Towards camera parameters invariant monocular depth estimation in autonomous driving

Karlo Koledić, Ivan Marković, and Ivan Petrović[1].

Abstract— **Monocular depth estimation is an effective approach to environment perception due to simplicity of the sensor setup and absence of multisensor calibration. Deep learning has enabled accurate depth estimation from a single image by exploiting semantic cues such as the sizes of known objects and positions on the ground plane thereof. However, learning-based methods frequently fail to generalize on images collected with different vehicle-camera setups due to the induced perspective geometry bias. In this work, we propose an approach for camera parameters invariant depth estimation in autonomous driving scenarios. We propose a novel joint parametrization of camera intrinsic and extrinsic parameters specifically designed for autonomous driving. In order to supplement the neural network with information about the camera parameters, we fuse the proposed parametrization and image features via the novel module based on a self-attention mechanism. After thorough experimentation on the effects of camera parameter variation, we show that our approach effectively provides the neural network with useful information, thus increasing accuracy and generalization performance.**

I. INTRODUCTION

Scene depth is a key information in many three dimensional reconstruction and perception tasks in robotics, autonomous driving, and virtual reality. While fusion of different sensor modalities increases robustness and accuracy, depth estimation from camera data is effective due to the richness of information and relative simplicity of the sensor setup. Traditionally, scene depth is estimated within geometric Structure-from-Motion or Visual Simultaneous Localization and Mapping frameworks. Sparse or dense correspondences are established across different camera poses, enabling triangulation and subsequent optimization. However, such systems usually calculate depth for a limited set of sparse correspondences with robustness issues due to challenging scenarios such as occlusions and textureless regions. Given that, deep learning-based methods have been increasingly used for monocular depth estimation (MDE). Even though depth estimation from a single image is an ill-posed problem, neural networks leverage large amount of data in order to learn semantic and geometric cues, such as the size of known objects or position on the ground plane [1], and use them to infer the scene depth.

Early MDE works [2], [3] establish a standard supervised learning procedure to directly regress a depth map within an encoder-decoder architecture, often with residual connections. Various attempts have been made in order to improve the results, with addition of recurrent neural networks [4]–[6], conditional random fields [7]–[10] or adversarial training [11], [12] into the architecture. Recently, with advancements of transformers [13] in vision tasks [14], many methods take advantage of the global receptive field of the transformer that naturally complements locality of the convolutions, thus consequently achieving state-of-the-art results [15]–[17]. However, the main drawback of such supervised methods is the necessity of ground truth data acquisition, which is often sparse and difficult to collect. This constrains the training data to a narrow distribution leading to overfitting and inaccurate generalization on unseen environments. To that end, self-supervised methods [18]–[21] use view synthesis of nearby frames as a supervision signal, removing the requirement of ground truth data during training.

Even though self-supervised methods make data collection within distinctive environments relatively straightforward, effects of different camera extrinsic and intrinsic parameters during test time are often ignored. As the training data is usually collected with a single vehicle-camera setup, networks tend to overfit due to the perspective geometry bias in gathered data [22]. Embedding of known focal length [23], camera intrinsics [24] or camera extrinsics [25] within neural networks, along with usage of diverse synthetic training data, has shown to improve generalization capabilities. Although synthetic data has been widely used for MDE in automotive scenarios [26]–[29], variation in camera parameters has been left largely unexplored. In theory, if trained on diverse enough real-world data containing various camera parameters, the network could learn to estimate depth for the camera parameters within the training set; however, we argue that the process of data acquisition with sufficiently diverse camera parameters in distinctive environments is infeasible, which is why we use synthetic data in the present work.

In this paper, we propose a novel approach for camera parameters invariant MDE for automotive driving scenarios. We demonstrate the effects of the camera parameters variation in MDE and design a novel architecture which enhances the generalization capabilities of the system. We test and train our method on synthetic data, while designing the architecture to support further work for domain adaptation to real data. Our main contributions are as follows:

- a novel parametrization of known camera intrinsic and extrinsic parameters as depth of the ground plane, which has a strong semantic and geometric meaning in MDE

[1] Authors are with University of Zagreb Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics, Zagreb, Croatia {name.surname@fer.hr}.

This research has been funded by the H2020 project AIFORS under Grant Agreement No 952275 and supported by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

979-8-3503-0704-7/23/$31.00 ©20212023 IEEE

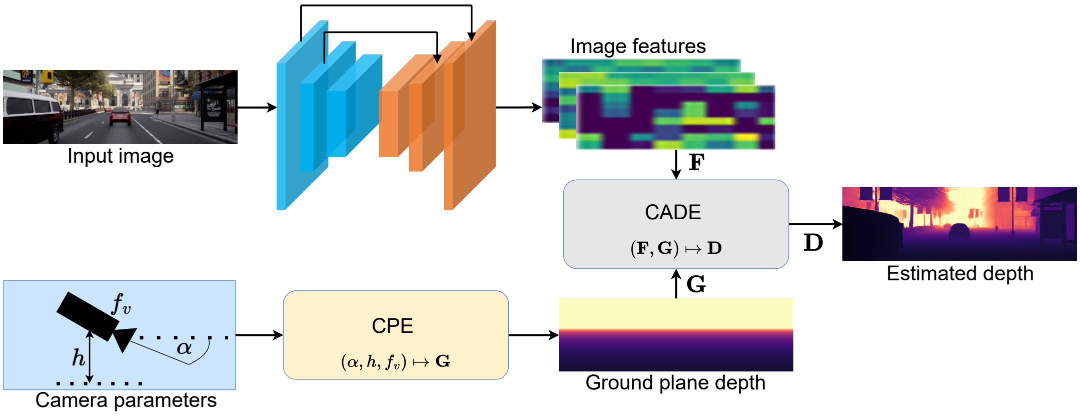


Fig. 1: Illustration of the proposed architecture. Our system embeds camera parameters as depth of the ground plane and learns depth that generalizes for various camera-car setups. CPE refers to the Camera Parameters Embedding described in Section II, while CADE refers to the Camera Adaptive Depth Estimation described in Section III.

for autonomous driving scenarios

- a network architecture with embedded parametrization as visualized in Fig. 1, specifically designed for generalization and further work in sim2real domain adaptation
- a large-scale annotated autonomous driving dataset within the CARLA simulator [30], created due to the unavailability of data with sufficiently diverse camera parameters[2]
- thorough experimentation on the effects of parameter variation and efficacy of the proposed approach.

II. Proposed Camera Parameters Embedding

Autonomous driving datasets such as the KITTI [31], Oxford RobotCar [32] or Cityscapes [33] frequently feature a single vehicle-camera setup. This means that correct depth values for certain pixels are almost identical across different images, e.g., on the ground plane. Even though convolution is an inherently positionally equivariant operation, convolutional neural networks tend to implicitly learn absolute position information from commonly used padding operations [34]. Additionaly, MDE networks have been shown to use the ground-plane contact point for object depth estimation [1].

In order for MDE to be practically used in automotive scenarios, depth estimation should be accurate for different vehicle-camera setups. However, if camera parameters during inference differ from the parameters used in training, depth estimation accuracy degrades significantly. For example, networks learn from the training data that the pixel at a particular position in the image tends to have certain depth value, which remains largely the same throughout the dataset. However, if the camera parameters are changed, this assumption breaks. Fig. 2 demonstrates the effects that camera parameters variation has on depth estimation accuracy. Changes in camera pitch, camera height, and vertical field of view (which also changes vertical focal length) during inference, compared to the training setup, significantly affect depth estimation accuracy, especially on the ground plane. On other hand, horizontal field of view and focal length changes do not significantly influence the estimation, as long as the context does not change dramatically.

In this work, we target the most plausible variations in vehicle-camera setups which can disturb depth estimation: camera height, camera pitch and vertical focal length. In order to learn metrically accurate depth with varying focal lengths, knowledge of the focal length should be embedded in the network due to inherent ambiguity between the focal length and depth [23], [24]. Additionally, while the network could learn the effects of camera height and pitch on the estimated depth, if trained with sufficiently diverse data, embedding of extrinsic parameters was shown to be beneficial [25]. Given that, we choose to embed all the three parameters in the network. To do so, for every pixel coordinate (u, v) we calculate the depth $\mathbf{G}(u, v)$ at which the optical ray intersects the ground plane via the following constraints

$$\begin{aligned} \mathbf{n}^{\mathrm{T}}\mathbf{R}^{\mathrm{T}}(\alpha)\mathbf{p} + h = \mathbf{0}, \\ \mathbf{p} = \mathbf{G}(u,v) \left[\begin{array}{ccc} \frac{u-c_u}{f_u} & \frac{v-c_v}{f_v} & 1 \end{array} \right]^T, \end{aligned} \tag{1}$$

where h is the camera height, $\mathbf{R}(\alpha)$ is the rotation matrix for camera pitch α, $\mathbf{n}$ is the ground plane normal, and $(f_u, f_v), (c_u, c_v)$ represent the camera focal length and principal point, respectively. With the assumption of ideal ground normal $\mathbf{n} = [0, -1, 0]^T$, depth $\mathbf{G}(u, v)$ can be calculated as a function of (α, h, f_v, c_v). In this work, we set the principal point at the center of the image plane, which is a fair assumption for most cameras. Our embedding function is thus a mapping

$$(\alpha, h, f_v) \mapsto \mathbf{G} \in [0, M]^{H \times W}, \tag{2}$$

where M is maximum depth specific to the dataset, and (H, W) are dimensions of the image. In Fig. 3 we show the visualization of our camera parameter embedding $\mathbf{G}$ for

[2] Dataset is publicly available at https://zenodo.org/record/7899804#.ZFT0oJFBzJV

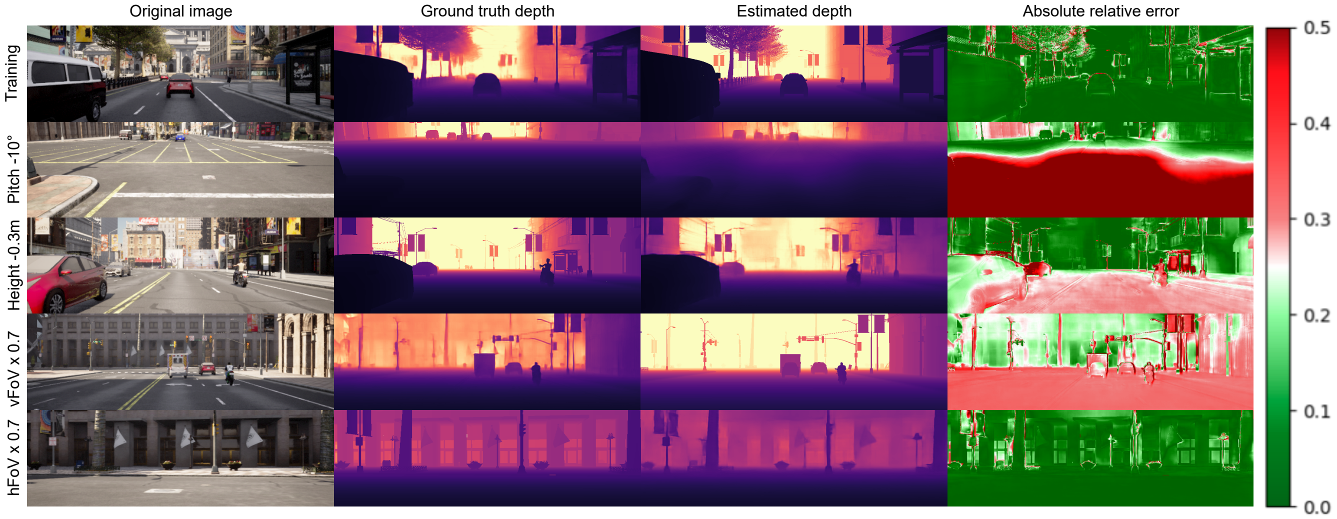
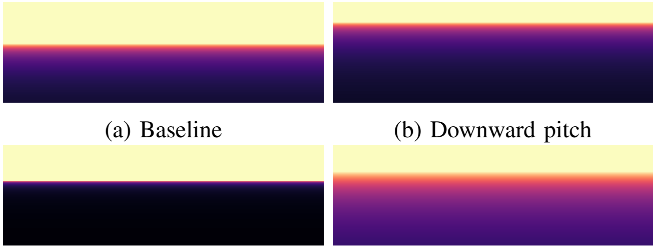


Fig. 2: Depth estimation results given common variations of the camera parameters compared to the training setup.


(a) Baseline

(b) Downward pitch

(c) Smaller height

(d) Larger focal length

Fig. 3: Visualization of the embedded camera parameters as ground plane depth $\mathbf{G}$.

different camera parameters. Variations of camera parameters (namely camera pitch, camera height and focal length) compared to the baseline are reflected in the embeddings, which provide the network with useful a-priori available information about the camera setup.

Our motivation for such a choice of camera parameters embedding is threefold:

- depth of the ground plane is a common and unique parametrization for camera pitch, camera height, and focal length, i.e., the mapping in (2) is injective
- embedding of $\mathbf{G}$ gives the neural network useful positional information, i.e., the network is explicitly informed about the expected depth for current camera parameters at a certain pixel position, if the ground plane is not occluded
- neural network can be forced to estimate depth as a function of $\mathbf{G}$, which leads to learning more robust features and better generalization accuracy for unseen camera parameters.

III. Proposed Network Architecture

Depth estimation networks often follow a standard encoder-decoder architecture with residual connections between encoder and decoder layers. Encoder learns spatially coarse features of higher dimensions, which are then continually upsampled towards original image resolution in the decoder. Our method is designed to work with arbitrary encoder-decoder architecture. While recent works use transformers for feature extraction and fusion [15]–[17], we choose to use a ResNet18 [35] encoder and a decoder combining convolutional and upsampling layers, thus recovering the feature map $\mathbf{F} \in \mathbb{R}^{C\times H\times W}$. Instead of directly regressing the depth map $\mathbf{D}$ from $\mathbf{F}$, we forward it along with the map of embedded camera parameters $\mathbf{G}$ into the Camera Adaptive Depth Estimation (CADE) module.

A. CADE module

CADE transforms image features and camera parameters embedded as ground plane depth into the depth map, i.e., it performs the mapping $(\mathbf{F}, \mathbf{G}) \mapsto \mathbf{D} \in [0, M]^{H\times W}$. We fuse $\mathbf{G}$ and $\mathbf{F}$ inside a novel transformer architecture visualized in Fig. 4, as we want to exploit the global receptive field of the attention mechanism.

Firstly, we rearrange $\mathbf{F}$ and $\mathbf{G}$ into

$$\mathbf{Z}'_{\mathbf{F}} = \begin{bmatrix} z'_{f_1} \\ \vdots \\ z'_{f_N} \end{bmatrix} \in \mathbb{R}^{N\times D'_f}, \mathbf{Z}'_{\mathbf{G}} = \begin{bmatrix} z'_{g_1} \\ \vdots \\ z'_{g_N} \end{bmatrix} \in [0, M]^{N\times D'_g}, \tag{3}$$

where $N = \frac{HW}{p^2}, D'_f = Cp^2, D'_g = p^2$, with p being patch size. After layer normalization, these are then processed through a linear layer with addition of learnable positional embedding, resulting in sets of image feature tokens $\mathbf{Z}_{\mathbf{F}} \in \mathbb{R}^{N\times D_f}$ and ground plane depth tokens $\mathbf{Z}_{\mathbf{G}} \in \mathbb{R}^{N\times D_g}$:

$$\mathbf{Z}_{\mathbf{F}} = \mathbf{Z'}_{\mathbf{F}}\mathbf{W}_{\mathbf{F}} + \mathbf{p}_{\mathbf{F}}, \mathbf{W}_{\mathbf{F}} \in \mathbb{R}^{D'_f\times D_f}, \tag{4}$$

$$\mathbf{Z}_{\mathbf{G}} = \mathbf{Z'}_{\mathbf{G}}\mathbf{W}_{\mathbf{G}} + \mathbf{p}_{\mathbf{G}}, \mathbf{W}_{\mathbf{G}} \in \mathbb{R}^{D'_g\times D_g}. \tag{5}$$

Afterwards, we proceed with calculation of query, key and

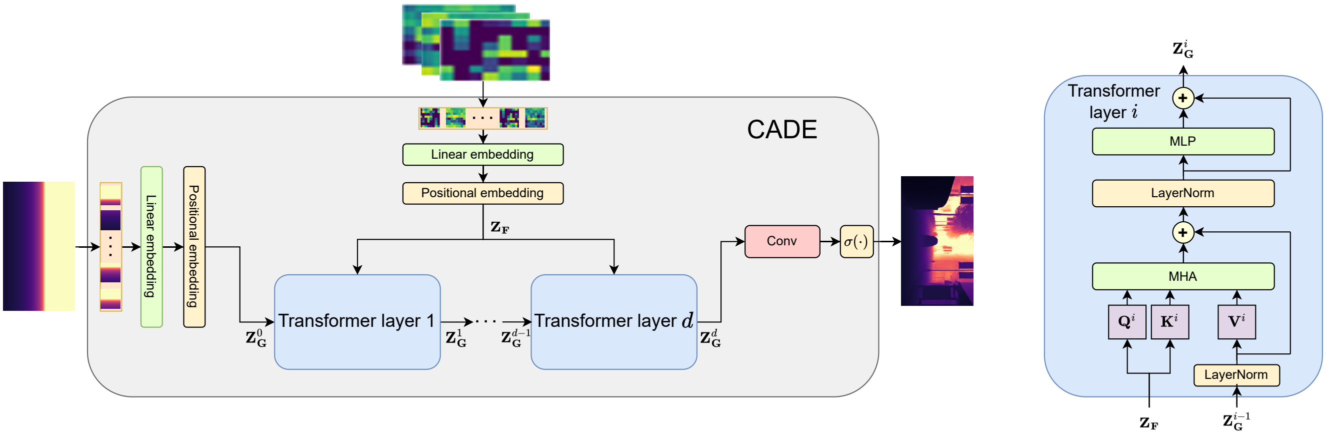


Fig. 4: Structure of the CADE module. Ground plane depths $\mathbf{Z_G}$ are processed through d successive transformer layers, with image features $\mathbf{Z_F}$ used in the calculation of attention weights.

value matrices needed for attention calculation:

$$\mathbf{Q} = \mathbf{Z_F}\mathbf{W_Q} \tag{6}$$

$$\mathbf{K} = \mathbf{Z_F}\mathbf{W_K} \tag{7}$$

$$\mathbf{V} = \mathbf{Z_G}\mathbf{W_V} \tag{8}$$

where $\mathbf{W_Q}, \mathbf{W_K} \in \mathbb{R}^{D_f \times D_h}$ and $\mathbf{W_V} \in \mathbb{R}^{D_g \times D_h}$ are projection matrices. Attended output is determined as

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\text{T}}{\sqrt{N}}\right)\mathbf{V}, \tag{9}$$

which is calculated for multiple heads and then fused via the linear layer. Notice how we calculate queries and keys from features tokens and values from ground plane depth tokens. This means that our attended output for particular token is a weighted function of ground plane depth tokens, with weights calculated as a self-attention of feature tokens.

We propagate tokens $\mathbf{Z_G}$ through d successive transformer layers consisting of multihead attention (MHA) and multilayer perceptron layers (MLP), along with residual connections where $\mathbf{Z}^1_\mathbf{G}$ is initialized via (5):

$$\mathbf{Z}^i_\mathbf{G} = \text{LayerNorm}(\mathbf{Z}^i_\mathbf{G}), \tag{10}$$

$$\mathbf{Q}^i = \mathbf{Z_F}\mathbf{W}^i_\mathbf{Q}, \mathbf{K}^i = \mathbf{Z_F}\mathbf{W}^i_\mathbf{K}, \mathbf{V}^i = \mathbf{Z}^i_\mathbf{G}\mathbf{W}^i_\mathbf{V}, \tag{11}$$

$$\mathbf{Z}^i_\mathbf{G} = \text{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) + \mathbf{Z}^i_\mathbf{G}, \tag{12}$$

$$\mathbf{Z}^i_\mathbf{G} = \text{LayerNorm}(\mathbf{Z}^i_\mathbf{G}), \tag{13}$$

$$\mathbf{Z}^{i+1}_\mathbf{G} = \text{MLP}(\mathbf{Z}^i_\mathbf{G}) + \mathbf{Z}^i_\mathbf{G}. \tag{14}$$

Notice how through all CADE layers we use feature tokens $\mathbf{Z_F}$ only in calculation of attention weights. Depth estimation is thus forced to be a function of the embedded camera parameters $\mathbf{G}$, with $\mathbf{F}$ serving as a clue on how to properly combine embedding $\mathbf{G}$ into $\mathbf{D}$. CADE module is purposefully appended at the end of the network, which makes $\mathbf{F}$ independent of $\mathbf{G}$. In such a manner, network is incentivized to learn $\mathbf{F}$ that are invariant to different camera parameters.

Dataset	Size	Description
$\mathcal{B}$	20000	$\alpha = -5, h = 1.5, f_v = 570$
$\mathcal{U}_\alpha$	10000	$\alpha \sim \mathcal{U}(-15, 5), h = 1.5, f_v = 570$
$\mathcal{U}_h$	10000	$h \sim \mathcal{U}(1, 2), \alpha = -5, f_v = 570$
$\mathcal{U}_{f_v}$	10000	$f_v \sim \mathcal{U}(260, 880), \alpha = -5, h = 1.5$
$\mathcal{U}_{\alpha,h,f_v}$	40000	$(\alpha, h, f_v) \sim \mathcal{U}(-15, 5) \times \mathcal{U}(1, 2) \times \mathcal{U}(260, 880)$
$\mathcal{D}_{\alpha,h,f_v}$	40000	$\alpha \in \{-15, 5, 5\}, h \in \{1, 1.5, 2\},$ $f_v \in \{260, 570, 880\}$

TABLE I: Camera parameter specifications used in collected datasets. $\mathcal{B}$ – baseline dataset, parameters are constant throughout the dataset, $\mathcal{U}_\alpha, \mathcal{U}_h, \mathcal{U}_{f_v}$ – single varying parameter sampled from continuous uniform distribution, $\mathcal{U}_{\alpha,h,f_v}$ – all varying parameters sampled from continuous uniform distribution, $\mathcal{D}_{\alpha,h,f_v}$ – all varying parameters sampled from discrete uniform distribution of 3 possible values. Values for α, h, f_v are expressed in degrees, meters and pixels respectively.

Finally, we use $\text{Rearrange}(\cdot) : \mathbb{R}^{N \times D_g} \rightarrow \mathbb{R}^{C' \times H \times W}$ and a final convolutional layer with a sigmoid activation to regress depth map $\mathbf{D} \in [0, M]^{H \times W}$:

$$\mathbf{D} = \sigma(\text{Conv}(\text{Rearrange}(\mathbf{Z}^d_\mathbf{G}))) * M. \tag{15}$$

For the training loss, we follow [15] and use Scale-Invariant loss (SI). With the logarithmic distance $g_i = log(\hat{d}_i) - log(d_i)$ between ground truth depth $\hat{d}_i$ and estimated depth d_i at pixel location i, SI loss is:

$$\mathcal{L} = \alpha\sqrt{\frac{1}{|\mathbf{D}|}\sum_i g_i^2 - \frac{\lambda}{|\mathbf{D}|^2}\left(\sum_i g_i\right)^2}, \tag{16}$$

where we use $\lambda = 0.85$ and $\alpha = 10$ as in [15].

IV. Experimental Results

A. Datasets

Due to unavailability of autonomous driving data with sufficiently diverse camera parameters, we created our own dataset using the CARLA simulator [30]. We simulate autonomous driving scenarios within urban, rural, and highway environments across 8 different maps *Town01 - Town07* and

	Method	Training	Testing	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
a)	Baseline	$\mathcal{B}$	$\mathcal{B}$	0.046	0.482	**4.541**	**0.108**	0.959	**0.987**	**0.995**
	CPE + CADE	$\mathcal{B}$	$\mathcal{B}$	**0.044**	**0.475**	4.552	0.110	**0.961**	0.986	0.995
b)	Baseline	$\mathcal{B}$	$\mathcal{U}_{\alpha,h,f_v}$	0.261	2.076	7.478	0.297	0.547	0.846	0.960
	Baseline	$\mathcal{U}_{\alpha,h,f_v}$	$\mathcal{U}_{\alpha,h,f_v}$	0.064	0.437	3.549	0.102	0.960	0.991	0.996
	CPE + CADE	$\mathcal{U}_{\alpha,h,f_v}$	$\mathcal{U}_{\alpha,h,f_v}$	**0.039**	**0.387**	**3.382**	**0.085**	**0.970**	**0.991**	**0.997**
c)	Baseline	$\mathcal{B}$	$\mathcal{U}_\alpha$	0.244	1.248	5.748	0.198	0.699	0.931	0.989
	Baseline	$\mathcal{U}_\alpha$	$\mathcal{U}_\alpha$	0.041	0.310	3.445	0.078	0.973	0.991	**0.997**
	CPE + CADE	$\mathcal{U}_\alpha$	$\mathcal{U}_\alpha$	**0.035**	**0.301**	**3.410**	**0.076**	**0.975**	**0.992**	0.997
d)	Baseline	$\mathcal{B}$	$\mathcal{U}_h$	0.214	1.245	6.035	0.232	0.666	0.924	0.988
	Baseline	$\mathcal{U}_h$	$\mathcal{U}_h$	0.038	0.319	3.600	0.085	0.971	0.991	0.997
	CPE + CADE	$\mathcal{U}_h$	$\mathcal{U}_h$	**0.035**	**0.312**	**3.581**	**0.083**	**0.972**	**0.992**	**0.997**
e)	Baseline	$\mathcal{B}$	$\mathcal{U}_{f_v}$	0.254	1.746	7.286	0.268	0.563	0.868	0.987
	Baseline	$\mathcal{U}_{f_v}$	$\mathcal{U}_{f_v}$	0.040	**0.351**	3.722	0.088	0.972	0.990	**0.997**
	CPE + CADE	$\mathcal{U}_{f_v}$	$\mathcal{U}_{f_v}$	**0.035**	0.353	**3.647**	**0.084**	**0.972**	**0.991**	0.997
f)	Baseline	$\mathcal{D}_{\alpha,h,f_v}$	$\mathcal{U}_{\alpha,h,f_v}$	0.102	0.633	4.655	0.148	0.889	0.987	0.996
	CPE + CADE	$\mathcal{D}_{\alpha,h,f_v}$	$\mathcal{U}_{\alpha,h,f_v}$	**0.067**	**0.458**	**3.920**	**0.110**	**0.945**	**0.991**	**0.997**

TABLE II: Results of various model and dataset configurations. Baseline refers to the standard encoder-decoder architecture, with ResNet 18 encoder and decoder from [20], where depth **D** is directly regressed from image features **F**, while CPE + CADE refers to addition of our contributions. Results are expressed in standard MDE metrics [20], red – lower is better, blue – higher is better.

Town10HD that include highly detailed and realistic textures. The maps are populated with a diverse set of traffic actors, which are then autonomously controlled while respecting the traffic rules.

In order to capture the training and testing data, we mount RGB and depth cameras in a way that no part of the car is within the field of view of the camera. Advanced RGB camera parameters such as distortions and postprocessing effects are adjusted to mimic the KITTI dataset [31] as close as possible. Camera sensors are repeatedly destroyed and reinitialized with new extrinsic and intrinsic parameters, thus avoiding the memory difficulties which are present with multiple camera sensors working at the same time. We collect several datasets with different distributions of camera parameters, as described in Table I.

B. Implementation details

As our method is adaptable for various encoder-decoder architectures, we use a simple convolutional residual network. Our encoder is a ResNet 18 network which encodes image features at a $\frac{H}{32} \times \frac{W}{32}$ resolution. Decoder then successively upsamples the features in 5 stages, each consisting of a 3x3 kernel convolution which fuses encoder features via skip connection and an upsampling layer followed by another convolutional layer. Finally, decoder outputs image features $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$, where we use $C = 16, H = 320, W = 1024$.

For our CADE module, we choose to use a light architecture in order to prevent a significant increase in computation time and memory consumption. We use a standard patch size $p = 16$, with inner embedding dimensions $D_g = 1024$ and $D_f = 4096$. We calculate the MHA with 8 heads and a head dimension $D_h = 64$, which is then followed by a MLP with one hidden layer which increases the embedded dimension by two times. In order to keep our CADE module lightweight, we choose $d = 2$ for a number of transformer layers. Finally, following the standard practice in depth estimation [20], we estimate depth up to a maximum value $M = 80$m.

We train our network with an Adam optimizer [36] with a batch size 12. We decrease the learning rate linearly from 4×10^{-5} to 4×10^{-6}. All networks are trained and tested on a single Nvidia RTX A5000 GPU.

C. Results

We conduct a thorough experimentation on the effect of camera parameter variation and efficacy of our approach. In Table II we present results for various combinations of methods and dataset configurations. In order to test the generalization of each approach, for each dataset we create a 90%/10% training and testing split.

First of all, in Table II a) we perform the ablative experiments on the baseline dataset, where we both train and test the networks on the data collected with a single vehicle-camera setup. As expected, despite the increase of the model complexity due to the addition of our contributions, usage of CPE and CADE does not improve performance compared to the standard encoder-decoder architecture, since ground plane depth **G** fused in CADE does not supplement the network with useful information. This is a desired behavior, considering that the camera parameters are constant throughout the dataset. Ground plane depth is mostly the same across all images, thus enabling the baseline model to easily learn the information which is otherwise supplemented with **G** in our approach.

Afterwards in Table II b) we test the performance on the dataset with varying camera parameters $\mathcal{U}_{\alpha,h,f_v}$. Naturally, baseline method trained on a dataset with a constant vehicle-camera setup performs poorly since it is biased to a particular perspective geometry induced in the training data. On the contrary, baseline method trained on $\mathcal{U}_{\alpha,h,f_v}$ performs surprisingly well, with good generalization performance on unseen images. This shows that, when presented with sufficiently diverse perspective geometry, network can exploit

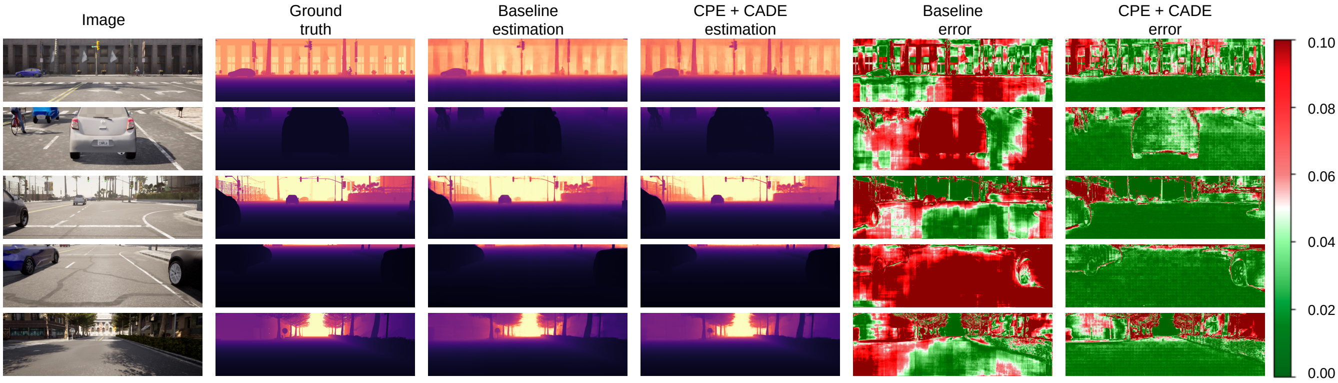


Fig. 5: Results of MDE trained and tested on the $\mathcal{U}_{\alpha,h,f_v}$ dataset with visualization of absolute relative error. Fusion of embedded camera parameters within CADE significantly reduces the absolute relative error compared to the baseline, especially on the ground plane.

Method	Abs Rel	Sq Rel	RMSE	RMSE log
Early fusion	0.051	0.401	3.401	0.090
Mid fusion	0.050	0.397	3.402	0.089
Late fusion	0.061	0.412	3.622	0.102
CADE	**0.039**	**0.387**	**3.382**	**0.085**

TABLE III: Results of the ablation experiments trained and tested on $\mathcal{U}_{\alpha,h,f_v}$, with fusion of $\mathbf{G}$ into convolutional channels at a certain point. Early fusion – fusion in the first encoder layer, Mid fusion – fusion in skip connections between encoder and decoder, Late fusion – fusion in last decoder layer.

semantic cues to infer depth for varying camera parameters. However, the accuracy of the baseline network is significantly lower compared to the proposed approach. Fusion of embedded camera parameters within the CADE module notably improves the results for all MDE metrics, proving the usefulness of information encoded in ground plane depth $\mathbf{G}$, and efficacy of its fusion within the CADE module. Figure 5 shows significant reduction in absolute relative error compared to the baseline, especially for inconsistent estimations on the ground plane.

In order to examine the generalization capability for each camera parameter separately, in Table II c) d) e) we repeat the same experiment while selectively varying only one camera parameter throughout the dataset. Again, while the model trained on a single vehicle-camera setup performs poorly, baseline network can learn to generalize when presented with diverse data in the training set. However, in contrast to results in Table II b), fusion of camera parameters in CADE module does not significantly improve the results. Since only one parameter is varied, network can learn to focus on semantic cues specific to that camera parameters, thus effectively reducing the need for embedding of $\mathbf{G}$.

In Table II f) we examine the ability of our approach to generalize for camera parameters not present in the training distribution. To do so, for training we use a sparse discrete distribution with 3 possible samples for each parameter, positioned at the tail ends and the mean of the continuous uniform distribution $\mathcal{U}_{\alpha,h,f_v}$. In such manner, network should learn to meaningfully predict the depth for camera parameters between those discrete samples. We show that our approach learns to generalize more effectively than the baseline, which means that the network successfully learns geometric relationship between embedded camera parameters $\mathbf{G}$ and scene depth. To that end, our approach is feasible to be utilized with real-world data, since the collection of data with distribution similar to $\mathcal{D}_{\alpha,h,f_v}$ is feasible. However, increase in accuracy is not as prominent as in Table II b), which means that the semantic cues, such as the horizon level, when varied throughout the training dataset provide useful information for training of camera invariant depth estimation, even when the network is supplemented with embedded camera parameters $\mathbf{G}$.

Finally, we assess the performance of various fusion methods for embedded camera parameters $\mathbf{G}$ in Table III. Early and mid fusion are similar to [25] and [24] respectively, but with different choice of embedded camera parameters and embedding function. The most meaningful result is the difference between performance of late fusion within convolutional layers and our CADE module. Even though the fusion happens at the same point in the network, CADE achieves better results by taking advantage of the global receptive field of self-attention, and by strict enforcement of estimating depth $\mathbf{D}$ as a function of $\mathbf{G}$.

V. Conclusion and further work

In this paper we have presented an approach for camera invariant monocular depth estimation for automotive scenarios. After detailed examination on the effects of varying camera parameters on depth estimation performance, we designed a novel camera parameters embedding procedure in order to supplement the network with useful information about the perspective geometry and to force the network to learn depth estimation as a function of embedded parameters, thus effectively enabling learning of camera invariant features. The proposed embedding is fused with image features within a novel module that exploits the global receptive field of the self-attention. We assess the accuracy of the proposed approach on various datasets with different camera extrinsic and intrinsic parameters distributions, collected within a simulated automotive environment. We show that the embedding provides useful information about the perspective geometry and enables better generalization on unseen data. Our method is specifically designed for further work in domain adaptation, where we aim to achieve camera invariant depth estimation on real-world data.

References

[1] T. v. Dijk and G. d. Croon, "How do neural networks see depth in single images?" in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2183–2191.

[2] 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.

[3] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 239–248.

[4] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[5] A. CS Kumar, S. M. Bhandarkar, and M. Prasad, "Depthnet: A recurrent neural network architecture for monocular depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 283–291.

[6] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza, "Toward domain independence for learning-based monocular depth estimation," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1778–1785, 2017.

[7] D. Xu, W. Wang, H. Tang, H. Liu, N. Sebe, and E. Ricci, "Structured attention guided convolutional neural fields for monocular depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3917–3925.

[8] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1119–1127.

[9] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5162–5170.

[10] Y. Cao, Z. Wu, and C. Shen, "Estimating depth from monocular images as classification using deep fully convolutional residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3174–3182, 2017.

[11] H. Jung, Y. Kim, D. Min, C. Oh, and K. Sohn, "Depth prediction from a single image with conditional adversarial networks," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 1717–1721.

[12] K. G. Lore, K. Reddy, M. Giering, and E. A. Bernal, "Generative adversarial networks for depth map estimation from rgb video," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2018, pp. 1258–12 588.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[15] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018.

[16] A. Agarwal and C. Arora, "Attention attention everywhere: Monocular depth prediction with skip attention," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5861–5870.

[17] Z. Li, Z. Chen, X. Liu, and J. Jiang, "Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation," *arXiv preprint arXiv:2203.14211*, 2022.

[18] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 340–349.

[19] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 7286–7291.

[20] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3828–3838.

[21] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7286–7291.

[22] K. Koledić, I. Cvišić, I. Marković, and I. Petrović, "Moft: Monocular odometry based on deep depth and careful feature selection and tracking," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 6175–6181.

[23] L. He, G. Wang, and Z. Hu, "Learning depth from single images with deep neural network embedding focal length," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4676–4689, 2018.

[24] J. M. Facil, B. Ummenhofer, H. Zhou, L. Montesano, T. Brox, and J. Civera, "Cam-convs: Camera-aware multi-scale convolutions for single-view depth," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 826–11 835.

[25] Y. Zhao, S. Kong, and C. Fowlkes, "Camera pose matters: Improving depth prediction by mitigating pose distribution bias," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 759–15 768.

[26] S. Saha, A. Obukhov, D. P. Paudel, M. Kanakis, Y. Chen, S. Georgoulis, and L. Van Gool, "Learning to relate depth and semantics for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8197–8207.

[27] A. Atapour-Abarghouei and T. P. Breckon, "Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2800–2810.

[28] S. Zhao, H. Fu, M. Gong, and D. Tao, "Geometry-aware symmetric domain adaptation for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9788–9798.

[29] K. PNVR, H. Zhou, and D. Jacobs, "Sharingan: Combining synthetic and real data for unsupervised geometry estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 974–13 983.

[30] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

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

[32] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[33] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[34] M. A. Islam, S. Jia, and N. D. Bruce, "How much position information do convolutional neural networks encode?" *arXiv preprint arXiv:2001.08248*, 2020.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.