Adaptive Object Tracking by Learning Background Context

Ali Borji* Simone Frintrop† Dicky N. Sihite* Laurent Itti*
* Department of Computer Science, University of Southern California, Los Angeles
† Institute of Computer Science, Rheinische Friedrich-Wilhelms Universität Bonn

Abstract

One challenge when tracking objects is to adapt the object representation depending on the scene context to account for changes in illumination, coloring, scaling, etc. Here, we present a solution that is based on our earlier approach for object tracking using particle filters and component-based descriptors. We extend the approach to deal with changing backgrounds by using a quick training phase with user interaction at the beginning of an image sequence. During this phase, some background clusters are learned along with object representations for those clusters. Next, for the rest of the sequence the best fitting background cluster is determined for each frame and the corresponding object representation is used for tracking. Experiments show a particle filter adapting to background changes can efficiently track objects and persons in natural scenes and results in higher tracking results than the basic approach. Additionally, using an object tracker to follow the main character in video games, we were able to explain a large amount of eye fixations higher than other saliency models in terms of NSS score proving that tracking is an important top-down attention component.

1. Introduction

In human robot interaction, an essential task of the robot is to focus on the same targets of interest as a human partner. For example, if a human shows an object to the robot, the robot has to be able to quickly learn the appearance of the object, to follow the object with the camera and, if told to bring the object to the human, to redetect the object in a different setting. Tracking interesting and task-relevant objects is also an important top-down factor in control of human visual attention [26].

In this work, we will focus on two problems 1) adaptive tracking of objects: we consider only camera-based approaches here since these are best suited to capture the appearance of arbitrary objects. Furthermore, cameras are light-weight, low-cost, and passive sensors that can be easily integrated on almost every robot platform and, 2) saliency modeling based on object tracking.

1.1. Object Tracking

Several 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 the requirements and the setting. In human robot interaction, methods are needed that are quick enough to deal with real-time requirements and are robust enough to deal with illumination and viewpoint changes as well as low resolution cameras. The approaches also have to work on a mobile system which requires to deal with changing backgrounds and motion blur. It is also desirable to be able to track arbitrary objects from a quick training phase. An exhaustive training phase, in which an object is trained under all possible viewpoints, object deformations and illumination changes from thousands of images is usually unacceptable. To meet all these requirements, feature-based methods are an especially suitable approach. They track an object based on simple features such as color cues or corners. An example is the Mean Shift algorithm [2] which classifies objects according to a color distribution or the CamShift algorithm which is based on the Mean Shift approach [3]. Other groups integrate color histograms into a particle tracker [4, 5, 30]. In a previous work, we have used a cognitive observation model for visual tracking that was based on features inspired by human visual perception [6]. In a recent work, we have extended this approach to a component-based tracking method [8, 7]. Over the last years, also techniques which use interest points, like colored Harris corners [9] or SIFT features [10] have been introduced for object tracking. Note that these approaches usually rely on textured objects and a certain image resolution and quality to work well.

One challenge for feature-based tracking is the adaption of the feature descriptor (representing object of interest) over time. A descriptor that is learned from a single frame can work well in the setting it was learned but might perform poorly if background and illumination change too strongly. Adapting the target descriptor accordingly is not a trivial task since, without user feedback, the risk is high that the system adapts the descriptor wrongly to the background. This happens especially if the target of interest is

978-1-4673-1612-5/12/$31.00 ©2012 IEEE

occluded for several frames. If this happens, the system will lose the target and will never be able to redetect it again. Most existing approaches ignore this problem by either not adapting the descriptor or by making sure that the target is not occluded within the image sequence.

In the long run, an object tracking system should be able to automatically detect when the descriptor shall be adapted, when it is safe to adapt it, and how the new setting is best integrated into the new target descriptor. Here, we make one step into this direction: we learn the appearance of backgrounds and the corresponding target descriptors from a training sequence and cluster the resulting descriptors according to the background appearance. This gives us a sparse, context-based object representation. In a test sequence, the context is analyzed automatically and the best fitting descriptor is selected to find the target. As target descriptor, we use a component-based descriptor that we have developed in previous work [8]. It is integrated into a CONDENSATION-based object tracker [11] that maintains a set of weighted particles over time. Each particle represents a hypothesis for the current object position and its weight is set according to the similarity to a target descriptor.

1.2. Saliency Modeling

The term "saliency" is often referred to visual attention where some parts of stimuli are selected for further processing. Selection mechanism could be bottom-up where it is derived by stimuli level competitions or top-down task-relevance mechanisms based on demand. Several factors have been shown to guide bottom-up visual attention such as intensity contrast, color contrast, orientation contrast, size, faces and text (see [19, 26] for a review). At the other hand discovering features influencing top-down attention is difficult due to their high dependency on task. However some general factors have been proposed including look-ahead fixations [24], scene context (gist) [25], etc. Despite this, defining a clear boundary between bottom-up and top-down attention if not impossible, is very difficult.

Most of research on saliency modeling have been focused on bottom-up models on single static images. Recently, some research have been reported on modeling saliency in spatio-temporal domain (e.g. free viewing of videos). There are few works on modeling top-down attention in near real world scenarios (e.g. interactive game playing, coffee making, driving) [17, 21, 20, 23, 18].

In this paper, we follow a different direction than spatio-temporal saliency models by tracking a task-relevant object. In the time course of attending to different objects in time, an agent should decide what object to attend and extract some information about it which needs to track its position for a period of time. We simply assume that overall behavior is selectively deciding what object to track.

In the rest, first we introduce the target descriptors including component-based descriptor and color-histogram features(Sec. 2), followed by a description of the context-based clustering of backgrounds (Sec. 3). In Section 4, we explain the visual tracking system. Section 5 presents experimental results on tracking. Section 6 shows results of saliency modeling. We finally conclude in Section 7.

2. Computation of the target descriptor

The target 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 a cognitive perception model [12]. Details about the descriptor computation can be found in [8], here we just present a brief overview.

First, six feature maps F_i are computed. They represent intensity and color contrasts based on color-opponent cells of the human visual system. The computation is performed according to [8], resulting in maps for bright-dark, dark-bright, green-red, blue-yellow, red-green and yellow-blue contrasts. Examples of such feature maps are illustrated in Fig. 1.

Second, we compute a component-based target descriptor from the feature maps. 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 finding local intensity maxima and then segmenting the region around each maximum with a region growing approach [13]. For easier computations, 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. 2, left). Now, we compute 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:

$$d_{i,j} = \frac{mean(m_{i,j})}{mean(F_i \backslash m_{i,j})} \quad (1)$$

The mean is computed with integral images [14], to speed up processing and enable constant computation times for each region, independent of the size of the region. Thus,

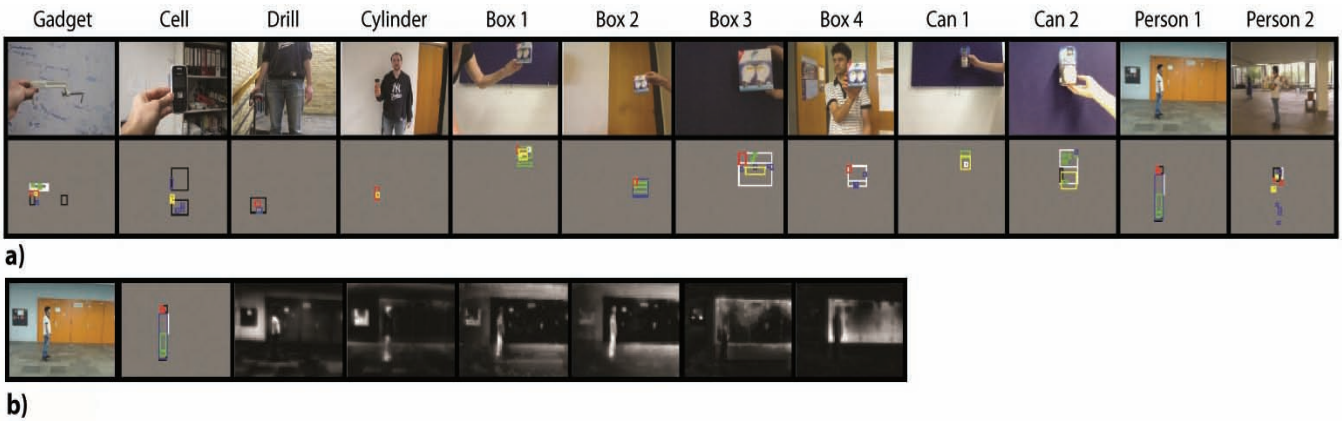

Figure 1. a) Used objects in this paper and their corresponding component-based descriptors, b) Descriptor and feature maps F_i for Person 1. From left to right: bright-dark, dark-bright, green-red, blue-yellow, red-green and yellow-blue contrasts.

the target descriptor that we obtain is $\vec{d^*} = \{d_{i,j} | i \in \{1,..,6\}, j \in \{1,..,l_i\}\}$.

Finally, we describe how the target descriptor $\vec{d^*}$ is matched to an image region $\vec{R'}$ of arbitrary size and dimensions. We first determine the factors f_w and f_h that represent the difference in size between the target region $\vec{R^*}$ and $\vec{R'}$. These factors are used to compute an adapted template $\vec{M_{R'}}$ by extending or compressing all $m_{i,j} \in \vec{M_{R^*}}$ with f_w and f_h. From this template, a descriptor $\vec{d'}$ is computed equivalently to (1). The descriptors $\vec{d^*}$ and $\vec{d'}$ are matched by computing the similarity of the vectors with the Tanimoto coefficient. This measure produces values in the interval $[0, 1]$, the higher the value the higher the similarity.

We also tried another feature descriptor that has been often used in tracking applications. We used a particle filter in a color-based context. Color distributions are used as target models as they achieve robustness against non-rigidity, rotation and partial occlusion. Assume that the distributions are discretized into m-bins. The histograms are produced with the function $h(x_i)$, that assigns the color at location x_i to the corresponding bin. In this paper, histograms are calculated in the RGB space using $8 \times 8 \times 8$ bins.

The color histogram $p_y = \{p_y^u\}_{u=1..m}$ at location y is calculated as:

$$p_y^u = f \sum_{i=1}^{I} k(\frac{||y - x_i||}{a}) \delta[h(x_i) - u] \qquad (2)$$

where I is the number of pixels in the region, δ is the Kronecker delta function. $k(.)$ is a weighting function that assigns smaller weights to the pixels that are further away from the region center. f is normalization factor ensuring that area under histogram sums to 1.

To measure the distance between two distributions $p(u)$ and $q(u)$, Bhattacharyya coefficient was used: (see [30]).

$$d = \sqrt{1 - \rho[p, q]}; \rho[p, q] = \sum_{u=1}^{m} \sqrt{p^u q^u} \qquad (3)$$

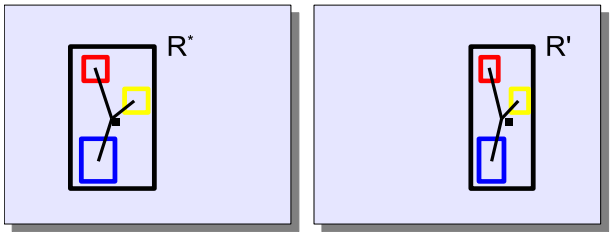

Figure 2. Left: An illustration of the template $\vec{M_{R^*}}$ for the target region $\vec{R^*}$. The three colored rectangles denote the $m_{i,j}$. Note that each of them comes from a different feature map which is illustrated here by different colors. Right: the template $\vec{M_{R'}}$ adapted to region $\vec{R'}$.

3. Clustering backgrounds to obtain context-based feature descriptors

In this section we describe how to adapt the object descriptors based on background changes. While the component-based object detection system is capable of detecting the same object in different backgrounds to a limited degree, it needs to be enhanced to work for situations when the background changes strongly. The reason is that when the background changes, the same descriptor from the previous background is not always capable to detect the same object in the new background. Due to this, it is necessary to learn new descriptors for different backgrounds and automatically apply the appropriate descriptor to the observed background or scene to detect the object of interest.

To do this, we first learn a number of background clusters from a train image sequence and also their corresponding object descriptors which can successfully detect the object in those backgrounds. Then over a test sequence, for each frame, first we find its background cluster and then apply the descriptor of that cluster to the frame.

Two important points should be noticed: First, we wanted our solution to be efficient, fast and comparable to other approaches which have just used color-based features (and not orientation maps). For this, we needed a repre-

sentation of the image which could be calculated quickly and allows to distinguish among many different background classes. Therefore, we extended the representation for calculating descriptors mentioned in Sec. 2, i.e. the 6D feature vector $\vec{b}$ from the six feature maps F_i where each value b_i is the average over the entire map F_i. To make this representation more discriminant, we derive feature values e_i in a $n \times m$ partition overlaid image:

$$\vec{e_i} = (E_i), E_i = \begin{bmatrix} F_i^{11} & F_i^{12} & \dots & F_i^{1m} \\ F_i^{21} & F_i^{22} & \dots & F_i^{2m} \\ & \dots & & \\ F_i^{n1} & F_i^{n2} & \dots & F_i^{nm} \end{bmatrix}, i = 1..6. \tag{4}$$

where element F_i^{pq} of E_i matrix is the normalized mean of $F_i((p-1)w : pw, (q-1)h : qh)$ region of map F_i as in (5) and $(.)$ generates a row vector of matrix E_i.

$$F_i^{pq} = \frac{Avg(F_i((p-1)w : pw, (q-1)h : qh)) - m_{i1}^{pq}}{m_{i2}^{pq} - m_{i1}^{pq}} \tag{5}$$

In (5), w and h are width and height of a partition cell and m_{i1}^{pq} and m_{i2}^{pq} are minimum and maximum of F_i^{pq} which are used for normalizing values to the range of $[0, 1]$. The resulting vector $\vec{e_i}$ is of size $6 \times n \times m$. In this paper m and n are set to 4 thus having feature vectors of size 96.

The second point is the way adaptive object tracking is done. When tracking starts, the user interacts with the tracking system online by spotting when background changes until the train phase is over. The tracking system may add more clusters by analyzing the frames between clusters spotted by the user. Training is online where user interactively sketches the object extent in a new background.

In what follows the algorithm for learning background clustering is described. The sequence containing the object is shown to the user and he/she starts to mark the object when a change in background occurs. To create a background cluster, user marks the extent of the object and the component-based descriptor (as mentioned in sec. 2) is calculated for this object. To eliminate the subjectivity in detecting background change, frames up to this point are processed and clustered using Basic Sequential Algorithmic Scheme (BSAS) [15] algorithm. So, from the last pause by the user till the current pause either 1 or more background clusters will be generated. The derived object descriptor is associated to the newly created cluster. In the rest we formalize this argument in more detail.

The first frame is considered as the first background cluster C_1. Let f_j and f_i be the current and previous frames paused by the user and d_j and d_i be the corresponding descriptors derived from the object in those frames. Assume that up to this point n clusters have been generated. To cluster $(j-i)$ frames in between, the j-th frame is considered as the first cluster center by the BSAS clustering algorithm in this range. To cluster a frame f_z in this range, the distance of this frame to all clusters generated from these frames at any time is calculated and if the minimum of these distances is less than a threshold θ, it is assigned to the nearest cluster C_k and then cluster center of this cluster is updated according to (6). Otherwise, a new cluster is generated with this frame as its center.

$$\vec{e}_{C_k}^{\,new} = \frac{(n_{C_k^{new}} - 1)\vec{e}_{C_k}^{\,old} + \vec{e}_z}{n_{C_k^{new}}} \tag{6}$$

The parameter threshold θ controls the sensitivity of the clustering thus number of clusters. Here, we set this value as 0.4. In (6), n_{C_k} denotes the number of elements in the k-th cluster and $\vec{e}_{C_k}^{\,new}(\vec{e}_{C_k}^{\,old})$ is the prototype of cluster C_k after (before) the assignment of $\vec{e}_z$ to it. The descriptor of the j-th frame is assigned as the descriptor of all newly generated clusters. This process is continued online and interactively until the user decides to stop it, for example when there is no new background cluster. After this process the rest of the sequence is considered as test sequence and for each frame of the rest, first the nearest background cluster of that frame is determined and the descriptor of this cluster is used for tracking the object via updating the descriptor of each individual particle as is mentioned in the next section.

4. Particle filter based tracking

The tracking system we present uses the target descriptors from Sec. 2 for the observation model of a particle filter approach. The descriptor is updated according to Sec. 3 if the background changes strongly. The tracker employs the standard Condensation algorithm [11] 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.

Initialization of the target region is done manually here but could also be done automatically by a detection module. The motion model is realized 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. This makes few assumptions on the movement of the target and thus enables to deal with arbitrary tracking situations.

The tracker maintains a population of J particles (here: $J = 500$) over time t. Each particle $\vec{\phi}_t^j$ has the following form: $\vec{\phi}_t^j = (\vec{s}_t^j, \pi_t^j, \vec{d}_t^j), \quad j \in \{1, ..., J\}$. Here, $\vec{s}_t^j$ is a state vector $\vec{s}_t^j = (x, y, v_x, v_y, w, h)$ that specifies the parti-

cle's region with center (x, y), width w and height 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_t^j determining the relevance of the particle with respect to the target, and the component-based descriptor $\vec{d_t^j}$ that describes the appearance of the particle region.

The most crucial part of the tracker is the observation model that is based on the component-based descriptor. First, we compute for each particle a descriptor $\vec{d_t^j}$ according to sec. 2, where j denotes the number of the particle and t the number of the current frame. 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 T as:

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

where $\vec{d^*}$ denotes the target descriptor. 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 $\sum_{j=1}^{J} \pi_t^j = 1$.

The target descriptor $\vec{d^*}$ represents the target and may either be determined from the first frame (as in our previous work [8]) or from a previously learned set of descriptors. In this paper, it is determined as context-based descriptor that fits best to the current background, as described in sec. 3.

Finally, the current target state, including target position and size, is estimated as weighted average of the particles:

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

To adapt the particle tracking to account for background changes, for each frame in a sequence we find its cluster among the learned background clusters from training frames and then use the descriptor of that cluster. Particles in the current population are updated with this new descriptor. The rest is the same as before.

5. Object Tracking Experiments

In this section, results of our approach for object and person tracking using component descriptors are shown. Six objects (e.g Can, Cell, Box) and a persons (cf. Fig.1) in different settings were tracked in natural setups with fixed and moving camera, other moving objects and partial and complete occlusions. We used image sequences from a handheld camera with a resolution of 320 $\times$240 pixels in colored PNG format. The lengths of the sequences is mentioned in Tab. 1. The same set of parameters was used in all experiments. The first parts of a sequence of an object is considered as training data and used for derivation of background clusters and the rest is considered as test data. The tracking algorithm runs in real-time (30Hz) on a 2.9 GHz PC.

Results of object and person tracking using particles are shown in two cases: in the first case, "first-frame case", the descriptor for the object is derived from the first frame of its corresponding train set and is used to track the object in frames of the test set. To derive the descriptor, the user selects the object extent manually in the first frame and then the object descriptor is calculated in the way presented in Section 2. In the second case, "clustering case", the nearest cluster for each test frame is determined and then the corresponding descriptor of this cluster is used for updating the descriptor of all particles (refer to Section 4).

Fig. 3 shows some frames of test sequences with particles and estimated rectangles locating objects. Yellow rectangles show that the tracker has high confidence (average confidence of all particles) while blue means less confidence. The confidence of each particle is the similarity of the object region that it points to and the descriptor of the cluster of the frame (or descriptor of the object in the first frame in first-frame case) [8].

Figure 4 shows traces in x and y dimensions for the Gadget and Cell objects, respectively over test sequences. It can be seen that the clustering case performs better than the first-frame case and is closer to the ground truth traces in both x and y dimensions. Ground truth values are center of the rectangle encompassing the object. The total Euclidean distance between the positions by the first-frame (clustering) case and ground truth positions are 1908 (1091) for Gadget in test phase except occluded frames, respectively. These values for the Cell are 3240 and 1517 for first-frame and clustering cases, respectively. In train frames from 1203 to 1269 and from 1674 to 1746, the Cell was not in the field of view. This object was out of view in test frames from 1116 and 1153 (black vertical lines in Fig. 4). As can be seen tracking is robust to occlusions and the out-of-view problem and is able to reacquire the object as it reappears.

Detection rate (percentage of frames with the object correctly detected) and detection enhancement rate (in parentheses) are reported in Tab. 1. In both train and test cases except the Box[3] (since it was a small easy case and without large changes in background) we observed an increase in detection rate of clustering compared to the first-frame case. An object is considered as detected if the center of the rectangle M_o proposed by the tracker is on the manually tagged target region M_t in each frame.

For Box and Can, we applied the clusters learned from one train sequence to other test sequences to check the generalization of the approach (5.b, 5.c and 10.b). Results are more reliable for the first 4 sequences since they are longer. For these objects variations in background were more (e.g Cylinder and Cell). Larger number of clusters for these sequences have been generated and have resulted in higher

Figure 3. Sample frames from Cell, Person[2] and Drill test sequences and estimated target rectangles. Green dots: particles that matched to target, cyan dots: particles that did not match. Yellow rectangle means high confidence and blue means less confidence.

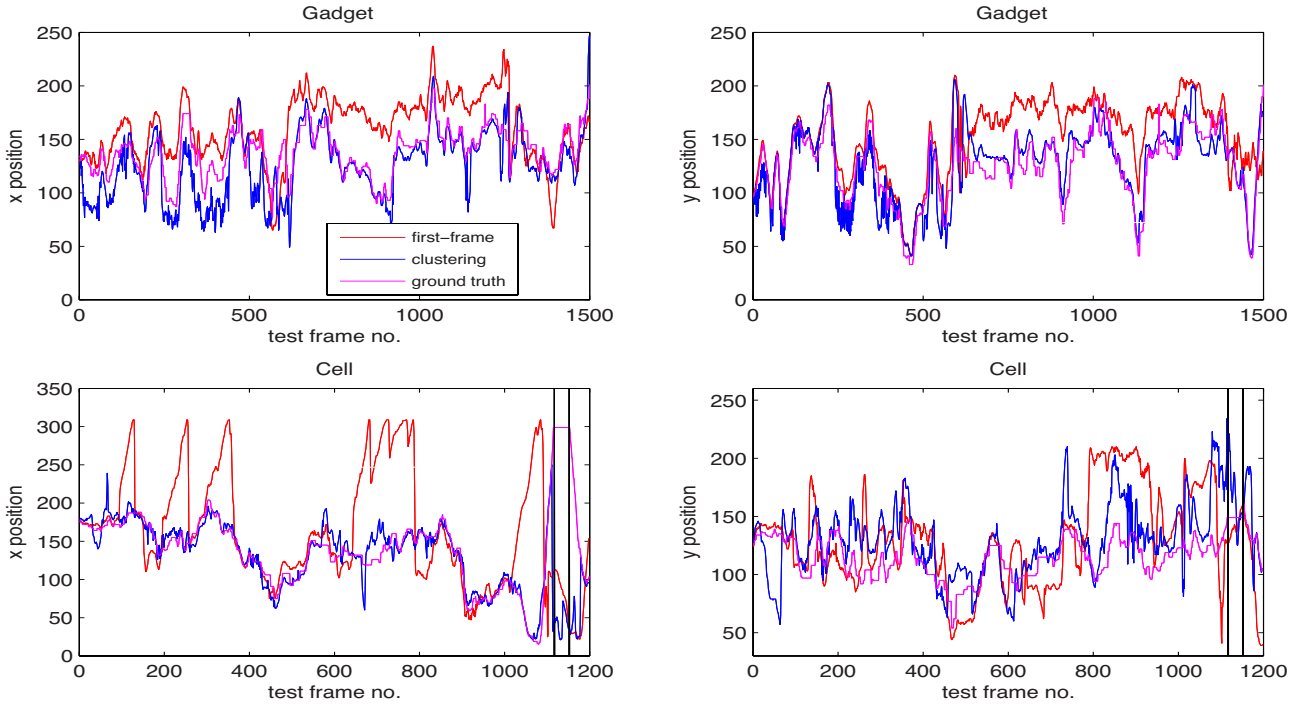

Figure 4. Traces of object position in x and y dimensions for Gadget and Cell objects. Black vertical bars show the occluded frames for Cell object.

enhancements. The average detection enhancement of the clustering case over all objects in train and test phases are 27.4% and 24.6%, respectively.

6. Saliency Modeling Experiments

In this section we aim to see how well a saliency model based on tracking the main character of a video (here Nintendo gamecube games) could predict eye fixations of human observers. The Main character was tagged in the first frame and particle filter was used to track the object in the course of the video game. Color-histogram based tracking was used in this section.

A subset of freely available CRCNS eye-1 database [29] was used. Since we were interested in object tracking, we chose those game playing videos with a main character doing a task (c.f. Fig. 5). There is no constraint in these stimuli and the appearance of the target changes dramatically. Between 5 to 8 normal, young adult human subjects have freely viewed the stimuli while an ISCAN RK-464 eye-tracker recorded their eye movements with 240Hz sampling frequency. Images were shown in 640×480 pixels resolution. As a metric to quantify how well a model predicts the actual human eye focusing positions, we used the normalized scan-path saliency (NSS) used in [27].

We compared our proposed saliency method based on tracking ("T") with the following models: 1) Inter-Observer model. This is not a really a computational model but a map that is constructed from eye fixations of other subjects over the same frame of the movie. Eye movements of other subjects (each one convolved with a small Gaussian filter) is used to predict the eye position of the remained subject in a leave-one-out way, 2) a classic saliency model (denoted CIOFM) consisted of contrast, intensity, orientation, flicker and motion channels [16], and 3) Motion channel alone ("M"), 4) Bayesian surprise (denoted surprise) [28]. Since we observed that human subjects do not always fol-

Table 1. Detection results for both first-frame and clustering cases.

	Object	# of frames in train set	# of frames in test set	# of Clusters	Detection rate (Train)		Detection rate (Test)	
					first-frame case	clustering case	first-frame case	clustering case
1	Gadget	1200	1500	3	91.6	97.6 (6)	66.6	96.7 (30.1)
2	Cell	2400	1200	8	82.9	95.6 (12.7)	49.6	95 (45.4)
3	Drill	2400	3300	10	67.1	93.5 (26.4)	71.3	80.3 (9)
4	Cylinder	2665	2100	13	78.3	82.6 (4.3)	36.8	69.2 (32.4)
5.a	Box^1	75	75	2	61.3	100 (38.7)	50.6	88 (37.4)
5.b	-	-	53	-	-	-	40	61.3 (21.3)
5.c	-	-	43	-	-	-	28	45.3 (17.3)
6	Box^2	100	100	3	63	100 (37)	67	90 (23)
7	Box^3	97	101	3	52.5	96.9 (44.4)	40.6	92 (51.4)
8	Box^4	50	65	2	100	100 (0)	100	100 (0)
9	Can^1	70	70	3	61.4	100 (38.6)	44.3	82.8 (38.5)
10.a	Can^2	75	100	3	64	100 (36)	57	78 (21)
10.b	-	-	100	-	-	-	67	90 (23)
11	$Person^1$	84	100	3	96.4	100 (3.6)	86	100 (14)
12	$Person^2$	158	161	5	41.8	93 (51.2)	87.5	92.5 (5)

low the main character and some bottom-up salient regions sometimes grab attention we also combined Tracking model and bottom-up models to build up two other models: 5) Tracking model plus (pixel-wise addition) motion channel ("MT") and 6) Tracking plus CIOFM ("CIOFMT").

Results of saliency modeling and eye movement prediction are shown in Fig. 5. Average NSS score for each game and overall all games are shown. Our results indicate that "CIOFMT" model performed slightly better (NSS = 1.01) than MT model (NSS = 0.98) and pure tracking (T) model (NSS = 0.85) but all significantly better than other bottom-up models (CIOFM, M and Surprise) averaged over all games. Inter-observer which is the agreement among subjects is the best model. These results imply that our proposed method can estimate human visual attention with high accuracy. A video demonstrating saliency modeling results is included as a supplementary material.

7. Conclusions

In this paper, we presented an approach for object and person tracking that takes into account changes in background and scene context. The descriptor of an object is updated based on the cluster of the frame it appears in. In some cases, e.g Cylinder, Cell and $Person^2$, the object is occluded for a while but the adaptation approach is able to find and track it after it reappears. One advantage of this approach is its ability to handle situations when an object appears in different forms in different background as for Cell.

In comparison to the basic approach [8] without particle adaptation, our approach shows enhanced detection results. The detection rate in the clustering case is higher than the first-frame case when the descriptor of the first frame is applied to all frames of the test set (see Fig. 4 and Table 1).

We also showed that a particle filter based on the histogram of color features performs better than bottom-up saliency models. Best results were achieved where tracking model was combined with bottom-up salient regions (here addition). The best model is the inter-observer model again indicating that the best model of saliency over movies (as well as static scene) is the eye movements of other subjects. Our results highlight that developing tracking approaches with higher accuracy not only has promises in robotics and computer vision but could be also used to predict human overt attention and eye movements.

References

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

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," CVPR, 2000. 1

[3] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, 1998. 1

[4] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," ECCV, 2002. 1

[5] P. Pérez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles," *Proc. of the IEEE*, vol. 92, no. 3, 2004. 1

[6] S. Frintrop and M. Kessel, "Most salient region tracking," ICRA, 2009. 1

[7] S. Frintrop, A. Königs, F. Hoeller, and D. Schulz, "A Component-based Approach to Visual Person Tracking from a Mobile Platform" in *Springer International Journal of Social Robotics*, 2009. 1

[8] S. Frintrop, "General object tracking with a component-based target descriptor," ICRA, 2010. 1, 2, 5, 7

[9] T. Mathes and J. H. Piater, "Robust non-rigid object tracking using point distribution manifolds," DAGM, 2006. 1

[10] F. Tang and H. Tao, "Object tracking with dynamic feature graph," in *Proc. of the IEEE Workshop on VS-PETS*, 2005. 1

[11] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," IJCV, vol. 29, pp. 5–28, 1998. 2, 4

[12] S. Frintrop, "VOCUS: a visual attention system for object detection and goal-directed search," Ph.D. dissertation, 2005. 2

[13] R. Adams and L. Bischof, "Seeded region growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 16, no. 6, pp. 641 – 647, 1994. 2

[14] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, 2004. 2

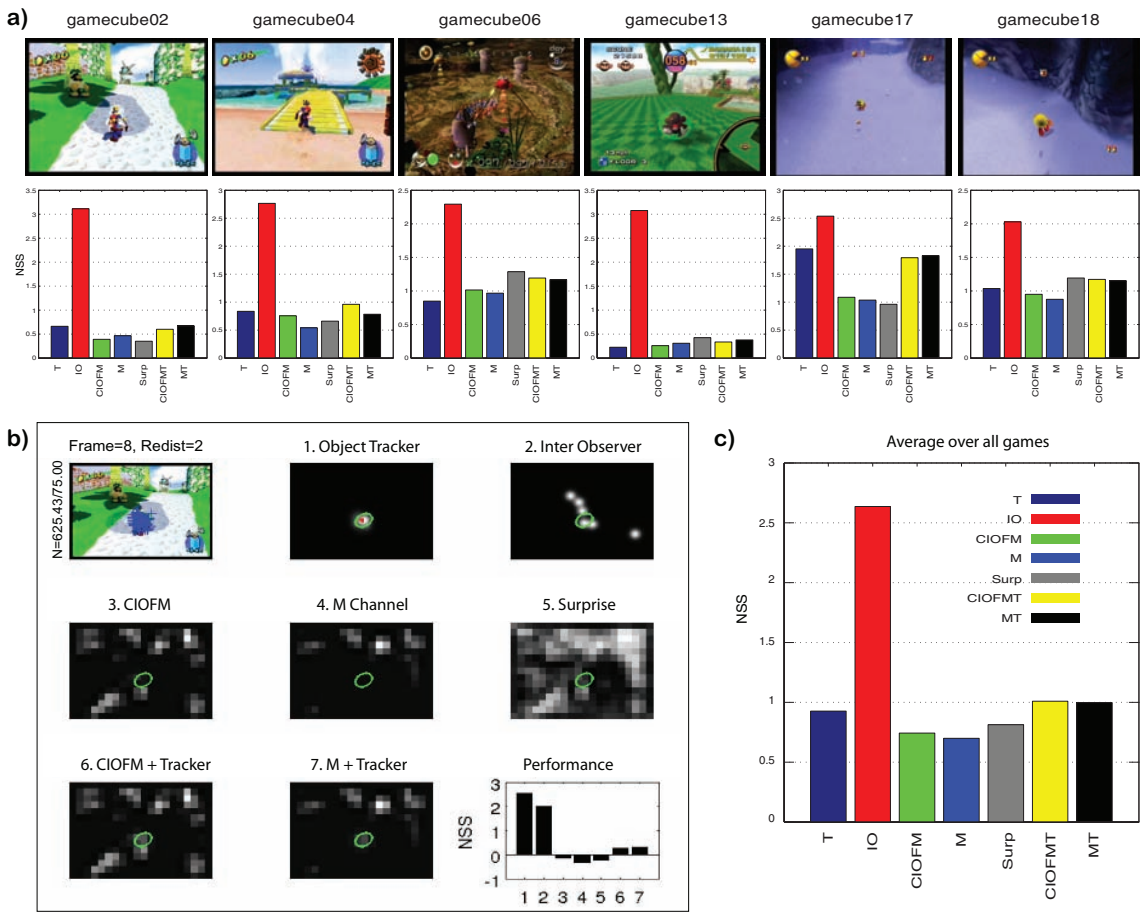

Figure 5. Saliency model based on tracking. a) sample frames from six game stimuli used in the experiments: Super Mario Sunshine (left two), Pikmin, Super Monkey ball, PacMan World (last two). Below each sample frame is the average NSS score over 1668, 1082, 2483, 687, 1863, 1548 frames, respectively for all mentioned models. b) A sample frame of Mario Sunshine game with particles overlaid. Sample saliency maps of models are also shown. The panel at the bottom-right is the instantaneous NSS score for this frame. Since subjects did not agree much in this frame NSS score for IO model is smaller than Tracking model. NSS scores for CIOFM, M and Surprise are negative indicating that bottom-up salient stimuli do not capture task-relevant attention, however when adding saliency map of Tracking model to this models NSS score increased to above 0. c) Average NSS score over all six games. As it shows CIOFM + Tracking model achieved the best score followed by Motion + Tracking. Tracking alone is higher than other pure bottom-up saliency models indicating that subjects most of the time tracked the main character in these games. There is still a big difference in performance of models and Inter-Observer model (more than 1.5 difference in NSS score). It means that there are some other task-relevant factors that has not been captured leaving room for future works. Over exploring games like Mario Sunshine and Pikmin NSS scores of models are small while in others performances are closer to Inter-Observer model.

[15] S. Theodoridis and K. Koutroumbas, "Pattern recognition." Elsevier, USA, 2006. 4

[16] 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. 6

[17] R. J. Peters and L. Itti. "Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention," *CVPR*, 2007. 2

[18] C. Teuscher and J. Triesch, "To each his own: the caregiver's role in a computational model of gaze following," *Neurocomputing*, vol. 70, pp. 2166–2180, 2007. 2

[19] L. Itti, C. Koch, "Computational Modelling of Visual Attention," *Nature Reviews Neuroscience*, vol. 2, no. 3, pp. 194-203, 2001. 2

[20] V. Navalpakkam, J. Rebesco, and L. Itti, "Modeling the influence of task on attention," *Vision Res*, vol. 45, no. 2, pp. 20–231, 2005. 2

[21] A. Borji, M. N. Ahmadabadi, B. N. Araabi and M. Hamidi, "Online learning of task-driven object-based visual attention control," *Image Vision Computing*, vol. 28, no. 7, pp. 1130-1145, 2010. 2

[22] M., Hayhoe and D. Ballard, "Eye movements in natural behavior," *Trends in Cognitive Sciences*, vol. 9, pp.188-194, 2005.

[23] N. Sprague, D. H. Ballard, "Eye Movements for Reward Maximization," *NIPS*, 2003. 2

[24] N. Mennie, M. Hayhoe, and B. Sullivan, "Look-ahead fixations: anticipatory eye movements in natural tasks," *Experimental Brain Research*, 179, 427–442, 2007. 2

[25] A. Oliva, and A. Torralba, "Modeling the Shape of the Scene: a Holistic Representation of the Spatial Envelope," *International Journal in Computer Vision*, vol. 42, pp. 145–175, 2001. 2

[26] J. M. Wolfe, and T. S. Horowitz, "What attributes guide the deployment of visual attention and how do they do it?" *Nat. Rev. Neurosci.*, vol. 5, pp. 1–7, 2004. 1, 2

[27] R. Peters, A. Iyer, L. Itti, and C. Koch, "Components of bottom-up gaze allocation in natural images," *Vision Research*, vol. 45, 2005. 6

[28] L. Itti, P. Baldi, "Bayesian Surprise Attracts Human Attention," *NIPS*, 2006. 6

[29] http://crcns.org/data-sets/eye/eye-1 6

[30] K. Nummiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter", *Image and Vision Computing*, vol. 21, 2003. 1, 3