This ICCV workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Real-time category-based and general obstacle detection for autonomous driving

Noa Garnett Shai Silberstein Shaul Oron Ethan Fetaya
Uri Verner Ariel Ayash Vlad Goldner Rafi Cohen Kobi Horn Dan Levi
Advanced Technical Center Israel, General Motors R&D
Hamada 7, Herzlyia, Israel
{noa.garnett,shaul.oron,uri.verner,ariel.ayash,vlad.goldner,rafi.cohen,dan.levi}@gm.com

Abstract

Detecting obstacles, both dynamic and static, with near-to-perfect accuracy and low latency, is a crucial enabler of autonomous driving. In recent years obstacle detection methods increasingly rely on cameras instead of Lidars. Camera-based obstacle detection is commonly solved by detecting instances of known categories. However, in many situations the vehicle faces un-categorized obstacles, both static and dynamic. Column-based general obstacle detection covers all 3D obstacles but does not provide object-instance classification, segmentation and motion prediction. In this paper we present a unified deep convolutional network combining these two complementary functions in one computationally efficient framework capable of real-time performance. Training the network uses both manually and automatically generated annotations using Lidar. In addition, we show several improvements to existing column-based obstacle detection, namely an improved network architecture, a new dataset and a major enhancement of the automatic ground truth algorithm.

1. Introduction

Autonomous vehicles depend on detecting static and dynamic obstacles in real-time and predicting their behavior with no room for error. High resolution Lidar has been the sensor to go in this domain for most projects aiming Level 5 automation such as the winning entries in the DARPA Urban Challenge [33] and Google's self driving car [14]. Recently, more projects aim at a camera-centric approach [23, 2, 1] due to the disadvantages of high-resolution Lidars (cost, packaging, moving parts) and the boost in computer vision accuracy. To fully or partially replace Lidars, vision-based obstacle detection should reach at least the same performance. We divide the task to two main sub-tasks: categorized and general obstacle detection.

Category-based detection (aka detection and classification), has been extensively studied in computer vision [7, 10] leading to dramatic performance improvements in recent years [28, 6, 22]. In the common scenario, during deployment, a bounding box and class is marked for each object belonging to a set of pre-defined classes. In addition, the object's pose may be assigned [25]. This is particularly useful since knowing the class and pose of the object is instrumental to predicting its behavior and motion. However, objects outside the pre-defined class set are not detected.

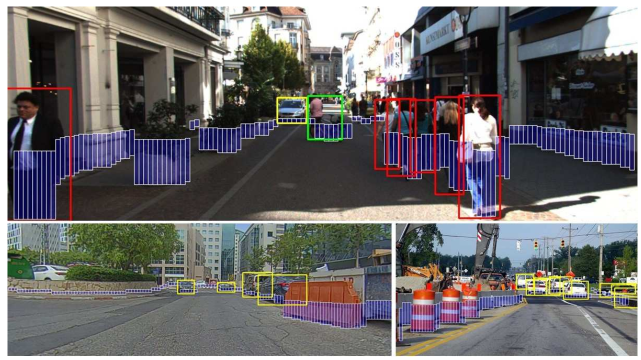

Figure 1. **Unified method output examples.** General obstacles are marked by blue bars. The bottom position of each bar is determined by the max probability bin of the StixelNet branch (See Section 2.1). Bar height is shown for display purposes only, and computed as a linear function of the bottom image position. Cars, pedestrians and cycles are marked as bounding boxes and color coded by their class. Notice in the examples that real world driving involves complex scenarios requiring both types of obstacles to be detected.

A complementary enabling technology is free-space or general obstacle detection [3, 19]. The task is to identify in each image column the row position of the nearest *roughly vertical* obstacle. In our formulation obstacles consist of every object higher than a typical curb. This allows detecting all relevant static and dynamic obstacles, both on the road and on the sidewalk. Several examples include construction zone delimiters, shopping carts, un-categorized animals and toddlers. We introduce a method performing both types of obstacle detection in one unified network capable of running in real-time (30fps) and sharing most computation between the tasks. Figure 1 shows detection examples of our network in three different scenes. Notice the complementary nature of the two capabilities. In addition, for each categorized object, the network is capable of estimating its pose with negligible additional computation cost.

Our contributions are as follows. First, we introduce a novel unified network architecture for categorized object detection, pose estimation and general obstacle detection running in real time. The architecture is single-shot and learned end-to-end. It combines Single-Shot Multi-Box Detector (SSD) [22] for categorized object detection and pose estimation, with our version of the StixelNet [19] for general obstacles. Training data consists of images with both manually and automatically generated ground truth (GT). Second, we introduce several significant improvements to the StixelNet: a new network architecture, a new automatic ground truth method for generating training data and the use of a newly collected dataset. In our experiments, we improve state-of-the-art general obstacle detection. The combined multi-task network maintains single-task networks accuracy while sharing most computation between the tasks. Finally, we show that column-based general obstacle detection can be generalized to cope with substantially different cameras and viewing angles.

1.1. Related Work

Network architectures for modern object detectors are divided to single shot [22, 27] and region-proposal based [28, 6]. The output of such detectors is a tight bounding box around each object instance. Recently, it has been shown that with little computational overhead rich additional information can be extracted for each instance. This includes object pose [25], instance segmentation [15] and 3D bounding box estimation [4]. For our combined network we built our architecture on the SSD [22] which was shown to have the best accuracy when processing time is the first priority [16].

There exist several approaches for general obstacle or free space detection. We follow the **column-based approach** [3, 34]. In particular we further develop and improve the StixelNet monocular approach introduced in [19]. This representation of the free space is both compact and useful since it can be efficiently translated to the occupancy grid representation commonly used by autonomous agents. A contrasting approach is based on **pixel-level road segmentation** [24, 32]. It has the advantage of detecting free space areas not immediately reachable at the expense of an over-parametrization: an output for each pixel. [26] detect unexpected obstacles using a joint geometric and deep learning approach. In [29] stereo-based stixels and pixel-level segmentation are combined to obtain "Semantic Stixels": Stixels with class labels. The most related to our work is MultiNet [32], a combined network for road segmentation and object detection. In comparison our approach uses column-based detection and operates over two times faster on a comparable hardware.

We believe automated ground truth **(AGT)** methods will be instrumental in next generation automotive computer vision. Their main appeal is the ability to collect large amounts of training data with a relatively low effort. Recent benefits of the approach were shown in learning monocular depth estimation from Lidar [18] and Stereo [11]. [19] have shown the effectiveness of AGT for general obstacle detection. Lidar is very efficient at detecting free-space and accuracy can be obtained by ignoring low confidence columns. In this paper we introduce a new and improved algorithm for the this task.

Finally, while numerous datasets with vehicle on board imagery exist [13, 5] only few exist from low mounted fisheye lens cameras [20]. Such setup is in wide use for surround view applications and requires special adaptation of computer vision methods. We introduce a new such dataset with automatically generated GT. The remaining of the paper is organized as follows: We start with the description of the combined network followed by an experimental validation on multiple datasets and conclusions.

2. Combined network for obstacle detection

We next describe our architecture for each of the three tasks we address: general obstacle detection, object detection and pose estimation. We then present the combined network architecture and its training procedure.

2.1. General obstacle detection

Our network for general obstacle detection is derived from the *StixelNet* [19]. The network gets as input an image of arbitrary width and predefined height $I_h = 370$. The final desired result is the pixel location y of the closest obstacle bottom point in each image column (with stride s). As in the original StixelNet the network outputs for each column are represented by k **position neurons**, representing the centers of k equally spaced bins in the y image axis. The output of each neuron represents the predicted probability that the obstacle position is in the respective bin.

An important addition we introduce is the ability to handle two edge cases: the presence of a near obstacle truncated by the lower image boundary ("near obstacle"), and no obstacle presence in the column ("clear"). This was not previously handled since the Automatic Ground Truth (AGT) [19] did not detect such columns and hence they were not included in the training set. Our AGT described below does handle these cases, however since the representation of these in the training set is extremely imbalanced, we introduced the following modification to the network: additional per-column **type neurons** with three possible output values: “near obstacle”, “clear” and “regular” according to the aforementioned cases. This dual per-column representation is combined to one position output as follows: if the type is one of the edge cases, the first or last position neurons probability is set to the type probability respectively. The rest of the position neurons are re-scaled proportionally s.t. the probabilities sum equals 1. During training, Softmax-loss is used for the type-neurons, while the *PL-loss* [19] is used for the position neurons. *PL-loss* has been shown to be effective in regression problems such as ours, which require a multi-modal representation while being able to preserve order information between neighboring position neurons.

Following [19] we use AGT to detect the obstacle position in each image column using the Lidar point cloud. The original AGT suffers from several drawbacks which we address. The proposed AGT has two main differences: an object-centric obstacle bottom position estimation and column-type detection. The target of AGT is to bring fully reliable and consistent GT while covering as many columns as possible. The new AGT, described in detail next, provides high reliability while covering a much larger percent of the columns as shown in the experimental section.

The AGT is composed of two algorithms: one detects for each obstacle its *bottom contour in the image*, and the second detects columns which are certain to be *clear of obstacles*. Both algorithms get as input the 3D point cloud, from which ground plane points are removed by fitting a plane model. We start by describing rest of the stages of the first algorithm.

The 3D point cloud is separated to clusters in 3D. Obstacles (clusters) with maximal height (position above ground) below 20cm are ignored. The algorithms continues processing each cluster *separately*. 3D Points are projected onto the image and dilated resulting in a dense point blob. Let I_B be the binary image of all lowest points in each image column. I_B is smoothed by a Gaussian kernel. Finally, a 1-dimensional conditional random field is applied to find an optimal trajectory with maximal value in I_B and minimal y-position discontinuity. This trajectory is considered the obstacle's bottom contour. An example result is depicted in figure 2.

Note that the Lidar points do not cover the entire image. Therefore a special handling is required for near objects whose bottom is below the Lidar coverage. We first detect such cases which occur when the bottom most Lidar point of an obstacle cluster is above the ground. Then, we project these points to the road plane in 3D and add them to the cluster.

The second part of the AGT consists of detecting “clear” columns. There are two conditions to be met for such a column: all points in it (projected from 3D) are lower than 5cm and there exist points beyond 18 meters in distance.

Figure 2. Example of the Lidar-based automatic ground truth for obstacle detection. Each obstacle instance is colored uniquely and obstacle bottom contour is marked in white.

The second condition prevents columns to be classified as clear although there is a close dark object absorbing the Lidar beams.

When training on an cropped image patch, if bottom-most cropping position is above or at the GT bottom of a valid column, then it is classified “near obstacle” (e.g. right most object in figure 2). Compared to the automatic ground truth procedure described in [19] which operates directly in the image domain our object-centric approach takes advantage of the obstacles continuity to output a more complete and smooth ground truth annotation.

We next describe the method for object detection and pose estimation, trained using manual annotations, in contrast to the general obstacle detection.

2.2. Category-based object detection

Our object detection is based on the SSD framework [22] which provides an excellent run-time vs. accuracy trade-off. The network is trained with four classes: vehicles, pedestrians, cycles (bicycles + motorcycles) and background. We found a slight modification to the ground truth bounding box association in the training procedure to improve the network accuracy. When a GT bounding box is associated to a proposal, a hard limit on the overlap ratio is originally used to classify the proposal as true or false. In our version a buffer zone in the overlap ratio value (0.4-0.6) is defined in which proposals are ignored. This helps preventing ambiguities in the association process and better defines class and background train samples. In addition we modify the learning such that difficult examples are ignored instead of treated as background. We used a version of the code provided by the authors in which we optimized the network deployment efficiency.

2.3. Object pose estimation

Object pose is defined for a car by its heading direction angle in top view Θ_H in camera-centric coordinates. The pose angle is defined as $\Theta_P = \Theta_H - \Theta_C$ where Θ_C is angle of the line passing through the camera and car center position. For representation in the network Θ_P is discretized to 8 equally spaced bins between 0 and 2π. For each bounding box proposal, the network outputs probability of each angle bin. A cyclic version of the *PL-loss* is used to train the output layer against the continuous ground-truth Θ_P. Supporting the cyclic nature of the output *cyclic-PL-loss* considers first and last bins as neighboring, such that angles close to zero contribute both. The SSD architecture is modified by adding per proposal, 8 pose neurons, similar to those for class and box regression.

2.4. The combined network

The combined network architecture is illustrated in 3. The feature extraction layers are based on the GoogLeNet[31] backbone. From this, two main branches split: object/pose (SSD+Pose) and general obstacle detection (StixelNet). The StixelNet branch is trained with AGT as described previously while the object detection, classification and pose estimation are trained with manually labeled data. Inspired by GoogLeNet [31] and VGG [30], our version of StixelNet uses a deeper architecture than the one described in [19]. Feature map sizes in Illustration 3 correspond to a 800×370 image. Note however that the two branches may operate on different image sizes by cropping feature maps accordingly when branching out.

The combined network objective loss is defined as a linear combination of the SSD object classification, bounding box regression, pose estimation, stixel-position and stixel-type losses with relative weighing of 1, 0.5, 0.5, 1, 1 respectively. These weights were experimentally set to minimize accuracy loss on each task in the combined network. We start all training sessions with the GoogLeNet layers pre-trained on the ImageNet [8] as provided by the authors. We found that fine-tuning the network with the combined objective to produce degraded results. Instead, we first train the SSD branch without pose, then fix all weights and train pose neurons only, then fix again and train the StixelNet branch, and finally allow all network weights to freely learn the combined objective loss.

3. Experimental results

We compare the new version of the StixelNet with the original one [19], and the combined network with partial networks each specializing in a single task. Since different tasks require specialized types of annotation we have multiple data-sets for training and testing each task as summarized in Table 1. All training datasets are used for the relevant tasks and are ignored for all other tasks during training. We show results on each test set according to available annotations for that set. Notice the public domain test datasets are actually validation sets we separated from the training data. In the **kitti-stixels** dataset we follow the same separation to train and test as in [19].

Data-set name / source	Stixels	Objects	Pose	# images	# instances
kitti-objects-train [13]		✓	✓	5K	15K
Cityscapes [5]		✓		3K	15K
TDCB [21]		✓		9K	15K
Caltech-peds [9]		✓		32K	53K
internal-objects-train		✓		3K	19K
internal-pose-train			✓	155K	160K
kitti-stixels-train [12]	✓			6K	5M
internal-stixels-train	✓			16K	20M
kitti-objects-test [13]		✓	✓	891	3.5K
internal-objects-test			✓	1K	8K
internal-stixels-test	✓			910	19K
kitti-stixels-test [12]	✓			760	11K

Table 1. Datasets used in paper

Three datasets in 1 were internally collected: **object-internal**, **pose-internal** and **stixel-internal**. The first two, used for object detection and pose estimation respectively were collected with a roof top mounted camera similar in setting to the **kitti-dataset** [13] and manually labeled. The **stixel-internal** dataset is aimed at short range general obstacle detection with fisheye-lens camera mounted in typical production surround vision systems position. To obtain accurate automatic ground truth we mounted a Velodyne HDL64 Lidar right below the camera as depicted in Figure 4. Co-locating the sensors eliminates differences stemming from viewpoint variation. The camera is triggered to capture an image every time the Lidar is pointed directly forward. Each image is corrected before processing by virtually rotating the camera to forward view, and un-distorting it to a cylindrical image plane. Figure 1 bottom left shows detection results on an image from the test portion of this dataset.

3.1. Implementation details

Implementation was done in the Caffe framework [17]. All training was carried out with extensive geometric data augmentation with crop, mirror and warp operations. For training, the SSD patches were cropped with width ranging from $320-1280$ and height $160-800$ such that their aspect ratio (width/height) was between 1.5 and 2.5, and warped to 800×400 training patches. For the StixelNet, patches were cropped with width $760-840$ and height $360-380$, and warped to 800×370. Kitti images are roughly 370 pixels height so there was no augmentation in the vertical axis. However, the internal dataset image height is 800 pixels allowing extensive variation in vertical position. We used an augmentation scheme in which the horizon vertical position varies by ± 50 pixels making the StixelNet less sensitive to

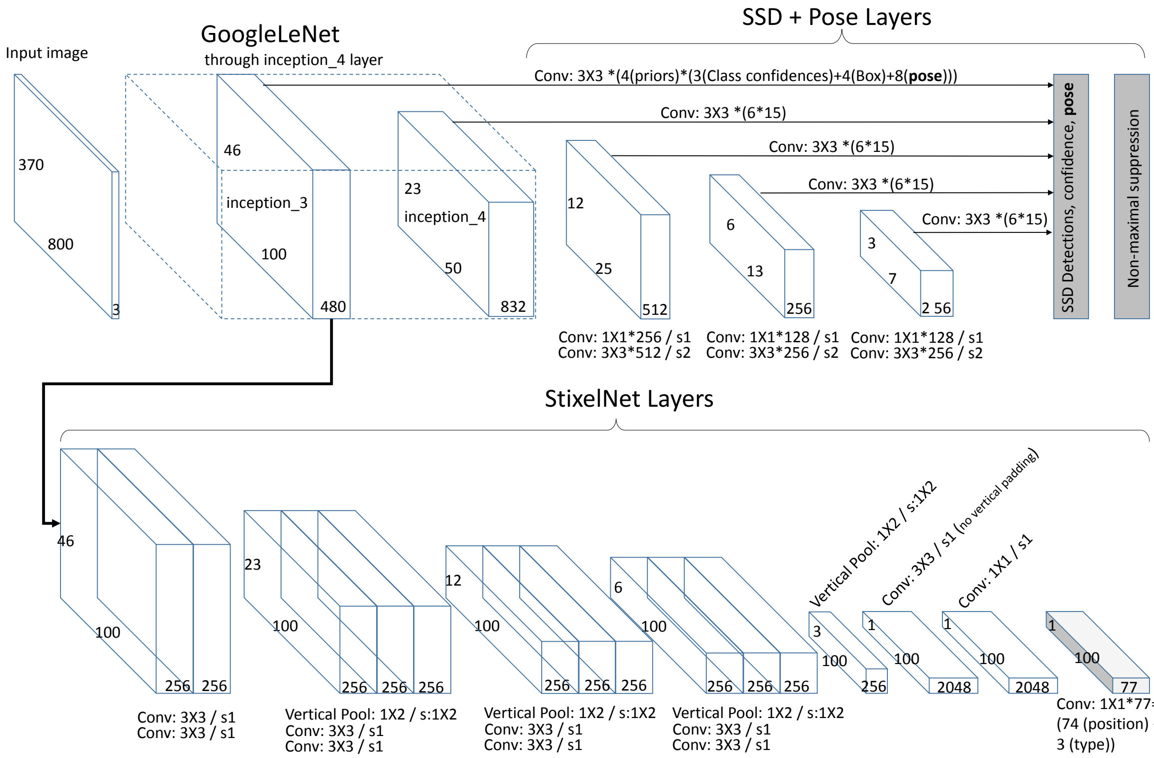

Figure 3. Combined SSD, Pose and StixelNet architecture.

Figure 4. General obstacle detection data collection setup for automatic ground truth. A fisheye-lens camera is mounted in typical production position (height 80cm, tilt 20 degrees downward). A Velodyne Lidar is mounted below the fisheye-lens camera.

the exact horizon position.

3.2. Combined network

Table 3.2 summarizes accuracy results vs. run-time on all tasks with the combined network and its subsets. Run-time is measured in milliseconds per 800×370 pixel frame on Nvidia Quadro M6000 GPU. Best accuracy per task is in bold. The **SSD-only** method is only trained for object detection. Accuracy is reported as area under curve (AuC) of the precision-recall following exactly KITTI evaluation protocol [13] in the "moderate" difficulty setting. A 0.5 minimal intersection over union overlap is required for correct bounding box detection. The **SSD + pose** adds pose detection functionality. Pose estimation accuracy is measured for correctly detected cars following the kitti-objects [13] methodology as well.

The **StixelNet-only** method is trained for general obstacle detection only. Results are reported for each of the two test sets separately and using the two measures described in [19]: AuC for correct predictions at maximal probability, and average probability.

The result show that combined method accuracy on each task is on-par or slightly degraded compared to each single task method. Two out of the eight measures suffer a 4.4% degradation in accuracy while for the remaining ones it is negligible. In terms of run-time, running the combined net-

	Accuracy measure									Run-time
DataSet:	{kitti, internal}-objects-test			kitti-objects-test	kitti-stixels-test		internal-stixels-test			
Test:	Car	Pedestrian	Bicycle	Pose	Max Pr.	Avg. Pr.	Max Pr.	Avg. Pr.		ms/f
SSD-only	**0.901**	0.574	**0.560**							27
SSD + pose	0.900	**0.578**	0.541	0.890						28
StixelNet-only					**0.854**	**0.824**	**0.827**	**0.774**		19
Combined net	*0.900*	*0.570*	*0.536*	**0.892**	*0.825*	*0.788*	*0.824*	*0.772*		*33*

Table 2. Accurracy vs. run-time measures (in milliseconds per frame) on the three tasks for the combined net and its subsets. See text for more details on accuracy measures and datasets.

work versus the SSD-only adds 20% to the total run-time, meaning most computation is shared between the tasks. At 30 frames per second (33ms/f) the combined network is most suitable for run-time even with more power efficient GPUs.

3.3. General obstacle detection

Due to the improved AGT, dataset and architecture the general obstacle detection module performs significantly better than the originally proposed StixelNet. Figure 5 illustrates these differences on some examples from the kitti-stixels-test set. Most apparent are, not surprisingly, for the edge cases (near object and clear column). The significant qualitative improvement has much to do with the new AGT which provides a much more complete coverage: 81% (internal-stixels-dataset) and 69% (kitti-stixels-dataset) are provided with valid ground truth. For comparison the original AGT provides a 25% coverage on the kitti-stixels-dataset. Specifically, the edge cases were almost completely excluded in the previous AGT.

In table 3 we show consistent performance improvement on the new test set compared to the original StixelNet. To have a fairer comparison we also show the results when edge cases are excluded from the test set, since the original StixelNet was not trained with such cases. A full ablation study to single out the different factors for the improvement is prohibitive since the new network architecture, training set, test set and AGT are coupled together, and in fact solve a slightly altered, more complete problem than the original one. We compared the effect of the backbone change in the architecture by altering the original StixelNet to use the same backbone as in our implementation (GoogLeNet[31]). The network was trained and tested on the original test set from [19]. Results show a marginal improvement in the accuracy (1% or less in all measures) indicating the backbone is not an important factor in the improvement.

To test the transferability of StixelNet from one camera setup to another we do a cross examination of the method trained and tested on both kitti and internal datasets. Note the large difference: a roof mounted forward camera versus a low mount fisheye-lens tilted downwards. As summarized in table 4 highest accuracy on each test set is at-

	kitti-stixels-test		internal-stixels-test	
	Max Pr.	Avg. Pr.	Max Pr.	Avg. Pr.
Original	0.689	0.671	0.681	0.654
Ours	**0.854**	**0.824**	**0.827**	**0.774**
Edge cases excluded from test				
Original	0.779	0.758	0.704	0.678
Ours	**0.827**	**0.807**	**0.819**	**0.760**

Table 3. Comparison of original StixelNet [19] and ours on stixel test sets

tained when trained only on the corresponding train set or on the combined one. Having negligible degradation in accuracy when trained on the entire dataset suggests that the network learned a general representation transferable to different cameras.

Train / Test	kitti-stixels-test		inter.-stixels-test	
	Max P.	Avg P.	Max P.	Avg P.
kitti-stixels-tr.	**0.855**	**0.827**	0.740	0.721
inter.-stixels-tr.	0.685	0.679	**0.833**	**0.776**
both	0.854	0.824	0.827	0.774

Table 4. Generalization across datasets

4. Conclusions and future work

We presented a unified network with real-time detection capability for both categorized and uncategorized object. The network is trained with a combination of manual and automatic ground truth based on Lidar. Our novel automated ground truth (AGT) algorithm covers most image parts facilitating the learning of a generic obstacle detection module. Using the new AGT, in combination with a new network architecture and dataset our version of the StixelNet improves state-of-the-art column-based general obstacle detection. We believe future research should focus on obtaining a unified AGT process that covers all aspects of obstacle detection.

References

[1] https://www.tesla.com/blog/all-tesla-cars-being-produced-now/

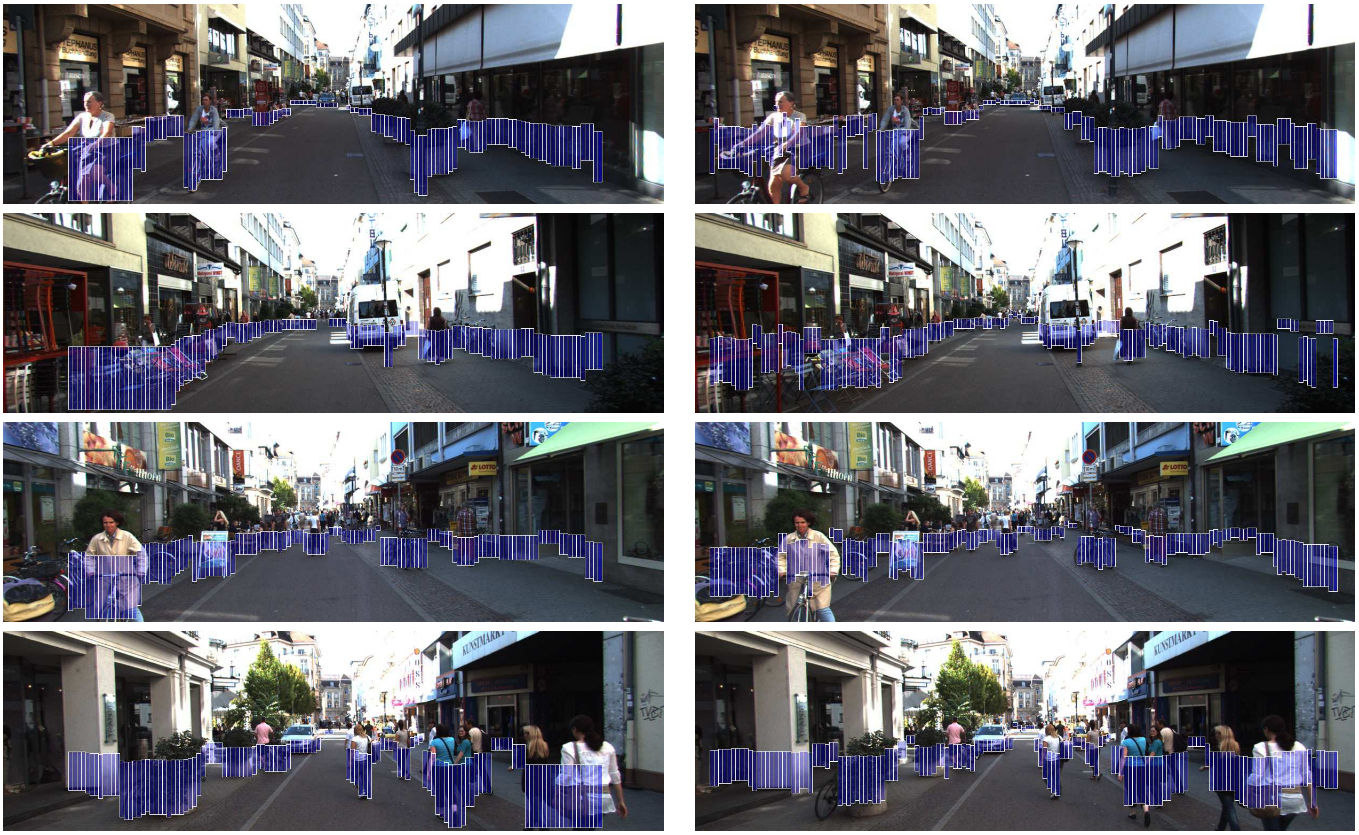

Figure 5. StixelNet example results on kitti-test dataset. **Left:** Ours, **Right:** [19].

-have-full-self-driving-hardware. 1

[2] http://www.mobileye.com/future-of-mobility/history-autonomous-driving. 1

[3] H. Badino, U. Franke, and D. Pfeiffer. The stixel world - a compact medium level representation of the 3d-world. In *Proceedings of the 31st DAGM Symposium on Pattern Recognition*, pages 51–60, Berlin, Heidelberg, 2009. Springer-Verlag. 1, 2

[4] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. Chateau. Deep MANTA: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. *CoRR*, abs/1703.07570, 2017. 2

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 4

[6] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 379–387, 2016. 1, 2

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society. 1

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 4

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *CVPR*, June 2009. 4

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, Sept. 2010. 1

[11] R. Garg, B. G. V. Kumar, G. Carneiro, and I. D. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII*, pages 740–756, 2016. 2

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 4

[13] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2, 4, 5

[14] E. Guizzo. Very deep convolutional networks for large-scale image recognition. *IEEE Spectrum*, October, 2011. 1

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. *arXiv preprint arXiv:1703.06870*, 2017. 2

[16] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016. 2

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 4

[18] Y. Kuznietsov, J. Stückler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[19] D. Levi, N. Garnett, and E. Fetaya. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 109.1–109.12. BMVA Press, September 2015. 1, 2, 3, 4, 5, 6, 7

[20] D. Levi and S. Silberstein. Tracking and motion cues for rear-view pedestrian detection. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 664–671, Sept 2015. 2

[21] X. Li, F. Flohr, Y. Yang, H. Xiong, M. Braun, S. Pan, K. Li, and D. M. Gavrila. A new benchmark for vision-based cyclist detection. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 1028–1033, June 2016. 4

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1, 2, 3

[23] R. Metz. Autox has built a self-driving car that navigates with a bunch of 50usd webcams. *MIT Technology Review*, March, 2017. 1

[24] G. L. Oliveira, W. Burgard, and T. Brox. Efficient deep models for monocular road segmentation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4885–4891, Oct 2016. 2

[25] P. Poirson, P. Ammirato, C. Fu, W. Liu, J. Kosecka, and A. C. Berg. Fast single shot detection and pose estimation. *CoRR*, abs/1609.05590, 2016. 1, 2

[26] S. Ramos, S. K. Gehrig, P. Pinggera, U. Franke, and C. Rother. Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. *CoRR*, abs/1612.06573, 2016. 2

[27] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 779–788. IEEE Computer Society, 2016. 2

[28] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015. 1, 2

[29] L. Schneider, M. Cordts, T. Rehfeld, D. Pfeiffer, M. Enzweiler, U. Franke, M. Pollefeys, and S. Roth. Semantic Stixels: Depth is not enough. In *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 110–117, Piscataway, NJ, 2016. IEEE. 2

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 4

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, June 2015. 4, 6

[32] M. Teichmann, M. Weber, J. M. Zöllner, R. Cipolla, and R. Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. *CoRR*, abs/1612.07695, 2016. 2

[33] C. Urmson, J. Anhalt, J. A. D. Bagnell, C. R. Baker, R. E. Bittner, J. M. Dolan, D. Duggins, D. Ferguson, T. Galatali, H. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, A. Kelly, D. Kohanbash, M. Likhachev, N. Miller, K. Peterson, R. Rajkumar, P. Rybski, B. Salesky, S. Scherer, Y.-W. Seo, R. Simmons, S. Singh, J. M. Snider, A. T. Stentz, W. R. L. Whittaker, and J. Ziglar. Tartan racing: A multi-modal approach to the darpa urban challenge. Technical Report CMU-RI-TR-, Pittsburgh, PA, April 2007. 1

[34] J. Yao, S. Ramalingam, Y. Taguchi, Y. Miki, and R. Urtasun. Estimating drivable collision-free space from monocular video. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 420–427, Jan 2015. 2