Texture-Independent Long-Term Tracking Using Virtual Corners

Karel Lebeda, Simon Hadfield, *Member, IEEE,* Jiří Matas, *Member, IEEE,*
and Richard Bowden, *Senior Member, IEEE*

***Abstract*—Long term tracking of an object, given only a single instance in an initial frame, remains an open problem. We propose a visual tracking algorithm, robust to many of the difficulties which often occur in real-world scenes. Correspondences of edge-based features are used, to overcome the reliance on the texture of the tracked object and improve invariance to lighting. Furthermore we address long-term stability, enabling the tracker to recover from drift and to provide redetection following object disappearance or occlusion. The two-module principle is similar to the successful state-of-the-art long-term TLD tracker, however our approach offers better performance in benchmarks and extends to cases of low-textured objects. This becomes obvious in cases of plain objects with no texture at all, where the edge-based approach proves the most beneficial.**

We perform several different experiments to validate the proposed method. Firstly, results on short-term sequences show the performance of tracking challenging (low-textured and/or transparent) objects which represent failure cases for competing state-of-the-art approaches. Secondly, long sequences are tracked, including one of almost 30 000 frames which to our knowledge is the longest tracking sequence reported to date. This tests the re-detection and drift resistance properties of the tracker. Finally, we report results of the proposed tracker on the VOT Challenge 2013 and 2014 datasets as well as on the VTB1.0 benchmark and we show relative performance of the tracker compared to its competitors. All the results are comparable to the state-of-the-art on sequences with textured objects and superior on non-textured objects. The new annotated sequences are made publicly available.

***Index Terms*—Machine vision, image motion analysis, visual tracking, long-term tracking, low texture, edge, line correspondence.**

I. Introduction

A. Motivation

THIS paper addresses the task of visual tracking of a priori unknown low-textured objects in long-term scenarios. To achieve such a goal, a tracker needs to learn and adapt a model of object appearance to follow the object or to redetect it after full occlusions. This is, however, often difficult due to factors such as low texture of the object, or varying lighting conditions. We directly address these common sources of failure by abandoning more common point features and using lines instead; line features are more robust to lighting and are present even in cases of low-textured objects where point features are scarce.

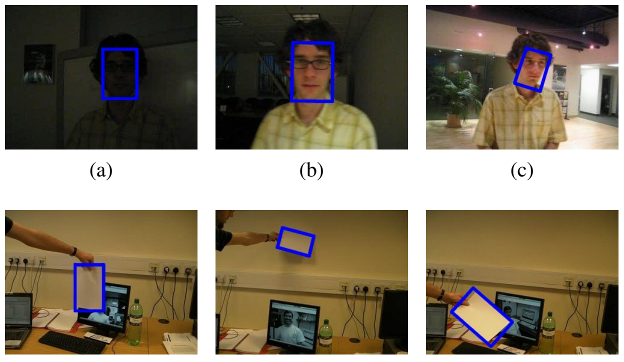

(a) (b) (c) (d) (e) (f)

Fig. 1: Examples from challenging sequences.

Visual tracking is a field of computer vision which has been thoroughly studied over the years. Many approaches have been proposed including tracking via local optimisation [22], [25], regression [9], detection[1], segmentation [26], generative models [27] and online learning [7], [8]. Long-term trackers attempt to model and adapt to changes in object appearance over time, using multiple observations to enrich their representations. This in turn leads to drift due to the difficulty of unsupervised learning. Recent approaches overcome this through a combination of detection combined with local search and intelligent online update strategies, which can compensate for drift via redetection. As such, consistency of appearance is extremely important, because radical changes can cause tracking failure or corruption of the model. The notion of appearance typically relies on texture or other strong visual attributes. However, there are a whole range of scenarios where sufficient texture is either absent or highly variable due to changes in pose or scene illumination. While this is only true for a subset of tracking sequences, it is a common source of tracking failure for most approaches.

Figure 1 shows example frames from two challenging sequences which typically lead to tracking failures. Figures 1a to 1c show images from Ross *et al*. [27] where the initial target face is so dark that visual features are almost indistinguishable to the human eye. However, as shown by the bounding box, the proposed LT-FLO tracker is capable of tracking the entire sequence (see supplementary material for a results video). Even Ross *et al*. do not track this sequence from the start, only beginning at the 300th frame (shown in 1b) where the visual

K. Lebeda (corresponding author), S. Hadfield and R. Bowden are with University of Surrey, GU2 7XH Guildford, United Kingdom (e-mail: {*k.lebeda,s.hadfield,r.bowden*}*@surrey.ac.uk*).

J. Matas is with Czech Technical University, Karlovo náměstí 13, 121 35 Prague, Czech Republic (e-mail: *matas@cmp.felk.cvut.cz*).

Manuscript created February 17, 2015; revised July 28, 2015, October 23, 2015 and October 29, 2015.

appearance from lighting is more consistent with the remainder of the video. This sequence was recently used in a visual tracking benchmark [31] starting from the latter frame as well. Similarly Figures 1d–1f show a texture-less object (a single piece of white paper on a light background). Trackers which rely on texture or appearance fail on this sequence. Again the bounding box shows LT-FLO's ability to successfully track in this scenario. Furthermore, we demonstrate LT-FLO successfully tracking a sequence of over 29 000 frames which to our knowledge is the longest sequence reported to date.

B. Related Work

One of the most influential works in the field of visual tracking is undoubtedly the Lucas-Kanade tracker (LK) [22]. This technique iteratively matches image patches by linearising the local image gradients and minimising the sum of squared pixel errors. It has been recently extended using segmentation-like object/background likelihoods [25]. Many current trackers follow and extend the idea of LK tracking by adding an upper layer managing a cloud of low-level *tracklets* [16], [19], [20], [23]. One recent example is a Local-Global Tracker (LGT) [4] by Cehovin *et al.*, using a coupled-layer visual model. The local layer consists of independently tracked visual patches, constraining the appearance of object components. The global layer models object features such as colour, shape or motion. The global model is learned from the local patches and in turn it constrains the addition of new patches. Consistency of local trackers is enforced; however changes in shape are possible, allowing LGT to track highly non-rigid objects. All of these approaches are based on variants of point features, which renders them ineffective in cases where these are scarce, or unstable.

Another approach is *tracking by detection* where tracking is defined as a classification task. Grabner *et al.* [7], [8] employ an *online boosting* method to update the appearance model while minimising error accumulation. Babenko *et al.* [1] use *multiple instance learning*, instead of traditional supervised learning, for a more robust tracker. Zheng *et al.* [32] address drift during tracking using a dynamic set of basis classifiers, employing different basis classifiers for different problems. All of these, however, suffer from the need to convert the estimated object position into a set of labelled training examples, and to couple the objective for the classifier (label prediction) to the objective for the tracker (object position estimation), which is difficult to perform optimally. The Struck tracker [9] solves this by explicitly using *structured output prediction* to avoid the need for these intermediate steps.

Kalal *et al.* combined tracking with detection in their Tracking-Learning-Detection (TLD) [15] framework, where (in)consistency of the tracker and detector helps to indicate tracking failure. While the *tracker* estimates the frame-to-frame motion, the *detector* treats every frame as independent (as in a tracking-by-detection scenario). Positive and negative examples are *learned* according to the (dis-)agreement of these two components, improving further detection. Explicit modelling of failures for both components coupled with independent detection makes this tracker suitable for *long-term tracking*, with inherent drift-resistance and redetection after full occlusions. In all of these cases, the learned detector needs a consistent texture to distinguish between the object and the background, making it unsuitable for texture-less tracking.

There are also many successful approaches using *particle filtering* for tracking. One of the most notable is CONDENSATION (Conditional Density Propagation) [13], which brought into the field of visual tracking the use of particles for non-parametric modelling of a pose probability distribution. Ross *et al.* [27] extended particle filtering by introducing *incremental learning* of an object appearance subspace that allowed the model to adapt to changes.

There have been previous attempts to decrease the reliance on object texture, including previous use of edge-based features. The aperture problem (see Figure 2) renders these spatially unstable, as neighbouring pixels along the direction of the edge are indistinguishable [11]. For this reason, conventional trackers often avoid edges. However, such features are still valuable: *e.g.* Smith *et al.* used lines in a visual SLAM framework [29]. For tracking, Tsin *et al.* [30] fit line segments modelling the object to detected edge-points. The same approach was used in [6], [10], when searching for the pose of a wireframe-represented, user-specified, 3D object.

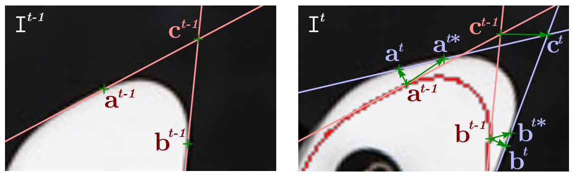
C. Contributions

Our approach uses principles similar to model-based tracking using edges, but the object model is learned in a completely unsupervised manner. We propose to overcome the aperture problem by estimating the geometry from edge-based line correspondences. To cope with situations involving insufficient point-to-point correspondences, line-to-line correspondences are used. These are robust to shifts along the edge. The line correspondences are employed within a robust motion estimation framework for frame-to-frame tracking. We also propose an approach to learning the manifold of observed object poses in a probabilistic manner. This probability is then directly used to reject "impossible" poses.

As a second major contribution we propose a redetection strategy that identifies when a tracking failure or object disappearance occurs and uses the online model of appearance, in conjunction with the learnt pose manifold, to relocate the object. This gives the tracker long-term stability, which combined with its inherent drift resistance renders it suitable for tracking long video sequences with full occlusions. It is the first long-term edge-based tracker. We dub the tracker LT-FLOtrack (Long-Term FeatureLess Object tracker).

Finally, we make the new sequences publicly available including manual ground-truth annotations. These will be made available for download from authors' website.

This paper unifies our previous conference publications [20] and [19], while bringing additional insight into the internal workings, new formalisation of the long-term module, and extended experimental evaluation. Performance of LT-FLOtrack is measured using the VOT2013 and VOT2014 benchmarks and it is shown to be one of the best performing (placed fifth out of almost 30 evaluated trackers on VOT2013). Additionally, the proposed tracker was evaluated using the VTB1.0 benchmark and placed third in the leader-board.



(a) Frame $t-1$ (b) Frame t

Fig. 2: Establishing line correspondences regardless of the aperture problem.

The LT-FLOtrack algorithm is introduced in Section II. It consists of two parts shown in Sections III and IV. Section V describes the probability density estimation and Section VI the redetection scheme. Our claims are experimentally validated in Section VII and conclusions are drawn in Section VIII.

II. Texture-less Tracking

The task of the tracker is to find a pose $\mathbf{P}^t$ (position, rotation, size) of an object in frame $\mathtt{I}^t$. The tracked object is represented as a set of edge points (locations of locally maximal intensity gradient). The fundamental elements for tracking are tentative correspondences of lines tangential to edges in the image. In other words, to estimate the frame-to-frame motion of the target object, we use correspondences of lines, which are in turn defined by correspondences of edge-points. This is a similar task to estimation of the image transformation from point correspondences. Edge points are present even in low-texture scenarios where point features are scarce, they are however more difficult to use.

Figure 2a shows two points $\{\mathbf{a}^{t-1}, \mathbf{b}^{t-1}\}$ identified on the contour of an object and their tangent lines. Attempting to locate the motion of these points in the next consecutive frame is ill-defined. Figure 2b shows that a local search normal to the edge direction incorrectly identifies correspondences $\{\mathbf{a}^t, \mathbf{b}^t\}$ which are shifted along the contour, instead of the true correspondence $\{\mathbf{a}^{t*}, \mathbf{b}^{t*}\}$. This is due to the well known aperture problem, which makes it impossible to detect the correct motion for points laying on edges. Should these incorrect corresponding pairs $\{(\mathbf{a}^{t-1}, \mathbf{a}^t), (\mathbf{b}^{t-1}, \mathbf{b}^t)\}$ be used directly as point-to-point corrsepondences, the estimated motion would be incorrect. However, under the assumptions of a small shift between two consecutive frames and a local linearity of the edges, these points generate the same tangent lines as the true correspondences. By using the intersection $\mathbf{c}^t$ of the tangent lines and its motion from the intersection in the previous frame ($\mathbf{c}^{t-1}$), transformations can be calculated using edge features while overcoming the aperture problem. The intersection can be seen as a *virtual corner point*, where the two edges forming the corner are allowed to be separated.

In other words, traditional techniques use point features (*e.g.* corners) to simultaneously provide the two orthogonal constraints needed to overcome the aperture problem. We instead use pairs of distant non-parallel lines, each providing one constraint, to define the virtual corners.

Figure 3 shows an overview of the LT-FLOtrack. It consists of two modules performing different tasks. The first is a *short-term tracker*, which finds line correspondences and estimates frame-to-frame transformations (block (i)). It then *updates* knowledge about the object in each frame (block (ii)), including the positions of good edges to track and observed edge stability (*edge quality field*). The short-term tracker is formalised in Algorithm 1.

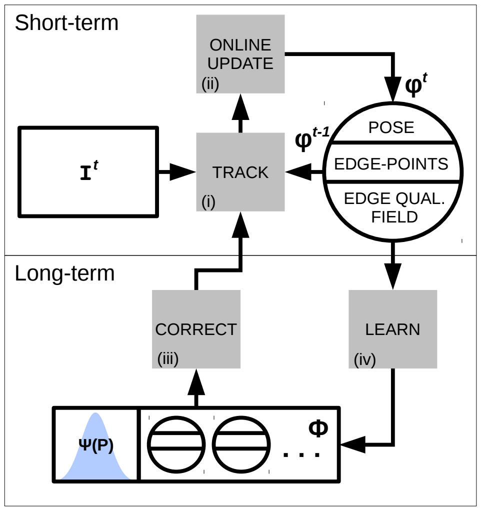


Fig. 3: Overview of the LT-FLOtrack algorithm.

TABLE I: Used notation.

Symbol	Meaning
$\mathtt{I}^t$	Frame t
$\mathbf{P}^t$	Object pose in frame t (location, size, orientation,...)
$(\mathbf{p}^t, \mathbf{q}^t)$	Point-to-point correspondence in frame t
$(\mathbf{k}^t, \mathbf{l}^t)$	Line-to-line correspondence in frame t ($\mathbf{k}^t$ is defined by $\mathbf{p}^t$ in $\mathtt{I}^{t-1}$, $\mathbf{l}^t$ is defined by $\mathbf{q}^t$ in $\mathtt{I}^t$)
S	Frame to frame transformation ($\mathbf{P}^t = \mathrm{S}(\mathbf{P}^{t-1})$)
$\mathtt{Q}^t$	Edge quality field
$\Psi(\mathbf{P})$	Probability distribution of $\mathbf{P}$
φ^t	State of the short-term tracker ($\mathbf{P}^t, \mathbf{p}^{t+1}, \mathtt{Q}^t$)
$d_G(\mathbf{l}, \mathbf{l}')$	Geometric difference between $\mathbf{l}$ and $\mathbf{l}'$

The second module maintains *long-term relations*. In cases of low confidence within the short-term tracker, a procedure to *correct* the pose is performed (block (iii)). Alternatively, if short-term tracking leads to a correct pose estimate, its state is stored (block (iv)) for future corrections. Block (iv) also includes updates to the observed pose probability distribution $\Psi(\mathbf{P})$. The used notation is summarised in Table I.

III. Short-term Tracker

To find the pose $\mathbf{P}^t$, the short-term tracker estimates a (similarity) transformation S, such that $\mathbf{P}^t = \mathrm{S}(\mathbf{P}^{t-1})$. Algorithm 1 describes this short-term part of LT-FLOtrack, where F is the total number of frames to process. Lines 4 to 8 correspond to the block (i) and lines 3 and 9 refer to the block (ii) of Figure 3. In each frame, a set of edge-points $\mathbf{p}^t$ is generated (line 3). Successfully matched correspondences (inliers to S) from the last frame ($\mathbf{q}^{t-1}$) are retained and new edge-points are generated to keep a stable number of correspondences. As this number has a significant impact on execution times, it should be as low as possible. To estimate a suitable number of correspondences for a given sequence, we propose a data-driven method, which accounts for the object's size and complexity; see Section III-A for details.

Algorithm 1 The Short-term Tracker

```
1: $\mathtt{Q}^1 \leftarrow$ initialise point quality field
2: for $t = 2 \rightarrow F$ do
3:     $\mathbf{p}^t, \mathbf{k}^t \leftarrow$ generate edge-points and lines $(\mathtt{I}^{t-1})$
4:     $\mathbf{q}'^t, \mathbf{l}'^t \leftarrow$ find tentative correspondences $(\mathbf{p}^t, \mathtt{I}^t)$
5:     $\mathrm{S}'' \leftarrow$ estimate transformation by LO-RANSAC $(\mathbf{k}^t, \mathbf{l}'^t)$
6:     $\mathbf{q}^t, \mathbf{l}^t \leftarrow$ find tent. correspondences $(\mathbf{p}^t, \mathtt{I}^t$, init by $\mathrm{S}'')$
7:     $\mathrm{S}' \leftarrow$ estimate transformation by LO-RANSAC $(\mathbf{k}^t, \mathbf{l}^t)$
8:     $\mathrm{S} \leftarrow$ refine transformation $(\mathrm{S}', \mathtt{Q}^{t-1})$
9:     $\mathtt{Q}^t \leftarrow$ update point quality field $(\mathbf{p}^t, \mathrm{S}, \mathtt{Q}^{t-1})$
10: end for
```

The first step is to determine which edges are good for tracking. These should be invariant to the brightness changes and evenly distributed on the object, *i.e.* strong edges should be selected where possible and weaker edges only in regions of low contrast. LT-FLOtrack uses an iterative procedure, which searches for strong nearby edges. A point $\mathbf{p}_0^t$ is randomly drawn from inside the object bounding box (given by $\mathbf{P}^{t-1}$). The edge search is then performed along a line normal to the edge direction [10], starting from this point, to find $\mathbf{p}_1^t$. This is iterated until convergence ($\mathbf{p}_n^t = \mathbf{p}_{n-1}^t$). Using only the gradient, this is dubbed an *unguided edge search*. The selected edge-points are defined as

$$\mathbf{p}_n^t = \underset{\bar{\mathbf{p}} \in \mathbf{p}_{n-1}^t + \lambda \vec{\nabla} \mathtt{I}^{t-1}(\mathbf{p}_{n-1}^t)}{\arg\max} ||\vec{\nabla}\mathtt{I}^{t-1}(\bar{\mathbf{p}})|| \cdot \exp(-\lambda^2/\sigma^2) \ , \tag{1}$$

where λ is the distance along the gradient direction and σ a scaling factor (based on the object size). Notice that since we search edge-point (and consecutively line) correspondences between the previous and the current frame, the points $\mathbf{p}^t$ are localised in $\mathtt{I}^{t-1}$.

When establishing correspondences (line 4), another edge search is required, however the task is different. Instead of locally strong edges, we seek edges similar to those from the previous frame. As such, a *guided* (using information from the previous frame) *edge search* is used. This searches for positions with similar gradient angle and local appearance. The search starts at the locations of edge-points from the previous frame:

$$\mathbf{q}'^t = \underset{\bar{\mathbf{q}} \in \mathbf{p}^t + \lambda \vec{\nabla} \mathtt{I}^{t-1}(\mathbf{p}^t)}{\arg\max} \left(\frac{\cos(\Delta\alpha)+1}{2}\right) \cdot \delta(\mathbf{p}^t, \bar{\mathbf{q}}) \cdot \Omega(\lambda) \quad \text{s.t.} \quad \bar{\mathbf{q}} \text{ is a local maximum of gradient} \ , \tag{2}$$

where $\delta(\mathbf{p}^t, \bar{\mathbf{q}})$ measures similarity of local appearance of $\mathtt{I}^{t-1}$ around $\mathbf{p}^t$ and $\mathtt{I}^t$ around $\bar{\mathbf{q}}$, $\Delta\alpha$ is the difference between gradient angles at $\mathtt{I}^{t-1}(\mathbf{p}^t)$ and $\mathtt{I}^t(\bar{\mathbf{q}})$ and Ω is a locality-preserving regularisation. We use cosine as a measurement of angle error due to its tolerance (the flat region around zero), robustness (no outlier over-penalisation) and absence of problems with angle periodicity. It is scaled and shifted to the $[0; 1]$ range. For the guided edge search, multiple search lines are used: normal to the edge in the last frame ($\vec{\nabla}\mathtt{I}^{t-1}(\mathbf{p}^t)$) and offset by $\pm\frac{\pi}{10}$.

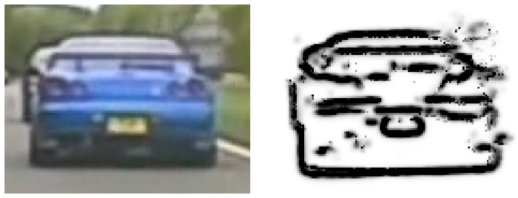


Fig. 4: Examples of an image and its edge quality field after several frames of tracking.

The tentative edge-point correspondences $(\mathbf{p}^t, \mathbf{q}'^t)$ are then transformed to line-to-line correspondences $(\mathbf{k}^t, \mathbf{l}'^t)$ using the gradient direction:

$$\mathbf{k}^t = \mathbf{p}^t + \lambda\mathbf{n}; \quad -\infty \leq \lambda \leq \infty \ , \tag{3}$$

where $\mathbf{n}$ is normal to the gradient:

$$\mathbf{n} = \mathtt{J}\vec{\nabla}\mathtt{I}^{t-1}(\mathbf{p}^t) \ , \tag{4}$$

using $\mathtt{J}$, a matrix of 2D counter-clockwise rotation,

$$\mathtt{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} . \tag{5}$$

The same process is performed for edge-points $\mathbf{q}'^t$ to get lines $\mathbf{l}'^t$.

LO-RANSAC [21] uses the correspondences $(\mathbf{k}^t, \mathbf{l}'^t)$ to estimate a geometric transformation S'' between the two frames (Algorithm 1, line 5), maximising *image evidence* E^t. This is defined as the average fit of edge-points from $\mathtt{I}^{t-1}$ to the edges in $\mathtt{I}^t$. The computation of image evidence is based on an *oriented Chamfer distance* [24], [28] as

$$E^t(\mathrm{S}'') = \frac{1}{N^t} \sum_{\mathbf{p}^t} e_{\mathbf{p}^t} \cdot \mathit{\Lambda} \ , \tag{6}$$

$$e_{\mathbf{p}^t} = \frac{1}{1 + \mathrm{d}(\mathrm{S}''(\mathbf{p}^t))} \cdot \frac{\cos(\Delta\alpha)+1}{2} \ , \tag{7}$$

where $\mathrm{d}(\cdot)$ is Euclidean distance of a point to the nearest Canny's edge [3], $\Delta\alpha$ is the difference between the gradient angle of a point and its nearest edge (taking rotation induced by S'' into account), N^t is the number of correspondences and $\mathit{\Lambda}$ is a regularisation term penalising large changes between $\mathbf{P}^{t-1}$ and $\mathbf{P}^t$ (smoothness enforcing prior). The minimal sample for RANSAC is a triplet of lines, whose intersections (the virtual corner points) are used to generate a transformation hypothesis. To get a more precise transformation with a higher number of inliers, we repeat the process using the transformation estimate as initialisation (lines 6 and 7). In this second iteration, the new locations of correspondences $\mathbf{q}^t$ and $\mathbf{l}^t$ are computed. LO-RANSAC is then executed again using the new correspondences.

The estimated transformation S' is usually more accurate than S'', however it may still be noisy, which would ultimately result in tracker drift. It is therefore necessary to stabilise the estimation relative to previous frames. In LT-FLO this is done by learning the locations of edges, which have previously predicted a correct transformation. This knowledge is stored as an *edge quality field* $\mathtt{Q}^t$, giving an estimate of object structure (stable edges, see Figure 4 for an example). We expect the corresponding edges in the new frame to fit to this model of previously stable edges w.r.t. the estimated transformation (line 8 of Algorithm 1):

$$\mathrm{S} = \arg\max_{\bar{\mathrm{S}}} \sum_{\mathbf{q}^t} \mathtt{Q}^{t-1}(\bar{\mathrm{S}}^{-1}(\mathbf{q}^t)) \ ; \tag{8}$$

maximised using Nelder-Mead iterative optimisation of the transformation parameters. In every frame, $\mathtt{Q}^t$ is updated as follows:

$$\mathtt{Q}^t = \omega \cdot \mathrm{S}(\mathtt{Q}^{t-1}) + \bigcup_{\mathbf{p}^t} e_{\mathbf{p}^t} \ , \tag{9}$$

where ω is a forgetting factor (line 9 of Algorithm 1). The field $\mathtt{Q}^1$ is initialised taking all the edge-points from the first frame as reliable (line 1).

We define a set of inliers $\mathcal{I}^t(\mathrm{S})$ as a subset of correspondences $(\mathbf{k}^t, \mathbf{l}^t)$ having low geometric error d_G with respect to the estimated transformation S (frame indices t omitted):

$$\mathcal{I}(\mathrm{S}) = \left\{ (\mathbf{k}, \mathbf{l}) \,\middle|\, \sqrt{d_G(\mathbf{l}, \mathrm{S}(\mathbf{k}))^2 + d_G(\mathbf{k}, \mathrm{S}^{-1}(\mathbf{l}))^2} \le \theta_d \right\} . \tag{10}$$

This can be equivalently seen as a set of point-to-point correspondences $(\mathbf{p}^t, \mathbf{q}^t)$, since lines are defined by edge-points. See Section III-B for details on the geometric error of line correspondences. The subset of edge-point inliers $\mathbf{q}^t$ (*i.e.* defining lines belonging to the inlier set) is retained for use in the next frame as $\mathbf{p}^{t+1}$. Additional edge-points are then generated as described above (line 3 of Algorithm 1), which encloses the short-term tracking loop.

To allow reference to the short-term tracker in a compact form, we adopt the following notation. The complete *state* of the tracker (or model of the object) will be referred to as φ^{t-1} and includes information about the object pose $\mathbf{P}^{t-1}$, edge-points[1] $\mathbf{p}^t$ and the *edge quality field* $\mathtt{Q}^{t-1}$. The short-term tracker can then be seen as a series of consecutive calls to a tracking function, initialised at $\mathbf{P}^{t-1}$:

$$\mathbf{P}^t = \mathrm{T}_{\varphi^{t-1}}(\mathtt{I}^t, \mathbf{P}^{t-1}) \ . \tag{11}$$

The update of the current state comprises of updating $\mathtt{Q}^{t-1}$ to $\mathtt{Q}^t$ according to Equation (9), estimating $\mathbf{P}^t = \mathrm{S}(\mathbf{P}^{t-1})$ and generating new points $\mathbf{p}^{t+1}$, which can be concisely summarised as

$$\varphi^t = \mathrm{update}(\varphi^{t-1}) \ . \tag{12}$$

A. On the Number of Generated Edge-points

To estimate a sufficient number of correspondences for the successful tracking of a given sequence, we employ an approach that accounts for the object's size and complexity. The number of final correspondences after the edge search is usually lower than the number of generated edge-points. Furthermore, this dependency is strongly non-linear and saturates (see Figure 5).

In LT-FLOtrack, this saturation level is found in the first frame by initialising with a high number of random points. Then the number of generated points is set proportional to the saturation level. This is adjusted according to the observed scale changes in subsequent frames.

[1]Points $\mathbf{p}^t$ are computed from $\mathtt{I}^{t-1}$.

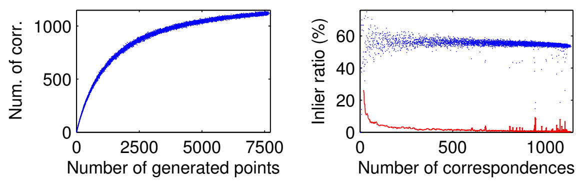


Fig. 5: Dependences of the number of correspondences and inliers on the number of generated points. The red line in the right image shows the standard deviation in bins of size 20. In this particular case (beginning of the DUDEK sequence), we can expect stable behaviour with about 350 correspondences, thus it is enough to generate 500 new points.

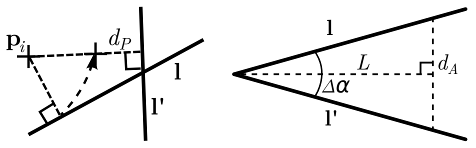


Fig. 6: Geometric meaning of d_P and d_A.

B. Geometric Error of Line Correspondences

Previously, we have worked with terms such as "inliers", or "consistent line correspondence", which require a measure of distance between projected and measured line features (an equivalent to the projection error for point correspondences). But what does it mean for two lines $\mathbf{l}$ and $\mathbf{l}'$ to be close to each other? Hartley [12] stated that distance (or geometric error) of lines has to be measured with respect to some point of interest. He suggested to use the distance between a line and line segment. This approach yields usable results. However, our lines are not defined by segments, and are infinite in extent. It is therefore necessary to calculate intersections between the lines and all four sides of the tracked object bounding box. Computational complexity is then prohibitively large.

A more feasible approach is to see the distance as two independent components – difference of angles d_A and difference of position d_P with respect to a given *point of interest* $\mathbf{p}_i$ (e.g. centre of gravity of the tracked object, as can be seen in Figure 6). The error of position is defined as the difference between the distances from the lines to $\mathbf{p}_i$, in normalised homogeneous coordinates as:

$$d_P = \left|\mathbf{p}_i^T \mathbf{l}\right| - \left|\mathbf{p}_i^T \mathbf{l}'\right| \ . \tag{13}$$

The angular error is defined as the length of the shortest possible line segment with endpoints on the lines. The distance between this segment and the intersection of the lines is a constant L, which can be derived from the size of the tracked object, or set manually.

$$d_A = 2 \cdot L \cdot \tan \frac{\Delta\alpha}{2} \ , \tag{14}$$

where $\Delta\alpha$ is the angle between the lines. Finally, the geometric error term is computed as

$$d_G = \sqrt{d_P^2 + d_A^2} \ . \tag{15}$$

This technique gives errors similar to Hartley's approach in significantly lower time (10-fold speed-up with correlation coefficient 0.9). It should be noted that d_P is strongly underestimated in the case of $\mathbf{p}_i$ laying *between* the lines. However, as we are usually concerned with the distance of lines that are close to each other, this condition appears rarely (correspondences are incorrectly classified as inliers less that one percent of the time).

IV. Long-term Module

Using line-correspondences for short-term tracking works well for short sequences. However, for longer sequences it suffers from error accumulation (drift) and is not robust to severe occlusions.

The long-term module of LT-FLO continuously checks the *image evidence* score E^t, as this is a good indicator of the quality of the estimated transformation. When E^t decreases suddenly, this indicates a problem (the confidence in the current solution is low). On such an occasion, the short-term tracker may experience difficulties and may need *correction* (block (iii) of Figure 3). The desired property of the (local) correction is that it can, given the last known pose of the tracked object, estimate its new pose regardless of any drift in the short-term tracker. Furthermore, it should identify a disappearance of the object (either because of a tracker failure or a full occlusion) and start a global *redetection*.

A correction procedure is proposed, which fulfils these requirements and works both on a local level and in a redetection scenario. The long-term module has several *states* of the short-term tracker stored – a set Φ (initialised as $\Phi = \{\varphi^1\}$). When a tracking failure is detected, the short-term tracker is initialised using each of the stored states φ^u at the last known valid pose $\mathbf{P}^{t-1}$. A process analogous to the standard short-term tracking function $\mathrm{T}_{\varphi^{t-1}}$ is performed, yielding several *correcting* hypotheses in addition to the *current* one (using φ^{t-1}). Each hypothesis is assigned a score Γ, based on transformation quality, temporal consistency and the inlier ratio (Equations (6, 8&10)), where

$$\Gamma\left(\mathrm{T}_\varphi(\mathtt{I}^t,\mathbf{P})\right) = E^t(\mathrm{S}_\varphi)\cdot\sum_{\mathbf{q}^t}\mathtt{Q}(\mathrm{S}_\varphi^{-1}(\mathbf{q}^t))\cdot\sqrt{\frac{|\mathcal{I}^t(\mathrm{S}_\varphi)|}{N^t}}\ , \tag{16}$$

S_φ is the estimate of the transformation given by tracking from a state φ. The best correcting hypothesis is selected as:

$$\varphi^* = \operatorname*{arg\,max}_{\bar\varphi\in\Phi}\Gamma\left(\mathrm{T}_{\bar\varphi}(\mathtt{I}^t,\mathbf{P}^{t-1})\right)\ , \tag{17}$$

where all the corrections are initialised from the position $\mathbf{P}^{t-1}$. It is then used for *correction* where appropriate, replacing the current estimate:

$$\mathrm{S} = \begin{cases}\mathrm{S}_{\varphi^*} \\ \ \text{if}\ \ \Gamma(\varphi^*) > \Gamma(\varphi^{t-1}) \bigwedge ||\mathbf{P}(\varphi^*) - \mathbf{P}(\varphi^{t-1})|| \geq \theta_2 \\ \mathrm{S}_{\varphi^{t-1}} \qquad \text{otherwise}\ ,\end{cases}$$

$$\varphi^t = \begin{cases}\text{update}(\varphi^*) \\ \ \text{if}\ \ \Gamma(\varphi^*) > \Gamma(\varphi^{t-1}) \bigwedge ||\mathbf{P}(\varphi^*) - \mathbf{P}(\varphi^{t-1})|| \geq \theta_2 \\ \text{update}(\varphi^{t-1}) \qquad \text{otherwise}\ ,\end{cases} \tag{18}$$

where $\mathbf{P}(\varphi^*)$ is a shorthand for $\mathrm{T}_{\varphi^*}(\mathtt{I}^t,\mathbf{P}^{t-1})$ and $\Gamma(\varphi^*)$ for $\Gamma\left(\mathrm{T}_{\varphi^*}(\mathtt{I}^t,\mathbf{P}^{t-1})\right)$ (and analogously for φ^{t-1}, they are used for conciseness also further in the text). The threshold θ_2, as well as θ_1 (see below), are calculated from the object size.

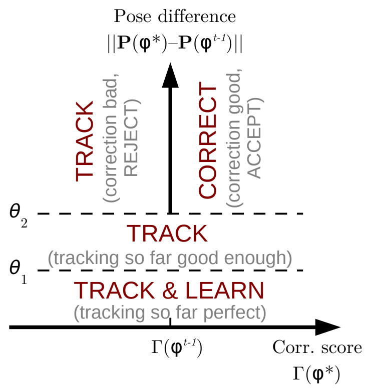


Fig. 7: Possible situations during correction. Thresholds for the decisions are calculated from the object size.

If the best *correcting* hypothesis estimated is an object pose similar to the *current* one, $||\mathbf{P}(\varphi^*) - \mathbf{P}(\varphi^{t-1})|| < \theta_2$, it is a signal that the original estimated pose was correct and the current state is not replaced as it is expected to be better adapted to the current object appearance. In the extreme case, when the agreement of the estimated pose is (almost) exact, the current state is stored for use in future corrections (block (iv) of Figure 3):

$$\Phi_{\text{new}} = \begin{cases}\Phi\cup\varphi^{t-1} & \text{if}\ ||\mathbf{P}(\varphi^*) - \mathbf{P}(\varphi^{t-1})|| < \theta_1 \\ \Phi & \text{otherwise}\ .\end{cases} \tag{19}$$

See Figure 7 for a schema of these situations. The estimated pose $\mathbf{P}$ and score Γ of φ^* are compared with the ones obtained before the correction (using φ^{t-1}).

The corrections may consume a significant portion of the execution time. As the asymptotic time complexity is $O(F\cdot|\Phi|)$, it is not feasible to keep all observed states. Therefore a method is needed to maximise the diversity of the learned states to cover as much variation in object appearance as possible in a fixed space and time (in our experiments, $|\Phi|_{\max} = 5$). Therefore a limit is placed on the cardinality of Φ and when it is reached, the state used least often in recent corrections is replaced, as this is a good indicator of a state's usefulness in the future:

$$\Phi_{\text{new}} = \Phi \setminus \operatorname*{arg\,min}_{\bar\varphi\in\Phi}\Upsilon(\bar\varphi)\ , \tag{20}$$

where Υ is the number of occasions in the past when $\bar\varphi$ was selected according to Equation (17). This prunning is carried out before expanding the set Φ in the former option of Equation (19).

V. Probability of Object Pose

For many sequences, the object-and-camera system does not use the entire parameter space of object poses. Instead, only a significantly smaller subspace is occupied. An example would be the appearance of a car, followed from the rear. In such a scenario, there is almost no rotation and scale is correlated

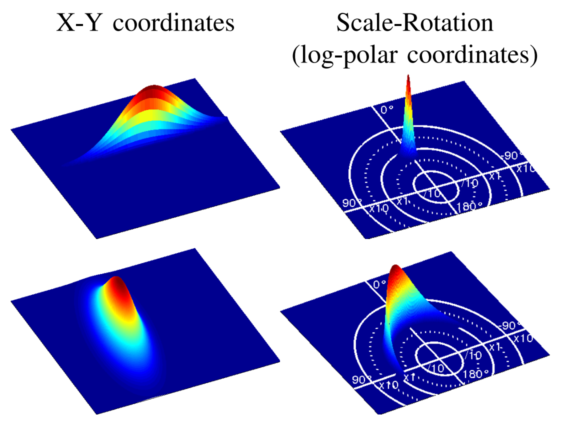


Fig. 8: Examples of probability distribution Ψ (scale relative to the initial size). Top row: tracking a car, change only in the x-coordinate (visible steps in the upper left image indicate how narrow Ψ is). Bottom row: tracking the PAGE sequence with large variation in rotation.

with y-coordinate (related to the actual distance between the cars), see Figure 8.

This property of the sequences can be exploited to obtain prior information about the object pose $\mathbf{P}^t$. The distribution Ψ of the object pose is modelled as a multivariate Gaussian distribution ($\Psi = \mathcal{N}(\boldsymbol{\mu}_\Psi, \Sigma_\Psi)$, log-normal for the scale component). This models the distribution well despite the fact that the true distribution is often not unimodal). We learn the distribution's parameters online, from observed object poses, according to [2]. In the t-th frame, the update is calculated as:

$$\boldsymbol{\mu}^t = \boldsymbol{\mu}^{t-1} + \frac{\mathbf{P}^t - \boldsymbol{\mu}^{t-1}}{t} \tag{21}$$

and

$$\Sigma^t_{i,j} = \frac{\Sigma^{t-1}_{i,j}(t-1) + \left(P^t_i - \mu^{t-1}_i\right)\left(P^t_j - \mu^{t-1}_j\right)\frac{t-1}{t}}{t} , \tag{22}$$

where P^t_i denotes the i-th component of a variable $\mathbf{P}$ in the t-th frame and similarly $\Sigma^t_{i,j}$ is the (i,j)-th element of the covariance matrix.

Given this probability distribution Ψ, we can make assumptions about the object's pose. For example, false "corrections" with extremely low probability can be rejected. This adds the following constraint to the optimisation in Equation (17):

$$\text{s.t.} \quad \Psi(\mathrm{T}_{\bar{\varphi}}(\mathtt{I}^t, \mathbf{P}^{t-1})) > \theta_\Psi \ . \tag{23}$$

An example of this constraint's value is a rectangular object, which looks very similar when rotated. However, the low probability of this pose helps us identify such a situation as incorrect.

VI. Failure and Redetection

The first step of the redetection procedure is to identify object disappearances and/or tracking failure. The proposed approach to the failure discovery is based on the following observation. In the correction stage of LT-FLO (block (iii) of Figure 3), the points from a stored state are applied to the current frame, to search for correspondences. When they are moved to an area of weak or noisy gradient, the direction of the gradient at a particular pixel is not in accordance with its neighbourhood, *i.e.* the following does *not* hold:

$$|\mathrm{atan2}(\mathtt{I}_x(\mathbf{p}), \mathtt{I}_y(\mathbf{p})) - \mathrm{atan2}(\bar{\mathtt{I}}_x(\mathbf{p}), \bar{\mathtt{I}}_y(\mathbf{p}))| < 90° \ , \tag{24}$$

where $\mathtt{I}_x, \mathtt{I}_y$ are components of the image gradient $\vec{\nabla}\mathtt{I}$ (i.e. $\mathtt{I}_x = \frac{\partial \mathtt{I}}{\partial x}, \mathtt{I}_y = \frac{\partial \mathtt{I}}{\partial y}$) and $\bar{\mathtt{I}}$ indicates the average over the 3×3 neighbourhood of $\mathbf{p}$. A failure is identified, when there is an unusually high number of such points across all the corrections. The threshold is set using the 99-th percentile of the fitted normal distribution for that sequence.

Object disappearance and tracking failures are also detected, based on the geometric properties of the object pose. Specifically, when the tracked rectangle is too small (sizes under 10 px render the similarity function δ unreliable, objects are thus considered too distant to track), larger than the image, when a large portion of the tracked rectangle is outside the image (more than three quarters of the object is out of scene) or when there are no inliers to the estimated transformation.

Once the tracker detects that the object is lost, a redetection occurs. Firstly, a local correction is performed (Section IV, Equation (17)). This is followed by a global redetection, employing the stored states Φ. Instead of initialisation by $\mathbf{P}^{t-1}$, the global redetection is initialised at several random poses $\mathbf{P}_{\mathrm{rand}}$, sampled from the observed object pose distribution Ψ:

$$\varphi^* = \underset{\bar{\varphi} \in (\Phi \cup \varphi^{t-1})}{\arg\max} \ \Gamma\left(\mathrm{T}_{\bar{\varphi}}(\mathtt{I}^t, \mathbf{P}_{\mathrm{rand}})\right) \ ; \quad \mathbf{P}_{\mathrm{rand}} \sim \Psi \ . \tag{25}$$

The best hypothesis is used for correction in the same way as T_{φ^*} in Equation (18).

After a failure, this global redetection is performed in every frame, until a good pose is found. Using the pose distribution to guide the search space is more efficient than a dense sampling of the whole parameter space. To improve the ability to generalise to possible, but previously unseen or rare poses, the search space is extended by multiplying the covariance matrix Σ_Ψ by a constant factor during redetection.

VII. Experimental Evaluation

A. Short-term Tracking

The performance of the LT-FLOtrack algorithm was first evaluated in a short-term tracking scenario, against a number of competitive SOTA approaches – TLD [15], LGT [4] and FoT [23] using both standard and low-textured sequences. Properties of the sequences are summarised in Table II and the first frames are shown in Figure 9. The same settings were used for all the sequences. Although the global redetection was enabled during all the experiments, it was not required in the short-term tracking scenario and only local corrections were employed.

The speeds are shown in Table III. It should be noted that while the trackers are generally implemented as a compiled (C/C++) core with Matlab front-end, the FoT tracker is written completely in C++. All the measurements in this publication were carried out on a computer with the Intel i7-2600 processor (3.4 GHz, single core used). LT-FLO takes about 300 MB of RAM.

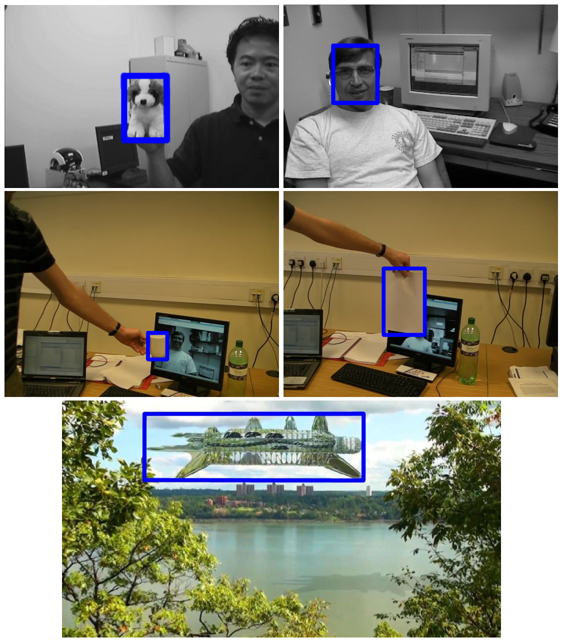


Fig. 9: Short-term tracking dataset. From top and left: DOG [5], DUDEK [14], MUG^N, PAGE^N and SPACESHIP^N.

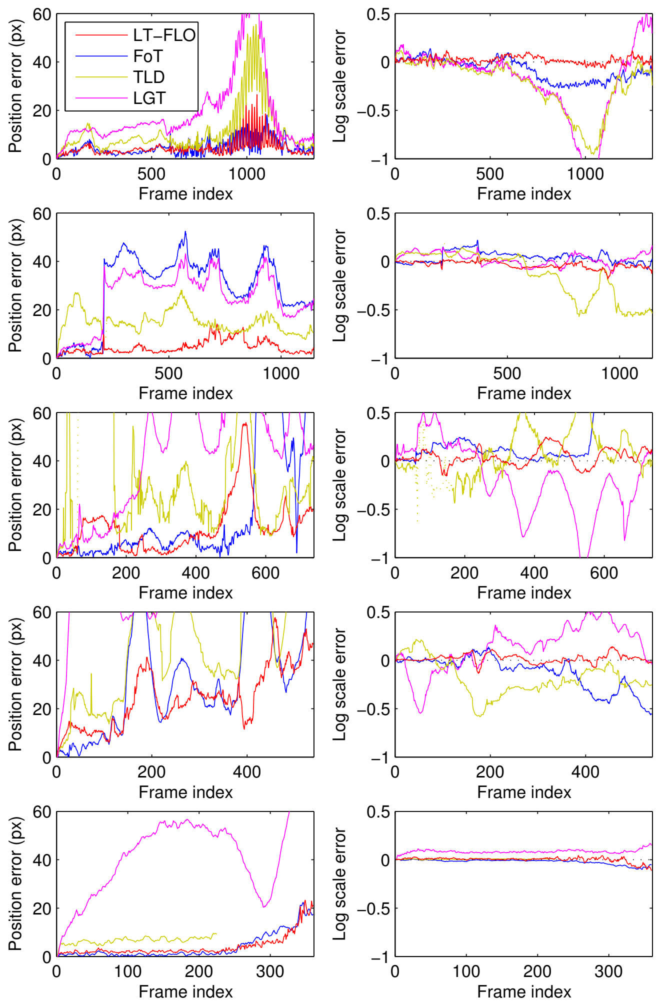


Fig. 10: Results of short-term tracking evaluation. From top to bottom: DOG, DUDEK, MUG, PAGE and SPACESHIP.

TABLE II: Experimental video sequences for short-term tracking. Sequences marked by ^{N} are new and will be made available online with ground truth.

Name	Resolution	Frames
DOG [5]	320×240	1 353
DUDEK [14]	720×480	1 145
MUG^N	640×480	737
PAGE^N	640×480	539
SPACESHIP^N	640×360	360

TABLE III: Speed comparison of different trackers (in FPS).

Name	LT-FLO	TLD	LGT	FoT
DOG	6.3	4.3	3.4	405.5
DUDEK	3.3	2.0	2.1	131.7
MUG	4.8	2.6	2.8	230.6
PAGE	3.1	2.7	2.5	212.4
SPACESHIP	1.5	5.0	4.6	112.9

Figure 10 shows the results. Performance is measured using *location error* (distance of the bounding box centre from its ground truth position) and *scale error* (logarithm of ratio of the estimated object size to its ground truth size, 0 means no error at average). The values were averaged over 20 executions.

For the DOG sequence, the most challenging part is between frames 700 and 1200, with a strong scale change and occlusion by the image boundary. While LT-FLO has no major problems and FoT experiences only light scale drift, LGT and TLD have severe problems, both in localisation and scale estimation.

The largest challenge of the DUDEK sequence comes around the 210th frame, when the face is occluded by the right hand. While LT-FLO's pose is corrected in several frames, TLD requires a significantly longer time and the other trackers never fully recover (see Figure 11 for details). LT-FLO also experiences difficulties around frame 800, where background points influence tracking and cause drift. Nevertheless, LT-FLO recovers in every run.

Due to non-existent texture in the MUG scene, LGT is unable to track this sequence; the points simply drift off the mug and stay at the person's wrist. TLD often suffers from under- or overestimation of the object size and sometimes loses it completely. We mark these frames, where the object was (incorrectly) reported as missing in more than half the runs, by a dotted line. Frames where the object is always lost have no yellow plot at all. FoT works well in this sequence until around frame 550, where it fails. LT-FLO is comparable up to around frame 500, at which point tracking is lost in some of the runs, resulting in a poorer average score. However, the object is successfully re-detected before frame 600.

For PAGE, LGT performs similarly to the MUG sequence, all the points stabilise at the person's hand and wrist. FoT uses only features from fingers and TLD often loses tracking and rarely re-detects even when the paper returns to a pose similar to the initial one. LT-FLO experiences difficulties, but still significantly outperforms all the others. Due to higher errors observed in this sequence for all the trackers, a rescaled plot of the position error is shown in Figure 11.

SPACESHIP is an *augmented reality* sequence (computer generated object on a real background). The tracked object becomes mostly transparent with only the outline roughly

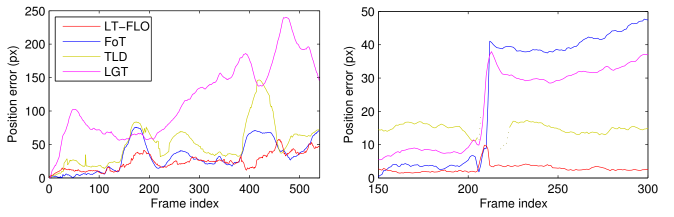


Fig. 11: Results of short-term tracking evaluation: the PAGE sequence and a detail of the most challenging part of the DUDEK sequence.

TABLE IV: Experimental sequences for long-term tracking.

Name	Resolution	Frames
CARCHASE [15]	290×217	9 928
PANDA[15]	312×233	3 000
VOLKSWAGEN[15]	640×480	8 576
LIVERRUN^N	320×240	29 700
NISSAN^N	640×480	3 800

visible. TLD reflects this by confidently reporting it as missing. LGT fails to separate the target from the background, leading to high errors from the beginning and a failure later when it starts moving. Both FoT and LT-FLO track successfully throughout the sequence until the end when the object is partially occluded by trees. Their performance is comparable; FoT having slightly lower position error and LT-FLO lower scale error.

B. Long-term Tracking

In the short-term tracking scenario, LT-FLOtrack's performance is comparable or superior to state-of-the-art trackers. However, it obviously still loses track in cases of full occlusion/disappearance, thus it is necessary to test robustness of trackers to these conditions. For the evaluation of long-term tracking, different sequences are necessary. Not only should they be longer, but more importantly they must include full occlusions, background clutter and scale and illumination changes [15]. A common source of such sequences are traffic scenes such as car chases, when a vehicle is followed by another (possibly aerial) vehicle. The majority of sequences in this evaluation are therefore of this type. These sequences explore redetection capability and predisposition to drift. Their properties are tabulated in Table IV and the first frames are shown in Figure 12.

In a long-term tracking scenario, it is necessary to check whether the detection/tracking is precise (when the overlap with the ground truth bounding box is higher than 50 %) as well as to check for successful detection of object disappearance. Kalal *et al.* [15] applied the precision/recall/F-measure comparison. To check sensitivity of trackers to the initialisation, we run the experiments multiple times with the bounding box in the first frame shifted by 5 % in both horizontal and vertical direction and also scaled up and down by 5 % (averaging the results over all 7 possibilities). We also introduce the concept of partial occlusion into the evaluation process. When the object is not fully visible, but not fully occluded, the tracker can receive a score for either overlap of bounding boxes or for reporting disappearance.

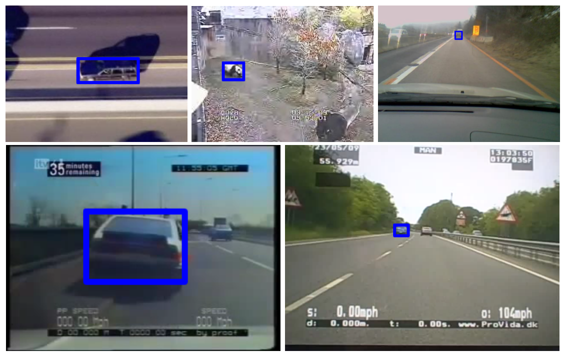


Fig. 12: Long-term tracking dataset. From top and left: CARCHASE [15], PANDA [15], VOLKSWAGEN [15], LIVERRUN^N and NISSAN^N.

TABLE V: Results of long-term tracking. Tabulated values are in the format: *F-measure (speed in FPS)*. The mean values (and mean st. deviations) are weighted by number of frames.

Name	LT-FLO	TLD
CARCHASE	**.42**±.07 (3.2)	.15±.08 (11.3)
PANDA	.17±.04 (4.1)	**.23**±.07 (12.0)
VOLKSWAGEN	.51±.20 (3.0)	**.62**±.13 (5.4)
LIVERRUN	**.56**±.20 (6.1)	.29±.28 (4.0)
NISSAN	**.88**±.14 (4.1)	.63±.14 (4.4)
mean	**.53**±.17 (4.9)	.36±.22 (6.0)

We compare our tracker with TLD [15], the first explicitly long-term tracker. The results can be found in Table V. The trackers were initialised by a tight bounding box. It should be noted that the experiments of TLD on their own dataset have significantly different results if their own initialisation is used (better on the CARCHASE and PANDA sequences and worse on the VOLKSWAGEN sequence, relating to F-measures of 0.45, 0.49 and 0.57 respectively). This indicates high sensitivity to the initialisation.

The results show that LT-FLO is capable of long-term tracking with competitive performance. In three out of five of the tested sequences it performed significantly better than TLD and comparably on one sequence. Also the average performance is better than TLD. The worst results of LT-FLO were acquired on the PANDA sequence. This highlights sensitivity to highly non-rigid objects with out-of-plane rotation.

Experiments were carried out with the short-term trackers as well, but results of these are not tabulated as they lost tracking at or before the first full occlusion (*e.g.* frame 440 for LIVERRUN or 1860 for NISSAN). These include FoT, LGT and LT-FLO without *global correction*, which proves its importance in long-term tracking.

C. Contributions of Algorithm Components

Local Corrections: To test the contribution of local corrections, only the short-term tracker (as introduced in Section III) was executed. See the results in Figure 13. First, we run the short-term tracker on the DOG sequence. The results are comparable to those of LGT or TLD, with errors almost an order of magnitude worse than those of LT-FLO.

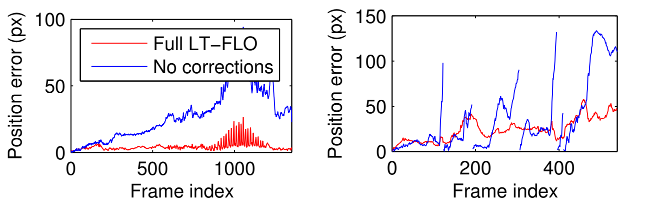


Fig. 13: Tracking without local corrections. Left: DOG, right: PAGE.

TABLE VI: Effect of elements in Equation (16): short-term tracking (mean bounding box overlap tabulated) and long-term tracking (F-measure tabulated).

Name	E	$\mathtt{Q}$	$\mathcal{I}$	$E\cdot\mathtt{Q}$	$E\cdot\mathcal{I}$	$\mathtt{Q}\cdot\mathcal{I}$	$E\cdot\mathtt{Q}\cdot\mathcal{I}$
DOG	0.76	0.73	0.74	0.73	0.76	0.74	**0.77**
PAGE	0.18	0.13	0.27	0.20	0.23	0.32	**0.61**
PANDA	0.14	0.10	0.08	0.12	0.10	**0.21**	**0.21**
NISSAN	0.20	0.42	0.31	0.25	0.41	0.78	**0.88**

The PAGE sequence is more difficult. Without the corrections, the short-term tracker failed completely several times, usually after approximately 100 frames. At this point (when the bounding box overlap decreased below 10 %) we restarted the tracking from GT, hence the low error after each discontinuity. In this case, the short-term tracker was not able to track the object throughout the sequence, with high errors reported very shortly after each manual intervention.

Transformation Quality Measures: To test the contribution of different components of the transformation quality measure (E, $\mathtt{Q}$ and $\mathcal{I}$ in Equation (16)), we tried removing each of them and testing the final performance of the tracker. Table VI summarises the effect of the elements in both the short-term (DOG, PAGE) and long-term (PANDA, NISSAN) scenarios. While in simple sequences (DOG) the variation is low, more challenging scenarios have significant differences in performance. The full technique performs the best in all cases.

Thresholds Selection: Finally, we test the effect of changing the algorithm parameters and its robustness to such selection. The most important parameters (besides the number of generated edge-points discussed in Section III-A) are the position error thresholds θ_1 and θ_2 as specified in Section IV. These are defined relative to the object size, as a portion of the bounding box diagonal. In our implementation, $\theta_1 = 0.03$ and $\theta_2 = 0.06$ are used.

Figure 14 shows the performance of LT-FLOtrack as a function of these thresholds. For the DOG sequence, there is no significant increasing or decreasing trend, indicating low sensitivity to these parameters. For the PAGE sequence a similar trend is seen, however the variance is greater. This is due to the more challenging nature of the sequence and the effects of random selection within the tracker itself.

D. VOT Challenge 2013 Results

In this section, we report the performance of our proposed tracker on the standard VOT2013 dataset, in the VOT Challenge benchmark scenario [17]. The results are tabulated in Table VII. We report only results of experiments 2 and 3,

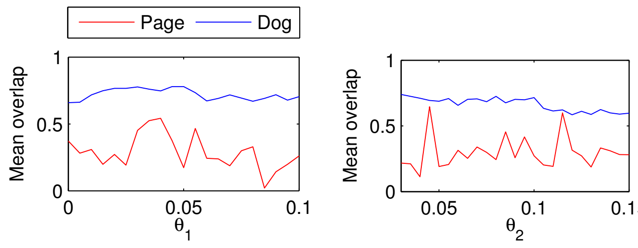


Fig. 14: Sensitivity of the method to parameters: the learning threshold θ_1 (left) and the correction threshold θ_2 (right).

TABLE VII: Results of the VOT2013 benchmark on LT-FLOtrack. The tabulated values are accuracy, robustness and speed in FPS.

	grayscale			region_noise		
Name	Acc.	Rob.	Speed	Acc.	Rob.	Speed
BICYCLE	0.58	1.73	4.52	0.57	1.60	4.10
BOLT	0.48	5.27	3.16	0.47	4.87	2.99
CAR	0.44	2.00	2.37	0.44	1.33	3.43
CUP	0.87	0.00	8.65	0.79	0.00	9.09
DAVID	0.64	0.07	6.54	0.63	0.33	5.70
DIVING	0.38	1.53	3.06	0.37	1.67	2.72
FACE	0.82	0.00	5.46	0.72	0.07	5.32
GYMNASTICS	0.53	1.33	2.95	0.50	1.20	2.45
HAND	0.45	4.93	3.89	0.45	4.67	4.38
ICESKATER	0.39	2.47	2.51	0.43	1.80	2.26
JUICE	0.89	0.00	6.74	0.78	0.07	6.87
JUMP	0.52	0.27	3.55	0.52	0.20	3.61
SINGER	0.60	0.53	0.67	0.60	0.20	0.76
SUNSHADE	0.68	1.40	5.33	0.63	1.47	5.73
TORUS	0.58	2.27	4.19	0.55	1.87	4.62
WOMAN	0.51	6.33	3.10	0.49	5.53	3.18

TABLE VIII: Overall results of the VOT Challenge 2013 [17]. The trackers are sorted according to combined performance.

Tracker	Accuracy	Robustness	Combined
PLT	5.9	3.0	4.5
FoT	4.3	10.4	7.3
EDFT	7.9	11.8	9.8
LGT++	14.8	5.2	10.1
LT-FLO	**7.3**	**14.8**	**11.0**
GSDT	10.7	11.4	11.1
SCTT	6.1	16.5	11.3
CCMS	10.0	12.7	11.4
LGT	17.3	5.9	11.6
Matrioska	10.1	13.3	11.7
AIF	7.7	16.2	12.0
Struck	11.5	12.5	12.0
DFT	10.8	13.7	12.2
IVT	10.4	14.8	12.6
ORIA	12.2	15.1	13.7
PJS-S	12.6	15.3	13.9
TLD	9.5	20.2	14.8
MIL	17.9	12.8	15.4
RDET	19.9	11.1	15.5
HT	20.0	12.9	16.5
CT	21.1	13.2	17.1
Meanshift	20.5	15.7	18.1
SwATrack	18.9	19.6	19.3
STMT	22.0	19.3	20.6
CACTuS-FL	24.2	17.8	21.0
ASAM	21.7	23.0	22.4
MORP	25.1	26.7	25.9

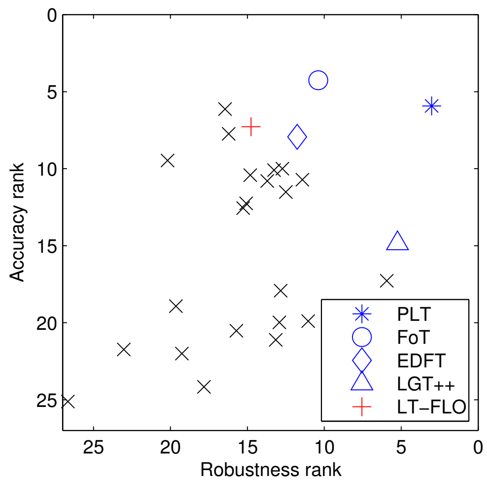


Fig. 15: Accuracy-robustness plot of the VOT Challenge 2013 results. For each tracker, the accuracy and robustness rankings were computed as a mean over all three experiments. Legend shown only for the best five trackers, see [17] for a full report.

since LT-FLOtrack performs equally well on both coloured and greyscale sequences. The performance of LT-FLOtrack is the best on sequences where a rigid object is tracked (Cap, Juice). On the other hand, sequences with highly non-rigid objects (Diving, Hand) score lower, as well as cases of strong out-of-plane rotation (Car).

Table VIII brings an overall comparison of all trackers competing in the VOT Challenge 2013. In this highly competitive challenge, LT-FLOtrack performed favourably compared to other state-of-the-art trackers and finished fifth place in the competition of 27. See Figure 15 for a visual comparison of trackers' performance in the challenge. Rankings in both accuracy and robustness are plotted, the higher and further right a tracker is, the better its rank. LT-FLOtrack, while having only average robustness (mainly due to its inability to track articulated objects), excels in the accuracy ranking, where its drift-resistant nature proves invaluable. However, it should be noted that the VOT Challenge does not test properties where LT-FLO is the strongest, in particular low texture and long sequences (possibly with full occlusions), but it does provide evidence that it remains comparable on these simpler sequences.

E. VOT Challenge 2014 Results

The same experiments were carried out on the VOT2014 dataset – see Table IX for results [18]. Among the best scoring are, unsurprisingly, rigid-object sequences such as Sphere or Car (not to be confused with the VOT2013 sequence of the same name). Although the tracked object in the Surfing sequence is a person, LT-FLOtrack performs well, since the person does not move his limbs such that body articulation creates a problem. On the other side of the spectrum is highly articulated Diving (as in VOT2013), and Ball, which, while rigid, contains strong out-of-plane rotation.

In Figure 16 we compare the performance of LT-FLO with other trackers competing in the VOT2014 challenge. As can be seen from the accuracy-robustness plot, the proposed tracker again performs well in terms of accuracy, being placed near the top of the vertical axis. It should again be noted that the benchmark has a very different focus than this paper and it fails to test our contributions: texture-less objects, transparencies and long-term sequences. However, it is useful to show the long-term and texture-less adaptations do not preclude competitive performance in standard sequences.

TABLE IX: Results of the VOT2014 benchmark on LT-FLOtrack. The tabulated values are accuracy, robustness and speed in FPS.

	baseline			region_noise		
Name	**Acc.**	**Rob.**	**Speed**	**Acc.**	**Rob.**	**Speed**
Ball	0.38	4.07	5.01	0.38	4.27	5.09
Basketball	0.52	4.60	2.04	0.48	4.80	2.12
Bicycle	0.58	1.67	5.05	0.57	1.60	4.41
Bolt	0.50	4.40	3.20	0.46	5.07	3.23
Car	0.75	0.87	4.27	0.68	0.87	4.62
David	0.68	0.00	7.60	0.62	0.20	6.51
Diving	0.24	3.27	2.68	0.28	3.53	3.02
Drunk	0.59	0.93	1.77	0.45	0.73	2.07
Fernando	0.32	1.27	1.49	0.32	1.40	1.21
Fish1	0.36	6.80	4.85	0.33	7.20	4.37
Fish2	0.27	6.27	2.61	0.23	6.13	2.73
Gymnastics	0.56	1.07	3.13	0.51	2.20	2.57
Hand1	0.48	3.80	3.70	0.47	4.67	4.77
Hand2	0.40	9.00	3.63	0.35	9.13	4.21
Jogging	0.68	1.13	5.46	0.63	1.07	5.57
Motocross	0.61	1.67	2.43	0.55	1.47	2.56
Polarbear	0.63	0.00	5.19	0.54	0.07	4.77
Skating	0.40	1.73	4.17	0.44	1.60	4.26
Sphere	0.81	0.00	4.48	0.76	0.00	4.50
Sunshade	0.70	1.33	5.28	0.69	1.20	5.45
Surfing	0.80	0.07	9.54	0.72	0.07	9.13
Torus	0.58	1.73	4.78	0.58	2.13	4.12
Trellis	0.62	2.13	5.58	0.60	1.67	5.93
Tunnel	0.60	0.60	2.68	0.55	0.67	2.05
Woman	0.55	5.53	3.57	0.50	5.40	3.72

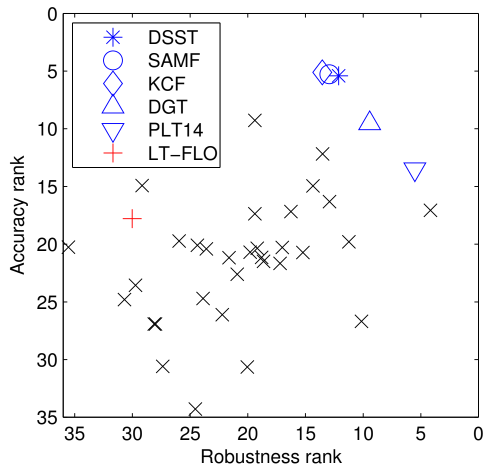


Fig. 16: Accuracy-robustness plot of the VOT Challenge 2014 results. For each tracker, the accuracy and robustness rankings were computed as a mean over the two experiments. Legend shown only for the best five trackers, see [18] for a full report.

F. Visual Tracker Benchmark 1.0 Results

The proposed tracker was additionally evaluated in the context of the Visual Tracker Benchmark 1.0 [31], also known

TABLE X: Results (AUC on OPE test) on the VTB1.0 benchmark [31]. The trackers are sorted according to overall performance.

Tracker	IV	OPR	SV	OCC	DEF	MB	FM	IPR	OV	BC	LR	All
SCM	0.473	0.470	0.518	0.487	0.448	0.298	0.296	0.458	0.361	0.450	0.279	0.499
Struck	0.428	0.432	0.425	0.413	0.393	0.433	0.462	0.444	0.459	0.458	0.372	0.474
LT-FLO	**0.422**	**0.398**	**0.453**	**0.430**	**0.374**	**0.326**	**0.315**	**0.409**	**0.456**	**0.412**	**0.369**	**0.444**
TLD	0.399	0.420	0.421	0.402	0.378	0.404	0.417	0.416	0.457	0.345	0.309	0.437
ASLA	0.429	0.422	0.452	0.376	0.372	0.258	0.247	0.425	0.312	0.408	0.157	0.434
CXT	0.368	0.418	0.389	0.372	0.324	0.369	0.388	0.452	0.427	0.338	0.312	0.426
VTS	0.429	0.425	0.400	0.398	0.368	0.304	0.300	0.416	0.443	0.428	0.168	0.416
VTD	0.420	0.434	0.405	0.403	0.377	0.309	0.302	0.430	0.446	0.425	0.177	0.416
CSK	0.369	0.386	0.350	0.365	0.343	0.305	0.316	0.399	0.349	0.421	0.350	0.398
LSK	0.371	0.400	0.373	0.409	0.377	0.302	0.328	0.411	0.430	0.388	0.235	0.395
DFT	0.383	0.387	0.329	0.381	0.439	0.333	0.320	0.365	0.351	0.407	0.200	0.389
L1APG	0.283	0.360	0.350	0.353	0.311	0.310	0.311	0.391	0.303	0.350	0.381	0.380
MTT	0.305	0.362	0.348	0.342	0.280	0.274	0.333	0.395	0.342	0.337	0.389	0.376
OAB	0.300	0.358	0.370	0.368	0.351	0.324	0.358	0.345	0.414	0.341	0.304	0.370
LOT	0.286	0.364	0.335	0.378	0.345	0.312	0.331	0.355	0.467	0.385	0.189	0.367
MIL	0.311	0.350	0.335	0.335	0.369	0.282	0.326	0.340	0.382	0.373	0.153	0.359
IVT	0.306	0.323	0.344	0.325	0.281	0.197	0.202	0.330	0.274	0.291	0.238	0.358
CPF	0.271	0.372	0.335	0.384	0.371	0.250	0.307	0.341	0.400	0.303	0.151	0.355
TM-V	0.291	0.323	0.320	0.325	0.308	0.362	0.347	0.340	0.429	0.312	0.214	0.352
Frag	0.269	0.332	0.289	0.360	0.366	0.253	0.281	0.300	0.319	0.325	0.149	0.352
RS-V	0.302	0.342	0.314	0.350	0.364	0.305	0.299	0.318	0.368	0.359	0.221	0.346
ORIA	0.321	0.346	0.317	0.327	0.256	0.193	0.221	0.362	0.302	0.275	0.180	0.333
SemiT	0.273	0.303	0.285	0.313	0.337	0.296	0.300	0.297	0.319	0.317	0.330	0.332
KMS	0.333	0.317	0.312	0.333	0.328	0.326	0.321	0.287	0.400	0.321	0.211	0.326
BSBT	0.278	0.319	0.266	0.322	0.295	0.289	0.289	0.313	0.401	0.275	0.229	0.322
PD-V	0.296	0.293	0.263	0.316	0.347	0.276	0.264	0.255	0.231	0.260	0.165	0.308
CT	0.295	0.297	0.302	0.321	0.345	0.269	0.298	0.282	0.359	0.273	0.120	0.306
VR-V	0.212	0.256	0.234	0.273	0.301	0.232	0.214	0.223	0.217	0.264	0.125	0.268
SMS	0.214	0.237	0.259	0.268	0.233	0.221	0.248	0.191	0.298	0.183	0.151	0.229
MS-V	0.225	0.215	0.212	0.200	0.166	0.234	0.230	0.203	0.305	0.212	0.121	0.212

as the CVPR tracker benchmark.[2] This reports results of a different (mostly disjoint with VOT) set of trackers and a more extensive dataset (although with a significant overlap). The dataset contains sequences with longer durations and larger, even full, occlusions. The results can be seen in Table X. Following common practice, we report *area under curve* (AUC) of the success plots on the OPE (one-pass evaluation) test, for all the sequences and broken down by the scene attributes.

The longer sequences with stronger occlusion allow the long-term module of LT-FLO to prove its value. It placed third (*i.e.* in the top 10 %) in the overall leader-board. Unsurprisingly, the results on sequences with occlusions (the OCC column of Table X) are even better. On the other hand, in sequences with out-of-plane rotations (OPR) the performance is not as good.

VIII. Conclusion

In this paper, a novel long-term, edge-based tracking algorithm is presented. A two module approach is used, where the short-term tracker finds the frame-to-frame transformations, using correspondences of lines tangent to edges. This works for textured, low-textured and even transparent objects. The long-term module facilitates learning of the changing appearance of the tracked object, recovery from failures and redetection after full occlusions. Strict learning rules make it very robust to drift. This, in conjunction with a guided redetection framework, renders LT-FLOtrack capable of tracking extremely long sequences robustly.

LT-FLOtrack covers several challenging cases, where traditional trackers often fail. However, it does not attain the highest score on every sequence. As the experiments show, there are several failure cases. These limitations stem from the data not following the assumptions of the method. Typically, performance is degraded for highly non-rigid or articulated objects (*e.g.* in the Panda sequence), as non-rigid objects break the definition of virtual corners (that distant edges are rigidly attached in 3D). LT-FLO can cover these only partially, using its multiple models. The same reasoning also applies to cases of strong out-of-plane rotation. Finally, edge-distorting compression artefacts can cause the edge-correspondence search to fail.

[2]IV: Illumination Variation, OPR: Out-of-Plane Rotation, SV: Scale Variation, OCC: Occlusion, DEF: Deformation, MB: Motion Blur, FM: Fast Motion, IPR: In-Plane Rotation, OV: Out-of-View, BC: Background Clutter, LR: Low Resolution.

Acknowledgment

This work was supported by the EPSRC grant ”Learning to Recognise Dynamic Visual Content from Broadcast Footage” (EP/I011811/1) and The Czech Science Foundation Project GACR P103/12/G084.

References

[1] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[2] J. Bennett, R. Grout, P. Pebay, D. Roe, and D. Thompson. Numerically stable, single-pass, parallel statistics algorithms. In *Proceedings of the International Conference on Cluster Computing*, 2009.

[3] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[4] L. Cehovin, M. Kristan, and A. Leonardis. Robust visual tracking using an adaptive coupled-layer visual model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.
[5] M. Chen, S.K. Pang, T.J. Cham, and A. Goh. Visual tracking with generative template model based on riemannian manifold of covariances. In *Proceedings of the International Colloquium on Information Fusion*, 2011.
[6] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.
[7] H. Grabner, M. Grabner, and H. Bischof. Real-Time Tracking via On-line Boosting. In *Proceedings of the British Machine Vision Conference*, 2006.
[8] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *Proceedings of the European Conference on Computer Vision*, 2008.
[9] S. Hare, A. Saffari, and P.H.S. Torr. Struck: Structured output tracking with kernels. In *Proceedings of the International Conference on Computer Vision*, 2011.
[10] C. Harris and C. Stennett. Rapid – a video rate object tracker. In *Proceedings of the British Machine Vision Conference*, 1990.
[11] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference*, 1988.
[12] R.I. Hartley. Projective reconstruction from line correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994.
[13] M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998.
[14] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003.
[15] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-Learning-Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.
[16] M. Kolsch and M. Turk. Fast 2D hand tracking with flocks of features and multi-cue integration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2004.
[17] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, et al. The visual object tracking VOT2013 challenge results. In *Proceedings of the ICCV workshop on Visual Object Tracking Challenge*, 2013.
[18] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojíř, G. Fernández, et al. The visual object tracking VOT2014 challenge results. In *Proceedings of the ECCV workshop on Visual Object Tracking Challenge*, 2014.
[19] K. Lebeda, S. Hadfield, J. Matas, and R. Bowden. Long-term tracking through failure cases. In *Proceedings of the ICCV workshop on Visual Object Tracking Challenge*, 2013.
[20] K. Lebeda, J. Matas, and R. Bowden. Tracking the untrackable: How to track when your object is featureless. In *Proceedings of the ACCV workshop on Detection and Tracking in Challenging Environments*, 2012.
[21] K. Lebeda, J. Matas, and O. Chum. Fixing the locally optimized RANSAC. In *Proceedings of the British Machine Vision Conference*, 2012.
[22] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, 1981.
[23] J. Matas and T. Vojir. Robustifying the flock of trackers. In *Proceedings of the Computer Vision Winter Workshop*, 2011.
[24] C. F. Olson and D. P. Huttenlocher. Automatic target recognition by matching oriented edge pixels. *IEEE Transactions on Image Processing*, 1997.
[25] S. Oron, A. Bar-Hillel, and S. Avidan. Extended Lucas-Kanade tracking. In *Proceedings of the European Conference on Computer Vision*, 2014.
[26] K.E. Papoutsakis and A.A. Argyros. Integrating tracking with fine object segmentation. *Image and Vision Computing*, 2013.
[27] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 2008.
[28] J. Shotton, A. Blake, and R. Cipolla. Multiscale categorical object recognition using contour fragments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.
[29] P. Smith, I. Reid, and A. Davison. Real-Time Monocular SLAM with Straight Lines. In *Proceedings of the British Machine Vision Conference*, 2006.
[30] Y. Tsin, Y. Genc, Ying Zhu, and V. Ramesh. Learn to track edges. In *Proceedings of the International Conference on Computer Vision*, 2007.
[31] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.
[32] F. Zheng, L. Shao, J. Brownjohn, and V. Racic. Learn++ for robust object tracking. In *Proceedings of the British Machine Vision Conference*, 2014.

Karel Lebeda received his BSc and MSc (both with honours) from Czech Technical University in Prague in 2010 and 2013. Currently, he is pursuing a PhD degree at the Centre for Vision, Speech and Signal Processing at the University of Surrey. His research interests include visual object tracking and 3D computer vision.

Simon Hadfield received his PhD from the University of Surrey in 2013 where he also received an MEng (distinction) in Electronic and Computer Engineering in 2009. He has been awarded the DTI MEng Prize, for the top performance by an MEng graduate, and the Associateship of the University of Surrey (AUS) for his industrial placement with the Home Office Scientific Development Branch. He is a research fellow at the Centre for Vision, Speech and Signal Processing at the University of Surrey. His research interests include motion estimation, scene understanding and multiview geometry.

Jiří Matas is a full professor at the Center for Machine Perception, Czech Technical University in Prague. He holds a PhD degree from the University of Surrey, UK (1995). He has published more than 300 papers in refereed journals and conferences. His publications have approximately 10000 citations in the ISI Thomson-Reuters SCI and about 22000 in Google Scholar. His h-index is 40 (SCI) and 52 (Scholar) respectively. He received the best paper prize at BMVC in 2002 and 2005 and at ACCV in 2007. J. Matas has served in various roles at major international conferences (e.g. ICCV, CVPR, ICPR, NIPS, ECCV), co-chairing ECCV 2004 and CVPR 2007. He is on the editorial board of IJCV and was the Associate Editor-in-Chief of IEEE TPAMI. His research interests include object recognition, image retrieval, tracking, sequential pattern recognition, invariant feature detection, and Hough Transform and RANSAC-type optimization.

Richard Bowden received a BSc and MSc from the Universities of London and Leeds and a PhD from Brunel University which was awarded the Sullivan Doctoral Thesis Prize. He is Professor of computer vision and machine learning at the University of Surrey leading the Cognitive Vision Group within the Centre for Vision, Speech and Signal Processing and was awarded a Royal Society Leverhulme Trust Senior Research Fellowship. His research centres on the use of computer vision to locate, track, and understand humans. He is an associate editor for the journals Image and Vision computing and IEEE TPAMI.