This CVPR workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

EgoTracker: Pedestrian Tracking with Re-identification in Egocentric Videos

Jyoti Nigam, Renu M. Rameshan
Indian Institute of Technology, Mandi
Mandi, HP-175001
jyoti_nigam@students.iitmandi.ac.in,renumr@iitmandi.ac.in

Abstract

We propose and analyze a novel framework for tracking a pedestrian in egocentric videos, which is needed for analyzing social gatherings recorded with a wearable camera. The constant camera and pedestrian movement makes this a challenging problem. The main challenges are natural head movement of wearer and target loss and reappearance in a later frame, due to frequent changes in field of view. By using the optical flow information specific to egocentric videos and also by modifying the learning process and sampling region of trackers which tracks by learning an SVM online, we show that re-identification is possible. The specific trackers chosen are STRUCK and MEEM.

1. Introduction

Use of wearable cameras is on the rise among civilians as well as law-enforcement personnel. These videos are captured from the first person view point. Examples of wearable devices are GoPro camera and GoogleGlass which can be worn on head or shoulder. The first person viewpoint provides a hands-free mode for recording indoor [5] as well as outdoor activities and allows capturing of wearer's social interactions as well as interactions of others. Here social interactions refer to any activity performed by one or more individuals, such as people forming a group, standing together, moving alone etc. In this paper we specifically focus on videos of social gathering captured by wearable cameras.

The automatic analysis of egocentric videos captured from a first person perspective provides several insights about group as well as individual behavior. As the first step towards this analysis one needs to detect and identify activities at individual and group levels in a video. The patterns of pedestrian (agent) movements play an important role in predicting the type of group activities. To model and compute these patterns, tracking of pedestrian targets with re-identification is required. Re-identification is important owing to wearer as well as target motion which results in frequent target loss and re-entry. In order to track an agent throughout the length of the video, the tracker should be able to identify the agent once re-entry occurs after target loss. Though the problem of re-identification looks trivial when there is only one pedestrian in the scene, it may be noted that a simple pedestrian detection method will not solve the tracking problem due to the following reasons. The performance of the tracker depends on the performance of the detector. Typical scenarios for this are (1) mis-detection, (2) false detection and (3) multiple detections. Our proposed algorithm identifies the re-entry of the person who is being tracked.

Though egocentric vision has the advantage of camera being close to individuals while capturing social interactions [4], it poses new challenges like (1) the camera is under constant motion and this unstructured camera motion leads to shaky and unstabilized videos, (2) due to closeness of target, even small rotation of camera can make the target leave the field of view, or change the appearance, (3) camera and target, both are in motion, which increases the frequency of target loss and re-entry. In addition to these, challenges like variation in illumination, viewing angle and scale get amplified here owing to the close proximity of camera and target. Note that most of these challenges do not pose a problem in a stationary camera capturing a social gathering.

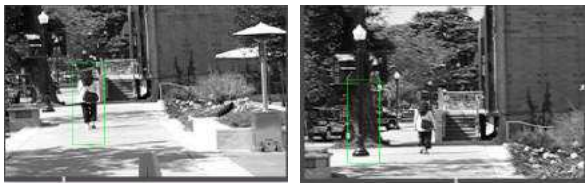
In literature, we could not find a general framework for tracking of pedestrians with re-identification in egocentric videos. The works which are closest to the problem addressed here are by Choi *et al.* [3] and Alletto *et al.* [2]. Choi *et al.* [3] uses a stereo camera setup and tracks multiple people using depth information from stereo, whereas we are looking for a single camera solution for tracking with re-identification. In [2], Alletto *et al.* proposes a novel approach based on odometry and 3D localization which is implemented on a wearable board. Although this work overcomes many issues typical to egocentric vision, it requires additional hardware whereas our algorithm does not have any such requirements.

Recently, several papers have used deep learning approaches to tackle the problem of person re-identification.

In [1], deep convolution architecture has been proposed that takes a pair of input images and outputs the similarity score. The network has been designed to learn features and corresponding similarity metric simultaneously. In [9], a recurrent supervised CNN has been proposed for tracking by detection that takes into account of previous locations along discriminative deep learned features.

It has been observed that the tracking algorithms designed for conventional videos [13] failed when used in first person perspective videos. As an example, Lukas-Kanade [8] failed, because the assumptions of brightness constancy, temporal persistence and spatial coherence fail due to discontinuity between successive frames. This discontinuity occurs due to rapid changes in the field of view of camera.

State-of-the-art methods like KCF [7], MEEM [14] and STRUCK [6] were also found to fail in tracking/re-identification. KCF [7] fails since it confuses between human and any upright structure in an image, as shown in Fig. 1. Both MEEM [14] and STRUCK [6] exploit an online SVM for tracking fail in re-identification because of the learning procedure. STRUCK and MEEM have an underlying similarity, hence it is possible to modify both for re-identification.



(a) (b)

Figure 1. Comparison of KCF and STRUCK. (a) The target is tracked by STRUCK in the presence of fast camera motion, whereas in (b) the bounding box has moved away from target on to the tree in case of KCF.

Both STRUCK and MEEM treat the tracking problem as a classification task and use online learning techniques to update the object model. Both the algorithms use the same sampling process and maintain a set of positive and negative support vectors which is shown in Fig. 2(a). Both the sets are updated in every frame. This non-discriminative updation leads to change in target model in the event of target loss. This can be seen in Fig. 2(b). The new target model contains support vectors corresponding to background, since both algorithms pick that patch which has the highest classification accuracy. It may be noted that this whole process leads to an initial fall in score value, which rises with subsequent frames and reaches a new high as the model changes. This can be inferred from Fig. 2(c).

The above described update procedure leads to a deviation of the target model from the ground truth model. By the time target re-enters, if the model drift is large, the target cannot be re-identified. By a simple modification to the algorithm using optical flow information specific to egocentric case and by using classification score values, we observe that our method is able to outperform state-of-the-art trackers on various egocentric videos.

2. Proposed System

We propose a scale invariant tracker for egocentric videos which is capable of doing target re-identification and is faster than the baseline trackers, STRUCK and MEEM. Since STRUCK and MEEM have similar structure for learning the SVMs we use them as the baseline trackers. Re-identification is done by controlling the learner using classification score values. The scale invariant tracking is achieved by exploiting the optical flow values. A faster tracking is developed by redefining the sampling region.

2.1. Re-identification

A target can be re-identified while tracking, by modifying the update process of the learner. The classification score value is used to decide whether the target is present or absent in the next frame. STRUCK as well as MEEM uses a classification score (ρ) to update the support vectors. This is defined as follows. Let P be the set of positive support vectors, S be the set of sampled patches and θ, the threshold. The classification score is defined as

$$\rho = \text{similarity}(s, p), \text{ where } s \in (S) \; and \; p \in (P) \quad (1)$$

The score going below a particular threshold indicates the absence of the target. At this point the learner is stopped, which also pauses the support vector update mechanism preventing irrelevant change in the target model. This is demonstrated in Fig. 2(e). At the re-entry of target the score goes above this threshold because the set of positive support vectors corresponds to the target. Hence, the target gets re-identified at re-entry to the field of view, which is shown in Fig. 2(f). At this point the update procedure restarts the learner.

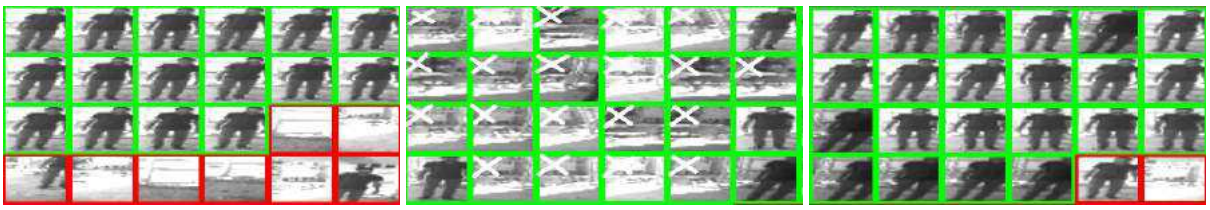
$$\text{Status of learner} = \begin{cases} \text{Continues (target is present)} & \rho > \theta \\ \text{Stops (target is absent)} & else \end{cases} \quad (2)$$

The results shown in Fig. 2 correspond to the baseline tracker STRUCK, similar results are achieved for MEEM as well which are not shown here because a visual representation of support vectors is not available for MEEM. Steps $9-10$ of Algorithm 1 take care of re-identification. In addition to this, the modifications in sampling process generate less number of support vectors which also aids in the re-identification task. This is explained in Section 3.3.

(a) Initial target and set of support vectors (b) Target lost : Learning introduces background in target model (c) Re-identification is failed

(d) Initial target and set of support vectors (e) Target lost : Learning is stopped, no update in target model (f) Target is re-identified

Figure 2. Upper row: shows the working of STRUCK, lower row: shows the working of EgoTracker



(a) Positive support vectors before target lost (b) STRUCK : Positive support vectors at the end (c) EgoTracker : Positive support vectors at the end

Figure 3. Comparison of STRUCK and EgoTracker with respect to positive support vectors. (a) Green boxes show positive support vectors and Red boxes show negative support vectors (best viewed in color). (b) Limitation of STRUCK in the case of target loss: less positive support vectors corresponding to target and background is considered as changed target (shown by crosses on green boxes). (c) After modifying the update rule, in EgoTracker the set of positive support vectors contains only target features in the case of target loss.

The effect of this modification on the support vectors is shown in Fig. 3. Fig. 3(a) shows the set of support vectors before target is lost and it may be observed that the positive support vectors correspond to the target. Fig. 3(b) shows the same set from STRUCK in the absence of target for 85 frames. All those marked by a cross indicate background elements and show the change in model. Fig. 3(c) shows the same set for EgoTracker in the absence of target for the same 85 frames. It is observed that EgoTracker preserves the model, whereas STRUCK completely redefines the model, leading to failure in re-identification in STRUCK as shown in Fig. 2(c).

2.2. Scale invariance

As mentioned in Section 1, scale variations are more prominent in egocentric videos compared to conventional videos. To ensure continuation of tracking in the presence of scale variations, we use the optical flow values. The change in target size in image plane can be identified by using ratio of optical flow values obtained from two consecutive pairs of frames. Scale changes of more than 50% are not tracked.

To identify target motion towards or away from the camera we use optical flow, if optical flow increases it shows that the target is moving towards the camera and vice-versa. Note that optical flow is calculated only within the bounding box. The optical flow values from two consecutive pairs of frames (f_t, f_{t-1}) and (f_{t-1}, f_{t-2}), where f_t is frame at time t, are used.

To give a simplified demonstration of the idea, we give an analysis considering movement along Z direction away from camera. There is no movement along X, Y and no rotation. Under these assumptions, the horizontal and vertical components (u^t, v^t) at time t of optical flow can be written as :

Let $(X, Y, Z)^T$ be the $3D$ point with positive Z direction away from camera. The origin is same as camera center. $(V_X, V_Y, V_Z)^T$ be the velocity vector in $3D$ and f, the focal length of camera. We use u to denote horizontal component of optical flow and v to denote vertical component.

$$u = \frac{f}{Z} V_X - \frac{x}{Z} V_Z \tag{3}$$

Since, there is no motion along X and Y axes, the veloc-

ity component along X axis, V_X is 0 and from projection equation x is $\frac{fX}{Z}$. So now u can be written as:

$$u = -\frac{x}{Z}V_Z \tag{4}$$

Using u^{t-1} for horizontal velocity for the frame pair (f_{t-1}, f_{t-2})

$$u^{t-1} = -\frac{x}{Z}V_Z \tag{5}$$

Assuming that the object is moving with constant velocity V_Z, u^t becomes

$$u^t = -\frac{x}{Z+\delta Z}V_Z \tag{6}$$

where δZ is the distance moved between f_t and f_{t-1}. Similarly the vertical component of optical flow becomes

$$v^{t-1} = -\frac{y}{Z}V_Z, \qquad v^t = -\frac{y}{Z+\delta Z}V_Z \tag{7}$$

The ratios of horizontal and vertical components (o_h, o_v) of optical flow are

$$o_h = \frac{u^t}{u^{t-1}} = \frac{1}{1+\frac{\delta Z}{Z}}, \tag{8}$$

$$o_v = \frac{v^t}{v^{t-1}} = \frac{1}{1+\frac{\delta Z}{Z}}. \tag{9}$$

Assuming that $\delta Z < Z$, o_h and o_v can be approximated as

$$o_h \approxeq 1 - \frac{\delta Z}{Z}, o_v \approxeq 1 - \frac{\delta Z}{Z}. \tag{10}$$

Hence, o_h and o_v are both less than one for a target moving away from camera. The values o_h and o_v are indicators to the movement of subject away or towards the camera. From (8) and (9) it is observed that for a subject moving away $o_h < 1, o_v < 1$.

In the actual scenario there is motion along X and Y also. Table 1 gives possible o_h and o_v values for different types of movements in X, Y without considering rotation. An entry of 1 indicates as increase in the value of the corresponding optical flow and 0 indicates no change. Case 1 need not to be considered in a tracking problem. In Case 2 and Case 3 there is no scale change, so bounding box size need not to be changed. Case 3 is shown in Fig. 4 where subject is moving in X direction alone and it can be noted that there is no change in the size of subject. In Case 4 since both are changing, the bounding box size needs to be changed. We look at the ratio $\gamma = \frac{o_h}{o_v}$. If $\gamma \in [0.2,\ 1.5]$, which is attained empirically, we change the bounding box size by an amount determined by average value of u^t which is denoted by u^t_{avg} along X direction and average value of v^t which is denoted by v^t_{avg} along Y direction. This procedure is explained in steps $3-8$ of Algorithm 1.

Table 1. Horizontal and vertical components of optical flow (o_h, o_v) to show relative movement between subject and camera

Case	o_h	o_v	Relative motion
1	0	0	NIL
2	0	1	only in Y direction
3	1	0	only in X direction
4	1	1	in both directions

Figure 4. Motion of target along X axis alone

2.3. Proposed sampling

Egocentric videos give additional information about relative wearer-target motion in the form of optical flow. It may be noted that over a period of time the wearer's field of view reaches back to the previous place in the case where the wearer is walking with natural head movements (left and right). In contrast to conventional videos, this left and right movement of camera exhibits a particular pattern in the optical flow. We leverage this information for deciding the location of bounding box in subsequent frames, *i.e.* the sampling process is modified and the sampling region $(\overline{S})$ in pixels is defined as follows:

$$\overline{S} = \{(x, y) \mid -5 \leq y \leq 5,\ \ l \leq x \leq h\} \tag{11}$$

It may be observed that while capturing egocentric videos, the wearer can move his/her head more often in horizontal direction rather than vertical direction. Hence, the vertical movement of bounding box is restricted to $\pm$ 5 pixels throughout the tracking process. When the target is lost, learning stops but search continues with l and h. The limits for l and h are decided using the average value of horizontal component of optical flow (u^t_{avg}). Here the sparse optical flow [12] for entire frame is used. The sampling procedure proceeds in this manner and similarity measure is computed for each candidate sample to predict the target presence/absence. The l and h values in (11) are fixed as

$$l = -r,\ \ h = 0 \qquad u^t_{avg} > 0 \tag{12}$$

$$l = 0, \qquad h = r \qquad u^t_{avg} < 0 \tag{13}$$

Here the assumption is that the target moved out of the frame from right. The limits are reversed for the opposite event.

This modification reduces the search space of target and generates less number of support vectors which in turn reduces the time required for calculating the similarity. It may be noted that the choice of $(\overline{S})$, aids in re-identification

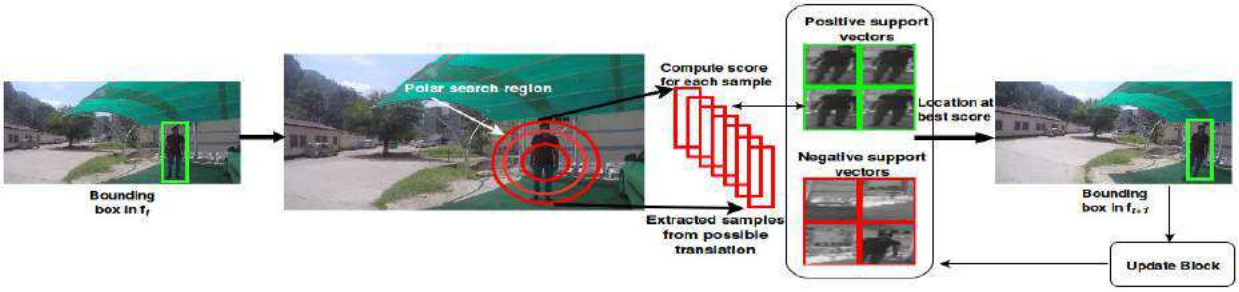


Figure 5. Block diagram of baseline trackers

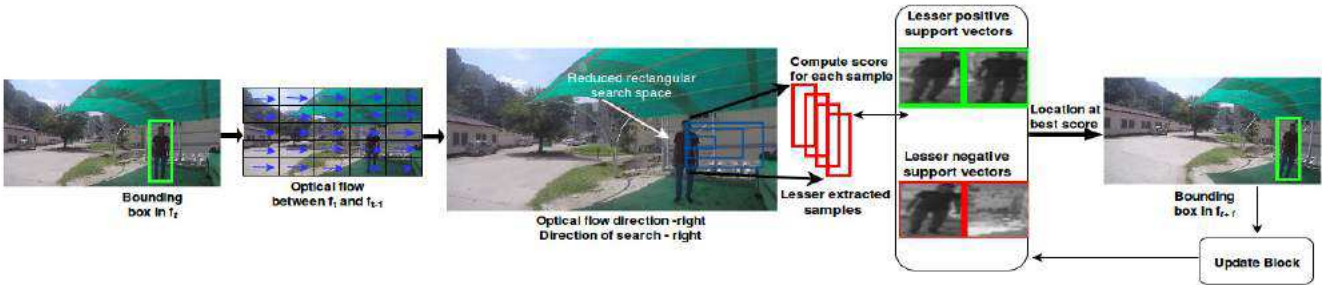


Figure 6. Block diagram of EgoTracker

which is discussed in Section 3.1. The detailed pictorial representation of baseline trackers and EgoTracker is shown in Fig. 5 and Fig. 6, where for better visualization we show the sampling region for the case where target is present. In implementation search space reduction happens when target is absent. The above procedure is explained in steps $9-18$ of Algorithm 1.

3. Experimental Analysis

In this section, we give details for the implementation of our proposed approach. We use the publicly available C++ code of STRUCK [6] and MATLAB code of MEEM [14] with the proposed changes in order to apply it for pedestrian tracking in egocentric videos. These egocentric videos are taken from HUJI dataset [11], [10] and are limited to short segments containing 300 to 900 frames. Apart from this dataset, we have generated our own data using GOPRO Black action camera, with eight different videos to test the addressed problems. The sample frames from the captured videos are shown in Fig. 7. We have named the videos Exp1 to Exp8 corresponding to the wearer/subject movement scenarios as given in Table 2.

In this work, we are dealing with videos containing a **single** pedestrian with possible scale variations and to extend our experimentation, we track a single pedestrian in the presence of multiple pedestrians. In the case of multiple pedestrians, at the target loss the target model will not be updated even in the presence of other pedestrians. This discriminative updating rule enables our algorithm to re-identify the target, it can be seen in Fig. 8.

The case of missed detection may occur due to partial occlusion or any heavy distortion in the target and in this

Algorithm 1 EgoTracker

Require:
(i). Sequence of raw images, f_t is frame at time t.
(ii). BB is bounding box at f_t.
(iii). OF is optical flow of f_t and f_{t-1} containing (u^t, v^t), $u^t_{avg\_BB}$ and $v^t_{avg\_BB}$ are average optical flow values within bounding box.
(iv). $bestscore$ is the classification score of best candidate.
(v). u^t_{avg} is average optical flow value of entire frame.
(vi). r is the search radius.

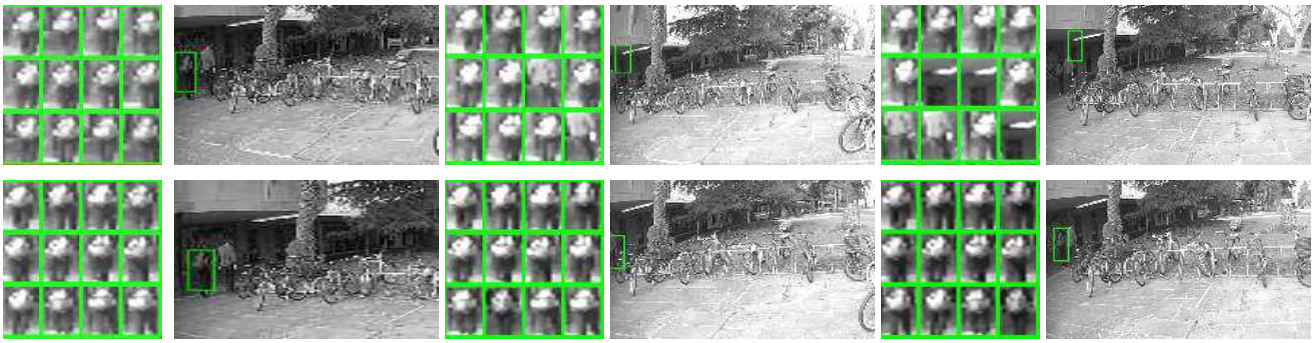
Ensure:
Tracking of a pedestrian, and maintaining the track in the event of target loss and re-entry along with limited amount of scale variations.

1: Given f_t, f_{t-1}, BB in f_{t-1} and threshold θ, estimate BB in f_t;
2: Start baseline tracker STRUCK with vertical search radius ± 5 pixels;
3: Compute OF within BB of f_t, f_{t-1} and f_{t-1}, f_{t-2}.
4: **if** $(\frac{u^t}{u^{t-1}} < 1 \;\&\; \frac{v^t}{v^{t-1}} < 1)$ **then**
5: Reduce the size of BB by average values of $u^t_{avg\_BB}$ and $v^t_{avg\_BB}$
6: **else**
7: Increase the size of BB by average values of $u^t_{avg\_BB}$ and $v^t_{avg\_BB}$
8: **end if**
9: **if** $(bestscore < \theta)$ **then**
10: Stop learner and compute block wise OF between f_{t-1}, f_t;
11: **if** $(u^t_{avg} > 0)$ **then**
12: Restrict the sampler to $-r$ to 0 pixels horizontally;
13: Continued until $(bestscore > \theta)$;
14: **else if** $(u^t_{avg} < 0)$ **then**
15: Restrict the sampler to 0 to r pixels horizontally;
16: Continued until $(bestscore > \theta)$;
17: **end if**
18: **end if**

Figure 7. Images from the generated dataset (a) [Top row]Exp1, (b)[middle row] Exp2 and (c)[bottom row] Exp6

Table 2. Dataset generated to address specific problems

Video Name	Wearer/Subject Motion	Addressed Problem
Exp1	Both wearer and subject are stationary and wearer is moving head left to right and reverse	Re-identification
Exp2	Only wearer is moving towards stationary subject with head rotation left to right and reverse	Re-identification
Exp3	Only subject is moving towards stationary wearer and wearer is moving head left to right and reverse	Re-identification
Exp4	Both wearer and subject are moving towards each other on straight path and wearer is moving head left to right and reverse	Re-identification
Exp5	Both wearer and subject are moving towards each other on a zig zag path and wearer is moving head left to right and reverse	Re-identification
Exp6	Only subject is moving towards stationary wearer	Scale variation
Exp7	Only subject is moving away from stationary wearer	Scale variation
Exp8	Only subject is moving but not towards stationary wearer, its movement is along horizontal direction only	Scale variation



(a) Frame - 70 (b) Frame - 100 (c) Frame - 150

Figure 8. Upper row: shows the working of STRUCK, lower row: shows the working of EgoTracker. To present a generalized version of our proposed method, we track a single pedestrian in the presence of one more pedestrian (video segment is taken from Huji dataset [11]). (a) Set of positive support vectors before target loss. (b) Shows update in target model at the time of target loss, in EgoTracker it is not updating while in STRUCK it is continuously updating. (c) As the target re-enters the field of view, it is identified and tracked again in the case of EgoTracker and re-identification is failed in the case of STRUCK.

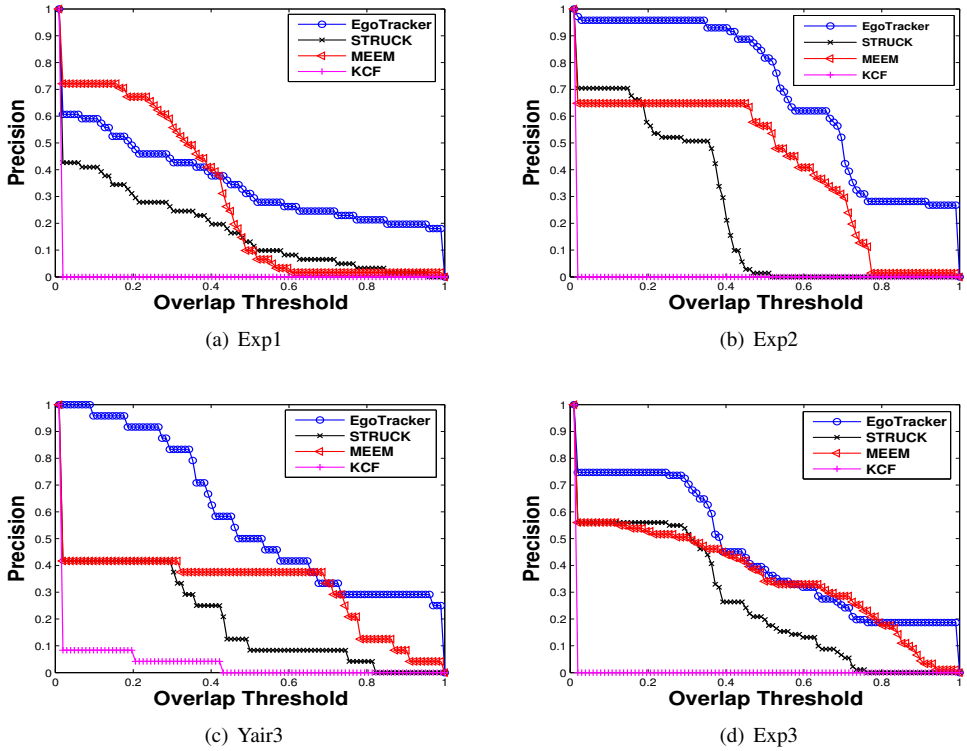
case the classification score will go below the threshold. This reduction in score value pauses the support vectors update mechanism thereby preventing patches from occluded or heavily distorted target from being included in the set of positive support vectors. Thus, the target model will only contain relevant support vectors. The running time is estimated on a laptop of configuration - 8 GB RAM, quad core intel i5 processor and 2.20 GHz speed. The performance metrics used for analysis of the proposed system are re-identification, scale invariance, computation time along with score analysis.

3.1. Performance Analysis

In this section we evaluate the performance of EgoTracker for twelve different video segments.

To assess tracking performance, we use the Pascal VOC overlap criterion [6]. Fig. 9 shows precision plots for four videos. These plots show the percentage of frames for which the overlap between the ground truth and tracker bounding boxes is greater than a particular threshold. We can see from these plots that the precision curves for EgoTracker are better than others in almost all the cases.

Fig. 10(a) gives a comparison between the score values



(a) Exp1 (b) Exp2

(c) Yair3 (d) Exp3

Figure 9. Comparative precision plots for EgoTracker, MEEM, KCF and STRUCK. These plots show the percentage of frames for which the overlap between the ground truth bounding box and tracker bounding box is greater than a particular threshold.

of STRUCK and EgoTracker. For STRUCK the score falls at target loss and reaches a new high corresponding to a wrong target identified by the new positive support vectors. With the proposed modification this does not happen in EgoTracker. Hence, once the score falls due to loss of target, it remains low until the target is recaptured after which the scores remain high as long as the target is present.

Table 3. Comparative Analysis of STRUCK and EgoTracker

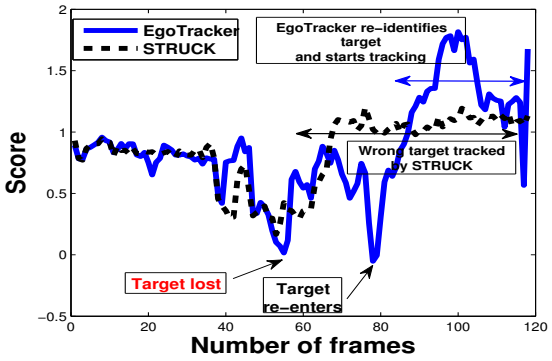
Videos	Time taken		Support Vectors		Re-identification	
	Ego Tracker	STRUCK	Ego Tracker	STRUCK	Ego Tracker	STRUCK
Exp-1	59s	114s	34	45	Yes	No
Exp-2	62s	122s	37	48	Yes	No
Exp-3	65s	125s	44	52	Yes	No
Exp-4	69s	128s	41	50	**No**	No
Exp-5	52s	110s	43	51	**No**	No
Yair-1	39s	68s	35	43	Yes	No
Yair-3	26s	45s	30	38	Yes	No
Huji-1	29s	48s	31	40	Yes	No
Huji-2	30s	68s	30	40	**No**	No
Human7	43s	90s	43	46	Yes	No

Fig. 10(b) shows the comparison between the time taken for STRUCK and EgoTracker. The time taken for EgoTracker includes time needed for optical flow computation. It is observed from the plot that the time required for repositioning is much smaller for EgoTracker even with optical flow computation.

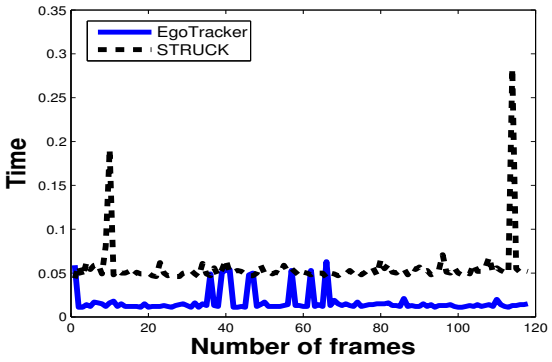
Table 3 gives additional results comparing EgoTracker and STRUCK in terms of time taken, number of support vectors and re-identification. It is observed that EgoTracker performs better compared to STRUCK in all the cases. There are three cases in which the proposed tracker fails, as in these cases the target re-enters the field of view with high scale change. Due to this heavy variation in scale, the extracted features are quite different from the existing support vectors in the target model, which results in low similarity score.

4. Conclusion

We present a fast tracker which re-identifies a pedestrian while tracking in egocentric videos in the presence of scale variations. Our work is motivated by the fact that in egocentric setting, the optical flow arising from head movement has a well defined pattern. The proposed tracker performs scale invariant tracking with re-identification in almost all the cases. Also the computation time is reduced by half as compared to STRUCK.



(a) Comparative scores



(b) Comparative timing

Figure 10. Comparative analysis of STRUCK and EgoTracker over Yair3 dataset.

References

[1] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[2] S. Alletto, G. Serra, and R. Cucchiara. Egocentric object tracking: an odometry-based solution. In *International Conference on Image Analysis and Processing*, pages 687–696. Springer, 2015.

[3] W. Choi, C. Pantofaru, and S. Savarese. A general framework for tracking multiple people from a moving camera. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1577–1591, 2013.

[4] A. Fathi, J. K. Hodgins, and J. M. Rehg. Social interactions: A first-person perspective. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1226–1233. IEEE, 2012.

[5] M. Funk, R. Boldt, B. Pfleging, M. Pfeiffer, N. Henze, and A. Schmidt. Representing indoor location of objects on wearable computers with head-mounted displays. In *Proceedings of the 5th Augmented Human International Conference*, page 18. ACM, 2014.

[6] A. S. Hare, Sam and P. H. Torr. Struck: Structured output tracking with kernels. IEEE International Conference on Computer Vision (ICCV), 2011.

[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, March 2015.

[8] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *In International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, 1981.

[9] G. Ning, Z. Zhang, C. Huang, Z. He, X. Ren, and H. Wang. Spatially supervised recurrent convolutional neural networks for visual object tracking. *CoRR*, abs/1607.05781, 2016.

[10] Y. Poleg, C. Arora, and S. Peleg. Temporal segmentation of egocentric videos. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[11] Y. Poleg, A. Ephrat, S. Peleg, and C. Arora. Compact cnn for indexing egocentric videos. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[12] J. Shi and C. Tomasi. Good features to track. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, Jun 1994.

[13] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, July 2014.

[14] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.