# MOTION DETECTION BASED ON CONTOUR STRINGS

*Michael Kellner and Tobias Hanning*

FORWISS, University of Passau, Innstrasse 29, D-94032 Passau, Germany
{kellner, hanning}@forwiss.uni-passau.de

## ABSTRACT

A novel method for motion detection is presented which combines three classical approaches. A modified image subtraction approach suppressing local lighting changes is used to determine contour point strings of moving objects. Thus the pixels processed by the proposed algorithms are restricted to contour points which increases computational efficiency. Initially we process the scene without moving objects to determine all contours appearing in the background image. Then the contour strings from the input image are compared with the background contours to separate the moving sub-contours from the static ones. The proposed algorithm is stable for illumination changes and can be computed in real-time.

## 1. INTRODUCTION

The recognition of the situation in a scene with a video camera plays an important role in surveillance of the working area of service robots and in video surveillance of disabled or handicapped people in clinics [1]. The proposed article arose out of the project FORSIP[1] which is engaged in the detection of the environment and situation in smart homes, and in particular the localization and tracking of non-rigid objects over time.

Initially the situation in the considered scene is reconstructed by extracting the appearing contours. Step by step moving non-rigid and rigid objects are tracked over time. Nevertheless the location of objects which do not move for a short time can be specified to continue tracking at the same position if motion occurs again.

The method proposed by this article is based on a sub-pixel accurate extraction of contour points and builds the fundament to assign grouped contours to objects to have the ability to track them over time.

The set-up consists of a stationary gray-value CCD-camera which observes an appropriate scene. Geometric image registration, i.e. aligning several images into the same coordinate frame (cf. [2]), is omitted in this paper due to the stationary camera. A gray-value image is defined by $f : D \rightarrow \{0, \ldots, 2^b-1\}$ with $D := \{0, \ldots, n\} \times \{0, \ldots, m\}$ (usually $b = 8$). Define $c_f : D \rightarrow \mathbb{R}^2 \cup \{\emptyset\}$ as the (sub-pixel accurate) contour point extractor on the image $f$. Set $c_f(p) := \emptyset$ in the case that the actual contour curve (sub-pixel accurate) does not intersect with the pixel $p$.

To achieve the sub-pixel accurate values of $c_f(p)$ the Directional Derivative Edge Finder by Haralick and Shapiro [3](vol. 1) can be applied to support the linking process in section 4.

First the set of all contour point strings $C_b$ is extracted from a given background image $f_b := f_0$ at time $t_0$. Then the set of all contour point strings $C_{f_i}$ is constructed from the input image $f_i$ at time $t_i > t_0$. For every image $f_i$ the proposed algorithm determines the subset $C' \subseteq C_{f_i}$ which contains all the contours of the objects moving in the given scene in the period $[t_{i-1}, t_i]$.

Section 2 gives a general overview of the well-known change detection techniques. To restrict the areas of interest of the moving regions, a modified image subtraction process is presented in section 3. In section 4 the contour point extractor is applied to moving regions to obtain contour strings. The resulting superset of foreground contours of interest will be reduced by comparing it to background contours which is described in section 5 in detail. In section 6 the results of the proposed method are discussed.

## 2. RELATED WORK

The most common approaches to motion detection systems retrieve a set of images of the same scene at several different times. Their processing techniques can be classified into two different methods, which provide a set of pixels that have changed significantly over time and mostly result in a so-called "change mask": One class only considers two consecutive images to make the change decision more or less sophisticated including significance and hypothesis tests as well as predictive and shading models. The other class of algorithms establishes a model of the background of the scene. A survey of these image change detection algorithms is given in [2] whose classes of methods are further discussed in the following.

Early change detection methods on consecutive images were

[1] http://www.forsip.de

0-7803-8554-3/04/$20.00 ©2004 IEEE.

based on simple signed image subtraction. A global or local empirically chosen threshold delivers the change-mask. In doing so the objects have to move continuously because the foreground cannot be detected in case of no motion between different frames. Illumination changes also raise problems with this approach since they are detected as movement, too [4]. The optical flow methods extend this approach, but they are computationally expensive. Also there are many contour based methods which track corresponding contours in consecutive images by applying the optical flow approach [5]. The active contour model extends the classical optical flow process, and is mainly used for contour extraction. For each contour point the correspondence relations are investigated, and the flow vectors are estimated [6]. At this point most systems struggle with the aperture and the correspondence problem, while matching of incidental contours in consecutive images. Some systems solve this problem by matching feature points (corners or dominant points) [7], while other methods compare curvature of contours [8].

The second common method for object tracking is to separate foreground from an initially known background. Since the background is exposed to permanent changes (eg. illumination changes and shadow), it has to be updated periodically. By considering the gray-value difference between the background and foreground image over time, the background can be updated appropriately [9]. An approach from Karmann and v. Brandt [10] considers the gray-value in addition to the first derivation for every pixel, and defines a signal processing system on it. This gives rise to problems during the processing of diffused contours of shadows. Another typical approach of many researchers is to model the background to be a mixture of several Gaussians, i.e. the probability of observing the intensity of a pixel at a time is the weighted sum of many Gaussian distributions [11].

The presented work combines both approaches addressed above, but it is solely restricted to contours. Thus the advantages of the methods mentioned above are combined, and the disadvantages are circumvented. Also the aperture and correspondence problem can be solved by subgraph matching ([12]) or by solving the "consistent-labeling problem" firstly stated in [3](vol.2).

## 3. MOVEMENT DETECTION

We concentrate on moving contours in the period $[t_{i-1}, t_i]$. Every contour appearing in the image $f_i$ which is not moving after $t_{i-1}$, has either a correspondence in the background or has moved before $t_{i-1}$.

Given two consecutive images $f_{i-1}$ and $f_i$, a subset $D' \subset D$ is looked for, such that $p \in D'$ satisfies the following:

Let $h$ be the half of the window width and $p_{lu} := (p_1 - h, p_2 - h)$, $p_{ll} := (p_1 - h, p_2 + h)$, $p_{ru} := (p_1 + h, p_2 - h)$ and $p_{rl} := (p_1 + h, p_2 + h)$ the corners of a window around $p$. Define $m_i : D \to \mathbb{R}$ by $m_i(p) := f_i(p_{lu}) + f_i(p_{ll}) + f_i(p_{ru}) + f_i(p_{rl})$. Then

$$D'_s := \left\{ p \in D' \mid \left| f_i(p) \frac{m_{i-1}(p)}{m_i(p)} - f_{i-1}(p) \right| > s \right\} \text{ (Eq.1)}$$

is well defined. The pixels in $D'_s$ are called DiffCorner pixels. The adjustment of the gray-value by computing the mean value over the corner points results in a suppression of uniform brightness differences in a short period which can arise from flickering light or shadows. The usage of a threshold $s$ affects the binarization of the calculated values. In Figure 1 the comparison with the standard image subtraction approach is shown.

A disadvantage of this approach can be noticed in homogeneously gray-valued moving regions moving in front of a homogeneous background as well. Due to the gray-value adjustment the difference in the definition of $D'$ becomes small (Eq.1). Therefore there are moving pixels which cannot be detected by the DiffCorner algorithm. However the standard image subtraction method determines these moving pixels correctly. They can be noticed in Figure 1 on the upper body of the depicted person. For most of those points the magnitude of the gradient is small enough since in these regions approximately constant gray-values were assumed. This disadvantage is acceptable since the set $D'$ is only used as the set of start pixels to link contour points (c.f. section 4), and moreover small magnitudes of contours appear rarely.

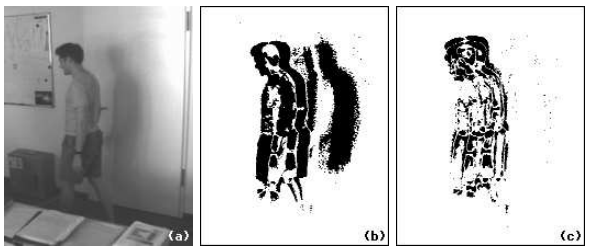


**Fig. 1**.
(a) Input image of a moving person, (b) standard difference image and (c) DiffCorner algorithm, $h = 5, s = 12$

## 4. CONTOUR STRINGS

Starting from an arbitrary point $p_0 \in D'$ the contour points are now linked by tracking neighbored contour pixels.

An element $S$ of the set of all contour strings $\mathbf{C}_f$ of an image $f$ has the following form: $S := (c_f(p_0), \ldots, c_f(p_{n-1}))$ with $p_{i+1} \in N_8(p_i)$ (the 8-neighborship of $p$ is $N_8(p)$). To extract the contour point strings the algorithm CONTOUR POINT LINKER is sketched which produces the set $C_f \in \mathbf{C}_f$ of the resulting strings. The set $\tilde{C}_f := \{p \in D | (..., c_f(p), ...) \in C_f\}$ denotes the set of all pixels linked into contours in the image $f$. In Figure 2 the gray deposited pixels represent the extracted DiffCorner pixels $D'$. The crosses mark the exact sub-pixel position $c_f(p)$ for pixel extracted by the contour extractor, and the arrows between the extracted points show the linking order. Note that $s_{e_0+1}$ is

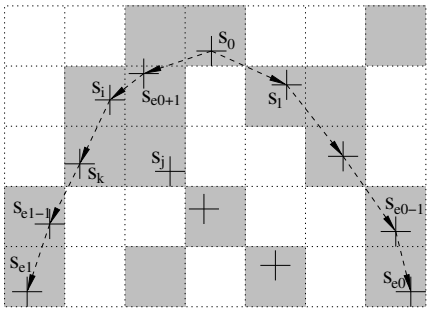


**Fig. 2**. A contour string is depicted with the respective numbering. The gray deposited areas are the pixels which belong to the set $D'$.

linked to $s_i$ and not to $s_j$. With the choice of a $p \in D'$ (cf. line 3) it is not guaranteed that $p$ is an end point of the contour (in the algorithm an end point is denoted as $s_{e_{\{0,1\}}}$). Therefore we have to iterate along the contour twice. In Figure 2 the right branch is linked first, then the left branch. This is achieved by a for-loop in the algorithm (line 5). Before the second iteration starts, $p$ is reset to $s_0$ again (line 11). Therefore the correct contour point sequence is given in line 13. In line 7 the successor of $p$ is determined by fulfilling the condition $N(p)$ (l. 2). In Figure 2 the successor $b$ with $c_f(b) = s_i$ is chosen for the pixel $p$ with $c_f(p) = s_{e_0+1}$ since $dist(s_{e_0+1}, s_i) < dist(s_{e_0+1}, s_j)$. It should be noted that the sub-pixel accurate extraction has a curvature preserving influence during contour linking.

**(CONTOUR POINT LINKER)**
**Input:** $f$ image, $D'$ start pixels, $c_f$ contour points
**Output:** $C_f \subset \mathbf{C}_f$ set of contour point strings

1: $\tilde{C}_f := \emptyset, C_f := \emptyset$
2: $N(p) := \{p' \in N_8(p) \mid c_f(p') \neq \emptyset$ **and** $p' \notin \tilde{C}_f\}$
3: **for** $p \in D'$ **with** $c_f(p) \neq \emptyset$ **and** $p \notin \tilde{C}_f$ **do**
4: $\quad i := 0; t := p; \tilde{C}_f := \tilde{C}_f \cup \{p\}; s_0 := c_f(p)$
5: $\quad$ **for** $branch \in \{0, 1\}$ /* right, left */ **do**
6: $\quad\quad$ **while** $N(p) \neq \emptyset$ **do**
7: $\quad\quad\quad b := \arg\min_{p' \in N(p)} dist(c(p'), c(p))$
8: $\quad\quad\quad \tilde{C}_f := \tilde{C}_f \cup \{b\}; i := i + 1; s_i := c_f(b)$
9: $\quad\quad\quad p := b$
10: $\quad\quad$ **end while**
11: $\quad\quad e_{branch} := i; p := s_0$
12: $\quad$ **end for**/* branch */
13: $\quad$ add $(s_{e_1}, s_{e_1-1}, \ldots, s_{e_0+1}, s_0, s_1, \ldots s_{e_0})$ to $C_f$
14: **end for**

To create the background contour set $C_b$ we have to use the start pixel set $D$ instead of $D'$ which results in a higher computational runtime at the initialization phase.

## 5. SUB-CONTOUR CORRESPONDENCES

Based on the extracted contours $C_b \subset \mathbf{C}_b$ of the background image and the contours $C_f \subset \mathbf{C}_f$ determined by the starting point set $D'$ a procedure which builds a contour string set $C'_f \subseteq C_f$ should satisfy the following property:

Let $\tilde{C}_{b,f}$ be the set of not isolated contour pixels appearing in the background and foreground defined by

$$\tilde{C}_{b,f} := \{s \in C \subseteq \tilde{C}_b \cap \tilde{C}_f \mid C \text{ not isolated in } \tilde{C}_b \cap \tilde{C}_f\} \quad \text{(Eq.2)}$$

then $C'_f$ has to fulfill: $\tilde{C'}_f \subseteq \tilde{C}_f \setminus \tilde{C}_{b,f}$

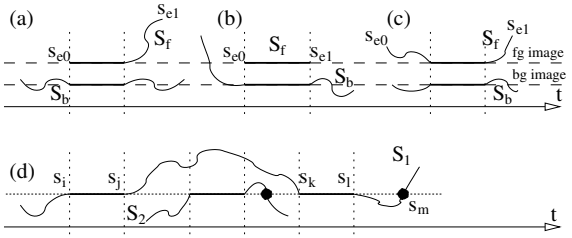


**Fig. 3**. The axis $t$ represents the parametrization of the curve. (a)-(c): The straight parallel curve parts demonstrate the corresponding parts of the respective contours, while the snaky ends mean no correspondence. In Fig. (d) two foreground contours lie above the corresponding background contour.

This issue is described by considering the four examples of Figure 3 in detail: Figures 3(a)-(c) depict three cases, where one end point (a), two end points (b), or no end point (c) of a contour lie on the corresponding string. Let $S_b \in C_b$ be a background contour and $S_f \in C_f$ a partly corresponding contour in the input image, then for $s_{e_0}, s_{e_1} \in S_f$ satisfies *(a)* $s_{e_0} \in S_b, s_{e_1} \notin S_b$ or *(b)* $s_{e_0} \in S_b, s_{e_1} \in S_b$ or *(c)* $s_{e_0} \notin S_b, s_{e_1} \notin S_b$, where $s_i \in S_b$ is a short notation for:
$\exists p \in D_f : c_f(p) = s_i$ and $(\ldots, c_b(p), \ldots) = S_b$.
Figure 3(d) depicts the case that two contours $S_1, S_2 \in C_f$ overlay a background contour $S_b \in C_b$, namely $S_1$ actually twice. The points plotted in black (also overlays of $S_b$) are isolated points (cf. (Eq.2)) and should not miss the set $C'_f$. Therefore the set $\tilde{C}_{b,f}$ consists of the points of the corresponding parts without the isolated points.
$(s_0, \ldots, s_i, \ldots, s_j, \ldots, s_k, \ldots s_l, \ldots, s_m, \ldots, s_{e_1}) \in C_f$
thus becomes
$(s_0, .., s_{i-1}), (s_{j+1}, .., s_{k-1}), (s_{l+1}, .., s_m, .., s_{e_1}) \in C'_f$
with $\{s_i, \ldots, s_j\}, \{s_k, \ldots, s_l\} \in \tilde{C}_b \backslash \tilde{C}_{b,f}$. This is achieved by scanning the foreground contours in $C_f$ and pruning them respectively.

## 6. RESULTS AND CONCLUSION

In Figure 4 the input image (a), the initial scene (background contours $C_b$) (b), the moving contours $C_f$ (c) and the resulting foreground sub-contours $C'_f$ are depicted (d). In the contour extraction process the contours with a significance (here: average magnitude of the gradient) below a threshold have been omitted. Without this filtering many short contour parts may appear due to camera noise which will often appear as background contours, too.
The proposed algorithm is weak in areas where a contour appears in the background which unfortunately corresponds

to a quite different one in the input image. This leads to a break-up of the contour, eg. the one surrounding the persons head in Figure 4 (first row). For such problems the method to determine the sub-contour correspondences can be refined (tests of multiple criteria, eg. comparison of pixel gray-values or extending the background contour coding by a mixture of Gaussian approach [11]), or a movement estimator can be applied. Also contours in temporarily non-moving regions (eg. feet in the first image row) should stay in memory to complete the set $C'_f$.

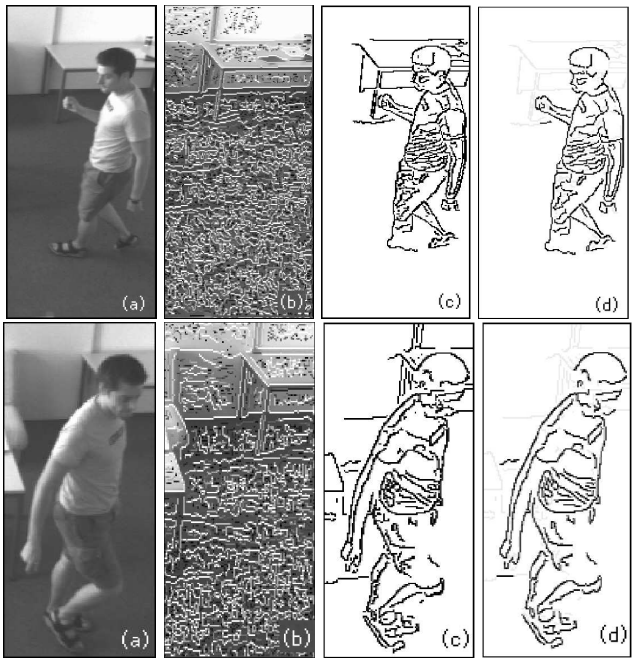


**Fig. 4**. (a) Input image, (b) background contours $C_b$ drawn on the background image, (c) foreground contour strings $C_f$, (d) foreground sub-contours $C'_f$

As a conclusion it can be noted that the proposed work reduces the problems of local lighting changes and smooth shadow described in section 2 since the DiffCorner algorithm compensates the shadow during detection of moving points and contours with a weak significance are omitted at all. The intensity adjustment of the images as a preprocessing step commonly applied in many change detection algorithms can be omitted here since the contour extraction is based on the spatial derivatives on every image.

The proposed method is computationally efficient since by restricting the computation with the starting set $D'$ only the moving regions in the image are processed.

A frame rate of 15 fps has been reached with our implementation including a sub-pixel accurate contour point extractor. Applying an extractor with discrete derivatives (eg. [13]) increases the frame rate up to 20 fps. (image size: 640x480 pixels, CPU: Intel Pentium 2400MHz).

## 7. REFERENCES


[1] M. Wolf, P. Mössner, T. Hanning, G. Pisinger, P. Weierich, and H. Niemann, "INSERVUM– a retargetable, intelligent, video-based observation system," *IROS '98*, vol. 3, pp. 1840–1846, 1998.

[2] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Transactions on Image Processing*, April 2004.

[3] Robert M. Haralick and Linda G. Shapiro, *Computer and Robot Vision*, vol. 1/2, Addison-Wesley, 1992/1993.

[4] Maylor K. Leung and Yee-Hong Yang, "Human body motion segmentation in a complex scene.," *Pattern Recognition*, vol. 20, no. 1, pp. 55–64, 1987.

[5] Hans-Hellmut Nagel, "On the estimation of optical flow: Relations between different approaches and some new results.," *Artificial Intelligence*, vol. 33, no. 3, pp. 299–324, 1987.

[6] Jong-Seung Park and Joon Hee Han, "A curvature-based approach to contour motion estimation," in *ICCV*, 1998, pp. 1018–1023.

[7] Jürgen Haas, *Echtzeit-Korrespondenzprobleme in Bildsequenzen und Subpixelgenauigkeit*, Ph.D. thesis, Universität Passau, 2000.

[8] Daniel Freedman, "Effective tracking through tree-search," *IEEE PAMI*, vol. 25, no. 5, pp. 604–615, 2003.

[9] Brofferio, L. Carnimeo, D. Comunale, and G. Mastronardi, "A background updating algorithm for moving object scenes," *Time-Varying Image Processing and Moving Object Recognition 2*, pp. 297–307, 1990.

[10] K. Karmann and A. von Brandt, "Moving object recognition using an adaptive background memory," *Time-Varying Image Processing and Moving Object Recognition 2*, pp. 289–296, 1990.

[11] Chris Stauffer and W. Eric L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE PAMI*, vol. 22, no. 8, pp. 747–757, 2000.

[12] C. Schellewald and C. Schnörr, "Subgraph matching with semidefinite programming," in *Electronic Notes in Discrete Mathematics*. 2003, vol. 12, Elsevier.

[13] J. Canny, "A computational approach to edge detection," *IEEE PAMI*, vol. 8, no. 6, pp. 679–698, 1986.