Hybridization of Data and Model based Object Detection for Tracking in Flash Lidars

Kruttidipta Samal
School of ECE
Georgia Institute of Technology
Atlanta, USA
ksamal3@gatech.edu

Marilyn Wolf
School of ECE
Georgia Institute of Technology
Atlanta, USA
mwolf@unl.edu

Saibal Mukhopadhyay
School of ECE
Georgia Institute of Technology
Atlanta, USA
saibal.mukhopadhyay@ece.gatech.edu

***Abstract*—In recent times deep neural networks have become very successful in solving traditionally hard problems in Computer Vision such as Object Detection. This is due to their ability to find hidden patterns in high dimensional data such as images. But if there is a known structure within data that can be accurately represented by a pre-defined model, then by merging this model based algorithm and deep neural network, overall system accuracy can be increased. We apply this idea for solving the task of flash lidar object detection and tracking.**
Flash lidar is an emerging lidar sensing technology which is getting a lot of attention lately due to their lack of moving parts compared to prevalent scanning lidars. Samples from flash lidar suffer from both spatial and temporal noise which coupled with low angular resolution and FoV lead to low accuracy in object detection.
In this paper we present a data driven deep learning based flash lidar object detector and tracker. To our knowledge, this is the first work to use deep learning for flash lidar object detection/tracking. Our tracker has two detectors- 1. supervised object detector and 2. unsupervised class agnostic foreground/moving object detector which are merged to achieve multi-object tracking accuracy of 47.9% on CAMEL dataset.

***Index Terms*—Flash Lidar, Object Detection, Object Tracking**

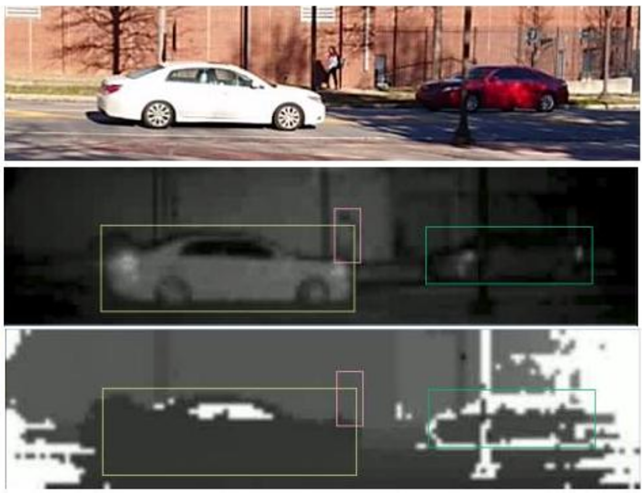

Fig. 1. Sample Image from extended CAMEL dataset. RGB Image(Top) has been cropped to cover area of intersection between FoVs of RGB camera and flash Lidar. Flash lidar Image with intensity channel is shown in middle row and depth channel in bottom row. Subjects of interest such as car, van and person are annotated in flash lidar image.

I. Introduction

Lidar cameras are being increasingly used in autonomous systems to create a 3D map of the world around. Most common lidars used in both academic research and AV industry are scanning lidars, especially Velodyne HDL-64 [1], which has fine angular resolution and 360° FoV. While such scanning lidars are very effective for creating accurate 3D point clouds, they are not ideal for long life and real-time applications due to presence of moving parts such as vertical sensor array which affect durability and fps. In order to address these concerns, there has been interest towards flash lidars [2], which operate on time-of-flight principle and do not have moving mechanical parts. Real-time tracking of objects is a critical computer vision task in many autonomous systems. Consequently, there is a strong interest in developing object tracker for flash lidar data. While there have been many works on object tracking based on scanning lidars [3], [4], there are relatively few works on flash lidars [5] [6].

Many state-of-the-art object detectors are based on tracking-by-detection. Prior work on flash lidar object detection and tracking [5] [6], use 3D point clustering with prior object model to detect objects of specific domain such as person or drogue(part of an aircraft). This static nature of object model makes it difficult to extend these techniques to be used ubiquitously across multiple object categories in real-world scenarios. A data-driven deep learning based object detector for flash lidar can overcome this challenge. But due to low angular resolution and noise in flash lidar, feature quality is bad, creating a challenge for using deep learning based detector. Hence, to successfully track objects of multiple categories in a flash lidar mode, we need to develop a data-driven detection technique that is robust to inherent shortcomings of the sensor.

This paper, for the first time, to the best of our knowledge, presents a data driven deep learning based object detector and tracker operating on a flash lidar. The underlying hypothesis of our approach is that when detection accuracy is low, relying on

The research reported here was supported by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-17-2-0045. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

a single detection algorithm will lead to low tracking accuracy. Hence, we propose to use multiple detection algorithms to infer state of a target to enhance tracking accuracy in flash lidar. The additional algorithm does not depend on the same features as primary detector and does not add excessive computational overhead. In particular the paper makes the following key contributions:

- We present a Detection and Tracking of Multiple Objects (DATMO) system for flash lidar that hybridizes supervised and un-supervised deep learning with model-based computer vision to improve tracking accuracy.
- We present a fully supervised deep-learning based baseline object detector for flash lidar.
- We develop an unsupervised method to create foreground/moving object mask. We first present a model-based approach for unsupervised detection; which is next used to self-supervise a full convolutional network(FCN) to develop a reduced complexity detector.
- We develop a tracking approach that improves accuracy by merging outputs from both supervised and unsupervised modules during ambiguity between tracker and detector.

Our system was evaluated on CAMEL [7] muti-spectral dataset which contains over 8000 flash lidar images collected from 13 sequences. We observe that by adding model-based foreground object detector to our DATMO system, tracking accuracy improves by 9.6% over a detector based only on a Deep Neural Network(DNN), however computational load increases from 0.08 GFLOPs to 0.43 GFLOPs. Replacing model-based foreground detector with self-supervised FCN, computational load reduces to 0.26 GFLOPs but we also observe a decrease in accuracy.

II. Dataset

Popular datasets with lidar data e.g. KITTI [1], APOLLO [8], nuScenes [9] capture scenes for AV using Velodyne HDL-64 scanner. Such detectors cannot be directly ported to flash lidars which have different angular resolution, operating wavelength, FoV etc. Therefore, we extended CAMEL [7] multi-spectral dataset by adding flash lidar data. This dataset has scenes of traffic and cluttered urban scenes with heavy pedestrian traffic.

The lidar data was captured from an ASC Peregrine Flash Lidar, operating laser wavelength is 1570 nm and has a maximum frame rate of 25 fps. It has an FoV of 30° azimuth and 7.5° elevation, frame rate of capture was 5-10 fps.

Lidar frames were annotated using CVAT [10] annotation tool. There are 5 classes annotated- person, car, van, bus, and bike. This dataset contains 8,282 annotated frames, out of which 6,876 frames are in training data and 1,406 frames are in validation data. We collected 13 sequences containing 374 unique tracks acquired from urban campus environments. Sample image from extended CAMEL dataset is shown in Figure 1.

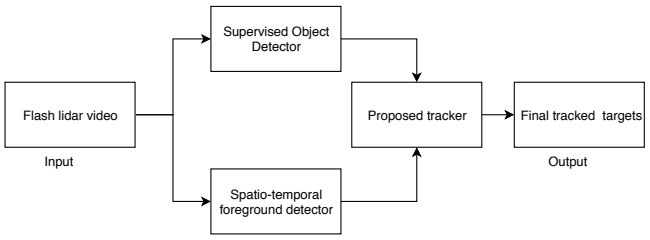

Fig. 2. System architecture of proposed DATMO system

III. Proposed Flash Lidar DATMO System

A. Overall System Architecture

Most existing single modality trackers use object detector as their only source of input for creating and updating new tracklets. This limits performance of tracking as it is limited by accuracy of object detector. We propose hybrid detection system that judiciously integrates multiple detection sources to enhance tracking accuracy (Figure 2). We first design a supervised object detector by adapting existing DNN-based object detector architecture for flash lidar dataset. We next design two class agnostic unsupervised foreground/moving object detectors using model-based vision algorithms and fully convolutional networks. Finally, we develop a tracking system that couples the detection from two different detectors. These multiple sources of detection help tracker to confirm or reject a hypothesis during periods of ambiguity and improve accuracy by a significant margin over single detector system.

B. Supervised Object Detector

For object detection we created a real time CNN based object detection network. Similar to SSD [11] and YOLO [12], we created an FCN with route connections, and then added 2 object regression layers at different depths (YOLO Layers in figure 3) to capture objects with different sizes. The choice of hyper-parameters and network parameters was obtained via grid search. Unlike SSD, which has 6 object regression layers, our architecture has 2 object regression layers (similar to YOLO) because the dimension of flash lidar image (32x128) is much smaller than typical RGB image (greater than 300x300) and we observed 2 YOLO layers are sufficient to detect objects of different sizes. Our supervised network has 25 layers with 102,460 parameters and has a computational cost of 0.08 GFLOPS. Network architecture of this object detector is shown in Figure 3 and detailed layer-wise parameter distribution is shown in Table I. This network was trained for 50 epochs with initial learning rate set to 0.01 and decayed by factor of 10 at epochs 20, 30 and 40. Sample outputs of supervised object detector are shown in Figure 4. Source code and training data for this network is available at- https://gitlab.com/deepsamal/flash-lidar-object-detection-and-tracking.

C. Unsupervised Object Detection

In addition to the supervised object detector, we created two unsupervised object detectors which do not depend on

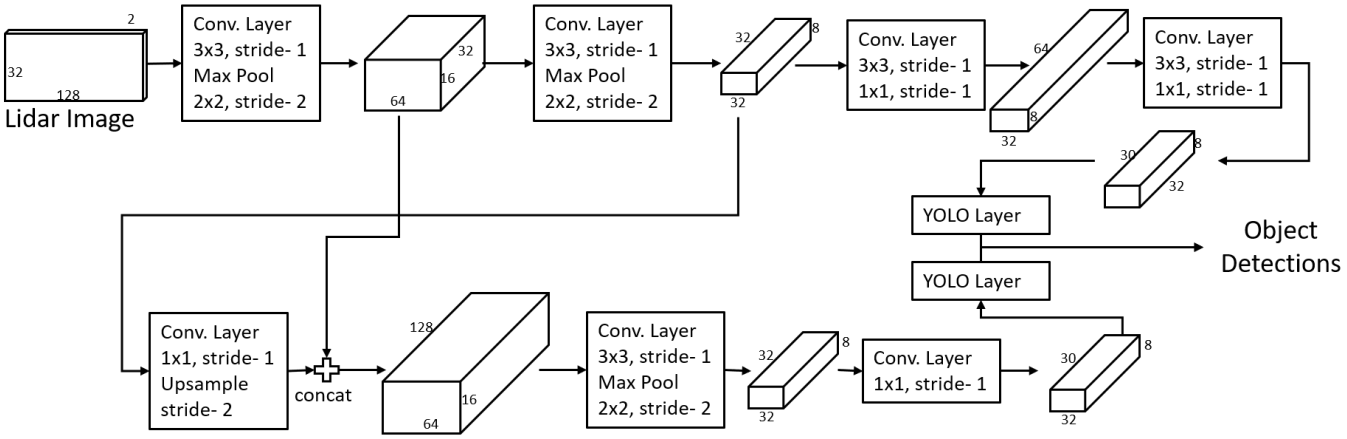

Fig. 3. Network Architecture of Supervised Object Detector.

Layer #	Layer Type	No. of Parameters	Size
0	Conv weight	576	[32, 2, 3, 3]
1	Batch Norm weight	32	[32]
2	Batch Norm bias	32	[32]
3	Conv weight	9216	[32, 32, 3, 3]
4	Batch Norm weight	32	[32]
5	Batch Norm bias	32	[32]
6	Conv weight	9216	[32, 32, 3, 3]
7	Batch Norm weight	32	[32]
8	Batch Norm bias	32	[32]
9	Conv weight	2048	[64, 32, 1, 1]
10	Batch Norm weight	64	[64]
11	Batch Norm bias	64	[64]
12	Conv weight	18432	[32, 64, 3, 3]
13	Batch Norm weight	32	[32]
14	Batch Norm bias	32	[32]
15	Conv weight	960	[30, 32, 1, 1]
16	Conv bias	30	[30]
17	Conv weight	4096	[64, 64, 1, 1]
18	Batch Norm weight	64	[64]
19	Batch Norm bias	64	[64]
20	Conv weight	55296	[64, 96, 3, 3]
21	Batch Norm weight	64	[64]
22	Batch Norm bias	64	[64]
23	Conv weight	1920	[30, 64, 1, 1]
24	Conv bias	30	[30]

TABLE I
NETWORK LAYER DETAILS OF SUPERVISED NETWORK (LEARNABLE PARAMETERS ONLY)

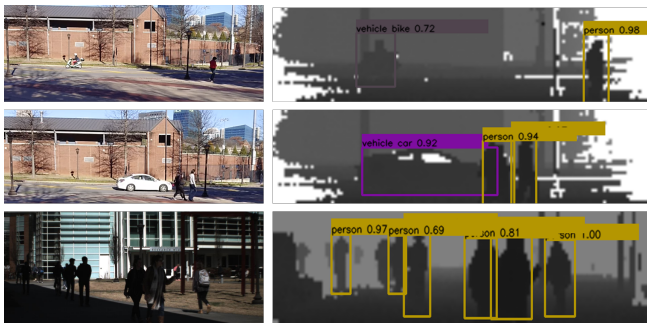

Fig. 4. Output of Supervised Object Detector(Right column). RGB Images(Left column) are for reference only

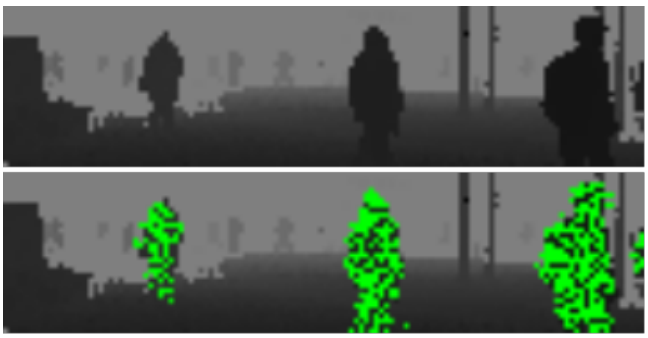

Fig. 5. Flash lidar range image(top). Foreground mask predicted by spatio-temporal foreground detector(bottom)

the same architecture or training method as that of supervised detector.

Spatio-Temporal Object Mask: We use ground removal [13] followed by DBSCAN clustering [14] on point cloud generated from depth map to create object candidates. This clustering algorithm is robust against noisy and low density data. Using local convexity criterion for clustering [13] is not optimal here as quality of neighborhood graph will suffer due to low angular resolution. After DBSCAN clustering, all points belonging to dominant clusters are projected back on depth map to create a spatial saliency map.

We use a Mixture of Gaussian (MOG) based background subtraction [15] to create temporal saliency map. This mask is noisy as MOG uses per pixel statistics to create a background model, and flash lidar images contain substantial amount of random noise.

Finally we combine both spatial and temporal saliency maps by selecting spatial clusters which have large number of moving points to create a spatio-temporal object mask. Figure 5 shows output of this algorithm, here detected foreground pixels are highlighted in green. Architecture diagram of this module is shown in figure 6.

While this method helps in creating good quality foreground object masks, computational cost is high due to $O(N^2)$ '3D

Layer #	Layer Type	No. of Parameters	Size
0	Conv weight	576	[32, 2, 3, 3]
1	Batch Norm weight	32	[32]
2	Batch Norm bias	32	[32]
3	Conv weight	9216	[32, 32, 3, 3]
4	Batch Norm weight	32	[32]
5	Batch Norm bias	32	[32]
6	Conv weight	36864	[128, 32, 3, 3]
7	Batch Norm weight	128	[128]
8	Batch Norm bias	128	[128]
9	Conv weight	8192	[64, 128, 1, 1]
10	Batch Norm weight	64	[64]
11	Batch Norm bias	64	[64]
12	Conv weight	36864	[64, 64, 3, 3]
13	Batch Norm weight	64	[64]
14	Batch Norm bias	64	[64]
15	Conv weight	4096	[64, 64, 1, 1]
16	Conv bias	64	[64]
17	Conv weight	73728	[128, 64, 1, 1]
18	Batch Norm weight	128	[128]
19	Batch Norm bias	128	[128]
20	Conv weight	46080	[32, 160, 3, 3]
21	Batch Norm weight	32	[32]
22	Batch Norm bias	32	[32]
23	Conv weight	18432	[32, 64, 3, 3]
24	Conv bias	32	[32]
25	Batch Norm bias	32	[32]
26	Conv weight	64	[2, 32, 1, 1]
27	Conv bias	2	[2]

TABLE II
NETWORK LAYER DETAILS OF UNSUPERVISED FCN (LEARNABLE PARAMETERS ONLY)

euclidean distance' operations in 'nearest neighbor search' during clustering. For each frame, worst case number of FLOPS in nearest neighbor search is 0.35 GFLOPs.

Self-Supervised FCN: We present a light weight FCN with encoder-decoder architecture(System architecture in Figure 8) to generate foreground object mask(Figure 7). Similar to U-NET [16], this FCN has upconvolutional layers (convolutional layer followed by nearest-neighbour upsampling layer) in decoder to increase feature map size. Route connections that concatenate feature maps from early stages with output of later stages are added to retain small object and low level information(detailed network architecture in Figure 9). This network has 28 layers with 235,202 parameters (layer and parameter details in Table II) consuming 0.26 FLOPS. Instead of using labelled foreground mask, we use spatio-temporal object mask generated by the model-based algorithm presented in Section III-C as supervisory signal during training. This strategy lets us train on unlabelled data. Training epochs, learning rate and momentum of this network is same as supervised network presented in Section III-B. Our FCN was trained on 32,358 samples which is almost four times the number of labelled samples in CAMEL dataset. This strategy is similar to work presented by Pathak et. al. [17], which uses motion as a pretext task to pre-train convolutional weights of a supervised object detection network. However, their method uses Non-Local Consensus voting [18] to create object mask from RGB video which cannot be extended to streaming lidar data.

D. Proposed Tracking Approach

Our tracking method is based on SORT [19] algorithm which is a detection based tracker with linear velocity Kalman filter as state predictor. At any given time-step, it takes detections from object detector and associates them with existing tracklets using Hungarian Association. Cost matrix of this hungarian association uses intersection-over-union of observed bounding box and expected bounding box of existing tracklets. The state of a tracklet in our system is modelled as:

$$x = [u\ v\ s\ r\ d\ \dot{u}\ \dot{v}\ \dot{s}\ \dot{d}]^T \tag{1}$$

Here, *u* and *v* represent horizontal and vertical position, *s* and *r* represent scale and aspect ratio of the target's bounding box, d represents target depth.

Our baseline DATMO system has supervised object detector as the only source of detection. This system has high false negatives during tracking, as inconsistency in detector is directly reflected in tracking.

Therefore, we merge detections from supervised detector with foreground object mask generated from spatio-temporal mask generator. This hybrid system refers to foreground mask when there is an absence of detection from supervised detector at predicted tracklet position. Figure 10 shows output of this system. At time t=T, all tracklets are dependent on supervised module for detection, but at t=T+1 and t=T+2, detection for tracklet 30(green) is switched to foreground mask as there is no corresponding detection from supervised detector. Following are conditions for assuming supervised detector made false negative when there is no corresponding detection for a tracklet:

- Foreground probability at predicted box position should be greater than fg^{thr}.

$$fg\_probability = \frac{\sum_{x,y \in B} fg\_mask(x,y)}{n(B)} \tag{2}$$

  where B is a set of all (*x,y*) positions within predicted bounding box.
- Observed depth at predicted box position is similar to predicted depth.

$$|depth\_map(x_c, y_c) - d| < d^{thr} \tag{3}$$

  Where (x_c, y_c) is the co-ordinate at predicted bounding box center, depth_map is depth channel of current lidar image, 'd' is depth element in target state and d^{thr} is depth threshold.

Here, fg^{thr} and d^{thr} are determined empirically.

IV. EXPERIMENTAL RESULTS

We evaluated proposed DATMO system on test sequences of CAMEL dataset using CLEAR MOT metrics [20], and supervised detector was evaluated on validation set of CAMEL dataset (see table III). Results are shown in table IV. We observe an increase in MOTA by 9.6% and 28% decrease in false negatives between baseline and hybrid system. Replacing spatio-temporal mask module with FCN, we observe a fall in

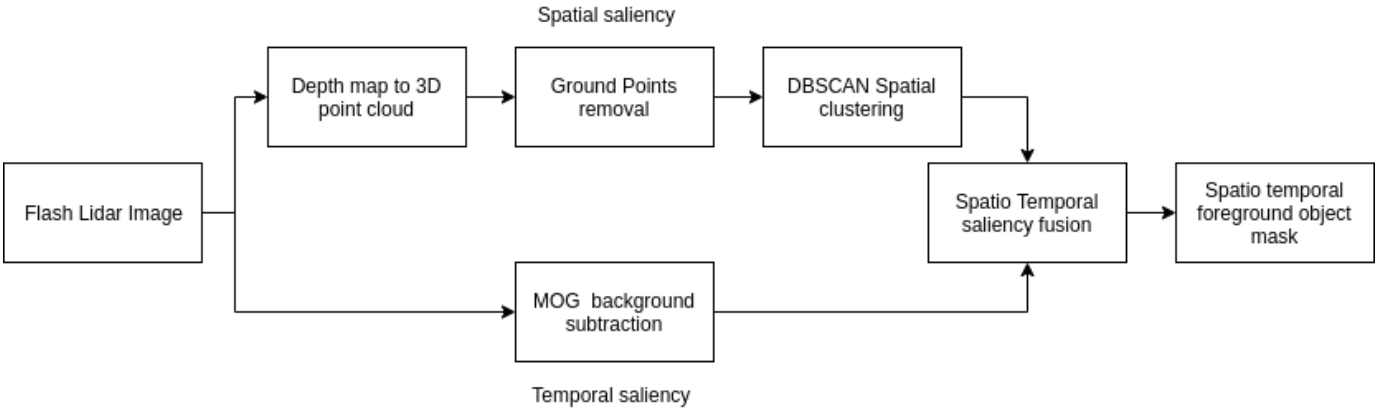

Fig. 6. Architecture for spatio-temporal object detector

Fig. 7. FCN generated mask(bottom) is noisier than Spatio-temporal object mask(top)

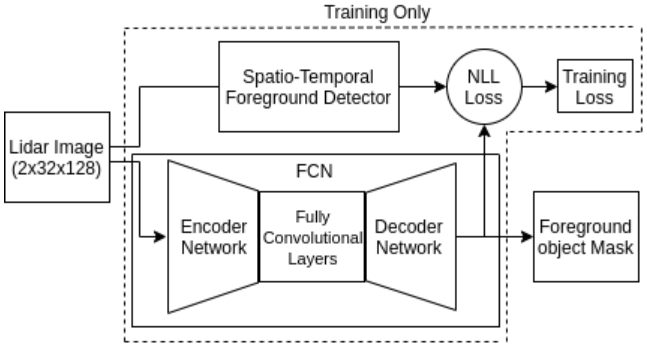

Fig. 8. System Architecture of FCN. Only FCN module used during operation, dotted box for training only.

MOTA due to rise in false positives as masks generated by FCN are not accurate, but FCN consumes significantly less computational resource compared to spatio-temporal algorithm (see 'Load' column in table IV).

In scenarios where a tracked object goes behind occlusion, ideally tracker should trust missing detection from supervised detector, but instead it trusts foreground mask, which is noisy and target agnostic. Thus if there are foreground pixels at expected position of an occluded object but they do not belong to the same tracked object, tracker maintains the tracklet

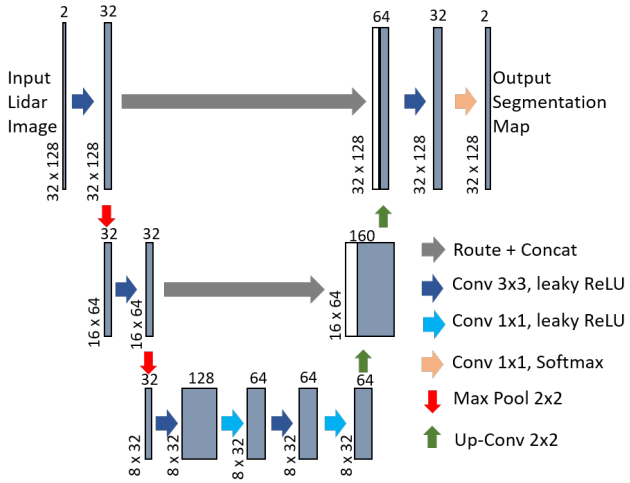

Fig. 9. Detailed Network Architecture of FCN.

instead of deleting it, this increases false positive. There are also scenarios where detector outputs a false positive, usually such detections are not consistent, it is expected in subsequent frames when object detector stops detecting the false positive, tracker will delete corresponding track. But in our proposed approach, when detector stops detecting the false positive instance, tracker refers to foreground mask for evidence. In such a scenario, false positive and noise of the foreground object detector is reflected in tracker performance, thus increasing false positives in hybrid systems compared to baseline.

The source code and data for unsupervised object detectors and proposed tracker will be made publicly available soon.

V. Conclusion

In this work we presented a hybrid DATMO system for flash lidar, which merges deep learning based and model based object detectors to achieve robust tracking performance. We observe that due to noise and low feature quality of flash lidar data, using just data-driven supervised object detector leads to

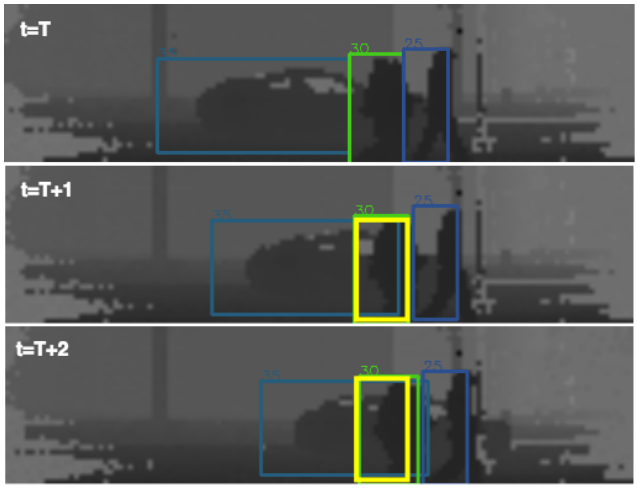

Fig. 10. Output of DATMO with different tracklets. Tracklets with yellow box indicate their detection is sourced from unsupervised detector and not from supervised detector.

Category	Precision	Recall	MAP
all	0.671	0.622	0.592
person	0.726	0.681	0.647
car	0.439	0.444	0.439
bike	0.160	0.165	0.160

TABLE III
SUPERVISED OBJECT DETECTOR ACCURACY

Name	FP↓	FN↓	MOTA↑	Load↓
Sup. Only	83	**1078**	**38.3**%	**0.08**
Sup.+ST Mask	199	**777**	**47.9**%	**0.43**
Sup.+FCN Mask	274	**772**	**44.1**%	**0.25**

TABLE IV
TRACKING EVALUATION AND PER FRAME COMPUTATIONAL LOAD OF BASELINE SYSTEM (TOP 1 ROW) AND PROPOSED SYSTEMS (BOTTOM 2 ROWS).

high false negatives. But when this module is coupled with a model based class agnostic detector, overall tracking accuracy improves by a significant margin.

REFERENCES

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[2] R. Thakur, "Scanning lidar in advanced driver assistance systems and beyond: Building a road map for next-generation lidar technology," *IEEE Consumer Electronics Magazine*, vol. 5, no. 3, pp. 48–54, 2016.

[3] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[4] S. Shi, Z. Wang, X. Wang, and H. Li, "Part-aˆ 2 net: 3d part-aware and aggregation neural network for object detection from point cloud," *arXiv preprint arXiv:1907.03670*, 2019.

[5] M. Hammer, M. Hebel, and M. Arens, "Automated object detection and tracking with a flash lidar system," in *Electro-Optical Remote Sensing X*, vol. 9988. International Society for Optics and Photonics, 2016, p. 998803.

[6] C.-I. Chen and R. Stettner, "Drogue tracking using 3d flash lidar for autonomous aerial refueling," in *Laser Radar Technology and Applications XVI*, vol. 8037. International Society for Optics and Photonics, 2011, p. 80370Q.

[7] E. Gebhardt and M. Wolf, "Camel dataset for visual and thermal infrared multiple object detection and tracking," *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, 2018.

[8] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," *CoRR*, vol. abs/1803.06184, 2018. [Online]. Available: http://arxiv.org/abs/1803.06184

[9] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," *arXiv preprint arXiv:1903.11027*, 2019.

[10] "Computer vision annotation tool (cvat)," April 2019. [Online]. Available: https://github.com/opencv/cvat

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[12] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[13] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion," in *2009 IEEE Intelligent Vehicles Symposium*. IEEE, 2009, pp. 215–220.

[14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise."

[15] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[17] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2701–2710.

[18] A. Faktor and M. Irani, "Video segmentation by non-local consensus voting," in *Proceedings of the British Machine Vision Conference. BMVA Press*, 2014.

[19] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," *CoRR*, vol. abs/1602.00763, 2016. [Online]. Available: http://arxiv.org/abs/1602.00763

[20] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *Journal on Image and Video Processing*, vol. 2008, p. 1, 2008.