December 9, 2019

Robust 3D Object Tracking in Autonomous Vehicles

Eric Ryan Chan
erchan@stanford.edu

Anthony Galczak
agalczak@stanford.edu

Anthony Li
antli@stanford.edu

Abstract

We present a stereo-camera-based 3D multiple-vehicle-tracking system that utilizes Kalman filtering to improve robustness. The objective of our system is to accurately predict locations and orientations of vehicles from stereo camera data. It consists of three modules: a 2D object detection network, 3D position extraction, and 3D object correlation/smoothing. The system approaches the 3D localization performance of LIDAR and significantly outperforms the state-of-the-art monocular vehicle tracking systems. The addition of Kalman filtering increases our system's robustness to missed detections, and improves the recall of our detector. Kalman filtering improves the MAP score of 3D localization for moderately difficult vehicles by 7.7%, compared to our unfiltered baseline. Our system predicts the correct orientation of vehicles with 78% accuracy. Our code, as well as a video demo, is viewable here.

1 Introduction

A requirement for safe autonomous vehicles is object tracking in 3D space. By tracking cars and other obstacles, an autonomous vehicle can plan a route and avoid collisions. However, tracking 3D objects and rotations is a notoriously difficult problem. Object tracking is most commonly solved by LIDAR, which is accurate but also very expensive. We propose a method utilizing inexpensive stereo cameras. However, visual 3D detection comes with a major challenge: precision rapidly decreases with increasing distance from the cameras. As a result, the bounding boxes generated by most image-based 3D object detection systems tend to jump around between frames, resulting in an unstable tracking. Some camera-based detections are dropped entirely, leading to tracking loss.

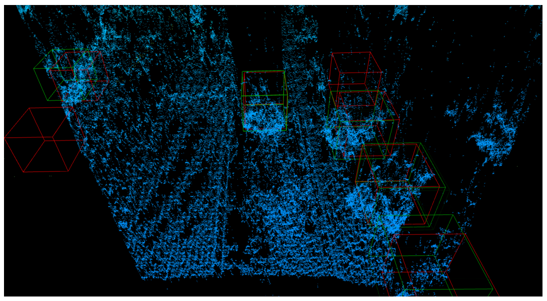
Figure 1: Visualization of our 3D Vehicle Tracker. Predictions are green, groundtruth labels are red.

Our goal is to predict stable and accurate 3D bounding boxes around multiple vehicles in a self-driving car data set. We aim to provide a vehicle-tracking system that is robust to dropped predictions and more stable frame-to-frame than a pure neural network object detection approach. We are applying our model to the KITTI Object Tracking 2012 data set[1]. The input to our 3D-tracking system is a sequence of 1382x512 stereo image-pairs at 10 fps. For each image pair, we use a 2D object detection network to localize vehicles in 2D image-space and predict orientations. We reproject the stereo images into 3D and use depth information to generate raw 3D localization predictions. We then utilize a filtering model to aggregate raw predictions over time, generating refined 3D localization predictions. The system outputs the 3D position and orientation over time for each tracked vehicle, which can be used to produce 3D bounding boxes.

This project combines elements of CS230 and CS238. Although we have three team members, only Anthony Galczak and Eric Chan are in CS230. The work for CS230 was focused on generating raw 3D predictions, while the work for CS238 was focused on specifics of object tracking including Kalman filtering.

2 Related Work

3D object tracking is a well-studied problem. The state-of-the-art in 3D object tracking for vehicles is LIDAR [24] [12], which is capable of precise measurements at long range. All of the top KITTI evaluated algorithms, such as *STD: Sparse-to-Dense 3D Object Detector for Point Cloud*[19], utilize LIDAR. The downside to LIDAR is cost and complexity–the sensor cost per car is generally upwards of $100,000, and it is computationally expensive to process point cloud videos using neural networks.

There have been attempts to replicate the performance of LIDAR using visual systems. The work of Godard et al.[25] focuses on monocular depth estimation, while the work of Simonelli et al. [13] focuses on monocular 3D localization and orientation for vehicles. However while orientation prediction performance has shown good results, the depth-perception and 3D localization precision of such systems has been shown to be extremely lacking compared to LIDAR.

Stereo depth-perception systems have shown better precision than monocular depth-perception systems, although it also suffers from imprecision at range. [15][25]. Our work focuses on combining the precision advantage of stereo systems with the principles of monocular tracking and orientation estimation.

We utilized proven methods for tracking moving objects. Both Kalman filtering and Particle filtering are appropriate for the task of tracking multiple objects[16]. These filtering algorithms are often used for the task of tracking moving objects for self-driving cars[17].

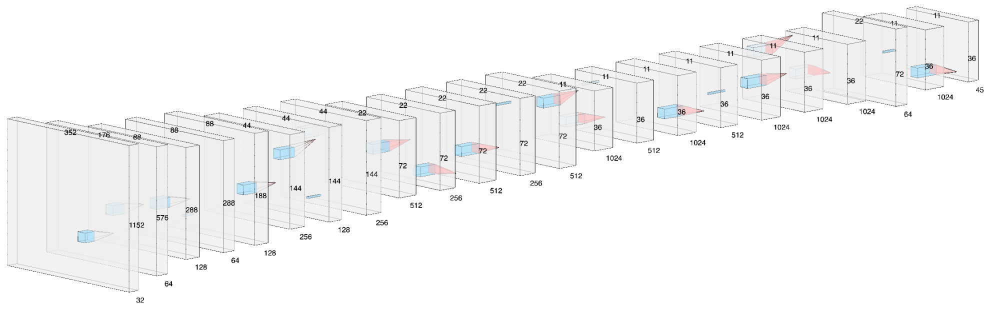
Figure 2: Modified YOLOv2 neural network for 2D object detection and orientation inference.

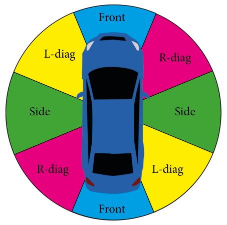

Figure 3: We discretize the observation angle into four distinct categories and treat orientation prediction as a classification problem.

Comparisons of our results to state-of-the-art LIDAR and Monocular detection systems can be found in the Discussion section.

3 Dataset

We train our 2D object detector network on the KITTI Object Detection 2012 data set[2]. Each 1382x512 RGB image is labeled with a ground truth 2D bounding box and a ground truth orientation angle. Following the recommendation of KITTI, we utilized a 7481/7518 train/test split.

We evaluate our tracking accuracy on the KITTI Object Tracking 2012 data set[1]. The dataset provides sequential stereo RGB images at 10fps along with ground-truth 3D bounding box labels. The dataset is comprised of 21 labeled sequences, each with between 200 and 1200 stereo frames.

4 Methods and Features

We have split the complex problem of 3D vehicle tracking into three separate tasks–2D Object Detection, 3D Position Extraction, and 3D Object Correlation and Filtering. The steps in our 3D detection pipeline are illustrated in Figure 4.

4.1 2D Object Detection Network

We utilize a YOLOv2 object detection network to extract image-space bounding boxes and observation angles from our imagery. Our choice of network was informed by prior research into 2D-object detection for vehicles. YOLOv2 has been shown to outperform YOLOv3 in both Mean Average Precision (MAP) and inference time on vehicle detection tasks[20]. While Faster-RCNN outperforms YOLOv2 in MAP, it is incapable of running in real-time for high frame rates[20]. Compared to YOLOv3[22] and Faster-RCNN, YOLOv2 has a good balance of accuracy and performance. Our network is based on the Tensorflow implementation of the YOLOv2, Darkflow [21].

We framed orientation prediction as a classification problem, and discretized the observation angles of vehicles into four separate categories, as shown in figure 3. Because the work of Rybski, et. al. [14] has shown that orientation estimation networks tend to confuse exactly opposite orientations, and because we can use tracked vehicle velocity to distinguish between opposite orientations, we made the decision to treat opposite orientation as the same class. By reducing the number of classes predicted by the network, we increased the robustness of our predictions.

We modified the YOLOv2 network to fit the task of vehicle detection. Because we were focused on vehicles and orientation prediction rather than the more complicated task of general object detection, we reduced the number of filters in the deep convolutional layers. Unmodified YOLOv2 has been shown to perform poorly on the KITTI dataset, likely because KITTI uses very high aspect-ratio (1382 x 512) images [23]. In order to improve performance on the KITTI dataset, we resized the layers. Empirically, random input cropping was shown to hurt performance, so we chose not to perform data augmentation.

Table 1: Network Parameters

Layer	Output HxWxD	Filter
Convolutional	352x1152x32	3x3
Max-Pool	176x576x32	2x2
Convolutional	176x576x64	3x3
Max-Pool	88x288x64	2x2
Convolutional	88x288x128	3x3
Convolutional	88x288x64	1x1
Convolutional	88x288x128	3x3
Max-Pool	44x144x128	2x2
Convolutional	44x144x256	3x3
Convolutional	44x144x128	1x1
Convolutional	44x144x256	3x3
Max-Pool	22x72x256	2x2
Convolutional	22x72x512	3x3
Convolutional	22x72x256	1x1
Convolutional	22x72x512	3x3
Convolutional	22x72x256	1x1
Convolutional	22x72x512	3x3
Max-Pool	11x36x512	2x2
Convolutional	11x36x1024	3x3
Convolutional	11x36x512	1x1
Convolutional	11x36x1024	3x3
Convolutional	11x36x512	1x1
Convolutional	11x36x1024	3x3
Convolutional	11x36x1024	3x3
Convolutional	11x36x1024	3x3
Concat	22x72x512	
Convolutional	22x72x64	1x1
Concat	11x36x1280	
Convolutional	11x36x512	3x3
Convolutional	11x36x45	1x1

Training Hyperparameters	
Optimizer	ADAM
Minibatch Size	8
Learning Rates	1E-4, 1E-3, 1E-6
Beta1:	0.9
Beta2:	0.999
Epochs:	20

We initialized the network using weights pre-trained on the COCO dataset, but fine-tuned it to the KITTI 2D Object Detection dataset over 20 Epochs. Initial training was conducted with a low learning rate of 1E-4 to prevent divergence. The bulk of training was conducted at a learning rate of 1E-3, and the final two epochs were trained at a learning rate of 1E-6. We also utilized K-means clustering to generate anchors that better fit the KITTI dataset. Our modified network structure is given in Figure 2. Network parameters are given in Table 1. We experimented with training the network on stereo disparity images concatenated to RGB images as a six-channel input volume. However, possibly due to a shortage of training data, the augmented network did not generalize well and our performance was generally worse than with the pure RGB approach.

4.2 3D Position Extraction

After localizing vehicles in 2D image space, we used stereo correspondence algorithms to estimate the 3D positions of our detections. In order to calculate stereo disparity from our rectified image pair, we utilized Semi-Global Block Matching[10]. Unlike previous scan-line approaches that calculated disparity only along the horizontal epipolar lines, Semi-Global Block Matching optimizes and averages across multiple directions, resulting in a smoother estimation of pixel disparity.

Despite the improvements of SGBM over older disparity estimation algorithms, SGBM is still prone to errors in areas of half-occlusions, as well as regions with depth continuities. In order to further improve the accuracy of the disparity map, we computed both the left-to-right and right-to-left stereo correspondences. We then combined the two estimations into a smoother, more complete disparity map using Weighted Least Squares Filtering[5].

After computing the accurate disparity map, we reprojected the 2D image space points into 3D using a perspective transformation.

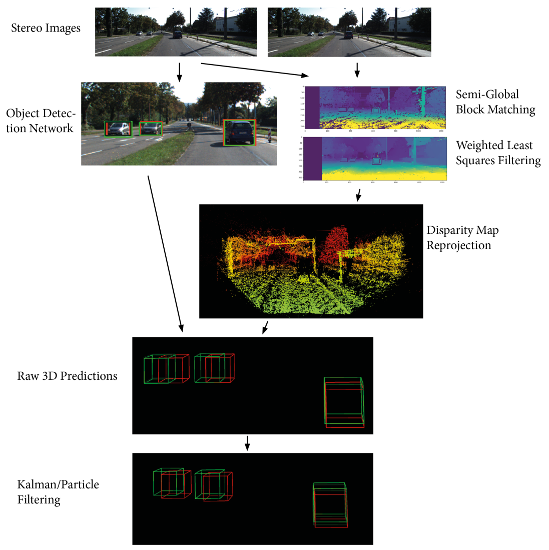
Stereo Images
Object Detection Network
Semi-Global Block Matching
Weighted Least Squares Filtering
Disparity Map Reprojection
Raw 3D Predictions
Kalman/Particle Filtering

Figure 4: Diagram illustrating steps in our 3D Vehicle-Tracking System.

4.3 Kalman and Particle Filtering

To improve the stability and robustness of our tracking system under short-term detection loss, we refined the raw 3D location estimations using filtering.

Our system implements object *tracking* rather than simply object *detection*. At each frame, raw detections are correlated with currently-tracked vehicles using a nearest-distance matching heuristic. Their velocities are estimated using numerical differentiation. Estimated-vehicle positions are propagated according to a constant-velocity transition model. By aggregating multiple noisy estimations over time, our model produces a better estimate of real-world position. Our transition model also allows our system to propagate tracked vehicles even in the absence of raw detections.

We chose to evaluate a Kalman-filtering model, a particle-filtering model, and a raw-prediction baseline model.

4.4 Evaluation Metrics

We evaluate our system using the criteria suggested by the KITTI vision benchmark. KITTI defines difficulty classes for detections. Vehicles classified as “Easy” are greater than 40 pixels in image-space height and fully visible. Vehicles classified as “Moderate” or “Medium” are greater than 25 pixels in height and “partly occluded”. Vehicles classified as “Hard” are greater than 25 pixels in height and “difficult to see” according to the authors of KITTI.

To measure localization accuracy for 3D tracking, we calculate Precision-Recall curves and a MAP metric. We consider a prediction to be a True Positive if the center-of-mass of the prediction lies within 1.5 m to the center-of-mass of its corresponding groundtruth location. In our results, we also evaluate at 3 m, 5 m, and 10 m distance thresholds.

To measure orientation prediction accuracy, we compare the predicted orientation class against the groundtruth orientation class for each matched vehicle pair. We calculate orientation prediction precision as the ratio of correctly classified vehicles to total orientation predictions.

For 2D object detection, we calculate Precision-Recall curves and a MAP metric. We consider a prediction to be a True Positive if the Intersection over Union (IoU) with an unmatched groundtruth box exceeds 0.5.

5 Results

Our combined pipeline is able take in raw stereo images and output filtered 3D bounding boxes. A visualization of our current results is given in Figure 1.

Table 2: Mean Average Precision comparison of our unfiltered, Kalman filtered, and Particle filtered results against state-of-the-art Monocular and LIDAR vehicle-detection systems.

	Unfiltered	**Kalman**	Particle	Mono[13]	LIDAR[19]
Easy	0.799	0.8079	0.7251	.1805	0.8661
Med	0.3737	0.4027	0.3671	.1498	0.7763
Hard	0.1198	0.1269	0.1192	.1342	0.7606

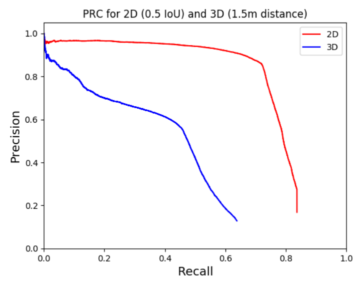

Figure 5: PR curve for 2D tracking vs. 3D tracking.

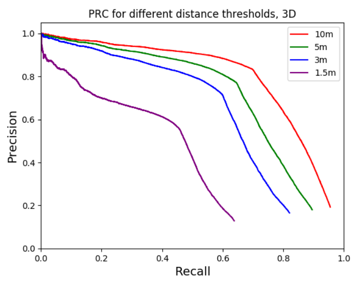

Figure 6: PR curve for each distance threshold, using Kalman filtering.

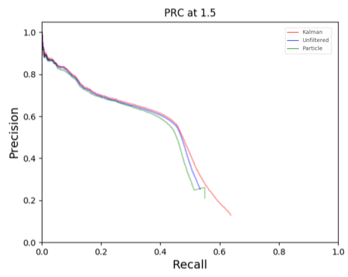

Figure 7: PR curve for filtering method comparison.

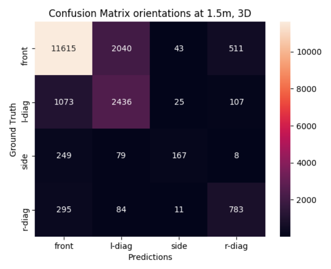

Figure 8: Confusion matrix for orientation predictions.

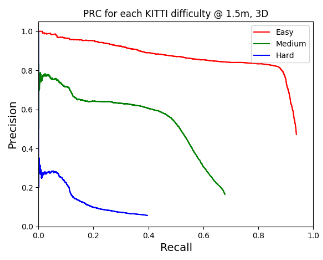

Figure 9: PR curve for each KITTI difficulty.

5.1 Localization Performance

Figure 9 and Table 2 compare the performance of our 3D tracking system at different difficulties. Our system performs very well on "Easy" detections–vehicles that are fully visible and relatively close to the camera. Our system has lower performance on vehicles that are far away and partially or significantly occluded, but still performs adequately. These results indicate that as expected, localization precision decreases with increasing distance to the cameras. Additionally, localization precision is worsened by large occlusions.

Figure 6 gives average precision curves for our system at different distance thresholds. Our system is able to localize the majority of vehicles to within 1.5 m of their real-world positions. It is able to localize nearly all vehicles to within 3 m of their real-world positions. Our system, especially at long ranges, is not perfectly precise, and will not give millimeter-perfect localization. However, we contend that the precision is adequate for most autonomous vehicle tasks.

5.2 Orientation Prediction Performance

The confusion matrix given in figure 8 compares our predicted orientations to groundtruth orientations. Our system correctly predicts the orientation of cars 76.8% of the time. The most common prediction errors are between adjacent angles. For example, the network is more likely to predict a front-facing car to be diagonal than it is to be completely sideways.

Because many training images are taken from highway driving, the KITTI dataset is heavily biased towards front-facing cars. There are few training images captured at intersections, so the performance of "side" predictions is comparatively low. We believe that additional training examples with more varied orientations would improve our network's orientation prediction performance.

5.3 Filtering performance

We evaluated the 3D localization performance of three different filtering models: an unfiltered baseline model, a particle filtered model, and a Kalman filtered model. The precision-recall curves for 3D localization with particle filtering and Kalman filtering are compared to unfiltered results in Figure 7. MAP values for the three filtering methods are given in Table 2.

Kalman filtering improved baseline 3D localization performance for all three difficulties. However, the greatest improvements came in localizing medium and hard vehicles, where Kalman filtering improved MAP by 7.7% and 5.9% respectively. Filtering provides two benefits: robustness to dropped predictions and smoothing of sensor noise. The raw prediction system sometimes fails to detect vehicles that are difficult to detect, "dropping" predictions. Our filtering solution provides robustness to dropped predictions by propagating tracked vehicles. When our object detector fails to predict a vehicle for 1-2 frames, the filter continues tracking through the brief loss of detection, updating the position of our estimate using our constant-velocity assumption. Additionally, by aggregating multiple noisy samples over time, Kalman filtering allows us to extract a better estimate of a tracked vehicle's true position, making the system more resistant to sensor imprecision.

6 Discussion

6.1 Comparison to LIDAR

Table 2 compares our localization performance to a state-of-the-art LIDAR vehicle-detection system, *STD: Sparse-to-Dense 3D Object Detector for Point Cloud*[19]. As expected, LIDAR outperforms our model at all difficulties. However, Figure 6 shows a compelling result when we relax the constraints on how precise the detections need to be. The performance of our model increases dramatically when we relax the detection distance threshold to 3 m. The takeaway is that although our system lacks the fine-grained precision of LIDAR, it can still adequately track most vehicles.

Our system achieves comparable precision to LIDAR for vehicles that are "Easy" to detect, but performs significantly worse for more difficult vehicles. Many of these difficult to detect vehicles are cars that are smaller than 40 pixels in image-space height, i.e. cars that are far away from the cameras. Part of our tracking difficulty likely comes from the inherent imprecision of stereo depth perception at longer ranges, where LIDAR possesses a significant resolution advantage. However, precise tracking at long range is often unimportant to autonomous vehicle tasks, where nearby objects pose the greatest collision risk. Because our system is comparable to LIDAR at close range, we believe it offers a compelling alternative to LIDAR.

6.2 Comparison to Monocular 3D object tracking

Table 2 compares our localization performance to a state-of-the-art Monocular 3D vehicle detection system, *Disentangling Monocular 3D Object Detection* by Simonelli, et al. [13]. Our system significantly outperforms the monocular object detector, achieving a 450% MAP improvement for "Easy" detections and a 270% improvement for "Medium" difficulty vehicles while performing comparably for "Hard" vehicles. Note that the monocular object detector MAP values are evaluated at a distance threshold of 2 m compared to our less-forgiving default threshold of 1.5 m; the performance gap would likely increase further at the same distance threshold.

6.3 2D vs. 3D object tracking

Figure 5 compares the accuracy of our 2D predictions to that of our 3D predictions. 2D detection performance is significantly greater than 3D detection performance. This is partially due to the "dimensionality curse"–the much larger scale of the 3D volume makes 3D localization an inherently more difficult problem.

However, the performance gap is also due to the limitations of our stereo ranging system. Our 3D detector performs much worse on vehicles that are significantly occluded. When objects are significantly occluded, our ranging method, which operates on the principle of median filtering, sometimes returns the distance to the *occluding* object rather than the occluded vehicle. We hypothesize that a more robust ranging algorithm could produce significant performance improvements for difficult vehicles. For example, if our model were to semantically segment the image, it could identify portions of the image that would produce the most accurate depth result, ignoring occluding objects.

6.4 Prediction Association

We examined several approaches for correlating raw 3D detections with tracked vehicles, known as "solving the data association problem". Compared to the methods used in the literature[17], our matching heuristic, which matches raw predictions to the nearest tracked vehicle, is relatively naive. Our algorithm was shown to perform well when vehicles were spread out, as in the case with highway driving, but less well when vehicles were closely clustered, such as when they were parked together alongside the road. We discussed using image embeddings of vehicles to create an appearance-based matching system, but we felt that this level of complexity was out of the scope of this project.

7 Conclusion/Future Work

We have implemented a successful stereo-camera-based 3D object tracking algorithm. Kalman filtering has shown to create a performance improvement over raw CV object detection, and our system has demonstrated comparable performance to LIDAR for nearby and fully-visible vehicles. Our performance could likely be significantly increased by implementing a more robust range-detection algorithm and a more robust prediction association algorithm.

8 Contributions

Anthony Li was the lead on implementation/tuning of Kalman and particle filtering along with the filter correlation algorithm.

Anthony Galczak was the lead on the initial implementation of C-based Darknet (YOLOv3), data grooming, and figures.

Eric Chan was the lead on the 2D object detection network, implementing 3D position extraction, and implementing evaluation metrics.

Anthony Galczak and Eric Chan are also utilizing this project for CS230. While much of this project is relevant to both courses, Kalman and particle filtering implementation and evaluation represent the main extension for CS238.

References

[1] (n.d.). Retrieved from http://www.cvlibs.net/datasets/kitti/eval_tracking.php.

[2] (n.d.). Retrieved from http://www.cvlibs.net/datasets/kitti/eval_object.php.

[3] Holger Caesar and Varun Bankiti and Alex H. Lang and Sourabh Vora and Venice Erin Liong and Qiang Xu and Anush Krishnan and Yu Pan and Giancarlo Baldan and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027, 2019.

[4] Waymo Open Dataset: An autonomous driving dataset. https://www.waymo.com/open. 2019.

[5] Dongbo Min, Sunghwan Choi, Jiangbo Lu, Bumsub Ham, Kwanghoon Sohn, and Minh N Do. Fast global image smoothing based on weighted least squares. Image Processing, IEEE Transactions on, 23(12):5638–5653, 2014.

[6] George Kour and Raid Saabne. Real-time segmentation of on-line handwritten arabic script. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[7] K. Bernardin, R. Stiefelhagen: Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. JIVP 2008.

[8] Andreas Geiger. 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (CVPR '12). IEEE Computer Society, Washington, DC, USA, 3354-3361.

[9] Mousavian, A., Anguelov, D., Flynn, J., & Kosecka, J. (2017). 3D Bounding Box Estimation Using Deep Learning and Geometry. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi: 10.1109/cvpr.2017.597

[10] Hirschmuller, H. (2005). Accurate and Efficient Stereo Processing by Semi Global Matching and Mutual Information. CVPR .

[11] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR (2016)

[12] Rangesh, Akshay. No Blind Spots: Full-Surround Multi-Object Tracking for Autonomous Vehicles Using Cameras & LiDARs. 23 Feb. 2018.

[13] Simonelli, Andrea et al. "Disentangling Monocular 3D Object Detection." ArXiv abs/1905.12365 (2019): n. pag.

[14] Rybski, Paul & Huber, Daniel & Morris, Daniel & Hoffman, Regis. (2010). Visual Classification of Coarse Vehicle Orientation using Histogram of Oriented Gradients Features. IEEE Intelligent Vehicles Symposium, Proceedings. 921 - 928. 10.1109/IVS.2010.5547996.

[15] Ginhoux, R. & Gutmann, Steffen. (2001). Model-based object tracking using stereo vision. Proceedings - IEEE International Conference on Robotics and Automation. 2. 1226 - 1232 vol.2. 10.1109/ROBOT.2001.932778.

[16] Marrón-Romera, Marta & Garcia Garcia, Juan Carlos & Sotelo, Miguel-Angel & Cabello, Mariana & Pizarro, Daniel & Huerta, Francisco & Cerro, Jaime. (2007). Comparing a Kalman Filter and a Particle Filter in a Multiple Objects Tracking Application. 1 - 6. 10.1109/WISP.2007.4447520.

[17] Janai, Joel. Computer Vision for Autonomous Vehicles: Problems, Datasets and State-of-the-Art. ArXiv, 18 Apr. 2017.

[18] W. Choi, C. Pantofaru and S. Savarese, "A General Framework for Tracking Multiple People from a Moving Camera," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 7, pp. 1577-1591, July 2013.

[19] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, Jiaya Jia; STD: Sparse-to-Dense 3D Object Detector for Point Cloud. The IEEE International Conference on Computer Vision (ICCV), 2019, pp. 1951-1960

[20] Wang, Yizhou. Object Detection on KITTI Dataset Using YOLO and Faster R-CNN. 20 Dec. 2018, http://yizhouwang.net/blog/2018/12/20/object-detection-kitti/.

[21] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6517-6525. doi: 10.1109/CVPR.2017.690

[22] Redmon, Joseph & Farhadi, Ali. (2018). YOLOv3: An Incremental Improvement.

[23] Asvadi, Alireza. “YOLOv2 416x416 Detection Framework Experiment on KITTI.” The KITTI Vision Benchmark Suite,

[24] Himmelsbach, Michael, et al. "LIDAR-based 3D object perception." Proceedings of 1st international workshop on cognition for technical systems. Vol. 1. 2008.

[25] Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.