PixelTrack: a fast adaptive algorithm for tracking non-rigid objects

Stefan Duffner, Christophe Garcia

PixelTrack: a fast adaptive algorithm for tracking non-rigid objects

Stefan Duffner and Christophe Garcia
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621, France
stefan.duffner@liris.cnrs.fr, christophe.garcia@liris.cnrs.fr

Abstract

In this paper, we present a novel algorithm for fast tracking of generic objects in videos. The algorithm uses two components: a detector that makes use of the generalised Hough transform with pixel-based descriptors, and a probabilistic segmentation method based on global models for foreground and background. These components are used for tracking in a combined way, and they adapt each other in a co-training manner. Through effective model adaptation and segmentation, the algorithm is able to track objects that undergo rigid and non-rigid deformations and considerable shape and appearance variations. The proposed tracking method has been thoroughly evaluated on challenging standard videos, and outperforms state-of-the-art tracking methods designed for the same task. Finally, the proposed models allow for an extremely efficient implementation, and thus tracking is very fast.

1. Introduction

Given a video stream, tracking arbitrary objects that are non-rigid, moving or static, rotating and deforming, partially occluded, under changing illumination and without any prior knowledge is a challenging task. This unconstrained tracking problem where the object model is initialised from a bounding box in the first video frame and continuously adapted has been increasingly addressed in the literature in the past years. When no prior knowledge about the object's shape and appearance is available, one of the main difficulties is to incrementally learn a robust model from consecutive video frames. This model should generalise to new unseen appearances and avoid drift, *i.e*. the gradual inclusion of background appearance, which can ultimately lead to tracking failure.

Our method addresses these issues with an adaptive approach combining a detector based on pixel-based descriptors and a probabilistic segmentation framework.

1.1. Related Work

Earlier works [21, 11, 32, 30, 41, 18] on visual object tracking mostly consider a bounding box (or some other simple geometric model) representation of the object to track, and often a global appearance model is used. These classical methods are very robust to some degree of appearance change and local deformations (as in face tracking), and also allow for a fast implementation. However, for tracking *non-rigid* objects that undergo a large amount of deformation and appearance variation, *e.g*. due to occlusions or illumination changes, these approaches are less suitable. Although some algorithms effectively cope with object deformations by tracking their contour (*e.g*. [31, 42, 10]), most of them require the object to be moving or need prior shape knowledge [12]. Others, describe an object by a relatively dense set of keypoints that are matched in each frame [26, 19] to track the object. However, these methods have mostly been applied to relatively rigid objects.

Many existing methods, follow a tracking-by-detection approach, where a discriminative model of the object to track is built and updated "online", *i.e*. during the tracking, in order to adapt to possible appearance changes. For example, Adam *et al*. [1] use a patch-based appearance model with integral histograms of colour and intensity. The dynamic patch template configuration allows to model spatial structure and to be robust to partial occlusions. Grabner *et al*. [16] proposed an Online Adaboost (OAB) learning algorithm that dynamically selects weak classifiers that discriminate between the object image region and the background. Later, they extended this method to a semi-supervised algorithm [17] that uses a fixed (or adaptive [39]) prior model to avoid drift and an online boosting framework learning with unlabelled data. Babenko *et al*. [4] presented another online method based on Multiple Instance Learning (MIL), where the positive training examples are bags of image patches containing at least one positive (object) image patch. Besides boosting algorithms, Online Random Forests have been proposed for adaptive visual object tracking [36, 37], where randomised trees are incrementally grown to clas-

sify an image region as object or background. Kalal *et al*. [22] also use randomised forests which they combine effectively with a Lucas-Kanade tracker in a framework called Tracking-Learning-Detection (TLD) where the tracker updates the detector using spatial and temporal constraints and the detector re-initialises the tracker in case of drift.

In order to cope with changing appearance, Mei *et al*. [28] introduced the l_1 tracker that is based on a sparse set of appearance templates that are collected during tracking and used in the observation model of a particle filter. Recently, several extensions have been proposed [6, 20, 43] to improve the robustness and reduce the complexity of this method. However, these approaches are still relatively time-consuming due to the complex l_1 minimisation. A sparse set of templates has also been used by Liu *et al*. [27], but with smaller image patches of object parts, and by Kwon *et al*. [23] in their Visual Tracking Decomposition (VTD) method. In a similar spirit, Ross *et al*. [34] propose a particle filter algorithm called IVT that uses an observation model relying on the eigenbasis of image patches computed online using an incremental PCA algorithm. Other approaches, more similar to ours, consist in using a pixel-based classifier [3, 9]. Avidan *et al*. [3], for example, proposed an ensemble tracking method that label each pixel as foreground or background with an Adaboost algorithm that is updated online. However, all of these methods still operate more or less on image regions described by bounding boxed and inherently have difficulties to track objects undergoing large deformations.

To overcome this problem, recent approaches integrate some form of segmentation into the tracking process. For example, Nejhum *et al*. [29] proposed to track articulated objects with a set of independent rectangular blocks that are used in a refinement step to segment the object with a graph-cut algorithm. Similarly, although not segmenting the object, Kwon *et al*. [24] handle deforming objects by tracking configurations of a dynamic set of image patches, and they use Basin Hopping Monte Carlo (BHMC) sampling to reduce the computational complexity. Other approaches [33, 40] use a segmentation on the superpixel level. Bibby *et al*. [8] propose an adaptive probabilistic framework separating the tracking of non-rigid objects into registration and level-set segmentation, where posterior probabilities are computed at the pixel level. Aeschliman *et al*. [2] also combined tracking and segmentation in a Bayesian framework, where pixel-wise likelihood distributions of several objects and the background are modelled by Gaussian functions the parameters of which are learnt online. In a different application context, pixel-based descriptors have also been used for 3D articulated human-body detection and tracking by Shotton *et al*. [38] on segmented depth images. In the approach recently proposed by Belagiannis *et al*. [7], a graph-cut segmentation is applied separately to the image patches provided by a particle filter.

The work of Godec *et al*. [15] is probably the most similar to ours. The authors proposed a patch-based voting algorithm with Hough forests [14]. By back-projecting the patches that voted for the object centre, the authors initialise a graph-cut algorithm to segment foreground from background. The resulting segmentation is then used to update the patches' foreground and background probabilities in the Hough forest. This method achieves state-of-the-art tracking results on many challenging benchmark videos. However, due to the graph-cut segmentation it is relatively slow. Also, the segmentation is discrete and binary, which can increase the risk of drift due to wrongly segmented image regions.

1.2. Motivation

The algorithm presented in this paper is inspired by these recent works on combined tracking and segmentation, which is beneficial for tracking non-rigid objects. Furthermore, *patch-based* local descriptors have shown state-of-the-art performance due to their possibility of handling appearance changes with large object deformations.

In this paper, we integrate these concepts and present a novel tracking-by-detection algorithm that relies on a Hough-voting scheme based on pixel descriptors. The method tightly integrates with a probabilistic segmentation of foreground and background that is used to incrementally update the local pixel-based descriptors and vice versa. The local Hough-voting model and the global colour model operate both at the pixel level and thus allow for very efficient model representation and inference.

The following scientific contributions are made:

- a fast tracking algorithm using a detector based on the generalised Hough transform and pixel descriptors in combination with a probabilistic soft segmentation method,
- a co-training framework, where the local pixel-based detector model is used to update the global segmentation model and vice versa,
- a thorough performance evaluation on standard datasets and a comparison with other state-of-the-art tracking algorithms.

Note that the main goal of the proposed approach is to provide an accurate *position estimate* of an object to track in a video. Here, we are not so much interested in a perfect segmentation of the object from the background. Better approaches for segmentation exist in the literature but they are rather complex. Thus, instead of using these, we aimed at reducing the overall computational complexity.

In the following, we will first describe the overall approach (Section 2). Then we will detail each of the com-

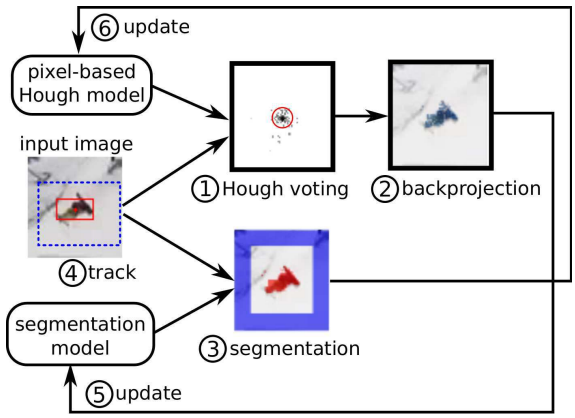

Figure 1. The overall tracking procedure for one video frame. Each pixel inside the search window (blue dotted rectangle) in the input image casts a vote (1) according to the current Hough transform model (darker: high vote sum, lighter: low vote sum). Then the maximum vote sum (circled in red) and all pixels that have contributed (*i.e.* the backprojection) are determined (2). In parallel, the current segmentation model is used to segment the image region inside the search window (3) (binarised segmentation shown in red). Finally, after updating the objects current position (4), the segmentation model is adapted (5) using the backprojected image pixels, and the Hough transform model is updated (6) with foreground pixels from the segmentation and background pixels from a region around the object (blue frame).

ponents of the algorithm (Sections 3-5). And finally, some experimental results will be presented (Section 7).

2. Overall Approach

The overall procedure of detection, segmentation, tracking, and model adaptation for one video frame is illustrated in Fig. 1. The algorithm receives as input the current video frame as well as the bounding box and segmentation from the tracking result of the previous frame. The pixel-based Hough transform is applied on each pixel inside the search window, the enlarged bounding box, *i.e.* each pixel votes for the centre position of the object according to the learnt model. Votes are cumulated in a common reference frame, the voting map, and the position with the highest sum of votes determines the most likely position of the object's centre (see Section 3 for a more detailed explanation). Then, the pixels that have contributed to the maximum vote are determined. This process is called *backprojection*. In parallel, the image region corresponding to the search window is segmented using the current segmentation model. That is, each pixel is assigned a foreground probability (see Section 4). The position of the tracked object is updated using the maximum vote position and the centre of mass of the segmentation output (see Section 5). Finally, the models are adapted in a co-training manner to avoid drift. That means, the segmentation model is updated using the backprojection, and the pixel-based Hough model is adapted according to the segmentation output (see Section 6 for more details).

3. Pixel-based Hough Voting

We developed a new detection algorithm relying on the generalised Hough transform [5]. In contrast to existing models developed recently for similar tasks (*e.g.* [14, 15]) which use Hough forests, *i.e.* Random Forests trained on small *image patches*, or the Implicit Shape Model (ISM) [25] our method operates at the *pixel-level*.

This has the following advantages:

- pixel-based descriptors are more suitable for detecting objects that are extremely small in the image (*e.g.* for far-field vision),
- the feature space is relatively small and does not (or very little) depend on spatial neighbourhood, which makes training and updating of the model easier and more coherent with the object's appearance changes,
- the training and the application of the detector is extremely fast as it can be implemented with look-up tables.

One drawback of using a pixel-based Hough model is when the object's image region contains primarily pixels of very similar colours (and gradients). In that case, the pixels on their own may not be discriminative enough to infer the objects centre position. Note that also patch-based methods have difficulties with uniform regions. In practice, however, this is rarely the case and may be controlled by increasing the discriminative power of the descriptors (at the cost of invariance). Also, in this tracking framework, this risk is considerably reduced by combining the detector with the segmentation output.

Let us now consider the model creation and application in detail. Figure 2 illustrates the model creation (training) and detection process.

3.1. Training

Let us denote $\mathbf{x} = (x_1, x_2)$ the position of a pixel $I(\mathbf{x})$ in an image I. In the training image, the pixels inside a given initial bounding box are quantised according to the vector composed of its HSV colour values (with separate V quantisation) and its x and y gradient orientation (with a separate quantisation for low gradient magnitudes) (see Fig. 2 left). Experiments showed that colour alone is also working well, but not gradient alone. This amounts to computing $\mathbf{D} = \mathbf{D}_z$ ($z = 1..N$), an N-dimensional histogram, which is referred to as *pixel-based Hough model* in this paper. Here, we use $N = (16 \times 16 + 16) \times (8 + 1) = 2448$ (16 colour bins and 8 gradient orientations). The vectors $\mathbf{D}_z = \{\mathbf{d}_z^1, \ldots, \mathbf{d}_z^{M_z}\}$

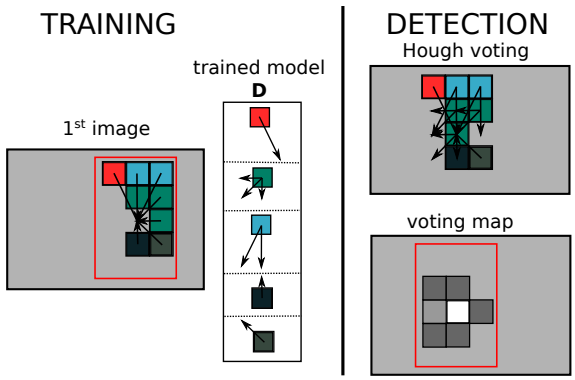

Figure 2. Training and detection with the pixel-based Hough model. Left: the model $\mathbf{D}$ is constructed by storing for each quantised pixel value in the given bounding box all the displacement vectors to the object's centre position (here only colour is used for illustration). Right: the object is detected in a search window by accumulating the displacement votes of each pixel in a voting map (bright pixels: many votes, dark pixels: few votes).

contain M_z displacement vectors $\mathbf{d}_z^m = (\mathbf{x}_{zm}, w_{zm})$, each associated with a weight $w_{zm} = 1.0$. Thus, training consists in constructing $\mathbf{D}$, where each pixel $I(\mathbf{x})$ in the given bounding box produces a displacement vector $\mathbf{d}_z$ (arrows in Fig. 2) corresponding to its quantised value $z_\mathbf{x}$ and pointing to the centre of the bounding box.

3.2. Detection

In a new video frame, the object can be detected by letting each pixel $I(\mathbf{x})$ inside the search window vote according to $\mathbf{D}_z$ corresponding to its quantised value $z_\mathbf{x}$. The right part of Fig. 2 illustrates this. Each vote is a list of displacements $\mathbf{d}_z^m$ that are weighted by w_{zm} and accumulated in a voting map. The detector's output is then simply the position in the voting map with the maximum value $\mathbf{x}_{max}$.

Note that, as illustrated in figure 2, the position estimation is "diffused" by two factors: the deformation of the object (one of the green pixels in the figure), and pixels of the same colour (green and blue pixels). But the maximum value in the voting map is still distinctive and corresponds well to the centre position of the object. This could also be observed in our experiments.

Nevertheless, to be robust to very small deformations we group the votes in small voting cells of 3×3 pixels (as [15]).

3.3. Backprojection

With the position of the maximum in the voting map $\mathbf{x}_{max}$, we can determine which pixels in the search window contributed to it during the detection. This process is illustrated in Fig. 1 and is called *backprojection*. More precisely, let z be the value of pixel $I(\mathbf{x})$. Then, the backprojection b at each position $\mathbf{x} \in \Omega$ is defined as:

$$b_\mathbf{x} = \begin{cases} w_{zm} & \text{if } \mathbf{d}_\mathbf{z}^\mathbf{m} \text{ voted for } \mathbf{x}_{max} \ , \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The backprojected pixels are used for adapting the segmentation model (see Section 6 for more details). The idea behind this is that, intuitively, pixels that contributed to $\mathbf{x}_{max}$ are most likely corresponding to the object.

4. Segmentation

Complementary to the local pixel-wise Hough model, a global segmentation model is trained and adapted to allow for varying object shapes and for a better discrimination between foreground (object) and background, especially when the shape and appearance changes drastically and abruptly.

A probabilistic soft segmentation approach is adopted here (similar to [2]). Let $c_{t,\mathbf{x}} \in \{0, 1\}$ be the class of the pixel at position $\mathbf{x}$ at time t: 0 for background, and 1 for foreground, and let $y_{1:t,\mathbf{x}}$ be the pixel's colour observations from time 1 to t. For clarity, we'll drop the index $\mathbf{x}$ in the following. In order to incorporate the segmentation of the previous video frame at time $t-1$ and to make the estimation more robust, we use a recursive Bayesian formulation, where, at time t, each pixel (in the search window) is assigned the foreground probability:

$$p(c_t = 1|y_{1:t}) = Z^{-1} p(y_t|c_t = 1) \sum_{c'_{t-1}} p(c_t = 1|c'_{t-1})\ p(c'_{t-1}|y_{1:t-1}) \ , \quad (2)$$

where Z is a normalisation constant to make the probabilities sum to 1. The distributions $p(y_t|c_t)$ are modelled with HSV colour histograms with 12×12 bins for the H and S channels and 12 separate bins for the V channel. The foreground histogram is initialised from the image region defined by the bounding box around the object in the first frame. The background histogram is initialised from the image region surrounding this rectangle (with some margin between). The transition probabilities for foreground and background are set to:

$$p(c_t = 0|c_{t-1}) = 0.6 \qquad p(c_t = 1|c_{t-1}) = 0.4 \quad (3)$$

which is an empirical choice that has been validated experimentally. Note that the tracking algorithm is not very sensitive to these parameters.

As opposed to recent work on image segmentation (*e.g.* [35]), we treat each pixel independently, which, in general, leads to a less regularised solution but at the same time reduces the computational complexity considerably. As stated in section 1.2, we are not so much interested here in a perfectly "clean" segmentation but rather in fast and robust tracking of the position of an object.

5. Tracking

In a new video frame, pixel-based detection and segmentation are performed inside a search window Ω, which is set here to twice the size of the object's bounding box. Then, the object's position $\mathbf{X}_t$ can be re-estimated. To this end, we utilise not only the output of the detector, *i.e*. the maximum position in the voting map, but also the segmentation output. Clearly, this makes the tracking algorithm more robust to non-rigid deformations. More precisely, we calculate the centre of mass of the soft segmentation produced by Eq. 2:

$$\mathbf{x}_s = \frac{1}{S} \sum_{\mathbf{x} \in \Omega} p(c_\mathbf{x} = 1|y)\ \mathbf{x}, \tag{4}$$

where S is the sum of all foreground probabilities $p(c_\mathbf{x} = 1|y)$ in the search window Ω. Then, we set the new object position to a linear combination of voting map maximum $\mathbf{x}_{max}$ and $\mathbf{x}_s$:

$$\mathbf{X}_t = \alpha \mathbf{x}_s + (1 - \alpha)\mathbf{x}_{max}\,. \tag{5}$$

The factor α determines how much we trust in the segmentation's position compared to the Hough model's estimation. It is computed dynamically at each frame by a simple reliability measure that is defined as the proportion of pixels in the search window that change from foreground to background or vice versa, *i.e*. crossing the threshold $p(c_\mathbf{x} = 1|y) = 0.5$. A high value of α means that the shape of the object has changed considerably from the previous frame to the current one, and consequently the Hough voting is assumed to be less accurate.

6. Model adaptation

Both pixel-based Hough model and segmentation model are updated at each frame in a co-training manner, *i.e*. the output of one model is used to update the other one. To update the Hough model, only foreground pixels are used, that is pixels for which $p(c_\mathbf{x} = 1|y) > 0.5$. For each of these pixels $\mathbf{x}$ the displacement $\mathbf{d}$ to the new object's centre is calculated, and its weight w is set according to its foreground probability:

$$w \leftarrow \begin{cases} \gamma\, p(c_\mathbf{x} = 1|y) + (1 - \gamma)\, w & \text{if } \mathbf{d} \in \mathbf{D}_z\ , \\ p(c_\mathbf{x} = 1|y) & \text{otherwise,} \end{cases}$$

where $\gamma = 0.1$ is the update factor. In the second case, $\mathbf{d}$ is added to $\mathbf{D}_z$. For computational and memory efficiency, we limit the size of each $\mathbf{D_z}$ and only keep the K displacements with the highest weights ($K = 20$ in our experiments).

The foreground and background distributions of the segmentation model are adapted using the backprojection $b_\mathbf{x}$. That is, the colour distribution $p(y|b > 0.5)$ of the backprojected pixels is calculated, and used to linearly update the current foreground colour distribution:

$$p(y_t|c_t = 1) = \delta\, p(y|b > 0.5) + (1 - \delta)\, p(y_{t-1}|c_{t-1} = 1)\ , \tag{6}$$

where $\delta = 0.1$ is the update factor. The background colour distribution is updated in the same way but using the colour distribution from a rectangular frame surrounding the object borders (as for initialisation).

7. Evaluation

7.1. Evaluation Protocol

We conducted quantitative evaluation on two sets of challenging standard videos that are commonly used in the literature. The tracking accuracy and speed on these datasets has been measured and compared to two state-of-the-art tracking methods.

The first dataset[1] has been constructed by Babenko *et al*. [4] from various other publications, and it has been used by Godec *et al*. [15]. It contains 8 videos (with more than 5 000 frames) of objects or faces that undergo mostly rigid deformations and some rather large lighting variations as well as partial occlusions. Most of these sequences are actually in grey-scale format (except "David", "Girl", and "Face Occlusions 1"). For this reason, we do not use the segmentation part of our algorithm here because it is based on colour. That means, tracking is only performed by Hough voting.

The second dataset[2] is composed of 11 videos (around 2 500 frames) showing moving objects that undergo considerable rigid and non-rigid deformations. This dataset has also been used by [15] and partially by [23] among others.

We compared our algorithm which we call *PixelTrack* to two state-of-the-art methods: HoughTrack (HT) proposed by Godec *et al*. [15] and Tracking-Learning-Detection (TLD) by Kalal *et al*. [22]. Although, these and other previous works have reported tracking accuracy results on some of the videos from our datasets, we evaluated these methods again using our performance measure in order to have a consistent comparison. Initialisation is done manually, where HT has been initialised by the values given by their authors, and the same initialisation has been used for TLD. For our method, the initial rectangle is smaller as it should contain as few background pixels as possible in order to obtain a good initial segmentation model.

To measure the performance of the different tracking algorithms, we determine, for each video, the percentage of frames in which the object is correctly tracked. In each video frame, the tracking is considered correct if the PASCAL VOCC [13] overlap measure $\frac{R_T \cap R_{GT}}{R_T \cup R_{GT}}$ is above a threshold, where R_T is the rectangle from the tracking algorithm, and R_{GT} is the ground truth rectangle surrounding the object. We set the threshold to 0.1. A higher

[1] http://vision.ucsd.edu/ bbabenko/project_miltrack.shtml

[2] http://lrs.icg.tugraz.at/research/houghtrack/

	HT [15]	TLD [22]	**PixelTrack**
David	**89.25**	77.42	45.16
Sylvester	55.02	**88.10**	36.80
Girl	92.22	**97.01**	93.21
Face Occlusions 1	99.44	**100.00**	**100.00**
Face Occlusions 2	**100.00**	46.01	88.34
Coke	72.88	88.14	**91.53**
Tiger 1	26.76	12.68	**46.48**
Tiger 2	41.10	19.18	**98.63**
average	72.08	66.07	**75.02**

Table 1. Babenko sequences: percentage of correctly tracked frames.

	HT [15]	TLD [22]	**PixelTrack**
Cliff-dive 1	**100.00**	69.12	**100.00**
Motocross 1	**100.00**	15.38	57.69
Skiing	**100.00**	6.85	**100.00**
Mountain-bike	**100.00**	81.36	94.55
Cliff-dive 2	**100.00**	8.20	32.79
Volleyball	45.12	42.28	**100.00**
Motocross 2	**100.00**	**100.00**	**100.00**
Transformer	38.71	33.06	**94.35**
Diving	21.21	24.68	**88.74**
High Jump	77.87	35.25	**94.26**
Gymnastics	98.87	84.75	**99.09**
average	80.16	45.54	**87.41**

Table 2. Non-rigid object tracking: percentage of correctly tracked frames.

value would discriminate our method too much because the bounding box that is output currently does not change its size and aspect ratio during the tracking, and it is rarely initialised to surround the complete object.

Finally, we want to emphasise that we strictly used the same parameters of our algorithm for all the videos and all the experiments.

7.2. Results

Table 1 summarises the evaluation results on the Babenko videos. Although our method is not designed for grey-scale videos and, thus, does not show its full potential, it still performs better than other state-of-the-art methods on many videos of the dataset and also on average. Fig. 3 shows some tracking results from the "Tiger 2" sequence. HoughTrack loses track after frame 100. Also TLD very early stops to detect the object and removes the track, whereas PixelTrack is able to follow the object despite the appearance changes and occlusions.

Table 2 shows the results for dataset 2 with non-rigid deformations. For 8 out of 11 video sequences our method outperforms the other algorithms or is on par. Also the average of correct tracking is almost 7 percentage points

HT	TLD	**PixelTrack**
2.3	5.2	**113.8**

Table 3. Average overall processing speed in frames per second. Measured without screen display.

higher than HT. The two videos that are the most difficult for the proposed method are "Motocross", where the motorbike does a complete flip, changes its size in the video, and where the background is very cluttered, and "Cliff Dive 2" where the appearance (and shape) of foreground and background changes dramatically.

Fig. 4 illustrates some tracking results of the three compared methods on one of the sequences: "Diving". The two baseline methods lose the track at some point. The proposed algorithm is able to track the diver despite considerable shape and appearance changes. Fig. 5 shows some more tracking results of PixelTrack.

Finally, we measured the average processing speed of each algorithm for all the 19 videos on a 3.4 GHz Intel Xeon processor. In terms of speed, the proposed method outperforms the other two by a factor of around 20 for TLD and 50 for HT. The reason why HT is relatively slow is the graph-cut segmentation, which is computationally demanding (but on the other hand also quite accurate). The code of HT is implemented in C++ using OpenCV library. Whereas, TLD uses Matlab with pre-compiled OpenCV code. The fast execution speed is a real advantage of the proposed approach. This is due to pixel-based Hough voting that allows for an extremely efficient implementation with lookup-tables as well as a fast segmentation algorithm.

8. Conclusions

We presented an algorithm for tracking generic objects in videos without any prior knowledge. It is an effective combination of a local pixel-based detector based on a Hough voting scheme and a global probabilistic segmentation method that operate jointly and update each other in a co-training manner. Our algorithm is very fast compared to existing methods, which makes it suitable for real-time applications, or tasks where many objects need to be tracked at the same time, or where large amounts of data need to be processed (*e.g.* video indexation). Experimental results show that the method outperforms state-of-the-art tracking algorithms on challenging videos from standard benchmarks.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proc. of CVPR*, pages 798–805, 2006. 1

[2] C. Aeschliman, J. Park, and A. C. Kak. A probabilistic framework for joint segmentation and tracking. In *Proc. of CVPR*, pages 1371–1378, June 2010. 2, 4

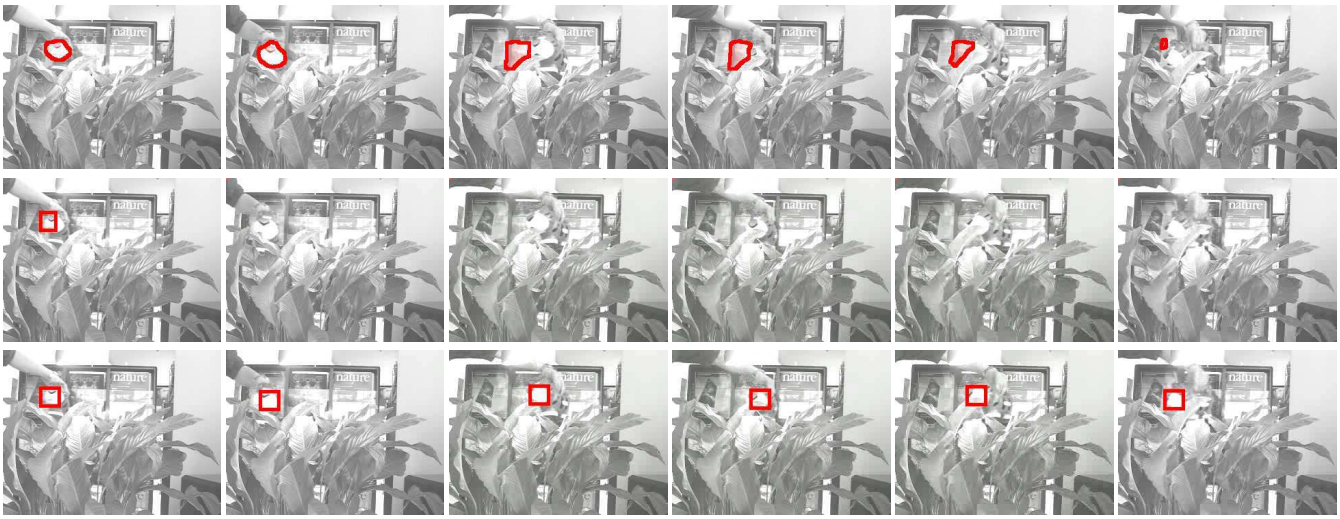

Figure 3. Tracking results for a small object with cluttered background and occlusions ('Tiger 2" sequence). *Top row*: HoughTrack [15], *middle row*: TLD [22], *bottom row*: proposed approach.

Figure 4. Tracking results for a non-rigid object with cluttered background ('Diving" sequence). *Top row*: HoughTrack [15], *middle row*: TLD [22], *bottom row*: proposed approach.

[3] S. Avidan. Ensemble tracking. *IEEE Transactions on PAMI*, 29(2):261–271, Feb. 2007. 2

[4] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Proc. of CVPR*, Dec. 2009. 1, 5

[5] D. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 1981. 3

[6] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *Proc. of CVPR*, pages 1830–1837, June 2012. 2

[7] V. Belagiannis, F. Schubert, N. Navab, and S. Ilic. Segmentation based particle filtering for real-time 2d object tracking. In *Proc. of ECCV*, pages 1–14, 2012. 2

[8] C. Bibby and I. Reid. Robust real-time visual tracking using pixel-wise posteriors. In *Proc. of ECCV*, 2008. 2

[9] K. Cannons, J. Gryn, and R. Wildes. Visual tracking using a pixelwise spatiotemporal oriented energy representation. In *Proc. of ECCV*, pages 511–524, 2010. 2

[10] P. Chockalingam, N. Pradeep, and S. Birchfield. Adaptive fragments-based tracking of non-rigid objects using level sets. In *Proc. of ICCV*, 2009. 1

[11] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. of CVPR*, volume 2, pages 142–149, 2000. 1

[12] D. Cremers and G. Funka-Lea. Dynamical statistical shape priors for level set based sequence segmentation. *IEEE Transactions on PAMI*, 28(8):1262–1273, 2006. 1

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 2010. 5

[14] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *IEEE Transactions on PAMI*, 33(11):2188–202, Nov. 2011. 2, 3

[15] M. Godec and P. M. Roth. Hough-based tracking of non-rigid objects. In *Proc. of ICCV*, 2011. 2, 3, 4, 5, 6, 7

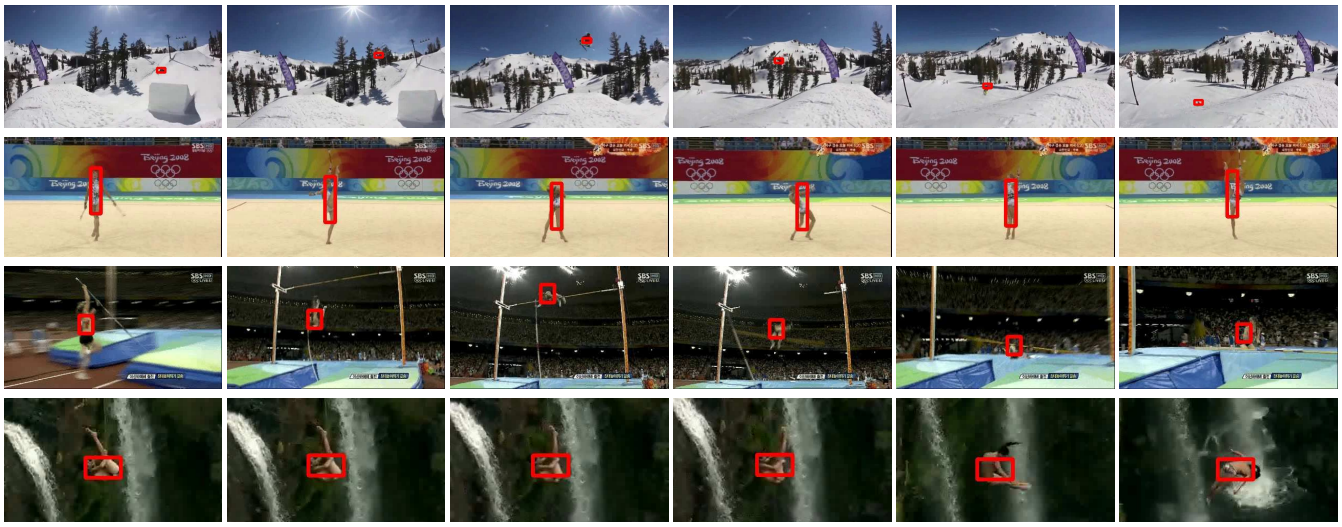

Figure 5. Some results of non-rigid object tracking with *PixelTrack*. From top to bottom: Skiing, Gymnastics, High-Jump, and Cliff-Dive1.

[16] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. of BMVC*, 2006. 1

[17] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. *Proc. of ECCV*, 2008. 1

[18] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *Proc. of ICCV*, 2011. 1

[19] S. Hare, A. Saffari, and P. H. S. Torr. Efficient online structured output learning for keypoint-based object tracking. In *Proc. of CVPR*, 2012. 1

[20] Z. Hong, X. Mei, and D. Tao. Dual-force metric learning for robust distracter-resistant tracker. In *Proc. of ECCV*, pages 513–527, 2012. 2

[21] M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *Proc. of ICCV*, 29(1):5–28, 1998. 1

[22] Z. Kalal, J. Matas, and K. Mikolajczyk. P-N learning: Bootstrapping binary classifiers by structural constraints. In *Proc. of CVPR*, 2010. 2, 5, 6, 7

[23] J. Kwon and K. Lee. Visual tracking decomposition. In *Proc. of CVPR*, pages 1269–1276, 2010. 2, 5

[24] J. Kwon and K. M. Lee. Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive Basin Hopping Monte Carlo sampling. In *Proc. of CVPR*, 2009. 2

[25] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV worksh. on statist. learning in comp. vis.*, 2004. 3

[26] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on PAMI*, 28(9):1465–79, 2006. 1

[27] B. Liu, J. Huang, L. Yang, and C. Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *Proc. of CVPR*, pages 1313–1320, 2011. 2

[28] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on PAMI*, 33(11):2259–72, Nov. 2011. 2

[29] S. S. Nejhum, J. Ho, and M.-H. Yang. Visual tracking with histograms and articulating blocks. *Proc. of CVPR*, 2008. 2

[30] J.-M. Odobez, D. Gatica-Perez, and S. O. Ba. Embedding motion in model-based stochastic tracking. *IEEE Trans. on Image Processing*, 15(11):3514–3530, 2006. 1

[31] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on PAMI*, 22(3):266–280, 2000. 1

[32] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. of ECCV*, 2002. 1

[33] X. Ren and J. Malik. Tracking as repeated figure/ground segmentation. In *Proc. of CVPR*, 2007. 2

[34] D. Ross, J. Lim, R. Lin, and M. Yang. Incremental learning for robust visual tracking. *IJCV*, 2008. 2

[35] A. B. C. Rother and V. Kolmogorov. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 2004. 4

[36] A. Saffari and C. Leistner. On-line random forests. In *Proc. of ICCV (Worksh. on Online Comp. Vis.)*, 2009. 1

[37] J. Santner, C. Leistner, A. Saffari, and T. Pock. PROST: Parallel robust online simple tracking. In *Proc. of CVPR*, pages 723–730, 2010. 1

[38] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake, and X. Incubation. Real-time human pose recognition in parts from single depth images. In *Proc. of CVPR*, 2011. 2

[39] S. Stalder, H. Grabner, and L. V. Gool. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *Proc. of ICCV (Worksh. on Online Comp. Vis.)*, 2009. 1

[40] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *Proc. of ICCV*, 2011. 2

[41] J. Yao and J.-M. Odobez. Multi-camera multi-person 3D space tracking with MCMC in surveillance scenarios. In *ECCV M2SFA2 Workshop*, 2008. 1

[42] A. Yilmaz, X. Li, and M. Shah. Object contour tracking using level sets. In *Proc. of ACCV*, 2004. 1

[43] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via structured multi-task sparse learning. *IJCV*, Nov. 2012. 2