Semi-SMD: Semi-Supervised Metric Depth Estimation via Surrounding Cameras for Autonomous Driving

Yusen Xie, Zhengmin Huang, Shaojie Shen, and Jun Ma

arXiv:2503.19713v3 [cs.RO] 9 Sep 2025

***Abstract*— In this paper, we introduce Semi-SMD, a novel metric depth estimation framework tailored for surrounding cameras equipment in autonomous driving. In this work, the input data consists of adjacent surrounding frames and camera parameters. We propose a unified spatial-temporal-semantic fusion module to construct the visual fused features. Cross-attention components for surrounding cameras and adjacent frames are utilized to focus on metric scale information refinement and temporal feature matching. Building on this, we propose a pose estimation framework using surrounding cameras, their corresponding estimated depths, and extrinsic parameters, which effectively address the scale ambiguity in multi-camera setups. Moreover, semantic world model and monocular depth estimation world model are integrated to supervise the depth estimation, which improve the quality of depth estimation. We evaluate our algorithm on DDAD and nuScenes datasets, and the results demonstrate that our method achieves state-of-the-art performance in terms of surrounding camera based depth estimation quality. The source code is available on GitHub[1].**

I. Introduction

Metric depth estimation provides absolute distance perception of the surrounding environment in autonomous driving scenarios, supporting the following tasks such as motion forecasting [1] and path planning [2], [3]. Some existing framworks [4], [5] directly detect depth by introducing LiDAR and Radar sensors, but these approaches lead to increased costs and algorithmic complexity. Currently, autonomous driving perception systems tend to favor visual-only surrounding camera solutions [2], [6] that are easier to implement. Some visual-based depth prediction algorithms [7], [8], [9], [10], [11] use large datasets to train monocular depth estimation (MDE), achieving impressive results in depth estimation as world models. However, the scale ambiguity encountered in monocular depth estimation makes it unsuitable for applications requiring precise depth information in autonomous driving. Surrounding cameras, which inherently provide scale information from extrinsic parameters, have seen rapid development in metric depth prediction [12], [13], [7]. These methods typically use data from the same frame [7] or from the same camera [12], [13], but do not take advantage of the stereo geometric constraints provided by spatial and temporal information, which results in insufficient accuracy and poor generalization. Additionally, some methods [14], [15] use semi-supervised training guidance but face challenges in decoupling complex tasks efficiently. Therefore, the results are not accurate and easy to convergence to local optima. Lastly, these methods fail to integrate semantic information effectively, leading to unclear boundaries in depth maps, particularly in areas where humans rely on semantic cues.

To address these issues, we propose a **s**urrounding camera **m**etric **d**epth estimation framework, named **Semi-SMD**. The results of the paper are briefly presented in Fig. 1. By utilizing two adjacent frames of surrounding camera images, we employ a unified spatial-temporal-semantic transformer [17] to fuse visual features extracted by ResNet [18] and semantic features from a semantic segmentation world model [19]. This fused features are then used for both depth prediction and pose estimation to boost computational efficiency. Furthermore, we integrate depth prediction and the extrinsic parameters of the surrounding camera into the pose estimation network, redesigning a surrounding camera pose estimation module with precise scale information. Additionally, we introduce a curvature loss based on the depth estimation world model [8], [9]. Experimental results demonstrate that the inclusion of this loss function significantly enhances the model's convergence speed and depth prediction accuracy. Our contributions are summarized as follows:

- We propose a unified spatial-temporal-semantic transformer that fuses surrounding cameras, adjacent frames, and semantic features. As the feature extraction backbone of our model, this module effectively merges spatial-temporal and semantic information to improve accuracy while reducing computational consumption.
- We design a joint pose estimation network for surrounding cameras that integrates depth and extrinsic parameters, and this improves the network's interpretability while enhancing the accuracy of pose prediction.
- We integrate the depth prediction world model into the loss function module and design a gradient-based curvature loss function, which accelerates the convergence and improves the quality of depth estimation.
- We conduct comprehensive validation on two widely used datasets. The results demonstrate the superior capability of our algorithm to achieve SOTA performance in metric depth estimation for autonomous driving.

II. Related Works

A. Scale-Ambiguous Depth Estimation

Monocular depth estimation begins by predicting depth information from a single image [20], [21], [22], [23].

Yusen Xie, Zhenmin Huang, and Jun Ma are with the Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China yxie827@connect.hkust-gz.edu.cn; zhuangdf@connect.ust.hk; jun.ma@ust.hk

Shaojie Shen is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China eeshaojie@ust.hk

[1]https://github.com/xieyuser/Semi-SMD

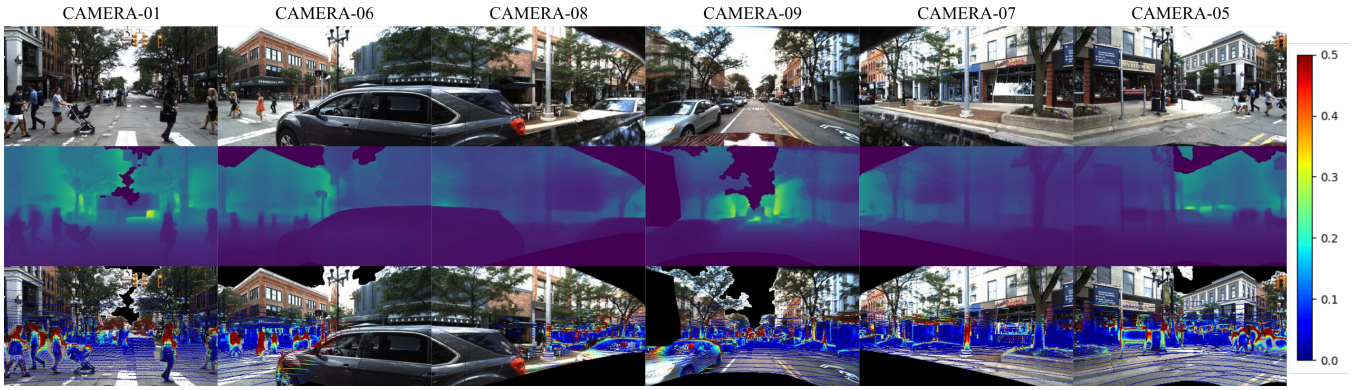


Fig. 1. We use adjacent surrounding camera images and the camera's intrinsic and extrinsic parameters as input, while simultaneously estimating the pose transformation between the two frames and the metric depth for each image. An illustration of our metric depth estimation results on the DDAD [16] dataset is shown. The first row displays the original surrounding RGB images, the second row shows the estimated metric depth, and the third row provides a quantitative visualization of *Abs.Rel.* with the projected LiDAR points. The colors represent the error distribution.

However, this approach typically builds a scale-ambiguous depth estimation world model rather than providing metric depth estimation due to the lack of scale information. Some studies [11], [10], [16] use adjacent frames to calculate scale information, enabling continuous depth estimation in image sequences under reprojection error supervision. Building on this, ManyDepth [24] and [25] introduce outlier rejection methods for dynamic objects to reduce errors. Surround-Depth [15] and [26] utilize additional data sources, such as depth priors and velocity, to improve results. However, due to the absence of reliable scale information, these methods are not applicable in scenarios requiring high metric precision.

B. Metric Depth Estimation

MonoRec [13] introduces a visual odometry system [27] to provide relative pose estimation and sparse depth supervision, enabling scale-aware depth estimation. R3D3 [12] proposes a multi-camera dense bundle adjustment method and a multi-camera co-visibility graph to compute accurate poses. However, modules in these works [13], [12] are not entirely differentiable, which increases the complexity of manually designed principles. VFDepth [28] constructs a volumetric feature representation as the backbone of the entire model, but the depth predictions are not clearly separated at the edge areas due to the lack of semantic information. SurroundDepth [15] uses a cross-view transformer to exchange information, but the pose estimation network is computationally expensive due to the separate feature encoder. To achieve accurate metric depth estimation, the extrinsic parameters of a multi-camera setup and precise adjacent pose estimation are key factors.s Moreover, pose estimation solely derived from image data is difficult to convergence. M^2-Depth [14] utilizes image pairs from a single camera to predict pose, but neglects the scale information provided by the extrinsic of surrounding cameras.

C. World Model Guided Metric Depth Estimation

In recent years, with the rapid advancement of data and computational resources, remarkable progress has been made in semantic segmentation world models [29], [30], [19], [31] and depth estimation world models [7], [8], [9]. Some methods [11], [10], [15] that rely on reprojection error supervision are prone to degeneration in areas with semantically similar features. In contrast, M^2-Depth [14] and DepthAnything [8], [9] incorporate semantic information into depth estimation tasks, significantly improving both the accuracy and plausibility of depth predictions. Although the depth estimated by world models [7], [8], [9] is scale-ambiguous, the relative depth information they provide can guide the estimation of metric depth.

III. Methodology

A. Problem Formulation and Semantic Feature Adapter

Problem Formulation. The input data in every iteration consists of G ($G = 2$ in our paper) adjacent surrounding frames, with each frame containing N images. The nth image is defined as $\mathcal{I}_n^{3\times H\times W}$, where $0 \leq n < N$. H and W denote the height and width of the image, respectively. We assume that the intrinsics π_n of nth image and the extrinsic $\mathcal{E}_n$ between the nth camera and the base coordinate system are known. The G frames are divided into source *(src)* frames and target *(tgt)* frames. Our framework aims to predict the inverse depth $\mathcal{D}_n^{1\times H\times W}$ for each image $\mathcal{I}_n^{3\times H\times W}$, as well as the pose $\mathcal{P}^{(G-1)\times 6}$ between the *src* frames and the *tgt* frames, represented by the 6-DOF axis-angle notation.

A brief overview of our framework is illustrated in Fig. 2. We extract visual features from combined images $\mathcal{I}^{G\times N\times 3\times H\times W}$ by ResNet [18], and the output of the L-layers extracted features is denoted as $\mathcal{F}_{vis} = \{\mathcal{F}_{(l)}^{G\times N\times C_l\times H_l\times W_l}, 0 \leq l < L\}$, where C_l, H_l, and W_l represent the number of channels, feature height, and feature width at the lth layer, respectively. Meanwhile, semantic features $\mathcal{F}_{seg}$ are obtained from combined images $\mathcal{I}^{G\times N\times 3\times H\times W}$ by the frozen SAM feature encoder [31]. Features $\mathcal{F}_{vis}$ and $\mathcal{F}_{seg}$ are then passed through a self-designed spatial-temporal-semantic transformer (STST) module (detailed in Sec. III-B) to fuse features. This fused feature, which aggregates temporal, spatial, and semantic information, serves as the input for both the depth decoder and the pose decoder, thereby avoiding redundant feature extraction [15], [14] from the raw images and making the network more lightweight. In Sec. III-C, we introduce the structure of the STST-enhanced joint pose estimation

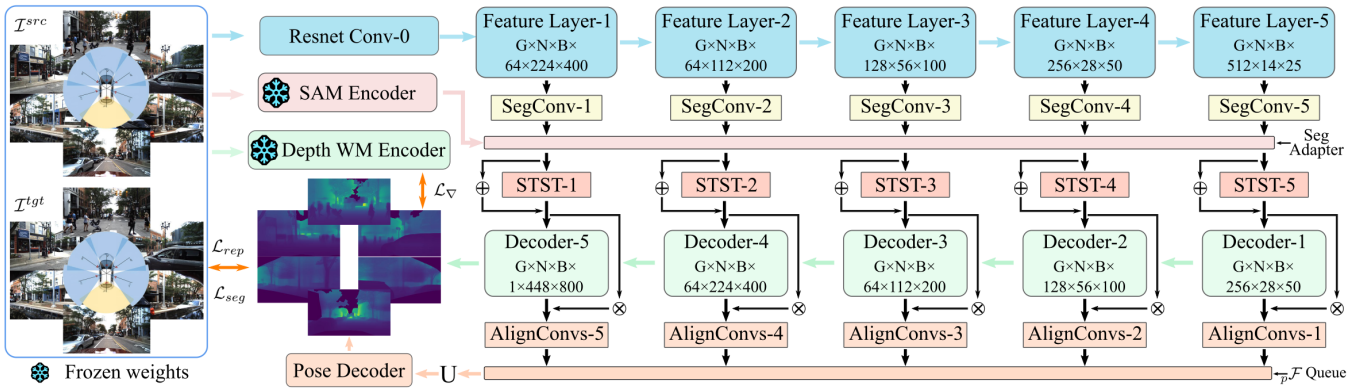


Fig. 2. Overview of our proposed framework. Our framework takes two adjacent surrounding frames as input. The core of the system consists of a feature extraction layer based on ResNet [18] and a depth decoder. Between the encoder and decoder in each layer, we use self-designed STST module to perform high-dimensional information fusion. Then the output from the depth decoder and the fused feature from the STST module are combined using the Hadamard product ($\otimes$), as the input of surrounding camera pose estimation network. The final output includes both a depth map and joint pose estimation. $\oplus$ represents element-wise addition of tensors. U refers to the operation of concatenating the features from all layers.

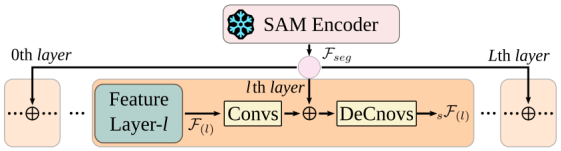


Fig. 3. Illustration of the semantic adapter network at the lth layer. All L layers receive the same semantic feature $\mathcal{F}_{seg}$.

network. Following this, we detail the implementation of our loss function and network in Sec. III-D.

Semantic Feature Adapter. We integrate semantic information pretrained by the semantic world model [31], [19] into the our network, constructing a fused feature to enhance depth estimation performance. $\mathcal{F}_{seg}$ denotes the semantic feature from MobileSAM [31] with dimensions $C_g \times H_g \times W_g$. For lth *layer* visual features $\mathcal{F}_{(l)}$, specific convolution and sampling operations are adapted to align with the dimensions of $\mathcal{F}_{seg}$. This process can be expressed as

$$_s\mathcal{F}_{(l)} = DeConvs^s_{(l)}\left(Convs^s_{(l)}(\mathcal{F}_{(l)}) + \mathcal{F}_{seg}\right) \quad (1)$$

where $Convs^s_{(l)}$ and $DeConvs^s_{(l)}$ represent the lth layers convolution and sampling operations designed to align the feature $\mathcal{F}_{(l)}$ with the dimensions of $\mathcal{F}_{seg}$.

The architecture we design integrates spatical-temporal features into the semantic layer, effectively avoiding the typical increase in parameters that arises when mapping from the semantic layer back to the feature layer. This approach can enhance depth estimation clarity in boundary area while maintaining computational efficiency.

B. Spatial-Temporal-Semantic Transformer

Framework of Spatial-Temporal-Semantic Transformer. We build a unified spatial-temporal-semantic transformer architecture to fuse the features $\mathcal{F}_{vis}$ and $\mathcal{F}_{seg}$ extracted from images. For all L-layers, we use L STST modules to facilitate feature exchange across surrounding cameras and adjacent frames. The following section explains the construction process of STST in detail, using the lth layer as an example.

The overall structure of the lth STST is shown in Fig. 4. For the feature at the lth layer, $_s\mathcal{F}_{(l)}$, we output the feature $_{sts}\mathcal{F}_{(l)}$ with spatial-temporal-semantic information, which retains the same dimension. To reduce computational cost, we first down-sample the feature $_s\mathcal{F}^{G\times N\times C_l\times H_l\times W_l}_{(l)}$ with different down-sampling ratios for different layers, such that the shape of down-sampled feature matches the unified input shape of STST $G \times N \times \overline{C} \times \overline{H} \times \overline{W}$. Note that $\overline{C}$, $\overline{H}$ and $\overline{W}$ denote the fixed number of channels, feature width, and feature height accepted by STST. Further details are provided in our code.

Then, by applying self-designed surrounding camera attention and adjacent frame attention detailed in Sec. III-B, followed by upsampling to restore the shape, we obtain the desired output $_{sts}\mathcal{F}^{G\times N\times C_l\times H_l\times W_l}_{(l)}$. Additionally, we use skip connections between the encoder and decoder of the lth layer to retain gradient information.

Spatial Surrounding Camera Attention and Temporal Adjacent Frame Attention. By leveraging features from overlapping regions and extrinsic parameters between cameras, we design a spatial attention mechanism for neighboring cameras to inference the metric scale information. Unlike SurroundDepth [15], we reduce computational load by only computing interactions with clockwise neighbors. Additionally, we introduce a temporal attention mechanism for inter-frame feature matching to improve joint pose estimation accuracy. The cross-attention module is shown in Fig. 4, the input feature is $_s\mathcal{F}^{G\times N\times\overline{C}\times\overline{H}\times\overline{W}}_{(l)}$ and the output feature with same dimension is $_{sts}\mathcal{F}^{G\times N\times\overline{C}\times\overline{H}\times\overline{W}}_{(l)}$.

In this module, we incorporate learnable positional encoding $L^{G\times N\times 1\times 1\times 1}_{GN}$ in both the G and N dimensions. The feature with element-wise positional encoding is represented as

$$_s\mathcal{F}^{G\times N\times\overline{C}\times\overline{H}\times\overline{W}}_{(l)} = {}_s\mathcal{F}^{G\times N\times\overline{C}\times\overline{H}\times\overline{W}}_{(l)} \oplus bc(L_{GN}) \quad (2)$$

where bc refers to using the broadcasting mechanism to expand the dimensions of L_{GN}, transforming it into $L^{G\times N\times\overline{C}\times\overline{H}\times\overline{W}}_{GN}$. Based on this, we use a linear layer $\langle \mathbf{W_{in}}, \mathbf{b_{in}} \rangle$ to map this feature to $^{proj}\mathcal{F}_{(l)} = \mathbf{W_{in}} \cdot Flatten(\mathcal{F}_{(l)}) + \mathbf{b_{in}}$, where $Flatten$ refers to the operation of converting a tensor into a one-dimensional array. Then,

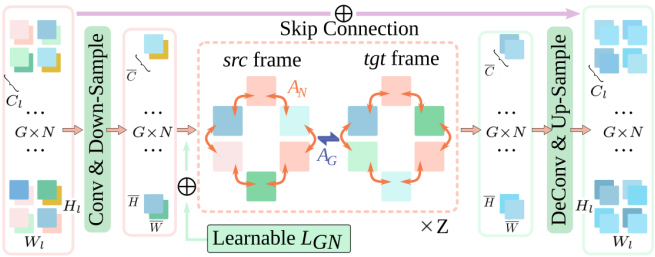


Fig. 4. Illustration of lth layer spatial-temporal-semantic transformer. Spatial attention is constructed using adjacent cameras, while temporal attention is built using adjacent frames. Before the attention calculation, the positions of the features are encoded in a learnable manner.

for the nth camera, we compute the cross-attention score A_N with its adjacent camera. The query, the key and value are defined as

$$\begin{aligned} Q_n &= {}^{proj}_{s}\,\mathcal{F}_{(l)}[:, n, ...] \\ K_{n+1}, V_{n+1} &= {}^{proj}_{s}\,\mathcal{F}_{(l)}[:, (n+1)\%N, ...] \end{aligned} \tag{3}$$

The computation of spatial attention A_N is given by

$$A_N = softmax\left(\frac{Q_n K_{n+1}^{\top}}{\sqrt{d_{n+1}}}\right) V_{n+1} \tag{4}$$

For the src frame, we compute the cross-attention score A_G with its adjacent tgt frame. The query, key and value are defined as defined as

$$\begin{aligned} Q_{src} &= {}^{proj}_{s}\,\mathcal{F}_{(l)}[src, ...] \\ K_{tgt}, V_{tgt} &= {}^{proj}_{s}\,\mathcal{F}_{(l)}[tgt, ...] \end{aligned} \tag{5}$$

The computation of temporal attention A_G is given by

$$A_G = softmax\left(\frac{Q_{src} K_{tgt}^{\top}}{\sqrt{d_{tgt}}}\right) V_{tgt} \tag{6}$$

We then combine the computed features ${}_{attn}\mathcal{F}_{(l)} = \langle A_N, A_G \rangle$ and apply layer normalization followed by linear transformation $\langle \mathbf{W_0}, \mathbf{b_0} \rangle, \langle \mathbf{W_{out}}, \mathbf{b_{out}} \rangle$ to project it back to the original dimension:

$$\begin{aligned} {}_{norm}\mathcal{F}_{(l)} &= LayerNorm({}_{attn}\mathcal{F}_{(l)}) \\ {}_{sts}\mathcal{F}_{(l)} &= \mathbf{W_{out}} \cdot ReLU(\\ &\quad \mathbf{W_0} \cdot_{norm} \mathcal{F}_{(l)} + \mathbf{b_0}) + \mathbf{b_{out}} \end{aligned} \tag{7}$$

To reduce the computational cost of the attention module, we employ a convolutional projection in practice to decrease the channel dimension C. For simplicity, this modification is omitted in the theoretical derivation above. The implementation details are available in our source code.

C. Multi-Camera Enhanced Pose Estimation

Unlike most image-based methods [14], [15], [10], we believe that the results of depth map prediction can provide geometric guidance for pose estimation. Furthermore, the coarse-to-fine pose refinement approach used in traditional methods can still be applied in deep neural network. Therefore, we combine the features ${}_{sts}\mathcal{F}_{(l)}$ extracted from the lth layer of the STST backbone with the depth prediction ${}_{d}\mathcal{F}_{(l)}$ from the lth depth decoder, and then utilize a multi-stage convolutional network to integrate these features.

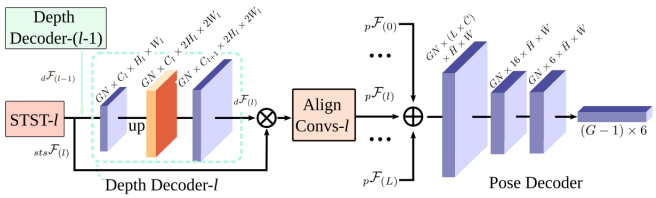


Fig. 5. Illustration of the pose feature fusion network at the lth layer and the pose decoder. By overlaying the predicted depth features onto the image features, depth-guided information is provided for the global pose estimation of the surrounding camera.

At the lth layer, we unitize scale-adaptive compression as:

$${}_{p}\mathcal{F}_{(l)} = Convs^{d}_{(l)}({}_{sts}\mathcal{F}_{(l)} \otimes {}_{d}\mathcal{F}_{(l)}) \tag{8}$$

where $Convs^{d}_{(l)}$ denotes the convolution and sampling kernel. The concatenated features of all L layers will be ${}_{p}\mathcal{F} = \cup_{l=1}^{L}({}_{p}\mathcal{F}_{(l)})$. After aligning the feature dimensions, a lightweight pose decoder with convolution and mean operations is employed to generate the final joint pose estimation $\mathcal{P}_{s\to t}$. The overview of pose decoder and lth depth decoder are shown in Fig. 5.

Upon obtaining the prediction of global pose transformation $\mathcal{P}_{s\to t}$. The extrinsic parameters are utilized to convert the global transformation into individual pose transformations within the coordinate system of each camera by

$$\mathcal{P}_n = \mathcal{E}_n^{-1} \cdot \mathcal{P}_{s\to t} \cdot \mathcal{E}_n \tag{9}$$

where $\mathcal{P}_n$ denotes the pose transformation of the nth camera coordinate system, while and $\mathcal{E}_n$ represents the pre-calibrated extrinsic parameters between nth camera and the base coordinate. Once the poses for all cameras are calculated, the image-level loss is computed to provide supervision information.

D. Implementation Details

In our framework, we predict a full-size depth map, and then reshape it to the kth level ($0 < k \leq K, K = 3$ in our experiments) in a pyramid structure. The loss is calculated across all K levels.

Sparse Depth Loss. Previous methods [14], [15], [28], [12] rely on LiDAR point clouds captured at a single moment within an image frame for depth estimation validation. To ensure fair comparison, we also utilize the same data for metric validation and sparse supervision.

To compute the loss between the ground truth depth image and the predicted depth image, we select depth values within a specific range $D_{min} \to D_{max}$, and calculate the absolute error by

$$L_d = \frac{1}{D_{max}} \sum_{k=0}^{K} \| \frac{1}{\mathcal{D}^{gt}_{(k)}} - \frac{1}{\mathcal{D}^{pred}_{(k)}} \|_1 \tag{10}$$

where $\mathcal{D}^{gt}_{(k)}$ and $\mathcal{D}^{pred}_{(k)}$ represent the ground truth and predicted inverse depth in kth level, respectively. Note that, this adjusts the data distribution of inverse depth resulting in a more uniform distribution that is more conducive for supervision.

Curvature Loss. In our experiments, first-order gradient-based methods [15], [14], [10] perform poorly in predicting

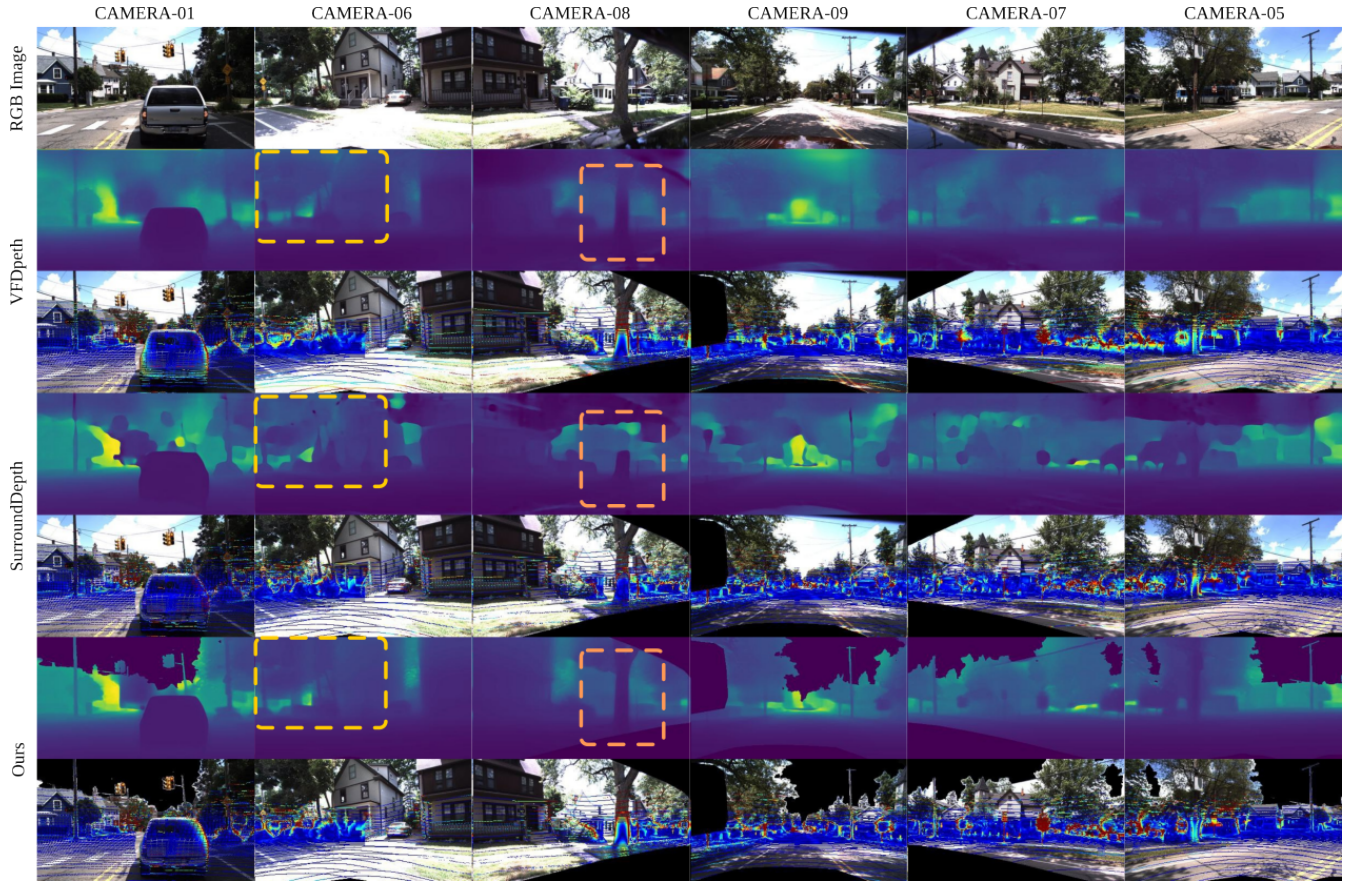


Fig. 6. Illustration of the metric depth estimation result on DDAD [16]. For each framework, we present the depth estimation result and the visilization of metric *Abs.Rel*. The error distribution is same as Fig. 1. The boxes highlight some significant comparison details.

depth details. We find that although the depth estimation world model [7], [8], [9] is not accurate in scale, its depth prediction distribution closely resembles the real distribution. Therefore, we use curvature (i.e., second-order gradients) to measure the loss between the predicted depth $\mathcal{D}^{pred}$ and the depth estimation $\mathcal{D}^{wm}$ from MobileSAM [9].

We define a t-step ($0 < t \leq T, T = 3$ in our experiments) depth gradient operator for coordinate x, y as

$$\nabla^{(t)} \{\mathcal{D}(y,x)\} = \mathcal{D}(y,x) - \mathcal{D}(y+t, x+t) \tag{11}$$

which can be applied to calculate curvature of the depth by

$$C^{(t)}\mathcal{D}(y,x) = \nabla^{(t)} \left\{\nabla^{(t)} \{\mathcal{D}(y,x)\}\right\} \tag{12}$$

Then, the curvature loss $\mathcal{L}_{\nabla}$ is defined as

$$\mathcal{L}_{\nabla} = \sum_{k=0}^{K}\sum_{t=1}^{T} \|C^{(t)}\mathcal{D}^{pred}_{(k)} - C^{(t)}D^{wm}_{(k)}\|_1 \tag{13}$$

Our experiments show that this curvature loss significantly improves the performance of depth prediction in edge details. More details can be found in Sec. IV-C.

Reprojection Loss and Semantic Loss. We predict the inverse depth $\mathcal{D}^{src}_{(n)(k)}$ of nth image in kth level and relative pose $\mathcal{P}_n$. The operator $\Lambda^{(k)}_{(n)}$ is defined as reprojecting image $\mathcal{I}^{src}_{(n)(k)}$ to 0th layer $\mathcal{I}^{tgt}_{(n)(0)}$ by combining with sky and optional vehicle body occlusion mask ($\mathcal{M}^{src}_{(n)(k)}$, $\mathcal{M}^{tgt}_{(n)(k)}$):

$$\Lambda^{(k)}_{(n)} = \pi^{tgt}_{(n)(0)}\mathcal{P}_n(\pi^{src}_{(n)(k)})^{-1}(\mathcal{M}^{src}_{(n)(k)} \odot \mathcal{D}^{src}_{(n)(k)} \odot \mathcal{I}^{src}_{(n)(k)}) \tag{14}$$

where $\pi^{src}_{(n)(k)}$ and $\pi^{tgt}_{(n)(0)}$ are the intrinsic of $\mathcal{I}^{src}_{(n)(k)}$ and $\mathcal{I}^{tgt}_{(n)(0)}$, respectively. $\odot$ denotes the Hadamard product applied element-wisely to the predicted depth. We calculate reprojection loss $\mathcal{L}_{rep}$ via L1 distance and SSIM distance [32] between N src frames $\mathcal{I}^{src}_{(n)(k)}$ and N tgt frames ${}^{M}\mathcal{I}^{tgt}_{(n)(0)} \leftarrow M^{tgt}_{(n)(0)} \odot \mathcal{I}^{tgt}_{(n)(0)}$. $\mathcal{L}_{rep}$ is defined as

$$\begin{aligned}\mathcal{L}_{rep} = \sum_{l=0}^{K}\sum_{n=0}^{N} \Bigg(&\lambda_{l1}\|{}^{M}\mathcal{I}^{tgt}_{(n)(0)} - \Lambda^{(k)}_{(n)}(\mathcal{I}^{src}_{(n)(k)})\|_1 \\ &+ (1-\lambda_{l1})\,\mathrm{SSIM}({}^{M}\mathcal{I}^{tgt}_{(n)(0)}, \Lambda^{(k)}_{(n)}(\mathcal{I}^{src}_{(n)(k)}))\Bigg)\end{aligned} \tag{15}$$

where $\lambda_{l1} = 0.2$ in our experiments.

As for semantic loss, a pre-trained semantic world model Seg [31] is used to extract semantic features $\mathcal{I}^{tgt}_{(0)}$. The goal is to align these features in a high-dimensional space during the training process. The alignment is supervised by minimizing

$$\mathcal{L}_{seg} = \sum_{n=0}^{N} \|Seg\left(\mathcal{I}^{tgt}_{(n)(0)}\right) - Seg\left(\Lambda^{(k)}_{(n)}(\mathcal{I}^{src}_{(n)(k)})\right)\|_1 \tag{16}$$

Total Loss. Four losses are incorporated in our experiment. Since it is quite challenging to balance the weights of these

TABLE I

QUANTITATIVE COMPARISON RESULTS OF OUR METHOD WITH THE BASELINES ON PUBLIC DATASETS. $*$ REPRESENTS THE RESULT FROM ORIGINAL PAPER AS NO AVAILABLE CODE. THE RESULTS RANKED FROM BEST TO WORST ARE HIGHLIGHTED AS FIRST, SECOND, AND THIRD.

	Image Size	Supervision	Dataset	*Abs.Rel.*↓	*Sq.Rel.*↓	*RMSE*↓	*RMSE log*↓	$\delta < 1.25$↑	$\delta < 1.25^2$↑	$\delta < 1.25^3$↑
R3D3 [12]	640 × 384	Semi	DDAD [16]	0.392	3.824	14.435	0.447	0.482	0.623	0.720
VFDepth [28]	640 × 384	Self		0.259	3.340	12.934	0.362	0.693	0.795	0.892
SurroundDepth [15]	640 × 384	Semi		0.273	3.540	12.651	0.372	0.790	0.883	0.930
M^2Depth* [14]	640 × 384	Semi		0.182	2.920	11.963	0.299	0.756	0.897	0.947
Ours	800 × 448	Semi		0.167	2.686	11.407	0.283	0.792	0.904	0.953
R3D3 [12]	768 × 448	Semi	nuScenes [33]	0.368	5.985	8.547	0.409	0.639	0.742	0.832
VFDepth [28]	640 × 352	Self		0.284	4.892	6.982	0.385	0.640	0.773	0.873
SurroundDepth [15]	640 × 352	Semi		0.309	5.232	7.106	0.342	0.672	0.741	0.869
M^2-Depth* [14]	640 × 352	Semi		0.259	4.599	6.898	0.332	0.734	0.871	0.928
Ours	800 × 448	Semi		0.197	2.624	6.094	0.297	0.789	0.903	0.946

four losses, we align the scales of all losses to the depth loss $\mathcal{L}_d$ and then apply the weights $\lambda_1 = \lambda_2 = 0.5$, $\lambda_3 = \lambda_4 = 3$. The total loss $\mathcal{L}_{all}$ is calculated by

$$\mathcal{L}_{all} = \lambda_1 \mathcal{L}_d + \lambda_2 \frac{|\mathcal{L}_d| \mathcal{L}_\nabla}{|\mathcal{L}_\nabla|} + \lambda_3 \frac{|\mathcal{L}_d| \mathcal{L}_{rep}}{|\mathcal{L}_{rep}|} + \lambda_4 \frac{|\mathcal{L}_d| \mathcal{L}_{seg}}{|\mathcal{L}_{seg}|}. \tag{17}$$

IV. EXPERIMENTS

In this section, we first introduce the experimental setup in Sec. IV-A, including datasets, evaluation metrics, baselines, and parameter settings, etc. Then in Sec. IV-B, we mainly compare the frameworks performance with baselines. Sec. IV-C shows ablation experiments of the proposed framework. Sec. IV-D shows GPU usage and inference time experiments of the proposed framework.

A. Environmental Setup

Baselines and Metrics. We compare our results with R3D3 [12], VFDepth [28], scale-aware SurroundDepth [15], and M^2-Depth [14]. It is pertinent to note that both SurroundDepth [15] and M^2-Depth [14] utilize the structure-from-motion (SfM) method to generate sparse depth for training supervision, contradicting their claim of being self-supervised in their papers. Therefore, in the comparative experiments of this paper, we categorize them as semi-supervised methods. Following prior works [34], [14], [15], the evaluation metrics we used are $Abs.Rel.$, $Sq.Rel.$, $RMSE$, $RMSE\,log$, and δ.

Datasets. We train and evaluate our framework on two public datasets, including DDAD [16] and nuScenes [33]. For the DDAD [16] dataset, we first crop the image from the top-left corner to a size of 1936×1084, and then resize it to 800×448. For the nuScenes [33] dataset, we directly resize the images to 800×448. To the best of our knowledge, the image sizes employed in our work are the largest, which helps improve the extraction of detailed features and their subsequent refinement.

The maximum evaluation depth D_{max} is set to 200 meters, averaged across all cameras in the DDAD [16] dataset, and 80 meters in the nuScenes [33] dataset, consistent with the baselines. For the nuScenes dataset [33], we select 300 scenes featuring aggressive driving scenarios to avoid overfitting and reduce the computational cost of experiments.

Training. We implement our approach using PyTorch and train the model with the Adam optimizer at a learning rate of 10^{-4}. The training runs for 100 epochs across all datasets, with the first 5 epochs dedicated to warm-up for the learning rate, which is then decayed using a cosine schedule. To ensure a fair comparison, we use a 34-layer ResNet [18] as the backbone. For semantic feature extraction, we utilize the frozen SAM encoder provided by MobileSAM [31] to reduce GPU usage. We use SegFormer [35] to generate the sky mask, removing interference from the sky in outdoor scenes. For the DDAD dataset [16], we also eliminate the vehicle occlusion areas using the corresponding mask data from [15]. Our experiments are conducted on 8 NVIDIA H100 80GB HBM3 GPUs, with 50 hours of training on the nuScenes [33] dataset and 16 hours on the DDAD [16] dataset. Additional details can be found in our released code.

B. Qualitative and quantitative analysis

Comparison with Baselines. We begin by presenting a qualitative comparison of different baselines and our method on the DDAD [16], as shown in Fig. 6. It is clear that our method yields clearer image details and sharper boundaries. And the visilization of *Abs.Rel.* demostrate the quality of depth estimation is also superior. The quantitative comparison on the DDAD [16] and nuScenes [33] datasets is shown in Table I.

Per-Camera Evaluation. We also do per-camera evaluation in Table II. M^2-Depth [14] estimate the pose by the front camera, so the metric *Abs.Sql* on front camera is the best. But our framework can achieve a error balance across surrounding cameras by global feature fusion.

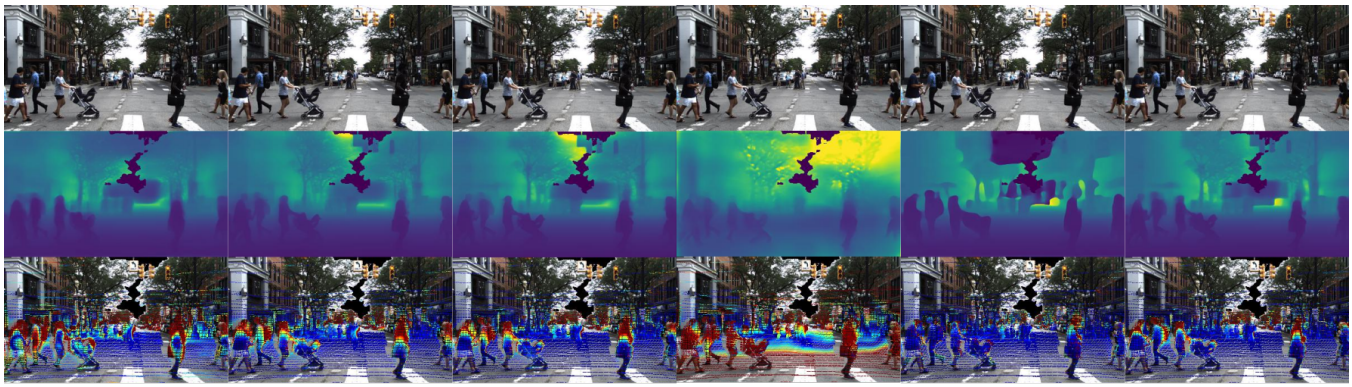
TABLE II

PER-CAMERA *Abs.Sql* EVALUATION ON DDAD [16].

Methods	*Abs.Sql*						
	CAM-01	CAM-05	CAM-06	CAM-07	CAM-08	CAM-09	Avg.
R3D3 [12]	0.375	0.428	0.417	0.388	0.381	0.363	0.392
VFDepth [28]	0.240	0.260	0.277	0.251	0.253	0.273	0.259
SurroundDepth [15]	0.263	0.279	0.258	0.254	0.292	0.291	0.273
M^2-Depth [14]	0.146	0.182	0.200	0.198	0.203	0.169	0.183
Ours	0.161	0.172	0.169	0.171	0.168	0.167	0.155

C. Ablation study

Ablation Study on Module Contributions. We systematically evaluate the impact of various modules and losses on the final performance, with a particular focus on the spatial-temporal attention (ST) mechanism, SAM feature integration (SAM), depth-enhanced motion estimation networks, the sparse depth loss $\mathcal{L}_d$, and curvature loss $\mathcal{L}_\nabla$. The quantitative effects of different module and loss combinations are presented in Table III and Table IV, while qualitative results are illustrated in Fig. 7. Fig. 8 shows the convergence of our different modules and their final metrics



(a) ① ⑦ (b) ② ⑦ (c) ③ ⑦ (d) ④ ⑤ (e) ④ ⑥ (f) ④ ⑦

Fig. 7. Visualizations of the ablation experiments on metric depth estimation results for various modules, with parameter configurations and combinations corresponding to the IDs provided in Table III and Table IV.

on the validation set. The results demonstrate that the key components of the framework significantly enhance overall performance, both in qualitative and quantitative evaluations. It is worth noting that with the introduction of our curvature loss $\mathcal{L}_{\nabla}$, although there is no significant improvement in the metrics in Table IV, there is a substantial enhancement in the smoothness and semantic consistency of the depth map shown in Fig. 7. This is because the depth loss is inherently sparse, constraining only a minimal number of regions in the depth predictions.

TABLE III

QUALITATIVE COMPARISON RESULTS ON DDAD (EVALUATE DEPTH: 80 M) OF THE THREE MODULES: SPATIAL-TEMPORAL ATTENTION (ST), SAM FEATURE (SAM), DEPTH-ENHANCED MOTION ESTIMATION (DM).

ID	ST	SAM	DM	*Abs.Rel.*↓	*Sq.Rel.*↓	*RMSE*↓	*RMSE log*↓	$\delta < 1.25^3$↑
①	✗	✗	✗	0.223	5.726	6.897	0.329	0.923
②	✓	✗	✗	0.206	4.513	6.708	0.310	0.959
③	✓	✓	✗	0.204	4.573	6.494	0.298	0.961
④	✓	✓	✓	0.155	3.307	5.934	0.276	0.968

TABLE IV

QUALITATIVE COMPARISON RESULTS ON DDAD (EVALUATE DEPTH: 80 M) OF THE TWO LOSSES: DEPTH LOSS $\mathcal{L}_d$ AND CURVATURE LOSS $\mathcal{L}_{\nabla}$.

ID	$\mathcal{L}_d$	$\mathcal{L}_{\nabla}$	*Abs.Rel.*↓	*Sq.Rel.*↓	*RMSE*↓	*RMSE log*↓	$\delta < 1.25^3$↑
⑤	✗	✓	3.319	165.719	36.792	1.366	0.323
⑥	✓	✗	0.163	3.310	6.337	0.279	0.967
⑦	✓	✓	0.155	3.307	5.934	0.276	0.968

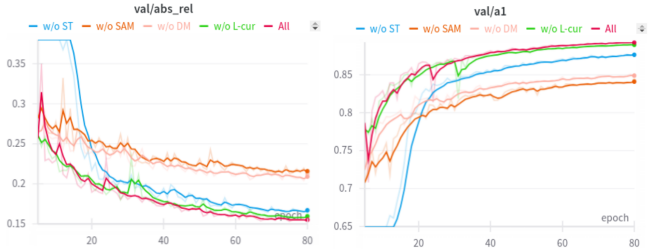


Fig. 8. Visualization of training iterations on DDAD [16] dataset. It can be observed that as we progressively incorporate the proposed modules, the overall model's metric accuracy and convergence speed improve steadily. val/abs_rel and $val/a1$ represent $Abs.Rel$ and $\delta < 1.25$, respectively.

Attention Map Visualizations. We present visualizations of attention maps for both spatial surrounding camera attention and temporal frame attention in Fig. 9. We observe that the spatial surrounding camera attention predominantly focuses on overlapping regions (highlighted in orange), effectively capturing matching relationships between adjacent surrounding cameras. Meanwhile, the temporal frame attention emphasizes regions with more distinctive features (highlighted in blue), which aids in robust frame-to-frame matching. These visualizations demonstrate the effectiveness of our attention modules in capturing spatial and temporal relationships.

(a) Source Frame (b) Nearby Frame

(d) Attention Map (d) Target Frame

Fig. 9. Visualization of attention map on DDAD [16] dataset. It can be observed that the attention module highlights certain key areas, which contribute to the estimation of scale information and the pose transformation estimation between adjacent frames.

D. GPU Usage and Time Consumption

We compare the GPU usage and time consumption of our models with other baselines, as shown in Table V. Note that we use the semantic world model [31] to obtain *online* semantic features, which we have taken into consideration. It is evident that our model demonstrates improvements in time efficiency. Although M^2-Depth [14] claims to also utilize semantic features, we cannot verify whether they use the offline semantic world model to reduce the GPU usage due to the lack of source code.

TABLE V

COMPARISON OF GPU USAGE AND INFERENCE TIME OF ONE FRAME. BEST RESULTS ARE UNDERLINED.

Models	GPU (Mb)	Inference Time (milliseconds)
SurroundDepth [15]	8732	248
M^2-Depth* [14]	<u>5546</u>	295
Ours	9137	<u>232</u>

V. Conclusion

In this paper, we propose **Semi-SMD**, a unified approach for metric depth prediction and pose estimation through spatial-temporal-semantic information fusion for surrounding cameras. By designing a unified transformer architecture, we effectively integrate these features, improving computational efficiency and reducing boundary ambiguity. We also introduce a joint pose estimation network for surrounding cameras that combines depth predictions with camera extrinsic parameters to achieve accurate scale estimation. Additionally, we propose a curvature loss function, where guidance from the depth estimation world model significantly accelerates convergence and improves depth prediction accuracy. Experimental results on two widely used datasets demonstrate the superiority of our method, showcasing its broad applicability in autonomous driving systems with surrounding cameras.

References

[1] J. Shi, J. Chen, Y. Wang, L. Sun, C. Liu, W. Xiong, and T. Wo, "Motion Forecasting for Autonomous Vehicles: A Survey," *arXiv preprint arXiv:2502.08664*, 2025.

[2] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang *et al.*, "Planning-oriented Autonomous Driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 853–17 862.

[3] S. Teng, X. Hu, P. Deng, B. Li, Y. Li, Y. Ai, D. Yang, L. Li, Z. Xuanyuan, F. Zhu *et al.*, "Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 6, pp. 3692–3711, 2023.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are We Ready for Autonomous Driving? The Kitti Vision Benchmark Suite," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[5] N. Karnchanachari, D. Geromichalos, K. S. Tan, N. Li, C. Eriksen, S. Yaghoubi, N. Mehdipour, G. Bernasconi, W. K. Fong, Y. Guo *et al.*, "Towards Learning-based Planning: The nuPlan Benchmark for Real-world Autonomous Driving," *arXiv preprint arXiv:2403.04133*, 2024.

[6] D. Zhang, G. Wang, R. Zhu, J. Zhao, X. Chen, S. Zhang, J. Gong, Q. Zhou, W. Zhang, N. Wang *et al.*, "SparseAD: Sparse Query-centric Paradigm for Efficient End-to-End Autonomous Driving," *arXiv preprint arXiv:2404.06892*, 2024.

[7] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot Transfer by Combining Relative and Metric Depth," *arXiv preprint arXiv:2302.12288*, 2023.

[8] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, "Depth Anything: Unleashing the Power of Large-scale Unlabeled Data," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 371–10 381.

[9] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth Anything V2," *arXiv preprint arXiv:2406.09414*, 2024.

[10] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into Self-supervised Monocular Depth Estimation," in *IEEE International Conference on Computer Vision*, 2019, pp. 3828–3838.

[11] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised Monocular Depth Estimation with Left-right Consistency," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 270–279.

[12] A. Schmied, T. Fischer, M. Danelljan, M. Pollefeys, and F. Yu, "R3D3: Dense 3D Reconstruction of Dynamic Scenes from Multiple Cameras," in *IEEE International Conference on Computer Vision*, 2023, pp. 3216–3226.

[13] F. Wimbauer, N. Yang, L. Von Stumberg, N. Zeller, and D. Cremers, "MonoRec: Semi-supervised Dense Reconstruction in Dynamic Environments from a Single Moving Camera," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6112–6122.

[14] Y. Zou, Y. Ding, X. Qiu, H. Wang, and H. Zhang, "M2Depth: Self-supervised Two-Frame Multi-camera Metric Depth Estimation," in *European Conference on Computer Vision*, 2024, pp. 269–285.

[15] Y. Wei, L. Zhao, W. Zheng, Z. Zhu, Y. Rao, G. Huang, J. Lu, and J. Zhou, "Surrounddepth: Entangling Surrounding Views for Self-supervised Multi-camera Depth Estimation," in *Conference on Robot Learning*, 2023, pp. 539–549.

[16] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3D Packing for Self-supervised Monocular Depth Estimation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2485–2494.

[17] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[19] C. Zhang, D. Han, Y. Qiao, J. U. Kim, S.-H. Bae, S. Lee, and C. S. Hong, "Faster Segment Anything: Towards Lightweight SAM for Mobile Applications," *arXiv preprint arXiv:2306.14289*, 2023.

[20] I. Alhashim and P. Wonka, "High Quality Monocular Depth Estimation via Transfer Learning," *arXiv preprint arXiv:1812.11941*, 2018.

[21] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision Transformers for Dense Prediction," in *IEEE International Conference on Computer Vision*, 2021, pp. 12 179–12 188.

[22] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From Big to Small: Multi-scale Local Planar Guidance for Monocular Depth Estimation," *arXiv preprint arXiv:1907.10326*, 2019.

[23] Y. Shi, H. Cai, A. Ansari, and F. Porikli, "EGA-Depth: Efficient Guided Attention for Self-supervised Multi-camera Depth Estimation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 119–129.

[24] J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The Temporal Opportunist: Self-supervised Multi-frame Monocular Depth," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1164–1174.

[25] R. Li, D. Gong, W. Yin, H. Chen, Y. Zhu, K. Wang, X. Chen, J. Sun, and Y. Zhang, "Learning to Fuse Monocular and Multi-view Cues for Multi-frame Depth Estimation in Dynamic Scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 539–21 548.

[26] X. Wang, Z. Zhu, G. Huang, X. Chi, Y. Ye, Z. Chen, and X. Wang, "Crafting Monocular Cues and Velocity Guidance for Self-supervised Multi-frame Depth Learning," in *AAAI Conference on Artificial Intelligence*, vol. 37, no. 3, 2023, pp. 2689–2697.

[27] N. Yang, R. Wang, J. Stuckler, and D. Cremers, "Deep Virtual Stereo Odometry: Leveraging Deep Depth Prediction for Monocular Direct Sparse Odometry," in *European Conference on Computer Vision*, 2018, pp. 817–833.

[28] J.-H. Kim, J. Hur, T. P. Nguyen, and S.-G. Jeong, "Self-supervised Surround-view Depth Estimation with Volumetric Feature Fusion," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4032–4045, 2022.

[29] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment Anything," in *IEEE International Conference on Computer Vision*, 2023, pp. 4015–4026.

[30] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson *et al.*, "Sam 2: Segment Anything in Images and Videos," *arXiv preprint arXiv:2408.00714*, 2024.

[31] C. Zhang, D. Han, S. Zheng, J. Choi, T.-H. Kim, and C. S. Hong, "MobileSAMv2: Faster Segment Anything to Everything," *arXiv preprint arXiv:2312.09579*, 2023.

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Sssessment: from Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[33] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A Multimodal Dataset for Autonomous Driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 621–11 631.

[34] V. Guizilini, I. Vasiljevic, R. Ambrus, G. Shakhnarovich, and A. Gaidon, "Full Surround Monodepth from Multiple Cameras," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5397–5404, 2022.

[35] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.