Motion-based Segmentation and Contour-based Classification of Video Objects

Gerald Kühne Stephan Richter Markus Beier
Praktische Informatik IV
University of Mannheim
L 15, 16, 68131 Mannheim, Germany
kuehne,richter@informatik.uni-mannheim.de
beier@pi4.informatik.uni-mannheim.de

ABSTRACT

The segmentation of objects in video sequences constitutes a prerequisite for numerous applications ranging from computer vision tasks to second-generation video coding.

We propose an approach for segmenting video objects based on motion cues. To estimate motion we employ the 3D structure tensor, an operator that provides reliable results by integrating information from a number of consecutive video frames. We present a new hierarchical algorithm, embedding the structure tensor into a multiresolution framework to allow the estimation of large velocities.

The motion estimates are included as an external force into a geodesic active contour model, thus stopping the evolving curve at the moving object's boundary. A level set-based implementation allows the simultaneous segmentation of several objects.

As an application based on our object segmentation approach we provide a video object classification system. Curvature features of the object contour are matched by means of a curvature scale space technique to a database containing preprocessed views of prototypical objects.

We provide encouraging experimental results calculated on synthetic and real-world video sequences to demonstrate the performance of our algorithms.

Categories and Subject Descriptors

I.4.6 [**Segmentation**]: Pixel classification; I.4.8 [**Scene Analysis**]: Motion; I.4.8 [**Scene Analysis**]: Object recognition

Keywords

Motion segmentation, object classification, curvature scale space, structure tensor

1. INTRODUCTION

Video object segmentation is required by numerous applications ranging from high-level vision tasks to second-generation video coding [25]. The MPEG-4 video coding standard [10] provides functionality for object-based video coding. Video information can be encoded in a number of arbitrarily shaped video object planes.

Automatic content analysis and indexing methods can benefit from object segmentation algorithms. For instance, it is possible to summarize videos based on the occurrence and activities of video objects [14].

Algorithms for high-level vision tasks such as shape-based object recognition [26, 19, 2] depend on information with regard to object outlines.

We propose an approach to segmenting video object based on motion cues. Motion estimation is performed by estimating local orientations in a spatio-temporal neighborhood with the 3D structure tensor. Thus, information from a number of consecutive frames is exploited. We present a new hierarchical algorithm that embeds the tensor-based motion estimation into a multiresolution framework to allow the calculation of large displacements. The final segmentation is performed by a geodesic active contour model, enabling the simultaneous detection of multiple objects.

Furthermore, we provide a video object classification system that categorizes the segmented video objects into several object classes (e.g. cars, people). This classification system matches curvature features of the object contour to a database containing preprocessed views of prototypical objects.

The remainder of the paper is organized as follows: After summarizing related work, Section 3 describes our segmentation approach. Section 4 introduces the video object classification system. Section 5 presents experimental results. Finally, Section 6 offers concluding remarks.

2. RELATED WORK

Various approaches have been proposed in the field of motion estimation and segmentation. A number of optical flow techniques are reviewed in [3, 4, 18].

Mech and Wollborn [16] estimate a change detection mask by employing a local thresholding relaxation technique. Regions of uncovered background are removed from this mask by using a displacement vector field.

In [13] an edge map is calculated from the inter-frame dif-

This is a preliminary version of an article published in
Proc. of the 9th ACM international conference on Multimedia (MM), pp. 41 – 50, Ottawa, Ontario, Canada, September 2001
by Gerald Kühne, Stephan Kopf, Markus Beier

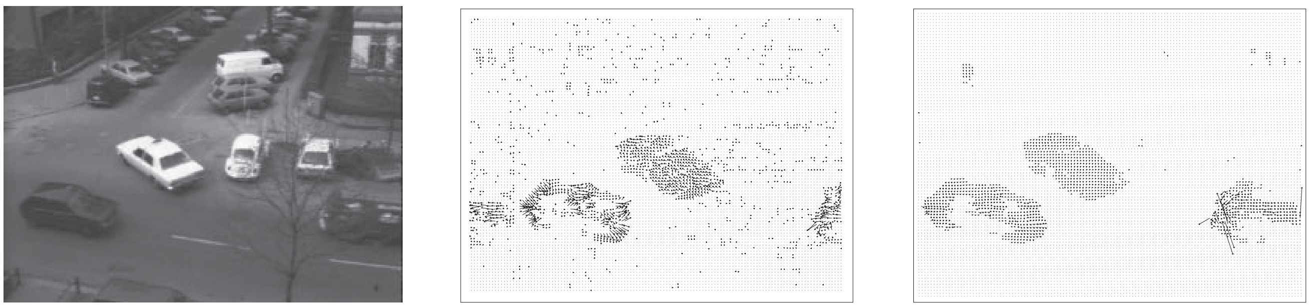

Figure 1: From left to right: (a) Frame 10 from the taxi sequence, (b) optical flow with Lucas Kanade algorithm ($\sigma = 0$, $\lambda_2 > 5$), (c) optical flow with 3D structure tensor.

ference image using the Canny edge detector [6]. The edge map—containing edge pixels from both frames—is compared to the edge map of the current frame and to a background reference frame. The final segmentation is achieved by morphological operators and an additional filling algorithm.

Meier and Ngan [17] propose two approaches. First, they combine an optical flow field with a morphological operator. Second, they employ a connected component analysis on the observed inter-frame difference in conjunction with a filling procedure.

Paragios and Deriche [21] propose a statistical framework based on Gaussian and Laplacian law to detect the moving object's boundary in combination with boundaries obtained from the current frame. They integrate the motion detection and the tracking problems into a geodesic active contour model.

In object classification, contour-based techniques have been under study for a long time. Overviews can be found in [22, 15, 8].

One of the more promising contour analysis techniques is the curvature scale space method (CSS) introduced by Mokhtarian [20, 19] for still images. Here, the contour of an already segmented object is compared to a database containing representations of preprocessed objects. The technique does not depend on size or rotation angle and is robust to noise. In [2] a modified CSS technique can be found. Recently, Richter et al. [23] extended the CSS technique to include the classification of video objects.

3. VIDEO OBJECT SEGMENTATION

In addition to the color and texture information already available in still images, a video sequence provides temporal information. While it is hard to extract semantically meaningful objects based on color and texture cues only, motion cues facilitate the segregation of objects from the background.

Consequently, the first step in our approach is to choose an appropriate motion detector. Various methods have been proposed to estimate motion [3, 4, 18]. However, most of them determine motion parameters on the basis of only two consecutive frames. Hence, these techniques are sensitive to noise and require appropriate compensation methods. Figure 1 illustrates this observation for the classical Lucas Kanade algorithm. We calculated the optical flow for frame 10 of the taxi sequence with the Lucas Kanade implementation used in [3]. The parameters were set to $\sigma = 0$ and $\lambda_2 > 5$ (see [3] for details), motion vectors shorter than 0.2 pixel/frame are suppressed in the figure. The taxi sequence contains four moving objects: the taxi in the middle, a car on the left, a van on the right and a pedestrian in the upper left corner. While the motion for the three main objects is calculated reliably, several misclassifications occur due to noise in the background. Note that the result can be improved significantly by preprocessing the sequence with a 3D Gaussian smoothing filter. However, a drawback to pre-smoothing is the elimination of small structures, e.g. the pedestrian in the upper left corner of the taxi sequence cannot be detected.

In our approach, we employ the 3D structure tensor to analyze motion [5]. Here, motion vectors are calculated by estimating local orientations in the spatio-temporal domain. Figure 1(c) depicts the result for the structure tensor. Here, background noise is eliminated without pre-filtering and reliable motion detection is possible. Even small structures like the pedestrian are identified.

In the following section we describe the structure tensor technique. Then we present a new algorithm that embeds the approach into a multiresoultion framework to allow the detection of large velocities.

3.1 Tensor-based Motion Estimation

Within consecutive frames stacked on top of each other, a video sequence can be represented as a three-dimensional volume with one temporal (z) and two spatial (x, y) coordinates. From this perspective, motion can be estimated by analyzing orientations of local gray value structures [5]. Assuming that illumination does not vary, gray values remain constant in the direction of motion. Thus, stationary parts of a scene result in lines of equal gray values in parallel to the time axis. Moving objects, however, cause iso-gray-value lines of different orientations. Figure 2 illustrates this observation.

Consequently, moving and static parts on the image plane can be determined from the direction of minimal gray value change in the spatio-temporal volume. This direction can be calculated as the direction $\mathbf{n}$ being as much perpendicular to all gray value gradients in a 3D local neighborhood Ω. Thus, for each pixel at position (x, y, z) we minimize

$$\int_{(x',y',z')\in\Omega(x,y,z)} (\nabla_3 I(x', y', z')\mathbf{n})^2 dx'\, dy'\, dz' \qquad (1)$$

where $\nabla_3 := (\partial_x, \partial_y, \partial_z)$ denotes the spatio-temporal gradi-

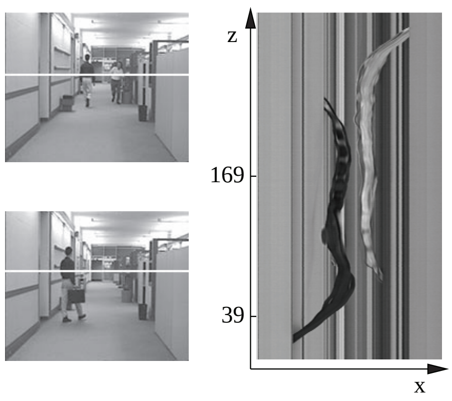

Figure 2: Local orientation of image structures. Left: Frame 169 (top) and frame 39 (bottom) of the "hall and monitor" sequence. Right: Slice of the corresponding spatio-temporal volume taken at the horizontal line marked by the white lines in the single frames.

ent, I the three-dimensional volume and Ω a 3D neighborhood around the pixel at position (x, y, z).

As described in [5, 9], minimizing Equation 1 is equivalent to determining the eigenvector to the minimum eigenvalue of the 3D structure tensor

$$\mathbf{J} = \begin{bmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{xy} & J_{yy} & J_{yz} \\ J_{xz} & J_{yz} & J_{zz} \end{bmatrix} \tag{2}$$

where $J_{pq}, p, q \in \{x, y, z\}$ are calculated within a local neighborhood Ω from

$$J_{pq}(x, y, z) = \int_{\Omega} \partial_p I(x', y', z') \partial_q I(x', y', z') dx'\, dy'\, dz'. \tag{3}$$

By analyzing the three eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ of the 3×3 symmetric matrix, we can classify the local neighborhood's motion. In general, an eigenvalue $\lambda_i \gg 0$ indicates that the gray values change in the direction of the corresponding eigenvector e_i. Figure 3 illustrates the relationship between local structures, eigenvalues, and eigenvectors in the two-dimensional case. Consider, for instance, case 1, where the local neighborhood Ω is centered over a horizontal structure. The gray values within this neighborhood change in one direction. Consequently, $\lambda_1 \gg 0, \lambda_2 = 0$ and the eigenvector e_1 gives the direction of the gray value change.

Within the context of a three-dimensional neighborhood the following observations can be made. All three eigenvalues equal to zero indicate an area of constant gray values, therefore no motion can be detected. If $\lambda_1 > 0$ and $\lambda_2 = \lambda_3 = 0$, gray values change only in one direction. This corresponds to a horizontal (or vertical) structure moving with constant velocity. Consequently, due to the correspondence problem we can only calculate normal velocity.

Real motion can be calculated if gray values remain constant in only one direction, hence, $\lambda_1 > 0, \lambda_2 > 0$ and $\lambda_3 = 0$. This occurs when a structure containing gray value changes in two directions moves at constant speed.

Finally, if all three eigenvalues are greater than zero, we

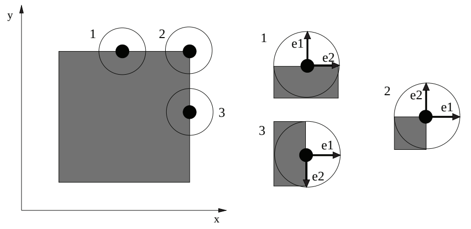

Figure 3: Local structures, eigenvalues, and eigenvectors in two dimensions. Case 1 (case 3): horizontal (vertical) structure, gray values change in one direction, i.e., $\lambda_1 \gg 0$, $\lambda_2 = 0$; case 2: corner, gray values change in more than one direction, i.e., $\lambda_1 = \lambda_2 \gg 0$.

cannot determine the optical flow due to noise.

In real-world video sequences, however, it is impractical to compare the eigenvalues to zero, since due to noise in the sequence small gray value changes always occur. Thus, we introduce normalized coherence measures c_t and c_s that quantify the certainty of the calculations. The coherence measure c_t indicates whether a reliable motion calculation is possible and is defined by

$$c_t = \begin{cases} 0 & \lambda_1 = \lambda_3, \\ \exp\left(\frac{-C}{|\lambda_1 - \lambda_3|}\right) & \text{else} \end{cases} \tag{4}$$

where $C > 0$ denotes a contrast parameter. Areas with $(|\lambda_1 - \lambda_3|) \ll C$ are regarded as almost constant local neighborhoods [27]. A value of c_t near 1.0 indicates that $\lambda_1 \gg \lambda_3$, therefore, a reliable motion calculation can be performed. The opposite is true if the c_t value approaches zero.

The coherence measure c_s,

$$c_s = \begin{cases} 0 & \lambda_2 = \lambda_3, \\ \exp\left(\frac{-C}{|\lambda_2 - \lambda_3|}\right) & \text{else} \end{cases}, \tag{5}$$

provides information whether normal or real motion can be determined. Values near 1.0 allow the calculation of real motion. Otherwise only normal velocities can be specified.

As depicted in Figure 1(c), the structure tensor allows reliable motion calculations and suppresses background noise due to the integration of several consecutive frames. The number of frames used in the motion calculation is determined by the size of the neighborhood Ω. Setting $|\Omega| = 7^3$, for instance, means that the motion calculation for each pixel is performed within a spatio-temporal area of $7 \times 7 \times 7$ pixels.

3.2 Multiscale Motion Estimation

The motion detection approach described so far exhibits problems with sequences containing large velocities. This results from the fixed size of the local neighborhood Ω. Consider an image feature that moves at high velocity. Consequently, it changes its position by a large displacement from one frame to the next. If the displacement exceeds the size of the local neighborhood, the motion of the feature cannot be detected.

To overcome this limitation we developed a new hierarchical algorithm that embeds the structure tensor technique in a linear scale-space framework. Hence, the calculations are performed in a coarse-to-fine manner.

First, a Gaussian pyramid of L levels is constructed from the video sequence. Let $I^1(x,y,z)$ denote the original sequence of size (n_x^1, n_y^1, n_z). Then, the coarser levels are constructed recursively, i.e., for $l = 2, 3, \ldots L$, I^l is calculated from I^{l-1} by spatial smoothing and spatial downsampling by a factor of two.

Then, for each position $p^1 = (x^1, y^1, z) \in [0, n_x^1] \times [0, n_y^1] \times [0, n_z]$ the optical flow vector is calculated. The calculations start at the coarsest level L. The position p^L within this level is determined as $(x^L, y^L, z) = (x^1/2^L, y^1/2^L, z)$. Then, within a local neighborhood Ω centered at the position p^L the structure tensor $\mathbf{J}$ is calculated and the corresponding eigenvalues are evaluated as described in Section 3.1. If motion calculation is feasible, a motion vector $v^L = (v_x^L, v_y^L)$ is determined at this position. Note that due to the subsampling procedure large displacements are reduced appropriately and therefore can be captured within the local neighborhood.

The motion vector v^L determined at the coarsest pyramid level L serves now as an initial guess g^{L-1} at the next pyramid level $L-1$. Since the spatial dimensions double from one level to the next, we adapt the initial guess accordingly, i.e., $g^{L-1} = (2v_x^L, 2v_y^L)$. Thus, at this state we know that at position $p^{L-1} = (x^1/2^{L-1}, y^1/2^{L-1}, z)$ an image feature moves roughly according to g^{L-1}.

The goal at level $L-1$ is now to refine the initial guess. This is done by (1) compensating for the motion vector g^{L-1} within the local neighborhood around p^{L-1} and (2) by calculating a displacement vector d^{L-1} on the modified neighborhood. Hence, the motion vector at this level, v^{L-1}, emerges from a combination of initial guess and displacement, $v^{L-1} = g^{L-1} + d^{L-1}$.

The motion vector v^{L-1} is used as initial guess for the consecutive pyramid level and the algorithm repeats until the highest resolution is reached.

A crucial part of the algorithm is the motion compensation that must be performed on each level in order to allow the displacement calculation. Remember the calculations of the structure tensor elements, e.g. the element J_{xx}:

$$J_{xx}(x,y,z) = \sum_{\Omega} \partial_x I(x',y',z') \partial_x I(x',y',z') dx'\, dy'\, dz'. \tag{6}$$

Here, spatial derivations are calculated within a spatio-temporal neighborhood around the position (x,y,z). If we consider $|\Omega'| = 3 \times 3 \times 3$, patches from three frames of the video sequence, namely, $z-1$, z, $z+1$, are involved in the calculations. Consider now, that an initial guess $g = (g_x, g_y)$ for this local neighborhood is available from the previous pyramid level. Thus, to determine the additional displacement d, it is first necessary to compensate for this guess.

Consequently, Equation 6 changes to

$$\begin{aligned} J_{xx}(x,y,z) = \quad \textstyle\sum_{\Omega'} \quad & \partial_x I(x' + g_x(z'-z), y' + g_y(z'-z), z') \\ & \partial_x I(x' + g_x(z'-z), y' + g_y(z'-z), z') \\ & dx'\, dy'\, dz', \end{aligned} \tag{7}$$

i.e., from frame $z+1$ a 3×3 patch around position $(x + g_x, y + g_y)$, from frame $z-1$ a patch around $(x - g_x, y - g_y)$, and from frame z a patch around (x,y) is used. Obviously, g_x and g_y need not be integer values. Thus, bilinear interpolation is used to determine image values at the subpixel level. Accordingly, the other elements of the tensor $\mathbf{J}$ are calculated under motion compensation.

The need for motion compensation and the use of bilinear interpolation techniques in the hierarchical algorithm clearly affect the performance of the whole method. In order to improve the efficiency it is useful to eliminate those positions in $[0, n_x^1] \times [0, n_y^1] \times [0, n_z]$ in advance where presumably a reliable motion calculation is not possible.

Again, the structure tensor, used here in the spatial domain,

$$\mathbf{J}' = \begin{bmatrix} J'_{xx} & J'_{xy} \\ J'_{xy} & J'_{yy} \end{bmatrix}, \tag{8}$$

is a reliable indicator for this task. Remember that in the two-dimensional case (see Figure 3) the eigenvalues λ_1 and λ_2 provide information about the texturedness of the local neighborhood. Both eigenvalues larger zero indicate a textured region. With respect to motion estimation it is probable that this region can be identified in the consecutive frame, too. Consequently, a full motion vector can be calculated. If only one eigenvalue is greater than zero, the area in question contains a horizontal or vertical structure. Therefore only motion in the direction of the gradient (normal motion) can be determined. On the other hand, a uniform region, i.e., no estimation of motion is possible, results in $\lambda_1 \approx \lambda_2 \approx 0$.

Thus, Shi and Tomasi [24] propose the following reliability measure:

$$\min(\lambda_1, \lambda_2) > T, \tag{9}$$

i.e., a position (x,y) in the image contains a good feature to track, if the lesser eigenvalue exceeds a predefined threshold T.

However, our purpose is slightly different because we want to calculate any kind of motion occurring in the video sequence. Therefore, we modify the reliability measure (Equation 9) to exclude only uniform regions from the motion calculation:

$$r = \begin{cases} 0 & \lambda_1 = \lambda_2, \\ \exp\left(\frac{-C}{|\lambda_1 + \lambda_2|}\right) & \text{else.} \end{cases} \tag{10}$$

A small sum of λ_1 and λ_2 result in values near zero, while in all other cases the reliability measure adopts values near one.

3.3 Motion-based Segmentation

As depicted in Figure 1(c), the motion estimation approach is able to reliably identify regions of interest, though some parts of the van are left out due to low contrast. However, tensor-based motion detection only is not sufficient to provide an accurate segmentation of the objects in question.

We observe two shortcomings that are inherent to this approach. First, due to areas of constant gray values within the moving objects we do not receive dense motion vector fields. In these areas all three eigenvalues are close to zero and therefore motion cannot be calculated. However, it is likely that motion can be estimated at the spatial edges of the moving objects.

Second, the tensor fails to provide the true object boundaries accurately since the calculations within the neighborhood Ω blur motion information across spatial edges.

Consequently, we need (1) a grouping step that will integrate neighboring regions into objects while closing gaps

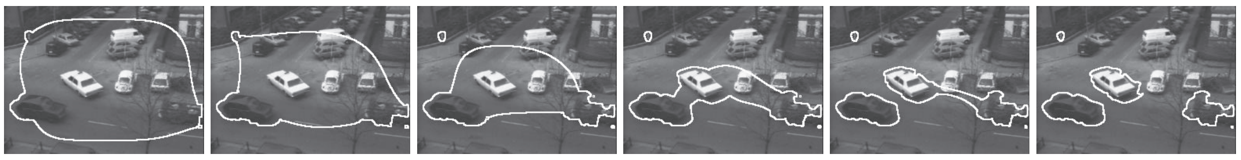

Figure 4: Tensor-driven geodesic active contour. From left to right: contour after 3000, 6000, 9000, 12000, 15000, 17392 iterations. Constant force $c = 0.02$.

and holes and (2) contour refinement based on spatial edge information.

Widely used within this context are active contour models. Basically, a planar parametric curve $\mathcal{C}(s)$ placed around image parts of interest evolves under smoothness control (internal energy) and the influence of an image force (external energy).

In the classical explicit snake model [11] the following functional is minimized

$$\oint_{\mathcal{C}(s)} \left(\alpha|\mathcal{C}'(s)|^2 + \beta|\mathcal{C}''(s)|^2 - \gamma|\nabla I(\mathcal{C}(s))|^2\right) ds \qquad (11)$$

where the first two terms control the smoothness of the planar curve, while the third attracts the contour to high gradients of the image.

To obtain a topological flexibility that will allow the simultaneous detection of multiple objects, we employ geodesic active contours [12, 7]. The basic idea is to embed the initial curve as a zero level set into a function $u : \mathbb{R}^2 \to \mathbb{R}$, i. e., $\mathcal{C}$ is represented by the set of points $\mathbf{x}_i$ with $u(\mathbf{x}_i) = 0$, and to evolve this function under a partial differential equation.

Using a modified energy term this results in the image evolution equation [12, 7]

$$\frac{\partial u}{\partial t} = g(I)(c + \kappa)|\nabla u| + \nabla u \cdot \nabla g \qquad (12)$$

where κ denotes the curvature of a level set, $\nabla := (\partial_x, \partial_y)$ is the spatial gradient, c adds a constant force for faster convergence, and g represents the external image-dependent force or stopping function.

By defining an appropriate stopping function g, we can integrate the tensor-based motion detection into the model. Choosing $g(I) = \hat{s}(I)$, where $\hat{s}$ is a smoothed version of

$$s(I(x,y,z)) = \begin{cases} 1 & |v(x,y,z)| < T_v, \\ 0 & |v(x,y,z)| \geq T_v \end{cases} \qquad (13)$$

stops the curve evolution ($g = 0$) when positions are reached that coincide with "motion pixels". Note that $v = (v_x, v_y)$ denotes the 2D velocity available from the motion estimation step. T_v is a predefined velocity threshold compared against the norm of the motion vector. Hence, our segmentation scheme assumes—in the current state —a static camera.

In the event of a moving camera, a global camera motion estimation has to be performed first. It should then be possible to compare the motion vectors determined from the structure tensor to the vectors resulting from the the global camera parameters [17].

Figure 4 depicts the evolution of the tensor-driven geodesic active contour. The contour succeeds in splitting up and detecting the four different moving objects.

In order to improve the segmentation results, we employ a refinement procedure based not on motion information but on the gradient values within a single frame. As can be seen in Figure 5 (left), the motion-based segmentation detects regions that are slightly larger than the moving objects.

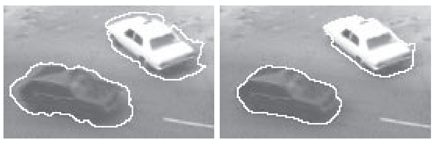

Figure 5: Contour refinement. Left: motion-based segmentation, right: motion-based segmentation with contour refinement (445 iterations, $\tilde{C} = 1.5$).

Thus, we restart the image evolution process using the result from the motion-based segmentation as the zero level set. However, this time a stopping function $\tilde{g}$ based on the spatial gradient is used:

$$\tilde{g}(I) = \frac{1}{1 + |\nabla\hat{I}|^2/\tilde{C}^2} \qquad (14)$$

Here, $\tilde{C}$ is a contrast parameter that diminishes the influence of low gradient values. Figure 5 depicts the performance of the refinement procedure.

4. VIDEO OBJECT CLASSIFICATION

Our system for object classification consists of two major parts, a *database* containing contour-based representations of prototypical video objects, and an *algorithm* to match extracted objects with the database. In the following we summarize the classification approach, for details see [23].

4.1 Curvature Scale Space Representation

The curvature scale space (CSS) technique [1, 20, 23] is based on the idea of curve evolution, i.e., basically the deformation of a curve over time. A CSS image provides a multi-scale representation of the curvature zero crossings of a closed planar contour.

Consider a closed planar curve $\Gamma(u)$,

$$\Gamma(u) = \{(x(u), y(u)) | u \in [0,1]\},$$

with the normalized arc length parameter u. The curve is smoothed by a one-dimensional Gaussian kernel $g(u,\sigma)$ of width σ. The deformation of the closed planar curve is represented by

$$\Gamma(u,\sigma) = \{(X(u,\sigma), Y(u,\sigma)) | u \in [0,1]\},$$

where $X(u,\sigma)$ and $Y(u,\sigma)$ denote the components $x(u)$ and $y(u)$ after convolution with $g(u,\sigma)$.

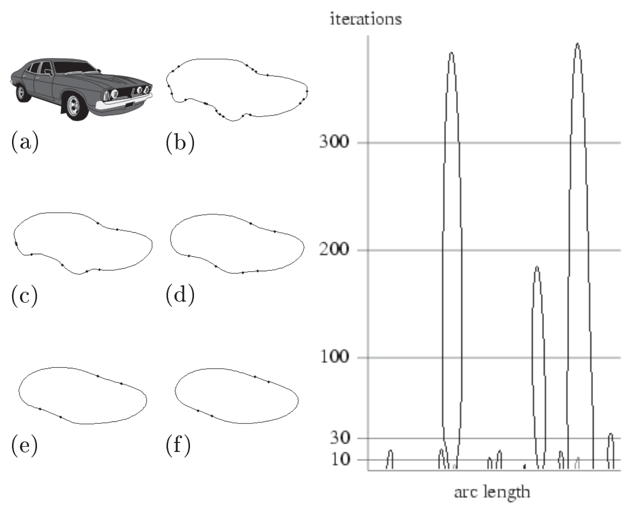

Figure 6: Construction of the CSS image. Left: (a)-(f) Object view and smoothed contour after 10, 30, 100, 200 and 300 iterations. The small dots on the contour mark the curvature zero crossings. Right: Resulting CSS image.

The curvature $\kappa(u,\sigma)$ of an evolved curve can be computed using the derivatives $X_u(u,\sigma)$, $X_{uu}(u,\sigma)$, $Y_u(u,\sigma)$, and $Y_{uu}(u,\sigma)$:

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)\cdot Y_{uu}(u,\sigma) - X_{uu}(u,\sigma)\cdot Y_u(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2)^{3/2}}$$

A CSS image $I(u,\sigma)$ is defined by

$$I(u,\sigma) = \{(u,\sigma)|\kappa(u,\sigma) = 0\}.$$

It shows the zero crossings with respect to their position on the contour and the width of the Gaussian kernel (or the number of iterations, see Figure 6). During the deformation process, zero crossings merge as transitions between contour segments of different curvature are equalized. Consequently, after a certain number of iterations, inflection points cease to exist and the shape of the closed curve is convex. Note that due to the dependence on curvature zero crossings, convex object views cannot be distinguished by the CSS technique.

Significant contour properties that are visible for a large number of iterations result in high peaks in the CSS image. However, areas with rapidly changing curvatures caused by noise produce only small local maxima.

In many cases the peaks in the CSS image provide a robust and compact representation of an object view's contour[19, 20]. Note that a rotation of an object view on the image plane can be accomplished by shifting the CSS image left or right in a horizontal direction. Furthermore, a mirrored object view can be represented by mirroring the CSS image.

A main drawback to the basic CSS technique—where only the two values (position, height) represent a peak in an CSS image—is the occurrence of ambiguities. Certain contours differing significantly in their visual appearance nevertheless have similar images. This is due to the fact that shallow and deep concavities on a contour may result in peaks of the same height in the CSS image.

Abbasi [2] presents several approaches to avoiding these ambiguities, raising the computational costs significantly.

In our extension [23] we extract the width at the bottom of the arc-shaped contour corresponding to the peak. The width specifies the normalized arc length distance of the two curvature zero crossings enframing the contour segment represented by the peak in the CSS image. For each peak in the CSS image three values have to be stored: the position of the maximum, its value (iteration or width of the Gaussian kernel), and the width at the bottom of the arc-shaped contour. It is sufficient to extract the significant maxima (above a certain noise level) from the CSS image. For instance, in the example depicted in Figure 6, and assuming a noise level of 30 iterations, only four data triples have to be stored.

The matching algorithm described in the following section utilizes the information in the peaks to compare automatically segmented video objects with prototypical video objects in the database.

4.2 Object Matching

The objects are matched in two steps. In the first, each automatically segmented object in a sequence is compared to all objects in the database. A list of the best matches is built for further processing. In the second step, the results are accumulated and a confidence value is calculated. Based on it, the object class of the object in the sequence is determined.

In order to find the most similar object in the database compared to a query object from a sequence, a matching algorithm is needed. The general idea is to compare the peaks in the CSS images of the two objects, based on the characterization by the triples (height, position, width).

- In a first step, the best position to compare the two images has to be determined. It might be necessary to rotate or mirror one of the images so that the peaks are aligned best. As mentioned before, shifting the CSS image corresponds to rotation of the original object. One of the CSS images is shifted so that the highest peaks in both CSS images are aligned.

- A matching peak is determined for each peak in the first object. Two peaks may match, if their position and width are within a certain range. Only for the highest peaks, does the height also need to be within a certain range.

- If a matching peak is found, the Euclidean distance of the height and position of the peaks is calculated and added to the difference between the images. If no matching peak can be determined, the height of the peak in the first query object is multiplied by a penalty factor and added to the total difference.

The matching algorithm might return ∞, e.g. if no adequate rotation could be found or if the highest maxima in the CSS images do not match within a given tolerance range. If this is the case, the two objects are significantly different.

All top matches which were recognized are used for accumulation. The object class with a percentage above 75 % is considered to be the class of the sequence.

5. EXPERIMENTAL RESULTS

We subdivide the experimental results achieved with our algorithms into three sections. First, we demonstrate the

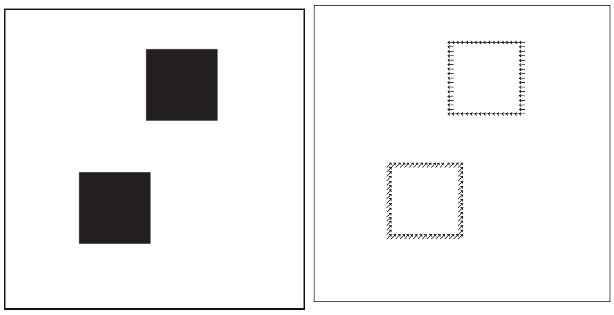

Figure 7: Left: frame from the synthetic sequence, right: motion field calculated by the hierarchical structure tensor approach (for better visibility the flow image is subsampled by a factor of 4).

performance of the multiscale structure tensor approach on a synthetic sequence containing large displacements. Second, we provide segmentation results obtained by the tensor-driven geodesic active contour with respect to two real-world sequences. Finally, results calculated by the object classification algorithm are presented.

5.1 Multiscale Motion Estimation Results

To measure the performance of the hierarchical approach described in Section 3.2, we created a simple synthetic video sequence for which the displacements from one frame to the next are known. Figure 7 (left) shows a frame in this sequence that contains two moving squares. While the upper square (square 1) moves at a constant velocity of 10 pixels per frame from the right to the left, the other square (square 2) moves diagonally upwards at velocity of $(8^2 + 8^2)^{1/2}$.

Figure 7 (right) depicts the result obtained by our multiresolution algorithm using 4 pyramid levels. A closer look at the motion vectors calculated by the algorithm reveals the following observations:

1. At the corners of the squares the velocity could be estimated exactly (square 1: $v = (-10, 0)$, square 2: $v = (8, -8)$. Remember that at a corner moving with constant speed enough texture is available to allow the calculation of the full image motion.

2. On account of the pyramidal structure the velocities at horizontal and vertical structures approximate the real image motion. In general, full motion calculation is possible for points near the corners. At the coarsest pyramid level each point on a horizontal or vertical structure is near the corner (in our specific example). Therefore, an initial full motion guess for these points can be calculated. However, for consecutive pyramid levels full motion calculation is no longer possible since the distance to the corners increases. Consequently, the displacements added to the initial guess refine the motion estimation only in the normal direction.

3. For pixels in the interior of the squares it is not possible to calculate image motion. The reliability measure described in Equation 10 eliminates these points from the calculations.

The results for the synthetic sequence indicate that—under specific circumstances—the proposed approach is able to estimate motion exactly even given the existence of large displacements.

Object	Frame	F_P	F_N	F_P (%)	F_N (%)
car	7	156	61	9.12	3.73
car	8	137	40	8.07	2.62
car	9	208	18	12.00	1.19
car	10	149	10	8.60	0.67
car	11	197	40	11.35	2.53
taxi	7	226	7	14.51	0.56
taxi	8	217	14	14.76	1.14
taxi	9	233	13	15.70	1.05
taxi	10	250	20	17.02	1.67
taxi	11	196	15	13.98	1.23
van	7	135	448	10.77	28.61
van	8	176	663	15.04	40.01
van	9	288	522	22.15	34.03
van	10	198	727	16.94	42.82
van	11	246	800	19.54	44.13

Table 1: Region-based distance for the taxi sequence. Columns 3, 4: false positives, false negatives. Columns 5, 6: percentage of mismatched pixels in comparison to the entire number of pixels of the manual segmentation.

Object	Frame	avg. d	d = 0 (%)	d = 1 (%)	d = 2 (%)	d > 2 (%)
car	7	1.27	38.12	35.15	14.36	12.38
car	8	1.41	27.27	35.23	18.18	19.32
car	9	1.54	33.82	29.90	8.82	27.45
car	10	1.32	35.88	32.35	12.35	19.41
car	11	1.39	34.98	29.06	13.30	22.66
taxi	7	1.42	47.17	23.90	9.43	19.50
taxi	8	1.31	53.97	26.98	3.70	15.34
taxi	9	1.44	44.21	28.95	6.84	20.00
taxi	10	1.52	48.09	24.04	4.92	22.95
taxi	11	1.23	54.89	22.28	7.07	15.76
van	7	4.21	20.59	18.24	10.00	51.18
van	8	6.20	11.36	17.05	11.36	60.23
van	9	6.05	11.18	11.18	7.65	70.00
van	10	6.82	11.54	6.59	7.69	74.18
van	11	7.43	9.95	9.42	7.85	72.77

Table 2: Edge-based distance for the taxi sequence. Column 3: average edge pixel distance. Colums 4-7: percentages of the distances 0,1,2,3..n.

5.2 Segmentation Results

We applied the segmentation algorithm described above to two real-world sequences. The first one is the Hamburg taxi sequence widely used within the computer vision community. Figure 8 illustrates the performance of our segmentation approach on this sequence. We employed the standard structure tensor described in Section 3.1. The parameters were set as follows: (1) The size of the local neighborhood Ω was set to $7 \times 7 \times 7$. (2) The contrast parameter for the coherence measures was set to 5. For all positions with a coherence $c_t > 0.75$ we performed motion estimation, positions with c_t below this value were rejected. Full motion vectors were calculated for positions with $c_s > 0.9$.

The motion estimates were integrated as an external force into the geodesic active contour model (see Section 3.3). For faster convergence we set the external force $c = 0.02$. Note that a value for c greater than zero forces the curve to shrink, while a value smaller zero causes an expansion. Finally, we employed the contour refinement step described in Section 3.3.

In addition to the visual results we provide quantitative measures in Tables 1 and 2. First, we used the region-based distance measures F_P and F_N to compare the automatic segmentation results to those of a manual segmentation. While F_P contains the number of pixels incorrectly marked as object pixels by the automatic segmentation (false positives), F_N sums up object pixels missed by the process (false negatives). Second, we employed an edge-based distance measure. For each contour pixel in the manual segmentation the distance to the closest contour pixel in the automatic segmentation was determined.

The following conclusions can be drawn from the measures: The segmentations of the car and the taxi are acceptable. While the number of pixels detected by the automatic segmentation is rather high, the miss rate is fairly low. Furthermore, the edge-based measure indicates that edges of the automatic and the manual segmentation coincide. However, the van could not be segmented accurately. Both region-based and edge-based distance measures return high error rates.

The second video sequence is a typical "head and shoulder" sequence. However, due to the low sampling rate the displacements of the moving person are large. Hence, we employed the multiresolution motion estimation with four pyramid levels and a local neighborhood of size $3 \times 3 \times 3$. To speed up the motion detection we employed the reliability measure provided in Section 3.2, i.e., positions with a reliability below 0.9 were rejected. The final segmentation was performed by the geodesic active contour model.

Figure 9 depicts the results of the motion estimation and segmentation for the second sequence. Our segmentation approach identifies the region of interest correctly. However, the accuracy is less than that for the taxi sequence. Especially, in areas containing strong but static edges, results from the hierarchical motion estimation blur across the moving edges, thus enlarging the segmented region. Tables 3 and 4 underline these observations. Especially the percentages of exactly matching edges ($d = 0$) are rather small.

5.3 Classification Results

Our test database [23] consists of five object classes containing *animals*, *birds*, *cars*, *people*, and *miscellaneous objects*. For each object class we collected 25 – 102 images from a clip art library. The clip arts are typical representatives of their object class with easily recognizable perspectives. The object class *people* contains the most objects (102 images). The contours of humans differ greatly in image sequences, e. g. the position of the arms and legs makes a great impact on the contour.

We applied the extended object matching algorithm on the automatically segmented cars in the Hamburg taxi sequence (see Figure 8) and in the person sequence (see Figure 9). The CSS matching was performed with the triples

Frame	F_P	F_N	F_P (%)	F_N (%)
25	759	121	13.22	2.37
34	336	77	6.55	1.58
35	601	106	11.72	2.29
38	566	109	12.27	2.62

Table 3: Region-based distance for the person sequence. Columns 2, 3: false positives, false negatives. Columns 4, 5: percentage of mismatched pixels in comparison to the entire number of pixels of the manual segmentation.

Frame	avg. distance d	d = 0 (%)	d = 1 (%)	d = 2 (%)	d > 2 (%)
25	1.97	21.23	31.36	24.20	23.21
34	1.20	25.40	41.98	23.53	9.10
35	2.17	12.81	38.15	26.16	22.89
38	2.04	12.61	38.71	24.93	23.75

Table 4: Edge-based distance for the person sequence. Column 2: average edge pixel distance. Colums 3-6: percentages of the distances 0,1,2,3..n.

Taxi sequence	**Good matches**	**Bad matches**	**Rejected frames**
car (left)	Cars 92%	0%	8%
taxi (center)	Cars 68%	Misc 8%	24%
van (right)	Cars 29%	Animals 39% People 32%	0%

Table 5: Results of the automatically segmented objects in the Taxi sequence matched to the objects in the database.

(position, height, width) for each peak in the CSS image.

Table 5 shows the result of the Hamburg taxi sequence. The perspective and segmentation of the car (left object) is best suited for recognition. At only 68% the taxi (center object) cannot be recognized reliably. The van cannot be recognized by the application.

The last row in Figure 8 shows the four best matches of the car (left object) in frame 12. The perspective of the car does not change, so the other frames show similar results. Figure 9 depicts classification results for the person sequence. The best matches for the frames 25, 33, 34, and 38 are displayed in the last row.

6. CONCLUSIONS

We presented an approach to the segmentation and classification of video objects. In the motion segmentation step we integrated the 3D structure tensor into a geodesic active contour model. While the structure tensor is able to estimate motion reliably in the presence of background noise, the active contour groups neighboring regions and closes holes and gaps. The level set-based implementation allows the simultaneous detection of multiple objects. To account for large displacements that cannot be handled by the standard structure tensor, we developed a new multiresolution tensor-based algorithm.

A contour-based video object classification system was presented as an application. The robustness of the curvature scale space method allows correct classification even in the presence of segmentation errors. We provided various experimental results. While the results for the segmentation algorithm driven by the standard tensor are very encouraging, the segmentation obtained in conjunction with the multiresolution algorithm are less accurate. Nevertheless, the classification algorithm was able to calculate reasonable categorizations.

There are, however, several areas that require further development. First, the segmentation performance of the multiresolution approach has to be improved. Second, to provide a complete segmentation module, it is necessary to integrate a tracking component.

7. ACKNOWLEDGMENTS

The authors would like to thank Changick Kim, University of Washington, for the provision of the “person sequence” on his website.

8. REFERENCES

[1] S. Abbasi and F. Mokhtarian. Shape similarity retrieval under affine transform: Application to multi-view object representation and recognition. In *Proc. International Conference on Computer Vision*, pages 450–455. IEEE, 1999.

[2] S. Abbasi, F. Mokhtarian, and J. Kittler. Enhancing css-based shape retrieval for objects with shallow concavities. *Image and Vision Computing*, 18(3):199–211, 2000.

[3] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal on Computer Vision*, 12(1):43–77, 1994.

[4] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Computing Surveys*, 27(3):433–467, 1995.

[5] J. Bigün, G. H. Granlund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:775–790, 1991.

[6] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):579–698, 1986.

[7] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[8] L. da Fontoura Costa and R. M. Cesar, Jr. *Shape Analysis and Classification*. CRC Press, Boca Raton, FL, September 2000.

[9] H. Haußecker and B. Jähne. A tensor approach for precise computation of dense displacement vector fields. In E. Paulus and F. Wahl, editors, *Proceedings Mustererkennung*, Informatik Aktuell, pages 199–208, Berlin, 1997. Springer.

[10] ISO/IEC 14496-2. Information technology – coding of audio-visual objects – part 2: Visual, 1999.

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.

[12] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi. Conformal curvature flows: from phase transitions to active vision. *Archive of Rational Mechanics and Analysis*, 134:275–301, 1996.

[13] C. Kim and J.-N. Hwang. A fast and robust moving object segmentation in video sequences. In *IEEE International Conference on Image Processing (Kobe, Japan)*, pages 131–134, 1999.

[14] C. Kim and J.-N. Hwang. An integrated scheme for object-based video abstraction. In *Proceedings ACM Multimedia*, 2000.

[15] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, August 1998.

[16] R. Mech and M. Wollborn. A noise robust method for segmentation of moving objects in video sequences. In *International Conference on Acoustics, Speech and Signal Processing (Munich, Germany)*, pages 2657–2660, 1997.

[17] T. Meier and K. N. Ngan. Extraction of moving objects for content-based video coding. In *Proceedings of SPIE, Visual Communications and Image Processing*, volume 3653, pages 1178–1189, 1999.

[18] A. Mitiche and P. Bouthemy. Computation and analysis of image motion: A synopsis of current problems and methods. *International Journal of Computer Vision*, 19(1):29–55, 1996.

[19] F. Mokhtarian, S. Abbasi, and J. Kittler. Efficient and robust retrieval by shape content through curvature scale space. In *Proc. International Workshop on Image DataBases and MultiMedia Search*, pages 35–42, 1996.

[20] F. Mokhtarian, S. Abbasi, and J. Kittler. Robust and efficient shape indexing through curvature scale space. In *British Machine Vision Conference*, 1996.

[21] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, March 2000.

[22] T. Pavlidis. Review of algorithms for shape analysis. *Computer Graphics and Image Processing*, 7(2):243–258, April 1978.

[23] S. Richter, G. Kühne, and O. Schuster. Contour-based classification of video objects. In *Proceedings of SPIE, Storage and Retrieval for Media Databases*, volume 4315, pages 608–618, Bellingham, Washington, January 2001. SPIE.

[24] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

[25] L. Torres and E. J. Delp. New trends in image and video compression. In *X European Signal Processing Conference*, September 2000.

[26] S. Ullman. *High-level Vision: Object Recognition and Visual Cognition*. MIT Press, Cambridge, MA, 1996.

[27] J. Weickert. Coherence-enhancing diffusion of colour images. *Image and Vision Computing*, 17:201–212, 1999.

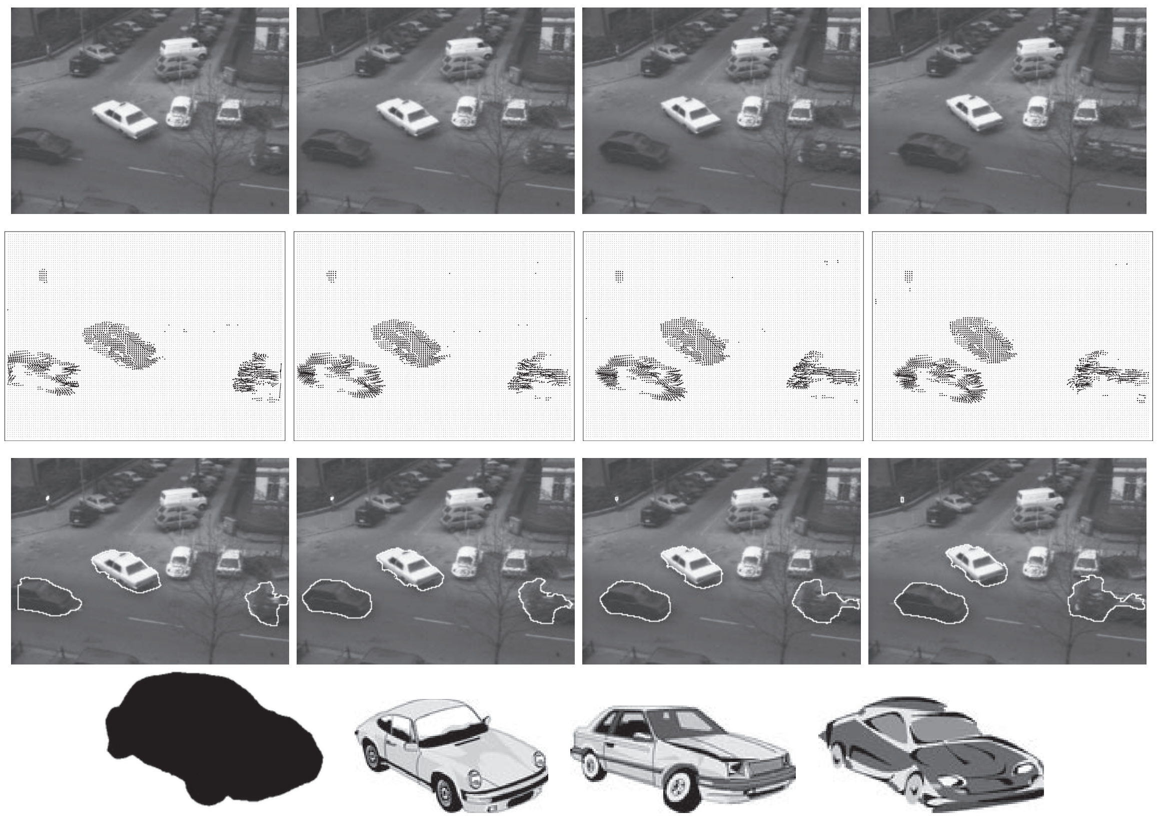

Figure 8: Segmentation and classification of the taxi sequence. Rows from top to bottom: (a) frames 4, 8, 12, 15, (b) motion estimation with structure tensor, (c) tensor-driven active contour with contour refinement, (d) classification of the car on the left in frame 12, the four top matches are displayed.

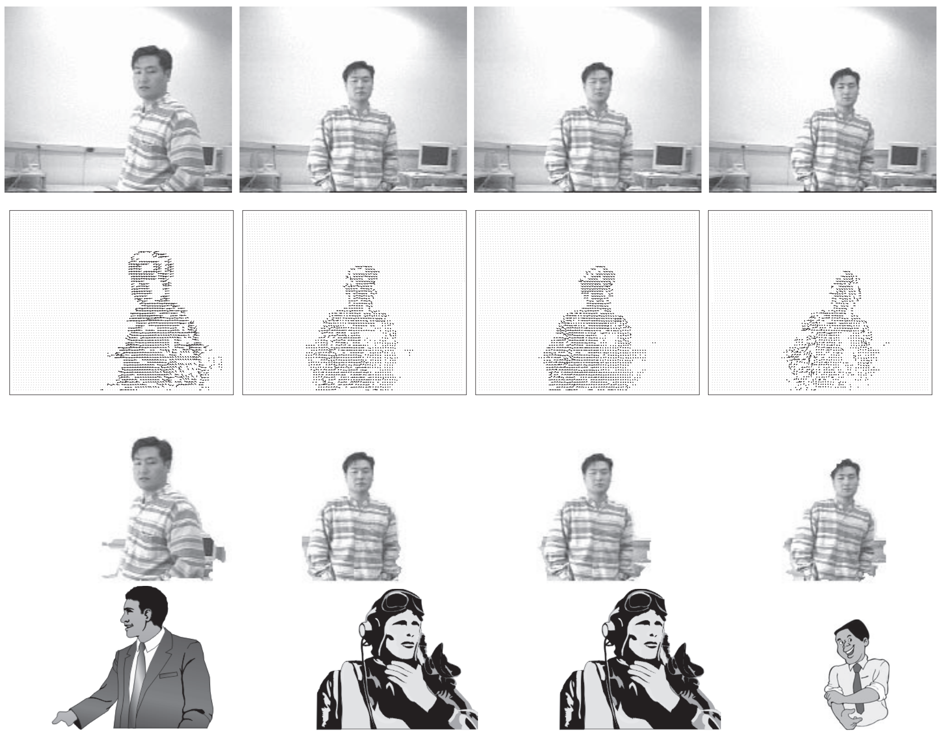

Figure 9: Segmentation and classification of the person sequence. Rows from top to bottom: (a) frames 25, 33, 34, 38, (b) motion estimation with hierarchical structure tensor, (c) tensor-driven active contour, (d) classification of the frames 25, 33, 34, 38, the top match is displayed.