ECSE 626 Final Project Report
Robust Tracking-by-Detection using a Detector Confidence Particle Filter

Grant Zhao
260563769
McGill University
Montreal, QC, Canada
nan.zhao2@mail.mcgill.ca

Abstract

The paper Robust Tracking Robust Tracking-by-Detection using a Detector Confidence Particle Filter by Breitenstein et. al.[1] is one of the early papers that combines particle filter with detection based multiple object tracking paradigm. The goal of this project is to reproduce the model and demonstrate its effectiveness, then try to improve the model proposed in that paper.

1. Introduction

1.1. Overview

In the field of computer vision, multiple object tracking (MOT) is one of the most important topics because of its wide set of applications, for example, sports analytics, video surveillance and self-driving cars. The multiple object tracking problem can be formulated as a multi-variable estimation problem [2]. Given a sequence of images that has multiple objects in then, for each object, its state in each frame can be represented as the size and position of its bounding box. In the whole image sequence, each object will have a sequence of state in the frames that it is in the field of view. Finding the optimal state sequence for each object in the goal of multiple object tracking.

MOT problem is an ill-posed inverse computer vision problem, the exact solution might not always exist. There are several key challenges in solving the multiple object tracking problem. Firstly, occlusion between tracking objects or between tracking object and obstacles (e.g. wall, a street sign, etc.). As shown in fig 1.1.a, the tracker fails to track two occluding persons in the left bottom corner of the image. Secondly, tracking objects can have a similar appearance, it is hard to distinguish between objects, for example, as shown in fig 1.1.b hockey players in each team wear the same colored jersey. Thirdly, irregular movement of objects will also deteriorate the tracker performance, thus the accuracy of the modeling of the object's motion is also crucial to the tracker performance.

Fig 1.1.a Fig 1.1.b

Fig 1.1.a: tracker missed the occluding person
Fig 1.1.b: the similar appearance of hockey players cause trouble to the tracker.

1.2. Related work

Throughout the years, many different approaches have been developed to solve the multiple object tracking problem. Due to the increase of popularity in the field of deep learning, the most common choice of handling sequential data is the recurrent neural network (LSTM), for example, the work Sadeghian et. al. [3] presented in ICCV 2017, they presented a structure of recurrent neural networks that is able to encode long-term temporal dependencies across multiple cues (appearance, motion, and interaction) [3]. Furthermore, graphically model is also a popular choice when it comes to offline multiple object tracking. For example, the work by Tang et. al., in which they proposed to cast multi-person tracking as the minimum cost lifted multi-cut problem [4]. There are also plenty of models that are using probabilistic inference [2], for example, Kalman filter, Extended Kalman filter, and Particle Filter.

All the previous works mentioned share one similarity which is they are detection-based methods. As shown in Fig 1.1.2, for detection based models, the objects are first detected using a multiple object detector, and the detections will be linked into trajectories by the tracker[2]. The detection driven tracker model is the most popular choice because the detection free model will require manual initialization of the objects, and most modern tracking applications require the tracker to be fully automatic.

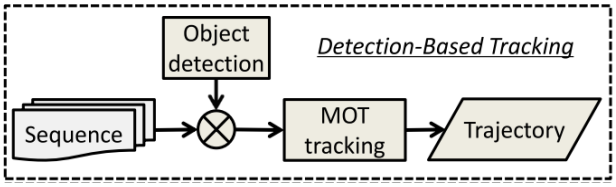

Fig 1.1.2 Detection based tracking procedure flow[2]

The goal of this project was to implement, evaluate and improve one of the works that employs a particle filter framework to solve the multiple object tracking problem by Breitenstein et. al. Section 2 will explain the theory of particle filter and how is it being applied to the multiple objects tracking in detail. Section 3 will present the methodologies using in this project. The experimental setup, evaluation matrix and the result of the will be presented in section 4. In section 5, the advantages and disadvantages of the particle filter tracker and possible improvements will be analyzed.

2. Theory

2.1. Particle Filter

The objective of a particle filter is to compute the posterior distributions of the state of some Markov process. The Particle filter uses a set of particles to represent the posterior distributions of a random process, and the density of the particles reflects the posterior's probability density distribution. The calculation of the current state posterior can be found in equation 1 below:

$$\mathrm{p}(x_t|y_{0:t}) = \frac{\mathrm{p}(y_t|x_t)p\ (x_t|y_{0:t-1})}{\mathrm{p}(y_t|y_{0:t-1})} \quad [5]\ (1)$$

where c denotes state and y denote observation. $\mathrm{p}(y_t|x_t)$ is the observation likelihood term which is the probability of getting the current observation given the current state. $p\ (x_t|y_{0:t-1})$ is the prediction about the current state given previous observations, it can be calculated by multiplying the transition probability $\mathrm{p}(x_t|x_{t-1})$ to the past state posterior $p\ (x_{t-1}|y_{0:t-1})$ then marginalizing x_{t-1} as shown in the equation 2 below.

$$\mathrm{p}(x_t|y_{0:t-1}) = \int \mathrm{p}(x_t|x_{t-1})p\ (x_{t-1}|y_{0:t-1})d\,x_{t-1} \quad [5]\ (2)$$

The probability $\mathrm{p}(y_t|y_{0:t-1})$ is a constant among particles and can be calculated using equation 3:

$$\mathrm{p}(y_t|y_{0:t-1}) = \int \mathrm{p}(x_t|y_{0:t-1})\ p\ (x_t|x_{t-1})\ p\ (y_t|x_t)\ d\,x_{t-1} \quad [5]\ (3)$$

2.2. Particle Filter and Multiple Object Tracking

To use particle filters to solve multiple object tracking problem, first, we need to formulate the problem as a Markov decision process, which means the state of the object in the current frame only depends on the object state in the last frame. In the Multiple Object Tracking problem setting, the object state is often defined as the object size and position, and the observations are detection bounding boxes [1].

An individual particle filter is assigned to each individual identity, it serves as the tracker of that identity. Since the model is detection driven, the bounding boxes (observations) output from the detector should be assigned to the individual particle filter in a way that maximizes the posterior of the particle filter $\mathrm{p}(x_t|y_{0:t})$. Usually, this is accomplished by calculate a similarity score for each detection particle filter pair, and assign the detection to particle filters using greedy algorithm [1] or Hungarian algorithm [6]. Most of the contributions on the field of particle filter multiple object tracking are made on the observation model. The reason is that the function being used to calculate the observation likelihood provides a platform that can incorporate multiple high-performance observation models to improve performance. For example, object interaction model, appearance model, and detector model.

The process of estimating the next state posterior probability distribution using Particle Filter in multiple object tracking problem setting can be split into three steps, as shown in the flow chart Fig 2.1.1. The first step is prediction step, given the posterior estimation of the last time step, propagate particles using the transition model $\mathrm{p}(x_t|x_{t-1})$ to get the hypothesis $\mathrm{p}(x_t|y_{0:t-1})$. In the second step, a new detection will be assigned to this particle filter (tracker), the observation likelihood $\mathrm{p}(y_t|x_t)$ measurement will be carried out for each particle, then the particles will be reweighted according to their observation likelihood. The final step is resampling, the particles will be resampled according to their weights, the new set of particles represents the posterior probability distribution of the current time step will be generated. The average of the particles' states will be the state of this track.

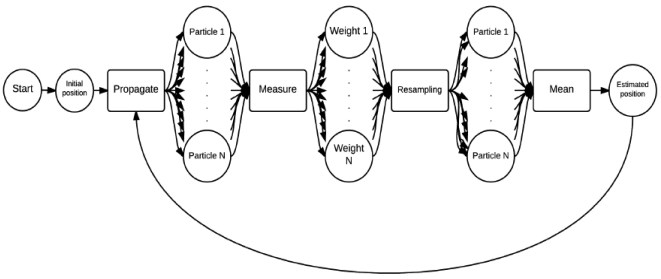

Fig 2.2.1 flow chart of a particle filter object tracker [7]

3. Methodology

The implementation of the particle filter based multiple object tracker in this project is loosely based on what was proposed in Breitenstein's original paper [1]. There are modifications being made on the detector and the appearance observation model.

3.1. Tracker Position and Size

As discussed in section 2.1, each particle in the particle filter carries the state of this particle, the particle position, and x, y directional speed is incorporated in the state. The position of the tracker is the strongest mode of the positions of all the particles in the particle filter [1], where the size of the track is set to the running average of the size of last four assigned detections. The reason that object size is not incorporated in the particle state is to reduce the dimension of the state space. According to Breitenstein et. al., to estimate a larger state space (position and size compare to position only), the number of particles required will increase exponentially [1].

3.2. Data Association

After running the object detector on each frame, the individual assignment of detection to their corresponding particle filter will be determined by a similarity score function. The original paper proposed a similarity function that is calculated between every detection and particle filter (tracker) pair. And the detections are assigned to the particle filters greedily according to the similarity score [1]. Statistically, the greedy assignment is in principle is the max a posteriori approach, such that the detection is assigned to the track that maximizes the posterior probability of the current state. The similarity score is calculated as using equation 4 shown below:

$$s(tr, d) = g(tr, d) \cdot (c_{tr}(d) + \alpha \cdot \sum_{p \in tr}^{N} p_{\mathcal{N}}(d - p))$$

[1] (4)

In which $c_{tr}(d)$ denotes the appearance comparison between the detection bounding box and the tracker (particle filter), the $p_N(d-p)$ term is comparing the position between each particle in the particle filter and the detection. g(tr,d) is the gating function calculated using equation 5 shown below:

$$g(tr, d) = p(size_d|tr)p(pos_d|tr)$$

[1] (5)

where $p(size_d|tr)$ and $p(pos_d|tr)$ are the comparisons of the size and position between the detection and the track respectively. Gaussian filters are used explicitly when calculating the comparison scores, take the size comparison as an example, the intersection over union (IOU) between the bounding box of the detection and the track is calculated, then the value (less than or equal to 1) will be plugged into a gaussian distribution with mean equals to 1 to get the size comparison score.

3.3. Likelihood calculation

As discussed in section 2, the observation likelihood function is where we can incorporate multiple observation models and combine their person reidentification capabilities. In the original work, Breitenstein et. al.[1] combined three observation models when calculating the observation likelihood weight for each particle. The three observation models being used are position, detection confidence, and appearance. The observation likelihood was calculated using equation 6 below:

$$w_{tr,p} = p(y_t|x_t^{(i)}) = \underbrace{\beta \cdot \mathcal{I}(tr) \cdot p_{\mathcal{N}}(p - d^*)}_{\text{detection}} + \underbrace{\gamma \cdot d_c(p) \cdot p_o(tr)}_{\text{det. confidence density}} + \underbrace{\eta \cdot c_{tr}(p)}_{\text{classifier}}$$

[1] (6)

Where the first term compares the position of the detection and the particle using a zero mean Gaussian. The second term is the HOG detector's SVM confidence output before non-maximum suppression [1]. In our implementation it is the detection confidence from the deep-learning based object detector, this will be discussed in detail in section 3.4. And the third term is the appearance comparison between the particle and the detection bounding box, in the original work [1], it is a boosted binary classifier's output. In our implementation, the third term is a two-part HSV histogram comparison score, which will be discussed in section 3.5. The integration of the detection confidence term in the likelihood computation is one of the major contributions of the original paper [1]. This enables the tracker to monitor the objector's continuous detection confidence and impart the tracking approaches with more flexibility to handle difficult situations where the detector output is unreliable [1].

3.4. Multiple Object Detector

A sliding window based HOG detector was used for object detection in Breitenstein's original paper [1]. Recent years, there has been tremendous improvement in deep learning-based object detectors [8], which was not available when Breitenstein et. al. was completing the paper. Moreover, the deep-learning based approach also outputs a confidence score for each bounding box which was needed in the likelihood weight calculation. Thus, both the HOG based detector and the deep-learning based detector were tested and evaluated in this project.

The detector used by the original paper was experimented, the HOG based pedestrian detector from OpenCV was used without fine-tuning. A visualization of the detection result can be found in Fig 3.4.1.b. It is shown

that the bounding boxes from the HOG detector are not precise. The reasons are the unavailability of their source code and lack of mentioning the hyperparameter setting of the detector, which makes reproducing the detection performance very difficult.

According to Huang et. al. [8], the Faster RCNN object detector with low proposal numbers has the best speed/accuracy trade-off among the deep-learning based detectors. Moreover, there are plenty of state-of-the-art pre-trained object detection models that are available online. One of the best performing pre-trained models from tensorflow object detection API [8] is the Faster RCNN Nas Low Proposals model, it was experimented and integrated into the tracker without fine-tuning. The visualization of the detection performance can be found in Fig 3.4.1.b.

The selected model itself was pretrained on the COCO dataset [9] that contains 91 class of objects, so it is capable of detect 91 different kinds of objects. In this project, only the person class bounding box was extracted as shown in fig 3.4.2.

Fig 3.4.1.a Fig 3.4.1.b

Fig 3.4.1.a: lack of detection accuracy of the HOG based detector

Fig 3.4.1.b: accurate detection output from the faster R-CNN detector

Fig 3.4.2: Default output of the faster RCNN detector from tensorFlow, only the green bounding boxes (person)will be extracted

3.5. Appearance Model

In the original work [1], a boosted binary object classifier that was trained online for each object, and it was used to compare the appearance between the detection and particles [1]. Their boosted binary classifier can classify whether the input image is from the same class it was trained on or not. Based on their experiment, they selected a classifier with 50 Red-Green-Intensity(RGI) and Local Binary Pattern (LBP) features with 3 bins per color channel [1]. However, in this project due to the time constraint, a simpler appearance comparison model was used. The model is based on the appearance model that was presented by Okuma et. al.[10]. The model itself is a two-part Hue-Saturation-Value (HSV) color histogram model shown in Fig 3.5.1. The two bounding boxes being compared are divided into two subregions horizontally, and the HSV histograms of the subregions will be compared individually using the Bhattacharyya distance. Because of the upper body and the lower body of a pedestrian generally have difference appearances, this two-part HSV histogram model can preserve some of the special information of the original bounding box. HSV decouples the intensity from color, thus the HSV histogram is reasonably insensitive to illumination changes [10], which makes the HSV histograms an ideal choice of appearance model to the multiple objects tracking application.

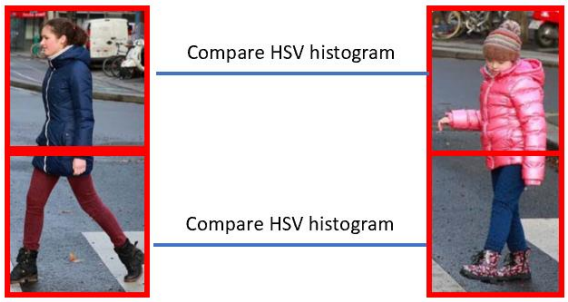

Fig 3.5.1: Two-part HSV histogram appearance model

3.6. Motion Model

From section 2, we know that motion model $p(x_t|x_{t-1})$ was used to make predictions about the current state given previous observations. A motion model that mimics the true movement of the object being tracked is vital to the success of the particle filter object tracking. In this project, the scenes which pedestrians are moving slowly are used for evaluation, thus the simple motion model from the original paper was sufficient. In this motion model, the magnitude and direction of the objects are assumed to be mostly constant with some random noises. As shown in the formula below, the position and speed of the object's current state are calculated using equation 7:

$$\begin{aligned}(x,y)_t &= (x,y)_{t-1} + (u,v)_{t-1} \cdot \Delta t + \varepsilon_{(x,y)} \\ (u,v)_t &= (u,v)_{t-1} + \varepsilon_{(u,v)}.\end{aligned}$$

[1] (7)

where ε are zero-mean Gaussian noise. And the variance of the position noise increases as the size of the object increases, and the variance of the velocity noise decreases as the number of successfully tracked frames for the object increases.

3.7. Track Initialization and Termination

In this project, to avoid persistent false positives caused by similar looking structures like trees and traffic signs, an assumption is made such that pedestrians can only walk into the field of view from the boundaries of the frame [1].

Recall section 3.2, after the greedy assignment of the detections to the trackers (particle filters), it is often that some detections or identities are not paired. Some of the left detections are false positives and some of the detections are new identities, similarly some of the left identities will be false negatives some of them needs to be terminated. The new identity initialization rule is that if the unassigned detection resides at the boundaries of the frame and has high detection confidence, it is considered as a new pedestrian walked into the frame and a new tracker will be initialized. The unassigned detections appear in the middle of the frame will be ignored.

Similarly, for the unassigned identities, if the identity resides around the boundaries of the frame, it is considered that this person has walked out of the frame, and the tracker will be terminated. However, if a tracker has not been associated with any detection for four consecutive frames, even if it is at the center of the frame, it will still be terminated.

4. Result

4.1. Tracking Evaluation

Because of the increase in popularity of the multiple objects tracking benchmarks like the MOT challenge, the evaluation criteria has been standardized [11]. The multiple object tracking accuracy (MOTA) score and the multiple object tracking position (MOTP) score has become two of the most popular measures when evaluating a tracker [11].

The MOTA score is a comprehensive measurement of the tracker's performance that combines three sources of errors, including the false positives (FP), false negative (FN) and ID switches (IDSW). The false negative occurs if the person is annotated but not detected or the intersection over the union between the ground truth bounding box and the tracker output bounding box is less than 0.5. The false positive is the tracker output bounding box does not correspond to any of the ground truth bounding boxes. The MOTA score is calculated by the equation 8 shown below:

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{IDSW}_t)}{\sum_t \mathrm{GT}_t}$$

[11] (8)

where t is the frame index and GT id the number of ground truth. Thus, MOTA reflects the accuracy of the tracker.

The MOTP score is the average dissimilarity between all the correctly tracked objects at each frame with their corresponding ground truth bounding boxes [11]. It is calculated using equation 9 shown below:

$$\mathrm{MOTP} = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t}$$

[11] (9)

where c_t denotes the number of the corrected tracked object in frame t and $d_{t,i}$ denotes the intersection over union between the tracker output bounding box and the ground truth. Thus, MOTP gives the average bounding box overlap between the correctly tracked object and ground truth, it is a measurement of the quality of the tracking outputs.

In this project, both MOTA and MOTP scores along with the percentage of false positives, the percentage of false negatives and the number of ID switches are used to evaluate the performance of the tracker.

4.2. Experiment Setup

The original paper [1] has evaluated their model on 6 different video clips, and one of their following works has evaluated the same model on 8 different video clips [12]. In this project, the model will be only evaluated on one of those clips due to time and computational resource constraint. The selected is the PETS 2009 S2 L1[13], it is an image sequence of a sparse crowd with 769 frames. The experiment was completed on a dual-core 2.4GHz Intel Core i5-6300U CPU, the deep-learning based object detection was performed separately on a Nvidia Tesla K80 GPU with the help of google Colab[1].

4.3. Source Code Delivered

A complete system of particle filter based multiple object tracking was developed and tested end to end. As shown in figure 4.3.1 below, the system takes the image sequence as input, and the default object detector (HOG) from openCV will be performed on each frame, then the detection output is fed into the tracker and it will track the objects, and finally, the bounding boxes will be visualized on the frames in real time. The deep learning-based object detection (from TensorFlow Object detection API) was performed separately on google Colab[1] GPU runtime.

A comprehensive evaluation program was developed, it can evaluate a tracker's performance given the ground truth and the tracker output. The output measurements are MOTA, MOTP, Precision, FN, FP, and IDSW.

[1] Google Colab is a cloud service provided by google that provides GPU access to researchers for free.

Other than the object detectors, HSV histogram calculation method and comparison method, everything else is implemented from scratch.

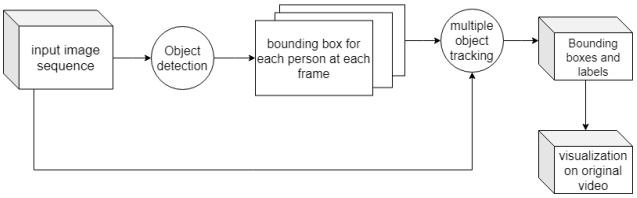

Fig 4.3.1: flow chart of the tracker system

4.4. Result Comparison

Visualizations of the particle filter object tracker output are demonstrated in Fig 4.4.1., as shown in the figure, the tracker has performed well, it even has the capability to handle occlusions (e.g. left side of the frame, two identities occluded).

As shown in table 4.4.1 below, the performance of this the model delivered in this project is better than the original work, although it is worse than some of the recent works. The MOTP score of this project is significantly higher than the original work from Breitenstein et. al. [1] and comparable to the recent work. The high MOTP score is mainly because the high-performance object detector being used in this project, it makes the correctly tracked object's bounding box very precise. The high-performance detector is likely the reason for the 2.5% improvement in the MOTA score despite the simpler appearance observation model being employed in this project.

Fig 4.4.1: a sequence that tracker handles occlusion

Model	MOTA	MOTP	precision	FN	FP	IDSW
this project	**82.4 %**	**76.3 %**	**93.7%**	**11.5 %**	**5.86 %**	**36**
Breitenstein et. al.[12]	**79.7 %**	**56.3 %**	-	-	-	-
Wang et.al. [14]	92%	65.6 %	-	4.7%	2.5%	6
Milan et. al.[15]	90.6 %	80.2 %	-	1.3%	6.8%	11

Table 4.4.1: Result Comparison

5. Discussion

5.1. Advantage

The biggest advantage of the particle filter framework is the simplicity it provides when combining the information acquired by multiple observation models to improve tracking performance. By incorporating the output of those observation models into likelihood weight calculation function. This project demonstrated the procedure of combining the knowledge from three observation models (detection position, detection confidence, and appearance). As shown in another work by Breitenstein et. al., they showed in their ablation study, the more observation being using the better the performance of the tracker [12], as shown in Fig 5.1.1.

Observation Models	MOTP	MOTA	FN	FP	ID Sw
(a): Det+Conf+Class	70.0%	72.9%	26.8%	0.3%	0
(b): Det+Conf	64.0%	54.5%	28.2%	17.2%	5
(c): Det+Class	65.0%	55.3%	31.3%	13.4%	0
(d): Conf+Class	68.0%	49.0%	37.7%	13.1%	5
(e): Det	67.0%	40.9%	30.7%	28.0%	10
(f): Conf	64.0%	47.6%	33.0%	19.1%	8
(g): Class	48.0%	25.3%	46.2%	27.9%	17
(h): N=25	63.0%	45.0%	33.4%	21.4%	6
(i): N=15	53.0%	23.4%	41.4%	34.7%	12
(j): N=10	51.0%	-5.6%	50.8%	53.9%	23
(k): N=5	40.0%	-59.4%	53.6%	104.7%	31
(l): N=1	36.0%	-104.1%	57.4%	144.8%	52
(m): $\tau = 0.5$	69.0%	60.1%	31.4%	8.4%	3
(n): $\tau = 0.2$	65.0%	32.0%	34.5%	33.2%	5

Fig 5.1.1: impact of multiple observation models [12]

The second advantage of the particle filter tracking model is the scalability. Because everything in the particle filter is parallelable, from the particles to the particle filters, this enables us to used GPU to improve the speed of the model.

The third advantage of this model is being an online model, unlike some graphical models [4] it requires no information from future frames. There are many applications that require the tracker model to be online, for example, robotic vision, autonomous driving, and visual surveillance.

5.2. Disadvantage

There are mainly two disadvantages of the particle filter tracking model. Firstly, the model has a Markovian assumption, i.e. the state of the object in the current frame only depends on the state of the object in the previous frame. However, this is generally not true in the real-world setting, furthermore, we can extract some useful information from those frames before the last frame. For example, the acceleration of the object, which cannot be estimated by only the last frame. This issue prevents us from building a more comprehensive transition model. The other disadvantage is the many assumptions being made in this project, for example, we assumed a new identity will not appear in the middle of the frame. This assumption will not generalize well given a more complex scene.

5.3. Future work

The particle-filter-based tracking model can be improved tremendously in many aspects. As mentioned in section 5.1, better performing observation models can be incorporated into the observation likelihood weight calculation. Firstly, the detector currently being used is a general-purpose object detector that is capable to detect 91 different classes of objects, however, it is not the best performing detector when it comes to single class (pedestrian) object detection. As shown in fig 3.1.b, in the bottom right corner, the detector fails to detect the two people that are occluding vertically. A deep-learning-based pedestrian detector will be a better choice for this project. Secondly, the appearance model of this project is simple but has average performance. Recently, the deep-learning based person re-identification (Re-ID) model has gained popularity in the field of pedestrian tracking. The goal of the Re-ID model is to determine whether the person in two bounding boxes are the same person. Some datasets dedicated to training and evaluating Re-ID models were developed, for example, MSMT17 [16], Market [17], and DukeMTMC-reID [18]. In table 4.4.1, one of the recent models was a particle-filter-based model proposed by Wang et. al., in which they used a deep learning based Re-ID model as their appearance model [14]. Their model has reached state-of-the-art performance in terms of the MOTA score on that clip.

The other area this model can improve is to use a better transition model that estimates the next state pedestrian position better. Currently, the motion model being used is a constant velocity model, however, pedestrians tend to change speed (magnitude and direction) from time to time. If the velocity state of each particle can be determined for each frame, the tracker performance can be improved.

6. Conclusion

The objective of the project was to implement, evaluate and improve the particle-filter-based object tracking model proposed by Breitenstein et. al. [1]. The results were better than the original paper after integrating the high-performance faster R-CNN object detector. This project demonstrates the effectiveness of a particle-filter-based tracker when combining multiple observation models. The performance of the model improves greatly when a better detector was incorporated into the observation likelihood calculation. The tracker model produced promising result on the PETS 2009 S2L1 clip [13]. The Markovian assumption was one of the biggest disadvantages of the model. To improve the tracking performance of this model, more advanced observation models like deep ReID model should be used. In conclusion, the particle filter method offers a framework that combines the multiple observation models to improve tracking performance effectively in a trivial manner.

References

[1] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 1515–1522.

[2] W. Luo *et al.*, "Multiple Object Tracking: A Literature Review," Sep. 2014.

[3] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking the Untrackable: Learning to Track Multiple Cues with Long-Term Dependencies," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, vol. 2017–Octob, pp. 300–311.

[4] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person re-identification," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017–Janua, pp. 3701–3710.

[5] A. Doucet and A. M. Johansen, "A Tutorial on Particle Filtering and Smoothing: Fifteen years later."

[6] J. Xing, H. Ai, and S. Lao, "Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses," *2009 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work. CVPR Work. 2009*, vol. 2009 IEEE, pp. 1200–1207, 2009.

[7] A. A. Muhammad, "Multiple Object Tracking Using Particle Filter," Bonn, 2016.

[8] J. Huang *et al.*, "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3296–3297.

[9] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," May 2014.

[10] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe, "A Boosted Particle Filter: Multitarget Detection and Tracking," in *European Conference on Computer Vision*, 2004.

[11] A. Milan, L. Leal-Taixé, T. Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A Benchmark for Multi-Object Tracking."

[12] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-

Meier, and L. Van Gool, “Online Multiperson Tracking-by-Detection from a Single, Uncalibrated Camera,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1820–1833, Sep. 2011.

[13] “PETS 2009.” [Online]. Available: http://www.cvg.reading.ac.uk/PETS2009/a.html. [Accessed: 12-Apr-2019].

[14] B. H. Wang, Y. Wang, K. Q. Weinberger, and M. Campbell, “Deep Person Re-identification for Probabilistic Data Association in Multiple Pedestrian Tracking,” Oct. 2018.

[15] A. Milan, S. Roth, and K. Schindler, “Continuous energy minimization for multitarget tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 58–72, 2014.

[16] L. Wei, S. Zhang, W. Gao, and Q. Tian, “Person Transfer GAN to Bridge Domain Gap for Person Re-Identification,” Nov. 2017.

[17] Y. Lin, L. Zheng, Z. Zheng, Y. Wu, and Y. Yang, “Improving Person Re-identification by Attribute and Identity Learning,” Mar. 2017.

[18] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara, and C. Tomasi, “Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking,” Sep. 2016.