2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

General Object Tracking with a Component-based Target Descriptor

Simone Frintrop

Abstract— **In this paper, we present a component-based visual object tracker for mobile platforms. The core of the technique is a component-based descriptor that captures the structure and appearance of a target in a flexible way. This descriptor can be learned quickly from a single training image and is easily adaptable to different objects. The descriptor is integrated into the observation model of a visual tracker based on the well known Condensation algorithm. We show that the approach is applicable to a large variety of objects and in different environments with cluttered backgrounds and a moving camera. The method is robust to illumination and viewpoint changes and applicable to indoor as well as outdoor scenes.**

I. INTRODUCTION

Object tracking is an important task in machine vision as well as in mobile robotics. Applications include surveillance systems, mobile robots that guide or follow people, or human-robot interaction in which a robot interacts with a human and both have to concentrate on the same objects.

Many good approaches for object tracking have been proposed during the last years (see survey in [1]). However, the methods that are applicable for a certain task vary largely depending on requirements and setting. If the type of object is known in advance, model-based trackers may be applied. In these approaches, a model of the object is learned offline, usually from a large set of training images which show the object from different viewpoints and in different poses [2], [3]. These methods are especially well-suited for specialized tracking tasks such as person or face tracking. In some applications however, the object of interest is not known in advance, e.g., if a user shows an object to the system which shall be able to immediately capture the appearance of the object and track it. A long training phase is inacceptable in such cases, online learning methods are required.

In systems with a static camera, it is possible to apply methods like background subtraction [4]. For statistical investigations that do not require immediate response, like e.g. counting people, it is possible to process the data offline which extends the range of applicable algorithms considerably.

On the other hand, systems which shall be applied on a mobile platform usually have to operate in real-time and have to deal with more difficult settings. The background changes, illumination conditions vary, and platforms are often equipped with low-resolution cameras. Such conditions require robust and flexible tracking mechanisms. Mostly, feature-based tracking approaches are applied in such areas. They track an object based on simple features such as color cues or corners. An example is the Mean Shift algorithm [5] which classifies objects according to a color distribution or the CamShift algorithm which is based on the Mean Shift approach [6]. Other groups integrate color histograms into a particle tracker [7], [8] or combine a color model with a template tracker [9]. In previous work, we have used a cognitive observation model for visual tracking that was based on features inspired by human visual perception [10], [11]. Several ideas from this work have been integrated into the current approach. Over the last years, techniques which use interest points, like colored Harris corners [12] or SIFT features [13] for object tracking have been introduced. Note that these approaches usually rely on textured objects and a certain image resolution and quality to work well.

For feature-based tracking, it is especially important to detect discriminative features that distinguish the target well from the background. However, the discriminability of different parts of the object may differ strongly depending on the appearance of object and background. If a person wears a shirt in a color similar to the background, it has a low discriminability while the trousers on the other hand might have a high discriminability. To consider the different discriminability of parts, Beuter et al. train a top-down attention model to learn the face and the torso of a person separately [14]. Pérez et al [7], [8] determine different color histograms for different, rigidly linked parts of the target. Similarly, Adam et al. represent the target by a rigid layout of vertical and horizonal patches [15]. All of these approaches define a rigid layout of the parts in advance. In contrast to this, we suggest to automatically detect the different parts of a target in a flexible and object-dependent way.

In this paper, we present a component-based approach to visual tracking that is able to automatically detect the most discriminative parts of a target and to quickly learn its appearance from a single frame. Depending on the appearance of the object, the system determines a flexible number of components, each representing a discriminative part with respect to a certain feature channel. The resulting components form a target template that is used in the following frames to detect the most likely target position. A similarity measure determines the similarity between the target template and image regions in the following frames. Instead of computing the similarity for each pixel, we employ the component-based approach within a CONDENSATION-based person tracker [16]. For this purpose, the similarity measure is converted to a likelihood function that is used as observation model within the particle filter.

This approach leads to a robust and flexible tracker that is quickly applicable to track arbitrary objects in unknown

The author is with the Institute of Computer Science III, University of Bonn, 53117 Bonn, Germany frintrop@iai.uni-bonn.de

environments. Currently, the system works on camera data from a hand-held camera. Thus, it provides all conditions which are necessary to use it on a mobile robot: it is real-time capable and it is able to deal with background changes, viewpoint changes and varying illuminations.

We evaluated the approach in different settings and compared it to other color-based tracking methods. We tested the ability of the methods to deal with illumination changes, scale changes, occlusions, motion blur, background changes and more. It shows that on average the performance of the component-based tracking outperforms the other approaches considerably.

In the following, we first introduce the component-based descriptor (Sec. II). In Section III, we explain the visual tracking system and Section IV presents experimental results. We finally conclude in Section V.

II. A Multi-Component Target Descriptor

In this section, we introduce the multi-component descriptor that represents a target object. The descriptor consists of a collection of components that have a strong contrast within a certain feature dimension. These regions are automatically and object-dependently extracted from the target region. The components are color-based and the computation is motivated from the cognitive perception model VOCUS [17] that mimics human early visual processing.

Determining the multi-component descriptor consists of two steps. First, six intensity and color feature maps are computed (sec. II-A), second, components are automatically determined within the feature maps and combined to form the descriptor (sec. II-B). Finally, we describe how the target descriptor is matched to a region in a different frame to test if the target is present or not (sec. II-C).

A. Feature map computations

In this section, we describe how six intensity and color maps are computed as a basis for the component-based descriptor. An overview is displayed in Fig. 1.

First, the input image is converted to an image in the CIELAB color space (also $L^*a^*b^*$), smoothed with a Gaussian filter and subsampled twice to reduce the influence of noise. The resulting image is called I_{lab}. CIELAB has the dimension L for lightness and a and b for the color-opponent dimensions; it is perceptually uniform, i.e., a change of a certain amount in a color value is perceived as a change of about the same amount in human visual perception. Each of the 6 ends of the axes that confine the color space serve as a prototype color, resulting in two intensity prototypes for white and black and four color prototypes for red, green, blue, and yellow (cf. Fig. 1, top right).

Then, the computation of feature maps is started. We treat intensity and color computations separately since this results in a higher illumination invariance. The intensity computations can be performed directly from the L channel I_l. The color computations are performed on the color layer I_{ab} spanned by a and b. Now, we determine four color

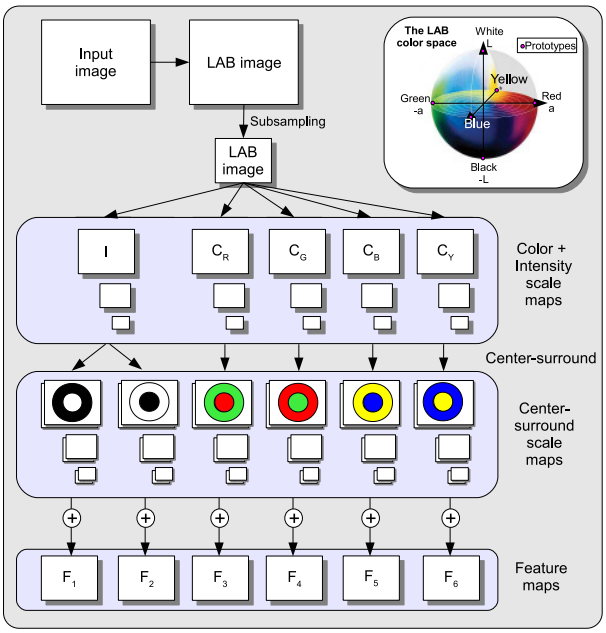

Fig. 1. The feature computations: from an input image, 6 feature maps are computed, showing bright-dark, dark-bright, red-green, green-red, blue-yellow, and yellow-blue contrasts.

specific maps C_i that represent the four colors red, green, blue and yellow.

For each of the color maps C_i, there is one prototype color P_i (cf. Fig. 1, top right) and each pixel $C_i(x, y)$ in a color map stores the Euclidean distance to the corresponding prototype color P_i:

$$C_i(x, y) = V_{max} - ||I_{ab}(x, y) - P_i|| \qquad i \in \{1, ..., 4\}, \quad (1)$$

where $V_{max} = 255$ is the maximal pixel value and the prototypes P_i are the ends of the a and b axes with coordinates $(0, 127), (127, 0), (255, 127), (127, 255)$ in an 8-bit I_{ab}.

Next, image pyramids with 3 levels are determined from I_l and C_i. This enables flexibility to scale changes. On each of these scale maps in the pyramids we perform *center-surround mechanisms*. These are filters that detect image contrasts between a center c and a surround region s, similar to ganglion cells in the human brain. Applied to our scale maps, the filters detect intensity and color contrasts. On the color maps, the filters react especially strong to red-green, green-red, blue-yellow, and yellow-blue contrasts. We use surround regions of two different sizes, resulting in six center-surround maps $S_{i,j}, j \in \{1, ..., 6\}$ for each color/intensity (details in [17]). Note that center surround applied to the intensity scale maps detects only bright-dark contrast. To additionally determine dark-bright contrasts, we compute the opposite difference $s - c$. To speed up processing, all center-surround filters are computed with integral images [18].

Finally, we sum up the 36 center-surround scale maps to obtain 6 feature maps $F_i = \sum_{j=1}^{6} S_{i,j}$. The feature maps for some example images are displayed in Fig. 2.

Fig. 2. An example image and the corresponding feature maps F_i.

B. Determining a target template and descriptor

Now, we determine a component-based template from the feature maps and derive a descriptor from the template. A component is a peak in one of the feature maps within the target region $\vec{R^*} = (x^*, y^*, w^*, h^*)$, where x^*, y^* denote the position and w^*, h^* the width and height of the region. The peaks are detected by first detecting local intensity maxima and then segmenting the region around the maxima with region growing. For easier computations, the regions are approximated by rectangular bounding boxes that we call $m_{i,j} = (x_{m_{i,j}}, y_{m_{i,j}}, w_{m_{i,j}}, h_{m_{i,j}})$, where i denotes the feature map and j the different maxima in a map. Hereby, the number of components per map is flexible and depends on the appearance of the object. Additionally, we add the whole target region as one of the $m_{i,j}$ to make the descriptor more robust.

The positions of the regions $m_{i,j}$ are stored relative to the center of $\vec{R^*}$ and represent a template $\vec{M_{R^*}} = \{m_{i,j} | i \in \{1,..,6\}, j \in \{1,..,l_i\}\}$, where l_i is the number of components detected in feature map F_i (cf. Fig. 3, left). Now, we derive a descriptor vector from the $m_{i,j}$. For each $m_{i,j}$, we compute the ratio of the mean intensity value within $m_{i,j}$ and the mean value of the background:

$$\rho_{i,j} = \frac{mean(m_{i,j})}{mean(F_i \backslash m_{i,j})} \tag{2}$$

The mean is computed with integral images, to speed up processing and enable constant computation times for each region, independent of the size of the region. Thus, the target descriptor that we obtain is $\vec{d^*} = \{\rho_{i,j} | i \in \{1,..,6\}, j \in \{1,..,l_i\}\}$.

C. Match descriptor to image region

In order to match the target descriptor $\vec{d^*}$ to an image region $\vec{R'}$ of arbitrary size and dimension, we first determine the factors f_w and f_h that represent the difference in size between the target region $\vec{R^*}$ and $\vec{R'}$: $f_w = R'_w / R^*_w, \quad f_h = R'_h / R^*_h$, where R'_w, R^*_w denote the width and R'_h, R^*_h the height of the regions. Now, an adapted template $\vec{M_{R'}}$ is computed by extending or compressing all $m_{i,j} \in \vec{M_{R^*}}$ with f_w and f_h: $w_{m'_{i,j}} = f_w * w_{m^*_{i,j}}, \quad h_{m'_{i,j}} = f_w * h_{m^*_{i,j}}$ (cf. Fig. 3, right). $\vec{M_{R'}}$ is now used to compute a descriptor $\vec{d'}$ equivalently as in eq. 2.

Finally, the descriptors $\vec{d^*}$ and $\vec{d'}$ are matched by computing the similarity of the vectors. As similarity measure, we use the Tanimoto coefficient:

$$T(\vec{d^*}, \vec{d'}) = \frac{\vec{d^*} \cdot \vec{d'}}{||\vec{d^*}||^2 + ||\vec{d'}||^2 - \vec{d^*} \cdot \vec{d'}}. \tag{3}$$

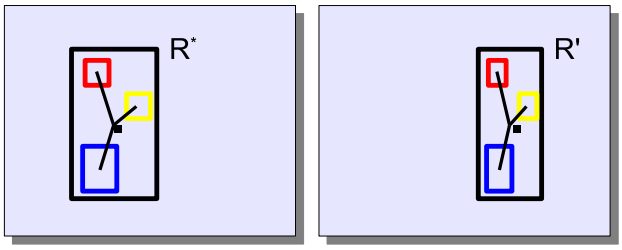

Fig. 3. Left: An illustration of the template $\vec{M_{R^*}}$ for the target region $\vec{R^*}$. The three colored rectangles denote the $m_{i,j}$. Note that each of them comes from a different feature map which is illustrated here by the different colors. Right: the template $\vec{M_{R'}}$ adapted to region $\vec{R'}$.

The Tanimoto coefficient produces values in the interval $[0, 1]$, the higher the value the higher the similarity. If the two vectors are identical, the coefficient is 1.

III. The Visual Tracking System

The tracking system uses the component-based descriptor from the previous section as observation model of a particle filter approach. It employs the standard Condensation algorithm [16] which maintains a set of weighted particles over time using a recursive procedure based on the following three steps: First, the system draws particles randomly from the particle set of the previous time step, where each particle is drawn with a probability proportional to the associated weight of the particle. Second, the particles are transformed (predicted) according to a motion model. Finally, all particles are assigned new weights according to an observation model and the object state is estimated.

Let us first introduce the notation. At each point in time $t \in \{1,..,T\}$, the particle filter recursively computes an estimate of the probability density of the object's location within the image using a set of J particles $\vec{\Phi}_t = \{\vec{\phi}^1_t, ...\vec{\phi}^J_t\}$ with

$$\vec{\phi}^j_t = (\vec{s}^j_t, \pi^j_t, \vec{d}^j_t), \quad j \in \{1, ..., J\}. \tag{4}$$

(here: $J = 500$). $\vec{s}^j_t = (x, y, v_x, v_y, w, h)$ is the state vector that specifies the particle's region with center (x, y), width w and height h – in the following, the region is also denoted as $\vec{R}^j_t = (x, y, w, h)$. The v_x and v_y components specify the current velocity of the particle in the x and y directions. Each particle additionally has a weight π^j_t determining the relevance of the particle with respect to the target, and the component-based descriptor $\vec{d}^j_t$ that describes the appearance of the particle region.

In the following, we first mention how the system is initialized (sec. III-A), second describe the motion model

(sec. III-B), and finally, specify the observation model as core of the system (sec. III-C).

A. Initialization

Before starting the tracking, the initial target region $\vec{R}^*$ has to be specified in the first frame. This can either be carried out manually or automatically using a separate detection module. We initialize manually here. Based on the initial target region $\vec{R}^*$, the component-based descriptor $\vec{d}^*$ is computed that describes the appearance of the object. The initial particle set

$$\vec{\Phi}_0 = \{(\vec{s}_0^j, \pi_0^j, \vec{d}_0^j) \,|\, j = 1, ..., J\} \tag{5}$$

is generated by randomly distributing the initial target location around the region's center (x^*, y^*). The velocity components v_x and v_y are initially set to 0 and the region dimensions of each particle are initialized with the dimensions of $\vec{R}^*$. The particle weights π_0^j are set to $1/J$.

B. Motion model

The object's motion is modeled by a simple first order autoregressive process in which the state of a particle depends only on the state of the particle in the previous frame:

$$\vec{s}_t^j = \mathbf{M} \cdot \vec{s}_{t-1}^j + \vec{Q}. \tag{6}$$

Here, $\mathbf{M}$ is a state transition matrix of a constant velocity model and $\vec{Q}$ is a random variable that denotes some white Gaussian noise. This enables a flexible adaption of position and size of the particle region as well as of its velocity. Thus the system is able to quickly react to velocity changes of the object.

C. Observation model

In visual tracking, the choice of the observation model is the most crucial step since it decides which particles will survive. It therefore has the strongest influence on the estimated position of the target. Here, we use the component-based descriptor to determine the feature description for the target and for each particle, enabling the comparison and weighting of particles.

First, we compute a descriptor $\vec{d}_t^j$ for each of the particles according to sec. II-B. That means, the target template $\vec{M}_{R^*}$ is adapted to the size of the current particle and the descriptor $\vec{d}_t^j$ is computed for the resulting template M_t^j. Then, the weight of a particle is computed based on the Tanimoto coefficient as

$$\pi_t^j = c \cdot e^{\lambda \cdot T(\vec{d}^*, \vec{d}_t^j)}. \tag{7}$$

This function prioritizes particles which are very similar to $\vec{d}^*$ by assigning an especially high weight. A value of $\lambda = 14$ has shown to be useful in our experiments. The parameter c is a normalization factor which is chosen so that $\sum_{j=1}^{J} \pi_t^j = 1$.

Finally, the current target state, including target position and size, can be estimated as a weighted average of the particles by

$$\vec{x}_t = \sum_{j=1}^{J} \pi_t^j \cdot \vec{s}_t^j. \tag{8}$$

IV. Experiments and Results

In this section, we compare three different approaches for visual object tracking. All methods use the same particle filter approach for tracking and a color-based observation model. The first approach is a standard method based on color histograms and was implemented according to [7][1]. The second approach that we call ROI tracking is a simplified version of the here presented method. It uses the same feature maps as in sec. II-A but no components. Instead, it considers the whole target region and computes a descriptor based on the ratio of the mean of the target region and the mean of the background as in eq. 2. Thus, it computes a 6-dimensional target descriptor.[2] The third approach is the here presented component-based tracking.

We test the three methods in seven different settings to illustrate different properties. In each setting, we tracked one object over a sequence of images (320×240 pixels, length of sequences: 125 – 388 frames). Examples of the settings together with the component-based descriptors are displayed in Fig. 4. The complete tracking results can be watched in a video on http://ivs.informatik.uni-bonn.de/research/tracking/.

For each estimated tracking trajectory, we computed the Euclidean distance to the real position of the target that was determined manually. Reference for the computation was the center of the object resp. the center of the estimated target position. These distances are displayed in Fig. 5. Since the distance of the estimation from the real position is not always meaningful (depending on the size of the object, the same distance might be still acceptably good or quite bad), we additionally determined whether the center of the estimation was on the target or not. This detection rate is displayed in Tab. I. The computation time varied between 69 and 90 ms per frame (av. 80 ms), depending on the complexity of the target template (on a 2.5 GHz dual core PC). This frame rate was sufficient for online tracking but a higher rate could be easily achieved by code optimization.

In the following, we describe the different settings.

A. Illumination Changes

In this example, we test the ability of the systems to deal with illumination changes. We tested a static scene in which only the illumination is varied by opening and closing the sun-blinds. It shows that the new component-based tracking is hardly effected by these changes, while both other approaches have problems. Note that the detection rate of histograms is better than the one of the ROI tracking

[1]The color histograms were implemented exactly as described in [7] (HSV color model, bin numbers $N_h = N_s = N_v = 10$), the particle filter was the same as for the other approaches (cf. sec. III) to concentrate the comparison on the observation model.

[2]We used almost the same method in [11], but we omitted the orientation features to make the approach comparable to the other methods which are purely color-based.

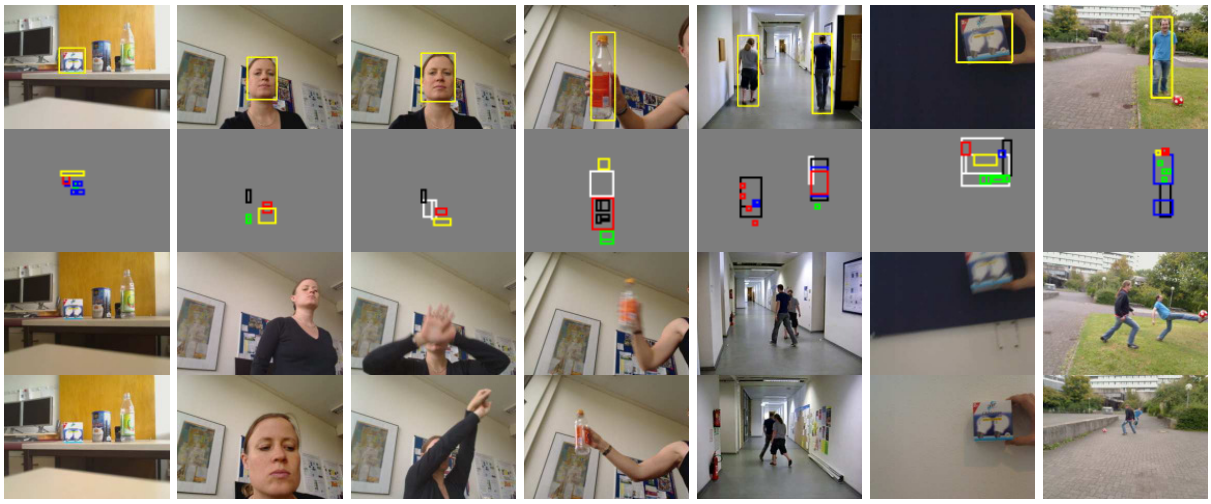

Fig. 4. The test sequences A - G. First row: the target region used for initialization (yellow rectangles). Second row: the component-based descriptors computed for the target region. Colors denote the feature map the component comes from (white: bright-dark map, black: dark-bright map, red: red-green map, ...). 3rd and 4th row: other example frames from the sequences. See also video on http://ivs.informatik.uni-bonn.de/research/tracking/

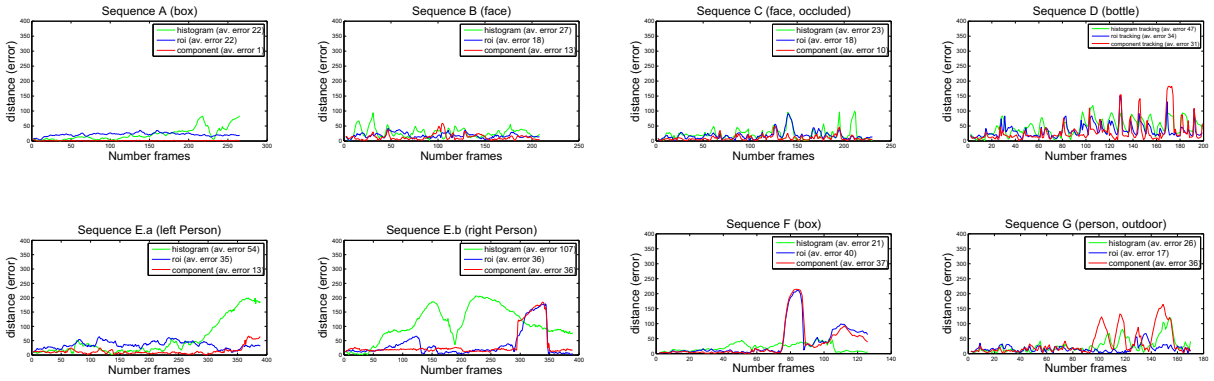

Fig. 5. Comparison of the trajectories of the three tracking methods with ground truth. The y axis shows the Euclidean distance of the center of the estimated region to the center of the real position of the target. average errors: histogram = 41, roi = 28, new component-based = 22.

while the average distance of the trajectories (cf. Fig. 5) is the same.

B. Object Motion and Scale Changes

Next, we test an object that is moving and changes strongly in scale. We use face tracking as example application. Again, the component-based method outperforms the other approaches clearly.

C. Temporal Object Occlusion

In this example, we test how the approaches deal with temporal occlusions of the object. The target is a face that is temporary occluded by hands and arms of the person. This is especially challenging since hands and face both have skin color. The fact that the results of all methods are better for this sequence than for seq. B shows that obviously the scale changes affect the methods stronger than a brief occlusion.

D. Quick Object Motion

Here, we test the ability of the methods to deal with extremely quick object motion. The object changes its direction abruptly and the motion is so quick that the object moves many pixels between consecutive frames: Rows 3 and 4 in column 4 of Fig. 4 show consecutive frames; the object position varies almost 1/3 of the image width. All methods show that the particle filter tracking needs some frames to follow the object if the motion is very fast. Thus, the target is briefly lost until the method adapts and redetects the target again. This results in relatively low detection rates of all methods (cf. Tab. I). From Fig. 5 it can be seen that the error grows quickly for each quick motion but is reduced briefly after when the target is redetected (see also video on http://ivs.informatik.uni-bonn.de/research/tracking/).

Seq.	Object	# Frames	detection rate [%]		
			Hist.	ROI	Comp. (our)
A.	Box	264	61	42	100
B.	Face	207	55	78	89
C.	Face	229	76	82	100
D.	Bottle	198	33	45	57
E.a	left Person	388	45	78	89
E.b	right Person	388	15	56	84
F.	Box	125	92	70	74
G.	Person	169	68	70	54
av.			56	65	81

TABLE I

COMPARISON OF THE THREE TRACKING METHODS BASED ON COLOR HISTOGRAMS (HIST.), SIMPLE REGION OF INTEREST TRACKING (ROI) AND THE HERE PRESENTED COMPONENT-BASED TRACKING (COMP.).

E. Moving Camera

While the previous examples have been tested with a static camera, the following three examples are recorded with a moving camera. This is considerably more challenging since it envolves illumination changes, motion blur, and background changes. The first example shows two people walking down a corridor, while the camera is following them. The persons cross their way twice. This is a typical setting for a service robot that shall follow a person and not confuse it with other people. We tested the tracking of each of the persons individually. In both cases, the component-based tracking clearly outperforms the other methods. Most difficulties has the histogram-based approach, especially when tracking the right person.

F. Moving Camera with Strongly Changing Background

The next example shows an extreme case of background change: the background changes from dark blue to white. Since the target object has similar colors (also mainly blue and white), the two tracking approaches that include the background (ROI and component-based) have some difficulties here. While including the background is usually helpful, it makes some problems in such an extreme case. We are currently working on ways to adapt the descriptor automatically to new environments.

G. Outdoor

Finally, we show an outdoor sequence that combines most of the previous challenges: the camera is moving, several objects (two people and a ball) are moving very quickly, the appearance of the target person changes strongly in scale, the shape of the person changes, e.g. when shooting the ball (cf. Fig. 4, 3rd row, right), and the illumination as well as the background change. Here, the purely color-based approaches, histogram and ROI, outperform the component-based tracking. The strong changes in shape are problematic in the latter case. We are currently working on ways to track the components of the target individually. This could help to cope with such difficulties. However, it can be seen from Fig. 5, that the approach is always able to redetect the target after some frames.

In average, the new component-based tracking has outperformed the other two methods considerably with an average error 22 and a detection rate of 81 %, compared to error 28 and detection rate 65 % for the ROI tracking and error 41 and detection rate 56 % for the histogram tracking.

V. CONCLUSION

We have presented a new approach for object tracking based on a component-based descriptor. The method grabs the appearance of an object in color and intensity together with a rough spatial layout which are quickly learned from a single training image. It can deal with different objects and settings, works in real-time, and is applicable on a moving platform. We have shown that it on average clearly outperforms other methods.

However, there is still room for improvements. We are currently working on ways to store the position of the components individually to achieve more flexibility to deformations and rotations of the objects. Additionally, we intend to adapt the target descriptor online if background and/or target appearance change strongly, as e.g. in [19].

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.

[2] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *Int'l J. of Computer Vision, Special Issue on Learning for Recognition and Recognition for Learning*, vol. 77, no. 1-3, pp. 259–289, 2008.

[3] K. Rohr, "Towards model-based recognition of human movements in image sequences," *CVGIP – Image Understanding*, vol. 59, no. 1, pp. 94–115, 1994.

[4] C. Wren, A. Azarbayejani, and A. Pentland, "Pfinder: Real-time tracking of the human body," *Trans. on PAMI*, vol. 19, no. 7, 1997.

[5] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," Proc. of CVPR, 2000.

[6] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, 1998.

[7] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," Proc. of ECCV, 2002.

[8] P. Pérez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles," *Proc. of the IEEE*, vol. 92, no. 3, 2004.

[9] V. Badrinarayanan, P. Pérez, F. L. Clerc, and L. Oisel, "Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues," in *Proc. of ICCV*, 2007.

[10] S. Frintrop and M. Kessel, "Most salient region tracking," in *Proc. of ICRA*, 2009.

[11] S. Frintrop, A. Königs, F. Hoeller, and D. Schulz, "Visual person tracking using a cognitive observation model," in *ICRA Workshop on People Detection and Tracking*, 2009.

[12] T. Mathes and J. H. Piater, "Robust non-rigid object tracking using point distribution manifolds," in *Proc. of DAGM*, 2006.

[13] F. Tang and H. Tao, "Object tracking with dynamic feature graph," in *Proc. of the IEEE Workshop on VS-PETS*, 2005.

[14] N. Beuter, O. Lohmann, J. Schmidt, and F. Kummert, "Directed attention - a cognitive vision system for a mobile robot," in *Proc. of ROMAN*, 2009.

[15] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. CVPR*, 2006.

[16] M. Isard and A. Blake, "Condensation - conditional desity propagation for visual tracking," *IJCV*, vol. 29, no. 1, pp. 5–28, 1998.

[17] S. Frintrop, "VOCUS: a visual attention system for object detection and goal-directed search," Ph.D. dissertation, 2005, in LNAI, Vol. 3899, Springer, 2006.

[18] S. Frintrop, M. Klodt, and E. Rome, "A real-time visual attention system using integral images," in *Proc. ICVS*, 2007.

[19] B. Han, D. Comaniciu, Y. Zhu, and L. S. Davis, "Sequential kernel density approximation and its application to real-time visual tracking," *IEEE Trans. on PAMI*, vol. 30, no. 7, pp. 1186–1197, 2008.