Joint Unsupervised Depth-Estimation and Obstacle-Detection from Monocular Images

Raphael Pisoni
Free University of Bolzano
Bolzano, IT 39100
raphael.pisoni@gmail.com

Abstract

Until recently monocular depth estimation has been dominated by supervised models but newer approaches have shown that self-supervised monocular depth estimation can achieve similar and in some cases even better results while at the same time using only cheap monocular or stereo imagery. Obstacle detection and segmentation on the other hand is currently done almost exclusively by supervised models using manually generated ground truth data because unsupervised approaches do not yet provide the needed quality. Using previous best practices and a set of new methods with our proposed model we significantly improve the state of the art in self-supervised monocular depth estimation and show that these results can be used to teach the same model to learn obstacle segmentation. We show the effectiveness of a U-Net variation and a number of modified loss functions, introduce a system that leverages inclination and roughness of the predicted depth to learn obstacle segmentation from monocular images and demonstrate the quality of our results on the KITTI benchmark as well as through visually compelling obstacle maps generated by our model.

1. Introduction

Monocular depth estimation is a nontrivial problem in computer vision due to the fact that any image can map to an infinite number of possible depth configurations. While originally it was tackled using mostly geometric approaches, since the advent of convolutional neural networks significant progress has been achieved in this area. Until recently this task was treated mostly as a supervised problem and networks were trained on large datasets of images and their corresponding laser-scans. Godard et al. [1] introduced self-supervised monocular depth estimation from stereo images by enforcing consistency between the left and right image. This approach has a number of advantages including the vast availability of stereo footage and the much lower cost of producing it, as well as its high quality and low sensor-noise and error-rate compared to LIDAR scans. Today multiple ways to estimate depth using self-supervised approaches from stereo [1] or motion parallax [2] have been described and have been shown to outperform traditional geometric and sometimes even supervised models [1]. In fact we will show in this work that self-supervised monocular depth estimation has gotten so good that the resulting estimations can be used to derive more complex measures. We propose to use an efficient but powerful model as well as some improvements in loss computation and training to significantly improve the state of the art in monocular depth estimation.

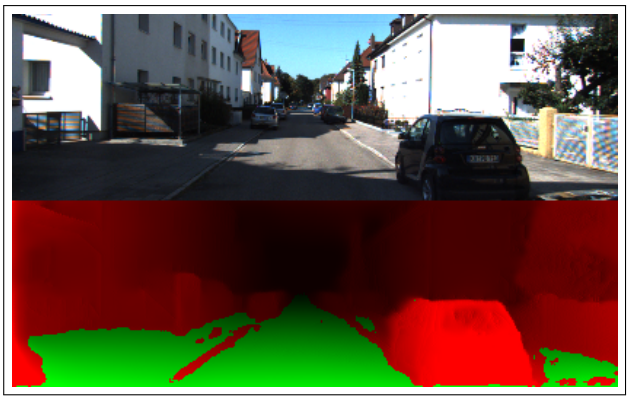

Figure 1. Original Image and Product of Estimated Disparity- and Obstacle-map.

In addition to depth-estimation we use a novel algorithm that uses angle and roughness of the predicted depth to train the our model to perform self-supervised obstacle detection with interesting results. In the past obstacle detection has mostly been treated as a general object detection problem from camera images, LIDAR measurements or a combination of both. While there has been limited suc-

cess for unsupervised approaches most of them were supervised and therefore required a large number of human-generated labels that the network could learn. This is very resource-intensive and depending on the quality of the labels can be very error prone. Object detection is usually not done through segmentation but through architectures that estimate the position and size of a bounding box for an object. This is done due to the fact that segmentation is relatively inefficient for a large number of object classes. Other approaches have managed to achieve good results through street segmentation or lane detection purely from images which reduces the number of classes but might miss some kinds of obstacles without using depth information. Mancini et al. [3] tried to combine the depth estimation and obstacle detection tasks and were able to show that this can be beneficial to both tasks, however their architecture was resource intensive and used bounding boxes which are of limited utility in real world scenarios. The exact implementation details of our approach will be made public[1].

2. Previous Work

As the notion of depth is a prerequisite for many tasks in computer vision and robotics there has been extensive research in the area of depth measurement and estimation. While currently the best direct results are achieved with modern laser scanners, their cost, weight and size is prohibitive for many areas where depth information could be of great use. In contrast cameras are passive sensors that are much smaller, cheaper and can be built to produce high quality images under most circumstances. With the rising success of learning based methods Eigen et al. [4] have shown the feasibility of depth estimation as a supervised learning problem while others were able to refine this approach [5] to even surpass traditional methods in some cases [6]. To reduce the need for ground truth depth data, that is often hard to acquire under significantly varying real world conditions, weakly supervised approaches have tried to exploit additional information in the form of sparse supervision, unpaired depth supervision [7], known object sizes [8] or other supervised appearance matching terms [9], however the need for high quality depth data or other annotations still limits these approaches. Similarly synthetic data, while being a valuable alternative, is not trivial to generate in the required amounts and quality, especially including varying weather and lighting conditions.

2.1. Self-supervised Stereo Training

As an alternative to supervised training, self-supervised learning of depth as an intermediate step during the image reconstruction between two images has emerged [10]. Following works used left-right consistency between two synchronized stereo pairs [1] or similar techniques for monocular videos [11]. While using stereo images at training time provides generally good results, they might not always be available. In contrast monocular videos are ubiquitous but many models using them encounter the problem of not handling movement well. While some approaches like egomotion estimation and individual tracking of moving objects exist [2], to solve these problems they introduce considerable complexity and computational overhead at training time and during online learning and refinement.

[1]Code repository to be announced.

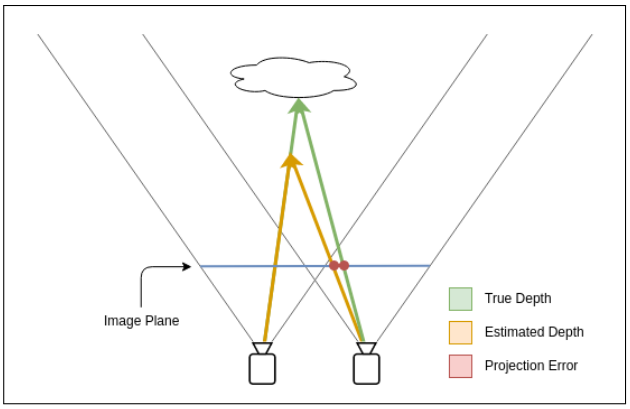

Figure 2. Concept of self-supervised depth estimation. A misestimation in the depth of the left image can lead to discrepancies between the true and the projected right images.

2.2. Joint depth estimation and obstacle detection

Path-finding and obstacle detection are closely connected and are in many cases the main reason to use any kind of depth perception. As both play an important role in self-driving technology and robotics in general, a large amount of work has been devoted to these topics, however most authors approach them as problems for supervised learning and segment obstacles separately from images [12] or LIDAR [13]. Another line of work has achieved noteworthy results by using unsupervised features like roughness and angle of the detected surface normal from LIDAR-scans [14, 15]. These are the concepts that come closest to our framework even though during inference we segment obstacles directly from monocular images using a convolutional neural network and do not use LIDAR measurements.

3. Methods

The main objective of our setup is the self-supervised learning of depth estimation and the segmentation of possible obstacles. The sole source of supervision for both tasks during training are pairs of rectified stereo images with known intrinsic and extrinsic parameters. In the following we will describe in detail the problem setup, the architectural details of our model as well as the objective functions

and compare them to architectures proposed by other authors.

3.1. Problem Setup

The input during training are two stereo RGB images $(I_1, I_2) \in R^{H \times W \times 3}$ as well as the extrinsic camera parameters $(C_1, C_2) \in R^{4 \times 4}$ for them. The model θ is a fully convolutional neural network that takes a single image I_i as input and predicts a depth $D_i \in \mathbb{R}^{H \times W}$ and a map of obstacles $O_i \in R^{H \times W}$. Similar to Zhou et al. [16] and Casser et al. [2] through the depth estimated by θ and a fully differentiable warping operation ϕ we can project the images I_1 and I_2 between the camera viewpoints defined by C_1 and C_2 such that $\hat{I}_{i \to j}$ denotes the j-th image constructed by warping the i-th image into the j-th position. The training signal can either be computed through photometric losses between the projected image $\hat{I}_{i \to j}$ and the original image I_j (Figure 2) such as the reconstruction loss $L_{rec} = |\hat{I}_{i \to j} - I_j|$, or through losses like our obstacle detection loss that are derived from the estimated depth.

3.2. Model Architecture

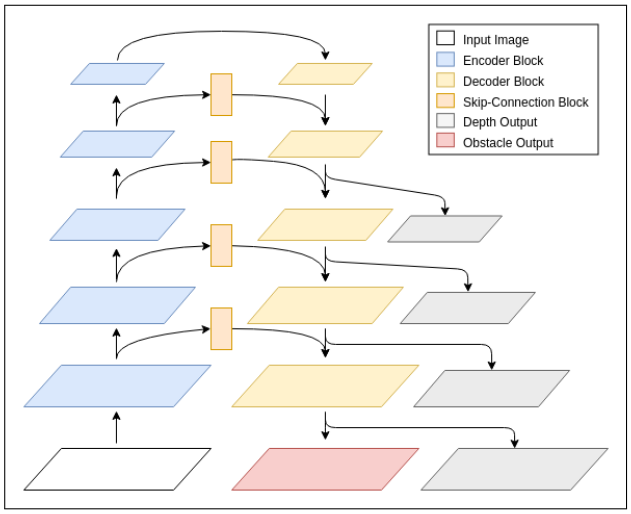

Figure 3. General Model Architecture. U-Net like structure with convolutional blocks in the skip-connections, depth-outputs on multiple scales and a final branch for obstacle detection.

The basic model architecture we use can be seen in Figure 3 and is based on a U-Net architecture [17] that consists of encoder-decoder structures with skip-connections, and applies best practices introduced in previous works [2, 18]. As an encoder ResNet18 is used [19] and as a decoder we use an improved variation of the one used by Godard et al. [18]. We use skip connections augmented with additional blocks consisting of two convolutional layers that are then reconcatenated with their input-tensor and output the estimated depth at four scales through a single convolutional layer with sigmoid activation. Then we apply a rescaling operation of the following form: $D = 1/(a * \sigma + b)$, where σ is the output from the sigmoid activation and a and b are chosen as 10 and 0.01 to constrain the output range of D. In all other layers we use RELU nonlinearities, and reflective padding instead of zero padding to reduce border artifacts. The obstacle segmentation branch consists of a padding layer and two convolutional layers and branches out before the last convolutional layer of the depth estimation network. In total the model has little over 18M parameters which allows for relatively efficient training and execution.

3.3. Objective Functions

For the depth estimation branch we use a relative reconstruction loss and a structural similarity loss. The relative reconstruction loss is computed as the sum of the absolute difference of the left and right images warped onto each other using the estimated depth, which is then divided by the target image plus one to amplify errors in very dark regions of the image:

$$L_{rec} = \frac{|\hat{I}_{1 \to 2} - I_2|}{I_2 + 1} + \frac{|\hat{I}_{2 \to 1} - I_1|}{I_1 + 1} \quad (1)$$

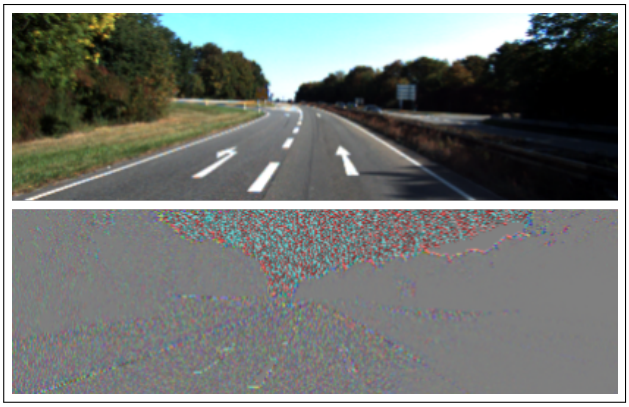

Figure 4. Image and its $atan2$ transformation.

In addition we use an SSIM loss proposed by Wang et al. [20], however instead of computing the structural similarity between the warped and the original image we compute the SSIM loss on the $atan2$ transformation of both images (Figure 4). This ensures high contrast even in under- or over-exposed images and amplifies small details that are making it easier for the SSIM loss to punish even small differences. To further improve the importance of close objects we weight the SSIM error by multiplying it with the SSIM between the unmodified left and right images plus one.

For both losses the error is only computed on areas that are not occluded during the warping operation. As a third loss function for the depth estimation task we choose to

use the extended scale invariant log RMSE introduced by Mancini et al. [3]

$$L_{depth} = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\Big(\sum_i d_i\Big)^2 + \frac{1}{n}\sum_i \Big(\nabla_x D_i + \nabla_y D_i\Big) \cdot N_j \tag{2}$$

where $d_i = \log D_i - \log D_j$, D_i and D_j are the left and right predicted depth warped onto each other. $\nabla_x D_i$ and $\nabla_y D_i$ are the horizontal and vertical predicted depth gradients of D_i and N_j is the 3D surface normal of the projected depth D_j.

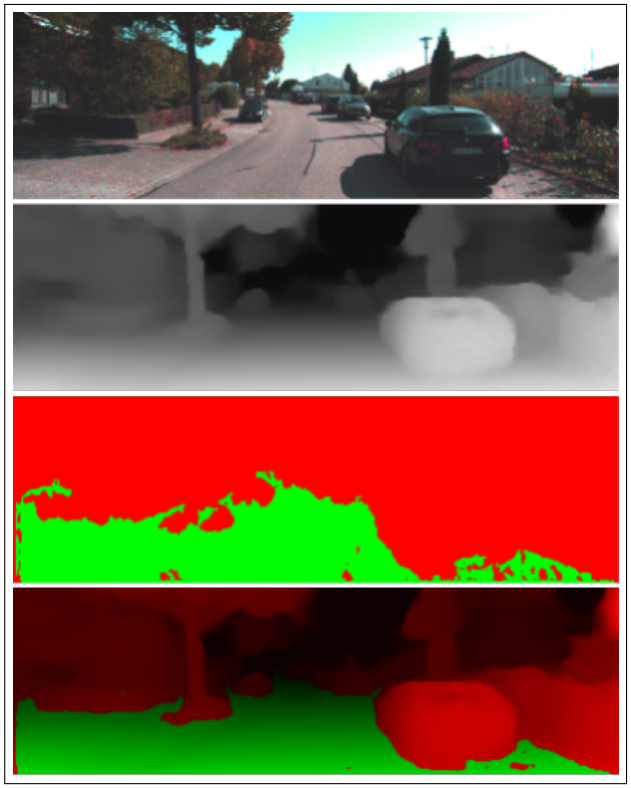

Figure 5. (from top to bottom) 1) Original Image, 2) Estimated Depth, 3) Computed Obstacle Map, 4) Product of Estimated Disparity and Estimated Obstacle Map.

While the first two terms correspond to the scale invariant log RMSE loss introduced by Eigen et al. [4], the third term was originally introduced to enforce the orthogonality between predicted gradients and a given ground truth normal, aiming to preserve geometric coherence by Mancini et al. [3]. We use this loss to verify the coherence between the left and right predicted depths.

A final loss function that is applied during training is an edge aware surface smoothness loss that punishes changes in the surface inclination that don't correspond to edges in the color gradients of the original image. This is done by taking the mean of the L2-norm between the surface normals of pixel p_k and its 8 neighbours $S(p_k)$ and dividing it by the two-dimensional image gradient of the input image.

In order to train the obstacle segmentation branch we generate a self-supervised obstacle map for every pixel p_k where $ob(p_k) = 0$ indicates drivable area and $ob(p_k) = 1$ indicates an obstacle. The values are derived from the roughness of the estimated depth $D(p_k)$ and the angle of the surface normal $\mathbf{N}(p_k)$. If we denote the 8 connected neighboring pixels of p_k as $S(p_k)$, a pixel is marked as obstacle if one of the following conditions is true:

$$\max_{p \in S(p_k)} (|D(p_k) - S(p_k)|) > \theta_1 \, (D(p_k))^2 \tag{3}$$

$$|D(p_k) - \bar{D}(p_k)| > \theta_2 \, (D(p_k))^2 \tag{4}$$

$$\arcsin\left(\frac{|N^y(p_k)|}{\|\mathbf{N}(p_k)\|}\right) < \theta_3 \tag{5}$$

where θ_1 and θ_2 are hyperparameters set to 0.006 and 0.003, whereas θ_3 is the maximum deviation angle from the vertical axis that we set to 82°. $\bar{D}(p_k)$ is the mean of the depth of $S(p_k)$ points. $N^y(p_k)$ is the projection of the norm $\mathbf{N}(p_k)$ on the y-axis. The intermediate obstacle maps are then combined via logical inclusive-or and we remove all non-obstacle patches that are smaller than 5% of the total image area (Figure 5). Finally the obstacle-loss is computed using the cross-entropy between the resulting map and the output of the obstacle-branch of the network where the weights for drivable and obstacle are 1.0 and 1.4. To keep obstacles consistent between left and right image and to remove small artifacts we also apply the SSIM loss on both estimated obstacles and their warped counterparts. The final loss is computed as a weighted sum off all the losses, applied to both stereo images:

$$L_{total} = \alpha \, L_{rec} + \beta \, L_{ssim} + \gamma \, L_{depth} + \delta \, L_{smooth} + \epsilon \, L_{obst} \tag{6}$$

where α, β, γ, δ and ϵ are hyperparameters that we decided to set to 1.0, 0.2, 0.002, 0.04 and 0.01. While at first it might seem more efficient to simply compute the ground truth obstacle map at inference time using the formulae listed above, we found that adding two additional layers as an obstacle branch to our network is not only faster but also leads to generally better results for obstacle detection with less noise and better left-right consistency, with an additional benefit of lower errors in the depth estimation.

3.4. Training

We train our model on the images from the KITTI Raw dataset and exclude all static scenes as well as the test-split used by Eigen et al. [4]. While we resize the images to a resolution of 416×128 pixels, we test the results on the ground

truth depth recreated at the original image resolution. We do this by interpolating the estimated depths bilinearly. Prior to training the encoder is pretrained on Imagenet and during training we augment images through horizontal flipping and by randomly varying the color space within the imagenet distribution. We use a batch-size of 4 and RAdam [21] as an optimizer with $\beta 1 = 0.9$, $\beta 2 = 0.999$ and add Lookahead [22] with $k = 5$. As an initial learning rate we use 0.0002 and halve it after every 200,000 batches without improvement. Even though the architecture of the model is relatively straightforward we have to apply a few tricks to allow it to train in a stable way. Our network contains few batch normalization layers in the decoder, which we assume helps with absolute scaling of the estimated depth outputs. This however makes it quite difficult to get the training started without exploding gradients. To mitigate this problem we start the training with five additional batch-normalization layers, located at the front of the decoder and between the decoder blocks. Once the model has started training successfully and the metrics have gone below a specific threshold we start removing the added normalization layers progressively starting from the end of the decoder stack. While without this system the model can start rarely if ever, with it starting the training is almost always successful. We train our Tensorflow implementation of the model for up to 50 epochs on a single RTX2080Ti which takes around 50 hours and select the checkpoint with the lowest training loss for testing.

4. Experimental Evaluation

In the following section we present results from our models trained on the KITTI Raw dataset created by Geiger et al. [23] and evaluated on the test-split proposed by Eigen et al. [4]. While we use a popular benchmark and common metrics in order to make the depth estimation results comparable, we are not aware of a benchmark to compare the results of our obstacle detection branch. Therefore we apply a naïve approach and compare its results to the 200 images of the training set for semantic segmentation of the KITTI benchmark. The applied training-set contains around 42K rectified stereo images from 61 different scenes and the test split consists of 697 stereo image pairs. The ground truths are produced by a Velodyne laser scanner with a range of up to 80 meters and are reprojected onto the left image at the original image resolution, that slightly varies around 1392 $\times$ 512 pixels. In general ground truth data is used for evaluation only and we resize the estimations to the original resolution through bilinear interpolation. When displaying our obstacle segmentation results we combine them with the disparity, which being the inverse of the depth, helps to better display nearby details, while depth shows details equally over the whole distance, which would be less suitable for obstacle detection. As shown in Table 1 we are able to significantly improve the state of the art for self-supervised depth estimation and as shown in Figure 6, models using the obstacle-segmentation branch are mostly able to produce convincing and reasonably detailed obstacle maps under varying circumstances. We test different variants of our model for Joint Unsupervised Depth and Obstacle Estimation (JUDO) and show a detailed ablation study for a number of loss variants in Table 2. While all models deliver comparably good results, we show that the addition of the various loss functions, as well as convolutional blocks in the skip connections and the addition of the obstacle-detection branch lead to a slight refinement of the model.

To create a baseline for the obstacle segmentation branch we compare its results to the semantic segmentation benchmark included in the KITTI dataset. While our definition of 'obstacle' might be negotiable, for the sake of simplicity we assume all classes that are part of the 'flat' category not to be obstacles. This includes the labels road, sidewalk, parking and rail-track. Evaluation is done on the 200 images of the training set and the model is able to achieve an mean IOU-score of 20.0191. This evaluation is certainly not ideal since for example the dataset does not consider curbs while the model does. Furthermore the model often has problems with sharp color-changes and tends to classify them as obstacles even though they lie on a flat surface. Nevertheless these results are promising and can be built upon.

Finally even with convolutional blocks in the skip connections and active obstacle detection branch the model still runs at over 70 frames per second during single-image inference on a single RTX2080Ti.

5. Conclusion

We have presented a model for the self-supervised learning of depth-map prediction as well as obstacle segmentation from monocular images using only aligned stereo images during training. Applying current best practices as well as some novel loss variations and techniques we are able to significantly improve the current state of the art in self-supervised depth estimation and show that self-supervised obstacle segmentation can achieve promising results. This should be encouraging for future research in these areas since supervised approaches in both areas require considerable technical and financial resources and might under some conditions not be feasible at all.

In the future we would like to refine the self-supervised obstacle detection approach, in addition to approaches for better temporal consistency of our results as well as simultaneous localization and mapping with included obstacle detection. Also the application of our system for semi-supervised learning of obstacle segmentation could be a promising topic for future exploration.

Table 1. Evaluation and comparison of monocular depth estimation results on the KITTI dataset and test-set used by Eigen et al. [4]. As supervision they use ground-truth (D), motion (M), or stereo (S). All results use a maximum range of 80 meters. For pink columns lower is better while for yellow columns higher values are better.

Method	Sup	Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
Eigen et al. 2014 [4]	D	0.214	1.605	6.563	0.292	0.673	0.884	0.957
Eigen et al. 2014 [4]	D	0.203	1.548	6.307	0.282	0.702	0.890	0.958
Zhou et al. 2017 [16]	M	0.208	1.768	6.856	0.283	0.678	0.885	0.957
Yang et al. 2017 [11]	S	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Godard et al. 2017 [1]	S	0.141	1.369	5.849	0.242	0.818	0.929	0.966
Godard et al. 2018 [18]	S	0.133	1.158	5.370	0.208	0.841	0.949	0.978
Yang et al. 2018 [24]	S	0.137	1.326	6.232	0.224	0.806	0.927	0.973
Yang et al. 2018 [24]	M	0.131	1.254	6.117	0.220	0.826	0.931	0.973
Casser et al. 2018 [2]	M	0.109	0.825	4.750	0.187	0.874	0.958	**0.983**
$\text{JUDO}_{\text{full}}$ (ours)	S	**0.0892**	**0.7217**	**4.2156**	**0.1729**	**0.9172**	**0.9641**	0.9809

Table 2. Ablation study of monocular depth estimation results on the KITTI dataset and test-set used by Eigen et al. [4]. Tested modificatin in subscript. For pink columns lower is better while for yellow columns higher values are better.

Method	Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
$\text{JUDO}_{\text{no reconstr. loss}}$	0.0959	0.8716	4.4808	0.1781	0.9085	0.9621	0.9806
$\text{JUDO}_{\text{reconstr. loss not relative}}$	0.0933	0.7592	4.3149	0.1762	0.9074	0.9626	0.9810
$\text{JUDO}_{\text{no ssim loss}}$	0.0982	0.7695	4.4951	0.1821	0.8956	0.9595	0.9805
$\text{JUDO}_{\text{no normals loss}}$	0.1016	0.9311	4.7859	0.1893	0.8967	0.9574	0.9782
$\text{JUDO}_{\text{no smooth loss}}$	0.0946	0.7413	4.3008	0.1769	0.9072	0.9623	0.9811
$\text{JUDO}_{\text{smooth loss not edge-aware}}$	0.0930	0.7522	4.2843	0.1768	0.9082	0.9622	0.9805
$\text{JUDO}_{\text{no obstacle detection}}$	0.0931	0.7487	4.3083	0.1755	0.9076	0.9629	**0.9813**
$\text{JUDO}_{\text{no conv. skip blocks}}$	0.0924	0.7464	4.2837	0.1750	0.9092	0.9628	0.9811
$\text{JUDO}_{\text{keep batch-norm. layers}}$	0.1139	1.2679	4.7633	0.1959	0.8878	0.9524	0.9756
$\text{JUDO}_{\text{no image-ssim weighting}}$	0.0962	0.7812	4.3598	0.1784	0.9045	0.9617	0.9809
JUDO	**0.0892**	**0.7217**	**4.2156**	**0.1729**	**0.9172**	**0.9641**	0.9809

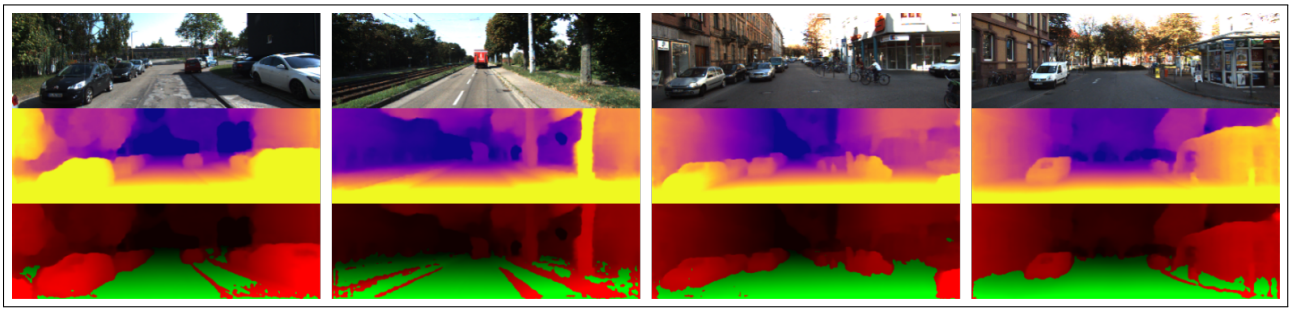

Figure 6. (top to bottom) 1) Original Image, 2) Estimated Depth, 3) Obstacle Disparity Product

References

[1] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611, 2017). 1, 2, 6

[2] Vincent Casser, Sören Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. *CoRR*, abs/1811.06152, 2018). 1, 2, 3, 6

[3] Michele Mancini, Gabriele Costante, Paolo Valigi, and Thomas Alessandro Ciarfuglia. J-mod2: Joint monocular obstacle detection and depth estimation. *IEEE Robotics and Automation Letters*, 3:1490–1497, 2018). 2, 4

[4] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014). 2, 4, 5, 6

[5] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. *2016 Fourth International Conference on 3D Vision (3DV)*, pages 239–248, 2016). 2

[6] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:2144–2158, 2014). 2

[7] Jogendra Nath Kundu, Phani Krishna Uppala, Anuj Pahuja, and R. Venkatesh Babu. Adadepth: Unsupervised content congruent adaptation for depth estimation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2656–2665, 2018). 2

[8] Yiran Wu, Sihao Ying, and Lianmin Zheng. Size-to-depth: A new perspective for single image depth estimation. *CoRR*, abs/1801.04461, 2018). 2

[9] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:65:1–65:32, 2016). 2

[10] Ravi Garg, G VijayKumarB., and Ian D. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016). 2

[11] Zhenheng Yang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency. *CoRR*, abs/1711.03665, 2017). 2, 6

[12] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:2481–2495, 2016). 2

[13] Jilin Mei, Biao Gao, Donghao Xu, Wen Yao, Xijun Zhao, and Huijing Zhao. Semantic segmentation of 3d lidar data in dynamic scene using semi-supervised learning. *CoRR*, abs/1809.00426, 2018). 2

[14] Zi yi Liu, Si yu Yu, Xiao Wang, and Narming Zheng. Detecting drivable area for self-driving cars: An unsupervised approach. *CoRR*, abs/1705.00451, 2017). 2

[15] Yigong Zhang, Shuo Gu, Jian Yang, Jose M. Alvarez, and Hui Kong. Fusion of lidar and camera by scanning in lidar imagery and image-guided diffusion for urban road detection. *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 579–584, 2018). 2

[16] Tinghui Zhou, Matthew R. Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, 2017). 3, 6

[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015). 3

[18] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. *CoRR*, abs/1806.01260, 2018). 3, 6

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016). 3

[20] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004). 3

[21] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *ArXiv*, abs/1908.03265, 2019). 5

[22] Michael Ruogu Zhang, James Lucas, Geoffrey E. Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back. *ArXiv*, abs/1907.08610, 2019). 5

[23] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *I. J. Robotics Res.*, 32:1231–1237, 2013). 5

[24] Zhenheng Yang, Yang Wang, Wei Xu, and Ramakant Nevatia. Lego: Learning edge with geometry all at once by watching videos. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 225–234, 2018). 6