Self-supervised learning of depth maps for autonomous cars

Andrei-Sebastian Petrescu
CEOSpaceTech
Univ. Politehnica of Bucharest
Bucharest, Romania
andrei.petrescu1812@upb.ro

Constantin-Cristian Damian
CEOSpaceTech
Univ. Politehnica of Bucharest
Bucharest, Romania
constantin.damian91@upb.ro

Daniela Coltuc
CEOSpaceTech
Univ. Politehnica of Bucharest
Bucharest, Romania
daniela.coltuc@upb.ro

***Abstract*—In autonomous driving, the knowledge of scene depth is essential for the control of car navigation. The depth can be measured by dedicated sensors like sonar, radar, LiDAR, ToF camera or obtained by computation from videos collected by an RGB camera. In this paper, we propose a novel solution for estimating depth maps from videos by using a neural network with self-supervised learning. The novelty consists in using ground truth position measured by IMU sensors on-board instead of motion estimated pose. The fusion of data from IMU sensors and RGB camera results in a lighter network and shorter training times. Results on depth accuracy and odometry are given after tests on KITTI dataset.**

***Index Terms*—Depth map, self-supervised learning, neural networks, structure-from-motion, automotive.**

I. Introduction

At autonomous cars, the typical architecture of the automation system has two distinct parts: the Perception system, which estimates the car position, and the Decision Making system that controls the navigation. Depending on the level of autonomy, the tasks of the automation system can go from simple warnings (level 0), to the partial or complete control of the car (levels 3 to 5). The intermediate levels 1 and 2 are reserved to driver assistance in traffic and can include cruise control, lane positioning, highway assistance etc. [1].

The automation system operates with data collected by various in-car sensors e.g., 2D and 3D cameras, radar, GPS, LiDAR, Inertial Measurement Unit (IMU) etc. The Perception system processes all these data to calculate static maps of the environment, estimates the car position in relation to these maps, tracks the surrounding moving objects, or detects and recognizes the traffic signalization.

Among in-car sensors, the depth sensors constitutes a distinct category. By using various physical principles, they are able measure the distance to the objects or persons surrounding the car [2]. Sonar (SOund Navigation And Ranging), in its active version, emits sound pulses and read echoes returned from physical surfaces. Mounted on cars, it can detect objects up to 5.5 m. It works on bad weather, fog or by night, newer versions have resolution compared to that of LiDAR and is relatively cheap. Radar (RAdio Detection And Ranging) is the most common depth sensor used on cars. It is likewise the sonar, an active sensor but uses pulses of radio waves instead of sounds. The radar sensors are mounted all around the car in such a way to detect objects at any angle. They are used for automatic distance control and brake assistance. The LiDAR (Light Detection And Ranging) sensors work similarly to radar, the main difference being the use of lasers instead of radio waves. Unlike radars, the LiDAR creates a full 360-degree map around the vehicle rather than relying on a narrow field of view. This advantage makes from the LiDAR the option preferred by autonomous vehicle manufacturers despite the significantly higher cost. Time-of-Flight (ToF) camera is a 3D camera that collects depth maps by measuring the round trip travel time of infrared light from the camera to the object. The depth range is up to 40 m and the accuracy is several millimeters. Nowadays, ToF cameras come to use in driver assistance technologies, emergency brake systems and pedestrian protection.

Sonar, radar, LiDAR or ToF camera are sensors that directly measure the depth of scenes. A further alternative is to obtain the depth by computation, from the side effects of other sensors like motion in videos or defocus blur in images taken with a 2D camera. There is a plethora of methods for inferring depth from videos, all under the generic name of Structure-from-Motion (SfM). These methods inherit the limitations of cameras i.e., they do not work well in low visibility conditions, like fog, rain or nighttime but they come with the advantage of providing dense depth maps instead of more or less sparse point clouds. Moreover, they use a RGB camera, a sensor already existing on cars and working for other tasks too.

As in many other areas, Deep Learning has brought a lot of progress in SfM due to the ability of learning data-driven models that outperform in terms of fidelity the classical ones. A particular interest has been in self-supervised learning methods that do not require ground truth (GT) depth for the training of the neural network (NN).

One of the earliest solutions was proposed by Zhou et al. in [3]. Inspired by prior works on view synthesis, the authors have proposed an end-to-end solution for learning dense depth maps from videos. The current frame is estimated by warping on a neighbouring frame and the loss function is build on the difference between the original and the estimate. A secondary network estimates in parallel, camera relative pose between the two frames. Specific problems like frame untextured regions that generate uncertainty, moving objects, which breaks the

TABLE I
ACCURACY OF DEPTH ESTIMATED BY NN WITH SELF-SUPERVISED LEARNING.

Network	Abs. Rel.	Sq. Rel.	RMSE	RMSE log.	$\delta > 0.25$	$\delta > 0.25^2$	$\delta > 0.25^3$
Zhou, T.H. et al. [3]	0.159	1.347	5.789	0.234	0.796	0.933	0.973
Godard, C. et al. [4]	0.115	0.903	4.863	0.193	0.877	0.959	0.981
Casser, V. et al. [5]	0.108	0.825	4.750	0.186	0.873	0.957	0.982
Klingner, M. et al. [6]	0.113	0.835	4.693	0.191	0.879	0.961	0.981
Zhou, J. et al. [7]	0.121	0.837	4.945	0.197	0.853	0.955	0.982
Johnston, A. et al. [8]	0.106	0.861	4.699	0.185	0.889	0.962	0.982
Li, Y. et al. [9]	0.098	0.810	4.672	0.177	0.890	0.964	0.983
Alleoti, F. et al. [10]	0.119	1.239	5.998	0.212	0.846	0.940	0.976
Wang, C. et al. [11]	0.159	1.347	5.789	0.234	0.796	0.933	0.973

working hypothesis of scene rigid motion, as well as artefacts like texture-copy, depth drift, incomplete structures or edge bleeding, have triggered numerous improvements.

Godard et al [4] used for their monodepth2 a simple auto-masking method that filters out the pixels without apparent motion or moving differently than the rest of the scene. More recent works use segmentation maps to learn distinct motion models for the moving objects in the scene [12], [5], [6]. Further, the frame warping is done by combining these models and the camera ego-motion. To improve the depth accuracy and reduce texture-copy and depth drift artefacts, some networks include self-attention modules that favor the use of larger contexts in estimating the depth. [7], [8], [9].

At the level of network architecture, the dominant solution is the U-Net with a pre-trained encoder and completed by a pose estimation network [3]. Another alternative has been GANs [10], [13], [14]. In [10] for instance, the generator estimates the depth map of the current frame, an estimate of this frame is built by warping using depth and is provided to the discriminator that tries to take a decision about its genuineness. The network is trained to provide better and better estimates such to cheat the discriminator, while the discriminator learns how to distinguish better between an estimated and a genuine frame. Architectural innovations have been done also for the U-Net based solution. In [7], the depth network, which is a deep NN of low resolution, is doubled by a high resolution shallow network, also of encoder-decoder type. The two networks are coupled by a self-attention module. The high resolution network is used to extract the fine-grained details, which are combined with global features from the low resolution network. The result is a high-resolution and high-accuracy depth map. Neither the pose network of the generic solution was speared of innovation. In [11], the authors replace this network by a direct visual odometry module that calculates the pose by using Lucas-Kanade algorithm. The list of SfM methods based on NN with self-supervised learning is long. Table I only summarizes the performances of the above mentioned ones in order to give a flavor about the capability of this approach. All the networks in the table were trained on KITTI dataset.

In this paper we propose a solution that goes toward the simplification of U-Net based architecture, without significantly reducing the accuracy of depth maps. The idea is to drop the pose network and instead, to use the position measured by IMU, which is already installed on autonomous cars. We use as baseline for our proof, the network in [4].

By tests on KITTI dataset, we show that using IMU data instead of estimated pose, the network estimates better the depth in long range, while at short range there is only a negligible loss in accuracy. The ablation of pose network impacts also on the training stage that demands less resources in terms of dataset and running time. We also give results about the odometry and show some differences between the estimated and IMU measured car trajectory and orientation.

II. IMU AND CAMERA DATA FUSION FOR DEPTH MAP ESTIMATION BY SFM

In U-Net based architectures, the backbone is a standard fully convolutional encoder-decoder network called depth network, that is fed in by a video sequence and for each frame, outputs a depth map (Fig. 1). Only for training, a warping block is plugged into the network with the task of projecting the current frame I_1 onto a neighbouring frame I_2 and to calculate an estimate $\widehat{I_1}$ of it. The projection consists in recalculating each pixel position as follows [3]:

$$x_r = K\, T_{1\mapsto 2}\, K^{-1}\, \widehat{D}_1(x_t)\, x_t \tag{1}$$

where x_t is the pixel position in I_1 also called target, and x_r is the new position in the next frame I_2, called reference. K is the camera intrinsic matrix, $T_{1\mapsto 2}$ is the camera pose transformation between I_1 and I_2 and $\widehat{D}_1$ is the estimated depth map of I_1. In most cases, the results x_r is not an integer meaning that the projected pixel is landing between the pixels of I_2. In order to get an estimate for its intensity, I_2 is resampled by bilinear interpolation.

The differences between the actual values $I_1(x_t)$ and the estimated ones is the ”engine” of the learning process. At the level of loss function, it translates into the common solution adopted in self-supervised learning [4]:

$$L = \mu L_p + \lambda L_s \tag{2}$$

where L_p is a photometric loss that combines $L1$ norm and SSIM to compare I_1 with $\widehat{I_1}$, and L_s is a smoothing regularization based on the gradient of $\widehat{D}_1$. It is to note that no GT for depth is necessary to train the network.

Our contribution concerns the camera pose transformation $T_{1\mapsto 2}$ in eq. 1. This transformation has to be calculated with:

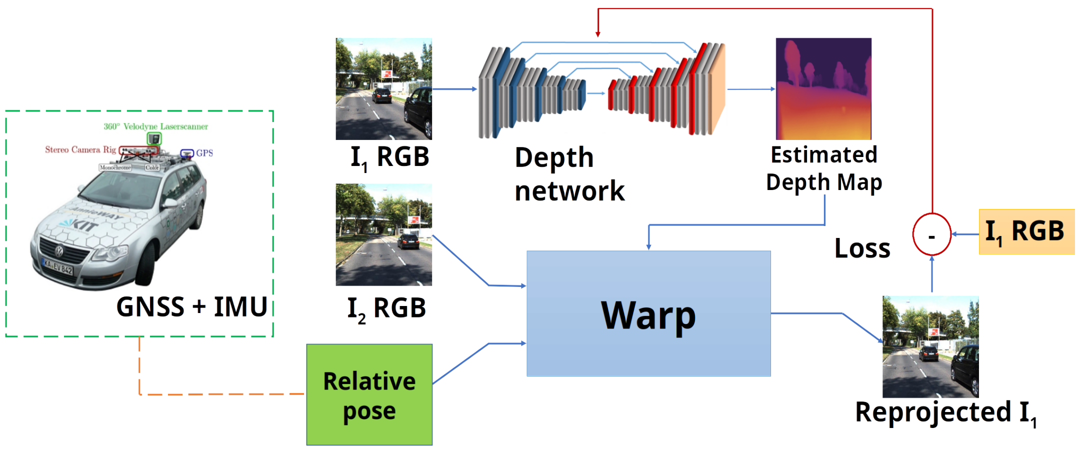

Fig. 1. **General scheme of $IMUcam$:** the depth network generates the depth map of every frame; the warping block calculates an estimate $\widehat{I}_1$ of the current frame (it is active only during the training); camera orientation and relative translation are provided by IMU processor directly to the warping block.

$$T_{1\mapsto 2} = T_{2\mapsto world}\, T^{-1}_{1\mapsto world} \tag{3}$$

where $T_{im\mapsto world}$ is the camera extrinsic matrix. The extrinsic matrix describes the camera location in the 3D world and its pointing direction. It takes the form of a rigid transformation consisting in a rotation matrix $R_{3\times3}$, and a translation vector $T_{3\times1}$:

$$T_{im\mapsto world} = \begin{pmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1 \end{pmatrix}_{4\times4} \tag{4}$$

The rotation matrix is calculated by using the camera attitude angles α, β and γ, called roll, pitch and yaw or Euler angles. The components of $T_{3\times1}$ are camera spatial coordinates (x_c, y_c, z_c) with respect to the navigation reference system. In $T_{1\mapsto 2}$, they become relative translations.

In [4], as in all other methods from the class of self-supervised learning SfM, the camera attitude and translation are estimated by a secondary NN, called pose network (Fig. 1). Likewise the warping block, this network is active only during the training stage. In autonomous cars equipped with an IMU, the angles and the relative translations already exist, they are calculated by IMU internal processor based on the data measured by gyroscopes, magnetometer and accelerometers.

In our approach, we eliminate the pose network that is redundant with IMU and instead, we supply the warping block with real data provided by IMU. By fusing data from multiple sensors - camera and IMU sensors - the depth network, that we shall call from now on $IMUcam$, becomes lighter, with less parameters to be learned. Besides, the incorporation of IMU data brings a supplement of ground truth in the network training. The impact on depth model accuracy as well as a comparison from the odometry point of view, are analysed in the next section by doing experiments on KITTI dataset.

III. Experimental results

In this section, we train $IMUcam$ and $monodepth2$ networks independently on KITTI dataset, then we analyse the differences at the level of odometry and depth estimation.

We use for training the odometry dataset of KITTI [15], containing 22 videos at 10 frames per second with a resolution of 1392×512 px. For 11 of the 22 video sequences, there are also GT position and orientation, measured by an OXTS RT3003 combined GNSS and IMU module. We split the dataset leaving 36671 frames for training, 4075 frames for validation and 697 frames for odometry tests (sequences 10 and 11). The tests for depth accuracy are done on Eigen split.

The training was performed on a computer with a 48 cores, 2.10 GHz, Intel(R) Xeon(R) Gold 6252 CPU with 786 GB of internal memory and Nvidia Quadro GV100 GPU with 5120 cores and 32 GB of GPU memory. We trained both networks for 40 epochs with a batch size of 12. The training method was ADAM with a learning rate of 10^{-4}, decreased by 10 every 15 epochs.

Figure 2 depicts the estimated and GT trajectory and orientation of the camera. The two trajectories do not overlap because of a significant drift between the estimate and the measurement. Noticeable errors also appear at the level of the orientation, especially for roll and yaw angles. The Root Mean Square Error (RMSE), calculated with a 5 samples sliding window, is 0.016 m for the trajectory and 7.642°for Euler angles (Euclidean norm of roll, pitch and yaw). Although high, these errors have a low impact on warping since the network is using only the relative pose of adjacent frames.

Table II shows the errors in depth estimation alongside the training time for both $monodepth2$ and $IMUcam$. Common metrics are used for depth accuracy: Absolute Relative Error (Abs. rel.), RMSE, their squared and logarithmic counterparts ($Sq.rel.$ and $RMSElog$), and the Thresholded Accuracy (δ), i.e. the probablity that the relative error is lower than a given threshold [16]. The last row shows the gain of $IMUcam$ by respect to $monodepth2$. Although negative, the differences are small. This might be because the measurements themselves have errors. For the training time, the gain is significant for $IMUcam$: 29% at the same training set.

Some examples of depth maps and corresponding frames

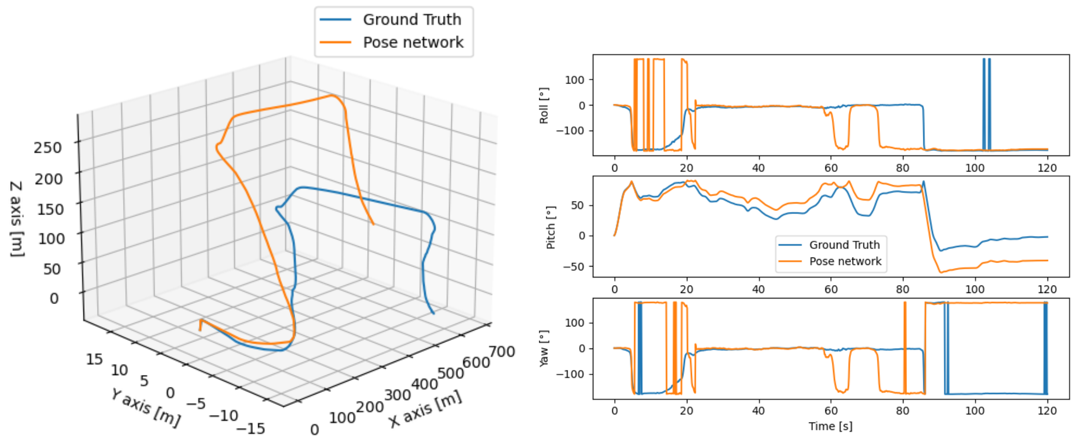

Fig. 2. Estimated (orange) and measured (blue) trajectories and Euler angles on sequence 10 from KITTI odometry dataset.

TABLE II
ACCURACY AND TRAINING TIME OF MONODEPTH2 AND IMUCAM.
↓MEANS LOWER IS BETTER, ↑MEANS HIGHER IS BETTER

Network	Abs. Rel. ↓	Sq. Rel. ↓	RMSE ↓	RMSE log. ↓	$\delta < 0.25$ ↑	$\delta < 0.25^2$ ↑	$\delta < 0.25^3$ ↑	Training time
Monodepth2	0.104	0.471	2.778	0.166	0.895	0.969	0.987	19h03m48s
IMUcam	0.106	0.496	2.846	0.169	0.887	0.968	0.986	13h30m28s
Gain	-0.002	-0.025	-0.068	-0.003	0.008	0.001	0.001	5h33m20s

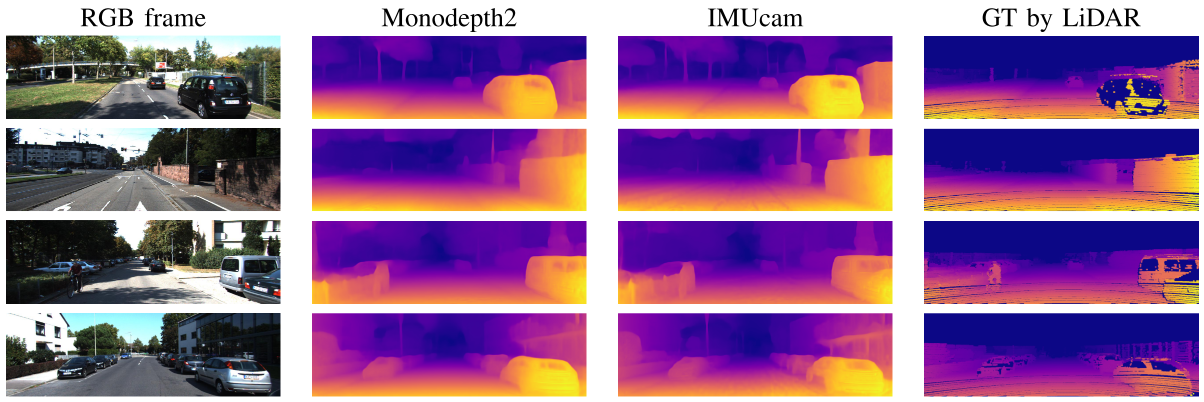

Fig. 3. Examples of frames from KITTI dataset and corresponding depth maps estimated by $monodepth2$, $IMUcam$ and measured by a LiDAR.

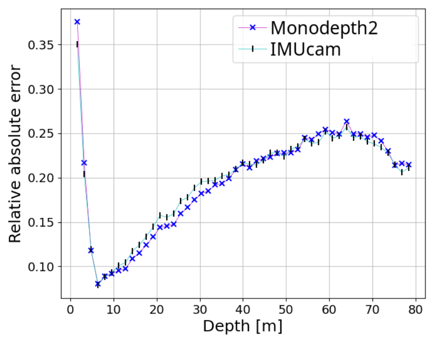

Fig. 4. Absolute relative error vs. depth. $IMUcam$ outperforms $monodepth2$ at long range.

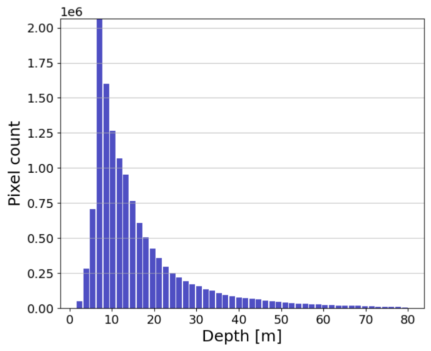

Fig. 5. Depth histogram in KITTI training set. The lower depths are better represented.

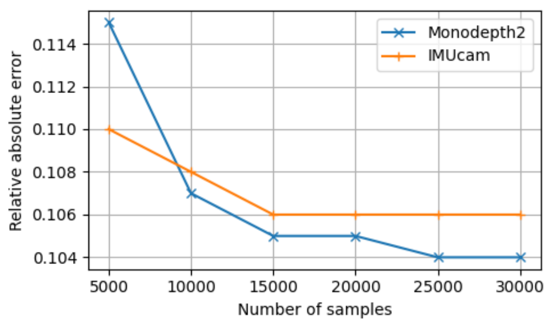

Fig. 6. Absolute relative error of estimated depth vs. number of samples. $IMUcam$ outperforms $monodepth2$ for reduced training sets.

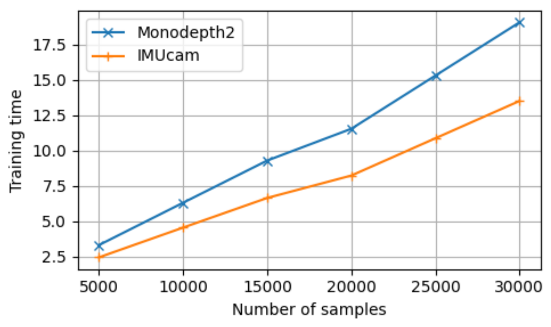

Fig. 7. $Monodepth2$ and $IMUcam$ training time vs. dataset size. $IMUcam$ is trained faster than $monodepth2$.

are presented in Fig. 3. On the last column, the are GT maps collected with a LiDAR. The black areas are invalid measurements. The differences between the maps estimated by $monodepth2$ and $IMUcam$ are almost imperceptible. However, the edges seem sharper in $IMUcam$ maps.

Figure 4 depicts the error of the predictions as a function of depth. $Monodepth2$ has an advantage between 10 and 40 m, a range that is better represented in the dataset, as shown by the histogram of depths in Fig. 5. Meanwhile, $IMUcam$ has an advantage at higher depths. This suggests that $IMUcam$ has a better capacity to generalize due to a lower complexity. To confirm this hypothesis, we used progressively less data in the training set and plotted the mean absolute relative error as a function of dataset size (Fig. 6). The same effect is present, $IMUcam$ outperforms $monodepth2$ for small sizes, less than $10,000$ samples. Figure 7 shows the training time as a function of dataset size. The training time increases almost proportionally with the dataset size. For $IMUcam$, it is always significantly shorter due to network lower complexity.

IV. Conclusion

We proposed a NN based solution for depth estimation from videos that uses data from the vehicle navigation system. It is basically a self-supervised network with the pose network ablated and warping block supplied with processed data from IMU sensors. This brings an advantage in terms of network complexity, which is significantly reduced. The immediate benefits are a shorter training time, a possibly smaller training set and a higher generalization capacity, all at the cost of a slightly lower depth accuracy at short range. In the future, we will explore the use of analytical visual odometry algorithms for improving the accuracy above the one of the of model using pose estimation.

Acknowledgment

This research was partially funded by the Ministry of Investments and European Projects through the Human Capital Sectoral Operational Program 2014-2020, Contract no. 62461/03.06.2022, SMIS code 153735.

References

[1] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixao, F. Mutz *et al.*, “Self-driving cars: A survey,” *Expert Systems with Applications*, vol. 165, p. 113816, 2021.

[2] M. Hirz and B. Walzel, “Sensor and object recognition technologies for self-driving cars,” *Computer-aided design and applications*, vol. 15, no. 4, pp. 501–508, 2018.

[3] T. Zhou, M. Brown, N. Snavely, and D. Lowe, “Unsupervised learning of depth and ego-motion from video,” 07 2017, pp. 6612–6619.

[4] C. Godard, O. M. Aodha, M. Firman, and G. Brostow, “Digging into self-supervised monocular depth estimation,” 2019.

[5] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova, “Unsupervised monocular depth and ego-motion learning with structure and semantics,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[6] M. Klingner, J.-A. Termöhlen, J. Mikolajczyk, and T. Fingscheidt, “Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance,” in *European Conference on Computer Vision*. Springer, 2020, pp. 582–600.

[7] J. Zhou, Y. Wang, K. Qin, and W. Zeng, “Unsupervised high-resolution depth learning from videos with dual networks,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6872–6881.

[8] A. Johnston and G. Carneiro, “Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume,” in *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2020, pp. 4756–4765.

[9] Y. Li, F. Luo, and C. Xiao, “Self-supervised coarse-to-fine monocular depth estimation using a lightweight attention module,” *Computational Visual Media*, vol. 8, no. 4, pp. 631–647, 2022.

[10] F. Aleotti, F. Tosi, M. Poggi, and S. Mattoccia, “Generative adversarial networks for unsupervised monocular depth prediction,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[11] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, “Learning depth from monocular videos using direct methods,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2022–2030.

[12] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova, “Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 8001–8008.

[13] A. Pilzer, D. Xu, M. Puscas, E. Ricci, and N. Sebe, “Unsupervised adversarial depth estimation using cycled generative networks,” in *2018 international conference on 3D vision (3DV)*. IEEE, 2018, pp. 587–595.

[14] A. CS Kumar, S. M. Bhandarkar, and M. Prasad, “Monocular depth prediction using generative adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 300–308.

[15] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: the kitti dataset,” *The International Journal of Robotics Research*, vol. 32, pp. 1231–1237, 09 2013.

[16] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” *Advances in neural information processing systems*, vol. 27, 2014.