Self-Supervised Surround-View Depth Estimation with Volumetric Feature Fusion

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this work, we propose a self-supervised depth estimation method using a unified volumetric feature encoded from surround-view. The proposed network architecture consists of three parts. First, given a set of surround-view images, the surround-view feature fusion module extracts image features from each view, aggregates the features into 3D space, and encodes them into a unified volumetric feature. We use a multilayer perceptron to refine the volumetric features, especially for features shared between multiple views. Second, we propose a depth fusion module that takes a specific viewpoint of interest from the volumetric feature and reconstructs a depth map at the corresponding view; therefore, our method is able to render scale-aware depth maps not only at known input camera views but also at any arbitrary rotated views. Lastly, assuming static camera extrinsics in the multi-camera system, we propose to estimate a single global motion according to a canonical camera coordinate system. The proposed method leverages 3D spatio-temporal context to learn metric-scale depth in a self-supervised manner. We adjust the intensity distribution around common image boundaries to avoid irregular photometric reconstruction errors. Through the extensive experiments on DDAD and nuScenes datasets, we show that our method outperforms the prior arts.

1 Introduction

Depth perception of the surrounding environment is one of the key components for 3D vision applications such as autonomous driving, robotics, and augmented reality. Specifically, for autonomous driving, 3D perception is beneficial for downstream tasks including 3D objects detection [6, 25, 40], 6D pose estimation [5, 19], object tracking [16, 35], and so on.

While human drivers can proactively move their head and eyes to observe their surrounding environment, an autonomous vehicle equips multiple cameras with fixed viewpoints and monitor its surroundings. The multi-camera system hence exhibits limitations subjective to the initial camera setup. In other words, the camera system might share a small portion between the adjacent viewpoints and rely on heterogeneous camera intrinsics. As each camera monitors different viewpoints, by nature, it also captures the scenes with diverse contents under uneven light conditions at all times.

Full Surround Monodepth (FSM) [13] extends a self-supervised monocular depth estimation method into a multi-camera setup of the autonomous vehicle. Despite minimal overlaps between the cameras, this method successfully exploits spatio-temporal context to learn metric-scale depth in practice. Yet, as FSM [13] individually estimates depth and camera motion for each view with a shared CNN, it does not utilize any relationship between neighboring views at test time. To date, architectural design choices in the spatio-temporal domain are relatively less investigated, creating room for improvements.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

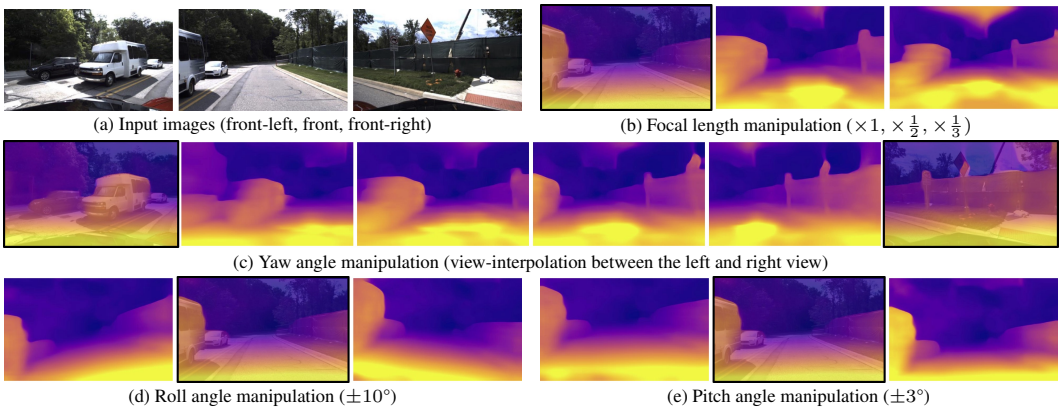


Figure 1: **Visualization of synthesized depths**: *(a)* Given a set of images (e. g. front-left, front, and front-right), we can synthesize depth map at imaginary views with different *(b)* focal lengths (zoom-out effect), *(c)* yaw angles, *(d)* roll angles, and *(e)* pitch angles. Depth maps at known camera views are overlayed with input images.

We introduce a self-supervised approach to estimate surround-view depth based on a unified volumetric feature representation. More precisely, given multiple images covering the surrounding view, we extract an image feature from each image, back-project the features into 3D space (i. e., voxel coordinate), and fuse them into a unified volumetric feature map. Each volumetric feature contains 3D information at its located voxel coordinate. Some volumetric features are shared between neighboring views due to their image overlaps; we present a tailored multilayer perceptron (MLP) to process the volumetric features with consideration of the superposition. Given the volumetric features, we project the features back to the image coordinate with known camera information and pass through the depth decoder to obtain the depth map.

Because volumetric feature contains 3D information at its local voxel coordinate, our method can also synthesize a depth map at arbitrary rotated viewpoints by projecting the features into an image coordinate with a target focal length and target extrinsics. Fig. 1 demonstrates the results of depth map synthesis at novel views. We can see how seamlessly our method synthesizes depth maps at imaginary viewpoints from the volumetric features with different focal lengths and camera angles.

Furthermore, unlike previous work [13], we leverage a canonical camera motion assuming static extrinsics between cameras. This geometry constraint leads to a stabilized ego-motion learning and enhanced metric-scale depth estimation for an arbitrary viewpoint regardless of rotation. On top of that, we adjust the intensity distribution for the neighboring viewpoints, which reduces irregular photometric reconstruction errors. We present a self-supervised learning framework that estimates scale-aware depth and predicts a camera motion, using 3D spatio-temporal reconstruction.

We summarize our contributions as: (i) we introduce a novel volumetric feature representation that effectively learns to estimate the surround-view depth and canonical camera motion. To the best of our knowledge, our method is the first approach that targets multi-camera feature fusion with self-supervised depth estimation. (ii) We demonstrate synthesizing depth maps at arbitrary views with the unified volumetric feature representation. (iii) On popular multi-camera datasets [2, 12], our method demonstrates consistent improvements compared to the state-of-the-art method [13].

2 Related Work

Depth estimation from a monocular image. Supervised-learning-based approaches [7, 8, 18, 20, 30] require precise annotations to estimate a depth map from a single image; however, due to the expense, acquisition of dense depth annotation for diverse scenes is limited in scalability. Instead, without annotated datasets, self-supervised methods rely on proxy learning tasks to render depth maps from sequential images [3, 12, 21, 22, 31, 41, 45, 47, 48] or stereoscopic pairs [9–11, 23, 28, 29]. Basically, these approaches attempt to minimize reprojection errors between the reference viewpoints and synthesized ones that are generated by the network predictions. Yet, a scale ambiguity remains since the network presents relative depth information. In this work, we estimate a depth map at an arbitrary viewpoint while taking the metric scale into account.

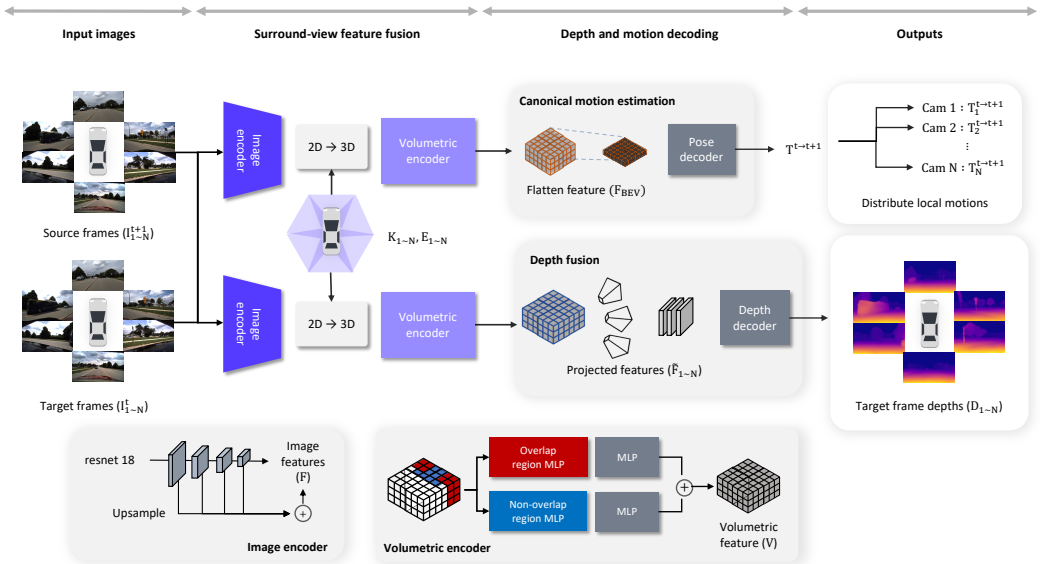


Figure 2: **Architecture overview**: The proposed network architecture consists of the depth fusion module and canonical motion estimation module. We extract image features, fuse them into a volumetric feature by considering overlapped features between neighboring views, and estimate depth for each view and canonical camera motion. For the canonical motion estimation module, the volumetric feature is collapsed in the z-axis and goes through the pose decoder. The output canonical motion can be distributed to each view as local motions.

Building cost volume. The cost volume provides enriched descriptions of the 3D geometry. Various cost volume approaches have been proposed to aggregate depth information from multiple viewpoints [15, 17, 46] or consecutive frames [42]. However, the cost volume often fails in conveying the depth information around moving objects. To alleviate this issue, several methods attempt to ignore cost volume from moving objects based on transfer learning [41], spatio-temporal attention [32], or recurrent neural network [27]. While the cost volume mainly embeds depth occupancies, it cannot carry out 3D structural information for undiscovered space. On the contrary, the proposed representation, a unified volumetric feature, can seamlessly draw a depth map for a novel view.

Omnidirectional depth estimation. To estimate depth information from a wide field of view (FOV), researchers seek an optimal projection geometry for handling visual distortions. Equidistant cylindrical projection [37–39, 44, 49] is popular; but, it yields severe distortions on the top and bottom parts of the image and is weak for occlusions. On the other hand, Cheng et al. [4] employed cubemap tiling in consideration of overlaps. Won et al. [43] directly fed spherical images into the network and refined the cost volume. Neural Ray Surfaces (NRS) [36] proposes a generic framework that jointly learns monocular depth and motion without assuming specific camera lens models. Since wide FOV images present a relatively low resolution at distance, these methods mainly target indoor usages, which do not require reliable long-range depth estimation.

3 Full Surround Depth via Volumetric Feature

Given a pair of temporally consecutive surround view images, $\mathbf{I}_i^t$ and $\mathbf{I}_i^{t+1}$, captured from multiple cameras C_i ($i \in \{1, 2, ..., 6\}$ in our paper) that cover a surrounding view of a vehicle, our method aims to estimate metric-scale depth $\mathbf{D}_i^t$ for each image $\mathbf{I}_i^t$ from each camera C_i (but also at an arbitrary rotated viewpoint) and a canonical camera motion of the vehicle $\mathbf{T}^{t\to t+1}$ as an auxiliary task. We assume known camera intrinsics $\mathbf{K}_i$ and extrinsics $\mathbf{E}_i$. However, each camera can have different intrinsics.

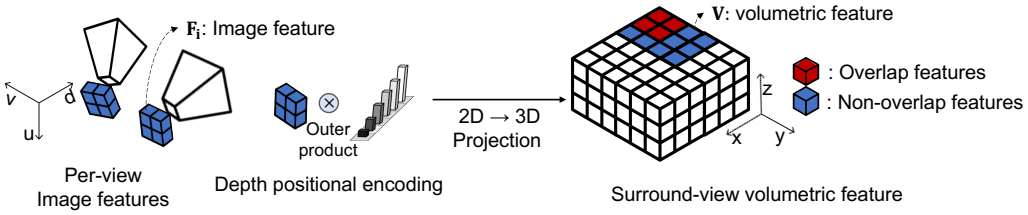


Figure 3: **Surround-view volumetric feature fusion**: First, we aggregate multi-scale image features from each viewpoint ($\mathbf{F_i}$). Then, the features from each view are encoded with depth information and projected to 3D using camera calibration parameters to form a surround-view volumetric feature($\mathbf{V}$).

3.1 Architecture overview

We propose a novel fusion module for surround-view depth estimation. Fig. 2 shows an overview of our approach, consisting of the surround-view volumetric feature fusion (Sec. 3.2), depth fusion (Sec. 3.3) and canonical motion estimation (Sec. 3.4) module.

The surround-view feature fusion module constructs multi-scale image feature pyramids of the surround-view and maps them onto a volumetric feature. More precisely, we transform the image features $\mathbf{F}$ to a volumetric feature $\mathbf{V}$ using known camera parameters. With the depth fusion module, we retrieve a per-frame feature representing a specific viewpoint from the volumetric feature $\mathbf{V}$ and pass it through a depth decoder to produce a depth map for the corresponding view. Since we process per-frame features in a unified volumetric feature space, the network presents spatially coherent depth maps regardless of viewpoints. Finally, the canonical motion estimation module uses the volumetric feature collapse in the z-axis and predicts a global motion referring to the canonical camera coordinate. This is one of the main differences compared to FSM [13] that regularizes individually estimated camera poses. On the other hand, our design reduces the number of unknown factors and makes the problem simple by globally reasoning a camera motion based on the multi-camera feature.

3.2 Surround-view feature fusion

To aggregate multi-camera features, the surround-view feature fusion pipeline employs a novel unified volumetric feature representation. Images from multi-camera are fed into a shared image encoder to produce a multi-scale feature pyramid for each image. Afterward, we transform the image features to voxel coordinate using the camera parameters. In addition, we coin multilayer perceptron to deal with superposed features from different views but back-projected to the same voxel.

Image feature encoding. Given a set of surround-view images, we pass each image $\mathbf{I}_i$ through a shared 2D image encoder to obtain a image feature map $\mathbf{F}_i$ for i^{th} view. We choose [14] as a 2D image encoder which takes an image $\mathbf{I}_i$ (with a resolution of $H \times W$) and outputs multi-scale feature pyramid. We pick the last three feature maps from the pyramid and reshape the features up to a resolution of $\frac{H}{8} \times \frac{W}{8}$. After concatenating resized features, we apply 1×1 convolution for channel reduction. Lastly, we obtain a single feature map $\mathbf{F}_i$ for each image, where the feature map conveys multi-scale and high-dimensional information of the image.

Volumetric feature encoding. We map the image features $\mathbf{F}_i$ to a unified 3D volumetric space. For convenience sake, we implement this mapping function as backward-warping using the known camera parameters (both intrinsics $\mathbf{K}_i$ and extrinsics $\mathbf{E}_i$). For each camera viewpoint i, we choose a set of voxels along a ray from a pixel $\mathbf{p}$ to the camera center and assign the image feature of a pixel $\mathbf{F}_i(\mathbf{p})$. As the feature $\mathbf{F}_i(\mathbf{p})$ at each pixel $\mathbf{p}$ contains high-level information along its pixel ray, we extract local 3D feature for each corresponding voxel coordinate (x, y, z) by concatenating it with a depth value of the voxel coordinate and pass it through a volumetric feature encoder. Some voxels are associated with multiple features due to spatial overlaps across the camera views. We, therefore, assign different MLPs based on the existence of the spatial overlaps and fuse the whole volumetric features. As a result, each voxel encodes a high-dimensional feature associated with 3D geometry.

3.3 Depth fusion

From the unified volumetric features, we aim to draw depth maps regarding an arbitrary rotated camera viewpoint. In this section, we present restructuring of the volumetric features with known camera parameters, a 2D depth map decoder, and a novel view depth synthesis.

Transformation of volumetric feature as projected image feature. Given the camera extrinsics $\mathbf{E}_i$, we transform the volumetric features $\mathbf{V}$ into the projected image feature $\tilde{\mathbf{F}}_i$ with a resolution of $\frac{H}{8} \times \frac{W}{8}$. For each pixel $\mathbf{p}$ at the target image coordinate, we uniformly sample volumetric features along its pixel ray and concatenate the sampled features to obtain the projected image feature $\tilde{\mathbf{F}}_i$.

Depth decoder. To predict a depth map from the projected image feature $\tilde{\mathbf{F}}_i$, we use a light-weighted depth decoder Dec_{depth} consisting of 4 convolutional layers: 3 convolutional layers for upsampling ($\frac{H}{8} \times \frac{W}{8} \rightarrow H \times W$) and one convolutional layer for the depth output.

Depth map synthesis at a novel view. We further extend our depth fusion module to produce a depth map for a novel viewpoint, which is not covered by the original multi-camera system. As the volumetric features encode the 3D structure in a certain range, we can represent a novel viewpoint using a projective matrix from the canonical coordinate. Without loss of generality, we transform the volumetric feature into a projected image feature using the projective matrix for the desired view instead of the camera pose intact. Specifically, we modulate a novel camera pose as a multiplication of canonical motion and the desired camera pose. Since spatial overlaps across the camera views share the common volumetric features, the proposed method predicts the consistent metric scaled depth maps. We visualize the synthesized depth maps at arbitrary rotated views in Fig. 1. Our method can seamlessly synthesize with variations of yaw/roll/pitch angles as well as arbitrary focal lengths.

3.4 Canonical motion estimation

The prior work [13] independently estimates the relative camera pose using a shared encoder-decoder network. However, this process is inefficient in that the multi-camera system shares the same canonical motion because they are attached to the vehicle. To improve, we introduce a canonical pose estimation network which takes the collapsed volumetric feature $\mathbf{F}_{BEV}$ and predicts a single canonical camera motion as illustrated in Fig. 2. We deterministically compute the rest of camera motions based on the known camera extrinsic parameters. Here, we use the same architecture design to obtain a volumetric feature, but with different trainable network weights and use two temporally consecutive images for the input.

Volumetric feature reduction. From given surround-view volumetric feature $\mathbf{V} \in \mathbb{R}^{\mathbb{X}\times\mathbb{Y}\times\mathbb{Z}\times\mathbb{C}}$ we collapse z-axis of the feature to flatten it as Birds-Eye-View (BEV) shape (see Fig. 3). We retain pose decoder from the PoseNet [11] while using the surround-view fusion feature where global ego-motion is reasoned. In detail, we collapse $\mathbf{Z}$ dimension into the channel dimension $\mathbf{C}$ (i. e. reshaping into a 3D tensor $\mathbf{V}' \in \mathbb{R}^{\mathbb{X}\times\mathbb{Y}\times(\mathbb{Z}\cdot\mathbb{C})}$) and apply 2D convolutions to reduce channel of the feature, resulting in a 3D tensor of $\mathbf{F}_{BEV} \in \mathbb{R}^{\mathbb{X}\times\mathbb{Y}\times\mathbb{C}'}$.

Computing local camera poses. Assuming a static relationship between the cameras, we distribute the predicted canonical camera motion to each camera motion. From the given extrinsic $\mathbf{E}_i$ of each camera C_i and canonical camera motion $\mathbf{T}^{t\rightarrow t+1}$, we compute the local camera pose as:

$$\mathbf{T}_i^{t\rightarrow t+1} = \mathbf{E}_i^{-1}\mathbf{E}_1\mathbf{T}^{t\rightarrow t+1}\mathbf{E}_1^{-1}\mathbf{E}_i, \tag{1}$$

where $\mathbf{E}_1$ is the extrinsic of the canonical camera motion. Note that any viewpoints could also be a canonical motion. In this work, we set the canonical motion as the front-view camera pose.

3.5 Self-supervised learning

Given the camera motion and depth map per each view, we apply our self-supervised loss given a temporal triplet of surround-view images. At test time, our method only requires images at the current time step t.

Self-supervised loss. Our self-supervised loss consists of the image reconstruction loss $\mathcal{L}_{\text{img}}$ and the depth synthesis loss $\mathcal{L}_{\text{depth}}$:

$$\mathcal{L} = \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{depth}}. \tag{2}$$

The image reconstruction loss penalizes the reconstruction error between the reference image $\mathbf{I}_t$ and the synthesized images $\tilde{\mathbf{I}}_{t+1}$ from each temporal, spatial, and spatio-temporal context (cf. Eq. (4)),

$$\mathcal{L}_{\text{img}} = \mathcal{L}_{\text{t}} + \lambda_{\text{sp}}\mathcal{L}_{\text{sp}} + \lambda_{\text{sp_t}}\mathcal{L}_{\text{sp_t}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}}. \tag{3}$$

Each term stands for the temporal $\mathcal{L}_{\text{t}}$, spatio $\mathcal{L}_{\text{sp}}$, spatio-temporal $\mathcal{L}_{\text{sp_t}}$, and smoothness loss $\mathcal{L}_{\text{smooth}}$. We use intensity difference and structure similarity [10, 11, 41] to measure the reconstruction error.

One main difference to the baseline work [13] is that we do not use the pose consistency loss because our method outputs one representative camera motion for all views, which reduces the number of unknowns and thus encourages convergence and improves accuracy.

Spatio-temporal context. The core idea is to exploit small overlaps between spatially or/and temporally neighboring frames for matching to provide supervisionary signal and scale information for camera motion and depth. For a reference image $\mathbf{I}_i^t$ from camera C_i at a time step t, it synthesizes reconstructed images $\tilde{\mathbf{I}}_i^t$ from *(i)* spatial neighboring frames $\mathbf{I}_j^t$ (j: indices of neighboring cameras), *(ii)* temporally neighboring frames $\mathbf{I}_i^{t'}$ ($t' \in \{t+1, t-1\}$), and *(iii)* spatio-temporally neighboring frames $\mathbf{I}_j^{t'}$, given the predicted camera motion. A pixel-wise warping operation for the image reconstruction is defined as:

$$\mathbf{\Pi}_{ij}^{t\to t'} = \mathbf{K}_j \mathbf{X}_{ij}^{t\to t'} \mathbf{D}_i \mathbf{K}_i^{-1} \tag{4a}$$

with

$$\mathbf{X}_{ij}^{t\to t'} = \begin{cases} \mathbf{T}_i^{t\to t'}, t' \in \{t-1, t+1\} & \text{for temporal context,} \\ \mathbf{E}_j \mathbf{E}_i^{-1} & \text{for spatio context,} \\ \mathbf{E}_j \mathbf{E}_i^{-1} \mathbf{T}_i^{t\to t'}, t' \in \{t-1, t+1\} & \text{for spatio-temporal context,} \end{cases} \tag{4b}$$

with estimated depth $\mathbf{D}_i$ and camera intrinsics $\mathbf{K}_i$ and $\mathbf{K}_j$. Here, $\mathbf{X}_{ij}$ is the camera motion between the two cameras depending on their spatio-temporal context.

By minimizing the difference between the reference image and reconstructed image, a self-supervised loss function guides the network to output metric-scale depths and camera motions that can satisfy the overall contexts.

Depth synthesis loss. We introduce a depth synthesis loss for the successful depth synthesis at a novel view:

$$\mathcal{L}_{\text{depth}} = \lambda_{\text{cons}}\mathcal{L}_{\text{cons}} + \lambda_{\text{depth_smooth}}\mathcal{L}_{\text{depth_smooth}}. \tag{5}$$

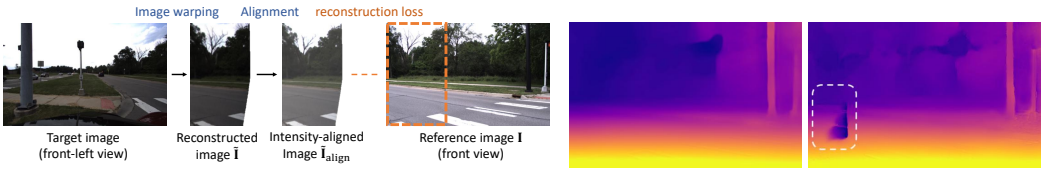
The depth consistency loss $\mathcal{L}_{\text{cons}}$ penalizes the depth difference between the synthesized depth at the novel view and depth at each known camera view i. We use the consistency loss from Bian et al. [1]. The smoothness loss $\mathcal{L}_{\text{depth_smooth}}$ [24] encourages the smoothness of the depth map at the novel view so that it regularizes a set of 3D voxel features retrieved at the novel view to improve 3D-awareness at each voxel coordinate. For the synthesis, we augment a depth map at a novel view for each known camera view i and apply the depth synthesis loss.

4 Experiments

4.1 Implementation details

Training. We implemented our networks in PyTorch [26] and trained on four A100-GPUs. For the image encoders we used ResNet-18 with weights pre-trained on ImageNet[33]. As some datasets exploit cameras with different focal lengths, we scaled the estimated depth range in accordance with the focal length of the camera.

All experiments used the same training hyper-parameters (unless explicitly mentioned): Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$; a mini-batch size of 2 per each GPU and a learning rate [34] with 1×10^{-4}, decaying at $\frac{3}{4}$ of the entire training schedule with a factor of 0.1; the previous $t-1$ and subsequent $t+1$ images are used as temporal context at the training time. Note that, our



(a) Image intensity alignment (b) With the alignment (c) Without the alignment

Figure 4: **Intensity distribution alignment**: *(a)* After warping the target image, we align the mean and variance of the reconstructed image to match with reference image's. *(b)* When aligning the distribution, it resolves the artifacts on the depth, which appears *(c)* when training without the alignment.

depth fusion module does not require temporal contexts, and therefore temporal contexts are not included in the test time. For our volumetric feature, we used voxel resolution of (1m, 1m, 0.75m) with spatial dimensions of (100, 100, 20) for (x, y, z) axis respectively. We use color jittering as data augmentation. For the depth synthesis loss, we use the random rotation with a range between [-5°, -5°, -25°] and [5°, 5°, 25°] for the depth map synthesis at a novel view. In the self-supervised loss in Eq. (2), we use depth smoothness weight $\lambda_{\text{smooth}} = 1 \times 10^{-3}$, spatio loss weight $\lambda_{\text{sp}} = 0.03$, spatio-temporal weight $\lambda_{\text{sp_t}} = 0.1$, depth consistency weight $\lambda_{\text{cons}} = 0.05$, and depth smoothness weight at novel views $\lambda_{\text{depth_smooth}} = 0.03$.

Dataset. We use the DDAD [12] and nuScenes [2] dataset for our experiments. Both datasets provide surround-view images from a total of 6 cameras mounted on a vehicle and LiDAR point clouds for the depth evaluation. We train our model on each train split and report the accuracy on the test split. During training, the input images are down-sampled to a resolution of 384×640 for the DDAD dataset, and that of 352×640 for nuScenes dataset. In evaluation, we considered the depth range up to 200m for the DDAD dataset, and 80m for the nuScenes dataset. We train our model on the DDAD dataset for 20 epochs and the nuScenes dataset for 5 epochs.

Evaluation protocols. We follow the same evaluation protocol from FSM [13], cropping depth range between up to 200m and using conventional monocular depth evaluation metric from Eigen et al. [7]. We use the same error metric from the standard evaluation with predicted depth d and ground-truth d^* for N pixels, computed as follows:

- Absolute relative error (Abs Rel): $\frac{1}{|N|}\sum_{i\in N}\frac{|d_i - d_i^*|}{d_i^*}$
- Square relative difference (Sq Rel): $\frac{1}{|N|}\sum_{i\in N}\frac{\|d_i - d_i^*\|^2}{d_i^*}$
- Root mean square error (RMSE): $\sqrt{\frac{1}{|N|}\sum_{i\in N}\|d_i - d_i^*\|^2}$
- Root mean squared logarithmic error (RMSE log): $\sqrt{\frac{1}{|N|}\sum_{i\in N}\|\log(d_i) - \log(d_i^*)\|^2}$
- Accuracy with threshold (δ): % of d_i s.t $\max\left(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}\right) = \delta < thr$

Intensity distribution alignment. We observe that different lighting conditions in neighboring images yields artifacts on the depth output (Fig. 4c) with the self-supervised loss in the previous work [13]. To address the issue, we align mean and variance on overlapped regions of the reference image $\mathbf{I}$ and synthesized image $\tilde{\mathbf{I}}$:

$$\tilde{\mathbf{I}}_{\text{align}} = \frac{\tilde{\mathbf{I}} - \tilde{\mu}}{\tilde{\sigma}}\sigma + \mu, \tag{6}$$

where $\{\mu, \sigma^2\}$ are the mean and variance of the reference image $\mathbf{I}$, and $\{\tilde{\mu}, \tilde{\sigma}^2\}$ are that of the synthesized image $\tilde{\mathbf{I}}$. The aligned image $\tilde{\mathbf{I}}_{\text{align}}$ is then used for the loss calculation in Eq. (2). Fig. 4a briefly illustrates this process. Here, the reconstructed image can be from either a spatial, temporal, or spatio-temporal context.

4.2 Surround-view evaluation

We compare the accuracy of our method with state-of-the-art methods. Especially for our closest baseline FSM [13], we use our own reproduced version of FSM for the comparison because the

Table 1: **Evaluation on the DDAD dataset [12] and nuScenes dataset [2]**. We compare our method with the state of the arts. We report the averaged accuracy from all views using the metric from Eigen et al. [7]. Median-scale accuracy uses ground truth for correctly scaling the depth map during evaluation [7].

Dataset	Scale	Method	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
DDAD	Median	Monodepth2 [11]	0.362	14.404	14.178	0.412	0.683	0.859	0.922
		PackNet-SfM [12]	0.301	5.339	14.115	0.395	**0.624**	0.828	0.908
		Our reproduce of FSM[13]	**0.219**	4.161	13.163	0.327	**0.703**	**0.880**	**0.940**
		Ours	0.221	**3.549**	**13.031**	**0.323**	0.681	0.874	**0.940**
	Metric	Our reproduce of FSM [13]	0.228	4.409	13.433	0.342	**0.687**	**0.870**	**0.932**
		Ours	**0.218**	**3.660**	**13.327**	**0.339**	0.674	0.862	**0.932**
nuScenes	Median	Monodepth2 [11]	0.287	3.349	**7.184**	**0.345**	0.641	0.845	0.925
		PackNet-SfM [12]	0.309	**2.891**	7.994	0.390	0.547	0.796	0.899
		Our production of FSM [13]	0.301	6.180	7.892	0.366	**0.729**	0.876	0.933
		Ours	**0.271**	4.496	7.391	0.346	0.726	**0.879**	**0.934**
	Metric	Our production of FSM [13]	0.319	7.534	7.860	0.362	**0.716**	0.874	0.931
		Ours	**0.289**	**5.718**	**7.551**	**0.348**	0.709	**0.876**	**0.932**

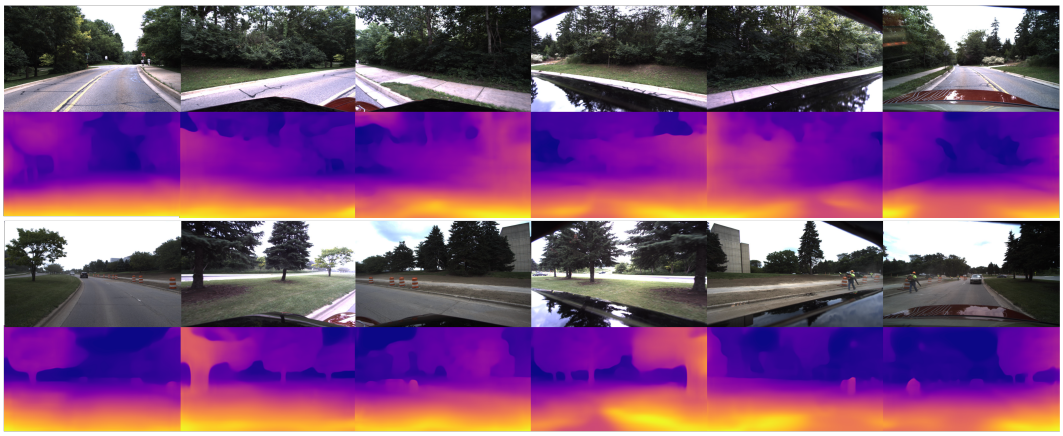


Figure 5: **Our results on DDAD dataset**: Our method successfully estimates depth at each view via volumetric feature fusion.

implementation is not provided. There is a gap between our reproduced version and the reported accuracy. However, for a fair comparison, we conduct experiments for all methods under the same training and evaluation settings and provide our attempts to reproduce and discussions in the supplementary material.

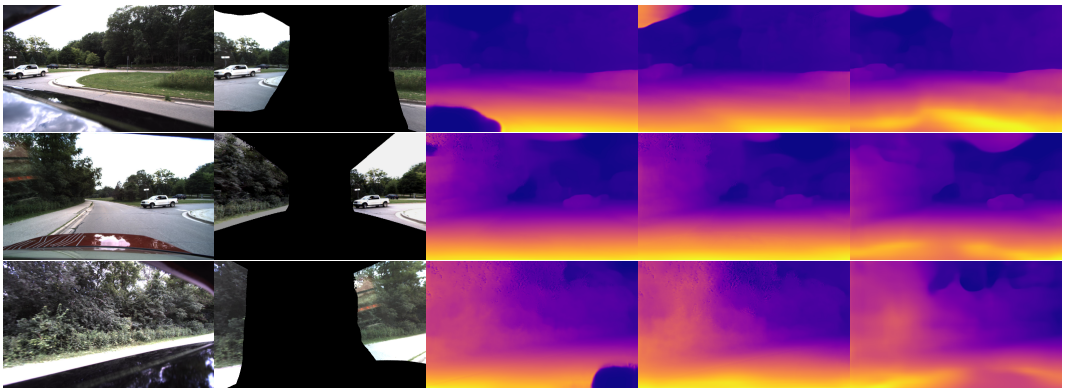
Table 1 demonstrates the evaluation on both DDAD [12] and nuScenes [2] datases. We also report the accuracy using per-frame median scaling for the reference. On the DDAD dataset, our method shows better/equal accuracy to the baseline method FSM [13] on 5 out of 7 metrics. Especially we substantially improve the accuracy on the *Sq Rel* metric by around 17%. On nuScenes dataset, our method demonstrates consistent improvement over the baseline, by outperforming the baseline on 6 out of 7 metrics, with over 24% improvement on the *Sq Rel* metric. Better accuracy is also observed when using the median scaling for evaluation.

4.3 Ablation study

To validate our main contributions, we conduct an ablation study of our depth fusion module and canonical motion estimation module over the FSM baseline [13]. We train the model on the DDAD dataset [12] while keeping the original experiment setup. As shown in Table 2, both of our contributions consistently improve the accuracy over the baseline. Our volumetric feature representation embeds effective 3D information and results in better depth accuracy; the only difference from previous works [11, 13] is a few additional multilayer perceptrons with volumetric feature representation while keeping the similar conventional encoder and decoder networks. Furthermore, we show the benefit of the volumetric feature presentation for the camera motion module; globally reasoning the camera motion benefits the depth estimation.

Table 2: **Ablation study on our pose, depth fusion modules**: Our canonical motion estimation and depth fusion module based on the volumetric feature representation consistently improve the depth accuracy, substantially on *Abs Rel* and *Sq Rel* metrics.

Canonical motion	Depth fusion	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
(Our reproduced baseline)		0.228	4.409	13.433	0.342	**0.687**	**0.870**	**0.932**
✓		0.222	3.969	13.492	0.344	0.677	0.865	0.931
✓	✓	**0.218**	**3.660**	**13.327**	**0.339**	0.674	0.862	**0.932**



(a) Input image (b) Overlapped region (c) Baseline (d) + Canonical motion estimation Our full model

Figure 6: **Qualitative comparison from our ablation study**: *(a)* Given a set of images (e. g. rear-left, rear, and rear-right), *(b)* estimated overlapped regions from nearby cameras, *(c)* depth results from our reproduced baseline, *(d)* depth results using the baseline model with our canonical motion estimation module, and our final results using both depth fusion and canonical motion estimation modules.

Fig. 6 visualizes each result in the ablation study. The depth map comparison results provide more evidence that our volumetric feature retrieves useful information as consistency between different views progressively improves as we add our individual modules.

5 Conclusions

We proposed a novel volumetric feature representation for self-supervised surround-view depth estimation. The volumetric feature representation aggregates image features from surround-view images, encodes 3D information, and then is used to estimate a depth map at each view and canonical motion between two temporally consecutive images. Furthermore, by projecting the volumetric feature at arbitrary rotated views, our method can also synthesize a depth map at the novel view, additionally with varying focal lengths. Our method outperforms the state-of-the-art surround-view depth method. The ablation study successfully validates our contributions. In future work, we expect our volumetric feature representation can benefit other computer vision tasks such as object detection, image segmentation, and motion estimation in the surround-view camera setup.

Limitation. Our volumetric feature representation is defined in a 3D cubic space; therefore it needs to take the trade-off between memory usage and the resolution into consideration. Because we focus on embedding 3D information to a novel representation, sometimes some blurred regions occurs in the depth map (e. g. in Fig. 6 and Fig. 1). The usage of cylindrical or spherical coordinates would improve the efficiency of the memory usage. Furthermore, some artifacts sometimes appear in the synthesized depth map in Fig. 1 where no input image feature is given. In future work, we will consider using additional regularization terms to preserve details of depth results as well as reduce the artifact.

Potential negative societal impacts. Methods for surround-view depth estimation require a large number of images for training, which may include images with privacy issues. To prevent this, the dataset should be precisely curated and be used for the research purpose only.

References

[1] Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *NeurIPS*, 2019.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020.

[3] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *AAAI*, pages 8001–8008, 2019.

[4] Hsien-Tzu Cheng, Chun-Hung Chao, Jin-Dong Dong, Hao-Kai Wen, Tyng-Luh Liu, and Min Sun. Cube padding for weakly-supervised saliency prediction in 360° videos. In *CVPR*, pages 1420–1429, 2018.

[5] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Federico Tombari. SO-Pose: Exploiting self-occlusion for direct 6D pose estimation. In *ICCV*, pages 12396–12405, 2021.

[6] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3D object detection. In *CVPR*, pages 11672–11681, 2020.

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

[8] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, pages 2002–2011, 2018.

[9] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *ECCV*, pages 740–756, 2016.

[10] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[11] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, pages 3828–3838, 2019.

[12] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3D packing for self-supervised monocular depth estimation. In *CVPR*, pages 2485–2494, 2020.

[13] Vitor Guizilini, Igor Vasiljevic, Rares Ambrus, Greg Shakhnarovich, and Adrien Gaidon. Full surround monodepth from multiple cameras. *RA-L*, 2021.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[15] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *CVPR*, pages 2821–2830, 2018.

[16] Jan Issac, Manuel Wüthrich, Cristina Garcia Cifuentes, Jeannette Bohg, Sebastian Trimpe, and Stefan Schaal. Depth-based object tracking using a robust gaussian filter. In *ICRA*, 2016.

[17] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *NIPS*, 2017.

[18] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, pages 239–248, 2016.

[19] Tuan-Tang Le, Trung-Son Le, Yu-Ru Chen, Joel Vidal, and Chyi-Yeu Lin. 6D pose estimation with combined deep learning and 3D vision techniques for a fast and accurate object grasping. *Robotics and Autonomous Systems*, 141:103775, 2021.

[20] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. In *arXiv:1907.10326 [cs.CV]*, 2019.

[21] Chenxu Luo, Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, Ram Nevatia, and Alan Yuille. Every pixel counts++: Joint learning of geometry and motion with 3D holistic understanding. *TPAMI*, 2019.

[22] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3D geometric constraints. In *CVPR*, pages 5667–5675, 2018.

[23] Ishit Mehta, Parikshit Sakurikar, and PJ Narayanan. Structured adversarial training for unsupervised monocular depth estimation. In *3DV*, pages 314–323, 2018.

[24] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. RegNeRF: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022.

[25] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3D object detection? In *ICCV*, 2021.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[27] Vaishakh Patil, Wouter Van Gansbeke, Dengxin Dai, and Luc Van Gool. Don't forget the past: Recurrent depth estimation from monocular video. *RA-L*, 5(4):6813–6820, 2020.

[28] Sudeep Pillai, Rareş Ambruş, and Adrien Gaidon. SuperDepth: Self-supervised, super-resolved monocular depth estimation. In *ICRA*, pages 9250–9256, 2019.

[29] Matteo Poggi, Fabio Tosi, and Stefano Mattoccia. Learning monocular depth estimation with unsupervised trinocular assumptions. In *3DV*, pages 324–333, 2018.

[30] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021.

[31] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J. Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, 2019.

[32] Patrick Ruhkamp, Daoyi Gao, Hanzhi Chen, Nassir Navab, and Beniamin Busam. Attention meets Geometry: Geometry guided spatial-temporal attention for consistent self-supervised monocular depth estimation. In *3DV*, pages 837–847, 2021.

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[34] Leslie N. Smith and Nicholay Topin. Super-Convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, page 1100612, 2019.

[35] Ales Leonardis Song Yan, Jinyu Yang and Joni-Kristian Kamarainen. Depth-only object tracking. In *BMVC*, 2021.

[36] Igor Vasiljevic, Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Wolfram Burgard, Greg Shakhnarovich, and Adrien Gaidon. Neural ray surfaces for self-supervised learning of depth and ego-motion. In *3DV*, pages 1–11, 2020.

[37] Fu-En Wang, Hou-Ning Hu, Hsien-Tzu Cheng, Juan-Ting Lin, Shang-Ta Yang, Meng-Li Shih, Hung kuo Chu, and Min Sun. Self-supervised learning of depth and camera motion from 360° videos. In *ACCV*, 2018.

[38] Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. BiFuse: Monocular 360° depth estimation via bi-projection fusion. In *CVPR*, pages 459–468, 2020.

[39] Ning-Hsu Wang, Bolivar Solarte, Yi-Hsuan Tsai, Wei-Chen Chiu, and Min Sun. 360SD-Net: 360° stereo depth estimation with learnable cost volume. In *ICRA*, pages 582–588, 2020.

[40] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving. In *CVPR*, pages 8445–8453, 2019.

[41] Jamie Watson, Oisin Mac Aodha, Victor Prisacariu, Gabriel Brostow, and Michael Firman. The Temporal Opportunist: Self-supervised multi-frame monocular depth. In *CVPR*, pages 1164–1174, 2021.

[42] Felix Wimbauer, Nan Yang, Lukas Von Stumberg, Niclas Zeller, and Daniel Cremers. MonoRec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera. In *CVPR*, pages 6112–6122, 2021.

[43] Changhee Won, Jongbin Ryu, and Jongwoo Lim. OmniMVS: End-to-end learning for omnidirectional stereo matching. In *CVPR*, pages 8987–8996, 2019.

[44] Shang-Ta Yang, Fu-En Wang, Chi-Han Peng, Peter Wonka, Min Sun, and Hung-Kuo Chu. DuLa-Net: A dual-projection network for estimating room layouts from a single RGB panorama. In *CVPR*, pages 3363–3372, 2019.

[45] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. LEGO: Learning edge with geometry all at once by watching videos. In *CVPR*, pages 225–234, 2018.

[46] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018.

[47] Zhichao Yin and Jianping Shi. GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018.

[48] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, pages 1851–1858, 2017.

[49] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. OmniDepth: Dense depth estimation for indoors spherical panoramas. In *ECCV*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Sec. 1
   (b) Did you describe the limitations of your work? [Yes] See Sec. 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Sec. 4, and additional information is described in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 4.1
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The experiments were done using the fixed seed value to reduce reproducibility issue. Further, code is provided in the supplementary material
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec. 4.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. 4, as we use public research dataset, we've cited their works.
   (b) Did you mention the license of the assets? [Yes] In the supplementary materials, we include the license condition of code basis and public datasets that we used.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our implementation in the supplementary material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We've used public dataset and cited there work in the paper.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] Public surround-view depth datasets do not contain any personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] Crowdsourcing or human subjected research was not included in this work.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]