Improved 2D Human Pose Tracking Using Optical Flow Analysis

Aleksander Khelvas[1] Alexander Gilya-Zetinov[1], Egor Konyagin[2], Darya Demyanova[1], Pavel Sorokin[1], and Roman Khafizov[1]

[1] Moscow Institute of Physics and Technologies, Dolgoprudnii, Russian Federation
khelvas.av@phystech.edu,
WWW home page: http://www.mipt.ru
[2] Skolkovo Institute of Science and Technology,
Bolshoy Boulevard 30, bld. 1 Moscow, Russian Federation

Abstract. In this paper we propose a novel human body pose refinement method that relies on an existing single-frame pose detector and uses an optical flow algorithm in order to increase quality of output trajectories. First, a pose estimation algorithm such as OpenPose is applied and the error of keypoint position measurement is calculated. Then, the velocity of each keypoint in frame coordinate space is estimated by an optical flow algorithm, and results are merged through a Kalman filter. The resulting trajectories for a set of experimental videos were calculated and evaluated by metrics, which showed a positive impact of optical flow velocity estimations. Our algorithm may be used as a preliminary step to further joint trajectory processing, such as action recognition.

Keywords: video processing, human pose detection, skeleton motion

1 Introduction

Human motion tracking is an important application of machine vision algorithms that could be used for many business purposes. The most popular tasks in the digital world include distributed video surveillance system, solutions for digital marketing, solutions for human tracking in an industrial environment.

This task can have different levels of details. The high-level approach is object detection, when the position of human as a whole object is extracted and its bounding box in 2D or 3D space is estimated.

A more interesting approach would be to detect a human pose in motion. This task is more complicated because human pose has substantially more dimensions compared to a bounding box.

Recent advances in deep learning have resulted in efficient single-frame pose tracking algorithms, such as [14] [6]. By applying them sequentially to a video stream, a set of trajectories for joints may be obtained. However, since these algorithms usually analyze input frames independently, the obtained trajectories usually have various artifacts, such as discontinuities or missing points.

In the reported research, we solve a task of enhancing obtained joint trajectories for multiple persons in a scene by leveraging the temporal information using an optical flow algorithm.

2 Related work

The task of retrieving pose dynamics for all persons in the video may be considered as a variant of multiple object tracking (MOT) task, where the considered objects are not persons but individual pose keypoints. There are two major paradigms in the field of MOT - detection-based tracking and detection-free tracking [11]. In the first case, machine vision algorithm capable of detecting individual objects is applied to every frame separately and then individual detections are linked into trajectories. The second approach has no detection algorithm and instead relies on temporal changes in the video stream to detect objects. With the development of efficient real-time object detection algorithms in recent years, the detection-based approach has become dominant in the literature. However, independent analysis of video frames results in inevitable loss of information conveyed by temporal changes in the video. This information may be relevant to object detection and could help improve the tracker performance. Various approaches were suggested to combine these individual frame and temporal features.

For example, in [12] a novel approach to combine temporal and spatial features was proposed by adding recurrent temporal component to a convolutional neural network (CNN) designed to detect objects in a single frame. The outputs of object detection network in sequential frames were fed into recurrent neural network (RNN). The resulting architecture is then trained to predict the refined tracking locations.

In [1] a tracker using prior information about possible person pose dynamics is proposed. This information is modelled as a hierarchical Gaussian process latent variable model, and allows to impose some temporal coherency in detected articulations.

In [17] a method leveraging optical flow for a pose tracking is proposed. The velocities obtained from flow data are used to generate expected coordinates of a pose in next frame. Predicted coordinates are used later to form tracks by greedy matching.

Our research is based on OpenPose as a body pose detector, proposed in [3]. It is a real-time solution capable to detect a 2D pose of multiple people in an image. It uses a non-parametric representation, which is referred to as Part Affinity Fields (PAFs), to learn to associate body parts with individuals in the image. This bottom-up system achieves high accuracy and real-time performance, regardless of the number of people in the image.

3 Definitions

Fist let us define several frames of reference (FoR) for our research, which are shown in figure 1.

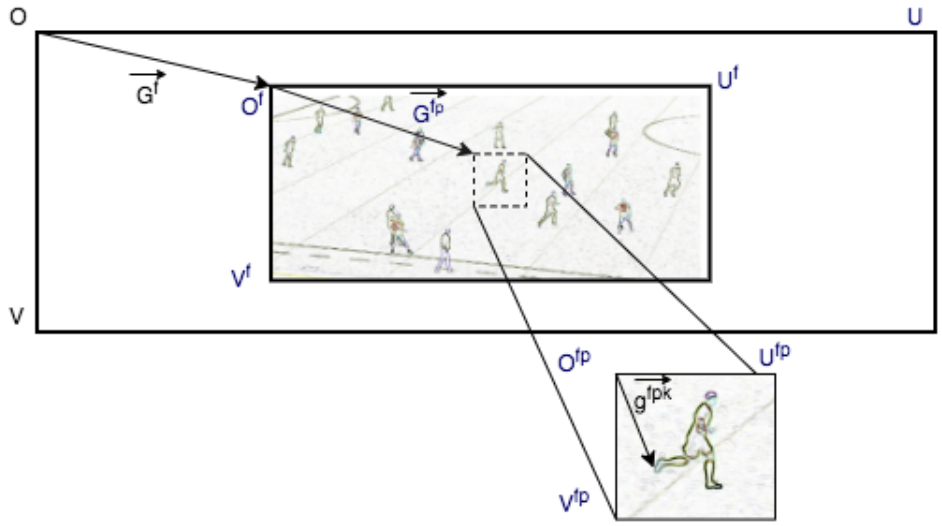

Fig. 1. Frames of references for 2D skeletons parameters calculation

U, V – this frame of reference is associated with virtual or real motionless camera.

U_c^f, V_c^f – this frame of reference is associated with frames in video. If camera is motionless, this FoR will be the same for all frames in video. This would be a common case of video surveillance systems for security or marketing.

U_p^{fk}, V_p^{fk} – this frame of reference is associated with object k, detected for the frame f.

We will not use index f for video processing of motionless camera viewed scenes.

4 Method

Our goal is to propose a novel algorithm for robust tracking of multiple person poses in the video stream by leveraging both temporal and spatial features of the data. To achieve this, we combine predictions done by a single-frame person/pose detection algorithm (such as OpenPose and YOLO) with Optical Flow - based estimations through a Kalman filter.

The complete algorithm is described below and shown in figure 2.

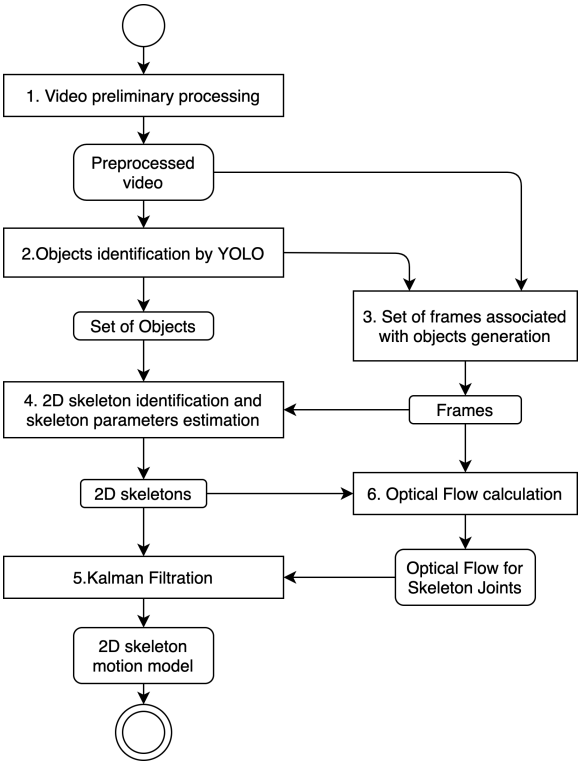

Fig. 2. Full algorithm for 2D skeleton model calculation and filtration

1. Video preliminary processing step produces a set of frames with normalized brightness/contrast and calculates G^f values.

2. Objects detection step provides a set of bounding boxes for each person, detected by YOLO or some other object detection algorithm.

3. Pose detection ROI generation step provides a set of input frame regions for further pose detection.

4. 2D pose estimation and person identification step computes a set of vectors $\mathbf{B}^{fp} = \{u_1^{fp}, v_1^{fp}, u_2^{fp}, v_2^{fp}, \ldots, u_N^{fp}, v_N^{fp}, \}$, where $N = 25$ is the number of joints for selected model of human body. (BODY-25 model provided by the OpenPose solution)

5. Optical Flow calculation step applies an optical flow estimation algorithm to the input frame, producing pixel velocity vectors for every joint position returned by the pose detector.

6. Kalman Filtration step calculates the time series for filtered movement vectors $\widehat{\mathbf{B}^{fp}} = \{\hat{u}_1^{fp}, \hat{v}_1^{fp}, \hat{u}_2^{fp}, \hat{v}_2^{fp}, \ldots, \hat{u}_N^{fp}, \hat{v}_N^{fp}, \}$

Let's discuss each step in detail.

After performing necessary source-specific video pre-processing, the next step would be extracting poses from single video frames where possible. We have selected an OpenPose-based solution as a human pose detector. OpenPose is a multi-person real-time pose keypoint detection algorithm, initially presented in [3]. An open-source implementation of OpenPose is used, providing pose keypoints in BODY-25 format. Reduced model example for 18 keypoints is shown in the fig. 3

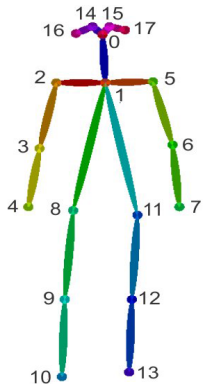

Fig. 3. Keypoints of a human pose model used in OpenPose (CMU-Perceptual-Computing-Lab, 2017)

However, direct application of OpenPose library to a high-resolution 4K video stream would not work. Since algorithm memory requirements grow linearly with input image area and amount of 3 GB for a default resolution of 656x656 is consumed, a distributed video processor system would be needed. Downscaling of the input image results in a drastic loss of detected pose quality. We solve this problem by splitting image into a set of overlapping regions and invoking the detector on these regions in a parallel manner, combining detection results afterwards. We can substantially boost the algorithm's efficiency, if a crowd on the video is sparse, which is often the case for video surveillance systems. Instead of processing the whole input frame, we employ an object detection algorithm to detect persons first and then build a set of regions that cover all persons' bounding boxes. We used YOLO (You Only Look Once)-based solution, as it performs fast detection of objects for 4K images. [13] It was observed that YOLO object detections are also useful for eliminating false positives generated by OpenPose.

For every frame in the input frame sequence, the algorithm first applies YOLO-based object detector trained on COCO dataset. [10] The result of YOLO

processing is a set of bounding boxes, with top left and bottom right corners defined in O_c^f, U_c^f, V_c^f FoR. A set of regions fully covering these rectangles is generated with resolution matching the selected input resolution of OpenPose network.

The result of video processing in the detection stage is a list of persons' bounding boxes for each frame. For each detected object we have the bounding box coordinates u_1, v_1, u_2, v_2, detection confidence and the OpenPose keypoint data if a skeleton was successfully matched with YOLO object: vector $\mathbf{B}^{fp}$. Additionally, we calculate approximate coordinates standard deviation for each keypoint by integrating over part heatmaps returned by the OpenPose. These values are later used as input for the Kalman filter as a measurement error estimate.

To further refine poses extracted from single frames, algorithm uses an optical flow solution. Optical flow is a technology used in various fields of computer vision to find displacement of individual pixels between two video frames. It is usually expressed as a 2D vector field $\{\mathbf{v}_{flow} = (dx, dy)^T\}$ for every pixel of the initial frame $I_n(x, y)$. Corresponding pixel in the next frame is $I_{n+m}(x + dx, y + dy)$. Many different approaches to optical flow calculation are proposed in the literature. In our work, we use several open source optical flow implementations provided by the OpenCV library.

The first one is presented in [7] called Dense Inverse Search. It belongs to the family of 'dense optical flow' algorithms and is notable for low computational complexity while it preserves good scores in standard optical flow benchmarks. The another one is called DeepFlow and was presented in paper [15].

The example of used soccer game frame for optical flow visualization calculated by two different algorithms is presented on figure 4.

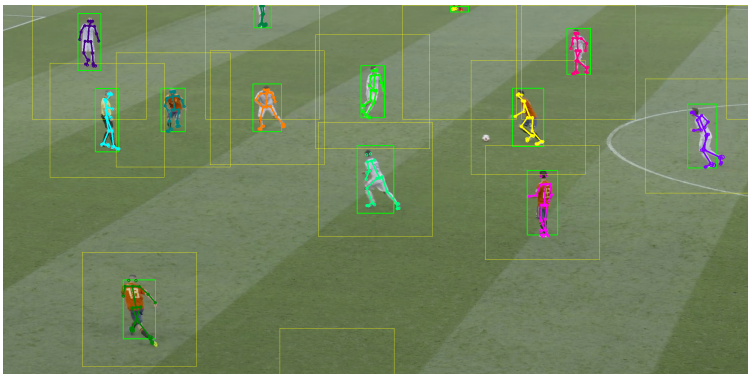

Fig. 4. The example of used soccer game frame for optical flow visualization calculated by two different algorithms

For Optical Flow visualization we use the HSV model. Hue represents the motion vector angle and saturation encodes the motion vector length.

The result of Dense-Inverse-Search algorithm for Optical Flow calculation is presented on figure 5.

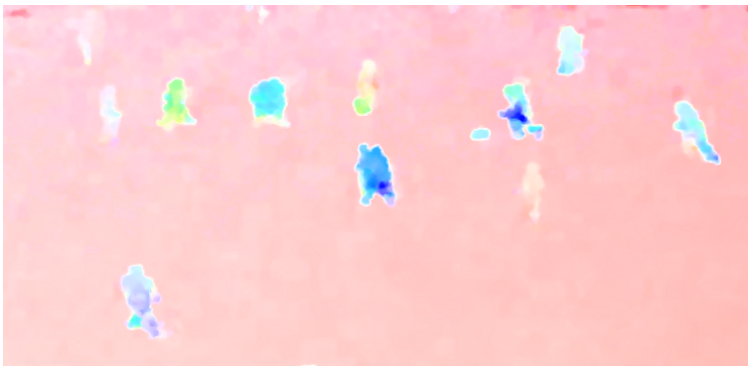

Fig. 5. Example of optical flow, calculated by Dense-Inverse-Search algorithm

The result of DeepFlow algorithm for Optical Flow calculation is presented on figure 6.

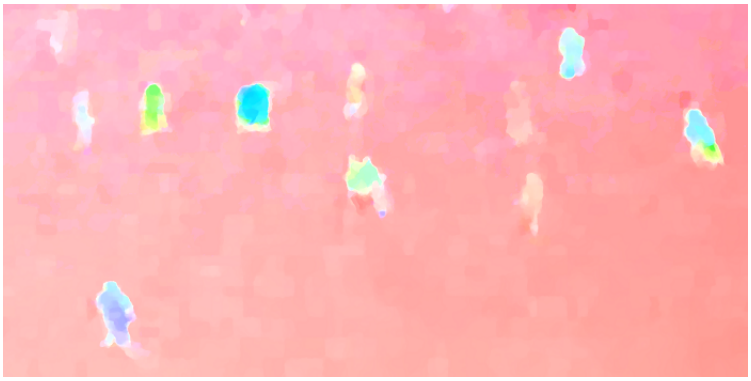

Fig. 6. Example of optical flow, calculated by DeepFlow algorithm

By applying a selected algorithm to every video frame in the input stream and by taking pixel velocity estimations at keypoints generated by OpenPose, we achieve a new velocity measurement.

We also need to build trajectories from individual detections in order to perform matching of detected poses belonging to the same person in different frames. [2] [9]

To combine pose keypoint measurements generated by OpenPose and corresponding pixel velocities estimated through optical flow, we use Kalman filter.

Kalman Filter was first proposed in 1960 [5] and, as a matter of fact, became the industry standard in the tasks related to fusion of measures performed by sensors of different types. Its application requires specification of a motion model for modeled object. There are several common motion models used in case the model of real motion is difficult or impossible to formalize. [8] Popular choices for a 2D case include a constant Cartesian velocity model and polar velocity non-linear model employing extended Kalman filter. [2] Alternative and more complex models can be implemented when 3D pose information is available, but their assessment lies beyond the scope of this work.

In our experiments we used the constant Cartesian velocity model applied independently to joint coordinates in 2D video frame FoR. In this instance, the state vector consists of 4 components representing estimated position and velocity of pose keypoint: $s(t) = (\hat{u}_t, \hat{v}_t, \dot{\hat{u}}_t, \dot{\hat{v}}_t)^T$. The state in the moment $t+1$ may be linked to the state values in the moment t with the following equation:

$$s(t+1) = \begin{pmatrix} 1 & 0 & \delta t & 0 \\ 0 & 1 & 0 & \delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \hat{u}_t \\ \hat{v}_t \\ \dot{\hat{u}}_t \\ \dot{\hat{v}}_t \end{pmatrix} + \begin{pmatrix} 0.5\delta t^2 & 0 \\ 0 & 0.5\delta t^2 \\ \delta t & 0 \\ 0 & \delta t \end{pmatrix} \begin{pmatrix} \nu_x(t) \\ \nu_y(t) \end{pmatrix} \tag{1}$$

and the process noise covariance matrix:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \bar{a}_u & 0 \\ 0 & 0 & 0 & \bar{a}_v \end{pmatrix} \tag{2}$$

For every experiment configuration, two independent trajectory estimation passes were performed - with and without optical flow velocity measurements.

5 Results

To evaluate performance of proposed solution for different applications, we prepared a set of video fragments.

They are:

1. a fragment of 4k soccer video broadcast
2. a video from indoor surveillance system in a supermarket
3. a video from outdoor surveillance system

The soccer match video had an additional pre-processing step - the fan stands were cut.

The fig 7 presents skeletons for soccer and supermarket cases.

Examples of filtered trajectories are presented on figs 8, 9.

The figure 8 a) presents selected and filtered U coordinate trajectories of an ankle of walking person on outdoor CCTV camera. Periodic increased difference

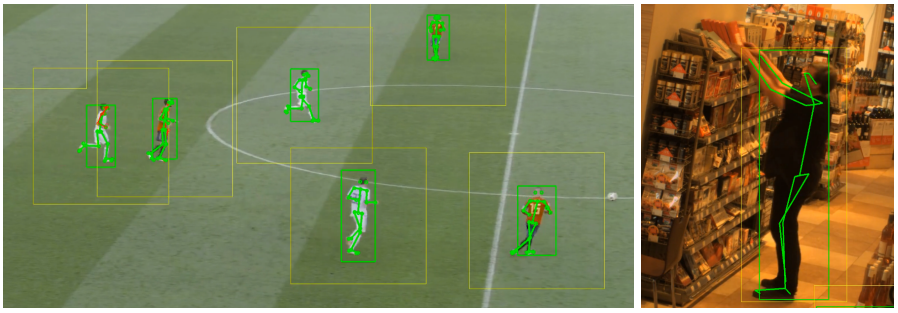

Fig. 7. Obtained execution results for the soccer and indoor scenes. Yellow rectangles correspond to different OpenPose calls, green rectangles are objects detected by YOLO.

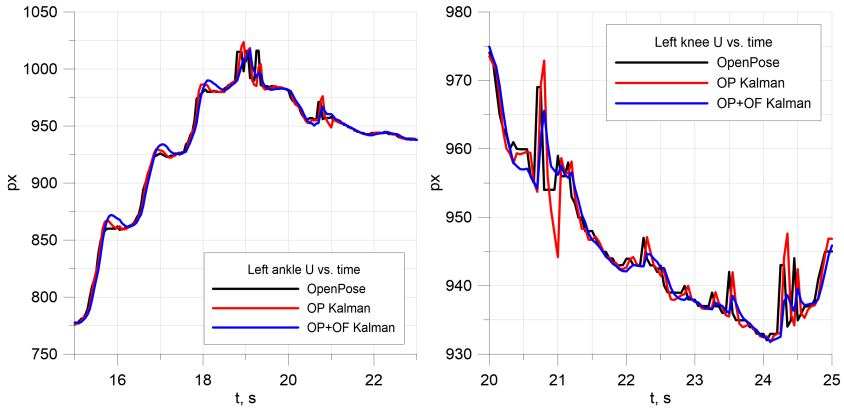

Fig. 8. a) Detected and filtered U coordinate trajectories of an ankle of walking person on outdoor CCTV camera. b) Detected and filtered U coordinate trajectories of a knee of walking person on outdoor CCTV camera

between filtered and detected value is caused by poor Kalman model prediction of footsteps.

The figure 8 b) presents detected and filtered U coordinate trajectories of a knee of a walking person captured by an outdoor CCTV camera. Short bursts of noise are caused by OpenPose detection errors occurring due to overlapping with other limbs.

The figure 9 a) presents detected and filtered U coordinate trajectories of a wrist of a walking person with partial visibility. Rough line sections are intervals of time with missing OpenPose detections, resulting in constant velocity for filtered trajectories.

The figure 9 b) presents detected and filtered U coordinate trajectories of an elbow of a soccer player.

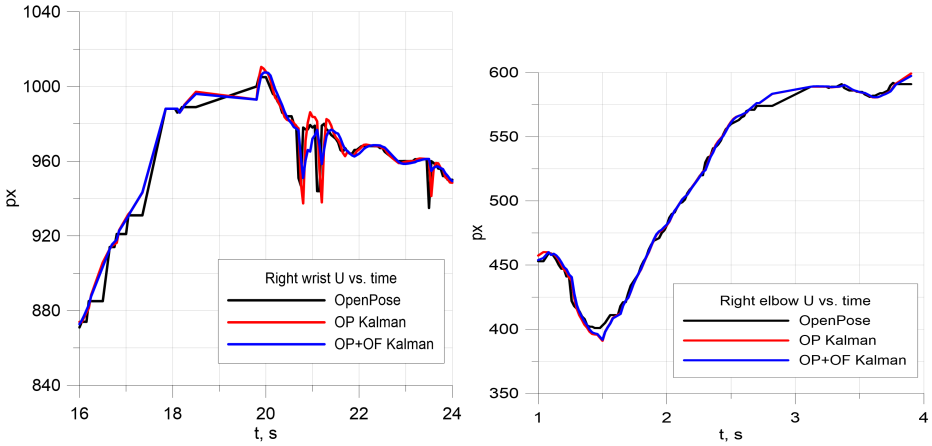

Fig. 9. a) Detected and filtered U coordinate trajectories of a wrist of walking person with partial visibility. b) Rough line sections are intervals of time with missing OpenPose detections, resulting in constant velocity for filtered trajectories.

It should be noted that the filter using Optical Flow measurements demonstrates a more stable behavior in parts of the video with frequent OpenPose detection errors. To evaluate the improvement, we introduced a quality metric.

To confirm the validity of Kalman filter application in the absence of known ground truth, a typical approach would be to extract standardized residuals and check if they follow the normal distribution with zero mean and constant variance. [4] However, in our case, a Kalman model is highly approximate and this method cannot be used for robust estimation of the impact of velocity measurements. Instead, we calculate a simpler metrics coming from a rather intuitive idea - the predicted values provided by a 'better' filter should have less difference from actual measured values on average. This may not be true, if prediction error can correlate with measurement error at the next step, but, in our case, the studied velocity measurement has different source of errors than measures provided by a single-frame detector.

With these assumptions, we calculated the average squared difference between predicted and measured coordinates for Kalman filter with and without optical flow velocity measurements:

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} |\mathbf{B}^{fp} - \widehat{\mathbf{B}^{fp}}|^2 \tag{3}$$

where Kalman output prediction is taken before adjusting for the measurement at that step. The sum is taken over all detected keypoints in the video with associated trajectory.

The results for different types of video are presented in the table 1. It may be concluded that both studied algorithms have a similar positive impact on the error, with Dense Inverse Search being slightly more effective.

Table 1. Results of quality metrics in absence of ground truth

Algorithm	σ^2, soccer	σ^2, outdoor	σ^2, PoseTrack
Open Pose + Kalman Filter	9.47	5.51	43.5
Open Pose + Kalman Filter + DIS	**8.86**	**4.93**	**38.4**
Open Pose + Kalman Filter + DeepFlow	9.06	5.06	39.1

To further validate performance of the proposed method, we used annotated videos from public pose tracking dataset called PoseTrack2018 [16]. This dataset provides means to quantify performance of algorithms in two different tasks - multi-person pose estimation task, through mean Average Precision metrics (mAP), and multiple object tracking task, through Multiple Object Tracker Accuracy metrics (MOTA). Effective multiple object tracking methods usually rely on global trajectory analysis which is necessary to process videos with crowds and large amount of occlusion. Since our method uses only local temporal data and does not aim to improve results in these cases, we used mAP metrics only, solving a multi-person multi-frame pose estimation (but not tracking) task. For PCKh and mAP values calculation, we used code provided in the evaluation repository referenced by PoseTrack official site.

The results for different algorithms are provided in the table 2.

Table 2. Results obtained on the PoseTrack dataset

Algorithm	Precision	Recall	AP
No filtering	79.4%	57.7%	51.9%
Kalman Filter	70.4%	56.6%	50.8%
Kalman Filter + DeepFlow	79.5%	**58.9%**	52.7%
Kalman Filter + DIS	**83.0%**	58.0%	**54.7%**

The OpenPose network in our case was not trained on the PoseTrack dataset, so the results should not be compared to its public leaderboard. In addition, the dataset used a different pose model with 18 keypoints per pose, which could have had a negative impact to metrics value. Also, the mAP metrics itself is not very good for estimating filtration quality since it classifies each predicted keypoint in a binary way - checking if it is closer to ground truth than some threshold or not. Nonetheless there is an improvement in experiments that used Optical Flow as an additional Kalman input. The performance degradation of filter without velocity measurements may be attributed to a lag due to trajectory smoothing, which was partially compensated by optical flow. Also, it was observed that errors in tracking cause notable disruptions in filtered trajectories absent in unfiltered case, which is an another source of errors.

6 Discussion and Conclusions

In our paper, we presented a method allowing to refine output of an arbitrary single-frame multi-person pose detection algorithm, by combining its output with an optical flow-based velocity estimator. We showed with openly available implementations that it is possible to improve pose estimation quality through Kalman filtration, both with annotated and unannotated data. However, the improvement achieved is relatively minor. We speculate that it may be developed further, taking the following points into account.

The used camera-space constant-velocity 2D Kalman model is actually one of the simplest models possible. Some of its limitations were shown in the results section - for example, it produces bad results in moments where limb acceleration is higher than usual - feet while stepping on the ground, ball strike, etc. It also does not take into account various kinematic restrictions pertaining to humans, such as limb length and angle restrictions. By replacing it with a more precise human body model, it may be possible to improve Kalman filter performance and overall result.

There is a vast amount of modern optical flow algorithms, including recent ones based on trainable convolutional networks, which were not tested. Also, their parameters may be fine-tuned, to better grasp individual limb movements. The used optical flow algorithms were observed to predict human movement as a whole object in some cases, depending on parameters, environment and image quality.

The filter performance in our experiments greatly depends on the object tracker, since Kalman filtration of wrong tracks tend to magnify errors even more. While missing limb detections for several frames can be handled by the Kalman filter well, detection and tracking artifacts that affect the whole pose (e.g. track swap happening during occlusion, merging of several poses to one) usually disturb output trajectory to a significant degree. For this reason, method can be futher improved by using Kalman predictions eariler at the pose generation step. (e.g. when building pose from heatmaps and PAFs in case of OpenPose detector).

In the future, we plan to validate the algorithm performance more using available open source pose tracking datasets and compare different optical flow algorithms. Also, it is our intention to validate performance of more complex Kalman models.

7 Acknowledgment

Authors thank XIMEA corp. CEO Max Larin for XIMEA cameras during the summer of 2019 for data collection and experiments with the adaptive video surveillance system.

Also, we would like to thank the MRTech directors Igor Dvoretskiy and Aleksandr Kiselev for the numerous and extensive discussions about video system architecture and video processing solutions and Fyodor Serzhenko from FastVideo corp. for comments about GPU usage for the on-board video processing.

Boris Karapetyan provided us with the skeleton visualization software components.

The reported study was funded by the RFBR, project number 19-29-09090.

References

1. Andriluka M., Roth S., Schiele B. (2008) People-tracking-by-detection and people-detection-by-tracking. In: 2008 IEEE Conference on computer vision and pattern recognition, IEEE, pp 1–8
2. Antonucci A., Magnago V., Palopoli L., Fontanelli D. (2019) Performance assessment of a people tracker for social robots. In: 2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), IEEE, pp 1–6
3. Cao Z., Hidalgo G., Simon T., Wei S.E., Sheikh Y. (2018) Openpose: Realtime multi-person 2d pose estimation using part affinity fields arXiv:1812.08008
4. Jalles J.T. (2009) Structural time series models and the Kalman filter: aconcise review
5. Kalman R.E., Others (1960) A new approach to linear filtering and prediction problems. Journal of basic Engineering 82(1):35–45
6. Kocabas M., Karagoz S., Akbas E. (2018) MultiPoseNet: Fast Multi-Person Pose Estimation Using Pose Residual Network: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI, pp 437–453. DOI 10.1007/978-3-030-01252-626
7. Kroeger T., Timofte R., Dai D., Van Gool L. (2016) Fast optical flow using dense inverse search. In: European Conference on Computer Vision, Springer, pp 471–488
8. Li X.R., Jilkov V.P. (2003) Survey of maneuvering target tracking.Part I. Dynamic models. IEEE Transactions on aerospace and electronic systems 39(4):1333–1364
9. Lin L., Lu Y., Pan Y., Chen X. (2012) Integrating graph partitioning and matching for trajectory analysis in video surveillance. IEEE transactions on Image Processing 21(12):4844–4857
10. Lin T.Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollar P., Zitnick C.L. (2014) Microsoft coco: Common objects in context. In: European conference on computer vision, Springer, pp 740–755
11. Luo W., Xing J., Milan A., Zhang X., Liu W., Zhao X., Kim T.K. (2014) Multiple object tracking: A literature review. arXiv preprint arXiv:14097618
12. Ning G., Zhang Z., Huang C., Ren X., Wang H., Cai C., He Z. (2017) Spatially supervised recurrent convolutional neural networks for visual object tracking. In: 2017 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, pp 1–4
13. Redmon J., Divvala S., Girshick R., Farhadi A. (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
14. Yuliang Xiu, Jiefeng Li, Haoyu Wang, Yinghong Fang, Cewu Lu. (2018). Pose Flow: Efficient Online Pose Tracking. In: British Machine Vision Conference (BMVC), 2018, arXiv:1802.00977
15. Weinzaepfel, Philippe and Revaud, Jerome and Harchaoui, Zaid and Schmid, Cordelia. (2013). DeepFlow: Large displacement optical flow with deep matching. In: Proceedings of the IEEE international conference on computer vision , 2013, pp 1385-1392

16. Andriluka, M., Iqbal, U., Insafutdinov, E., Pishchulin, L., Milan, A., Gall, J., Schiele, B. (2018). Posetrack: A benchmark for human pose estimation and tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5167-5176
17. Xiao, B., Wu, H., Wei, Y. (2018). Simple baselines for human pose estimation and tracking. In: Proceedings of the European conference on computer vision (ECCV), pp. 466-481.