Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 875–881

DOI: 10.15439/2015F158
ACSIS, Vol. 5

Pedestrian tracking in video sequences: a particle filtering approach

Mateusz Owczarek, Przemysław Barański, Paweł Strumiłło
Institute of Electronics, Lodz University of Technology
Email: {mateusz.owczarek, przemyslaw.baranski, pawel.strumillo}@p.lodz.pl

***Abstract*—In this work we study the methods for pedestrian tracking in video sequences and indicate various applications of these methods ranging from surveillance systems to aiding the visually impaired persons. First, we define the general problem of object tracking that comprises the tasks of object detection, identifying the flow of object location in consecutive video images and finally analysis of the tracked trajectory data. We review the well known object tracking techniques i.e. the Mean-Shift and the CAMSHIFT algorithm and discuss their properties. Then we introduce the computational technique known as particle filtering (PF) and explain how we have applied it to the tasks of pedestrian tracking. We compare the PF approach against the Mean-Shift and the CAMSHIFT algorithms in terms of tracking robustness and the required computational demand. We conclude, that on the tested video sequences, the PF tracker outperforms the Mean-Shift and by a small margin the CAMSHIFT algorithm. The PF tracker requires more computational power, however, its tracking performance can be flexibly adjusted to the application requirements.**

I. Introduction

HUMAN BEINGS, whether in a standstill or in motion, have an extraordinary visual capability of detecting objects and tracking them in the environment. This powerful property of the human visual system allows people, e.g. to manoeuvre in crowded pavements without bumping into other pedestrians. Implementation of object tracking functionality in computer vision systems is a challenging task and has attracted researchers' interest for decades. This is because there are numerous applications in which object tracking is important. They range from civilian applications (human-computer interaction, robotics, surveillance systems, crowd sourcing systems) to military applications (guided missile systems) [1], [2], [3]. Tracking of objects in video sequences is a particularly difficult task due to the following reasons:

- varying illumination of the monitored scene,
- loss of depth information in mono-camera image acquisition systems,
- varying size and shape of the tracked object (due to changes in orientation and distance to the camera),
- occlusions of the tracked objects,
- motion of the tracking camera (i.e. both the tracked object and the background move in reference to the camera).

The object tracking task can be subdivided into the three major steps:

1) Object detection in a scene and determining its location (i.e. applying methods for segmenting out the object of interest from the background).
2) Identifying object position changes in consecutive image frames, termed object tracking.
3) Analysis of the object tracking data (e.g. determination of the motion trajectory, path prediction, etc.).

The latter processing step strongly depends on the application. Our research goal is to develop a system that would serve as a vision based travel aid for the visually impaired. Although, a number of GPS-based navigation systems or remote guidance systems were designed especially for the visually impaired, they are expensive and offer poor position accuracy in urban environments [4]. Also, electronic travel aids in which ultrasound or laser sensors are embedded into white canes and also more advanced sensory substitution solutions (e.g. using auditory display techniques) have not found wider acceptance among the visually impaired users [5]. Our approach to aiding the visual impaired in mobility and travel is to track the position of a blind pedestrian in a city environment on the basis of video sequences. Then the positioning data will be integrated with the digital map data to work out the navigation instructions for the blind user. Such a solution does not require any additional hardware to be carried by a blind pedestrian except for a mobile phone that would serve as a communication device between the system and the user.

In this communication we report on our preliminary studies aimed at developing and testing robustness of the vision based object tracking methods with special focus on systems used for pedestrians' tracking. In Section II we review the related work and highlight the widely used Mean-Shift and CAMSHIFT object tracking algorithms. In Section III we introduce the particle filtering algorithm and explain how we apply this powerful computing technique to person tracking. In section IV we describe the experimental tests of the algorithms and evaluate their performance on example object tracking tasks.

II. Review of related work

A. Object detection methods

Every tracking method requires an object detection technique to distinguish the tracked object from the background and/or other objects. The result of tracking strongly relies on the applied object detection method. These methods can be subdivided into four categories [1]:

This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No 643636 "Sound of Vision."

– **Point detectors** are used to detect characteristic points in a processed image (i.e. corners, edges). Among the most commonly used methods are: the *Moravec's operator*, the *Harris point detector* and the *Scale invariant Feature Transform* (SIFT). A comparative survey [6] provides more information on the topic.
– **Segmentation** leads to partitioning an image into characteristic regions (i.e. perceptually similar regions that can be used for tracking). The *Mean-shift* clustering [7], [8], the *Graph-cuts* and *Active contours* are within the most relevant image segmentation techniques.
– **Background modelling** is based on an assumption that moving objects appear on a very largely stable background [9], therefore the foreground can be obtained by "subtracting" the estimated background image from the current frame.
– **Supervised classifiers** that are based on a large collection of labeled samples composed of feature vectors. Samples are used as training data for the supervised learning method to generate a function that maps inputs (e.g. object features) to the desired outputs (e.g. class labels unambiguously binding the object with given features). These methods include neural networks, *adaptive boosting* (e.g. *Viola-Jones object detection* framework used to detect pedestrians [10]), *decision trees* and *Support Vector Machines* (SVM) in many variations [3].

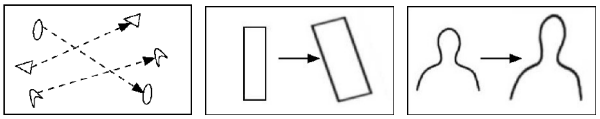
Among the image features commonly used in object tracking are [1], [11]:

– **Color** is used as a feature in histogram-based methods, where the object is represented by its appearance. Color spaces such as *L*a*b* and *HSV* (Hue, Saturation, Value) are more preferred in image processing, due to more perceptual uniformity.
– **Edges** are strong changes in the intensity or color in an image generated by object boundaries. Edges are less sensitive to illumination changes than color features.
– **Optical flow** defines the apparent motion of an object by a dense field of displacement vectors across consecutive image frames.
– **Texture** of an object described by a number of properties (such as lightness, density, regularity, linearity, directionality, smoothness, etc. [12]) is an excellent feature to track.

B. Approaches to object tracking

Object tracking techniques can be subdivided into the three major categories:

– **Point Tracking** relies on the positions and motion of points representing the target object in consecutive frames. Therefore, object tracking can be defined as the problem of finding points' correspondences (Fig. 1a).
– **Kernel Tracking** relies on the shape or appearance of the target object (referred to as *kernel*), which is represented by a geometric primitive (e.g. a rectangular patch or an ellipse), see Fig. 1b. The most popular representative of this category is the *Mean-Shift* [7], [8], [13] and its modification, the *CAMSHIFT* (*Continuously Adaptive Mean-Shift*) algorithm [14] (see Fig. 2).
– **Silhouette Tracking** relies on the information encoded inside the region of a tracked object which usually, due to the complexity of its shape, cannot be described well using simple geometric primitives (Fig. 1c).



(a) Multi-point correspondence (b) Parametric transformation of a rectangular patch (c) Contour evolution

Fig. 1. Different tracking approaches (source: [1])

(a)

(b)

Fig. 2. In the Mean-Shift algorithm (a) the search window size is fixed throughout the tracking session, whereas in the CAMSHIFT (b) the search window continuously adapts itself in size and orientation to fit the target object

The idea of applying particle filtering to object tracking was independently proposed by several research groups and is described in [15], [16] and [17] among others. Although the computational and probabilistic origins of the particle filter usually remains (more or less) the same, different approaches use different features to define the target model. Typically, edge-based image features are used [18], [15], [19], but color-based image features are also an option. The latter is even more robust against the out-of-plane rotations (e.g., when a person turns around), scale and rotation invariant [20], but more sensitive to the illumination changes (as indicated in Section II-A). For this reason, the color-based particle filter has become the object of our study.

Reference [2] presents recent trends in object detection, while [3] evaluates the state-of-the-art tracking algorithms.

III. Application of Particle Filtering to Person Tracking

A. Particle filter basis

Particle filtering, also called the *Sequential Monte Carlo* (SMC) [21] method, is a simulation-based technique which stems from the Monte Carlo method. The latter is a simple,

yet effective, way of finding an optimal solution for multidimensional problems by randomly generating a large number of possible system states. This enables to observe the overall system behaviour and select the best solution.

In the particle filter approach, the distribution of the $\mathbf{s}_t$ is approximated by a set of so called particles. Every particle is represented by the vector:

$$\mathbf{c}_t^{(n)} = \left[\mathbf{s}_t^{(n)} \ \pi_t^{(n)}\right]^T \tag{1}$$

where the superscript (n) denotes a particle number ranging from 1 to N being the size of the particle set and t denotes a time instant. Hence, the vector $\mathbf{s}_t^{(n)}$ represents the system state, that is the variables of interest, $\pi_t^{(n)}$ is a particle weight, i.e. a value that reflects how accurately a given particle approximates the system state.

The posterior distribution of $\mathbf{s}_t$ is approximated by the probability mass function:

$$p(\mathbf{s}_t|\mathbf{y}_{0:t}) \approx \sum_{n=1}^{N} \delta(\mathbf{s} - \mathbf{s}_t^{(n)}) \cdot \pi_t^{(n)} \tag{2}$$

where $\delta(\cdot)$ is the Dirac delta function.

The algorithm can be summarized in its basic form as follows [22]:

1) **Initialization**. At the algorithm's outset, all particles' states $\mathbf{s}_0^{(n)}$ are randomly initialized according to a given distribution. The weights $\pi_0^{(n)}$ are assigned equal values of $\frac{1}{N}$.
2) **Prediction.** New particle states are predicted on the base of the transition equation:

$$\mathbf{s}_t^{(n)} = \mathbf{f}\left(\mathbf{s}_{t-1}^{(n)}, \mathbf{u}_{t-1}, \mathbf{w}_{t-1}^{(n)}\right) \tag{3}$$

where $\mathbf{u}_{t-1}$ is a driving vector and $\mathbf{w}_{t-1}^{(n)}$ is the noise vector introduced to the state due to the measurement error of $\mathbf{u}_{t-1}$. Each particle is perturbed with an individually generated vector $\mathbf{w}_{t-1}^{(n)}$; $\mathbf{f}(\cdot)$ is the transition function that calculates a new state on the base of the previous one and the driving signals.
3) **Measurement update**. Each measurement $\mathbf{z}_t$ updates the weights of the particles by the equation:

$$\pi_t^{(n)} = \pi_{t-1}^{(n)} \cdot p\left(\mathbf{z}_t|\mathbf{s}_t^{(n)}\right) \tag{4}$$

where $p\left(\mathbf{z}_t|\mathbf{s}_t^{(n)}\right)$ is a conditional probability density of measuring $\mathbf{z}_t$ given the particle state $\mathbf{s}_t^{(n)}$. Particles that diverge in the long run from measurements will have small weights $\pi_t^{(n)}$.
4) **Weights' normalization.** For the sake of the next steps, the weights need to be normalized so that they sum up to 1:

$$\pi_t^{(n)} := \frac{\pi_t^{(n)}}{\sum_{i=1}^{N} \pi_t^{(n)}}. \tag{5}$$

5) **State estimation**. The system state is the weighted average of all particles' states:

$$\bar{\mathbf{s}}_t = \sum_{i=1}^{N} \mathbf{s}_t^{(n)} \cdot \pi_t^{(n)}. \tag{6}$$

6) **Resampling**. After a number of algorithm iterations, all but a few particles have negligible weights and therefore do not participate in the simulation effectively. This situation is detected by calculating the so-called *degeneration indicator*, expressed by:

$$d_t = \frac{1}{N \sum_{i=1}^{N} \left(\pi_t^{(n)}\right)^2}. \tag{7}$$

As the weights start to differ, the d_t indicator decreases. If d_t falls below a given threshold, then a process called resampling is evoked and a new set of particles is created. Resampling causes that the probability density function of $\mathbf{s}_t$ is refined in the next iterations of the algorithm. Thus, a better estimate can be found.
7) **Go to point 2**

B. Implementation of particle filter for person tracking

The implementation of the particle filter tracker described herein is based on the idea behind the *ConDensation Algorithm* [15], once implemented in the OpenCV library, yet presently deprecated due to some imperfections in the resampling part.

Target object is represented by a rectangular patch (Fig. 4a). Such representation is suitable for representing simple non-rigid targets and works well in terms of kernel-based tracking methods [1]. Target model is defined by a (8×8×4 bins) HSV histogram of the target object's representation and an initial size of the representation rectangle (during the tracking process it is scaled in relation to its initial size). Reduced number of levels for the V-component of the histogram makes the method less sensitive to the changes in lighting conditions. The use of HSV histogram may, however, make the method more sensitive to noise [1]. Current version of the algorithm does not update the target model automatically, but a new target model can be built on-the-fly.

Due to the fact that the principle of operation of the particle filter was described in the preceding section, in the subsequent paragraphs we would like to limit ourselves just to the adjustments which make use of particle filter in object tracking.

a) Initialization: A set of N-samples is generated. Each sample represents the quantities of the target object's representation and is defined by a state vector and an initial weight:

$$\begin{aligned} \mathbf{s} &= [x, y, \frac{\mathrm{d}x}{\mathrm{d}t}, \frac{\mathrm{d}y}{\mathrm{d}t}, k]^T \\ \mathbf{s}_0^{(n)} &= [x, y, 0, 0, 1]^T \text{ for } n = 1, ..., N \\ \pi_0^{(n)} &= N^{-1} \text{ for } n = 1, ..., N \end{aligned} \tag{8}$$

where (x, y) are the coordinates of the center of the representation rectangle, $\left(\frac{\mathrm{d}x}{\mathrm{d}t}, \frac{\mathrm{d}y}{\mathrm{d}t}\right)$ represents velocity vector (motion model), k is the scale of the representation rectangle in relation to its initial size and N is the number of samples.

b) Evolution of the samples: In every new frame a state of each sample is generated by a second order linear difference equation based on prior observations:

$$\mathbf{s}_t^{(n)} = \mathbf{A}\, \mathbf{s}_{t-1}^{(n)} + \mathbf{w}_{t-1}^{(n)} \tag{9}$$

where $\mathbf{w}_{t-1}^{(n)} \sim N(\mathbf{0}, \mathbf{Q}_t)$ is a vector of multivariate normally distributed random variates (stochastic component) and $\mathbf{A}$ is a *transition matrix* (deterministic component) defined for state vector (8):

$$A = \begin{bmatrix} 1 & 0 & \mathrm{d}t & 0 & 0 \\ 0 & 1 & 0 & \mathrm{d}t & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

c) Weighting the samples: Each sample represents a hypothetical location of the tracked object. To weight the sample set, a histogram of each hypothetical target representation is compared to the target model. To quantify similarity between the two histograms, the *Hellinger distance* is used:

$$\begin{aligned} d_t^{(n)} &= \sqrt{1 - \frac{1}{M\sqrt{\bar{H}_1 \bar{H}_2}} \sum_{u=1}^{M} H_1(u) \cdot H_2(u)} \\ \bar{H}_k &= \frac{1}{M} \sum_{u=1}^{M} H_k(u), \end{aligned} \tag{11}$$

where H_1 and H_2 are the histograms to be compared and M is a total number of histogram bins. The result is a score in range $[0, 1]$, where the first value indicates a perfect match and the latter a complete mismatch. The distance is used in the measurement equation to update the weights of the samples, according to:

$$\pi_t^{(n)} = \pi_{t-1}^{(n)} \cdot p(\mathbf{z}_t | \mathbf{s}_t) \tag{12}$$

$$\pi_t^{(n)} = \pi_{t-1}^{(n)} \cdot \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(d_t^{(n)}\right)^2}{2\sigma^2} \right), \tag{13}$$

where σ is the standard deviation. The higher the weight, the higher chance that the particle will be drawn during the next iteration (particles with the lowest weights are to be replaced).

TABLE I
DETAILS OF THE TEST VIDEO SEQUENCE

Parameter	Value
Format	320×240 at 25 frames/s
Length	1017 frames ($\sim$ 41 sec.)
Target object	person
Keywords	moving cam, moving target, non-rigid target, rotation, similar distractors, full occlusion, outdoor
Link to dataset	[23] /datasets/seqI.zip

The state of the tracked object at a time-step t can be therefore estimated as the mean state [15]:

$$\bar{\mathbf{s}}_t = \mathrm{E}\left[\mathbf{s}_t^{(n)}\right] = \sum_{n=1}^{N} \pi_t^{(n)} \cdot \mathbf{s}_t^{(n)}. \tag{14}$$

d) Re-initialization: The process of propagation of samples is being continued until the mean state moves off the image or the probability of the mean state drops below a certain threshold. It usually means that the target object has been lost due to a mismatch between the predicted and actual motion. In such scenario, particles are redistributed in accordance with the continuous uniform distribution to re-acquire the target:

$$\mathbf{s}_t^{(n)} = \begin{bmatrix} x \\ y \\ \mathrm{d}x/\mathrm{d}t \\ \mathrm{d}y/\mathrm{d}t \\ k \end{bmatrix} \sim \begin{bmatrix} U(0, W_{image}) \\ U(0, H_{image}) \\ U(-0.05, 0.05) \\ U(-0.05, 0.05) \\ U(1.0, 2.0) \end{bmatrix} \tag{15}$$

where (W_{image}, H_{image}) is the size of the image.

IV. RESULTS OF EXPERIMENTAL TESTS

A. Test results and performance measures

The implemented PF tracker has been tested and evaluated on a few short video sequences from the BoBoT benchmark on tracking dataset [23]. Here we present results obtained for one of the particularly difficult sequence which shows an individual walking along a pavement and is being followed by a camera (Fig. 4a). Ever and again a different person appears and passes by, shadowing the tracked pedestrian (so-called *occlusion*, cf. Fig. 4b), or is dressed in the same manner (so-called *similar distractor*, cf. Fig. 4c), trying to confuse the algorithm. Table I provides more details on the dataset.

The basic evaluation of the tracking algorithm relies on the overlap rate calculated every frame of the video sequence and defined as $\frac{\mathrm{area}(R_T \cap R_G)}{\mathrm{area}(R_T \cup R_G)}$, where R_T is the rectangular region of the tracking result and R_G denotes the ground truth [3]. If the overlap is larger than or equal to $1/3$, it is considered a hit (as shown in Fig. 5), otherwise a miss of target is denoted. Tests with a various number of particles show that $N = 100$ particles seems to be a good trade-off between accuracy and processing time of the algorithm (cf. Fig. 3). An interested reader is referred to the video sequence published at [25], which shows how the result of the tracking algorithm depends on the number of particles.

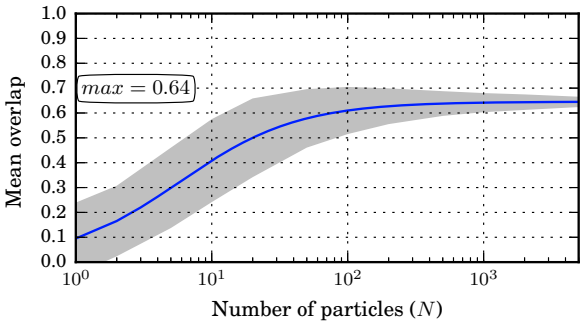


Fig. 3. The mean of ground truth and the tracking result overlap for $N = 100$ particles is equal to 0.62, which seems to be a trade-off between accuracy and processing time. Gray region is the standard deviation

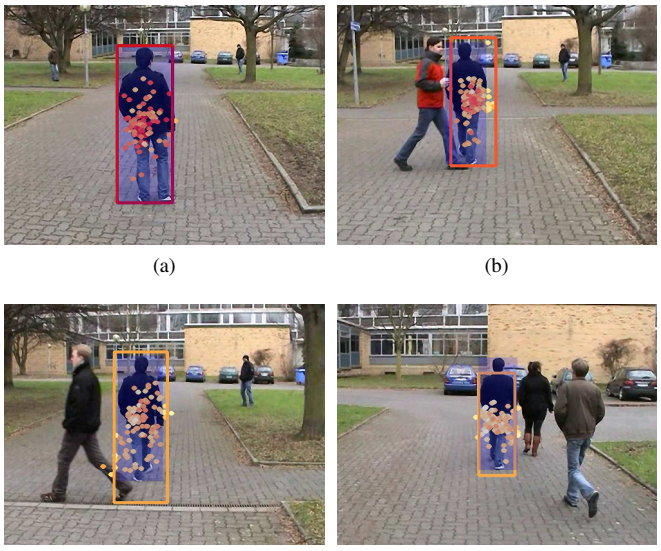


Fig. 4. PF tracker in action — tracked person is being followed by "particles" (each particle represents a hypothetical location of the tracked object). The tracking result is an outline around the target. Blue semi-transparent rectangle is the ground truth (source: [24])

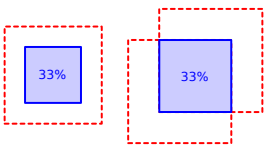
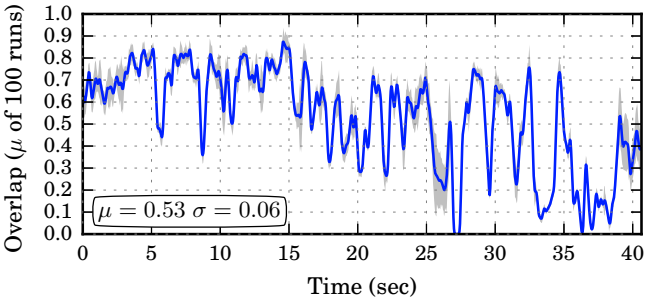
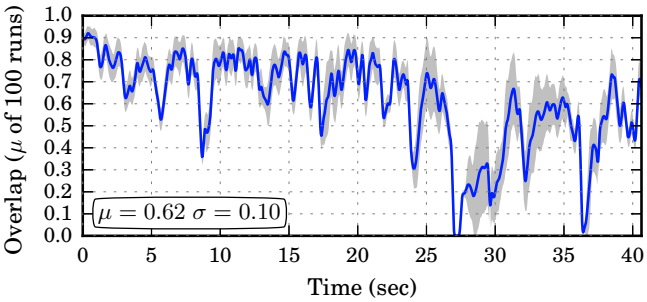


Fig. 5. Two examples of the ground truth (blue rectangle in the middle) and the tracking result (red dashed rectangle) overlap of $1/3$



(a) CAMSHIFT (best 100 of 278,784 results)



(b) Implemented PF tracker (for $N = 100$ particles)

Fig. 6. The ground truth and tracking result overlap per frame for the test video sequence. Grey region is the standard deviation

Fig. 6b shows the overlap for each frame of the test video sequence (mean of 100 iterations) using $N = 100$ particles. At the beginning the overlap reaches up to 92% and oscillates around 80% for most of the time. Each drop below $1/3$ (a miss) is caused by a full occlusion, when the algorithm has lost the target and needed to be re-initialized. Slightly worse results in the last quarter of the video sequence are caused by the out-of-plane rotation of the tracked individual. After the person turned right, his right side is visible on the video instead of his back, on which the target model used to be built (as it was mentioned before, current version of the algorithm does not update the target model automatically, yet an adaptive approach is within the further work). Notwithstanding, the results are satisfactory — the PF tracker copes well with similar distractors and occlusions. A non-rigid target and motion of both the camera and the target also pose no problem.

B. PF based tracker vs. the MeanShift and the CAMSHIFT

As indicated in Section II-B, color-based image features are robust against the out-of-plane rotations, scale and rotation changes. Additionally, the use of histogram matching results in relatively low computational cost. For this reason, color-based segmentation and tracking methods such as *Mean-Shift* and its extension *CAMSHIFT* (*Continuously Adaptive Mean-Shift*, cf. Fig. 2 upon the differences between these two methods), both within the OpenCV library, gained great popularity over the years.

CAMSHIFT is a parameter-free[1] tracking technique. It re-

[1]In the OpenCV library user can, however, define the maximum number of Mean-Shift iterations to converge

TABLE II
COMPARISON OF THE CAMSHIFT AND THE IMPLEMENTED PF TRACKER

	CAMSHIFT	**PF tracker**
Mean of ground truth and tracking result overlap	0.53 ± 0.06	0.62 ± 0.10
Mean of hit ratio[a]	0.79 ± 0.02	0.87 ± 0.04
Mean number of Mean-Shift iterations per frame	1.90 ± 0.52	n.a.
Mean execution time per frame[b]	0.73 ± 0.03 ms	5.55 ± 0.00 ms

[a]Hit ratio is the percentage of overlaps greater or equal to $1/3$.
[b]It should be noted that both the PF tracker and the evaluation script for the CAMSHIFT were implemented in Python, which tends to be slow.

quires a probability image of the target object (i.e. the back-projection of the object histogram) and an initial search window. Our test CAMSHIFT-based pedestrian tracker was based on the sample from the OpenCV library source code. It utilizes the HSV histogram of the target object masked with a binary image resulting from the thresholding of the initial image of the target object with different boundaries of colors. The purpose of masking was to extract as much of the target object from the background as possible to avoid false positives. A total number of over 270,000 different masks were used and only first one hundred results in terms of mean overlap were taken into account. The results are presented on Fig. 6a and tabulated in Table II.

As one can note, in several cases the CAMSHIFT-based pedestrian tracker performs marginally better than the implemented PF tracker, although the mean overlap and hit ratio of the former one are clearly lower. The CAMSHIFT-based tracker is also firmly faster. Please bear in mind though, that the result of the CAMSHIFT tracker strongly relies on the probability image of the target object. In this context it was selected carefully so as to correspond strictly to the video sequence used.

Finally, we encourage the reader to watch our short comparison video sequence available at [24].

V. CONCLUSIONS

In this study we have undertaken the problem of object tracking in video sequences with a special focus on pedestrian tracking. Our objective was to compare the performance of the Mean-Shift and Camshift trackers to the powerful computation technique known as particle filtering. We have explained theoretical background of the PF and shown how we have adapted it for the purpose of pedestrian tracking. From the tests of the compared trackers on example video sequences (shot from a camera in motion) we conclude that the PF can outperform the the Mean-Shift and Camshift trackers. This is, however, at the cost of higher computational demand (cf. Table II summarizing quantitative comparison of the performance of the compared trackers). We argue, however, that the PF can be employed to track multi-modal distributions, i.e. that are not confined to Gaussian distributions. Finally, the strong advantage of the PF is that its computing implementations can be mapped to parallel processing architectures with a flexibly chosen number of particles. Our intention is to employ the developed tracking technique in a multi-camera video system aimed at aiding the visually impaired in mobility and navigation.

ACKNOWLEDGMENT

This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No 643636 "Sound of Vision."

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, dec 2006. doi: http://dx.doi.org/10.1145/1177352.1177355

[2] J. Kulchandani and K. Dangarwala, "Moving object detection: Review of recent research trends," in *Pervasive Computing (ICPC), 2015 International Conference on*, Jan 2015. doi: 10.1109/PERVASIVE.2015.7087138 pp. 1–5.

[3] Y. Wu, J. Lim, and M. Yang, "Online object tracking: A benchmark," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, June 2013. doi: 10.1109/CVPR.2013.312 pp. 2411–2418. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2013.312

[4] P. Baranski and P. Strumillo, "Enhancing positioning accuracy in urban terrain by fusing data from a GPS receiver, inertial sensors, stereo-camera and digital maps for pedestrian navigation," *Sensors*, vol. 12, no. 6, pp. 6764–6801, 2012. doi: http://dx.doi.org/10.3390/s120606764

[5] M. Bujacz, P. Skulimowski, and P. Strumillo, "Naviton-a prototype mobility aid for auditory presentation of three-dimensional scenes to the visually impaired," *Journal of the Audio Engineering Society*, vol. 60, no. 9, pp. 696–708, 2012.

[6] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, Oct 2005. doi: http://dx.doi.org/10.1109/cvpr.2003.1211478

[7] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," *Computer Vision and Pattern Recognition*, vol. 2, no. 1, pp. 142–149, 2000. doi: http://dx.doi.org/10.1109/cvpr.2000.854761

[8] D. Comaniciu and V. Ramesh, "Mean shift and optimal prediction for efficient object tracking," in *Image Processing, 2000. Proceedings. 2000 International Conference on*, vol. 3, 2000. doi: http://dx.doi.org/10.1109/icip.2000.899297. ISSN 1522-4880 pp. 70–73.

[9] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981

[10] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, vol. 2, Oct 2003. doi: http://dx.doi.org/10.1109/iccv.2003.1238422 pp. 734–741.

[11] B. Deori and D. M. Thounaojam, "A survey on moving object tracking in video," *International Journal on Information Theory*, vol. 3, no. 3, July 2014. doi: http://dx.doi.org/10.5121/ijit.2014.3304

[12] A. Materka and M. Strzelecki, "Texture analysis methods–a review," Technical University of Lodz, Institute of Electronics, Brussels, Tech. Rep., 1998, cOST B11 report.

[13] A. Yilmaz, K. Shafique, N. Lobo, X. Li, T. Olson, and M. A. Shah, "Target-tracking in flir imagery using mean-shift and global motion compensation," in *Workshop on Computer Vision Beyond the Visible Spectrum, Kauai*, 2001, pp. 54–58.

[14] G. R. Bradski, "Real time face and object tracking as a component of a perceptual user interface," in *Proceedings of the Fourth IEEE Workshop on Applications of Computer Vision*, 1998. doi: http://dx.doi.org/10.1109/acv.1998.732882

[15] I. Michael and B. Andrew, "Condensation — conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998.

[16] N. Gordon and D. Salmond, "Bayesian state estimation for tracking and guidance using the bootstrap filter," *Journal of Guidance, Control and Dynamics*, vol. 18, no. 6, pp. 1434–1443, 1995. doi: http://dx.doi.org/10.2514/6.1993-3701

[17] G. Kitagawa, "Monte carlo filter and smoother for non-gaussian nonlinear state space models," in *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, 1996. doi: http://dx.doi.org/10.2307/1390893 pp. 1–25.

[18] T. Heap and D. Hogg, "Wormholes in shape space: Tracking through discontinuous changes in shape," in *International Conference on Computer Vision*, 1998. doi: http://dx.doi.org/10.1109/iccv.1998.710741 pp. 344–349.

[19] J. MacCormick and A. Blake, "A probabilistic exclusion principle for tracking multiple objects," vol. 1, pp. 572–587, 1999. doi: http://dx.doi.org/10.1007/978-1-4471-0679-1_6

[20] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "A color-based particle filter," in *First International Workshop on Generative-Model-Based Vision*, A. Pece, Ed., vol. 2002/01. Datalogistik Institut, Kobenhavns Universitet, 2002, pp. 53–60.

[21] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, 1998. doi: http://dx.doi.org/10.2307/2669847

[22] S. Ceranka and M. Niedzwiecki, "Application of particle filtering in navigation system for the blind," in *Proceedings of the IEEE 17th International Symposium on Signal Processing and its Applications*, 2003. doi: http://dx.doi.org/10.1109/isspa.2003.1224922 pp. 495–498.

[23] D. A. Klein, "Bobot - bonn benchmark on tracking," 2010. [Online]. Available: http://www.iai.uni-bonn.de/~kleind/tracking/

[24] M. Owczarek, "Pedestrian tracking in video sequences: Camshift-based vs. particle filter-based approach," 2015, visited on 2015-05-05. [Online]. Available: https://vimeo.com/mateuszowczarek/pf2014b

[25] ——, "Color-based particle filter pedestrian tracker: different number of particles," 2015, visited on 2015-05-05. [Online]. Available: https://vimeo.com/mateuszowczarek/pf2014a