

FlowDepth: Decoupling Optical Flow for Self-Supervised Monocular Depth Estimation

Yiyang Sun, Zhiyuan Xu, Xiaonian Wang and Jing Yao[†]

Abstract—Self-supervised multi-frame methods have currently achieved promising results in depth estimation. However, these methods often suffer from mismatch problems due to the moving objects, which break the static assumption. Additionally, unfairness can occur when calculating photometric errors in high-freq or low-texture regions of the images. To address these issues, existing approaches use additional semantic priori black-box networks to separate moving objects and improve the model only at the loss level. Therefore, we propose FlowDepth, where a Dynamic Motion Flow Module (DMFM) decouples the optical flow by a mechanism-based approach and warps the dynamic regions thus solving the mismatch problem. For the unfairness of photometric errors caused by high-freq and low-texture regions, we use Depth-Cue-Aware Blur (DCABlur) and Cost-Volume sparsity loss respectively at the input and the loss level to solve the problem. Experimental results on the KITTI and Cityscapes datasets show that our method outperforms the state-of-the-art methods.

Index Terms—Depth Estimation, Self-supervised, Motion Flow, Cost Volume.

I. INTRODUCTION

Estimating the depth of each pixel in an image is an essential task for obtaining 3D scene geometry information, which provides necessary geometric cues in robot navigation and self-driving. However, depth labels for supervised training are always difficult to obtain, and even using LiDAR sensors to capture depth ground truth can only provide sparse points compared to the density of pixels. Since spatiotemporal continuous images are commonly available in reality, the self-supervised way [1]–[4], which minimizes the photometric error between the original and synthetic images, has become more popular in recent years.

Early convolutional self-supervised depth estimation methods [5], [6] only use adjacent frames to compute reprojection loss in training, but they do not utilize geometric constraint information of the temporal sequence frames in inference, which limits the performance of depth estimation. Therefore, recent methods [2], [7] use multi-frame to construct a similar cost-volume in stereo-match tasks during inference to improve estimation accuracy.

However, when using reprojection loss to calculate the photometric error, there are two problems that break the photometric consistency assumption. In **low-texture regions** such as road surfaces or building wall surfaces, a low photometric error will be calculated even if the matched pixel has a large deviation from gt [8]. In **high-freq texture**

regions such as leaves or the edges of pavement lanes, a small deviation from gt may result in a large photometric error [9]. In addition, both the reprojection loss and the cost-volume-based approach suffer from the **dynamic objects** which can violate the static environment assumption and lead to the mismatch problem in the dynamic regions. Recent works usually introduce auxiliary semantic priors to mask out dynamic objects to solve this problem. Nevertheless, the accuracy is directly influenced by the pre-trained semantic network.

Therefore, we propose FlowDepth in order to solve these problems and improve the depth estimation accuracy. First, we predict depth, camera motion, and optical flow prior, and decouple the optical flow into a static/rigid part (caused by ego camera motion) and a dynamic part (caused by dynamic object motion) through Dynamic Motion Flow Module (DMFM). Then the dynamic part is applied to the source frame and ‘moves’ the dynamic object to where it should be if it keeps the global position in the target frame. The new source frame and target frame can guarantee the static environment assumption. Then, we use the Depth-Cue-Aware Blur (DCABlur) module to only blur the high-freq regions or edges with dramatic color changes to increase the one-pixel perceptual field to make the photometric error fairer, while ensuring the depth cues in the image are not affected by blur. In the training phase, we propose cost-volume sparse loss to alleviate the unfairness of reprojection loss in low-texture regions. We summarize the contributions of our paper as follows:

- We propose a novel Dynamic Motion Flow Module (DMFM) that decouples the optical flow by a mechanism-based approach and then warps the dynamic objects in the source frame to solve the mismatch problem with no additional annotations.
- We design a Depth-Cue-Aware Blur (DCABlur) module and a cost-volume sparse loss to mitigate the reprojection mismatch problem.
- We experimentally validate the performance of our method on KITTI, Cityscapes, and our own VECAN datasets. The results show that our model outperforms prior methods.

II. RELATED WORKS

A. Single-frame monocular depth

[1] first proposes a self-supervised depth estimation framework that simultaneously predicts depth and ego-motion. Based on [1], Monodepth2 [6] introduces per-pixel

[†]indicates the corresponding author.

Y.Sun, Z.Xu, X.Wang, and J.Yao are with the College of Electronical and Information Engineering, Tongji University, China. E-mail: {2130734, zhiyuan, dawnyear, yaojing}@tongji.edu.cn

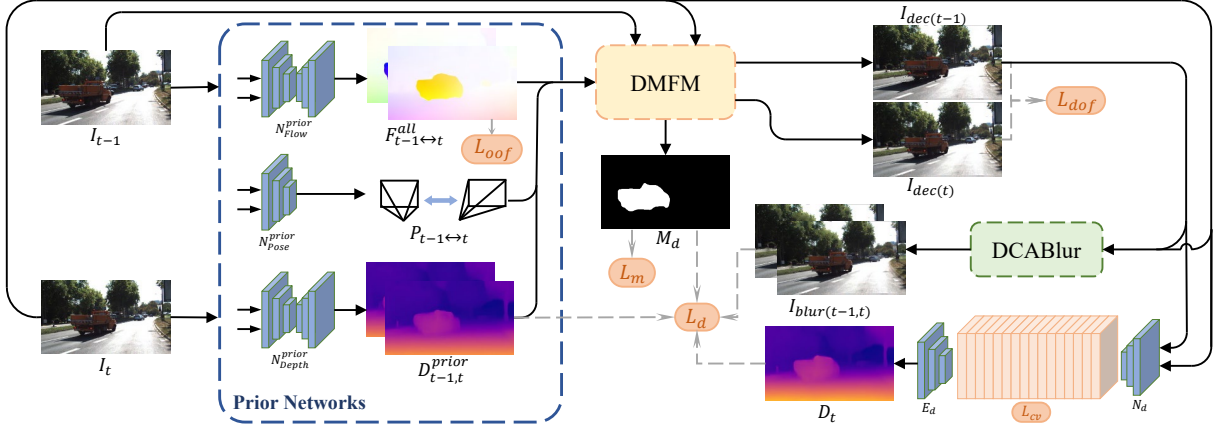


Fig. 1. Architecture of our FlowDepth. The images I_{t-1} and I_t are first passed through the prior networks to get depth, camera motion, and optical flow prior. Then DMFM decouples the moving objects with these prior. The new images are fed into a multi-frame depth estimation network constrained by the cost-volume sparse loss. Finally, it generates the depth estimation results. Before calculating the multi-frame depth reprojection loss, the images will go through the DCABlur module for blurring to mitigate high-freq texture problems.

minimization reprojection loss and auto-masking to address the dynamic objects problem. In order to meet the real-time requirements, PackNet [10] utilizes 3D convolution to explore more effective model. Lite-Mono [11] further designs a lightweight model based on the transformer and CNN. RM-depth [12] tackles the issue from the perspective of model volume, employing RMU to substantially reduce the model's parameters. The idea of all these methods is to project adjacent frames during training to satisfy reprojection photometric consistency [13]. However, the reprojection loss will meet the mismatch problem caused by dynamic objects or the high-freq/low-texture region. In contrast, our proposed DMFM, DCABlur module, and CV Loss can solve the mismatch problem, thereby improving the accuracy of depth estimation.

B. Multi-frame monocular depth

Due to the aforementioned methods using only a single image during inference, it fails to consider temporal constraints, thereby limiting the model's performance. Therefore, a direct approach is to incorporate multi-frame inputs by utilizing recurrent networks both during the training and inference stages. [14] employed ConvLSTM to predict the depth sequentially. Compared to recurrent networks, MonoRec [7] and Manydepth [2] introduce the cost-volume in stereo-match, enabling geometric constraints during inference and greatly saving the inference time. However, cost-volume-based methods still rely on static environment assumptions. In contrast, our DMFM module fundamentally 'staticizes' all dynamic objects at the input level, thereby enhancing the depth estimation in dynamic object regions.

C. Depth estimation with auxiliary tasks

Prior works address the issue of dynamic objects by employing object motion prediction networks or additional semantic prior. Since both the depth estimation and the flow estimation adhere to the scene geometric consistency constraint, [15]–[17] simultaneously predict depth and optical flow using a multitask model to enhance the results.

Nevertheless, multitask models are usually hard to converge during the training. [8], [18]–[21] try to predict the dynamic objects motion using black-box networks, followed by pixel-wise warping of the dynamic object. DynamicDepth [3] directly uses a pre-trained segmentation network to separate objects and performs warping with depth prior. TriDepth [4] even directly uses the semantic labels to get object edges and alleviate the ubiquitous edge-fattening issue by using triplet loss. However, these approaches have drawbacks such as the low interpretability of the black-box object motion network, and the need for additional segmentation networks that should be trained with expensive manual annotations. In contrast, our proposed DMFM decouples the optical flow using a mechanism-based approach, which is consistent with the scene geometry. So the whole training process requires only image data without any manual annotations.

III. METHODOLOGY

A. Overall Structure

As shown in Fig.1, We first use prior networks to predict depth, camera motion, and optical flow prior, which are sent to the Dynamic Motion Flow Module (Sec.III-B) to decouple the moving objects in I_{t-1} in order to solve the mismatch problem caused by dynamic objects. Then the decoupled frame $I_{dec(t-1)}$ and I_t are Depth-Cue-Aware Blurred (Sec.III-C) in the high-freq regions in images to mitigate the unfairness of reprojection loss. Finally, they are sent to a cost-volume-based depth estimation network to predict the depth. To solve the depth uncertainty caused by unfair photometric errors in the low-texture regions, we introduce the cost-volume sparse loss (Sec.III-D).

B. Dynamic Motion Flow Module (DMFM)

Preliminaries. When there is relative motion between the camera and the scene, the optical features of the scene object surface are projected onto the image plane, resulting in the optical flow. Given two continuous frames I_{t-1} and I_t , the pixel motion (denoted as flow F^{all}) can be decomposed into two components: one is the static/rigid flow F^s representing

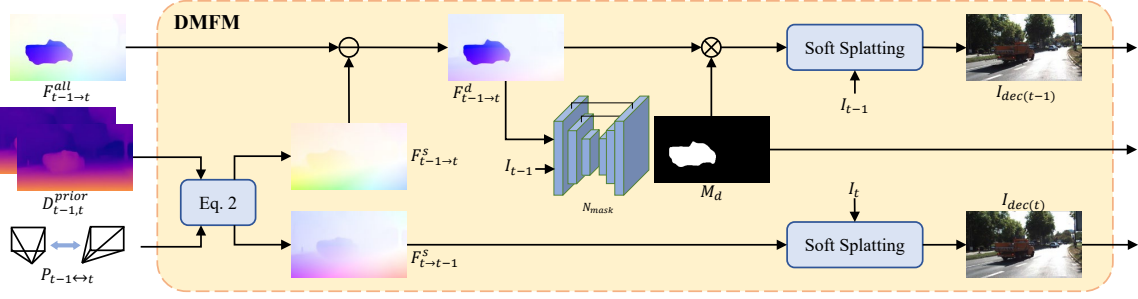


Fig. 2. The detailed structure of DMFM. Firstly, the static optical flow F^s is obtained using depth and pose priors. Then, the overall optical flow $F_{t-1 \rightarrow t}^{all}$ is decoupled by $F_{t-1 \rightarrow t}^s$ to obtain the dynamic optical flow $F_{t-1 \rightarrow t}^d$. By learning a mask network, I_{t-1} is warped according to the $F_{t-1 \rightarrow t}^d$ in dynamic area to get $I_{dec(t-1)}$. Meanwhile, by directly applying the $F_{t \rightarrow t-1}^s$ to I_t , we can also obtain $I_{dec(t)}$. Theoretically, $I_{dec(t-1)}$ and $I_{dec(t)}$ are the same.

the background variations in the scene caused by the ego-motion, and the other is the dynamic flow F^d induced by the self-motion of moving objects in the scene. The F^s adheres to the geometric consistency of the scene and is related to the scene depth and camera motion. When a 2D point $\mathbf{x} \in \mathbb{R}^2$ in the image is given, it can be reprojected back to $\mathbf{X} \in \mathbb{R}^3$ in the 3D world coordinate system:

$$\mathbf{X}^T = d(\mathbf{x})K^{-1}[\mathbf{x}; 1]^T, \quad (1)$$

where $K \in \mathbb{R}^{3 \times 3}$ is the camera intrinsic parameter, d represents the depth in 3D world corresponding to each pixel. Based on Eq. 1, we can calculate the difference between \mathbf{x}_{t-1} and \mathbf{x}_t , which represents the static flow F^s :

$$F_{t-1 \rightarrow t}^s = \frac{1}{d_t} K P (d_{t-1}(\mathbf{x}_{t-1}) K^{-1} [\mathbf{x}_{t-1}; 1]^T) - [\mathbf{x}_{t-1}; 1]^T, \quad (2)$$

where the first term calculates as \mathbf{x}_t , $P = [R; T] \in \mathbb{R}^{3 \times 4}$ is the relative pose change of the camera, and d_t is the normalization coefficient from 3D to 2D, which is commonly considered as a constant. In Eq. 2, once we have d_{t-1} and P , we can obtain $F_{t-1 \rightarrow t}^s$, which represents the static flow from $t-1$ to t .

DMFM implementation. The key idea of DMFM is to relocate the moving objects in the source frame (I_{t-1}) to where they should be if the objects are stationary in the target frame (I_t). In other words, it can be seen as fixing the dynamic objects in $t-1$ at their world coordinates in t . We denote the new frame after warping the objects as I_{dec} .

As shown in Fig.2, we first obtain depth, pose, and optical flow (all) priors through the teacher networks. As mentioned in the preliminaries, we can decompose the overall flow between two frames into dynamic and static flow components. Therefore, we have two approaches to obtain I_{dec} from I_{t-1} and I_t respectively. The first one is to obtain the dynamic motion flow $F_{t-1 \rightarrow t}^d$ by

$$F_{t-1 \rightarrow t}^d = F_{t-1 \rightarrow t}^{all} - F_{t-1 \rightarrow t}^s. \quad (3)$$

$F_{t-1 \rightarrow t}^{all}$ is the optical flow prior obtained from RAFT [22], and $F_{t-1 \rightarrow t}^s$ is computed using the depth and pose prior in Eq.2. Then $I_{dec(t-1)}$ is obtained by applying forward warping on I_{t-1} using $F_{t-1 \rightarrow t}^d$. For the second approach, we only compute $F_{t \rightarrow t-1}^s$ using Eq.2 and forward warp I_t by $F_{t \rightarrow t-1}^s$ to get $I_{dec(t)}$. In theory, $I_{dec(t-1)}$ and $I_{dec(t)}$

should be almost the same. Finally, these two approaches are simultaneously employed, and $I_{dec(t-1)}$ is selected as the lookup frame for multi-frame depth estimation.

Why both approaches are used? Although $I_{dec(t-1)}$ and $I_{dec(t)}$ should ideally be consistent, due to the imperfect estimation from the teacher networks, they may be slightly different. Therefore, we introduce an additional self-supervised loss called L_{dof} , which will be described in detail in the Appendix. The introduction of this loss not only improves the performance of the teacher network but also enhances the quality of I_{dec} , thus improving the performance of multi-frame depth estimation. Additionally, when the dynamic objects in I_{t-1} are warped, part of pixels at the original location of the objects will have no value due to the occlusion. These pixels can be filled in by the background information, which typically does not have foreground occlusions in $I_{dec(t)}$, effectively resolving the occlusion issue.

Why don't we use $I_{dec(t)}$ as the lookup frame? This is because $I_{dec(t)}$ is obtained by warping I_t with $F_{t \rightarrow t-1}^s$, which heavily relies on the depth and pose priors of I_t and totally loses the information of I_{t-1} . So the inaccurate estimation results from the teacher network will limit the performance of the multi-frame network.

Then, to achieve better warp results, we adopt softmax splatting [23] for forward warping. This approach effectively resolves the issue of multi-to-one mapping in forward warping and allows for gradient computation.

Finally, since the inaccurate prior depth estimate makes it difficult to subtract $F_{t-1 \rightarrow t}^{all}$ and $F_{t-1 \rightarrow t}^s$ to equal zero in the background region, we use a U-Net, denoted as N_{mask} , to obtain a valid dynamic motion flow region:

$$M_d = [\theta(N_{mask}(F_{t-1 \rightarrow t}^d, I_{t-1})) > 0.6], \quad (4)$$

where θ is the Sigmoid function and $[\cdot]$ is the Iverson bracket. $M_d = 1$ means that there is a moving object and we only do soft splatting here.

C. Depth-Cue-Aware Blur (DCABlur)

The reprojection loss used in self-supervised depth estimation calculates photometric error to indirectly constrain the depth information. Consequently, it can lead to a large loss for small depth errors due to drastic color changes in high-frequency regions, or a small loss for large depth errors due to the same color in low-texture regions. These

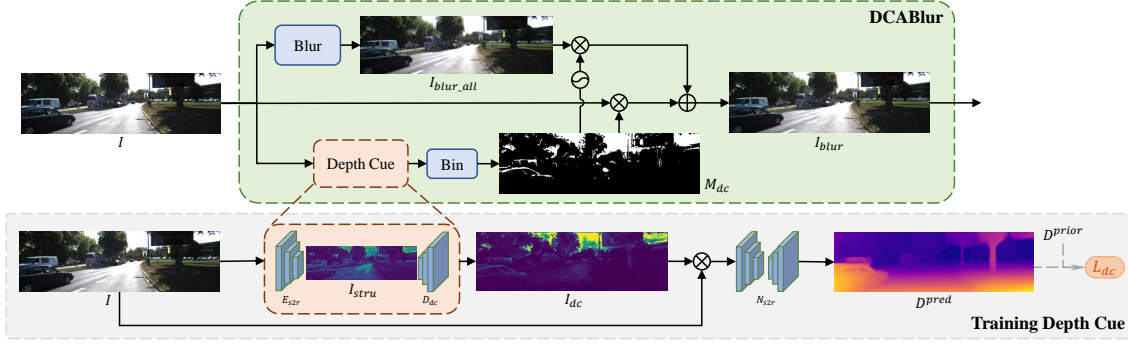


Fig. 3. Illustration of DCABlur. The Depth Cue is pre-trained as shown in the grey area.

situations result in unfairness in calculating photometric errors. Therefore, the concept of auto-blur [9] is to compute a frequency map by calculating the color differences and then the regions with frequencies above a threshold are blurred.

We propose the DCABlur module based on auto-blur as shown in Fig.3. The high-freq edges obtained by auto-blur include both texture edges and depth edges. Texture edges often correspond to small depth variations and are suitable for blurring methods. On the other hand, depth edges typically have significant depth changes, and blurring them would reduce the overall photometric error, resulting in a worse distinction between foreground and background. Therefore, our DCABlur aims to identify depth edges in images and only applies blurring to texture edges.

Inspired by [24], we found that depth cues are inherent semantic properties of objects, which do not vary with the style and texture of the object. Whereas a scene only consists of style information and structure information, we infer that depth cues are only related to structure information. So we pre-train a network to extract depth cues useful for depth estimation in images as shown in Training Depth Cue of Fig.3. The network is primarily an encoder-decoder model, where the structure encoder (E_{s2r}) uses the pre-trained style transfer encoder in [24] and the depth cue decoder (D_{dc}) is randomly initialized. The depth estimation network (N_{s2r}) also uses the pre-trained one in [24] to output a depth map for supervised learning. In training, we fix the structure encoder and the depth estimation network and only train the depth cue decoder to find the depth cue. D^{prior} is directly used as the ground truth. Because we are not looking for a particularly good depth output from this network, we just need the decoder to be able to learn the depth cues that the network considers helpful for depth estimation. The loss function is:

$$L_{dc} = \frac{1}{HW} \|N_{s2r}(D_{dc}(E_{s2r}(I)) \otimes I) - D^{prior}\|_1 + \frac{\sigma}{HW} \|D_{dc}(E_{s2r}(I))\|_1, \quad (5)$$

where the first term is directly supervised using an L1 loss and the second term is used to enforce the sparsity of the depth cue mask. H , W are the height and width of input image, σ is a coefficient which controls the degree of sparsity.

Finally, the trained depth cue network ($E_{s2r}+D_{dc}$) is

directly integrated into the DCABlur module and then the blur operation is applied to the image with the depth cue mask M_{dc} .

D. Loss Functions

When training our FlowDepth framework, we introduce a new cost-volume loss function to address the unfair photometric error in low-texture regions. Additionally, since we propose several modules, corresponding loss functions are designed for these modules as well.

Cost-Volume Sparse Loss. For more information about cost volume please refer to [2]. In short, each channel of the cost volume represents equally spaced candidate depths. Therefore, for each pixel in the image, we aim to have the minimum matching cost at the true depth. However, as shown in Fig.4, due to the low-texture regions, pixels in this area may get multiple small matching costs in the cost volume. To address this issue, we introduce probabilities as constraints, where smaller matching costs correspond to higher probability values. The depth estimation for each pixel is approximated as a classification task, and its corresponding entropy can be computed as:

$$\begin{aligned} En(d) &= - \sum_{i=1}^n P(d = d_i) \log P(d = d_i) \\ &\leq - \sum_{i=1}^k \frac{1}{k} \log\left(\frac{1}{k}\right) = \log k, \end{aligned} \quad (6)$$

where n is the number of candidate depths, and the entropy is maximized when the random variable follows a uniform distribution. In other words, we can control the number of confident indices with k by an entropy boundary loss

$$L_{entropy} = \frac{1}{HW} \sum_{\mathbf{x} \in I} \max(0, En_{\mathbf{x}} - \log k). \quad (7)$$

Since the number of candidate depths is usually large ($n = 90$) and the desired number of confident indices is small ($k = 3$), we apply $L_{1/2}$ sparsity loss in order to encourage more sparse constraints on the probabilities:

$$L_{sparsity} = 2\overline{P(d)} \sum_{i=n}^n \sqrt{1 + \frac{P(d = d_i)}{P(d)}}. \quad (8)$$

The final cost-volume loss is

$$L_{cv} = L_{entropy} + L_{sparsity}. \quad (9)$$

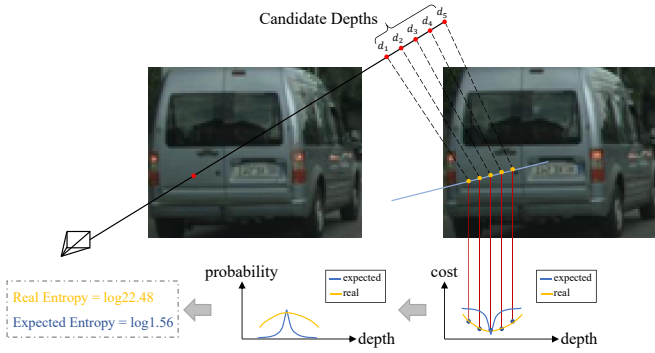


Fig. 4. The depth uncertainty caused by low-texture regions in cost volume. When the pixels in the low-texture region are projected to the matching map according to the candidate depths, several extremely close losses are obtained. Ideally, the probability of the feature cost in the candidate depth domain should show an unimodal distribution, but the real entropy is much larger than the expected one.

Besides, we use and design several other self-supervised loss functions. In brief, **Depth Loss** L_d is to train the final multi-frame depth estimation. **Origin Optical Flow Loss** L_{oof} is to constrain the outputs of N_{Flow}^{prior} . **Decouple Optical Flow Loss** L_{dof} is to get a better quality decoupled lookup frame. **Mask Loss** L_m is used to train U_{mask} .

Finally, all losses are combined to train FlowDepth as:

$$L_m = \lambda_{cv}L_{cv} + \lambda_dL_d + \lambda_{oof}L_{oof} + \lambda_{dof}L_{dof} + \lambda_mL_m. \quad (10)$$

where $\lambda_{cv/d/oof/dof/m}$ are learnable parameters and are used to balance the loss items.

IV. EXPERIMENTS

A. Implementation Details

Network Architecture. The overall architecture is depicted in Fig.1. The depth and camera motion prior networks, and multi-frame depth estimation network are designed following Monodepth2 [6], with ResNet18 backbones pre-trained on ImageNet. For the optical flow prior network, we employ RAFT-small [22], which can be self-supervised trained and provide excellent performance for ensuring the effective warping of dynamic regions. In DFMM, the N_{mask} utilizes a U-Net architecture with three layers and a sigmoid activation function. In the DCABlur module, the E_{s2r} and N_{s2r} inside Depth Cue are the same as [24], while D_{dc} consists of three upsampling layers. **It is important to emphasize that** all parts of our model only use RGB images to train because both depth estimation and optical flow estimation tasks adhere to geometric consistency thus allowing for self-supervised training.

Training. Before we train the FlowDepth, we will pre-train a depth cue model as described before. We fix E_{s2r} and N_{s2r} and use the output of the prior depth network to supervise the training of D_{dc} for 40 epochs with an Adam optimizer. The initial learning rate is set to $2e-4$ and will be decreased by 50% every 10 epochs. The batch size is set to 8. Then, just like other methods, the entire training process of FlowDepth is divided into two stages. The first stage is to train teacher networks. As most multi-frame methods do, we use frames $\{I_{t-1}, I_t, I_{t+1}\}$ for training

as well as frames $\{I_{t-1}, I_t\}$ for testing. This stage trains both the prior networks and the multi-frame network for 2 epochs with an Adam optimizer. We set the batch size to 8, and the learning rate to $1e-5$. The second stage is to fix all prior networks and N_{mask} , and only train the multi-frame network with the learning rate of $1e-6$ for another 8 epochs. The initial value of the learnable parameters are set as $\{\lambda_{cv}, \lambda_d, \lambda_{oof}, \lambda_{dof}, \lambda_m\} = \{0.2, 1, 1, 10, 1\}$.

Dataset. To keep consistency with previous work and ensure fairness, we conducted experiments on the KITTI dataset [27]. Furthermore, as the KITTI dataset contains a small number of dynamic objects, we also report the performance on the more challenging Cityscapes dataset [28], which contains more moving objects. To further validate the transferability of our model, we also do experiments on our own VECAN dataset.

B. Results and Analysis

We compare our FlowDepth with prior state-of-the-art methods. The final depth map is capped to 80m and is normalized using median scaling [29].

KITTI results. The results on the KITTI dataset by the testing split of [30] are shown in the upper half of TABLE I. As previous works do, we rank all methods based on Frames and AbsRel. Our results demonstrate that FlowDepth surpasses the performance of other comparative methods. In comparison to the baseline, FlowDepth relatively outperforms ManyDepth by 5.1% in AbsRel. Besides, when compared to the SOTA DynamicDepth, FlowDepth exhibits a relative improvement of 3.1% in AbsRel, substantiating its effectiveness in multi-frame monocular depth estimation. To ensure a fair comparison, we evaluate our model to TriDepth alone because it directly employs semantic ground truth as input. So we also leverage the semantic ground truth as M_d for the moving objects directly. The new experimental result is shown in the row of FlowDepth*. We can find that the model has a significant improvement compared to the original FlowDepth and also outperforms TriDepth* by 2.2% in AbsRel. This is because the original FlowDepth uses the network to learn the mask, which may filter out distant or slow-moving objects, resulting in these objects not being 'staticized' by the DMFM module. In contrast, using semantic information as a mask can address this issue. A visualization of every part's result in FlowDepth is shown in Fig.5. However, the number of dynamic objects in KITTI is too small, making the DMFM module proposed much less useful, so we focus on the Cityscapes dataset.

Cityscapes results. FlowDepth outperformed other SOTA methods in all metrics, as shown in the bottom half of TABLE I on Cityscapes. FlowDepth relatively outperforms ManyDepth by 14.0% and DynamicDepth by 5.8% in AbsRel. A comparison example is shown in Fig.6, showcasing a significant improvement in depth estimation. For dynamic objects, there are no more holes and unreasonable depth results, which verifies the DMFM module. In high-freq texture regions, our model not only produces smoother results compared to other methods, as evident in the road in the

TABLE I

COMPARISON ON THE KITTI AND CITYSCAPES DATASETS. THE BEST IN EACH METRIC IS IN **BOLD** AND THE SECOND BEST IS UNDERLINED.
 * MEANS THAT THE MODEL DIRECTLY USES THE SEMANTIC GROUND TRUTH AS AUXILIARY INFORMATION. ADDTL MEANS WHETHER THE MODEL
 REQUIRES ADDITIONAL MANUAL LABELING FOR SELF-SUPERVISED TRAINING ACROSS DIFFERENT DATASETS

	Method	Frames	ADDTL	W×H	AbsRel↓	SqRel↓	RMSE↓	RMSE log↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
KITTI	Zhou et al. [1]	1		416×128	0.208	1.768	6.856	0.283	0.678	0.885	0.957
	Struct2depth [18]	1	•	416×128	0.141	1.026	5.291	0.215	0.816	0.945	0.979
	CC [16]	1		832×256	0.140	1.070	5.326	0.217	0.826	0.941	0.975
	GLNet [17]	1		416×128	0.135	1.070	5.230	0.210	0.841	0.948	0.980
	Gordon et al. [21]	1	•	416×128	0.128	0.959	5.230	0.212	0.845	0.947	0.976
	Monodepth2 [6]	1		640×192	0.115	0.903	4.863	0.193	0.877	0.959	0.981
	Packnet-SFM [10]	1		640×192	0.111	0.785	4.601	0.189	0.878	0.960	0.982
	RM-Depth [12]	1		640×192	0.107	0.687	4.476	0.181	0.883	0.964	0.984
	Lite-Mono [11]	1		640×192	0.107	0.765	4.561	0.183	0.886	0.963	0.983
	FSRE-Depth [25]	1	•	640×192	0.105	0.722	4.547	0.182	0.886	0.964	0.984
	Patil et al. [14]	N		640×192	0.111	0.821	4.650	0.187	0.883	0.961	0.982
	ManyDepth [2]	2 (-1,0)		640×192	0.098	0.770	4.459	0.176	<u>0.900</u>	<u>0.965</u>	<u>0.983</u>
	DynamicDepth [3]	2 (-1,0)	•	640×192	0.096	0.720	4.458	<u>0.175</u>	0.897	<u>0.965</u>	0.984
	FlowDepth(ours)	2 (-1,0)		640×192	0.093	0.681	4.232	0.172	0.904	0.966	0.984
Cityscapes	TriDepth* [4]	2 (-1,0)	•	640×192	<u>0.093</u>	0.665	4.272	<u>0.172</u>	0.907	0.967	0.984
	FlowDepth*(ours)	2 (-1,0)	•	640×192	0.091	<u>0.675</u>	4.243	0.170	<u>0.905</u>	0.967	0.984
	Struct2depth [18]	1	•	416×128	0.145	1.737	7.280	0.205	0.813	0.942	0.976
	Monodepth2 [6]	1	•	416×128	0.129	1.569	6.876	0.187	0.849	0.957	0.983
	Gordon et al. [21]	1	•	416×128	0.127	1.330	6.960	0.195	0.830	0.947	0.981
	Li et al. [26]	1		416×128	0.119	1.290	6.980	0.190	0.846	0.952	0.982
	Lee et al. [20]	1	•	832×256	0.116	1.213	6.695	0.186	0.852	0.951	0.982
	InstaDM [19]	1	•	832×256	0.111	1.158	6.437	0.182	0.868	0.961	0.983
	ManyDepth [2]	2 (-1,0)		512×192	0.114	1.193	6.223	0.170	0.875	0.967	<u>0.989</u>
	DynamicDepth [3]	2 (-1,0)	•	512×192	<u>0.103</u>	<u>1.000</u>	<u>5.867</u>	<u>0.157</u>	<u>0.895</u>	<u>0.974</u>	0.991
	FlowDepth(ours)	2 (-1,0)		512×192	0.097	0.974	5.693	0.152	0.901	0.975	0.991

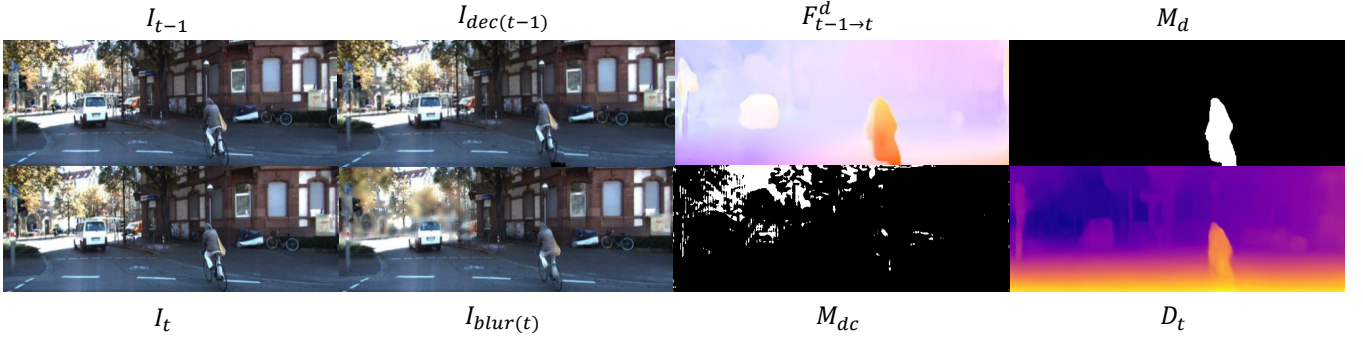


Fig. 5. Examples of outputs of each part on KITTI.

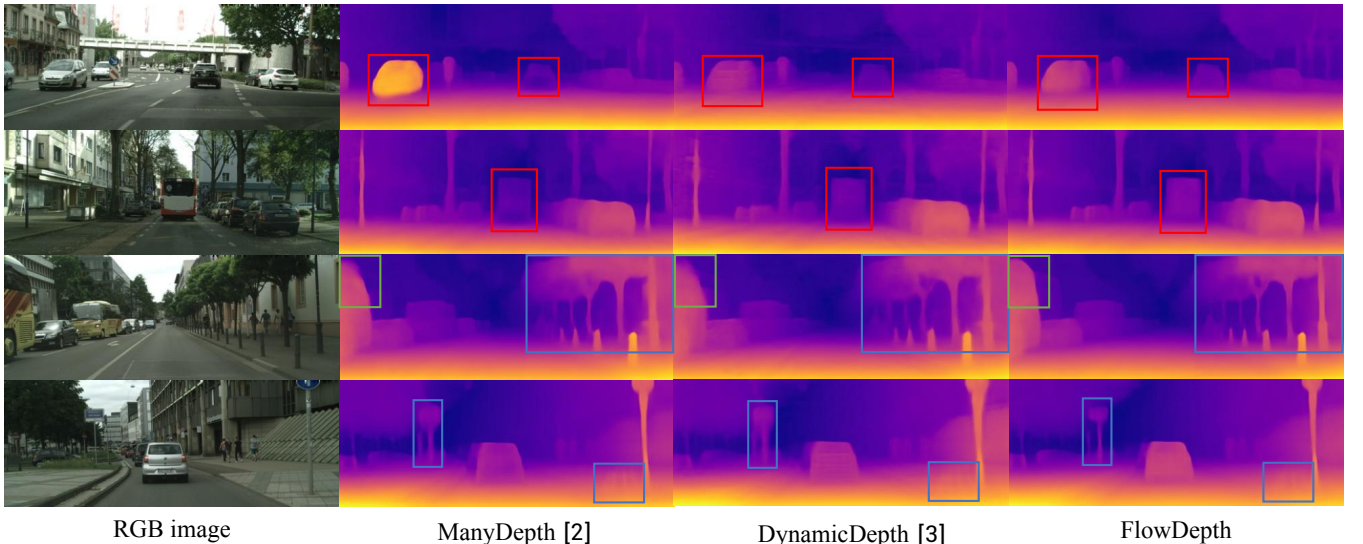
Fig. 6. Examples of depth estimations on Cityscapes. Different color boxes represent: **Dynamic Objects**, **high-freq texture**, and **low-texture**

TABLE II
ABLATION STUDY ON THE CITYSCAPES DATASET.

DMFM		DCABlur	CVloss	AbsRel↓	SqRel↓	RMSE↓	RMSE log↓	$\delta < 1.25 \uparrow$
w/o mask	w/ mask							
✓				0.113	1.191	6.217	0.168	0.879
				0.111	1.219	6.132	0.165	0.883
				0.101	1.019	5.851	0.157	0.896
	✓			0.100	0.997	5.720	0.154	0.901
	✓	✓	✓	0.098	0.982	5.797	0.156	0.900
	✓	✓	✓	0.097	0.974	5.693	0.152	0.901

TABLE III
DEPTH ERROR ON DYNAMIC OBJECTS ON THE CITYSCAPES DATASET.

Methods	AbsRel↓	SqRel↓	RMSE↓	RMSE log↓	$\delta < 1.25 \uparrow$
MonoDepth2 [6]	0.159	1.944	6.461	0.214	0.820
ManyDepth [2]	0.164	2.140	6.597	0.219	0.779
DynamicDepth [3]	0.129	1.274	4.626	0.168	0.862
FlowDepth	0.122	1.203	4.561	0.159	0.874

TABLE IV
MODEL COMPLEXITY ANALYSIS.

Methods	AbsRel	RunTime (ms)	Inference params (Mb)	Total params (Mb)
ManyDepth [2]	0.114	34.0	29.8	29.8
DynamicDepth [3]	0.103	191.8	85.5(+55.7)	85.5(+55.7)
FlowDepth	0.097	78.5	54.2(+24.4)	79.0(+49.2)

fourth row, but it also has the ability to distinguish more accurately between foreground objects and backgrounds, such as the trees or the signs. This capability underscores how DCABlur effectively preserves valuable depth cues while blurring useless texture edges during the training stages. In low-texture regions, our model can estimate more precise depths, as demonstrated by the gradient depth estimation of the bus in the third row, This result validates the effectiveness of our proposed cost-volume sparse loss.

TABLE III shows the depth error on dynamic objects on the Cityscapes dataset. Our proposed DMFM can effectively improve the depth estimation accuracy in dynamic regions, surpassing all previous works on all metrics.

Complexity analysis. Running time and model parameter size are listed in TABLE IV. Our method and DynamicDepth are both based on the ManyDepth framework. However, compared to DynamicDepth which uses the semantic segmentation model EfficientPS [31], our model has a smaller size of model parameters both in training and inference, and it has an inference speed that is approximately 2.5 times faster. This is because the proposed DCABlur module and CVloss are only used during the training to help the model find the correct optimization direction. During the inference process, only the optical flow prior network RAFT-small in DMFM is used, which is smaller and faster compared to EfficientPS.

Transferability analysis. We also further tested the transferability of FlowDepth on VECAN datasets collected by

TABLE V
MODEL TRANSFERABILITY ANALYSIS.

Methods	W×H	AbsRel↓	RMSE↓	$\delta < 1.25 \uparrow$
ManyDepth [2]	512×192	0.116	4.779	0.871
DynamicDepth [3]	512×192	0.109	4.686	0.883
FlowDepth	512×192	0.096	4.243	0.899

‘RUIYU’ autonomous driving platform, independently developed by the VECAN lab at Tongji University, to validate its practical application capabilities. TABLE V presents a comparison of quantitative metrics, while Fig.7 shows a comparison of qualitative results. Since FlowDepth does not rely on any additional labeling and only requires video data for training, it demonstrates superior transferability compared to DynamicDepth, which can only use a frozen semantic segmentation network.

C. Ablation Study

In order to further analyse the role of each module proposed, we conducted ablation experiments on Cityscapes. As shown in TABLE II, we examined the effects of DMFM, DCABlur and CVLoss respectively. **DMFM:** Comparing the first three rows, we can see that adding the DMFM module without N_{mask} improves the performance slightly, but there is a larger decline for the SqRel metric due to the fact that the F^d , which is directly calculated by prior, contains the warp of background. Therefore, the introduction of N_{mask} solves this problem and improves the overall performance. **DCABlur:** Comparing the third and fourth rows, we can find that DCABlur mainly works on the high-freq regions to correct the fusion problem of foreground and background, which get a boost in the absolute metrics (RMSE/RMSE log). **CVLoss:** From the third and fifth rows, we can see that CVLoss does help on low-texture regions, so it has more improvement for the relative evaluation metrics (AbsRel/SqRel). Therefore, each module contributes to the overall depth estimation, and when all modules work together, they can significantly improve the accuracy.

V. CONCLUSION

We proposed a novel self-supervised multi-frame monocular depth estimation model FlowDepth. It decouples the moving objects using depth, pose, and flow prior in order to solve the mismatch problem caused by dynamic objects. The depth-cue-aware blur operation and the cost-volume loss also

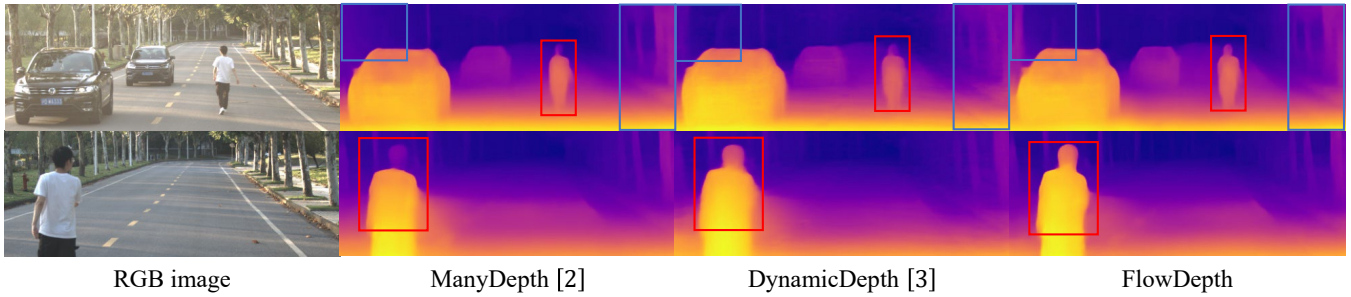


Fig. 7. Examples of depth estimations on our own VECAN Dataset which is used to verify the model transferability.

mitigate the unfairness of reprojection loss due to the high-freq or low-texture regions in the images. With the proposed innovations, our model outperforms other methods on the KITTI and Cityscapes datasets. Moreover, FlowDepth also demonstrates lower complexity and better transferability.

REFERENCES

- [1] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Nov 2017.
- [2] Jamie Watson, Oisín Mac Aodha, Victor Prisacariu, Gabriel Brostow, and Michael Firman. The temporal opportunist: Self-supervised multi-frame monocular depth. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nov 2021.
- [3] Ziyue Feng, Liang Yang, Longlong Jing, Haiyan Wang, YingLi Tian, and Bing Li. Disentangling object motion and occlusion for unsupervised multi-frame monocular depth. In *European Conference on Computer Vision*, pages 228–244. Springer, 2022.
- [4] Xingyu Chen, Ruonan Zhang, Ji Jiang, Yan Wang, Ge Li, and ThomasH. Li. Self-supervised monocular depth estimation: Solving the edge-fattening problem. Oct 2022.
- [5] Clément Godard, Oisín Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *arXiv: Computer Vision and Pattern Recognition*, Sep 2016.
- [6] Clément Godard, Oisín Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. *Cornell University - arXiv*, Jun 2018.
- [7] 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. *Cornell University - arXiv*, Nov 2020.
- [8] Yang Jiao, TracD. Tran, and Guangming Shi. Effiscene: Efficient per-pixel rigidity inference for unsupervised joint learning of optical flow, depth, camera pose and motion segmentation. *Cornell University - arXiv*, Nov 2020.
- [9] Xingyu Chen, ThomasH. Li, Ruonan Zhang, and Ge Li. Frequency-aware self-supervised monocular depth estimation. Oct 2022.
- [10] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. *arXiv: Computer Vision and Pattern Recognition*, May 2019.
- [11] Ning Zhang, Francesco Nex, George Vosselman, and Norman Kerle. Lite-mono: A lightweight cnn and transformer architecture for self-supervised monocular depth estimation. Nov 2022.
- [12] Tak-Wai Hui. Rm-depth: Unsupervised learning of recurrent monocular depth in dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1675–1684, 2022.
- [13] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, page 600–612, Apr 2004.
- [14] Vaishakh Patil, Wouter Van Gansbeke, Dengxin Dai, and Luc Van Gool. Don’t forget the past: Recurrent depth estimation from monocular video. *arXiv: Computer Vision and Pattern Recognition*, Jan 2020.
- [15] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. *DF-Net: Unsupervised Joint Learning of Depth and Flow using Cross-Task Consistency*, page 38–55. Jan 2018.
- [16] 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 *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019.
- [17] Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019.
- [18] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. *arXiv: Computer Vision and Pattern Recognition*, Nov 2018.
- [19] Seokju Lee, Sunghoon Im, Stephen Lin, and InSo Kweon. Learning monocular depth in dynamic scenes via instance-aware projection consistency. *arXiv: Computer Vision and Pattern Recognition*, Feb 2021.
- [20] Seokju Lee, Francois Rameau, Fei Pan, and InSo Kweon. Attentive and contrastive learning for joint depth and motion field estimation. *International Conference on Computer Vision*, Jan 2021.
- [21] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Feb 2020.
- [22] Zachary Teed and Jia Deng. *RAFT: Recurrent All-Pairs Field Transforms for Optical Flow*, page 402–419. Nov 2020.
- [23] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Aug 2020.
- [24] Xiaotian Chen, Yuwang Wang, Xuejin Chen, and Wenjun Zeng. S2r-depthnet: Learning a generalizable depth-specific structural representation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nov 2021.
- [25] Hyunyoung Jung, Eunhyeok Park, and Sungjoo Yoo. Fine-grained semantics-aware representation enhancement for self-supervised monocular depth estimation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2021.
- [26] Hanhan Li, Ariel Gordon, Hang Zhao, Vincent Casser, and Anelia Angelova. Unsupervised monocular depth learning in dynamic scenes. *Cornell University - arXiv*, Oct 2020.
- [27] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.
- [28] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.
- [29] Clément Godard, Oisín Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.
- [30] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015.
- [31] Rohit Mohan and Abhinav Valada. Efficientpts: Efficient panoptic segmentation. *International Journal of Computer Vision*, page 1551–1579, May 2021.