Most Salient Region Tracking

Simone Frintrop and Markus Kessel

Abstract— **In this paper, we introduce a cognitive approach for object tracking from a mobile platform. The approach is based on a biologically motivated attention system which is able to detect regions of interest in images based on concepts of the human visual system. A top-down guided visual search module of the system enables to especially favor features which fit to a previously learned target object. Here, the appearance of an object is learned online within the first image in which it is detected. In subsequent images, the attention system searches for the target features and builds a top-down, target-related saliency map. This enables to focus on the most relevant features of especially this object in especially this scene without knowing anything about a particular object model or scene in advance. The system is able to operate in real-time and to cope with the requirements of real-world tasks such as illumination variations and other moving objects.**

I. INTRODUCTION

The ability to accurately detect and keep track of objects is of large interest in machine vision as well as in mobile robotics. Example applications are surveillance systems, mobile robots which shall guide or follow people, or human-robot interaction in which a robot shall interact with a human and both have to concentrate on the same objects.

The requirements as well as the applicable methods vary largely from task to task. If the object to track is known in advance, model-based trackers may be applied which require an initial training phase. In some applications however, the object of interest is not known in advance. A user might for example show an object to the system which shall immediately react. A long training phase is usually inacceptable in such applications, online learning methods are required. In systems with a static camera, it is possible to apply methods like background subtraction. If interest is for example in counting people or other statistical investigations which do not require immediate response, it is possible to process the data offline which extends the range of applicable algorithms considerably. On the other hand, systems which shall operate 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 which track an object based on simple features such as color cues or corners. An example is the Mean Shift algorithm [1] which classifies objects according to a color distribution or the CamShift algorithm which is based on the Mean Shift approach [2]. An interesting extension is presented in [3], in which the authors suggest to on-line select the currently most discriminative features. One limitation with the above methods is that they operate only on color and are therefore dependent on colored objects.

The required steps in a visual tracking system are first, the detection of the target of interest, second, the data association, i.e. redetecting the same target in subsequent frames and third, the tracking itself which means the inference about the motion of the target given a sequence of previous measurements. Here, we will focus on the second aspect, the data association. The first aspect is an important and difficult research topic on its own which is not tackled here; we initialize manually instead. Inferring the motion of the tracked object can be solved efficiently with Kalman or Particle filters. Note however that in contrast to a tracker based on data from radar or laser scanners, a visual tracker with perfect data association might even be able to work completely without motion inference. Here, we use a simple motion model which assumes that the tracked object will appear in the next frame in a close neighborhood of its current position.

In this paper, we present a system for tracking objects from a mobile platform. The system is capable to learn the appearance of objects online from a single frame in real-time. It is not restricted to a certain type of object but can deal with different types of objects. 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.

For the data association, we follow a feature-based approach based on the biologically motivated attention system VOCUS [4], [5]. The system first determines the most salient region of the object to track and computes a feature vector which describes the region. This vector is utilized to search for the object in a top-down manner within the subsequent frames. An advantage of such an attention system is that it determines several basic features (e.g., intensity, color, orientation) in parallel and determines automatically the ones which discriminate the object best from its surrounding. This is an advantage over other approaches which define the type of feature in advance (e.g., corner-based [6] or color-based [1] trackers).

Visual attention systems have been investigated for about two decades [7], [8], [9], [4], but only recently approaches have been presented which are real-time capable and therefore applicable on mobile platforms [10], [5], [11], [12].

[1]The authors are with the Institute of Computer Science III, Rheinische Friedrich-Wilhems-Universität, 53117 Bonn, Germany. Contact: `frintrop@iai.uni-bonn.de`

The authors want to thank Prof. A.B. Cremers for supporting this work.

Most of these systems operate in a purely bottom-up, image-based manner, i.e., they do not consider pre-knowledge about the scene or a target. Some attention systems which are able to perform visual search for targets in a top-down manner are presented in [13], [4]. Visual attention systems have been used, e.g., for object recognition [14] or robot localization [15]. However, they have rarely been applied to visual tracking. Some approaches track static visual landmarks from a robot [15]. This task is easier than tracking a moving object since the environment of the target remains stable. Other approaches aim to track moving objects such as fish in an aquarium [16], or the most salient regions in an office environment [17]. These approaches base on bottom-up visual attention, thus they are only able to track the most salient spots in a scene. To our knowledge, the here presented work is the first approach which investigates the use of top-down information for tracking, i.e., the active adaptation of feature computations to detect target specific features.

II. Most Salient Region Tracking

Let us start with a brief summary of the Most Salient Region (MSR) Tracker. Input is an image sequence I_t and a target region R_0 in the first frame I_0. To learn the target appearance, the visual attention system VOCUS determines the most salient region M_0 in R_0 and computes a feature vector $\vec{w}_0$ which describes the region (see sec. III-A).

In the following frames VOCUS performs visual search for the target object by computing a top-down saliency map $S_{td}(I_t)$ (sec. III-B), determining the most salient region M_t within a search window R_t and a corresponding feature vector $\vec{w}_t$ (sec. III-C). Feature matching is realized by computing the Tanimoto coefficient $T(\vec{w}_0, \vec{w}_t)$ (sec. III-D). If this value exceeds a threshold δ, the target was recognized (M_t.matched = TRUE), otherwise the target was not recognized. In the latter case the search region is extended until it covers the whole image. As soon as the target was found again ($T(\vec{w}_0, \vec{w}_t) >= \delta$), the search window is again restricted to the neighborhood of M_t. The approach of the MSR tracker is summarized in Algorithm 1.

Algorithm 1 Most Salient Region Tracking

```
Input: Image sequence $I_t$ with $t \in \{0, .., T-1\}$, target region $R_0$ in $I_0$ (determined manually or with automatic recognition)
Output: For each $I_t$: estimated position of target region $M_t$ and variable $M_t$.matched that says if $M_t$ was matched to the target

Initialization:
$F_j(I_0)$ := compute_FeatureMaps($I_0$), $j \in \{1, .., 13\}$
$S_{bu}(I_0)$ := compute_bu_saliencyMap($I_0, F_j(I_0)$)
$M_0$ := compute_mostSalientRegion($S_{bu}(I_0)$,$R_0$)
$\vec{w}_0$ := compute_featureVector($F_j(I_0), M_0$)
$R_1$ = determine_searchWindow($M_0$)

for all frames $I_t, t > 0$ do
  $F_j(I_t)$ := compute_FeatureMaps($I_t$), $j \in \{1, .., 13\}$
  $S_{td}(I_t)$ := compute_td_saliencyMap($I_t, F_j(I_t), \vec{w}_0$)
  $M_t$ := compute_mostSalientRegion($S_{td}(I_t)$,$R_t$)
  $\vec{w}_t$ := compute_featureVector($F_j(I_t), M_t$)
  if $T(\vec{w}_0, \vec{w}_t) < \delta$ then
    $M_t$.matched = FALSE
    $R_{t+1}$ = extend_searchWindow($M_t, R_t$)
  else
    $M_t$.matched = TRUE
    $R_{t+1}$ = determine_searchWindow($M_t$)
  end if
end for
```

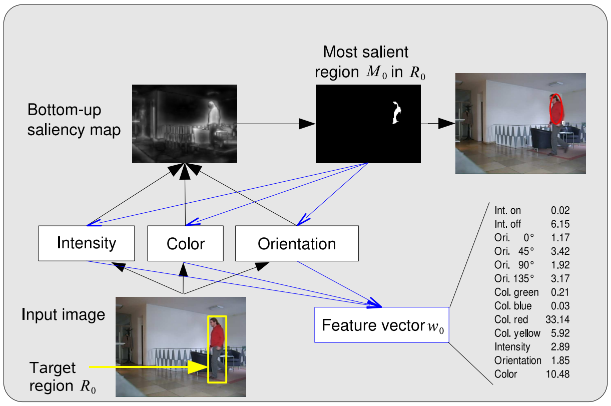


Fig. 1. Initialization: the attention system VOCUS learns the target appearance by computing the most salient region M_0 within search region R_0 (yellow rectangle) and a corresponding feature vector $\vec{w}_0$.

III. Cognitive Data Association

The detection of the features which are used for tracking is performed with the attention system VOCUS (Visual Object detection with a CompUtational attention System). It is based on concepts of the human visual system [18], [19] and detects the most salient regions in images. VOCUS consists of a bottom-up part (similar to [9]) which computes saliency purely based on the content of the current image and a top-down part which considers pre-knowledge and target information to perform visual search. Here, the bottom-up computations are used to learn the appearance of the target (sec. III-A) and the top-down part is used to find the target in the following frames (sec. III-B). More details on VOCUS can be found in [4], [5].

A. Initialization: Learning target appearance

Before the actual tracking can start, the target has to be detected. How this shall be done depends on the application. If the target is known in advance, for example in case of a people tracker, an object classifier can be trained and applied to detect the target. If an arbitrary object shall be tracked which is presented to the system, online learning is required. In our system, the user initializes manually by drawing a rectangle around the target of interest (cf. Fig. 1). Now, VOCUS first computes a bottom-up saliency map (sec. III-A.1), second, determines the most salient region within the rectangle (sec. III-A.2), and third computes a feature vector for this region (sec. III-A.3).

1) Bottom-up saliency map: To compute the bottom-up saliency map, image contrasts and uniqueness of several features (intensity, orientation, and color) are computed independently. The feature computations are performed on 3 different scales with image pyramids. The feature intensity is computed by *center-surround mechanisms* and *integral images* (see [5]); on-off and off-on contrasts are computed separately. After summing up the scales, this yields 2 intensity maps. Similarly, 4 color maps (green, blue, red, yellow) and 4 orientation maps ($0^\circ, 45^\circ, 90^\circ, 135^\circ$) are computed. For the orientation maps, Gabor filters are used (see [4]).

Before the features are fused, they are weighted according to their *uniqueness*: a feature which occurs seldomly in a scene is assigned a higher saliency than a frequently occurring feature. This is a mechanism which enables humans to instantly detect outliers like a black sheep in a white herd. The uniqueness $\mathcal{W}$ of map X is computed as $\mathcal{W}(X) = X/\sqrt{m}$, where m is the number of local maxima that exceed a threshold and '/' is here the point-wise division of an image with a scalar. The maps are summed up to 3 conspicuity maps for intensity, orientation, and color. We denote the 10 feature and 3 conspicuity maps for image I_t in the following as $F_j(I_t), j \in \{1,..,13\}$. The conspicuity maps are finally weighted again and combined to form the *bottom-up saliency map* (according to [4]):

$$S_{bu}(I_t) = \sum_{j=11}^{13} \mathcal{W}(F_j(I_t)) \qquad (1)$$

2) Extracting the most salient region: Next, we compute the most salient region M_0 within the search area R_0. M_0 is computed by first, determining the most salient (brightest) point in S_{bu}, and then with *seeded region growing* the surrounding region (MSR). This method finds recursively all neighbors with sufficient saliency. Although M_0 is irregularly shaped, we display it for simplicity as an ellipse.

3) Determining a feature vector: Now, a feature vector $\vec{w}_0$ (descriptor) is computed which describes M_0 and its surrounding. The value $w_{0,j}$ for map $F_j(I_0)$ is the ratio of the mean saliency in the target region M_0 and in the background $I_0 \backslash M_0$ (image without M_0) [4]:

$$w_{0,i} = \frac{mean(M_0)}{mean(I_0 \backslash M_0)}, \qquad i \in \{1,..,13\}. \qquad (2)$$

This computation does not only consider which features are the strongest in the target region, it also regards which features separate the region best from the rest of the image. After the feature vector for the target has been computed, this vector can be used to search for the target in other frames.

B. Visual Search: Finding the target

In search mode, we first determine a top-down, target-related saliency map S_{td} (cf. Fig. 2). This map is composed of an excitation and an inhibition map. VOCUS uses the previously learned feature vector $\vec{w}_0$ to weight the feature and conspicuity maps according to the target. Depending on the values w_i, the maps are used to compute the excitation map E or the inhibition map I. E is the weighted sum of all feature and conspicuity maps $F_j(I_t)$ that are important for the learned region, i.e., $w_{t,j} > 1$:

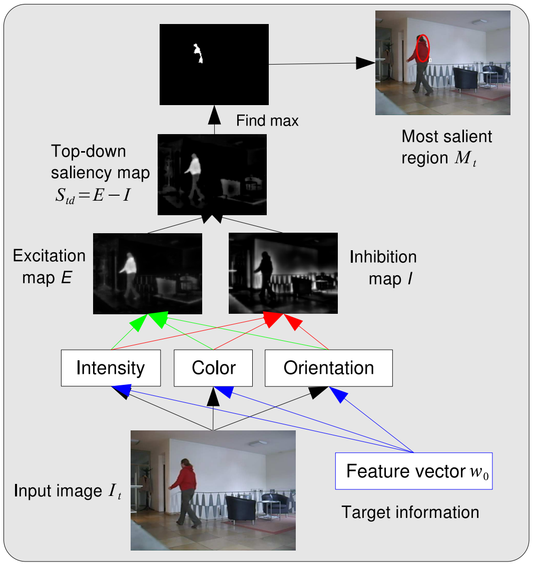


Fig. 2. Visual search for target: the attention system VOCUS weights the feature maps according to values of $\vec{w}_0$. A top-down saliency map S_{td} is computed which highlights the target-specific regions of interest and the most salient region M_t determines the position of the object.

$$E(I_t) = \sum_j (w_{t,j} * F_j(I_t)) \quad \forall j \in \{1..13\} : w_{t,j} > 1 \quad (3)$$

The inhibition map I considers the features more present in the background than in the target region, i.e., $w_{t,j} < 1$:

$$I(I_t) = \sum_j (\frac{1}{w_j} * F_j(I_t)) \quad \forall j \in \{1..13\} : w_{t,j} < 1 \quad (4)$$

Maps with $w_{t,j} = 1$ are completely unimportant for the target and are ignored. The top-down saliency map $S_{td}(I_t)$ results from the difference of E and I and a clipping of negative values: $S_{td}(I_t) = E(I_t) - I(I_t)$. Finally, the most salient region M_t in $S_{td}(I_t)$ is determined as in sec. III-A.2 and the feature vector $\vec{w}_t$ as in sec. III-A.3. The coordinates of M_t determine the position estimate of the target and are the output of the tracking algorithm.

C. Motion Model

Since objects in the real world follow physical rules, it is reasonable to assume that their motion is continuous (at least if an adequate frame rate of at least 30 Hz is available) and that the search area R_t can be restricted to a neighborhood of the target position M_{t-1} in the previous frame. Here, we use a simple motion model: R_t is k_1 pixels larger in width and height than M_{t-1} (here we use $k_1 = 8$). In future work, this simple estimation shall be replaced by a Particle filter.

D. Feature Matching

In general, it is not necessary to perform a feature matching between the region of the current frame and the target region since the top-down search already assures the similarity. In certain cases, a feature matching is reasonable

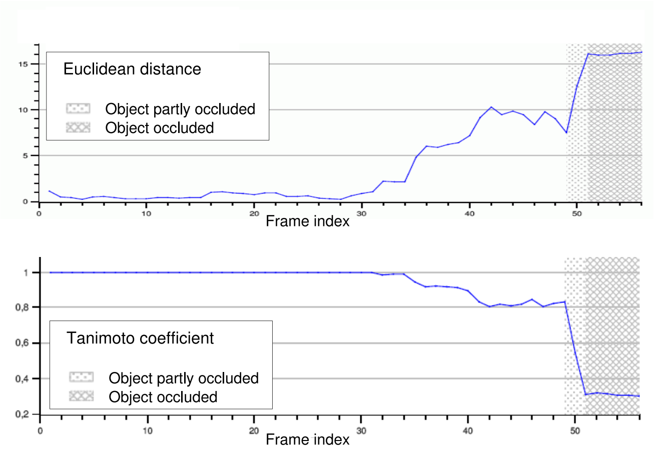


Fig. 3. The Euclidean distance (top) and the Tanimoto coefficient (bottom) for the first 55 frames of the ball sequence (cf. Fig. 4). Around frame 50, the ball disappears behind a box which is clearly visible with both measures.

anyway. If the target is suddenly occluded by another object, the top-down search has no target to detect and will detect the region which is next similar to the target. Since it is reasonable to know that the target was lost, the regions are matched.

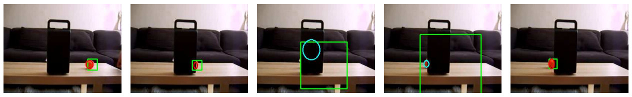
We tested two different measures to determine the similarity of the two feature vectors $\vec{w}_0$ and $\vec{w}_t$, first, the Euclidean distance $d(\vec{w}_0, \vec{w}_t)$, and, second, the Tanimoto coefficient

$$T(\vec{w}_0, \vec{w}_t) = \frac{\vec{w}_0 * \vec{w}_t}{||\vec{w}_0||^2 + ||\vec{w}_t||^2 - \vec{w}_0 * \vec{w}_t}.$$

While the Euclidean distance measures the dissimilarity and is 0 if the vectors are the same, the Tanimoto coefficient measures the similarity and is 1 for identical vectors. Fig. 3 shows both measures for an image sequence in which a ball is tracked and disappears behind an object. Both measures show the disappearance of the object clearly. Since the difference between the values for "target present" and "target absent" is clearer for the Tanimoto coefficient, we use this measure. As long as $T(\vec{w}_0, \vec{w}_t) >= \delta$ (here: $\delta = 0.6$) the target is tracked successfully. If $T(\vec{w}_0, \vec{w}_t) < \delta$, the target was probably lost. In this case, a reinitialization is necessary. Here, we choose a simple solution and extend the search area successively by k_2 pixels (here: $k_2 = 20$) in width and hight in the next frame. This is repeated until the region covers the whole image or until the target is redetected. In the latter case, the search area is restricted to its previous size. Fig. 4 illustrates this.

IV. Experiments and Results

The experiments are divided into two parts. First, we investigate the performance of the MSR tracker on different scenes and highlight the characteristics of the approach. Second, we compare our approach with one of the standard approaches for object tracking, the CamShift algorithm [2]. In both sections, we used image sequences from a hand-held camera with a resolution of 320×240 pixels. Most sequences were obtained indoors, but we also investigated some outdoor scenes. The algorithm works in real-time (30 Hz) on a 1.66 GHz PC. The algorithm was applied with the same parameter set in all experiments.



frame 41 frame 50 frame 55 frame 57 frame 58

Fig. 4. The ball sequence (cf. Table I, row 1). Red ellipse: target matched. Cyan ellipse: target not matched. Green rectangle: search area R_t. When the ball disappears, the algorithm does not detect the target and enlarges the search area. In frame 57, the ball is focused again but not yet matched. In frame 58, it is matched and thus successfully redetected.

A. The MSR tracker

In this section, we show the results of the MSR tracker for different objects in different settings. We investigated how well the approach copes with a moving camera, with other moving objects, and with partial and complete occlusions of the objects. The objects ranged from toy and everyday objects (ball, mammoth, cup) to faces, animals (cat) and persons. To evaluate the tracking, we counted the number of *object hits* manually. An object hit occurs if the center of the estimated target region M_t was on the target object[1] and the matching method reported a match. If the object was focused (M_t on target) but not matched, this is counted as a miss. If the object was not visible (occluded) and the system reported no match, this was counted as correct recognition.

Some frames of the image sequences are displayed in Fig. 5, the corresponding tracking performance is shown in Table I.[2] In most examples, the target is tracked very well and recognized by the matching method, resulting usually in recognition rates of over 90 %. If the object is occluded for several frames, the method is in most cases able to redetect it when reappearing. This is only prevented if the algorithm focuses in the meantime on an object with similar appearance so that the feature vectors match. In such cases, it is difficult to redetect the target. An example is scene 2: Here, two identical balls move through the scene, one from left to right, one in the other direction. If the right ball is the target, it is tracked successfully throughout the scene, but when the left ball is tracked, it is lost when the balls cross their way and the focus follows the other ball instead.[3]

In examples 10 – 13 we investigated the special case of person tracking since following individuals with a mobile robot is one of our planned future applications. We followed the person with a hand-held camera through a corridor (scene 10, 12, 13) and in one case outdoors (scene 11). In scenes 12 and 13, two people were walking next to each other, in 13 they crossed their way twice. Although the people did not

[1] This is an approximation which is actually too optimistic since the region might include a part of the background and still have its center on the region. However, this occurs only rarely since the MSR is usually small.

[2] Parts of the results can be also seen in the accompanying video attachment and on http://ivs.informatik.uni-bonn.de/research/tracking

[3] While it is possible to prevent this by a more sophisticated motion model, note that this might prevent dealing with unexpected motion.

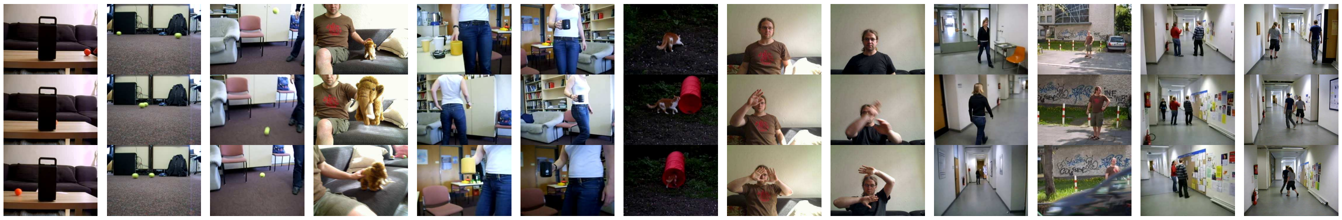


Fig. 5. Some frames from the image sequences used for the experiments in Tables I and II. The number of the column corresponds to the example number in the tables. Columns: 1) red ball moving from right to left, partly occluded 2) two balls crossing their way 3) jumping ball 4) toy mammoth 5) yellow cup carried around, partly occluded 6) black cup carried around 7) cat outside, finally disappearing 8) & 9) face, partly occluded by hands 10) person inside 11) person outside 12) two persons 13) two persons crossing their way (see also video attachment).

	Object	part-time occluded	camera moving	other moving obj.	indoor/ outdoor	# Frames	correct recognitions [%]	misses [%]	false positives [%]
1.	Ball (red)	yes	no	no	i	74	99	1	0
2.a	2 balls (track right)	no	no	yes	i	38	100	0	0
2.b	2 balls (track left)	yes	no	yes	i	38	47	0	53
3.	Jumping ball	no	yes	no	i	100	82	15	3
4.	Toy mammoth	no	yes	yes	i	749	97	3	0
5.	Yellow cup	yes	yes	yes	i	540	96	2	2
6.	Black cup	no	yes	yes	i	581	91	1	8
7.	Cat	yes	yes	no	o	1167	97	1	2
8.	Face	yes	no	yes	i	300	94	0	6
9.	Face	yes	yes	yes	i	808	36	48	16
10.	Person	no	yes	no	i	1306	98	2	0
11.	Person	yes	yes	yes	o	782	99	1	0
12.a.	2 persons (track left)	no	yes	yes	i	650	98	2	0
12.b	2 persons (track right)	no	yes	yes	i	650	100	0	0
13.a	2 persons (track left)	yes	yes	yes	i	390	82	10	8
13.b	2 persons (track right)	yes	yes	yes	i	390	99	1	0

TABLE I

Detection performance for the scenes of Fig. 5. In each frame, we determine only one target region, therefore each frame contains either a correct recognition, a miss, or a false positive, whereas a false positive is always a miss at the same time.

wear special clothes, tracking was successful in most cases (close to 100 %). Even when the persons crossed their way, the system was usually able to resolve the ambiguity. Only when tracking the left person in scene 14, the target was lost sometimes, but it was redetected after several frames. Note, that the algorithm is also able to track objects without color (black or white) by exciting the intensity contrast and inhibiting irrelevant features like color.

B. Comparison with CamShift

Most similar to the here presented MSR tracking are color-based trackers such as trackers based on the MeanShift algorithm [1]. One well-known approach is the CamShift algorithm [2] which is publically available from the OpenCV library[4]. We used this algorithm as benchmarking system for our approach.

Although the CamShift algorithm has shown good results in other applications, it is only of limited use for a flexible online tracker. Usually, it is necessary to adapt the parameters of the algorithm newly for each object (or object class) to obtain good results. While this may be acceptable for some applications like face tracking, it is difficult if the system shall be able to track different objects which are not known in advance. Since our MSR tracker is applicable to different objects without adapting parameters, we used for the CamShift algorithm the standard parameter set of the OpenCV implementation for all test sequences to make the approaches comparable. Another difference between the approaches is that the feature matching of our approach enables us to not only detect the most likely hypothesis but to classify whether the object was detected or not. The available CamShift implementation does not support this, it returns instead the most likely hypothesis in each frame, even if the object is not present. Therefore, we adapted the MSR tracker for this comparison accordingly and count the detections instead of the recognitions: if the target was focused but not matched, this counts as a detection, if the target is absent and something else is focused instead, this counts as a miss in both approaches. As before, a detection is counted if the center of the detected region is on the target.

Table II shows the results of the comparison. In most cases (14 out of 16) the MSR tracker outperforms the CamShift algorithm clearly. The main reason is that without adapting the parameters to the particular target object, the CamShift algorithm has problems to distinguish between target and background, especially if the background contains similar colors as the target. This was for example the case in scene 1 (also Fig. 4): the wood of the table has a similar hue value as the ball which made it impossible to distinguish

[4] http://opencvlibrary.sourceforge.net/

	Object	# Frames	correct detections [%]	
			MSR	CamShift
1.	ball (red)	74	92	3
2.a	ball (track right)	38	100	47
2.b	ball (track left)	38	47	34
3.	jumping ball	100	83	100
4.	toy mammoth	749	99	35
5.	yellow cup	540	95	37
6.	black cup	581	91	0.2
7.	cat	1167	97	91
8.	face	300	94	1
9.	face	808	42	93
10.	person	1306	99	44
11.	person	782	99	98
12.a	2 persons (track left)	650	100	79
12.b	2 persons (track right)	650	100	27
13.a	2 persons (track left)	390	78	27
13.b	2 persons (track right)	390	99	10
.	Average	535	88	45

TABLE II

COMPARISON MSR TRACKING AND CAMSHIFT TRACKING [2]

ball and table. Illumination variations, changing backgrounds and similar colors also lead to difficulties as well as the absence of color (black cup, persons in scene 10, 12b, 13a). Since VOCUS computes center-surround contrasts of colors and intensities instead of absolute values, it is less sensitive to illumination changes etc. and the combination of exciting and inhibiting a variation of features facilitates distinguishing between target and background. Overall, the CamShift tracker shows an average performance of 45%, the MSR tracker achieves 88%.

V. CONCLUSION

In this paper, we have presented the MSR tracker, a cognitive approach for visual object tracking from a mobile platform. The appearance of an object of interest is learned from an initially provided target region and the resulting target feature vector is used to search for the target in subsequent frames. The main advantage of the system is that it determines autonomously the most relevant target features in a current setting as well as the irrelevant features which are inhibited.

Other advantages are that the system is quickly adaptable to a new target without a time-consuming learning phase, that it works on a mobile platform and does not rely on a static background, and that it works in real-time and under varying illumination conditions.

However, there is still room for improvement and in hard conditions problems might occur. Such difficult conditions are e.g. situations in which the target object is very similar to the background or to another object in close neighborhood. As long as a region has only temporarily a similar appearance as the target, the algorithm will recover. Only if the system confuses the target with a region which is consistently very similar, recovering is difficult since the system does not notice its failure. Integrating more features to the attention system might help to reduce this problem but will also need additional computational power.

Additionally, the feature 'orientation' is only helpful if the target does not rotate. Otherwise, an adaptation of the feature vector or a collection of several feature vectors that represent different object states would be helpful. The same applies for other strong changes in object or background appearance. As several experiments have shown, an adaptation of the feature vector is not trivial since it includes the risk of false adaptation. Thus, the algorithm might include features of the background into the adapted feature vector, resulting in false detections. This happens especially if the target is occluded for several frames. An optimal solution is difficult and a challenging issue for future work. Additionally, we plan to integrate motion features into VOCUS and to use a Particle filter for tracking to deal with several object hypotheses.

REFERENCES

[1] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," Proc. Conf. Computer Vision and Pattern Recognition (CVPR), vol. 2, 2000.

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

[3] Y. R.Collins and M.Leordeanu, "On-line selection of discriminative tracking features," *IEEE Trans Pattern Analysis and Machine Intelligence (PAMI)*, 2005.

[4] S. Frintrop, "VOCUS: a visual attention system for object detection and goal-directed search," Ph.D. dissertation, Lecture Notes in Artificial Intelligence (LNAI), Vol. 3899, Springer, 2006.

[5] S. Frintrop, M. Klodt, and E. Rome, "A real-time visual attention system using integral images," in *Proc. of the 5th Int'l Conf. on Computer Vision Systems (ICVS)*, Bielefeld, Germany, March 2007.

[6] J. Shi and C. Tomasi, "Good features to track," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[7] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," *Human Neurobiology*, vol. 4, no. 4, pp. 219–227, 1985.

[8] J. K. Tsotsos, S. M. Culhane, W. Y. K. Wai, Y. Lai, N. Davis, and F. Nuflo, "Modeling visual attention via selective tuning," *Artificial Intelligence*, vol. 78, no. 1-2, pp. 507–545, 1995.

[9] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[10] L. Itti, "Real-time high-performance attention focusing in outdoors color video streams," in *Proc. SPIE Human Vision and Electronic Imaging IV (HVEI)*, San Jose, CA, 2002.

[11] S. May, M. Klodt, and E. Rome, "GPU-accelerated Affordance Cueing based on Visual Attention," in *Proc. of Int'l Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2007, pp. 3385–3390.

[12] M. Björkman and J.-O. Eklundh, "Vision in the real world: Finding, attending and recognizing objects," *Int'l Journal of Imaging Systems and Technology*, vol. 16, no. 2, pp. 189–208, 2007.

[13] V. Navalpakkam, J. Rebesco, and L. Itti, "Modeling the influence of task on attention," *Vision Research*, vol. 45, no. 2, pp. 205–231, 2005.

[14] D. Walther, U. Rutishauser, C. Koch, and P. Perona, "Selective visual attention enables learning and recognition of multiple objects in cluttered scenes," *Computer Vision and Image Understanding (CVIU)*, vol. 100, no. 1-2, pp. 41–63, 2005.

[15] S. Frintrop and P. Jensfelt, "Active gaze control for attentional visual SLAM," in *Proc. of the IEEE Int'l Conf. on Robotics and Automation (ICRA'08)*, 2008.

[16] M. Veyret and E. Maisel, "Attention-based target tracking for an augmented reality application," Int'l Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision, 2006.

[17] N. Ouerhani and H. Hügli, "A model of dynamic visual attention for object tracking in natural image sequences," in *Int'l Conf. on Artificial and Natural Neural Networks (IWANN)*, vol. 2686. Springer Verlag, Lecture Notes in Computer Science (LNCS), 2003, pp. 702–709.

[18] A. M. Treisman and G. Gelade, "A feature integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97–136, 1980.

[19] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus-driven attention in the brain," *Nature Reviews*, vol. 3, no. 3, pp. 201–215, 2002.