Adaptive Real-Time Video-Tracking for Arbitrary Objects

Dominik A. Klein, Dirk Schulz, Simone Frintrop, and Armin B. Cremers

***Abstract*— In this paper, we present a visual object tracker for mobile systems that is able to specialize to individual objects during tracking. The core of our method is a novel observation model and the way it is automatically adapted to a changing object and background appearance over time. The model is integrated into the well known Condensation algorithm (SIR filter) for statistical inference, and it consists of a boosted ensemble of simple threshold classifiers built upon center-surround Haar-like features, which the filter continuously updates based on the images perceived. We present optimizations and reasonable approximations to limit the computational costs. Thus, the final algorithms are capable of processing video input at real-time. To experimentally investigate the gain of adapting the observation model we compare two different approaches with a non-adapting version of our observation model: maintaining a single observation model for all particles, and maintaining individual observation models for each particle. In addition, experiments were conducted to compare system performances between the proposed algorithms and two other state of the art Condensation based tracking approaches.**

I. Introduction

To track arbitrary objects is a key ability for autonomous agents to fulfill many different tasks like surveillance, guiding or following as well as interacting with and learning from humans. Many successful and accurate object tracking approaches have been proposed in recent years (see survey in [1]). However, many of them are not applicable for the tasks of mobile robots, because the domain violates several of the underlying assumptions. There is no static background and no fixed target appearance and the image quality can be bad due to insufficient illumination or glare. In some applications one cannot build a complex target model off-line, because the kind of object to track is not known in advance. Generally, one does not have a set of calibrated cameras for 3D-reconstruction. And finally, the computational power is very limited because of small form factors and the available energy, but at the same time quick reactions are needed when interacting with a rapidly changing environment.

For these reasons, feature based kernel tracking approaches are mostly applied in the domain of mobile robotics. These techniques either build a pixel-wise or a spatial model of the target's characteristics from different features such as intensity and color cues or corners and edges. Statistical inference methods like Kalman filtering, Mean Shift [2] or particle filters [3] are applied to evaluate the temporal evolution of the probability density function of the state of the target object.

The main challenge is to build an accurate model of the target's appearance that also generalizes to possible future appearances. One way to achieve this for spatial models is to prefer features that best discriminate the target from the background [4], for example by learning a binary classifier on features. Another way is to integrate discriminability already into the feature computation, as cognitive observation models do [5][6]. Compared to spatial models, pixel-wise models are said to deal better with non-rigid objects like persons [7]. As they do not rely on fixed spatial properties, they generalize better to target transformations. However, shape transformations are only one source for changes of the object and background appearance over time. To keep up with the various changes possibly occurring in a real-world scenario, we believe it is best not to rely on a fixed target model, but to adapt the model over time. This way, spatial target properties can be updated as well and strengthen the significance of features.

Adapting the observation model during tracking is not straightforward, because to ensure the correct adaption the exact target location within a training image needs to be known. Otherwise the model may diverge from the real target over time. Several groups investigate in how to adapt the target appearance model. Han et al. [8] introduced a sequential kernel density approximation technique based on mean-shift, that is used to update a target appearance model on-line. Lei et al. [9] try to adapt an off-line learned ensemble classifier of a particular object class to the changing appearance of a tracked instance of such class. Avidan [7] presents an algorithm to adapt the constituent parts and combination of an ensemble of classifiers itself to new appearances. Grabner et al. [10] demonstrated a semi-supervised on-line learning scheme to tackle the problem of uncertain class assignments of training examples collected while tracking the object.

Instead, we present a classifier-based approach that trains threshold classifiers on spatially distributed Haar-like center-surround features which are boosted to select and combine the most discriminative ones into a strong classifier. The initial target appearance model is quickly learned from a single frame and the resulting classifier is used to detect the most likely target position in the following frame. For this purpose, the confidence of the classifier is converted to a likelihood function of the target state that is used as the observation model within a Condensation-based tracker. Subsequently the classifier is adapted to the object and background appearance in the new frame via fast re-learning, where robustness is achieved by taking the different loca-

Dominik A. Klein, Simone Frintrop, and Armin B. Cremers are with the Intelligent Vision Systems Group, Dept. of Computer Science III, Rheinische Friedrich-Wilhelms-Universität Bonn, 53117 Bonn, Germany kleind@iai.uni-bonn.de

Dirk Schulz is head of the Unmanned Systems Group, Fraunhofer FKIE, 53343 Wachtberg, Germany

tion hypotheses of condensation filter into account during this process. This approach leads to a precise and flexible tracker that is quickly applicable to track arbitrary objects in unknown environments in real-time. Currently, the system works on video data from a freely moving hand-held camera. Thus, it is also ready to be mounted on a mobile robot.

In our experiments we compare two different adaptation schemes, one that adapts a single observation model based on the expected target state provided by the Condensation filter, and a second one that maintains individual models for each particle conditioned on the particles' unique state histories. We evaluated the approach in different settings to demonstrate the advantage of the adaptation techniques in comparison to our own classifier-based non-adaptive approach, but also in comparison to other non-adaptive tracking methods. We tested the ability of the methods to deal with perspective transformations, background changes, occlusions, illumination changes and more. It shows that the performances of the adapting approaches are considerably superior to the other, non-adapting tracking approaches.

In the following, we first give an overview on the particle filter based visual tracking system (Sec. II). In Section III, we explain our classifier-based observation model and how it is adapted. Section IV presents experimental results. We finally conclude in Section V.

II. The Visual Tracking System

The visual tracking system is based on the Condensation algorithm [3], a sequential Monte Carlo method also known as particle filter or Sampling Importance Resampling (SIR) filter. A distribution $p(X)$ of the state of the tracked object is approximated by maintaining a set of weighted particles (samples) $S_t = \{s_t^j\}, j \in \{1 \dots J\}$ over time, where each particle $s_t^j = (\mathbf{x}_t^j, \pi_t^j)$ consists of its state vector $\mathbf{x}_t^j$ and an importance weight π_t^j. The set of particles is updated from one frame to the next by the following recursive procedure: first, a new sample set S_t is drawn with replacement from the previous set S_{t-1}, where a sample s_{t-1}^i from the old set is chosen with probability proportional to its weight π_{t-1}^i. Second, for each sample a new state $\mathbf{x}_t^j$ is determined by sampling from the motion model $p(X_t|X_{t-1} = \mathbf{x}_{t-1}^i)$, and finally the measurement of the new frame Z_t is integrated by updating the importance weights π_t^j with the likelihood of the observation, i.e. $\pi_t^j = p(Z_t|X_t = \mathbf{x}_t^j, Z_0, Z_1 \dots Z_{t-1})$. The likelihood depends on all frames $Z_0, \dots, Z_{t-1}$ because the observation model is adapted over time. In case of a static model we have $\pi_t^j = p(Z_t|X_t = \mathbf{x}_t^j, Z_0)$.

The observation model is the core of our approach. Before we present it in detail in Sec. III, we will first briefly describe the remaining parts of the algorithm.

A. The Object State Space

The state of a particle is modeled as vector

$$\mathbf{x} = (x, y, w, h, v_x, v_y, C)^T \ ,$$

in which x, y is the position of the tracked object in the image with its respective first moments v_x, v_y and w, h are the dimensions of the target rectangle. C is the particle's object classifier (to be described in Sec. III) that determines its observation model.

B. Initialization

In the beginning, the target rectangle x, y, w, h in the first frame must be given to the system. For instance a gesture recognition module could pass the information about the object of interest to the system, or, like in our case, the user marks the target rectangle manually. A single binary classifier C is learned from the initial target and background to initialize the observation models of all particles. While x, y, w, h and C are identical for all J particles, their velocities v_x, v_y are sampled randomly from Gaussian distributions modeling the error in the different dimensions according to the motion model. The particle weights are initialized to $\pi_0^j = \frac{1}{J}$.

C. Motion Model

We apply a first order autoregressive motion model to predict particle positions. The estimate of the new state of a particle is a linear extrapolation of the previous state plus white Gaussian noise. In other words we calculate

$$\begin{aligned} v_{i,t} &= v_{i,t-1} + \mathrm{G}\left(0, \sigma_i^2\right), \qquad i \in \{x, y\}, \\ x_t &= x_{t-1} + v_{x,t}, \\ y_t &= y_{t-1} + v_{y,t}, \\ w_t &= w_{t-1} + \mathrm{G}\left(0, \sigma_w^2\right), \\ h_t &= h_{t-1} + \mathrm{G}\left(0, \sigma_h^2\right) \ . \end{aligned} \tag{1}$$

Within our experiments (cf. Sec. IV) we used $\sigma_x = \sigma_y = 6.4$ and $\sigma_w = \sigma_h = 0.64$.

To recover the state of the tracked object, the current state of the target is estimated as the weighted average over the states of the particles, hence

$$(\bar{x}, \bar{y}, \bar{w}, \bar{h})^T = \sum_{j=1}^{J} \pi_t^j \cdot (x_t^j, y_t^j, w_t^j, h_t^j)^T. \tag{2}$$

D. Observation Likelihoods

For the weighting of the particles we need to compute the likelihood $p(Z_t|X_t = \mathbf{x}_t^j, Z_0, Z_1 \dots Z_{t-1})$. In our case, we determine this value for each particle based on its binary classifier C_t^j. This classifier decides between background and target at a given image location. It is a continuous function of a target rectangle (x, y, w, h) and an image Z that returns values between 0 and 1. We employ an exponential function to convert the classifier responses to observation likelihoods, i.e. we compute

$$\pi_t^j = p(Z_t|X_t = \mathbf{x}_t^j, Z_0, Z_1 \dots Z_{t-1}) \tag{3}$$

$$= c \cdot \exp\left(\lambda \cdot C_t^j(x_t^j, y_t^j, w_t^j, h_t^j, Z_t)\right). \tag{4}$$

Here, c is a normalization constant which ensures that the new particle weights add up to 1. In Eq. 4 it is assumed that that the classifier C_t^j is sufficient statistics for the images (and the objects state history). The exponential weighting function was proposed in [11]; it emphasizes the reward of a classifier result with high confidence in comparison to a lower one with

lower confidence. The influence of exponential weighting is adjusted by λ. We chose $\lambda = 20$ as suggested in [11].

The observation model is the most important component since it assesses which hypotheses should be followed and which ones will die out. We will now explain, how the classifier-based model operates and how it is adapted over time.

III. The Ensemble Classifier Based Observation Model

Ensemble techniques like boosting have become popular for classification during the last years, because it was shown that such classifiers can be precise and operate very fast [12][13]. To adapt these techniques to real-time tracking they must be optimized for very short learning times as well.

A. The Initial Classifier

Gentle AdaBoost [14] is used to build a strong classifier consisting of a weighted linear combination of n weak classifiers. In our case, weak classifiers are simple threshold classifiers on Haar-like center-surround features varying in size, relative position and RGB color channels. Because the representation of the tracked object is a rectangle flexible in position, size and aspect ratio, we define features relative to an object coordinate system that is transformed to image coordinates for feature computation as illustrated in Fig. 1. These kind of features based on differences of average intensities in upright rectangular regions can be computed in constant time using integral images [13]. Results from queries located between image pixels are interpolated bilinearly. We restrict the number of possible features to choose from to a pool of 539 in order to speed up the learning process. In our case, AdaBoost iteratively picks out the $n = 32$ best features based on a weighted set of training examples. For the initial classifier, the only positive example is given by the user and the negative examples are then randomly sampled from the remainder of the first frame. This way the observation model incorporates target and background information.

We introduce a new spatial constraint to the normal boosting algorithm and force AdaBoost to choose a spatially distributed set of weak classifiers. Therefore, we enforce that each quarter of the object window (top left, top right, bottom left, bottom right) is covered by one quarter of the weak classifiers chosen by the algorithm. To distribute the weak classifiers in this way during the iterative selection process, we reduce the pool of candidate classifiers to the ones centered in the quarters that have not yet reached their limit of $\frac{n}{4}$ weak classifiers. Although this constraint can prevent AdaBoost from selecting the optimal combination of weak classifiers for a given training set, we think that this spatial spreading strengthens the classifiers robustness and precision. For the same reason we prevent AdaBoost from choosing the same feature twice within one classifier.

B. Adapting the Observation Model

To adapt the observation model of a particle, we re-train its classifier from frame $t-1$ to t based on updated training sets. Because it would be inefficient to store the image data of all past frames and always calculate the feature results again when needed, we represent training examples as the set of all its feature results directly. Note that this is only possible because our pool of features is rather small. After the first frame particles start to evolve differently. However, at every step t in time the current particles will have some common ancestors due to resampling. Like in a pedigree, the further one looks back in time, the more of the current particles share common ancestors. We utilize this fact by sharing the past training data of akin particles if possible.

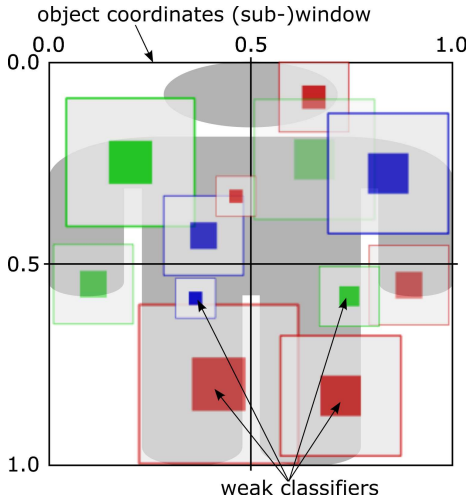

Fig. 1. The observation model is an ensemble of boosted weak classifiers on center-surround features.

From the current frame, we treat the estimate of the system or respectively the state of the particle as new positive training example and the remainder of the frame as source for new negative examples. Every observation model has a maximum capacity for positive and negative examples (we used $pos_{max} = 20$ and $neg_{max} = 100$). Until pos_{max} positive examples are obtained, we simply add the new ones. Thereafter, we always discard the positive example, the observation model from $t-1$ is most certain about, and keep all others. This approach has two positive effects: first, we introduce new object appearances to the classifier fast, this way. Second, the diversity of training examples will be increased for particles, whose target prediction is largely wrong. A rather inconsistent and diverse training set will produce less confident classifiers. This way, particles with the most self-similar history of positive examples will receive a higher rating from their classifiers and will have more successors after resampling.

As a special case, we always keep the positive example from the first frame given by the user in order to avoid the template drift problem [15][10]. Additionally, we always initialize this given first positive example with a higher weight when (re-)learning the classifier. The negative examples are treated differently. We replace the oldest $\lceil \frac{neg_{max}}{50} \rceil$ ones with randomly generated strong negative examples from the current background. This way the classifier is adapted to

new backgrounds. Note that in general it would be better to keep more training examples and to replace more negative examples for every new frame, but this is the maximal adjustment we are currently able to compute in real-time when using the per particle observation model.

Once the training sets are updated, they are used to adjust the observation model. Inspired by [7] a simplified boosting is conducted to select the optimal $n-k$ (we used $k=1$) weak classifiers out of the n weak classifiers of the strong classifier from $t-1$ and adapt their confidences. This is very fast because the set of possible features is small and it is done without re-learning the threshold of the chosen weak classifiers. Note that if the training set has changed in such a way that we cannot find $n-k$ weak classifiers with a still meaningful threshold, we update less. We continue with the constrained boosting algorithm we used for the initial classifier to select the remaining k optimally complementing weak classifiers from the whole pool of features.

Adapting the observation model is the most costly part of the algorithm. Because we cannot update every particle that survived the resampling in real-time, we concentrate on the up to ten best rated particles. This is still influential, because of their high weights those ten particles are the ancestors of more than 50% of the next generation of particles most of the times. Additionally we stop adapting the observation model if the confidence of the classifier on the new positive example is below a threshold θ, in order to handle temporary occlusions of the target object.

IV. Experiments and Results

In this section we present a qualitative comparison of five tracking approaches. All of them are based on particle filter techniques. We use the same particle filter implementation for all approaches and change only the observation model. The first approach is the well known color histogram tracking as described in [11]. The second is a more recent approach namely component-based tracking [6]. It computes center-surround feature maps from color and intensity and builds an object description from the relative positions of multiple local maxima and their circumference within these maps. The others are our Haar-like center-surround feature based classifiers. The first of these does not adapt its observation model after the first frame, the second holds and adapts only one observation model for all particles, and the third holds and adapts one observation model per particle.

We recorded nine test sequences with a total of 5485 frames (320×240 at 25fps) and manually marked the smallest rectangle containing the whole target object in each frame. Between this ground truth and the results of the approaches we measure the fraction of the intersection to the union. This measure is more precise than the distance of the centers of the rectangles, because it incorporates not only differences in position but also in size. For better comparison to other groups' results an overlap below 33.33% in a frame can be considered as a miss. We provide this data on our webpage[1] and kindly invite everyone to evaluate their own approaches with it.

[1] `http://www.iai.uni-bonn.de/~kleind/tracking/`

Seq.	# Fr.	average score [%]				
		Histo-gram	Multi-Comp.	n. ad. H.-cs	adapt. H.-cs	adapt. p. part. H.-cs
A.	601	**70.73**	63.24	38.35	65.06	59.35
B.	628	67.02	50.73	6.02	**79.01**	77.38
C.	403	47.58	63.71	89.33	90.66	**91.33**
D.	946	63.35	**76.39**	62.78	71.12	75.21
E.	304	78.21	77.42	83.12	84.49	**86.32**
F.	452	44.43	40.02	63.99	60.82	**68.32**
G.	715	46.27	49.62	34.34	**77.30**	71.16
H.	411	62.19	86.50	**95.79**	94.41	94.47
I.	1016	68.94	47.63	48.97	**75.02**	56.33
av.		60.97	61.70	58.08	**77.54**	75.54

TABLE I

Comparison of the five tracking methods based on color histograms, multi-component-descriptor, non adaptive and (per particle) adaptive Haar-like center-surround features.

The parameters are chosen to meet the demands for real-time tracking (not less than 25fps) on a modern CPU (Intel Q9550) with our slowest approach and are not altered between sequences. We used $J=2000$ particles for the experiments.

In the following we describe the test sequences (cf. Fig. 2) and explain the results shown in Fig. 3 and subsumed in Table I.

A. Rapidly Changing Object Appearance (Ball)

A red ball with white spots is kicked back and forth. While the histogram representation is well suited in this case and performs best, our approach is struggling a little to keep track of the locations of the white spots. However, the advantage of adapting our classifier is clearly visible.

B. Challenging Background Alterations (Cup 1)

A blue cup moves along a heavily cluttered background. The component-based model and our non-adaptive classifier lose the object when the background becomes mainly blue. The component-based model manages to recover afterwards. Our adaptive classifier models are able to learn the new background appearances and perform best.

C. Fast Moving Object and Size Changes (Juice)

A juice box stands on a table. The camera pans very fast. This results in quick object motion without bigger appearance changes. Thus our adaptive and non adaptive approaches differ very little. In the middle of the sequence the camera zooms out. Because of that the histogram degrades most, the component-based model also worsens whereas this has no effect on our Haar-like center-surround feature based approaches.

D. Non-rigid Object in an Outdoor Scene (Person 1)

A person is walking and turning around multiple times. All approaches successfully estimate the person's position, but the component-based model and our per particle adaptive

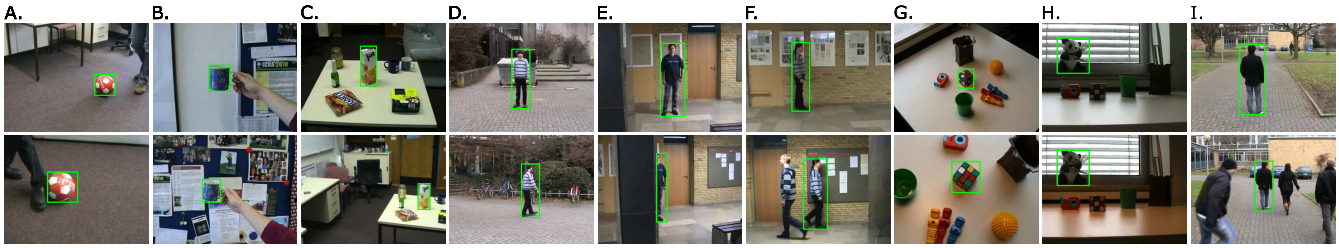

Fig. 2. The test sequences A. - I. . First row: each first frame with the region that was given the algorithms for initialization (green rectangles). Second row: an example frame with manually marked ground truth used for evaluation. See also the accompanying video.

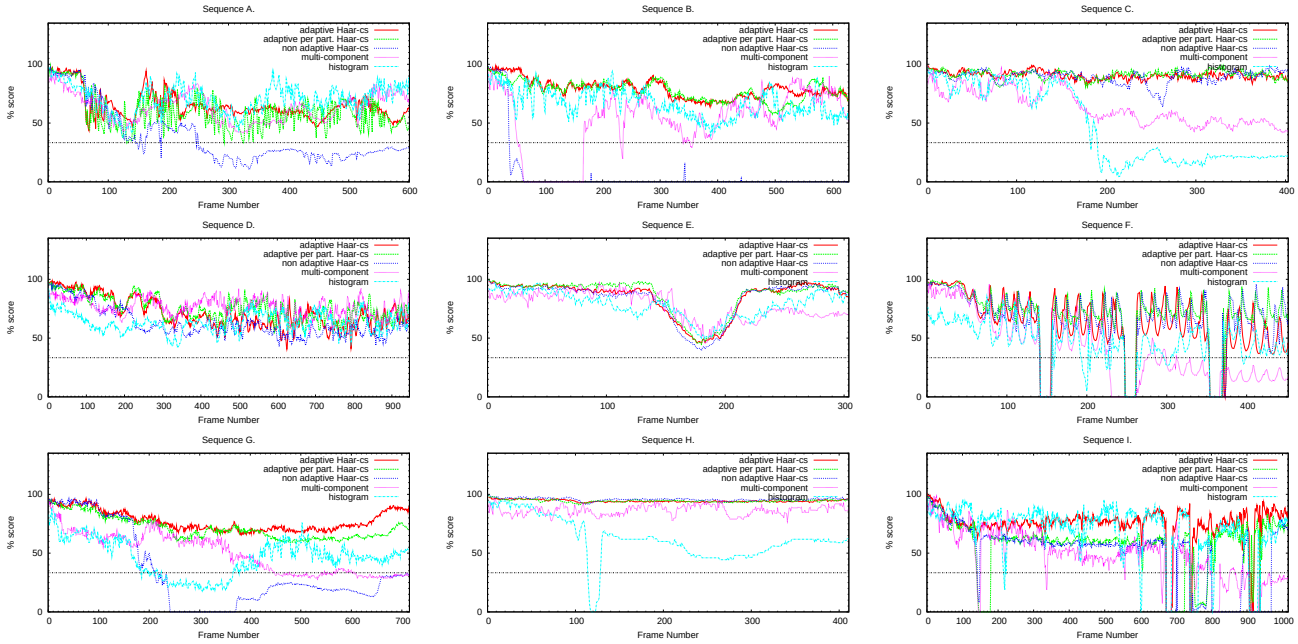

Fig. 3. Results on tracking the target object with four different observation models on the test sequences A. - I. . Plotted against the y-axis is the fraction of the intersection to the union between the rectangular area of the manually marked ground truth and the estimates of the systems. For better comparison to other group's results one can consider a score above $1/3$ as correct match and below as miss.

classifier are a little bit more precise in following the variation of the size of the person in the images.

E. Partial Occlusion (Person 2)

The person is stationary, but the camera moves so that the person gets half occluded and visible again. The noticeable valley in the graph is caused by this partial occlusion of the person. We marked only the visible parts of the person as ground truth, while all approaches tend to estimate the person's position and size behind the occluding object.

F. Full Occlusion of a Non-rigid Object (Person 3)

A person walks along a corridor and becomes fully occluded by a pillar three times. In these situations it is important to stop adapting the models if the object is not visible. Fortunately, it turned out that a simple confidence threshold on the classifier response is sufficient to handle such situations for our adaptive approaches (cf. Fig. 4). Note that the short oscillation in Fig. 4 between the first two full occlusions is caused by another person crossing. Interestingly, our non-adaptive approach is also very precise and superior to color histogram and component-based tracking. Likely this is because the person is seen from the side during the whole sequence and his upper part of the body appears constantly the same. The regular spike pattern shown by all approaches is because the ground truth width pulsates with every step of the person. The per particle adaptive classifier again is best to imitate this transformations.

G. Appreciable Viewpoint Changes (Rubik's Cube)

The camera pans around a Rubik's Cube from left to right and then flies over it. This causes heavy changes in shape and color of the object. Before the camera starts to move, adaptive and non-adaptive Haar-like center-surround feature based classifiers perform equally well and are superior to the other approaches. While our non-adaptive classifier starts to fail when viewpoint changes become larger, the adapting ones retain a good performance. Thanks to the rather uniform background, color histogram and component-based tracking

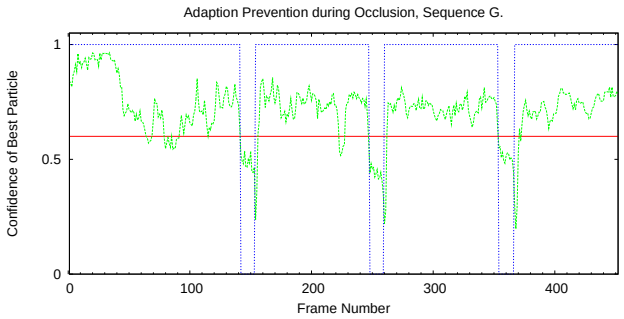

Fig. 4. Green: confidence of best particle. Red: adaption threshold $\theta = 0.6$. Blue: Object visible or fully occluded.

are also able to identify the object, but with a loss of precision.

H. Illumination and Backlight Changes (Panda)

During the sequence the sun-blinds are opened/closed and the artificial light is switched on and off, while the camera is not moving. The histogram is confused most, but recovers very quickly. The other approaches are robust against changes in lightening, while the component-based approach in general is less exact than our Haar-like center-surround feature based classifier.

I. Real-world Person Following Scenario (Person 4)

The camera follows a person walking outdoor while other persons cross him 13 times. Our global adaptive observation model is the only approach able to differentiate the persons and track the correct one all the time. Color histogram tracking performs also very well on this sequence.

In summary, one can say that the three non-adaptive approaches on average all perform similarly, but it depends strongly on the type of sequence which approach performs best (compare also the results in [6] and [16] for other types of sequences in which the component-based approach outperforms the histogram tracking clearly). The histogram tracking is the most general and is therefore not affected much by deformations of the target, even without adaption. On the other hand, it generally has problems with illumination changes which is often a problem in real-world robotic settings (cf. [16]). The component-based approach is currently not able to deal with rotations of the object, but it is mainly robust against illumination change and transformations in size. At the beginning of every sequence one can see that our Haar-like center-surround feature based classifier is the most exact observation model. This is an advantage if the scene does not change a lot. However, without adaption it does not generalize sufficiently well to deal with bigger changes. When adapting the model to new appearances this effect is compensated. Our global adaptive observation model turned out to be the most exact and most robust model as it was the only approach that was able to keep track of the targets in all test sequences. In theory, a per particle adapting observation model should be able to better deal with multi-modal distributions. But experiments showed that the classifier is so reliable that such situations are very rare. The advantages of the global adaptive observation model, speed and robustness, weigh more heavily, so we recommend this approach. Note that for an adaptive observation model it is beneficial if the different object appearances are introduced rather slowly to the model for the first time to enable a more proper adaption.

Please see the accompanying video for a visualization of the results of our proposed new visual tracking system.

V. Conclusion

In this paper we presented a new particle filter based approach for real-time video tracking of arbitrary objects. The heart of this new approach is the adapting observation model. For this, a strong classifier composed of an ensemble of Haar-like center-surround features is learned from a single positive training example with Gentle AdaBoost and quickly updated to new object and background appearances in every frame. The system deals with different objects and settings and is robust to perspective transformations, rotations and lightening conditions. Thus, it is disposed to the deployment on a mobile platform. In experiments we found that it considerably outperforms other methods.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.
[2] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, 2002.
[3] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
[4] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.
[5] S. Frintrop and M. Kessel, "Most salient region tracking," in *Proc. of ICRA*, 2009.
[6] S. Frintrop, "General object tracking with a component-based target descriptor," in *Proc. of ICRA*, 2010.
[7] S. Avidan, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261–271, 2007.
[8] B. Han, D. Comaniciu, Y. Zhu, and L. S. Davis, "Sequential kernel density approximation and its application to real-time visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1186–1197, 2008.
[9] Y. Lei, X. Ding, and S. Wang, "Adaboost tracker embedded in adaptive particle filtering," in *Proc. of ICPR*, 2006.
[10] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *ECCV*, 2008, pp. 234–247.
[11] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proc. of ECCV*, 2002.
[12] Y. Freund and R. E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.
[13] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.
[14] J. Friedman, T. Hastie, and R. Tibshirani, "Special invited paper. additive logistic regression: A statistical view of boosting," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–374, 2000.
[15] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 810–815, 2004.
[16] S. Frintrop, A. Königs, F. Hoeller, and D. Schulz, "A component-based approach to visual person tracking from a mobile platform," in *Journal of Social Robotics, Special Issue on People Detection and Tracking*, vol. 2, no. 1. Springer, Netherlands, 2010, pp. 53–62.