Improving target tracking robustness with Bayesian data fusion

Yevgeniy Reznichenko
yevgeniy.reznichenko@marquette.edu
Henry Medeiros
henry.medeiros@marquette.edu

Opus College of Engineering
Marquette University
Milwaukee, USA

Abstract

Intelligent data fusion is an active area of research. Most recent works in data fusion for object tracking employ machine learning techniques that lack flexibility due to their inability to adapt to changing conditions in the presence of limited amounts of training data. Our work explores a hierarchical Bayesian fusion approach, which aggregates information from multiple tracking algorithms into a more robust estimate and hence outperforms its constituent trackers. This adaptive and general data fusion scheme takes advantage of each tracker's local statistics and combines them using a global softened majority voting. The widespread availability of high-performance multicore processors has allowed parallel threads to run multiple trackers asynchronously, which means that the algorithm can be executed in real time as it is only limited by the slowest tracker in the ensemble. The proposed approach is corroborated and evaluated on the OTB-50 dataset.

1 Introduction

Given an initial frame and initial bounding box that delimits an object of interest in that frame, the purpose of a single-target tracking algorithm is to follow this subject through subsequent frames without being told the new bounding box that encompasses the object. Due to the very open-ended and multifaceted nature of this problem, a myriad of different trackers has been developed.

Trackers such as TLD [10], for example, use a nearest neighbor based approach to address the issue of tracking an object between frames. State-of-the-art trackers such as GOTURN [9], use deep convolutional neural networks to track objects in a search space. Due to their inherent design characteristics, the performance of these trackers differ with respect to issues such as occlusion, illumination, motion, deformation, blur and rotation. As such, different trackers have been found to perform better depending on the scenario. With this in mind, while some trackers show overall better performance in standardized datasets, there are instances in which these trackers are outperformed by less sophisticated methods due to the fact that they do not handle a particular situation well. As it stands now, the tracking problem is still a largely open research topic.

There are many benchmarks and datasets that allow the object-tracking research community to evaluate the performance of their algorithms. The Object Tracking Benchmark

(OTB) [22], for example, allows for quantitative measurement and a comparison of the performance of trackers.

Our work applies the method of hierarchical Bayesian data fusion developed by Echeverri *et al*. [4] for use in an autonomous target following with drones to object tracking. This approach has the benefit of increasing tracking robustness for situations where improvement is critical. Autonomous vehicles, for example, are high-stakes machines where human safety is a concern, and it is hence paramount that performance is optimal. Because of the flexible and adaptive nature of the framework, it can continue to be used with any tracker, and any number of trackers greater than three. Our contribution evaluates this approach on the OTB-50 dataset and compares the performance relative to the trackers that constitute this approach. Because the ultimate focus is robotics, the four fast state-of-the-art trackers we used in our evaluation are GOTURN [9], TLD [10], CMT [18] and Struck [8].

2 Previous Work

Object tracking is a key element of computer vision and a heavily researched topic. This problem has multiple applications in autonomous following drones [6], autonomous vehicles, augmented reality [5] and robotics [7, 22]. Wu *et al*. [22] and Kristan *et al*. [11] discuss at least 50 unique trackers that have been developed. A common approach is to learn something about the object's appearance, features, and motion. The tracker searches the entire image for another area that most closely fits the learned representation. Sift-points [18], nearest neighbor classifiers [10], support vector machines (SVM) [8] and neural networks [9] are just a few of the various methods that have been applied to tackle this problem.

Originally proposed by Kalal *et al*. TLD [10] uses a framework that learns and detects the object while it is not tracking. To track, TLD uses a nearest neighbor approach with multiple "experts" to both track and detect objects. This results in a good performance overall, but the tracker is weak in situations such as out-of-plane rotations and tends to perform poorly on situations with extended occlusion.

To specifically target the issue of deformation, CMT [18] was introduced. To track, CMT uses the initial bounding box to generate keypoints. The tracker attempts to follow these points frame by frame using optical flow and descriptors. Each of these points then "votes" on where they believe the center of the object to be. This tracker performs well with partial occlusion as well as scale variations but is not very robust.

Struck [8] is a recent contribution that uses an adaptive SVM approach. It is accurate and is fairly robust to challenging conditions, but it does not deal with scale variation. Hence, the target bounding box is kept at a constant size, which limits the tracker to sequences in which the distance between the target and the camera remains relatively constant.

GOTURN [9] is a fast neural network based approach proposed by Held *et al*. It is based on the feed-forward output of a pretrained convolutional neural network. Due to the nature of neural networks, this tracker is particularly powerful on sequences similar to those in which it had been trained. As with all neural network based approaches, it can suffer from overfitting or encounter novel scenarios unfamiliar to the network and generate incoherent results.

Individually these are all relatively recent and state-of-the-art real-time trackers that perform well but each has specific weaknesses in certain scenarios. With this in mind, it is necessary to then intelligently fuse the information from these trackers to mitigate one another's weaknesses.

2.1 Data Fusion

Initially, the idea of adaptive fusion began in the 1960s with the introduction of the Kalman filter (KF). Additionally, other approaches based on particle filters (PF) [5] and fuzzy logic [4] have also recently risen to prominence. Each of these methods assumes some sort of prior knowledge about the trajectory of what you are tracking. For single-object-tracking, this is generally a valid assumption given the relatively locally linear nature of the motion of most objects when observed at reasonable frame rates. Our work is most similar to the work done by Bailer [1] and Biresaw [3], which use a hierarchical state fusion interpretation. However, we use a different data fusion approach than Biresaw, and our calculations are done with a Bayesian framework in contrast to the work done by Bailer. Additionally, similar Bayesian frameworks were proposed by Yang *et al.* [23] to perform multimodal tracking for healthcare applications with the use of a different weighting scheme but a similar Bayesian fusion approach. One of the most common modern approaches to data fusion is the use of machine learning. These methods are powerful but are challenging to realistically implement due to their reliance on large amounts of training data. The approach presented in this paper forgoes these issues by using an adaptive Bayesian model that "adapts" its behavior based on the performance of the trackers. A hierarchical Bayesian data fusion approach requires only that the user provides weights to the trackers as a tuning parameter and a motion model which can be assumed.

2.2 Visual Tracking Benchmarks

To measure that the output of our data fusion method is working better than the trackers that comprise it, it is necessary to test the results on a benchmark. The OTB-50 benchmark is one of the most common tools used to evaluate these various performance scores. Originally introduced in [22], the OTB-50 benchmark has 50 specific data sequences that it uses to provide different measurements of performance on various attributes. The most general of these measurements is the success, which measures how well the tracker can track the object throughout all of the image sequences. The OTB benchmark makes it possible to quantitatively evaluate the results generated by the tracker. Other publicly available visual tracking benchmark datasets include VOT [11] and ALOV [20]. We chose OTB-50 due to its simple integration.

3 Hierarchical Bayesian Data Fusion for Target Tracking

The method proposed in this work will be henceforth referred to as hierarchical adaptive Bayesian data fusion (HAB-DF). The approach proposed is a variation of the mixture of experts framework [24]. The main difference being that the gate is substituted with a Bayesian approach. Each separate tracker s_j acts as an "expert" asynchronously when it is run through a Kalman filter. The motion model and observation models are given by

$$x(t) = Ax(t-1) + w(t) \tag{1}$$

$$y(t) = Cx(t) + v(t), \tag{2}$$

where x is the state vector and y is the observation vector. Equation 1 represents the system dynamics with A representing the transition matrix, B being the control matrix and w modeling process noise. In Equation 2, C is the observation matrix, and v is the measurement

noise. Both of the noises are assumed to be white and Gaussian with variances R_{ww} and R_{vv}. HAB-DF uses two sources of information to penalize detectors and to vote on a global output. The first mechanism through which the framework assigns weights to each of the detectors is based on the Mahalanobis distances (MD) [15] of the observations

$$M(y) = \sqrt{(y-\mu)^T \Sigma^{-1} (y-\mu)}, \tag{3}$$

As shown by Pinho *et al.* [19] the MD can be approximated by

$$M(y) = \sum_{i=1}^{N} \left(\frac{(y_i - \mu_i)^2}{\Sigma_{ii}} \right), \tag{4}$$

where y_i and μ_i are the elements of y and μ and Σ_{ii} are the diagonal elements of the innovation covariance matrix Σ.

Rather than using the MD values directly as weights in our framework, in order to soften transitions as the performance of the individual trackers fluctuates, a sigmoid function is employed

$$w_M = \frac{1}{1 + e^{(-M(y)+\xi)}}, \tag{5}$$

where ξ is a value chosen based on the χ^2 number of degrees of freedom of the system and the desired confidence level. This step takes advantage of the Bayesian framework but rather than using those statistics to correct the tracker as done in [12, 13], here they are applied as weights in a voting scheme.

The other mechanism involved in the assignment of weights to the outputs of the individual trackers is the majority voting scheme based on the pairwise Euclidean distances between the various trackers. Let x_i and x_j represent the state vectors corresponding to two different trackers. Let the Euclidean distance between x_i and x_j be

$$d_{i,j} = ||x_i - x_j||. \tag{6}$$

Then min_d represents the smallest distance between tracker i and all of the other trackers in the framework.

$$min_d = \min_{\substack{j=1,2,\ldots,n \\ j \neq i}} (d_{i,j}). \tag{7}$$

Again a softening mechanism is applied to avoid abrupt changes in the tracker operation. In this case, the method chosen was a hyperbolic tangent function

$$w_d = \omega_0 + \omega(1 + \tanh(min_d - \lambda)), \tag{8}$$

where ω_0 is the minimum value and λ represents the minimum value required for penalization to take place.

The filtered outputs of all the trackers, the bounding boxes x_j, are provided as inputs to another KF. This acts as the fusion center. The fusion center adapts itself to changes in the performance of individual trackers after each new measurement is collected by updating its measurement noise covariance according to

$$R_{\sigma\sigma}(w_d, w_M) = \Gamma w_d + \Delta w_M, \tag{9}$$

where $\Gamma = diag(\gamma_1, \gamma_2, \cdots, \gamma_n)$, $\Delta = diag(\delta_1, \delta_2, \cdots, \delta_n)$, and $diag(.)$ represents a diagonal matrix whose elements are the function parameters. γ_i and δ_i are set to 1 if there is no a priori knowledge of the system, but they can be adjusted individually if there is prior information about expected tracker performance. That is, the majority voting weight w_d and the MD weight w_M are used by the global tracker to update $R_{\sigma\sigma}$, which is then used in the global correction stage of the Kalman filter to generate the fused bounding box x_f. Eq. 9 allows the Kalman filter to have less trust in measurements that have lower weights. The Kalman filter for the fusion center is essentially identical to those applied to the individual trackers (Eqs. Eq. 1,2) but the observation matrix C reflects the fact that the observations are given by the outputs of the n trackers. Algorithm 1 summarizes the proposed approach.

Algorithm 1 HAB-DF

Input: Set of n trackers $s_j \in \mathbb{S}$, initial bounding box x_0, set V of images
Output: Bounding box s_f representing the fused output

```
Initialize all trackers s_j with x_0.
Initialize Kalman filter for each algorithm implementation s_j
Initialize Kalman filter for fused data model
while V has new images do
    Load new image
    for Each tracker s_j ∈ S do
        Generate bounding box x_j for each tracker s_j
        Apply Kalman filter (Eq. 1,2) to x_j
        Compute Mahalanobis Distance weight w_M
    end for
    Wait for all trackers s_j
    Apply majority voting to find w_d
    Calculate R_σσ according to Eq. (9)
    Apply Kalman filter (Eq. 1,2) using R_σσ as the observation covariance to generate
      the global estimate x_f
end while
```

The weighted Kalman filter data fusion step in Eq. 9 allows for flexibility. The framework can be designed to take data of any size and fuse it. Our implementation takes advantage of multithreading so that computational performance is only limited by the slowest tracker. Theoretically, we can improve our results by using as many trackers as necessary, but in real-time applications, we are limited by the computation capabilities of our processor. By taking advantage of the confidence levels generated by each Kalman filter, the tracker ensembles are able to perform a majority vote that weights the generated Bayesian statistics.

If there are multiple trackers, we can expect a fusion that weighs the trackers relative to their performance to perform better than any individual tracker. However, because we do not have the accuracies available to us a priori we estimate the weights in real-time using a Bayesian approach.

4 Results

We initially carried out separate evaluations of the four trackers that comprise our implementation of the proposed framework on the OTB-50 dataset. For STRUCK[1] and TLD[2] our results were close to the results reported in [8, 10]. For CMT and GOTURN, to the best of our knowledge, OTB-50 results are not publicly available so those had to be generated[3]. We then evaluated our approach on the same dataset with these same four trackers as part of our ensemble. In order to account for the expected overall performance of individual trackers, we adjusted the values of ω proportionally to the success rate of each of the trackers on the OTB-50 dataset. The method showed a 5.5% increase in success relative to the best tracker in the ensemble and a 2.6% increase in precision. Since our method focuses on improving the overall robustness of the trackers, we expected the larger increase in success rate. The improvement in precision demonstrates that this method is not penalized by "imprecise" trackers such as CMT. This method can be used to fuse any tracking algorithm as it only requires that a bounding box be fed into a Kalman filter.

Figure 1 details how the framework is implemented. Note that any tracking algorithm other than the four approaches illustrated in the diagram can be used in our proposed approach since it only requires the tracker bounding boxes as inputs.

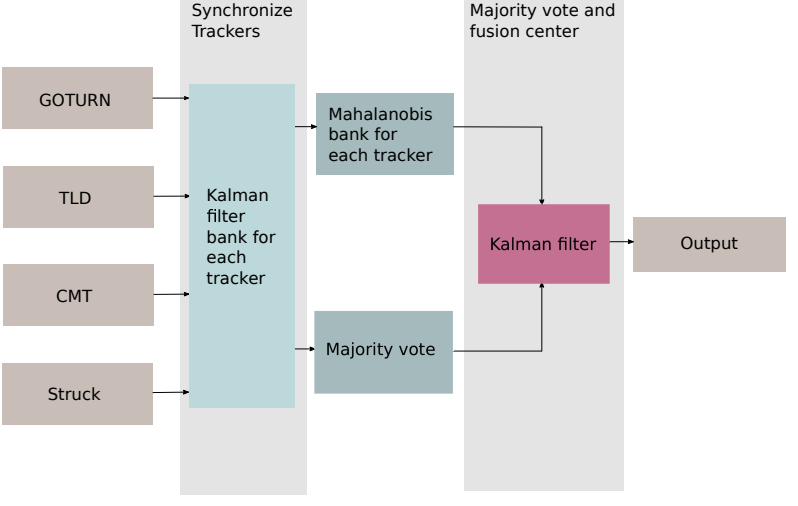

Figure 1: Schematic representation of the implementation of the proposed framework

Our results in Figure 2 illustrate the performance of the proposed approach. Our ensemble leverages the individual strengths of each tracker to obtain higher levels of robustness throughout the various datasets. Even the best tracker in our ensemble performed poorly in certain scenarios, and despite providing the largest influence on the input, the other trackers help improve performance overall.

[1] the results were obtained using source code available at https://github.com/samhare/struck

[2] the results were obtained using source code available at https://github.com/klahaag/CFtld

[3] the results for CMT and GOTURN were obtained using source code available at https://github.com/gnebehay/CMT and https://github.com/davheld/GOTURN respectively. We applied these methods "as shipped"

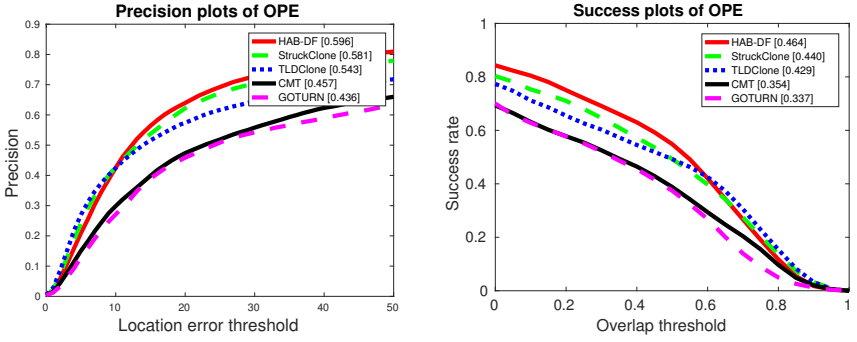

Figure 2: Results of our Tracker (HAB-DF) on OPE for OTB-50

Our ensemble is robust to failures from the lower ranked trackers such as GOTURN or CMT, and the failures of these trackers did not affect the overall performance when they occurred individually as seen in Figures 3 and 4. In the Figures our tracker is denoted by the yellow bounding box, and the color scheme for the other trackers is red/blue for TLD (blue when it is lost since TLD can make that determination), purple for GOTURN, green for CMT and white for Struck.

Figures 5 and 6 illustrate that our tracker is also robust to failures generated by the stronger trackers in the ensemble such as TLD or Struck.

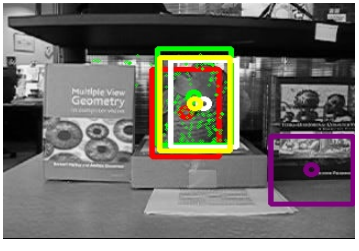

Figure 3: Robustness to failure from GOTURN

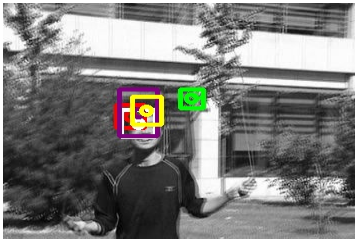

Figure 4: Robustness to failure due to CMT

A simple majority voting approach would have allowed poorly performing trackers to degrade the overall results. Our method mitigates this issue by assigning weights based on the Mahalanobis distances of the measurements generated by each tracker, and also by incorporating previous knowledge about the performance of the individual trackers. In Figures 3 and 4, it can be observed that these anomalous measurements have a minimal effect on the overall tracking result. In the first subfigure, GOTURN's distance from the other trackers assigns the tracker a high weight due to Eq (8). Hence, GOTURN has a minimal effect on the final estimate and continues to do so due to the motion model associated with the resultant tracker.

In Figure 5, the best tracker in the ensemble has failed. Because our weighing mechanism is not just a weighted voting scheme, our tracker is able to disregard the measurements from Struck. In Figure 6, we see that two trackers are lost, but our ensemble is still able to perform very well. By taking advantage of the proximity between Struck and GOTURN as dictated by Eq. (8) these trackers have a much higher influence on the output. CMT and TLD,

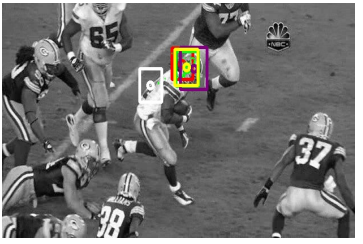

Figure 5: Robustness to failure from the strongest tracker in the ensemble

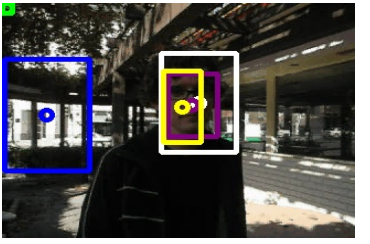

Figure 6: Robustness to the failure of multiple trackers

on the other hand, are far from any other tracker, and accrue a higher penalty and do not significantly influence the output. The weights generated by the Mahalanobis distance allow the framework to smooth out the estimate and engender a smoother and more robust output.

Besides positively impacting success and precision, the method also significantly increased the score when all the trackers had similar scores for a specific metric. This benefit is especially obvious for the OPE of out-of-plane rotations as shown in Figure 7.

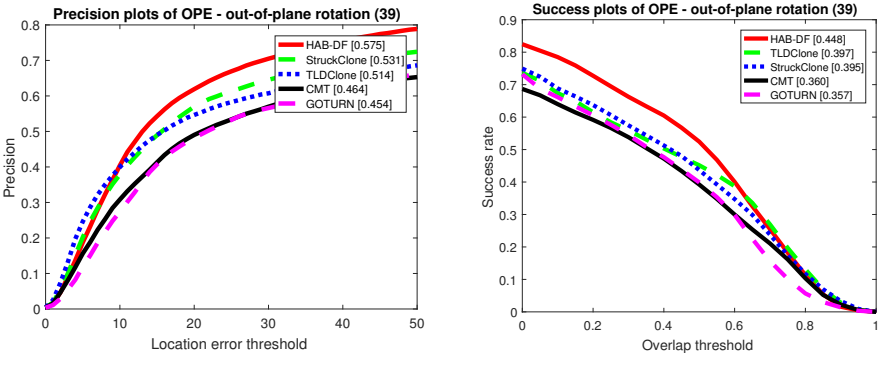

Figure 7: The increase in performance for out-of-plane rotation

Despite the significant improvement in most scenarios, in the rare situations where performance was drastically different among trackers, a decrease in performance was observed relative to the best tracker in the ensemble. When multiple trackers are significantly worse than the best trackers, the performance may decrease. This is especially obvious for the case of low resolution images in which GOTURN and CMT perform very poorly and hence debilitate the overall performance.

As Table 1 indicates, our approach improves the performance in 8 of the 12 scenarios. In the cases where the performance decreases, it is important to note the large discrepancy between the best tracker and the other trackers in the ensemble. Because the method uses the confidence generated by the Kalman Filter, when multiple trackers show poor performance, our results can be negatively affected. The improvement is most obvious and prominent when the trackers show similar performances. This problem can be mitigated by either refraining from using trackers that perform very poorly under certain scenarios or by adjusting its prior weight according to its worst-case performance.

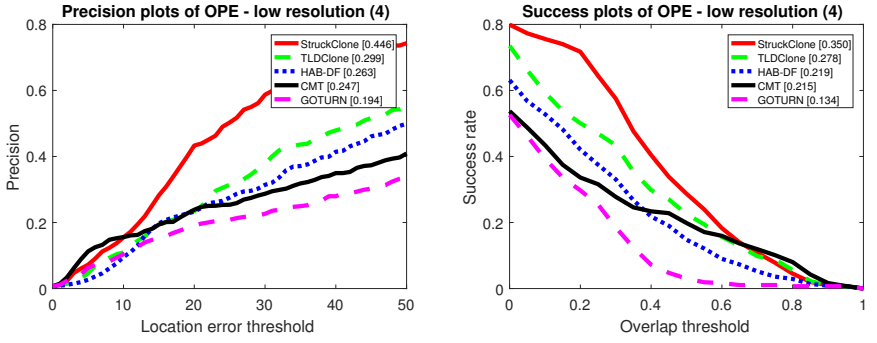

Figure 8: The decrease in performance for low resolution

Table 1: Summary of Results on OPE

Scenario	Best Tracker	Best Tracker Score (Precision/Success)	Worst Tracker	Worst Tracker Score (Precision/Success)	Fusion Score (Precision/Success)	Percent Change Relative to Best Tracker
Total	Struck	.581/.440	GOTURN	.436/.337	.596/.464	+2.581% / +5.454%
illumination	Struck	.581/.413	GOTURN	.347/.291	.524/.427	+1.158% / +3.389%
out-of-plane rotation	Struck/TLD	.531/.397	GOTURN	.454/.357	.575/.448	+8.286% / +12.846%
scale variation	Struck	.562/.386	CMT	.435/.327	.574/.432	+2.135% / +11.917%
occlusion	Struck/TLD	.521/.408	CMT/GOTURN	.404/.311	.548/.431	+5.182% / +5.637%
deformation	Struck	.516/.414	CMT	.373/.301	.568/.450	+10.078% / +8.696%
motion blur	Struck	.487/.406	GOTURN	.300/.254	.450/.366	-8.222% / -10.929%
fast motion	Struck	.520/.424	GOTURN	.410/.282	.455/.377	-14.286% / -12.467%
in-plane rotation	TLD	.552/.435	GOTURN	.332/.326	.556/.439	+0.725% / +0.920%
out of view	Struck	.482/.444	GOTURN	.332/.316	.465/.441	-3.656% / -0.680%
background clutter	Struck	.530/.429	CMT/	.341/.263	.547/.437	+3.207% / +1.864%
low resolution	Struck	.446/.350	GOTURN	.194/.134	.263/.219	-69.582% / -59.818%

5 Conclusion

A novel Bayesian data fusion approach was applied to the problem of vision-based target tracking and showed promising results in the OTB-50 dataset. Significant increases in robustness were observed despite the weaknesses of certain trackers. The method provides an adaptive framework that uses both the local statistics generated by each tracker as well as a weighted majority voting mechanism to determine the target bounding box at each frame. Pretraining is not required, and the method is robust in practical scenarios due to its ability to integrate multiple source of data.

One simple way to extend this work would be to consider the problem of outlier detection. If it is known with high probability that a tracker is lost, it can be reinitialized. Fault detection and correction would improve overall success and greatly assist in generating a better, more robust framework [21]. In particular, it would alleviate the issues that occur when some trackers perform substantially worse than the others. One avenue is to simply use the confidence generated by the Mahalanobis distance to determine if a tracker is an outlier. Possible alternative approaches include supervised ideas such as those presented in [24]. Keeping with the Bayesian and unsupervised nature of the proposed framework, an unsupervised approach is more fitting and some possible ideas include those presented in [14, 16, 17].

References

[1] Christian Bailer, Alain Pagani, and Didier Stricker. *A Superior Tracking Approach: Building a Strong Tracker through Fusion*, pages 170–185. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10584-0. doi: 10.1007/978-3-319-10584-0_12. URL http://dx.doi.org/10.1007/978-3-319-10584-0_12.

[2] Tewodros Atanaw Biresaw, Andrea Cavallaro, and Carlo S. Regazzoni. Tracker-level fusion for robust bayesian visual tracking. *IEEE Trans. Circuits Syst. Video Techn.*, 25(5):776–789, 2015. URL http://dblp.uni-trier.de/db/journals/tcsv/tcsv25.html#BiresawCR15.

[3] Erkan Bostanci, Betul Bostanci, Nadia Kanwal, and Adrian F. Clark. Sensor fusion of camera, GPS and IMU using fuzzy adaptive multiple motion models. *Soft Computing*, Feb 2017. ISSN 1433-7479. doi: 10.1007/s00500-017-2516-8. URL http://dx.doi.org/10.1007/s00500-017-2516-8.

[4] Andres F. Echeverri, Henry Medeiros, Ryan Walsh, Yevgeniy Reznichenko, and Richard J. Povinelli. Hierarchical bayesian data fusion for robotic platform navigation. *CoRR*, abs/1704.06718, 2017. URL http://arxiv.org/abs/1704.06718.

[5] Mahanth Gowda, Justin Manweiler, Ashutosh Dhekne, Romit Roy Choudhury, and Justin D. Weisz. Tracking drone orientation with multiple gps receivers. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*, MobiCom '16, pages 280–293, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4226-1. doi: 10.1145/2973750.2973768. URL http://doi.acm.org/10.1145/2973750.2973768.

[6] Yu Gu, Jason N. Gross, Matthew B. Rhudy, and Kyle Lassak. A Fault-Tolerant Multiple Sensor Fusion Approach Applied to UAV Attitude Estimation. *International Journal of Aerospace Engineering*, 2016, 2016. ISSN 16875974. doi: 10.1155/2016/6217428.

[7] Andrés Echeverri Guevara, Anthony Hoak, Juan Tapiero Bernal, and Henry Medeiros. *Vision-Based Self-contained Target Following Robot Using Bayesian Data Fusion*, pages 846–857. Springer International Publishing, Cham, 2016. ISBN 978-3-319-50835-1. doi: 10.1007/978-3-319-50835-1_76. URL http://dx.doi.org/10.1007/978-3-319-50835-1_76.

[8] Sam Hare, Amir Saffari, and Philip H. S. Torr. Struck: Structured output tracking with kernels. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, editors, *ICCV*, pages 263–270. IEEE Computer Society, 2011. ISBN 978-1-4577-1101-5. URL http://dblp.uni-trier.de/db/conf/iccv/iccv2011.html#HareST11.

[9] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference Computer Vision (ECCV)*, 2016.

[10] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1409–1422, July 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.239. URL http://dx.doi.org/10.1109/TPAMI.2011.239.

[11] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Luka Cehovin, Gustavo Fernandez, Tomas Vojir, Gustav Hager, Georg Nebehay, and Roman Pflugfelder. The visual object tracking vot2015 challenge results. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, December 2015.

[12] Ido Leichter, Michael Lindenbaum, and Ehud Rivlin. A general framework for combining visual trackers–the" black boxes" approach. *International Journal of Computer Vision*, 67(3):343–363, 2006.

[13] Gareth Loy, Luke Fletcher, Nicholas Apostoloff, and Alexander Zelinsky. An adaptive fusion architecture for target tracking. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 261–266. IEEE, 2002.

[14] Yunlong Ma, Peng Zhang, Yanan Cao, and Li Guo. Parallel auto-encoder for efficient outlier detection. *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013*, 2(3):15–17, 2013. doi: 10.1109/BigData.2013.6691791.

[15] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936.

[16] Markos Markou and Sameer Singh. Novelty detection: A review - Part 2:: Neural network based approaches. *Signal Processing*, 83(12):2499–2521, 2003. ISSN 01651684. doi: 10.1016/j.sigpro.2003.07.019.

[17] Emilie Morvant, Amaury Habrard, and Stéphane Ayache. *Majority Vote of Diverse Classifiers for Late Fusion*, pages 153–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-662-44415-3. doi: 10.1007/978-3-662-44415-3_16. URL http://dx.doi.org/10.1007/978-3-662-44415-3_16.

[18] Georg Nebehay and Roman Pflugfelder. Consensus-based matching and tracking of keypoints for object tracking. In *Winter Conference on Applications of Computer Vision*, pages 862–869. IEEE, March 2014.

[19] Raquel Ramos Pinho, João Manuel R. S. Tavares, and Miguel V. Correia. Efficient approximation of the mahalanobis distance for tracking with the kalman filter. In João Manuel R. S. Tavares and Renato M. Natal Jorge, editors, *Computational Modeling of Objects Represented in Images-Fundamentals, Methods and Applications, First International Symposium CompIMAGE 2006, Coimbra, Portugal, October 20-21, 2006.*, pages 349–354. Taylor & Francis, 2006.

[20] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[21] Ryan Walsh and Henry Medeiros. Detecting tracking failures from correlation response maps. In *International Symposium on Visual Computing*, pages 125–135. Springer, 2016.

[22] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418. IEEE Computer Society, 2013. ISBN 978-0-7695-4989-7. URL http://dblp.uni-trier.de/db/conf/cvpr/cvpr2013.html#WuLY13.

[23] Mau-tsuen Yang and Shen-yen Huang. Appearance-Based Multimodal Human Tracking and Identification for Healthcare in the Digital Home. pages 14253–14277, 2014. doi: 10.3390/s140814253.

[24] Seniha Esen Yuksel, Joseph N. Wilson, and Paul D. Gader. Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learning Syst.*, 23(8):1177–1193, 2012. URL http://dblp.uni-trier.de/db/journals/tnn/tnn23.html#YukselWG12.