Intra-Frame Object Tracking by Deblatting

Jan Kotera
UTIA, Prague, Czech Republic
kotera@utia.cas.cz

Denys Rozumnyi
CMP, Prague, Czech Republic
rozumden@cmp.felk.cvut.cz

Filip Šroubek
UTIA, Prague, Czech Republic
sroubekf@utia.cas.cz

Jiří Matas
CMP, Prague, Czech Republic
matas@cmp.felk.cvut.cz

Abstract

Objects moving at high speed along complex trajectories often appear in videos, especially videos of sports. Such objects elapse non-negligible distance during exposure time of a single frame and therefore their position in the frame is not well defined. They appear as semi-transparent streaks due to the motion blur and cannot be reliably tracked by standard trackers.

We propose a novel approach called Tracking by Deblatting based on the observation that motion blur is directly related to the intra-frame trajectory of an object. Blur is estimated by solving two intertwined inverse problems, blind deblurring and image matting, which we call deblatting. The trajectory is then estimated by fitting a piecewise quadratic curve, which models physically justifiable trajectories. As a result, tracked objects are precisely localized with higher temporal resolution than by conventional trackers.

The proposed TbD tracker was evaluated on a newly created dataset of videos with ground truth obtained by a high-speed camera using a novel Trajectory-IoU metric that generalizes the traditional Intersection over Union and measures the accuracy of the intra-frame trajectory. The proposed method outperforms baseline both in recall and trajectory accuracy.

1. Introduction

The field of visual object tracking has progressed significantly in recent years. The area encompasses a wide range of problems, including single object model-free short-term tracking where a single target is localized in a video sequence given a single training example [43, 19, 21, 20], long-term tracking methods requiring redetection and learning[16, 29, 28, 38], multi-target multi-camera tracking [31], multi-view methods [22] and methods targeting specific objects such as cars [3], humans [27], or animals [11]. Many variants of the problems have been considered – static or dynamic cameras or environments, RGBD input, use of inertial measurement units to name a few.

This work was supported by Czech Science Foundation grant GA18-05360S and partially also by the Praemium Academiae, awarded by the Czech Academy of Sciences.

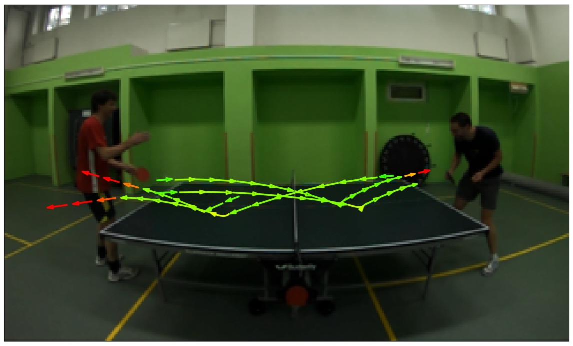
Figure 1. Tracking by Deblatting (TbD) successfully recovers trajectory on a pingpong sequence from the proposed TbD dataset. Color encodes Trajectory Intersection over Union (TIoU) with ground truth trajectories from high-speed camera. Arrows indicate the direction of the motion.

Recently, Rozumnyi *et al.* [32] have shown that the performance of standard state-of-the-art trackers drops significantly when applied to Fast Moving Objects (FMO), apparently due to the effect of blur – such objects appear only as semi-transparent streaks. Examples of applications with FMOs include tracking of balls and ball-like objects in sports videos, particles in scientific experiments, and flying birds and insects.

Standard trackers, both long and short term, provide information about the object location in a frame in the from of a single rectangle. The true, continuous trajectory of the object center is thus sampled with the frequency equal to the video frame rate. For slow moving objects, such sampling is adequate. For fast moving objects, especially if their trajectory is not linear (due to bounces, gravity, friction), a single location estimate per frame cannot represent the true trajectory well, even if the fast moving object is inside the reported bounding box. Moreover, standard trackers typically fail even in achieving that [32].

We propose a novel method for tracking fast-moving, blurred objects. The approach untangles the image formation by solving two inverse problems: *motion deblurring* and *image matting*. We therefore call the method *Tracking by Deblatting*, TbD in short.

The deblatting procedure is inspired by [18] and recovers the trajectory of the object, its shape, and appearance. We introduce a strong prior on the blur kernel and force it to lie on a 1D manifold. The corresponding curve models the object trajectory within a frame. Unlike a standard general tracker, TbD does not need a template of the object, since the representation of the shape and appearance of the object is recovered on the fly. Experiments show that the estimated trajectory is often highly accurate; see Fig. 1.

2. Related work

Object tracking methods are based on diverse principles, such as correlation [4, 9, 10, 25, 37], feature point tracking [39], mean-shift [7, 40], and tracking-by-detection [44, 13]. In addition, several surveys of object tracking have been compiled [1, 2, 12]. Excellent performance in visual object tracking has been shown by discriminative correlation filters [4, 9, 10, 25], yet all the methods fail when the tracked object is blurred as demonstrated in [32].

Methods proposed for object motion deblurring try to estimate sharp images from photos or videos without considering the tracking goal. Early methods worked with a transparency map (the alpha matte) caused by the blur, and assumed linear motion [14, 8] or rotation [34]. Blind deconvolution of the transparency map is better posed, since the latent sharp map is a binary image. Accurate estimation of the transparency map by alpha matting algorithms, such as [23], is necessary and this is not tractable for large blurs. Other methods are based on the observation that autocorrelation increases in the direction of blur [17, 36]. Autocorrelation techniques require a relatively large neighborhood to estimate blur parameters and such methods are not suitable for small moving objects. More recently, deep learning has been applied to motion deblurring of videos [41, 35] and to the generation of intermediate short-exposure frames [15]. The proposed convolutional neural networks are trained only on small blurs; blur parameters are not available as they are not directly estimated. Deblurring of motion-blurred object in a static scene was proposed in [18]. Our core deblatting step extends this method by blind shape estimation and the recovered motion blur is further processed to infer the object intra-frame trajectory.

Tracking methods that consider motion blur has been proposed in [42, 33, 26], yet there is an important distinction between models therein and the FMO problem considered here. The blur is assumed to be caused by camera motion and not by the object motion, which results in blur affecting the whole image and in the absence of alpha blending of the tracked object with the background. Methods for tracking motion-blurred objects exist [30, 24] but assume that the object motion is approximately linear and relatively small compared to the object size, so they ignore blending with background due to blur and their output per frame is a position bounding box instead of an intra-frame trajectory, as in our case.

To our knowledge, the only method that tackles the similar problem of tracking motion-blurred objects remains the work in [32]. The authors assume linear motion and the trajectories are calculated by fitting a line segment to a morphologically thinned difference image between the given frame and the estimated background.

3. Tracking By Deblatting

The proposed method formulates tracking as an inverse problem to the video formation model. Suppose that within a single video frame I an object F moves along the trajectory $\mathcal{C}$ in front of background B. Frame I is then formed as

$$I = H * F + (1 - H * M)B, \tag{1}$$

where $*$ denotes convolution, H is the Point Spread Function (PSF) of the object motion blur corresponding to trajectory $\mathcal{C}$, and M is the binary mask of the object shape (*i.e.* the indicator function of F). We refer to the pair (F, M) as the object model. The first term is the tracked object blurred by its own motion, the second term is the background partially occluded by the object, and the blending coefficients are determined by $H{*}M$. Inference under the assumption of this formation model consists of solving simultaneously two inverse problems: blind deblurring and image matting. The solution is the estimated PSF H and the object model F and M.

Motion blur in (1) is modeled by convolution, which implies the following assumption about the object motion: The object shape and appearance remain constant during the frame exposure time. Scenarios that satisfy the assumption precisely are, *e.g.*, an object of arbitrary shape undergoing only translational motion or a spherical object of uniform color undergoing arbi-

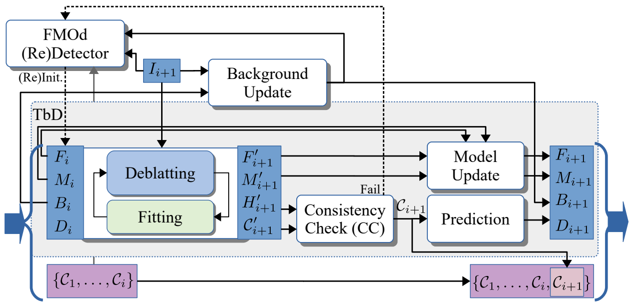
FMOd (Re)Detector
(Re)Init.
I_{i+1}
Background Update
TbD
F_i M_i B_i D_i
Deblatting
Fitting
F'_{i+1} M'_{i+1} H'_{i+1} $\mathcal{C}'_{i+1}$
Consistency Check (CC)
Fail
$\mathcal{C}_{i+1}$
Model Update
Prediction
F_{i+1} M_{i+1} B_{i+1} D_{i+1}
$\{\mathcal{C}_1,\dots,\mathcal{C}_i\}$
$\{\mathcal{C}_1,\dots,\mathcal{C}_i,\mathcal{C}_{i+1}\}$

Figure 2. Long-term Tracking by Deblatting (Sec. 3). The FMO detector is activated during initialization or if the consistency check fails.

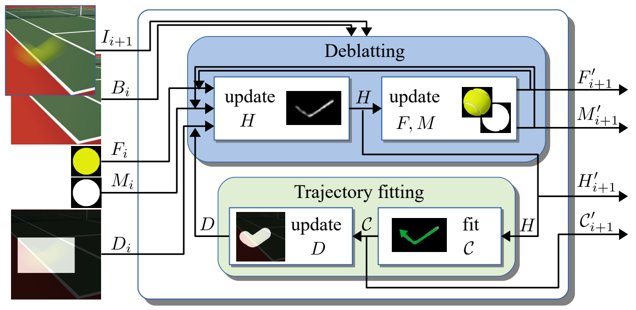

Figure 3. *Deblatting, i.e. deblurring and matting* – Sec. 3.1, with trajectory fitting – Sec. 3.2.

trary motion under spatially-uniform illumination. In addition, the motion must be in a plane parallel to the camera image plane to guarantee constant size of the object. For the purpose of tracking and trajectory estimation we claim that the formation model (1) with convolution is sufficient as long as the assumption holds at least approximately, which is experimentally validated on the presented dataset.

The proposed method is iterative and causal processing of a new frame I_{i+1} using only knowledge acquired from earlier frames $I_1, \dots, I_i$; Fig. 2 (shaded area) provides the overview. Inputs are the current estimates of the object model F_i and M_i, background B_i, and a region of interest D_i in I_{i+1}, which is the neighborhood of the predicted object location. Three main steps are performed in TbD:

1. *Deblatting*: Iteratively solve blind deblurring and matting in the image region D_i with the model (1) and estimate F'_{i+1}, M'_{i+1}, and H_{i+1}; see Sec. 3.1
2. *Trajectory fitting*: Estimate physically plausible motion trajectory (parametric curve) $\mathcal{C}_{i+1}$ corresponding to H_{i+1} and optionally adjust D_i according to $\mathcal{C}_{i+1}$; see Sec. 3.2.
3. *Consistency check & model update*: Verify that the error of the mapping $H \to \mathcal{C}$ is below threshold τ, predict the new region of interest D_{i+1} for the next frame, and update the object model to F_{i+1} and M_{i+1}.

A more detailed illustration of Steps 1 and 2 is in Fig. 3. Step 1 stops after reaching either a given tolerance or a maximum number of iterations. Steps 1 and 2 are repeated only if the newly fitted $\mathcal{C}$ touches the boundary of D – in this case the new D is the $d-$neighborhood of $\mathcal{C}$ where d is the object diameter. Adjusting D this way helps to eliminate the detrimental influence of other moving objects to correct estimation of H.

If the consistency check (CC) passes, we extrapolate the estimated trajectory to the next frame and D_{i+1} is again d-neighborhood of this extrapolation. To update the appearance model we use exponential forgetting

$$F_{i+1} = \gamma F_i + (1-\gamma)F'_{i+1}; \tag{2}$$

M is updated analogically.

To enable long-term tracking, the FMO detector (FMOd) from [32] determines the new input if CC fails. First, FMOd tries detecting the object in an gradually enlarged D. If it succeeds, the main TbD pipeline is reinitialized with D set as a neighborhood of the FMOd detection. If FMOd fails, TbD returns the extrapolation of trajectory $\mathcal{C}_i$ as the best guess of $\mathcal{C}_{i+1}$ and tracking is restarted anew on the next frame. The background B_i is estimated as a temporal median of frames $B_{i-1}, B_{i-2}, \dots$, optionally including video stabilization if necessary. The first detection is also performed automatically by FMOd. The object appearance model is either learned "on the fly" starting trivially with $F_0 \equiv 1$, $M_0 \equiv 1$, or the user provides a template of the tracked object, *e.g.* a rectangular region from one of the frames where the object is still.

3.1. Deblatting

The core step of TbD is the extraction of motion information H from the input frame, which we formulate as a blind deblurring and matting problem. Inputs are the frame I, domain D, background B, and the object appearance model $\hat{F}$. The inverse problem corresponding to (1) is formulated as

$$\begin{aligned} \min_{F,M,H} \frac{1}{2} \|H * F + (1 - H * M)B - I\|_2^2 \\ + \frac{\lambda}{2}\|F - M\hat{F}\|_2^2 + \alpha_F\|\nabla F\|_1 + \alpha_H\|H\|_1 \end{aligned} \tag{3}$$

s.t. $0 \le F \le M \le 1$ and $H \ge 0$ in D, $H \equiv 0$ elsewhere. The primary unknown is H, but F and M are estimated as by-products. The first term in (3) is the fidelity to the model (1). The second λ-weighted term is a form of "template-matching", an agreement with a prescribed appearance. The template $\hat{F}$ is multiplied by M because if $\hat{F}$ is initially supplied by user as a rectangular region from a video frame, it contains the object and partially also the surrounding background. When processing the i-th frame, we set $\hat{F} = F_{i-1}$ as the updated appearance estimate (2) from the previous frame. The first L^1 term is the total variation that promotes smoothness of the recovered object appearance. The second L^1 regularization enforces sparsity of the blur and reduces small nonzero values.

If M is a binary mask then the condition $F \leq M$ states that F cannot be nonzero where M is zero – pixels outside the object must be zero. For computational reasons, we relax the binary restriction and allow M to attain values in the range $[0, 1]$. The correct constraint corresponding to this relaxation is then exactly $F \leq M$, assuming F alone is bounded in $[0, 1]$. The inequality constraint $H \geq 0$ prohibits negative values in H, which are physically implausible for motion blur, and H is estimated only within the domain D.

We solve (3) in an alternating manner, fix (F, M) and solve for H and vice versa, until convergence.

Minimizing (3) w.r.t. H with (F, M) fixed becomes

$$\min_H \frac{1}{2} \|H * F + (1 - H * M)B - I\|_2^2 + \alpha_H \|H\|_1 \quad (4)$$

s.t. $H \geq 0$. We use ADMM (e.g. [5]) to solve (4), which leads to the linear system

$$\begin{aligned}\left((\mathbf{F} - \mathbf{BM})^T(\mathbf{F} - \mathbf{BM}) + \rho\right) H& \\ = (\mathbf{F} - \mathbf{BM})^T(I - B) + \rho(z - u),& \quad (5)\end{aligned}$$

where $\mathbf{F}$ and $\mathbf{M}$ are the convolution operator given by F (*i.e.* convolution with F) and M, respectively. $\mathbf{B}$ is the pixelwise multiplication by background B and z, u, ρ are related to ADMM variable splitting for the non-smooth L^1 term and the inequality constraint; see supplementary for more details.

Minimizing (3) w.r.t. the joint unknown (F, M) with H fixed is

$$\begin{aligned}\min_{F,M} \frac{1}{2} \|H * F + (1 - H * M)B - I\|_2^2& \\ + \frac{\lambda}{2}\|F - M\hat{F}\|_2^2 + \alpha_F \|\nabla F\|_1& \quad (6)\end{aligned}$$

s.t. $0 \leq F \leq M \leq 1$. We again solve this problem using ADMM, which leads to the linear system

$$\begin{bmatrix}\mathbf{H}^T\mathbf{H} + \rho_1\nabla^T\nabla + \lambda + \rho_2 & -\mathbf{H}^T\mathbf{B} - \lambda\hat{F} \\ -\mathbf{H}^T\mathbf{B} - \lambda\hat{F} & -\mathbf{H}^T\mathbf{B}^2\mathbf{H} + \lambda\hat{F}^2 + \rho_2\end{bmatrix}\begin{bmatrix}F \\ M\end{bmatrix}$$
$$= [\mathbf{H},\, -\mathbf{BH}]^T(I - B) + \rho_1\nabla^T(z_1 - u_1) + \rho_2(z_2 - u_2), \quad (7)$$

where $\mathbf{H}$ is the convolution operator given by H, and z_1, u_1, ρ_1 are related to ADMM variable splitting due to the nonsmooth regularization. To enforce the constraint $(F, M) \in C$ where C is a convex set defined by $0 \leq F \leq M \leq 1$, we use the ADMM splitting $z_2 := (F, M)$ and then each ADMM iteration requires projecting z_2 onto C. Note that $C \subset \mathbb{R}^4$ and correspondingly $z_2 \in \mathbb{R}^4$ since each pixel in F has three RGB channels and M is a single-channel mask. Since C is an intersection of half-spaces, we can use iterative Dykstra's projection algorithm [6]. The rest of the minimization is standard; see supplementary for details.

To summarize, the alternating H–(F, M) estimation loop for the i-th frame proceeds as follows:

1. Initialize $M := M^{i-1}$ (if available from previous detection) or $M \equiv 1$; initialize $\hat{F} := F^{i-1}$, $F := M\hat{F}$.
2. Calculate H by solving (4).
3. Check convergence, exit if satisfied.
4. Calculate (F, M) by solving (6), go to 2.

3.2. Trajectory fitting

Fitting the PSF H, which is a gray-scale image, with a trajectory $\mathcal{C}(t) : [0, 1] \to \mathbb{R}^2$ serves three purposes. First, we use the error of the fit in the Consistency Check to determine if H is the motion blur induced by the tracked object and thus whether to proceed with tracking, or to declare the deblatting step a failure and to reinitialize it with different parameters. Second, the trajectory as an analytic curve can be used for motion prediction whereas H cannot. Third, $\mathcal{C}$ defines the intra-frame motion, which is the desired output of the proposed method.

The fitting is analogous to vectorization of raster images. It is formulated as the maximum a posteriori estimation of $\mathcal{C}$, given H, with the physical plausibility of the trajectory used as a prior. Let $\mathcal{C}$ be a curve defined by a set of parameters θ (*e.g.* polynomial coefficients) and $H_\mathcal{C}$ be a raster image of the corresponding $\mathcal{C}$ (*i.e.* blur PSF). We say that the curve $\mathcal{C}$ is the *trajectory fit* of H if θ minimizes

$$\min_\theta \|H_\mathcal{C} - H\| \quad \text{s.t. } \mathcal{C} \in \Psi, \quad (8)$$

where Ψ is the set of admissible curves.

Our main tracking targets are balls and similar free-falling objects, therefore our assumption is that between impulses from other moving objects (*e.g.* players), tracked objects remain in free flight or bounce off static rigid bodies. We then define Ψ as a set of piecewise quadratic continuous curves – quadratic to account for deacceleration due to gravity and piecewise to account for abrupt change of motion during bounces. $\mathcal{C} \in \Psi$ is defined as

$$\mathcal{C}(t) = \begin{cases}\sum_k^2 c_{k,1}t^k & 0 \leq t \leq \tilde{t}, \\ \sum_k^2 c_{k,2}t^k & \tilde{t} \leq t \leq 1,\end{cases} \quad (9)$$

s.t. $\sum_k^2 c_{k,1}\tilde{t}^k = \sum_k^2 c_{k,2}\tilde{t}^k$. Single linear or quadratic curves are included as special cases when $\tilde{t} = 1$. The problem (8) is non-convex and thus a good initial guess is necessary for gradient-descent optimization to perform well. To this end, we employed a four-step procedure: (see Fig. 4 for illustrations)

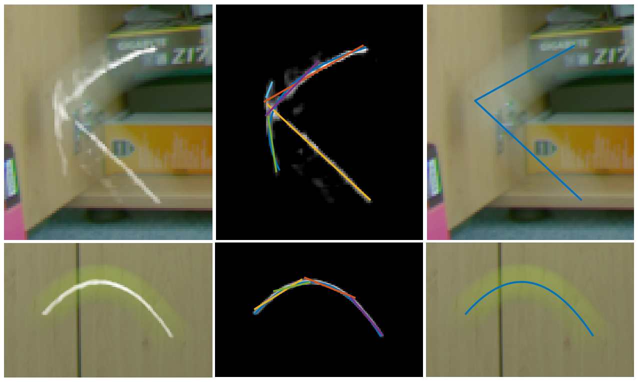

Figure 4. Trajectory fitting. *Left* input image with estimated blur superimposed in white, *middle* linear and parabolic segments found by RANSAC, *right* final fitted trajectory.

1. Identify the most salient linear and quadratic segments in H by RANSAC.
2. Connect segments to form a curve $\mathcal{C}$ of the kind (9).
3. Refine $\mathcal{C}$ to be a locally optimal fit of H in terms of pointwise distance.
4. Calculate the loss (8) and choose the best candidate.

Let us view the blur H as a set of pixels with coordinates x_i and intensities $w_i > 0$. Sequential RANSAC finds line segments as follows: sample two points, find inliers of the corresponding line, find the most salient consecutive run of points on this line and in each round remove the winner from the sampling pool. The saliency is defined as $\sum w_i$ for x_i in the inlier set and "consecutive" means that the distance between neighboring points is bounded by a threshold. The search stops when the saliency drops bellow a specified threshold or there are no more points. We denote the set of collected linear segments as $\mathcal{M}_1$. Parabolic arcs are found similarly. We sample four points, find two corresponding parabolas, project the remaining points on the parabolas to determine the distance and inlier set as well as the arc-length parametrization of inliers (required for correct ordering and mutual distance calculation of inliers) and again find the most salient consecutive run. We denote the set of collected parabolic segments as $\mathcal{M}_2$.

The solution will be close to a curve formed from one or two segments (linear or parabolic) found so far. Let $\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{M}_1$ be two linear segments. If the intersection P of the corresponding lines is close to the segments (w.r.t. some threshold), the curve connecting $\mathcal{C}_1 \to P \to \mathcal{C}_2$ is a candidate for the piecewise linear trajectory fit. This way we construct a set $\mathcal{M}_3$ of all candidate and similarly $\mathcal{M}_4$ with candidates of parabolic pairs.

Curves in $\mathcal{M}_0 = \bigcup \mathcal{M}_i$ are approximate candidates for the final trajectory, yet we first refine them to be locally optimal robust fits to H. We say that a curve $\mathcal{C}$ defined by a set of parameters θ is locally optimal fit to $\{x_i\}$ if θ is the minimizer of the problem

$$\min_{\theta} \sum_{x_i \in K} w_i \, \mathrm{dist}(x_i, \mathcal{C}) + \lambda \int_0^1 \mathrm{dist}(\mathcal{C}(t), \{x_i\}) \mathrm{d}t \quad (10)$$

where $K = \{x_i | \, \mathrm{dist}(x_i, \mathcal{C}) < \rho\}$, $\mathrm{dist}(x, \mathcal{C})$ is the distance of the point x to the curve $\mathcal{C}$ and $\mathrm{dist}(\mathcal{C}(t), \{x_i\})$ is the distance of the curve point $C(t)$ to the set $\{x_i\}$. In the first term, K is a set of inliers defined by the distance threshold ρ and then $\mathcal{C}$ is the distance-optimized fit to inliers. The second term restricts curve length.

The gradient of (10) is intractable since the distance of a point x to a non-convex set (in our case the curve $\mathcal{C}$) is intractable. We therefore resort to a procedure similar to the Iterative Closest Point (ICP) algorithm. In each iteration, we fix the currently closest curve counterpart $y_i = \mathcal{C}(t_i)$ for each point x_i by solving $t_i = \mathrm{argmin}_t \, \mathrm{dist}(x_i, \mathcal{C}(t))$, and in (10) we approximate $\mathrm{dist}(x_i, \mathcal{C}) \approx \|x_i - y_i\|$ and analogically for $\mathrm{dist}(\mathcal{C}(t), \{x\})$. Then (10) becomes a tractable function of θ. We find the solution using the Iteratively Reweighted Least Squares algorithm and proceed with the next iteration of ICP. The algorithms converges in a few iterations and the optimization is fast.

We then refine every curve $\mathcal{C}_0 \in \mathcal{M}_0$ by solving (10) with the ICP-like algorithm and denote the set of solutions as $\mathcal{M}$. Finally, for each curve $\mathcal{C} \in \mathcal{M}$ we construct $H_{\mathcal{C}}$, measure the error $\|H_{\mathcal{C}} - H\|$ and choose the best candidate as the trajectory fit. In TbD, the Consistency Check of the trajectory fit $\mathcal{C}$ is performed by evaluating the criterion $\|H_{\mathcal{C}} - H\| / \|H\| < \tau$. The value of τ was set experimentally on a validation set; high value results in increased number of false positives while low value correspondingly increases the number of false negatives.

4. Experiments

We show the results of Tracking by Deblatting and compare it with other trackers on the task of long-term tracking of motion-blurred objects in real-life video sequences. As a baseline, we chose the FMO detector (FMOd, [32]), specifically proposed for tracking fast moving objects, and the Discriminative Correlation Filter with Channel and Spatial Reliability (CSR-DCF, [25]), which performs well on standard benchmarks such as VOT [19]. CSR-DCF was not designed to track objects undergoing large changes in velocity within a single sequence and would perform poorly in the comparison. We therefore augmented CSR-DCF by FMOd reinitialization every time it outputs the same bounding box in consecutive frames, which is considered a fail. We use FMOd for automatic initialization

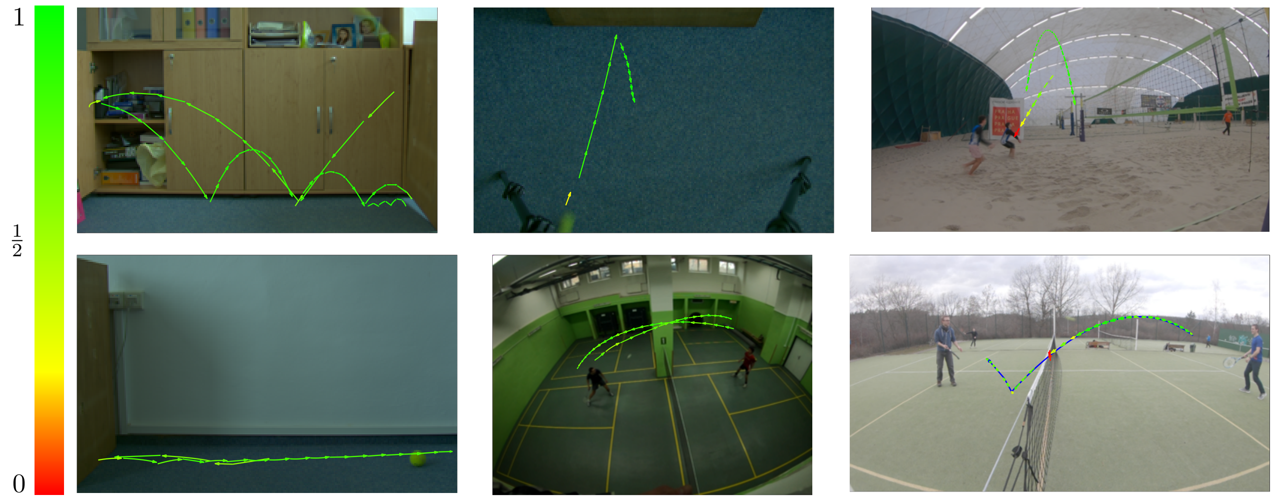

Figure 5. Trajectory recovery for selected sequences from the TbD dataset. Color encodes the Trajectory Intersection over Union (TIoU) with ground truth (scale on the left), arrows indicate the direction of the motion. The trajectory over the whole sequence (bottom right in blue) is obtained by fitting a piecewise quadratic curve to all frames jointly.

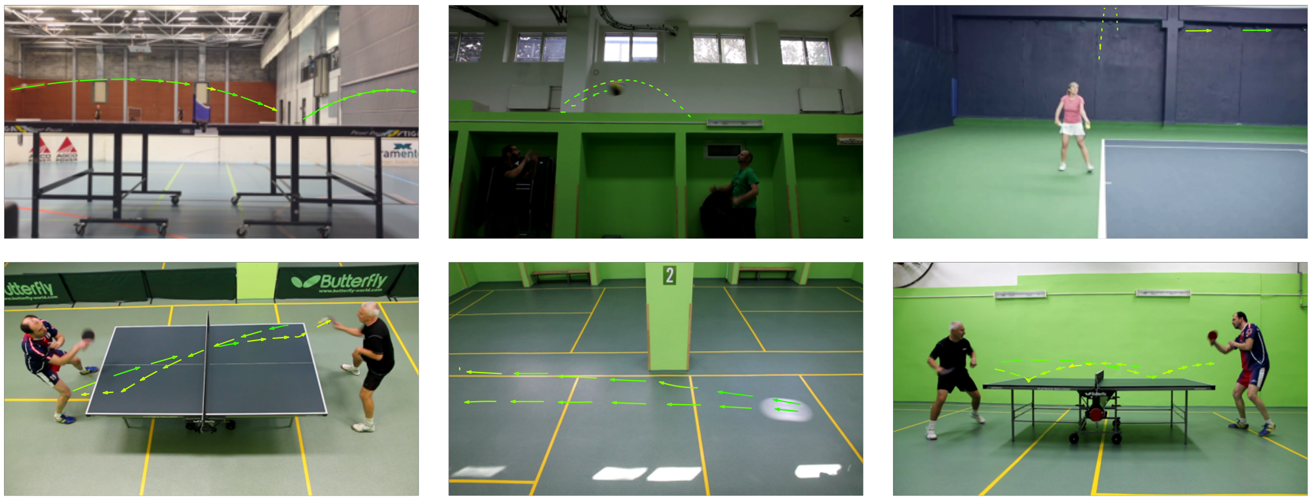

Figure 6. Trajectory recovery for selected sequences from the FMO dataset [32]. Intersection over Union (IoU) with the ground truth occupancy mask is color coded using the scale from Figure 5. Arrows indicate the direction of the motion.

of both TbD and CSR-DCF to avoid manual input and we skip the first two frames of every sequence to establish background B and initialize CSR-DCF. The rest of the sequence is processed causally, B is estimated as a moving median of the past $3-5$ frames.

The goal of TbD is to produce a precise intra-frame motion trajectory, not only a single position per frame in the form of a bounding box. Fig. 4 shows examples of trajectory estimation. The left column is the input image with the estimated PSF superimposed in white and the right column shows the estimated motion trajectory. The efficacy of trajectory fitting is a crucial part of the framework, the estimated blur can contain various artifacts (*e.g.* in the top example due to the ball shadow) and the trajectory fit still recovers the actual motion.

The comparison with baseline methods was conducted on a new dataset consisting of 12 sequences with different objects in motion and setting (different kinds of sports, objects in flight or rolled on the ground, indoor/outdoor). The sequences contain abrupt changes of motion, such as bounces and interactions with players, and a wide range of speeds. The dataset is annotated with the ground-truth trajectory for each frame, obtained from a high-speed camera footage. We compare the method performance in predicting the motion trajectory in each frame. We therefore generalize IoU, the standard measure of position accuracy, to trajectories and define a new measure *Trajectory-IoU* (TIoU):

$$\mathrm{TIoU}(\mathcal{C},\mathcal{C}^*;M^*) = \int_t \mathrm{IoU}\left(M^*_{\mathcal{C}(t)},\, M^*_{\mathcal{C}^*(t)}\right)\mathrm{d}t, \quad (11)$$

where $\mathcal{C}$ is the predicted trajectory, $\mathcal{C}^*$ is the ground-truth trajectory, M^* is a disk mask with true object diameter obtained from the ground truth, and M_x denotes M placed at location x. TIoU can be regarded as the standard IoU averaged over each position on the estimated trajectory. In practice, we discretize the exposure time into evenly spaced timestamps and calculate IoU of the ground-truth location and prediction by the tracker at the timestamps and average these measurements. CSR-DCF tracker only outputs positions, so in this case we estimate linear trajectories from positions in neighboring frames and then calculate TIoU.

The results of the comparison are presented in Table 1. We evaluated three flavors of TbD that differ in the presence of the initial user-supplied template $\hat{F}$ and the learning rate γ of the object model in (2). The presented flavors are:

- TbD-T0,0: Object template not available, model update is instantaneous (memory-less), $\gamma = 0$.
- TbD-T0,0.5: Object template not available, model is updated with the learning rate $\gamma = 0.5$.
- TbD-T1,1: Object template available, model remains constant and equal to the template, $\gamma = 1$.

The TbD outperforms baseline methods on average by a wide margin, both in the traditional recall measure (a detection is called true positive if it overlaps with the ground truth) as well as in trajectory accuracy TIoU. FMOd is less accurate and more prone to false positives as it lacks any prediction step and by design ignores slow objects. CSR-DCF, despite reinitializations by FMOd, fails to detect fast moving objects accurately. Among TbD flavors, it is no surprise that availability of the object template is beneficial and outperforms other versions. However, even if the template is not available, TbD can learn the object model and updating the appearance model gradually during tracking is preferable to instantaneous updates.

To evaluate the performance of the core part of TbD that consists of deblatting and trajectory fitting alone, we provide results of a special version of the proposed method called "TbD with oracle", TbD-O. This behaves like regular TbD but with a perfect trajectory prediction step. We use the ground-truth trajectory to supply the region D to the deblatting step exactly as if it were predicted by the prediction step, effectively bypassing the long-term tracking logic of TbD. The rest is identical to TbD-T1,1. TbD with oracle tests the performance and potential of the deblatting and trajectory estimation alone because failures do not cause long-term damage – success in one frame is independent of success in the previous frame.

The average speed on the dataset was 4 seconds per frame. When the TbD tracker is initialized and frame-to-frame tracking operates smoothly (shaded area in Fig. 2), the average speed is close to 2 seconds per frame. When TbD fails, the necessary re-initialization takes 10-15 seconds. In comparison, the FMO detector [32] needs on average 1.5 seconds per frame. All implementations are written in Matlab.

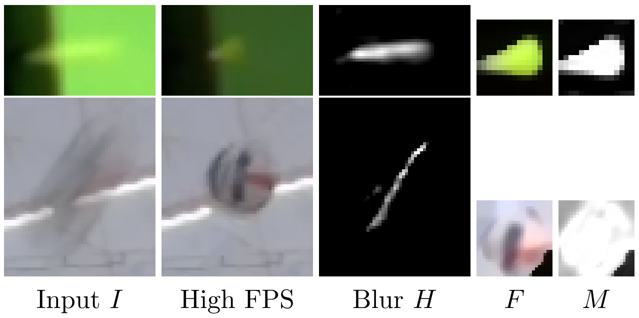

Figure 7. Deblatting examples. From left to right: the input image, corresponding high-speed camera frame, estimated blur H, estimated appearance F and shape M.

Table 2 shows aggregated results for the FMO dataset [32]. This dataset does not contain ground-truth trajectories, we therefore report traditional precision/recall measure, which is derived from the detection and ground-truth bounding-box IoU. On this dataset, the proposed TbD method is only slightly better in recall, owing to the fact that initial detection is done by FMOd and if FMOd fails then TbD cannot start the tracking, but significantly better in terms of precision. Detailed results are in the supplementary.

A visual demonstration of the tracking by the proposed method on the TbD dataset and the FMO dataset is shown in Figs. 5 and 6. Each image shows results of tracking in one sequence from the evaluation dataset superimposed on a single image from the sequence. Arrows depict trajectories detected in a particular frame and the color encodes the corresponding TIoU from green=1 to red=0 (false positive). We can see that the trajectory is estimated successfully with the exception of frames where the object is in direct contact with other moving objects, which throws off the local estimation of background.

Examples of the deblatting alone are in Figs. 7 and 8. Fig. 7 contains from left to right the input frame (crop), corresponding frame from the high-speed camera, estimated motion PSF H, estimated object F and object shape M. In the top row, we see that the shape of the badminton shuttlecock, though not circular, is estimated correctly. In the bottom row, we see that if the non-uniform object undergoes only small rotation, the appearance estimation can also be good. In this case, the shape estimation is difficult due to the mostly homogeneous background similar to the object.

Fig. 8 is another interesting example of the deblat-

Sequence name	#	CSR-DCF [25]		FMO [32]		TbD-T0, 0		TbD-T0, 0.5		TbD-T1, 1		TbD-O
		TIoU	Rcl	TIoU	Rcl	TIoU	Rcl	TIoU	Rcl	TIoU	Rcl	TIoU
badminton_white	40	.275	0.39	.242	0.34	.673	0.92	.674	0.95	.711	0.95	.792
badminton_yellow	57	.047	0.11	.236	0.31	.615	0.89	.623	0.89	.633	0.85	.788
pingpong	58	.060	0.14	.064	0.12	.583	0.89	.587	0.89	.536	0.91	.697
tennis	38	.249	0.83	.596	0.78	.577	0.81	.573	0.81	.633	0.86	.827
volleyball	41	.373	0.69	.537	0.72	.552	0.87	.587	0.90	.741	0.92	.836
throw_floor	40	.262	0.74	.272	0.37	.746	1.00	.768	1.00	.817	1.00	.864
throw_soft	60	.470	0.93	.377	0.57	.585	0.90	.539	0.90	.641	0.95	.707
throw_tennis	45	.347	0.91	.507	0.65	.688	1.00	.781	1.00	.852	1.00	.872
roll_golf	16	.406	1.00	.187	0.71	.414	1.00	.346	1.00	.851	1.00	.898
fall_cube	20	.422	0.89	.408	0.78	.553	0.89	.669	0.89	.704	0.89	.744
hit_tennis	30	.316	0.93	.381	0.68	.564	0.93	.570	0.93	.662	0.93	.828
hit_tennis2	26	.289	0.79	.414	0.71	.459	0.83	.493	0.83	.627	0.83	.738
Average	39	.293	0.70	.352	0.56	.584	0.91	.601	0.92	.701	0.93	.799

Table 1. Trajectory Intersection over Union (TIoU) and Recall (Rcl) on the TbD dataset – comparison of the TbD, CSR-DCF[25] trackers and the Fast Moving Object method [32]. CSR-DCF is a standard, well-performning [20], near-real time tracker. TbD tracker settings: TbD without template and with exponential forgetting factors (2) $\gamma = 0$ (TbD-T0, 0) and $\gamma = 0.5$ (TbD-T0, 0.5), TbD with template and $\gamma = 1$ (TbD-T1, 1), TbD with oracle (TbD-O). The highest TIoU for each sequence is highlighted in blue color and the highest recall in cyan color. TbD-O shows the highest attainable TIoU for TbD as a reference point when predictions are precise. The number of frames is indicated by #.

FMO dataset	FMO [32]		TbD-T0, 0.5	
	Prec.	Recall	Prec.	Recall
Average	59.2	35.5	**81.6**	**41.1**

Table 2. Precision and recall of the TbD tracker (setting: TbD without template and with exponential forgetting factor (2) $\gamma = 0.5$) and the FMO method [32], average on the 16 sequences of the FMO dataset.

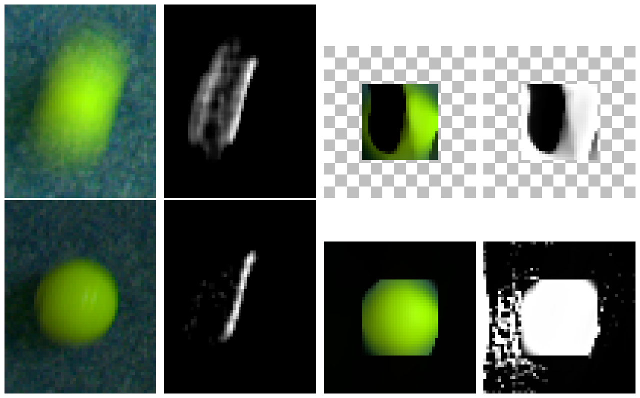

Figure 8. Shadow and blur estimation. Top: the domain of F is set too small and the shadow causes artifacts in H. Bottom: the domain of F is larger, M can compensate for the shadow and the blur H is estimated correctly.

ting behavior. The input frame is in the top left corner and the corresponding part from the high-speed camera is bellow. The object casts significant shadow. If we set the size of F too small, the model cannot cope with the shadow and the estimated blur will contain artifacts in the locations of the shadow as is visible in the top row. If instead we make the support of F sufficiently large, the estimated mask compensates for the shadow and the estimated blur is clean as shown in the bottom row.

5. Conclusion

We proposed a novel approach – Tracking by Deblatting – intended for sequences in which the object of interest undergoes non-negligible motion within a single frame, which needs to be specified by intra-frame trajectory rather than a single position. The method is based on the observation that motion blur is directly related to the motion trajectory of the object. Blur is estimated by a complex method combining blind deblurring, image matting and shape estimation, followed by fitting a piecewise linear or quadratic curve that models physically plausible trajectories. As a result, we can precisely localize the object with higher temporal resolution than by conventional trackers.

The proposed TbD tracker was evaluated on a newly created dataset of videos with ground truth obtained by a high-speed camera using a novel Trajectory-IoU metric that generalizes the traditional Intersection over Union and measures the accuracy of the intra-frame trajectory. The proposed method outperforms baseline techniques both in recall and trajectory accuracy.

Due to the complexity of blind deblurring, the method is currently limited to objects that do not significantly change their perceived shape and appearance within a single frame, the method works best for approximately round and uniform objects.

References

[1] S. Avidan. Ensemble tracking. *IEEE TPAMI*, 29(2):261–271, Feb. 2007.

[2] B. Babenko et al. Robust object tracking with online multiple instance learning. *IEEE TPAMI*, 33(8):1619–1632, Aug 2011.

[3] M. Betke et al. Real-time multiple vehicle detection and tracking from a moving vehicle. *Machine Vision and Applications*, 12(2):69–83, 2000.

[4] T. A. Biresaw et al. Correlation-based self-correcting tracking. *Neurocomput.*, 152(C):345–358, Mar. 2015.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, Jan. 2011.

[6] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in Order Restricted Statistical Inference*, pages 28–47, New York, NY, 1986. Springer New York.

[7] D. Comaniciu et al. Kernel-based object tracking. *IEEE TPAMI*, 25(5):564–575, May 2003.

[8] S. Dai and Y. Wu. Motion from blur. In *IEEE CVPR*, pages 1 –8, June 2008.

[9] M. Danelljan et al. Accurate scale estimation for robust visual tracking. In *BMVC*. BMVA Press, 2014.

[10] M. Danelljan et al. Learning spatially regularized correlation filters for visual tracking. In *IEEE ICCV*, pages 4310–4318, 2015.

[11] S. Fry et al. Tracking of flying insects using pan-tilt cameras. *Journal of Neuroscience Methods*, 101(1):59–67, 2000.

[12] M. Godec et al. Hough-based tracking of non-rigid objects. *CVIU*, 117(10):1245–1256, Oct. 2013.

[13] S. Hare et al. Struck: Structured output tracking with kernels. *IEEE TPAMI*, 38(10):2096–2109, Oct 2016.

[14] J. Jia. Single image motion deblurring using transparency. In *IEEE CVPR*, pages 1–8, 2007.

[15] M. Jin et al. Learning to extract a video sequence from a single motion-blurred image. In *IEEE CVPR*, pages 6334–6342, June 2018.

[16] Z. Kalal et al. Tracking-learning-detection. *IEEE TPAMI*, 34(7):1409–1422, 2012.

[17] T. H. Kim and K. M. Lee. Segmentation-free dynamic scene deblurring. In *IEEE CVPR*, pages 2766–2773, 2014.

[18] J. Kotera and F. Šroubek. Motion estimation and deblurring of fast moving objects. In *IEEE International Conference on Image Processing (ICIP)*, pages 2860–2864, Oct 2018.

[19] M. Kristan et al. *The Visual Object Tracking VOT2016 Challenge Results*, pages 777–823. Springer International Publishing, Cham, 2016.

[20] M. Kristan et al. The sixth visual object tracking vot2018 challenge results. In L. Leal-Taixé and S. Roth, editors, *ECCV 2018 Workshops*, pages 3–53, Cham, 2019. Springer International Publishing.

[21] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2137–2155, Nov 2016.

[22] T. Kroeger et al. Multi-view tracking of multiple targets with dynamic cameras. In *GCPR*, pages 653–665, 2014.

[23] A. Levin et al. A closed-form solution to natural image matting. *IEEE TPAMI*, 30(2):228–242, Feb. 2008.

[24] X. Lingyun, H. Luo, H. Bin, and Z. Chang. Real-time robust tracking for motion blur and fast motion via correlation filters. *Sensors*, 16:1443, 09 2016.

[25] A. Lukezic et al. Discriminative correlation filter with channel and spatial reliability. In *IEEE CVPR*, 2017.

[26] B. Ma et al. Visual tracking under motion blur. *IEEE TIP*, 25(12):5867–5876, Dec. 2016.

[27] A. Mittal and L. S. Davis. M 2 tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. *IJCV*, 51(3):189–203, 2003.

[28] A. Moudgil and V. Gandhi. Long-term visual object tracking benchmark. *arXiv preprint arXiv:1712.01358*, 2017.

[29] M. Mueller et al. A benchmark and simulator for uav tracking. In *ECCV*, pages 445–461, 2016.

[30] Y. Park, V. Lepetit, and W. Woo. Esm-blur: Handling rendering blur in 3d tracking and augmentation. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 163–166, Oct 2009.

[31] E. Ristani et al. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV 2016 Workshops*, pages 17–35, 2016.

[32] D. Rozumnyi et al. The world of fast moving objects. In *IEEE CVPR*, pages 4838–4846, July 2017.

[33] C. Seibold et al. Model-based motion blur estimation for the improvement of motion tracking. *CVIU*, 160:45–56, 2017.

[34] Q. Shan et al. Rotational motion deblurring of a rigid object from a single image. In *IEEE ICCV*, pages 1–8, Oct. 2007.

[35] S. Su et al. Deep video deblurring for hand-held cameras. In *IEEE CVPR*, pages 237–246, July 2017.

[36] J. Sun et al. Learning a convolutional neural network for non-uniform motion blur removal. In *IEEE CVPR*, pages 769–777, 2015.

[37] M. Tang et al. High-speed tracking with multi-kernel correlation filters. In *IEEE CVPR*, pages 4874–4883, June 2018.

[38] R. Tao et al. Tracking for half an hour. *arXiv preprint arXiv:1711.10217*, 2017.

[39] C. Tomasi and T. Kanade. *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh, 1991.

[40] T. Vojir et al. *Robust Scale-Adaptive Mean-Shift for Tracking*, pages 652–663. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[41] P. Wieschollek et al. Learning blind motion deblurring. In *IEEE ICCV*, pages 231–240, Oct 2017.

[42] Y. Wu et al. Blurred target tracking by blur-driven tracker. In *IEEE ICCV*, pages 1100–1107, Nov. 2011.

[43] Y. Wu et al. Online object tracking: A benchmark. In *IEEE CVPR*, 2013.

[44] J. Zhang et al. *MEEM: Robust Tracking via Multiple Experts Using Entropy Minimization*, pages 188–203. Springer International Publishing, Cham, 2014.