Further Improvement on an MCMC-based Video Tracking Algorithm*

***D. Kuplyakov*[1,2], *E. Shalnov*[1,2], *A. Konushin*[2]**
dener.kup@gmail.com | eshalnov@graphics.cs.msu.ru | ktosh@graphics.cs.msu.ru
Russia, [1]Department of Computational Mathematics and Cybernetics
Lomonosov Moscow State University,
Moscow, Russia, [2]NRU Higher School of Economics

The paper considers a problem of multiple person tracking. We present the approach to automatic people tracking on surveillance videos recorded by static cameras. Proposed algorithm is an extension of [1] and is based on tracking-by-detection of people heads. It performs data association using Markov chain Monte Carlo (MCMC). Tracklet postprocessing and accurate results interpolation were shown to reduce number of false positives. We use position deviations of tracklets and revised entry/exit points factor to separate pedestrians from false positives. Finally, the paper presents a new method to estimate body position. Our evaluation shows results competitive to modern tracking methods.

Keywords: *computer vision, video analytics, multi-person tracking, MCMC DA, energy minimization.*

1. Introduction

Video tracking is an open computer vision problem. The goal is to detect all persons in video and estimate a track for each of them. Track is uniquely specified with the person and contains his location on every frame where he is presented. Tracking algorithms is a basic stage in the pipelines of video analysis systems. Tracking allows indexing of surveillance materials, detecting illegal actions on streets or collecting marketing statistics.

We address a problem of fully automatic video tracking recorded by static camera. The task has the following formal definition: as the input algorithm uses a video sequence from static camera $\{F_t\}_{t=1}^{N}$ with calibration matrix C. As the output it provides a set of tracks: $\{T_i\}_{i=1}^{M}$. Each track is a set of person locations in the video, specified by bounding boxes on the frames:

$$T_i = \left\{ \left(t_j^{(i)}, x_j^{(i)}, y_j^{(i)}, w_j^{(i)}, h_j^{(i)} \right) \right\}_{j=1}^{K_i}.$$

Modern multi-target tracking methods use the tracking-bydetection approach. It consists of three steps: 1) detection of objects on key frames; 2) building of tracks; 3) estimation of object locations on non-key frames.

Within the approach algorithms can be classified by temporal context used to build results for the specific frame. The first group uses only previous frames [2], another additionally uses information from the next frames in sliding window manner [1], [3]. The latter allows to achieve better accuracy with a drawback of increased latency.

The second key aspect is a detected object. The most popular case is to detect whole person image on the frame [2], [4], [5]. In that case relatively small number of false positives can be achieved, but occlusions significantly reduce performance. Another variant is to detect person head [1], [3]. It is more robust against occlusions, but detectors tend to produce more false positives. The third variant is to use DPM detectors [6] and incorporate parts locations together for tracking.

The third aspect is a choice of a data association algorithm. Association of tracking data is a merging of detections into tracks. Usually this problem is reduced to energy minimization, where the lower energy describes better grouping of detections. The next step is to find a suitable algorithm of energy minimization. Unfortunately, chosen energy function restricts set of inference techniques. The first group of algorithms takes into the account only connections between sequential detections of the track [1]–[3], [6]. Using only pairwise and unary energy terms allows optimization with Hungary algorithm [2] (in case of using only previous frame information) or min-cost max-flow technique. In case of high-order dependencies, like introducing type of the track, MCMC optimization is used [1], [3]. Second group use all pairwise connections of detections in the frame by solving generalized minimum clique graphs problem with specially developed methods [5].

Energy terms may use different features like color, speed, location, size, optical flow [4], etc. Many algorithms build tracklet for each detection as an intermediate step. Tracklet – is a part of track on a short segment of the video. It consists of object detection and estimates of it's position on near frames. Usually tracklets are the results of local tracking algorithms. They allows to build more accurate energy terms for the model.

In the paper we introduce an extension of algorithm [1] (baseline). It takes origins from the work [3]. The algorithm uses the approach of tracking-by-head-detection withMCMCdata association. With our

This work was supported by RFBR grant no. 14-01-00849 and by the Skolkovo Institute of Science and Technology, the contract 081-R, Annex A2. Работа опубликована по гранту РФФИ №16-07-20482

modifications it outperforms baseline on tracking quality. It's not the first attempt to modify baseline. In the work [7] authors use estimated 3D positions of pedestrians to perform tracking in world coordinates.

2. Proposed algorithm

The proposed algorithm consists of 4-stage pipeline (fig. 1). Below we describe each stage.

2.1 Detection

The tracked object of interest is a person. We detect people heads as they are more robust against occlusion, especially for high mounted cameras. HOG-based detector [8] is used for the task and applied only on key frames. F_1, F_{1+step}, $F_{1+2step}$,... are considered a key frames. Then detections are filtered using camera calibration to eliminate too small or big ones, that can't be some person head.

2.2 Building of tracklets

The algorithm applies "Flock of Features" [9] local tracking algorithm to construct tracklets. It is used to perform local tracking and constructing tracklets. Algorithm provides a confidences of estimations, that allows us to filter wrong ones. Local tracking tend to increase error during the time, thus it can be applied only in a small temporal neighborhood of the key frame.

2.2.1 Tracklet postprocessing

A confidence value provided by the "Flock of features" algorithm does not allow us to detect all misses of tracked target. Consider a person that leaves a camera field of view (fig. 2). That causes a lot of wrong estimations on one of tracklet's tails within the same location on the frames. It confuses probability model and produces a lot of false positives, on the next stages (section 2.4.1). We propose a postprocessing algorithm (alg. 1), that allows to remove such wrong estimates. It finds a group of non-moving estimates on the

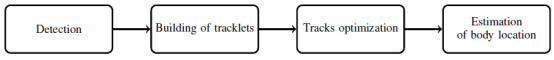

Рис. 1: Algorithm pipeline

tails of tracklet. If such group of acceptable size is found only on the one tail, it is erased from tracklet. In case of both tails have nonmoving estimations, we assume that the tracklet has a false positive base detection. Thus its non-moving estimates contain useful information. We are leaving such tracklets untouched.

2.3 Optimization

We use the same inference method as baseline [1], but change optimized energy function. Each

Рис. 2: Example of false positives on the tail of the tracklet. Head hid from the view on the 8-th frame, but local tracking were giving confident result up to the 33-rd frame after the base detection.

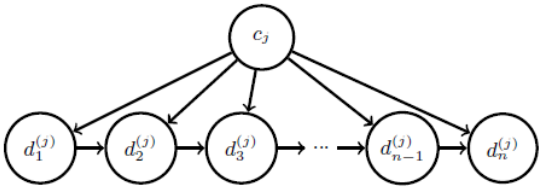

Рис. 3: Graphical model of track. c_j – type and $d_i^{(j)}$ – tracklets of the track.

track is described by graphical model (fig. 3), that incorporates type of the track, which can be one of c_{ped} or c_{fp}. That allows to filter tracks of false positive detections. According to this following probability is set as:

$$p(D|H) = \prod_{T_j \in H} \left(p(d_1^{(j)}|c_j) \prod_{d_n^{(j)} \in T_j \setminus d_1^{(j)}} p(d_n^{(j)}|d_{n-1}^{(j)}, c_j) \right) \quad (1)$$

where D – a set of all tracklets in a sliding window, H – a set of current tracks.

We propose two enhancement to increase tracking quality by changing the following model probabilities from the baseline:

$$p(d_1|c_j) = p(s_1)\, p(x_1)\, p(m_1|c_j) \quad (2)$$
$$p(d_n|d_{n-1}, c_j) = p(s_n|s_{n-1})\, p(x_n|x_{n-1}, c_j)\, p(m_n|c_j) \quad (3)$$

to new ones:

$$p(d_1|c_j) = p(s_1)\, p(x_1|c_j)\, p(pd_1|c_j) \quad (4)$$
$$p(d_n|d_{n-1}, c_j) = p(s_n|s_{n-1})\, p(x_n|x_{n-1}, c_j)\, p(pd_n|c_j) \quad (5)$$

Algorithm 1: Algorithm of deleting false positives from tracklet tails

Input : Tracklet $d = \{b_{back}, \cdots b_{base}, \cdots, b_{forw}\}$, $base$ – base frame; b_i, $i \in \overline{back, forw}$ – estimations; w, h – base detection size

Output: Tracklet $\hat{d} = \{b_{\widehat{back}}, \cdots, b_{base}, \cdots, b_{\widehat{forw}}\}$, $back \le \widehat{back} \le base \le \widehat{forw} \le forw$ without false positives

$theshold_{size} \leftarrow 1.25\, maxside(b_{base})$
/* $maxsize$ - the maximal side of rectangle */
$theshold_{count} \leftarrow 40$
/* two constants above should be considered as parameters */
Function *NumberToCut*$(start, \Delta)$
  $union \leftarrow rectangle(b_{start}, w, h)$
  /* builds rectangle by center and sizes */
  $i \leftarrow start + \Delta$
  $result \leftarrow 0$
  while $i \neq base + \Delta$ **do**
    if $maxsize(union) > theshold_{size}$ **then**
      break
    end
    $union \leftarrow unite(union, rectangle(b_i, w, h))$
    /* *unite* builds minimal rectangle, including given ones */
    $result \leftarrow result + 1$
    $i \leftarrow i + \Delta$
  end
  return $result$
$cut_{back} = NumberToCut(back, +1)$
$cut_{forw} = NumberToCut(forw, -1)$
if $cut_{back} \ge threshold_{count}$ **and** $cut_{forw} \ge threshold_{count}$ **then**
  return // base detection is false positive
end
delete from back tail cut_{back} estimations
delete from forward tail cut_{forw} estimations

Below we describe both of them in details.

2.3.1 Position deviation

In the work [3] authors propose to use histograms of motion magnitude to reveal false positive tracks. That approach has several drawbacks:

— there is no way to differ slowly moving, but covering a lot of distance, pedestrian from sticking around the same place false positive track;
— motion magnitude estimates are very inaccurate for nearest frames: pedestrians sometime don't move fast enough to move more than one-two pixels of distance per frame, which is less than detector and local tracking positioning precision.

To overcome these problems we take into the account deviation of estimated locations instead of motion magnitudes:

$$pd_n^{(j)} = \frac{1}{forw - back + 1} \sum_{i=back}^{forw} ||b_i - b_{base}|| \tag{6}$$

We take appropriate probabilities from the normal distribution:

$$p(pd_n^{(j)}|c_{ped}) \sim N(\mu_{ped}, \sigma_{ped}) \tag{7}$$

$$p(pd_n^{(j)}|c_{fp}) \sim N(\mu_{fp}, \sigma_{fp}) \tag{8}$$

, where $\mu_{ped}, \sigma_{ped}, \mu_{fp}, \sigma_{fp}$ is model parameters. We assume that false positive tracks cover less distance than pedestrians and have smaller variance:

$$\mu_{fp} < \mu_{ped} \quad \sigma_{fp} < \sigma_{ped} \quad \mu_{fp} \approx 0 \tag{9}$$

Introduced parameters are learned in the same fashion as in the baseline algorithm.

2.3.2 Entry/exit border accounting

The method of building a set of entry/exit points (border) were proposed in the baseline algorithm. It is the set of points, where algorithm expects to detect people entering/leaving of the camera field of view. Instead of only limiting location of the first tracklet of track, we will assume, that false positives don't depend on enter/exit border location.
As in the baseline, we have two probabilities:

$$p_1(x_1^{(j)}) = p(\rho(x_1^{(j)}, border) \sim N(0, \sigma_d^2)) \tag{10}$$

$$p_2(x_1^{(j)}) = S \tag{11}$$

The first makes tracklets to be located near the border, the second allows equiprobable positioning on the every point of the frame.
We are making $p(x_1^{(j)})$ dependent on the type of the track following way:

$$p(x_1^{(j)}|c_{ped}) = \frac{p_1(x_1^{(j)}) + p_2(x_1^{(j)})}{2} \tag{12}$$

$$p(x_1^{(j)}|c_{fp}) = p_2(x_1^{(j)}) \tag{13}$$

2.4 Estimation of body location

As the detection is done only on key frames, we need to get results for other frames. We split this problem in the following way: we estimate head locations first, then using results estimate body locations to provide the final algorithm output.

2.4.1 Estimation of head location on non-key frames

We propose to use local tracking results to estimate head location for non-key frames. The baseline use simple linear interpolation to solve this problem (fig. 4, a). Our approach will decrease positioning errors by:

— compensation of head up/down movement while walking;
— ability to estimate positions before the first base detection of the track and after the last one;
— not providing results for the frames, where no location estimates are presented; that will save algorithm from false positives on full occlusions.

The proposed algorithm is simple (fig. 4, b): we average location estimates from the two nearest tracklets of track for the frame (or one tracklet for frames before the first tracklet and after the last

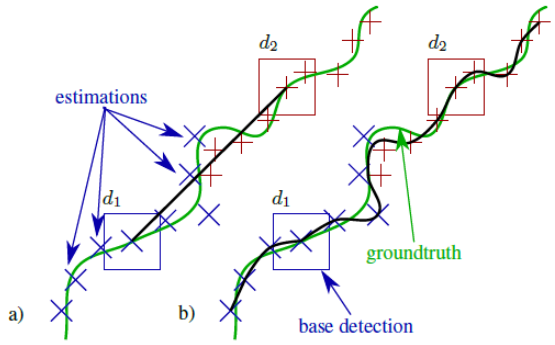

Рис. 4: Estimation of head location. There is track of two tracklets: blue d_1 and red d_2. Green line is the groundtruth track. Black line is the result of a) linear interpolation; b) using local tracking estimations.

one). If there is nothing to average (no tracklet covers specified frame), the algorithm produces no result.

2.4.2 Estimation of body location using average height

Camera calibration allows us to estimate location of feet from head position on the frame. Baseline algorithm used assumption that human height and size of the head are linked with constant coefficient. But size of the detections isn't accurate enough to achieve robust results (fig. 5a). We propose to use constant average height assumption to couple with the problem (fig. 5b). The width of bounding box is still estimated with fixed coefficient from height.

Рис. 5: Estimation of body location based on ...

3. Experimental evaluation

We have chosen TownCenter [3] dataset for experimental evaluation. It has camera calibration, 4500 frames with provided groundtruth tracks, including head and body bounding boxes. Video was recorded by the high-mounted static camera in 1920x1080x25fps format.
Standard MOTA/MOTP criteria [10] was used for comparison. MOTA measure takes into the account number of false positives, false negatives and identity switches. It shows how good tracks are build. MOTP is evaluated as an average intersection over union measure for corresponding detection bounding boxes of algorithm and groundtruth. Increase of measures denote improvement of algorithm. Both of them are bounded by 1 from the above, MOTP can't be less than zero. Values in tables will be given in percents rounded to have two digits. We used IoU threshold 0.2 for head and 0.5 for body evaluation.
To avoid influence of body location estimation stage we have evaluated algorithm on head bounding boxes tracks groundtruth (table 1).

Table 1: Experimental evaluation on head tracks.

Algorithm	MOTA	MOTP	FP	FN	ID
Benfold11 [3]	45	51			
baseline	62	56	7604	19144	217
↑ + in-extrapolation	66	**59**	9686	14526	**198**
↑ + tracklet-cut	69	**59**	**6980**	15243	207
↑ + pos-deviation	70	**59**	7667	13358	208
↑ + border-mod	**71**	**59**	7607	**13205**	246

We were adding each modification one by one to baseline algorithm:
1. in-extrapolation – estimation of head locations on non-key frames using tracklets (section 2.4.1);
2. tracklet-cut – tracklets postprocessing (section 2.2.1);
3. pos-deviation – using of position deviation (section 2.3.1);
4. border-mod – modified border accounting (section 2.3.2).

The first modification allows us to increase both MOTA and MOTP. Tracklets postprocessing allows decreasing number of false positives significantly. Position deviation and new border accounting allowed more robust track construction.
Evaluation of the resulting algorithm (with all proposed modification) on body tracks showed significant increase of tracking quality (table 2). Algorithm achieved performance comparable with modern methods.

Table 2: Experimental evaluation on body tracks. Comparison with modern methods. * means IoU threshold is 0.3 during evaluation

Algorithm	MOTA	MOTP	FP	FN	ID
Benfold11 [3]	61	**80**			
baseline [1]	53	68	11065	22605	**229**
all modifications	73	72	**6896**	**12494**	257
Izadinia12 [6]	**76**	72			
Dehghan15 [5]*	**76**	66			

4. Conclusion

We presented several modifications to the work [1]: 1) tracklet postprocessing, 2) using of position deviation instead of motion histograms, 3) new way to account entry/exit border, 4) estimation of head locations on non-key frames using local tracking results, 5) estimation of body location with average height assumption. Experimental evaluation showed that all proposed modifications increase tracking quality.

Литература

[1] Eugeniy Shalnov, Vadim Konushin, and Anton Konushin. "An improvement on an MCMC-based

video tracking algorithm". In: Pattern Recognition and Image Analysis 25 (2015), pp. 532–540.

[2] Alex Bewley, ZongYuan Ge, Lionel Ott, et al. "Simple Online and Realtime Tracking." In: CoRR abs/1602.00763 (2016).

[3] Ben Benfold and Ian Reid. "Stable Multi-Target Tracking in Real-Time Surveillance Video". In: CVPR. June 2011, pp. 3457–3464.

[4] Wongun Choi. "Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor." In: CoRR abs/1504.02340 (2015).

[5] Afshin Dehghan, Shayan Modiri Assari, and Mubarak Shah. "GMMCP tracker: Globally optimal Generalized Maximum Multi Clique problem for multiple object tracking." In: CVPR. IEEE Computer Society, 2015, pp. 4091–4099. ISBN: 978-1-4673-6964-0.

[6] Hamid Izadinia, Imran Saleemi, Wenhui Li, et al. "(MP)2T: Multiple People Multiple Parts Tracker." In: ECCV (6). Ed. by Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, et al. Vol. 7577. Lecture Notes in Computer Science. Springer, 2012, pp. 100–114. ISBN: 978-3-642-33782-6.

[7] Alexander Gringauz, Eugeniy Shalnov, and Anton Konushin. "Modification of the Multi-target Tracking Algorithm Based on Energy Minimization". In: GraphiCon- 2014. 2014, pp. 139–142.

[8] Victor Prisacariu and Ian Reid. "fastHOG-a real-time GPU implementation of HOG". In: Department of Engineering Science 2310 (2009).

[9] Mathias Kölsch and Matthew Turk. "Fast 2d hand tracking with flocks of features and multi-cue integration". In: Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on. IEEE. 2004, pp. 158–158.

[10] Keni Bernardin, Alexander Elbs, and Rainer Stiefelhagen. "Multiple object tracking performance metrics and evaluation in a smart room environment". In: Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV. Vol. 90. Citeseer. 2006, p. 91.