Multiple Object Tracking Using Particle Filter

Muhammad Abduh Arifin

***Abstract*—This report shows a way how to tackle occlusion problem in multiple object tracking using particle filter and tracking by detection. We describe about methods that we use to address the problems. Several methods to tune the parameters are also described in this report. We evaluate our tracker and figure out the effect of number of particles, propagation noise in prediction step, and different sequences.**

I. Introduction

Multiple object tracking is one of many research topics in computer vision. Many researches have been done to improve the robustness and accuracy in tracking the objects. Tracking-by-detection is an instance of a method to track the objects. But by relying on the detection, many problems will arise when there is an occlusion on the scene. Several occlusion handling have been proposed by many researchers. For example, Breitenstein et al[3] used explicit interobject occlusion reasoning. This method uses intermediate output of object detector and then find detecotr confidence density at each particle position. Zhang et al[10] used robust part matching, and Sun et al[11] used symmetric stereo matching which uses energy minimization framework. This occlusion handling utilizes intermediate result from object detector by computing confidence density at particle position. In this report, we propose a way to tackle occlusion handling by considering the detection from previous frame. The method that we use is described in Part III. The implementation and evaluation of our algorithm are described in Part IV and Part V.

II. Theory

A. Particle Filter

Particle filter is a method to estimate the next state condition of the system, given only a few information from the state space. Generally, particle filter can be divided into three stages. The first stage is prediction. This step is executed to create hypotheses about next state. At the prediction step, particles are propagated using probability density function. Second is measurement. Given the observation model, measurement step will measure the confidence values for every propagated particle. Then the weighting function will give weighting for every particle using confidence value as weighting parameter. The last stage is resampling stage. Having weighted value for every particle, we generate new particles based on the confidentiality of the particles from previous frame. At the end, The estimated value could be obtained by taking the mean value of every particle.

B. Tracking by detection

Tracking by detection is a technique to track the object(s) by relying on the detection of the object(s). Many detection method could be use to track the object. The output from detector will be considered as an input for tracker. There are several ways to do tracking, one of them is using particle filter to localize the estimated object position.

III. Methodology

In our experiment, we are using detection maps from Hough Forest output as the input of tracker. We are using particle filter to estimate the position of the object. Beside that, we also try to tackle occlusion problem that will happen when the detector fail to detect the object because it is occluded.

A. Detector Output

The detection map from detector output is processed, so the detector will provide data of the position and bounding box size of the objects. To obtain position and bounding box size information, we normalize detection map image and the normalized detection map is thresholded using binary threshold with threshold θ_T. From the thresholded detection map, we localize the object positions by locating the blob contour using algorithm proposed by [1] and calculate the center moment of every detected contour. To eliminate false positive, another threshold θ_A is also used when calculating area of the contour. If there are areas of the contour that have the size less than θ_A, those could be ignored.

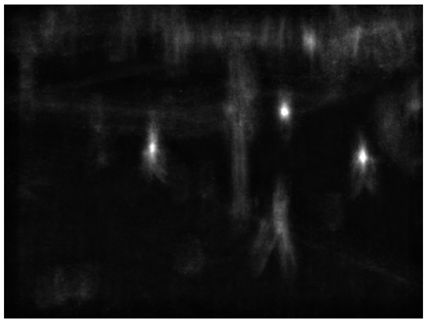


Fig. 1: Detection map from detector output

$$TI(x,y) = \begin{cases} 255 & \text{if } NI(x,y) > \theta_T \\ 0 & \text{Otherwise} \end{cases}$$

The center moment is considered to be the object position and bounding box size is the area of the contour. Position of the objects and size of bounding boxes are passed to tracker to be estimated.

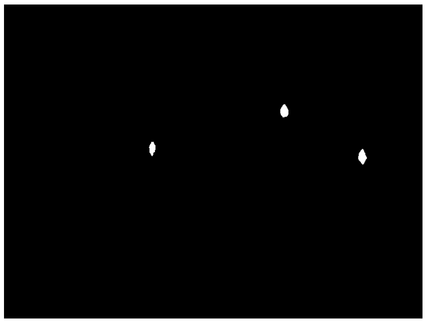


Fig. 2: Thresholded detection map

B. Particle Filter

Tracker receives data position and bounding box size of the object from detector. Our tracking algorithm uses data from previous frame to estimate position of tracked object in the current frame. We use several models to deal with particles propagation and observation. We can divide particle filter process into three stages: prediction, measurement, and resampling. The illustration of particle filter with flow diagram is depicted in the Fig. 3.

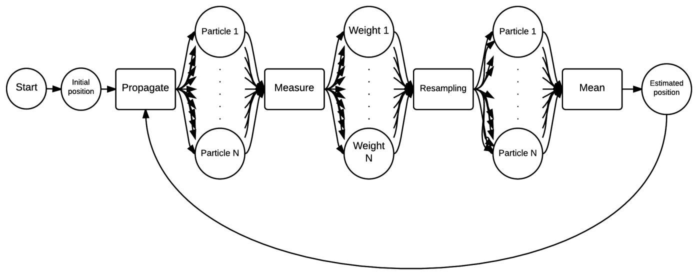


Fig. 3: Flow diagram of Particle Filter

1) Prediction step: In prediction step, we set N particles on initial position. Then we propagate all of particles using a constant velocity model. With $p(x, y)$ is position of particle, $v(x, y)$ is velocity for particle to propagate, $\epsilon_p(x, y)$ and $\epsilon_v(x, y)$ are noise of position and velocity respectively for propagating the particle, and $p^*(x, y)$ and $v^*(x, y)$ are new position and velocity of particle.

$$p^*(x, y) = p(x, y) + v(x, y) + \epsilon_p(x, y)$$

$$v^*(x, y) = v(x, y) + \epsilon_v(x, y)$$

In constant velocity model, we add noises for particle position $\epsilon_p(x, y)$ and for velocity $\epsilon_v(x, y)$. The value of ϵ_p and ϵ_v are generated from a zero mean normal distribution with variance σ_p^2 and σ_v^2 for particle position and velocity. $p^*(x, y)$ and $v^*(x, y)$ are the propagated particle and velocity of current frame respectively.

2) Measurement step: The propagated N particles will be assessed by observation model. Our observation model consists of detection term and color appearance term. The output of this measurement using observation model is confidence value which can be considered as weight value. With $w_d(i)$ is a confidential value (weight) of detection model for particle i, w_c is a confidential value of color appearance model for particle i, and α is a constant to weight both of confidential values.

$$w(i) = (1 - \alpha) \cdot w_d(i) + \alpha \cdot w_c(i)$$

$$0 < i \le N, i \in Z$$

Detection term. For detection term, we need the detection data of the current frame from detector and measure the weight by using gaussian function of the distance between current detection and propagated particles.

$$w_d(i) = \frac{1}{\sqrt{2\pi\sigma_m^2}} \exp - \left(\frac{(p^*(x) - d(x))^2 + (p^*(y) - d(y))^2}{2\sigma_m^2} \right)$$

Color appearance term. In the initial frame or when the objects enter the scene at the first time, we obtain the objects appearance information by dividing bounding boxes vertically into three parts. At each part, we calculate its histogram and store it as histogram template for a certain part of object. Beside the histograms of template objects, the histograms of tracked object in previous frame also be calculated. Histograms of template objects are interpolated with histogram from detected object in the previous frame.

$$\bar{H}_I^k = (1 - h_c) \cdot \bar{H}_T^k + h_c \cdot \bar{H}_P^k$$

Every bounding box for the tracker depends on the contour area from detector. In this case, if the bounding box less than certain threshold, default size of bounding box $width_d$ and $height_d$ are used. After all, we obtain the histograms for every propagated particle p^* and calculate distances between them and the interpolated histograms using Bhattacharyya distance.

$$H_{P*}^k = \text{Histograms information of propagated particles}$$

$$d(\bar{H}_I^k, \bar{H}_{P*}^k, i) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_I^k \bar{H}_{P*}^k N_H^2}} \sum_n \sqrt{\bar{H}_I^k(n) \cdot \bar{H}_{P*}^k(n)}}$$

$$w_c(i) = 1.0 - d(\bar{H}_I^k, \bar{H}_{P*}^k, i)$$

The most similiar area of particles are weighted bigger than the others and total weighting information $w(i)$ are calculated using equation above.

3) Resampling step: . The weighted propagated particles are finally resampled. In resampling step, the new collection of particles are regenerated based on confidential value from previous particles. The new generated particles will approximate confidentiality distribution of new position. There are many method to resample particles, one of them is systematic resampling method, which we use in this experiment.

The idea behind systematic resampling is comparing a certain ordered random number with cumulative sum of data. If the value of random data for a certain index is less than cumulative sum value of weight data for an index, the index of cumulative sum data will be taken and continue with next index of random data. Otherwise, increase the index of cumulative sum data.

```
Require: $w$ = weight data for every particle
  $w_{norm} \leftarrow$ Normalize($w$)
  $Q \leftarrow$ CumulativeSum($w_norm$)
  for all $i \in N_{particles}$ do
    $T_i \leftarrow (i+$RandomReal$(0,1))/N_{particles}$
  end for$i = 0, j = 0$
  while $i < N_{particles}$ do
    if $T_i < Q_j$ and $Q_j > \theta_{res}$ then
      $Index_i = j$
      $i = i + 1$
    else
      $j = j + 1$
      if $j \geq N_{particles}$ then
        $j = N_{particles} - 1$
      end if
    end if
  end while
  return $Index$
```

Beside using cumulative sum value to resample the data, we also could use threshold θ_{res} to obtain our preferred behavior of resampled data. Finally, we calculate mean value of resampled particles to obtain the predicted object position.

C. Detection-Tracker association

Detected objects on the scene will be associated with the trackers in every frame. We do the association using Hungarian Algorithm by matching tracker and detection that have the smallest distance. Beside that, there are also some conditions to determine when the tracker is initiated or terminated. By giving offset around the scene to create the boundary, a tracker will be initiated if there is a detection on the boundary but there is no matching tracker around it in the previous frame. On the other hand, if there is a detection around the boundary in the previous frame, but in the current frame only tracker without matching detection left, this is the condition for termination. The tracker then will be deleted. In addition to detection-tracker association, index for every tracker is also maintained. The indices for both of trackers and detections are preserved to track the the detected objects after all.

D. Occlusion handling

Occlusion is a crucial part when we are going to track the object, especially using tracking by detection. If there is an occlusion, detector will not detect the present of the object on the scene. To tackle this problem, we propose a method by considering the informations of previous detected object and associate those informations with the information of trackers. Furthermore, the information about detections from previous frame are also associated with the information about detections from current frame. This would give the result about how is the occlusion happen. We distinguished two types of occlusion regarding what is the occluder. First type of occlusion is if the occluder is a tracked object and the second one is if the occluder is a static object on the scene. We also can create the condition when the object is entering occlusion state and when the object is exiting the occlusion state.

1) Tracked object occluder: In the condition where a tracked object occludes other tracked object, we get the information from the first time the occludee is occluded. Because occlusion will happen if there is more trackers than detections, it means there are one or more trackers that do not have matching pair. With this condition, we can check the distance between previous detections and current detections. The occluded object is the one which appear in previous detection but does not appear in current detection. By using this kind of matching information, we compare the previous position of occluded object with the position of trackers. Then, we obtain the information about which tracker has the smallest distance with the occluded object. If the distance between the tracker with smallest distance and occluded object is less than a certain threshold θ_{dist}, we conclude that the tracker is occluding the occluded object. After all, we maintain the pair of index of occluder and occluded object. During the runtime, if the

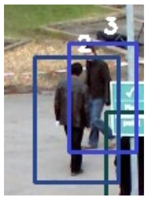


Fig. 4: Object occluded by another tracked object

number of currents detection is bigger than number of previous detections, this is the condition when the occluded object is exiting occlusion state. In this condition, we again take the non-matching detection between current detection and previous detection. The distance between this non-matching detection position and the occluded tracker is obtained. If the non-matching detection match with one of occluded tracker, we check the occluded tracker whether it has the occluder and occlued object pair index or not. If it is so, we know that the new detection (non-matching detection) is previously occluded by the tracker then we can restore the index information.

2) Static object occluder: It is slightly different with tracker object occluder condition, static object occluder has no information about occluder. Therefore, it cannot store pair index of occluder and occluded object as processed in tracker object occluder condition. Also, we cannot sure how long the object is occluded and where the occluded object would reappear. If the occluder is so big or the occluded object is occluded within a long time, particles of the occluded tracker will spread out from its initial position on the scene. In this condition, detection term will not reliable anymore because there is no detection. To tackle this issue, we try to change the weighting constant in weighting function so it could change adaptively depending on whether the tracker in the occlusion state or not. If the tracker is in the occlusion state, weighted value for color appearance term will be higher than detection, so the tracker

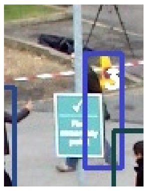


Fig. 5: Object occluded by a static object

will be more relying on color measurement.

IV. Implementation

The program run in the system with these specification: 3.5 GHz Intel Core i7, 4 GB 1333 MHz DDR3, Ubuntu 14.04 LTS. The implementation is created using C++ and OpenCV 3.0.0 library. There are many parameters in the implementation and all of them are obtained by tuning them so the best result is achieved. For detector threshold, the value of θ_T is 160 and threshold area θ_A is 30. In prediction step, we use σ_p^2 is 7.0 and σ_v^2 is 1.0. For color appearance term in measurement step, α is 0.4. Threshold for bounding box size is half of default bounding box size $\theta_{BB_{height}} = 0.5 \cdot height_{default}$ and $\theta_{BB_{width}} = 0.5 \cdot width_{default}$ and weighting constant for histogram h_c is 0.3. In detection term, there is parameter of variance σ_m^2 which has value 0.5. In occlusion handling, we set θ_{dist} with 20.0. Then, we change the α in the measurement model with 0.6. The implementation will first localize the objects from detection maps and then run the tracking algorithm with information from previous and current detection maps. Once it has done, the trackers position information will be written in the directory for each frame. For example, if the tracking algorithm is executed on frame 0, then trackers information will be written in the file frame_0000.txt. The data format starting from the first column is, frame number, x coordinate (top-left), y coordinate (top-left), width, height, tracker index. These trackers information will be compared with ground truth information to obtain the accuracy and precision of the trackers. The tracking result is also stored as a sequence of images. There is also a parameter configuration file that could be used to run experiments with various value of parameters. Each experiment could be run more than one time with the same value by changing the value of iteration variable in the main function.

V. Evaluation

Our tracking algorithm is evaluated using quantitative evaluation criteria from MOTChallange 2015 [2]. We use Multi Object Tracking Accuracy (MOTA) to evaluate how accurate our tracker to track the objects and Multi Object Tracking Precision (MOTP) to evaluate how precise every successful tracked objects. MOTA and MOTP based on number of TP (true positive), FP (false positive), FN (false negative), IDSW (ID switch) values, and compare them with the ground truth data. Figure 6 shows the occurrence of true positive, false positive, and false negative. True positive happens if a person is tracked correctly (bounding box number 3). False positive happens if the tracker track something but the person (bounding box number 4). False negative happens when there is a person, but the tracker fails to detect (a person with the blue jacket). IDSW happens if the index between two persons are switched. MOTA and MOTP are calculated with these formula.

$$MOTA = 1 - \frac{\sum_t (FP_t + FN_t + IDSW_t)}{\sum_t (GT_t)}$$

$$MOTP = \frac{\sum_{t,i} (d_{t,i})}{\sum_{t,i} (TP_t)}$$

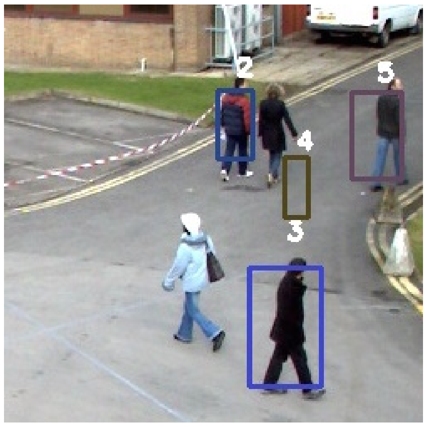


Fig. 6: TP, FP, and FN on the image

Where t is the frame number and GT is number of ground truth objects. In the first experiment, we evaluate the effect of using various number of particles to the accuracy and precision by comparing our tracking algorithm using detector with using ground truth information. To test our algorithm, we use PETS2009 S2.L1 datasets and only considering first 200 frames.

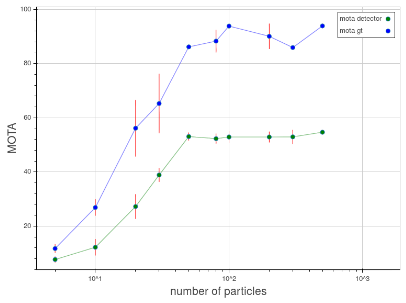


Fig. 7: MOTA between algorithm using detector and ground truth with various number of particles

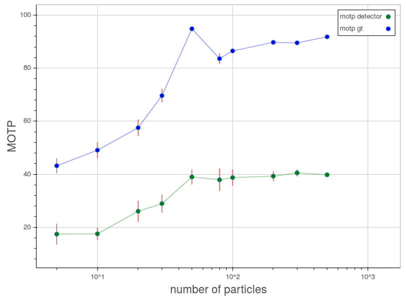


Fig. 8: MOTP between algorithm using detector and ground truth with various number of particles

From this observation, we know that the graph is asymptotically constant starting around 50 particles. We then do other experiments based on this observation with only 50 particles.

We figured out the effect of changing σ_p with σ_v constant 1.0 with intention that we did not want to perturb the particles radically by changing σ_v. We test with various σ_p value from 1.0 until 100.0. The graph in Fig. 9 and 10 depicts the various σ_p noise in prediction step. The best result is when σ_p is 10.0. Figures 9 and 10 show us that increasing sigma parameter

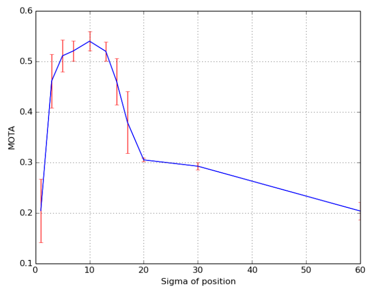


Fig. 9: MOTA with various value of σ_p

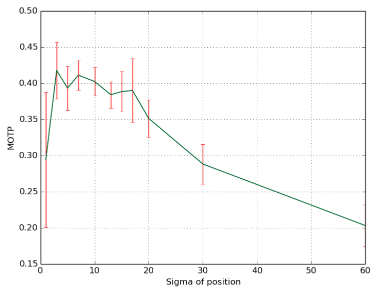
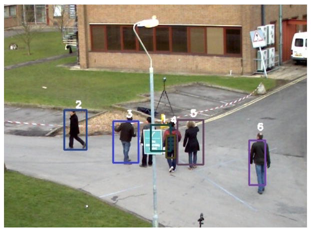


Fig. 10: MOTP with various value of σ_p

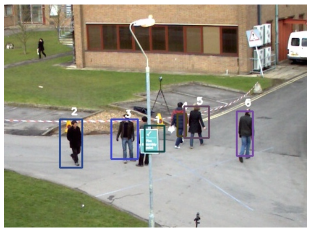
for particle to move is really affecting the outcome of particle filter. The σ_p value should be set correctly in order to obtain the more accurate result. With PETS09-S2L1 sequence, we get MOTA more than 50% with the sigma starting from 5.0 until 15.0. It means the tracker has very narrow range comparing with maximum tested σ_p value = 100.0. On the Fig. 12 and Fig. 11, show the result of tracked objects when we run the algorithm with occlusion handling and without occlusion handling for frame 71, 81, 91, 101.

After having all of the parameters, we compare the accuracy between using the occlusion handling and not using occlusion handling. This experiment is done over 200 frames.

We also trying to evaluate our tracking algorithm by testing it with another sequences and comparing it with [7] tracker. Other sequences that we use are AVG-TownCentre and ETH-Bahnhof and we only test with the first 200 frames. AVG-TownCentre is a sequence with bigger density and higher resolution than PETS09-S1L1, the camera is static and pointed out from elevated point. While ETH-Bahnhof is a sequence


(a) Frame 71 with occlusion handling

(b) Frame 81 with occlusion handlin


(c) Frame 91 with occlusion handlin

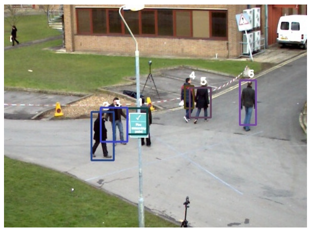
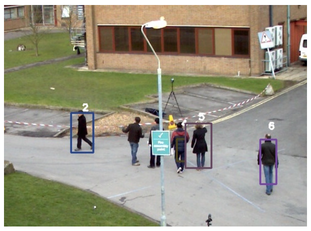
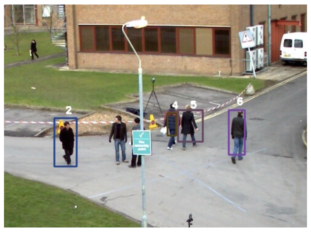

(d) Frame 101 with occlusion handlin

Fig. 11: Tracked persons in PETS2009 S2.L1 datasets with annotated box and index with occlusion handling


(a) Frame 71 without handling

(b) Frame 81 without handling


(c) Frame 91 without handling

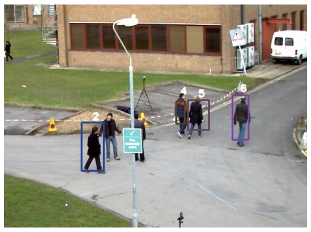

(d) Frame 101 without handling

Fig. 12: Tracked persons in PETS2009 S2.L1 datasets with annotated box and index without occlusion handling

TABLE I: Comparation of accuracy between using occlusion handling and without occlusion handling over 200 frames

	With Handling		Without Handling	
	mean	stddev	mean	stddev
MOTA	53.07%	1.467%	42.67%	2.512%
MOTP	38.97%	2.867%	26.07%	1.15%

TABLE II: Comparation of accuracy between our algorithm and hungarian tracker

		Our algorithm		Hungarian tracker
		mean	stddev	
PETS09-S2L1	MOTA	53.07%	1.467%	42.67%
	MOTP	38.97%	2.867%	26.07%
AVG-TownCentre	MOTA	47.57%	3.596%	56.19%
	MOTP	45.27%	0.779%	54.72%
ETH-Bahnhof	MOTA	24.21%	0.56%	45.48%
	MOTP	40.89%	2.19%	90.24%

with moving camera and pointed out with low angle. The result is shown on the comparation table.

The tables show us that our algorithm improves around 10% when using occlusion handling. Comparing with tracker implemented by [7], our algorithm run better with PETS09-S2L1 sequence, which is generated by the static camera on elevated position. Our tracker does not work good enough with AVG-TownCentre which has more density because the more dense the sequence, we need more robust object detector and also for occlusion handling, our tracker only consider an event when two objects occlude each other. The occlusion between more than two objects is more likely happened in the more dense sequence. Our tracker also does not work good with ETH-Bahnhof which generated by moving camera in low angle position because with moving camera, the whole scene of sequence will be drifted over the time and our constant velocity model does not accommodate this event.

VI. Conclusion

By using particle filter and only considering the information about previous frame to handle the occlusion, we can conclude that :

1) The relation between number of particle and accuracy in tracking-by-detection using particle filter is not linear and could be constant starting at a certain number. The minimum number of particle filter so the tracker runs effectively should be determined so the computation effort is reduced.
2) On the image with 768x576 pixels width, the value of σ to randomize particles position also gives the effect to the accuracy. The range of this σ value occlusion handling is really important to improve tracker accuracy. In out experiment, σ value has very narrow range to obtain MOTA more than 50%.
3) Our tracker does not work good with the kind of sequence with moving and low angle camera. It works better with the sequence with static and elevated camera position
4) Occlusion handling is very important to create the more accurate tracker. Our occlusion handling itself can improve the accuracy around 10% comparing with the tracker that only relies on detector.

References

[1] S. Suzuki and K. Abe, ”Topological Structural Analysis of Digitized Binary Images by Border Following,”*CVGIP* vol.30,no.1, pp.32-46, 1985.

[2] L. Leal-Taix e, A. Milan, I. Reid, S. Roth, and K. Schindler, ”MOTChallenge 2015:Towards a Benchmark for Multi-Target Tracking,”*arXiv*, 1504.01942, 2015.

[3] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier and L. V. Gool ”Online multi-person tracking-by-detection from a single, uncalibrated camera”, IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 9, pp.1820 -1833 2011.

[4] A. Yao , D. Uebersax , J. Gall , L. V. Gool, ”Tracking people in broadcast sports”, Proceedings of the 32nd DAGM conference on Pattern recognition, September 22-24, 2010, Darmstadt, Germany.

[5] J. Wang and Y. Yagi ”Integrating color and shape-texture features for adaptive real-time object tracking”, IEEE Trans. Image Process., vol. 17, no. 2, pp.235 -240 2008.

[6] J. D. Hol , T. B. Schn and F. Gustafsson ”On resampling algorithms for particle filters”, Proc. IEEE Nonlin. Statist. Signal Process. Workshop, pp.79 -82 2006.

[7] A. Geiger, ”Probabilistic Models for 3D Urban Scene Understanding from Movable Platforms”, KIT, 2013.

[8] A. Geiger, M. Lauer, C. Wojek, C. Stiller, R. Urtasun, ”3D Traffic Scene Understanding from Movable Platforms”, Pattern Analysis and Machine Intelligence (PAMI), 2014.

[9] H. Zhang, A. Geiger, R. Urtasun, ”Understanding High-Level Semantics by Modeling Traffic Patterns”, International Conference on Computer Vision (ICCV), 2013.

[10] T. Zhang, K. Jia, C. Xu, Y. Ma, N. Ahuja, ”Partial Occlusion Handling for Visual Tracking via Robust Part Matching”, *CVPR*, 2014.

[11] J. Sun, Y. Li, S. B. Kang, H-Y. Shum, ”Symmetric Stereo Matching for Occlusion Handling”, *CVPR*, 2005.