Exploiting contextual motion cues for visual object tracking

Stefan Duffner and Christophe Garcia

Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621, France

Abstract. In this paper, we propose an algorithm for on-line, real-time tracking of arbitrary objects in videos from unconstrained environments. The method is based on a particle filter framework using different visual features and motion prediction models. We effectively integrate a discriminative on-line learning classifier into the model and propose a new method to collect negative training examples for updating the classifier at each video frame. Instead of taking negative examples only from the surroundings of the object region, or from specific distracting objects, our algorithm samples the negatives from a contextual motion density function. We experimentally show that this type of learning improves the overall performance of the tracking algorithm. Finally, we present quantitative and qualitative results on four challenging public datasets that show the robustness of the tracking algorithm with respect to appearance and view changes, lighting variations, partial occlusions as well as object deformations.

Keywords: object tracking, adaptive particle filter, motion cues

1 Introduction

We consider the problem of automatically tracking a single arbitrary object in a video, where the algorithm is initialised in the first frame from a bounding box around the object to track. No prior knowledge about appearance, shape, or motion of the objects or the environment is used. Also, we focus here on *on-line* tracking, *i.e.* at each time step, only past and present but no future information is used. Applications for on-line visual object tracking are numerous, including, for example, video indexation, Human-Computer or Human-Robot Interaction, video-surveillance, traffic monitoring, or autonomous driving.

In real-world scenarios, this problem is challenging as the object to track may change considerably its appearance, shape, size, and pose in the image (like the articulated human body for example). Furthermore, the object can be partially occluded by itself, other objects, or the environment. The object may also move abruptly or in unpredictable ways. Finally, the environment, *i.e.* the image background, may change considerably and rapidly in videos from moving cameras and be affected by varying illumination.

Recent works [7,1,2,13,9] propose to tackle this problem by a tracking-by-detection framework, where a discriminative detector is trained with object and background samples. At each frame of the video, this detector is applied in a search window to estimate the current position of the object, and the model is updated using this estimate. The advantage of this approach is that no specific motion model needs to be designed and parameterised, and the output is deterministic. Classical tracking algorithms are based on recursive Bayesian filters like Kalman filters or particle filters [17,12,19]. These methods are able to estimate the posterior state distribution of the tracked object and allow for maintaining several state hypotheses. Usually, they explicitly integrate motion models used to predict the next object state by defining a probabilistic transition function independent from the image observations. Some particle filter techniques use some more advanced motion models, like [15], *i.e.* an optical flow-like dense parametric motion estimator with an affine model to propose new state values, as we propose in this paper. Also similar to this paper, parametric motion models have been used to estimate background (*i.e.* camera) motion [6] and segment the object region from the background, *e.g.* [24].

Other recently proposed approaches have also included this type of contextual motion information. For example, Yang *et al.* [23] introduced a method that, throughout a video, continuously discovers objects that move in the same direction as the tracked object by performing a motion correlation analysis. These auxiliary objects help to support and improve tracking by performing inference in a star-structured graphical model that includes their state. Spatial context has also been exploited by using supporters, *i.e.* other objects or feature points around the target in the image. Grabner *et al.* [8], for example, extended the well-known Implicit Shape Model by detecting feature points in the image that have a correlated motion with the target. These supporters are matched from frame to frame and their relative displacement vectors are updated on-line. Also, Wen *et al.* [21] proposed a method that detects supporters (here called contributors), *i.e.* interest points within a local neighbourhood around the target, in order to improve the tracking performance. Similarly, the approach proposed by Sun *et al.* [18] tracks "helper" objects using an on-line Adaboost detector, initialised manually at the first frame. Their relative position is learnt on-line and used to predict the target object's position. Finally, Dinh *et al.* [3] proposed a method using supporters as well as distractors, which are objects with similar appearance to the target. The distractors help to avoid confusion of the tracker with other similar objects in the scene, and they can possibly be used to reason about the objects' mutual occlusion. Supporters are not used directly for the target's state estimation but only to disambiguate between the target and its distractors. Hong *et al.* [10] recently proposed an approach based on the L1 tracker [13] that deals with distractors by automatically learning a metric not only between positive and negative examples but also within the collected negative examples, effectively replacing the originally proposed Euclidean distance.

The disadvantage with using supporting and distracting objects is that several objects need to be detected and tracked, which can be computationally expensive especially with a larger number of objects. Moreover, the success or failure of data association or, in some methods, matching local features points in successive video frames, heavily depends on the type of object to track and the surrounding background. This process can be error-prone and, in some situations, may rather harm the overall tracking performance. Finally, modelling the spatial, temporal, or appearance-based pairwise relationships between objects and/or interest points can lead to a combinatorial explosion and make the inference on the state space difficult.

To alleviate this problem, in this work, we propose a probabilistic method that dynamically updates the foreground and background model depending on distracting objects or image regions in the scene background. This contextual appearance information is extracted from moving image regions and used to train on-line a discriminative binary classifier that, in each video frame, detects the image region corresponding to the object to track.

Traditionally, these discriminative on-line classifiers used in tracking-by-detection approaches learn negative examples extracted from the image region surrounding the current target object region. This choice is motivated by the fact that the object will move only slightly from one frame to the other w.r.t. the background or other objects, and by computational speed. In contrast, our method uses a stochastic sampling process to extract negative examples from image regions that move. We call these: *contextual motion cues* (see Fig. 1). In that way, regions that correspond to possibly distracting objects are detected efficiently and early, *i.e.* without them having to be inside a search window and without scanning the whole image at each point in time. The contributions of this paper are the following:

– a method for on-line learning of a discriminative classifier using stochastic sampling of negative examples from contextual motion cues in videos,
– the integration of this incremental discriminative model in an efficient adaptive particle filter framework combining effectively several visual cues,
– a thorough evaluation on difficult public benchmarks experimentally showing the performance increase from this type of online learning as well as an improvement over state-of-the-art tracking methods.

2 Tracking algorithm

Our tracking algorithm is based on a recursive Bayesian framework implemented with a particle filter:

$$p(\mathbf{X}_t|\mathbf{Y}_{1:t}) = \frac{1}{C}\, p(\mathbf{Y}_t|\mathbf{X}_t) \times \int_{\mathbf{X}_{t-1}} p(\mathbf{X}_t|\mathbf{X}_{t-1}) p(\mathbf{X}_{t-1}|\mathbf{Y}_{1:t-1})\, \mathrm{d}\mathbf{X}_{t-1}\ , \qquad (1)$$

where C is a normalisation constant, $\mathbf{Y}_{1:t}$ are observations from time 1 to t, and $\mathbf{X}_t$ denotes the state at time t. Before describing the main contribution of the paper in section 3, for the sake of completeness, we will first describe the main elements of this model.

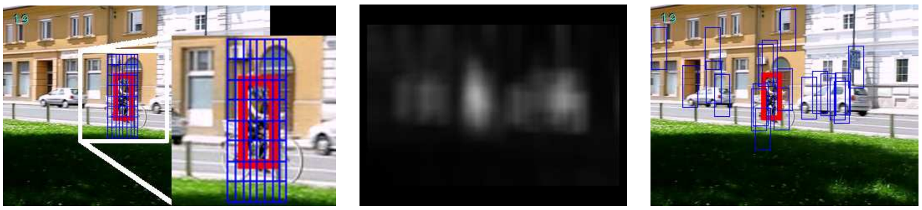

Fig. 1. Illustration of different sampling strategies of negative examples (blue). *Left:* traditional sampling at fixed positions within a search window around the object (red). *Middle:* the motion probability density function m (Eq. 11). *Right:* the proposed negative sampling from m.

2.1 Object state representation and inference

The state $\mathbf{X} = (x, y, v_x, v_y, s, e) \in \mathbb{R}^6$ of the object to track is described by an upright bounding box defined by the object's centre (x, y) in the image, its speed (v_x, v_y), scale (s), and eccentricity (e), *i.e.* the ratio of height and width. The state $\mathbf{X}_0$ is initialised manually (for each particle) by a bounding box around the object in the first frame. Then, for each video frame, the particle filter performs its classical steps of *predicting* particles $\mathbf{X}^{(i)}$ sampled from the proposal distribution $q(\mathbf{X}_t|\mathbf{X}_{t-1})$ and *updating* their weights according to the observation likelihood, state dynamics and proposal (see Section 2.2): $w_i = p(\mathbf{Y}_t|\mathbf{X}_t)\frac{p(\mathbf{X}_t|\mathbf{X}_{t-1})}{q(\mathbf{X}_t|\mathbf{X}_{t-1})}$, for each particle $i \in 1..N$. At the end of each iteration, the observation likelihood model parameters are updated using the mean particle of the posterior distribution $p(\mathbf{X}_x|\mathbf{Y}_{1:t})$, and systematic resampling is performed.

2.2 State dynamics and proposal function

The state dynamic model $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ is defined for each individual component of $\mathbf{X}$. The position and speed components of the object are described by a mixture of a first-order auto-regressive model with additive Gaussian noise and a uniform distribution allowing for small "jumps" coming from the proposal function (Eq. 2). A simple first order model is used for the scale and eccentricity parameters, s and e.

In order to cope with fairly complex motion of arbitrary objects in videos from a possibly moving camera, we use a proposal function composed of a mixture of three distributions:

$$q(\mathbf{X}_t|\mathbf{X}_{t-1}) = \beta_m p(\mathbf{X}_t|\mathbf{X}_{t-1}) + \beta_f p_f(\mathbf{X}_t|\mathbf{X}_{t-1}) + \beta_d p_d(\mathbf{X}_t|\mathbf{X}_{t-1}) \ , \tag{2}$$

where β_m, β_f and β_d define the mixture weights, and $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ is the state dynamics model. The function

$$p_f(\mathbf{X}_t|\mathbf{X}_{t-1}) = \mathcal{N}(\mathbf{X}_{t-1} + \mathbf{d}; 0, \boldsymbol{\Sigma}^f) \tag{3}$$

predicts the new state by performing a parametric robust motion estimation of the image region defined by $\mathbf{X}_t$ like in [14]. The output of this multi-level estimation is the differential vector $\mathbf{d}$ which updates position and scale. The last term:

$$p_d(\mathbf{X}_t|\mathbf{X}_{t-1}) = \mathcal{N}(\mathbf{X}^d; 0, \boldsymbol{\Sigma}^d) \tag{4}$$

uses the output $\mathbf{X}^d$ of a detector (see Section 3) that has been trained on-line and that is applied in the neighbourhood around $\mathbf{X}_t$ to predict the new object position and scale (as in [16]). See Section 4 for a summary of parameter values.

2.3 Observation likelihood

The observation likelihood function $p(\mathbf{Y}|\mathbf{X})$ is a geometric mean of three distributions corresponding to different visual cues described in the following:

$$p(\mathbf{Y}_t|\mathbf{X}_t) = \left(p_H(\mathbf{Y}_t|\mathbf{X}_t)\, p_S(\mathbf{Y}_t|\mathbf{X}_t)\, p_T(\mathbf{Y}_t|\mathbf{X}_t)\right)^{1/3} . \tag{5}$$

Histogram likelihood ratio. The histogram likelihood function is defined as a ratio of foreground and background likelihoods:

$$p_H(\mathbf{Y}_t|\mathbf{X}_t) = \frac{p_{FG}(\mathbf{Y}_t|\mathbf{X}_t)}{p_{BG}(\mathbf{Y}_t|\mathbf{X}_t)} , \tag{6}$$

where

$$p_{FG}(\mathbf{Y}_t|\mathbf{X}_t) = \exp\left(-\lambda_{FG}\sum_{r=1}^{9}(D^2[h_t^*(r), h(r, \mathbf{X}_t)])\right) , \tag{7}$$

is the foreground likelihood defined over a grid of 3×3 regions r. D computes the Bhattacharyya distance between the HSV histograms h_t extracted from state $\mathbf{X}_t$ and the respective reference histograms h_t^* initialised from the first frame, and λ_{FG} is a constant. Similarly, the background likelihood:

$$p_{BG}(\mathbf{Y}_t|\mathbf{X}_t) = \exp\left(-\lambda_{BG}(D^2[\hat{h}_t^*, \hat{h}(\mathbf{X}_t)])\right) , \tag{8}$$

is computed over the image region surrounding the object's bounding box.

Global colour segmentation likelihood. In addition to the more local colour models with one histogram per object part, we also use a global colour histogram model based on a pixel-wise colour segmentation of foreground and background. To this end, as above, HSV colour histograms with separate colour and greyscale bins are extracted, one inside the current bounding box of the object, and one around it. Then a probabilistic soft-segmentation is performed computing the probability $p(c_i|z_i)$ of each pixel i inside a search window belonging to the foreground $c = 1$ or background $c = 0$ given its colour z_i.

Then, the likelihood function is defined as:

$$p_S(\mathbf{Y}_t|\mathbf{X}_t) = \frac{\exp(-\lambda_S S_{FG}(\mathbf{X}_t)^2)}{\exp(-\lambda_S S_{BG}(\mathbf{X}_t)^2)} , \tag{9}$$

where λ_S is a constant, S_{FG} is the proportion of foreground pixels, *i.e.* for which $p(c = 1|z) > 0.5$, *inside* the object's bounding box and S_{BG} is the proportion of foreground pixels *outside* the bounding box.

Texture likelihood. The likelihood $p_T(\mathbf{Y}|\mathbf{X})$ is based on the (greyscale) texture of the object to track. A discriminative classifier is trained at the first frame using the object region as positive and the background regions as negative examples. Then, the classifier is updated at each iteration collecting positive and negative examples from the foreground and background respectively (see Section 3). We use the On-line Adaboost classifier presented by Grabner *et al.* [7] that uses Haar-like features, but any other on-line classifier could be used as well.

The likelihood is based on the detector's confidence $c_D \in [0, 1]$ for the image patch defined by $\mathbf{X}_t$:

$$p_D(\mathbf{Y}_t|\mathbf{X}_t) = \exp(-\lambda_D(1 - c_D)^2) \ . \tag{10}$$

3 Model adaptation with contextual cues

As mentioned earlier, in the particle filter, we use a binary discriminative classifier based on the On-line Adaboost (OAB) algorithm [7] for proposing new particles (Eq. 4) as well as for evaluating the observation likelihood (Eq. 10). The classifier is trained with the first video frame using the image patch inside the object's bounding box as a positive example and surrounding patches within a search window as negative examples. Then, the authors propose to update the classifier at each tracking iteration using the same strategy for extracting positive and negative examples. We refer to [7] for details on the model.

3.1 Background sampling

We propose to sample negative examples from image regions that contain motion and thus likely correspond to moving objects (see Fig. 1). The idea is that these regions may distract the tracker at some point in time. Therefore it is preferable to learn these negative examples as early as possible, *i.e.* as soon as they appear in the scene. To this end, we first compensate for camera motion between two consecutive frames using a classical parametric motion estimation approach [14]. We apply a three-parameter model to estimate the translation and scale of the scene, and then compute the intensity differences for each pixel with its corresponding pixel in the previous frame. This gives an image $M(x, y)$ approximating the amount of motion present at each position (x, y) of the current frame of the video. We then transform this image into a probability density function (PDF) $m(x, y)$ over the 2-dimensional image space:

$$m(x, y) = Z^{-1} \sum_{(u,v)\in\Omega(x,y)} M(u, v) \ , \tag{11}$$

where $\Omega(x, y)$ defines an image region of the size of the bounding box of the object being tracked, centred at (x, y), and Z is a constant normalising the density function to sum up to 1. Thus, $m(x, y)$ represents the relative amount of motion inside the region centred at (x, y). Finally, N^- image positions (x, y) are sampled from this PDF corresponding to rectangles centred at (x, y), where, statistically, regions with high amount of motion are sampled more often than static image regions. This process is illustrated in Fig. 1.

3.2 Classifier update

The N^- image patches corresponding to the sampled regions as well as the positive example coming from the mean particle of the tracker are then used to update the classifier. In this case, the OAB method needs a balanced number of positives and negatives, thus the positive example is used N^- times, alternating positive and negative updates.

The advantage of sampling positions from these motion cues is that we don't need to care about explicitly detecting, initialising, tracking, and eventually removing a certain number of distracting objects at each point in time. Note that, we could also sample regions of different scales but as scale does not change rapidly in most videos the benefit of this would be relatively small. Note also that the PDF could as well include appearance similarity with the tracked target. However, this would considerably increase the computational complexity.

4 Experiments

4.1 Parameters

The following tracking parameters that have been used for all the experiments:

$\hat{\Sigma}$	$\bar{\Sigma}$	$\Sigma^{f/p}$	β_m	β_f	β_d	λ_{FG}	λ_{BG}	λ_S	λ_D
$(7, 7)$	$(0.001, 0.001)$	$(1, 1, 10^{-4}, 10^{-4})$	0.7	0.2	0.1	120	36	0.1	10

The variances for x and y values are scaled by $\frac{w}{200}$, w being the current width of the bounding box. We should highlight that only 100 particles have been used throughout all experiments. This turns out to be sufficient due to our effective proposal and discriminative likelihood functions.

4.2 Datasets

We performed a quantitative evaluation on 4 challenging public tracking datasets:

Babenko[1] [2] contains 8 videos of objects that undergo mostly rigid deformations and some rather large lighting variations and partial occlusions. Most of the videos are in grey-scale format (except "David", "Girl", and "Face Occl. 1").

Non-rigid objects[2] is a more challenging dataset composed of 11 videos showing moving objects that undergo considerable rigid and non-rigid deformations.

VOT2013[3] is the Visual Object Tracking (VOT) Benchmark 2013 [11] containing 16 videos that show a large variability in terms of camera motion, illumination change, occlusion, object size, and motion. Four of these sequences (David, diving, face, jump) are also part of the first or second dataset.

VOT2014[3] contains 25 challenging videos including eight from VOT2013.

[1] http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml
[2] http://lrs.icg.tugraz.at/research/houghtrack/
[3] http://votchallenge.net/

Note that long-term tracking datasets like LTDT2014 are not suitable for evaluating our approach as these videos contain longer periods of full occlusion which requires the algorithm to be able to re-detect the tracked object after occlusion.

4.3 Evaluation

We performed several experiments with different evaluation protocols. For the first two datasets we evaluated the robustness of the proposed algorithm by measuring the proportion of correctly tracked frames. A frame is counted as correct, if the tracking accuracy $A = \frac{R_T \cap R_{GT}}{R_T \cup R_{GT}}$ is greater than a threshold, where R_T is the rectangle corresponding to the mean particle from the tracking algorithm, and R_{GT} is the ground truth rectangle surrounding the object. We set the threshold to 0.1 in order not to penalise fixed-size, fixed-ratio trackers in our comparison. For every experiment and video sequence, the proposed algorithm has been run 5 times and the average result is reported.

For the VOT datasets, we used the evaluation protocol of the VOT2013 benchmark, which measures accuracy and robustness. For evaluating the accuracy, the measure A, defined above, is used. The robustness is measured in terms of number of tracking failures, where trackers are re-initialised after failures. Every video sequence is evaluated 15 times and the average results are reported. In addition to this "baseline" experiment, there are two other experiments using the same data. In the "region-noise" experiment the initial bounding box is randomly, slightly shifted for each run, and in the "greyscale" experiment, each video is transformed into greyscale format. See [11] for more details.

4.4 Results

In the first experiments, we evaluated four different strategies for the collection of negative examples of the discriminative OAB classifier (*c.f.* section 3):

fixed: N^- negatives are taken from fixed positions around the positive example inside the search window, which is twice the size of the object's bounding box.
fixed+random: $N^-/2$ examples are taken from fixed position (as for "fixed"), and $N^-/2$ examples are sampled from random image positions.
motion: N^- negative examples are sampled from the contextual motion distribution m (Eq. 11).
fixed+motion: $N^-/2$ examples are taken from fixed positions, and $N^-/2$ examples are sampled from the contextual motion distribution.

In any case, the negative examples do not overlap more than 70% with the positive ones in the image.

Table 1 and 2 show the results for the first two datasets in terms of the percentage of correctly tracked frames. In most cases, the sampling of negative examples from the contextual motion PDF, *i.e.* "motion" and "fixed+motion", improves the tracking performance. For the Babenko sequences, the improvement is smaller because there are not many other moving objects that can distract

	fixed	fixed+rand.	motion	fixed+mot.
David	**62.6**	61.9	60.6	61.5
Sylvester	49.8	59.6	**81.6**	78.7
Girl	67.9	44.4	73.0	**74.2**
Face Occlusions 1	98.4	**100.0**	**100.0**	**100.0**
Face Occlusions 2	97.7	95.8	95.0	**98.4**
Coke	88.8	92.5	92.9	**93.2**
Tiger 1	**60.3**	58.9	59.7	59.7
Tiger 2	90.4	93.2	**97.3**	**97.3**
average	76.99	75.80	82.51	**82.88**

Table 1. Babenko sequences: percentage of correctly tracked frames with fixed negative sampling, sampling from motion, combined fixed+random, and fixed+motion sampling.

	fixed	fixed+rand.	motion	fixed+mot.
Cliff-dive 1	**100.0**	**100.0**	**100.0**	**100.0**
Motocross 1	75.9	84.7	94.1	**99.1**
Skiing	98.1	89.6	96.4	**99.2**
Mountain-bike	**100.0**	**100.0**	**100.0**	**100.0**
Cliff-dive 2	51.8	70.2	63.3	**73.8**
Volleyball	99.9	88.5	**100.0**	99.9
Motocross 2	**100.0**	**100.0**	**100.0**	**100.0**
Transformer	91.1	92.9	**94.4**	91.5
Diving	75.0	76.0	70.5	**77.4**
High Jump	52.5	59.8	**69.7**	66.6
Gymnastics	88.9	**99.1**	**99.1**	**99.1**
average	84.83	87.35	89.76	**91.5**

Table 2. Non-rigid object sequences: percentage of correctly tracked frames with fixed negative sampling, sampling from motion, combined fixed+random, and fixed+motion sampling.

the tracker. On average, the best strategy is “fixed+motion”, with a relative improvement of around 7.5%. We use this strategy for the following experiments and call the overall tracking algorithm “Motion Context Tracker” (MCT).

We further evaluated MCT with the VOT2013 dataset using the protocol of the VOT challenge and comparing it with 27 other state-of-the-art tracking methods. Table 3 shows the average accuracy and robustness with the three different experiments explained above: baseline, region-noise, and greyscale.

Table 4 lists the top 6 ranks for each experiment, combining accuracy and robustness. The results of MCT are very competitive, being the second-best method for baseline and region-noise and third-best for greyscale. Only, one method, the Pixel-based LUT Tracker (PLT), is consistently outperforming MCT on this dataset. It is an optimisation of the tracker called “Struck” [9], currently unpublished but some explanation can be found in [11]. Note that, PLT is a single-scale tracker and it uses different feature sets for greyscale and colour video sequences.

Table 5 shows the accuracy and robustness results for the VOT2014 dataset.

Finally, Fig. 2 shows some qualitative tracking results on some of the video sequences. One can see that the algorithm is very robust to changes in object appearance, illumination, pose as well as complex motion, and partial occlusions. The algorithm runs at around 4fps (or with a single-scale OAB detector: at

	accuracy			**robustness**		
	baseline	region-noise	greyscale	baseline	region-noise	greyscale
average	0.597	0.579	0.590	0.458	0.417	0.867

Table 3. Results of the proposed algorithm on the VOT2013 dataset.

baseline		region-noise		greyscale	
PLT	4.96	PLT	3.58	PLT	3.96
MCT	**6.62**	**MCT**	**5.08**	FoT [20]	4.75
FoT [20]	8.25	CCMS	8.33	**MCT**	**6.25**
EDFT [4]	9.5	FoT [20]	9.04	EDFT [4]	7.5
CCMS	9.54	LGT++ [22]	9.04	GSDT [5]	9.5
LGT++ [22]	10.2	EDFT [4]	9.08	LGT++ [22]	9.58

Table 4. Overall ranking result with the VOT2013 dataset. Only the first 6 out of 28 ranks are shown. The numbers represent the actual average ranking.

20fps) for a frame size of 320×240 on an Intel Xeon 3.4GHz not counting the initialisation phase and screen display.

5 Conclusions

We presented a new efficient particle filter-based approach for tracking arbitrary objects in videos. The method combines generative and discriminative models, by effectively integrating an online learning classifier. We propose a new method to train this classifier that samples the position of negative examples from contextual motion cues instead of a fixed region around the tracked object. Our extensive experimental results show that this procedure improves the overall tracking performance. Further, the proposed tracking algorithm gives state-of-the-art results on four different challenging tracking datasets, effectively dealing with large object shape and appearance changes, as well as complex motion, varying illumination conditions and partial occlusions.

References

1. Avidan, S.: Ensemble tracking. IEEE Trans. on Pattern Analysis and Machine Intelligence 29(2), 261–271 (Feb 2007)
2. Babenko, B., Yang, M.H., Belongie, S.: Visual tracking with online multiple instance learning. In: Proc. of the International Conference on Computer Vision and Pattern Recognition (Dec 2009)
3. Dinh, T., Vo, N., Medioni, G.: Context tracker: Exploring supporters and distracters in unconstrained environments. In: Proc. of the Computer Vision and Pattern Recognition (2011)
4. Felsberg, M.: Enhanced distribution field tracking using channel representations. In: Visual Object Tracking Challenge (VOT2013), ICCV (2013)
5. Gao, J., Xing, J., Hu, W., X., Z.: Graph embedding based semi-supervised discriminative tracker. In: Visual Object Tracking Challenge (VOT2013), ICCV (2013)

	accuracy		**robustness**	
	baseline	region-noise	baseline	region-noise
ball	0.58	0.42	0.40	0.20
basketball	0.54	0.52	1.53	1.67
bicycle	0.54	0.52	0.00	0.00
bolt	0.58	0.62	0.60	1.13
car	0.66	0.62	0.00	0.00
david	0.70	0.65	0.00	0.00
diving	0.38	0.36	1.27	1.20
drunk	0.55	0.53	0.00	0.07
fernando	0.32	0.33	2.07	3.00
fish1	0.33	0.31	1.67	1.73
fish2	0.31	0.28	3.40	4.13
gymnastics	0.53	0.51	1.07	1.20
hand1	0.46	0.40	1.53	1.93
hand2	0.35	0.34	6.20	6.40
jogging	0.75	0.69	0.93	1.00
motocross	0.45	0.47	1.27	2.00
polarbear	0.70	0.64	0.00	0.00
skating	0.55	0.47	0.33	0.80
sphere	0.75	0.82	0.00	0.00
sunshade	0.63	0.58	0.00	0.00
surfing	0.71	0.71	0.00	0.00
torus	0.53	0.51	1.13	1.47
trellis	0.59	0.53	1.00	1.07
tunnel	0.35	0.41	0.33	0.53
woman	0.58	0.61	0.00	0.13
average	**0.54**	**0.51**	**0.99**	**1.19**

Table 5. Results of the proposed algorithm on the VOT2014 dataset.

6. Gengembre, N., Pérez, P.: Probabilistic color-based multi-object tracking with application to team sports. Tech. Rep. 6555, INRIA (2008)
7. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: Proc. of the British Machine Vision Conference (2006)
8. Grabner, H., Matas, J., Van Gool, L., Cattin, P.: Tracking the invisible: Learning where the object might be. In: Proc. of the Computer Vision and Pattern Recognition. vol. 3, pp. 1285–1292 (Jun 2010)
9. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: Proc. of the International Conference on Computer Vision (2011)
10. Hong, Z., Mei, X., Tao, D.: Dual-force metric learning for robust distracter-resistant tracker. In: Proc. of the European Conference on Computer Vision (2012)
11. Kristan, M., Cehovin, L., Pflugfelder, R., Nebehay, G., Fernandez, G., Matas, J., et al.: The Visual Object Tracking VOT2013 challenge results. In: Proc. of the International Conference on Computer Vision (Workshops) (2013)
12. Maggio, E.: Adaptive multifeature tracking in a particle filtering framework. IEEE Trans. on Circuits and Systems for Video Technology 17(10), 1348–1359 (2007)
13. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. IEEE Trans. on Pattern Analysis and Machine Intelligence 33(11), 2259–72 (Nov 2011)
14. Odobez, J.M., Bouthemy, P.: Robust multiresolution estimation of parametric motion models. Journal of Visual Communication and Image Representation 6(4), 348–365 (Dec 1995)
15. Odobez, J.M., Gatica-Perez, D., Ba, S.O.: Embedding motion in model-based stochastic tracking. IEEE Trans. on Image Processing 15(11), 3514–3530 (2006)

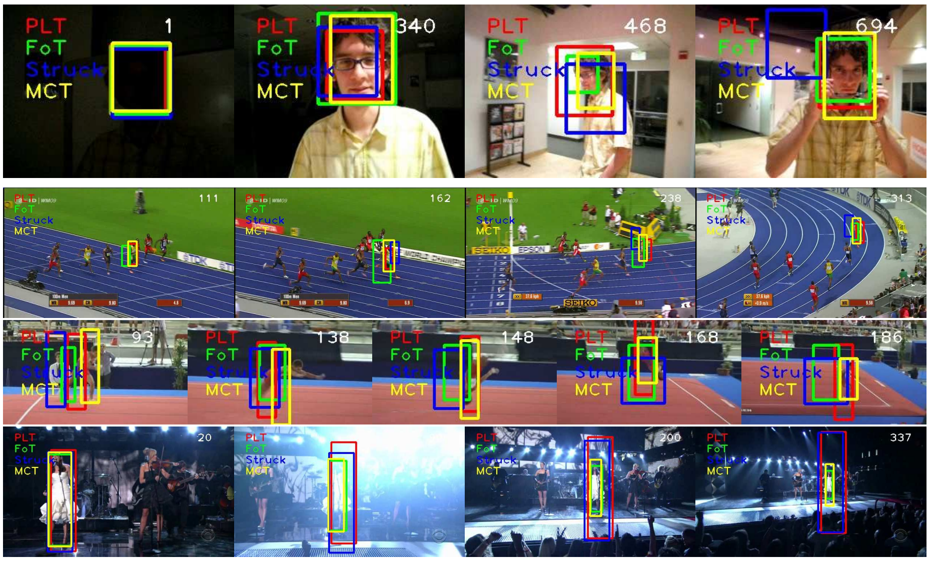

Fig. 2. Tracking results for PLT, FoT, Struck, and MCT on the sequences "David", "Bolt", "Gymnastics", and "Singer" (VOT2013/2014). Tracking is very robust to partial occlusions, illumination changes, deformations, pose or other appearance changes. The second last example shows some difficulties of MCT to adapt to different aspect ratios. And the last example illustrates the problem of drastic size change for single-scale trackers like Struck and PLT.

16. Okuma, K., Taleghani, A., Freitas, N.D.: A boosted particle filter: Multitarget detection and tracking. In: Proc. of the European Conference on Computer Vision. pp. 28–39 (2004)
17. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based probabilistic tracking. In: Proc. of the European Conference on Computer Vision (2002)
18. Sun, Z., Yao, H., Zhang, S., Sun, X.: Robust visual tracking via context objects computing. In: Proc. of the International Conference Image Processing. pp. 509–512 (Sep 2011)
19. Čehovin, L., Kristan, M., Leonardis, A.: Robust visual tracking using an adaptive coupled-layer visual model. IEEE Trans. on Pattern Analysis and Machine Intelligence 35(4), 941–953 (Apr 2013)
20. Vojíř, T., Matas, J.: Robustifying the flock of trackers. In: Computer Vision Winter Workshop. pp. 91–97 (2011)
21. Wen, L., Cai, Z., Lei, Z., Yi, D., Li, S.: Robust online learned spatio-temporal context model for visual tracking. IEEE Trans. on Image Processing 23(2) (2013)
22. Xiao, J., Stolkin, R., Leonardis, A.: An enhanced adaptive coupled-layer lgtracker++. In: Visual Object Tracking Challenge (VOT2013), ICCV (2013)
23. Yang, M., Wu, Y., Hua, G.: Context-aware visual tracking. IEEE Trans. on Pattern Analysis and Machine Intelligence 31(7), 1195–1209 (Jul 2009)
24. Zhang, G., Jia, J., Xiong, W., Wong, T.T., Heng, P.A., Bao, H.: Moving object extraction with a hand-held camera. In: Proc. of the International Conference on Computer Vision. pp. 1–8 (2007)