
- If $c^1\|\mathbf{v}^1\|^2 = c^2\|\mathbf{v}^2\|^2$, then λ is maximal at $x = 0$ — where the edge is located.
- If $c^1\|\mathbf{v}^1\|^2 > c^2\|\mathbf{v}^2\|^2$, then λ is maximal at some $x > 0$, and vice-versa; in other words, the detected edge location is biased towards the layer with lower intensity variance and smaller absolute motion.

The biasing effect of the occlusion relation is not evident due to the particular assumption we have made, although it was observed in our experiments. Note that λ is affected by absolute velocity, unlike the velocity-adapted operator λ_T.

5 Experimental Results

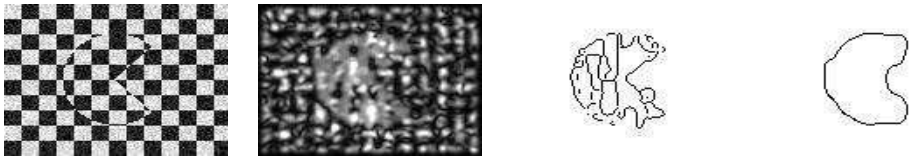
In our experiments we compared our algorithm with the most prominent motion segmentation approaches, wherever code was available. To begin with, we establish the baseline result by segmenting the optical flow. Such a segmentation lies at the heart of some more elaborate segmentation methods, such as [15]. We used a robust and reliable implementation of the Lucas-Kanade algorithm [10], and computed segmented it using a variety of edge operators, including Canny and various anisotropic diffusion methods and clustering methods (e.g., [20]), presenting the best results for each example.

One influential motion segmentation approach relies on graph cuts [6] (and is therefore related to the more traditional regularization based approaches [11]). Code for two variants of this approach is available on the web by the respective authors [6, 18], and we could therefore use their code to establish credible comparisons. We note, however, that in both cases the publicly available code can only work with rectified images. Therefore, in order to obtain fair comparisons, we compared our results to the results of these algorithms only with rectified image pairs, when possible.

Figure 6 demonstrates our algorithm on a stereo pair. The most salient motion boundary is shown in Fig. 6b superimposed on the first input image. Fig. 6c illustrates the baseline result - the edges of the optical flow. Although it is highly unstable in some textureless areas, this does not affect our algorithm's performance, as it is tolerant to poor estimation of optical flow in such regions. Fig. 6d illustrates the best MRF-based segmentation using graph cuts [18]. See also results in Fig. 7.

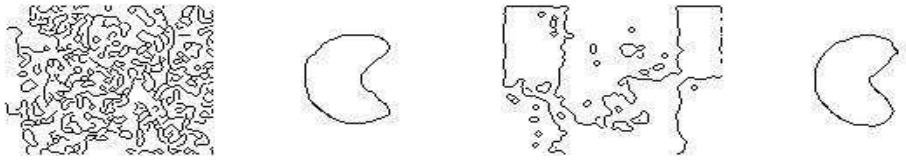
(a) (b) (c)

Figure 8: Octopus example. (a) A frame from the sequence. (b) The most salient closed contour detected by our algorithm. (c) Edges in the optical flow.



(a) (b) (c) (d)

Figure 9: Checkerboard example with 25% white noise. (a) One of the frames; (b) Lucas-Kanade optical flow magnitude; (c) Segmentation using graph cuts; (d) The most salient contour found by our algorithm.



(a) (b) (c) (d)

Figure 10: Random dots example (see Fig. 1). With 20% white noise: (a) Segmentation using graph cuts; (b) The most salient contour found by our algorithm. With smooth non-linear deformation: (c) Segmentation assuming affine motion using an implementation of [20]; (d) The most salient contour found by our algorithm.

Figure 8 shows our algorithm's performance on a video sequence with non-rigid motion and illumination changes. The octopus and the reef below have similar color and texture, and thus spatial coherence is unreliable (note in particular the triangle-shaped projection near the octopus' head, which is in fact a background feature). Although optical flow is inaccurate at motion edges (Fig. 8c), this does not affect the quality of the boundary extracted by our algorithm which uses it (Fig. 8b).

The tolerance to poor optical flow estimation is further demonstrated in Figure 9, where a large amount of noise was added to the synthetic checkerboard sequence, causing numerous optical flow estimation errors. The magnitude of the flow estimation error is often greater than the true flow (Fig. 9b), particularly around the centers of the squares, making segmentation based directly on the optical flow impossible. Results of the MRF-based method are also shown.

The main weakness of many MRF-based methods is the absence of spatial coherence. This is demonstrated on the random dots example in Fig. 10a,b where such methods have no spatial support and therefore fail.

Fig. 10c,d demonstrates our algorithm's advantage when no global motion model can be assumed. In this example, the texture of both the moving object and the background undergo smooth non-linear deformation. The results of applying [20] show that when motion varies smoothly within an object, global model methods fail.

6 Discussion

The occlusion detector we have presented is useful for extracting motion boundaries. Since we do not make any assumption regarding the color or texture properties of objects, or about the geometric properties of the motion, our algorithm works well on natural video sequences where these assumptions cannot be made.

Although our algorithm uses precomputed optical flow, it is only used for estimating the derivatives, and motion properties are not inferred from it. The algorithm is therefore not sensitive to the quality of the optical flow estimation, especially in textureless regions where optical flow estimation is hard.

The algorithm relies mainly on background features which disappear and reappear as a result of occlusion. These features may be sparse and still indicate the location of motion boundaries, as the algorithm processes the data in multiple scales. As opposed to algorithms that rely on motion estimation, our algorithm usually does not require any texture on the occluding object.

Since occlusion is the main cue used by our algorithm, it works well when velocity differences between moving objects are small, since features will still disappear due to occlusion. Algorithms that rely on motion differences may find it hard to distinguish between different objects in such cases.

References

[1] M.J. Black, D.J. Fleet. Probabilistic Detection and Tracking of Motion Boundaries. *IJCV* 38(3):231–245, 2000.

[2] E. Gamble, T. Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. AI Lab, MIT, A.I. Memo No. 970, 1987.

[3] C. Harris, M. Stephens. A combined corner and edge detector. In *Proc. of Alvey Vision Conference*, 147–151, 1988.

[4] B. Horn, B.Schunck. Determining optical fbw. *Artificial Intelligence*, 17:185–203, 1981.

[5] Q. Ke, T. Kanade. A subspace approach to layer extraction. In *Proc. of CVPR'01*, 255–262.

[6] V. Kolmogorov, R. Zabih. Computing Visual Correspondence with Occlusions using Graph Cuts. In *Proc. of ICCV'01*, 2:508–515.

[7] I. Laptev, T. Lindeberg. Space-time Interest Points. In *Proc. of ICCV'03*, 432–439, 2003.

[8] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. *IJCV*, 30(2):117–154, 1998.

[9] I. Laptev, T. Lindeberg. Velocity adaption of space-time interest points *ICPR'04*, 1:52-56.

[10] B. D. Lucas, T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. of IJCAI'81*, 674–679, 1981.

[11] D. W. Murray, B. F. Buxton. Scene segmentation from visual motion using global optimization. *TPAMI*, 9(2):220–228, March 1987.

[12] M. Middendorf, H.-H. Nagel. Estimation and Interpretation of Discontinuities in Optical Flow Fields. In *Proc. of ICCV'01*, 1:178-183.

[13] M. Nicolescu, G. Medioni. A Voting-Based Computational Framework for Visual Motion Analysis and Interpretation. *TPAMI*, 27(5):739–752, May 2005.

[14] J.M. Odobez, P. Bouthemy. MRF-based motion segmentation exploiting a 2D motion model robust estimation. In *Proc. of ICIP'95*, 3:628–631, 1995.

[15] A. S. Ogale, C. Fermüller, Y. Aloimonos. Motion Segmentation Using Occlusions. *TPAMI* 27(6):988–992, June 2005.

[16] H. S. Sawhney, S. Ayer. Compact Representations of Videos Through Dominant and Multiple Motion Estimation. *TPAMI*, 18(8):814–830, August 1996.

[17] J. Shi, J. Malik. Motion Segmentation and Tracking Using Normalized Cuts. In *Proc. of ICCV'98*, 1154–1160, 1998.

[18] M. Tappen, W. T. Freeman. Comparison of Graph Cuts with Belief Propagation for Stereo, using Identical MRF Parameters. In *Proc. of ICCV'03*, 900–907, 2003.

[19] Y. Weiss. Smoothness in layers: motion segmentation estimation using nonparametric mixture. In *Proc. of CVPR'97*, 520–526, 1997.

[20] Y. Weiss, E. H. Adelson A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models. In *Proc. CVPR'96*, 321–326.