2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Channel-wise Motion Features for Efficient Motion Segmentation

Riku Inoue, Masamitsu Tsuchiya, Yuji Yasui

***Abstract*— For safety-critical robotics applications such as autonomous driving, it is important to detect all required objects accurately in real-time. Motion segmentation offers a solution by identifying dynamic objects from the scene in a class-agnostic manner. Recently, various motion segmentation models have been proposed, most of which jointly use subnetworks to estimate Depth, Pose, Optical Flow, and Scene Flow. As a result, the overall computational cost of the model increases, hindering real-time performance.**

In this paper, we propose a novel cost-volume-based motion feature representation, Channel-wise Motion Features. By extracting depth features of each instance in the feature map and capturing the scene's 3D motion information, it offers enhanced efficiency. The only subnetwork used to build Channel-wise Motion Features is the Pose Network, and no others are required. Our method not only achieves about 4 times the FPS of state-of-the-art models in the KITTI Dataset and Cityscapes of the VCAS-Motion Dataset, but also demonstrates equivalent accuracy while reducing the parameters to about 25%.

(a) Ground Truth (b) Output Mask (c) Feature Map

Fig. 1. The proposed model outputs masks and feature maps. (a) shows the Ground Truth masks from the VCAS-Motion Dataset [1], (b) shows the output motion segmentation masks from the proposed model, and (c) shows the feature maps of Channel-wise Motion Features. The feature maps of Channel-wise Motion Features are described in section IV-D.

I. INTRODUCTION

Understanding the visual scenes is foundational to computer vision in various applications, such as autonomous driving and robotics. When these systems operate in real-world environments, they often encounter objects that were not present in the training data. A crucial technique for effectively addressing these unknown scenarios is motion segmentation, which identifies moving objects from the scene in a class-agnostic manner [1]. This task is a very challenging problem because it requires an understanding of complex information in the scene. As a result, many recent deep learning-based motion segmentation models showcasing significant advancements employ multiple subnetworks to estimate Optical Flow, Depth, Pose, and Scene Flow, in turn raising concerns regarding computational efficiency and speed [1]–[5].

In this paper, we introduce a novel motion segmentation model that achieves real-time performance while maintaining competitive accuracy compared to methods integrating multiple subnetworks. Our approach is inspired by the multi-frame depth estimation model, ManyDepth [6], and the real-time instance segmentation model, SparseInst [7].

ManyDepth converts the 4D Cost Volume calculated from consecutive frames into a 3D Cost Volume by aggregating the channel directions and uses this information to estimate the depth of the target image. It's observed in the Cost Volume that regions with moving objects show irregular values compared to their surroundings, and ManyDepth introduces a method to address this issue. We hypothesize that this characteristic of Cost Volume can be effectively applied to motion segmentation.

SparseInst performs instance segmentation based on instance activation maps that emphasize information-rich regions of each object in a scene. Each activation map is tailored to highlight a specific instance, subsequently guiding the creation of object masks. We hypothesized that by aggregating the 3D information from the feature maps produced by the backbone of an instance segmentation model designed to learn instance activation maps, we could obtain the 3D motion information of each instance.

In this paper, we propose Channel-wise Motion Features, a novel representation built on the properties of Cost Volume and instance activation maps. Channel-wise Motion Features leverages feature maps from a backbone trained similarly to SparseInst to generate instance activation maps and constructs a 4D Cost Volume using a method consistent with ManyDepth, aggregating depth-direction information for each channel within the volume. This represents the 3D motion information of each instance emphasized in each channel. Moreover, since Cost Volume values for moving objects are inconsistent, this further enhances information beneficial for detecting moving objects. By integrating a 3D convolutional network designed to produce this representation into a model based on SparseInst, we have achieved an efficient motion segmentation model. Additionally, when constructing the 4D Cost Volume, we introduce a new depth range setting distinct from ManyDepth, specifically

The authors are with Honda R&D Co., Ltd., Akasaka, Minato-ku, Tokyo, 107-6238, Japan. (Email: riku_inoue@jp.honda)

tailored for detecting moving objects. As shown in Fig. 1, Channel-wise Motion Features show activations for each moving instance, making them effective features for motion segmentation.

In summary, the main contributions of this work are:

- We introduce Channel-wise Motion Features, which achieves approximately four times the frames per second and reduces parameters by about 25%, with only a 6.09% F-measure drop in the KITTI Dataset and a 0.48% CAQ decrease in the VCAS-Motion Cityscapes Dataset compared to state-of-the-art models.
- Through extensive experiments and comparison with other methods, we demonstrate the effectiveness of Channel-wise Motion Features.
- We propose a new depth range-setting method for cost volume construction and show its effectiveness through experiments.

II. RELATED WORK

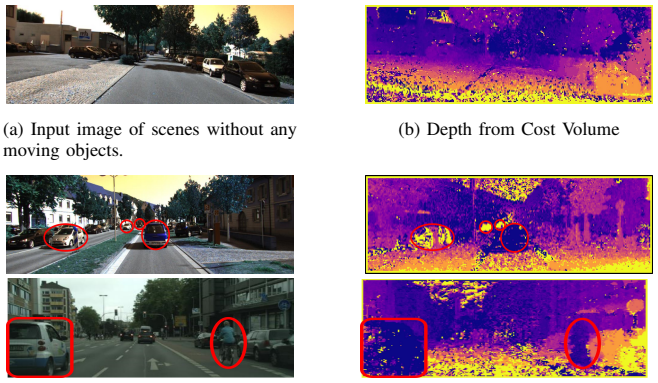
A. Motion Segmentation

Motion segmentation, which aims to detect moving objects from images, is recognized as a challenging task, as discussed in various prior works. The problems include motion degeneracy [3], the need for recognition of motion and 3D geometry [4], changes in camera ego-motion, lighting changes between consecutive frames, motion blur, and variations in pixel displacement due to different movement speeds [8]. Due to the complexity of motion segmentation, multiple subnetworks have been utilized to infer scene dynamics, such as optical flow [9], depth, optical flow [8], depth, ego-motion, optical flow [1], [4], [5], depth, optical flow, optical expansion [2], [3], and optical flow combined with LiDAR [10]. As a result, the real-time performance of these models is compromised, making their application in real-world systems, such as autonomous driving, challenging. On the other hand, our proposed model effectively addresses the trade-off between accuracy and inference speed, demonstrating real-time capabilities that make it viable for real-world applications.

B. Cost Volume

Cost volumes store data matching costs at corresponding pixels between two frames and are commonly used in multi-frame depth estimation [6], [11], [12], stereo matching [13], [14], and optical flow estimation [15], [16]. In self-supervised multi-frame depth estimation, the source image is warped by considering only ego-motion, which is suffered by moving objects, when acquiring monocular depth information from the cost volume. ManyDepth [6] addresses this problem through consistency with single-frame depth networks, and DepthFormer [11] adopting the same approach. MonoRec [12] handles moving objects using dynamic masks predicted by MaskModule.

Our model exploits the problem of moving objects in Cost Volume to enhance motion segmentation information.



(a) Input image of scenes without any moving objects.

(b) Depth from Cost Volume

(c) Input image of scenes with moving objects present.

(d) Depth from Cost Volume

Fig. 2. Inconsistency value of cost volume in dynamic object region. (a) and (c) are input images without and with moving objects present, respectively. (b) and (d) represent the lowest values extracted along the depth direction from the cost volumes corresponding to their respective input images. In (b), due to the absence of moving objects, it appears continuous, much like a depth map. However, in (d), the moving object indicated by the red circle is clearly discontinuous in value compared to its surroundings.

As seen in Fig. 2, in regions where a moving object is present, the Cost Volume has irregular values relative to the values in the surrounding region. Channel-wise Motion Features, which was created by taking advantage of this fact is an effective and efficient representation for moving object detection.

C. Instance Segmentation

The task of instance segmentation is to segment object instances and estimate the mask and class of each. As a pioneering model, Mask R-CNN [17] extends the framework of Faster R-CNN [18], an object detection model, by adding a mask branch for predicting segmentation masks, setting a new standard in instance segmentation. Region-based methods such as Mask R-CNN first detect object bounding boxes as in Faster R-CNN, and then RoI operations, such as RoI-Pooling [18] or RoI-Align [17] are applied. These operations extract region features for estimating object classes and segmentation masks. Subsequent research has primarily focused on addressing the low-quality segmentation masks predicted by Mask R-CNN [19], [20] and improving the precision of bounding box detections [21], [22].

Another class of methods, instance activation based methods, has been proposed to learn the features of these pixels in order to highlight the information-rich regions of each foreground object. Typical of such methods are those based on center activation [23], [24], which represent objects with center pixels instead of bounding boxes and segment them using center features. SOLO [25], [26] generates mask kernels for segmentation by object centers. In recent years, several real-time instance segmentation models of instance activation methods have been proposed [7], [27]. SparseInst [7] proposes a sparse set of instance activation maps as a new object representation and achieves fast and high performance. Our method is based on SparseInst, but the base model alone

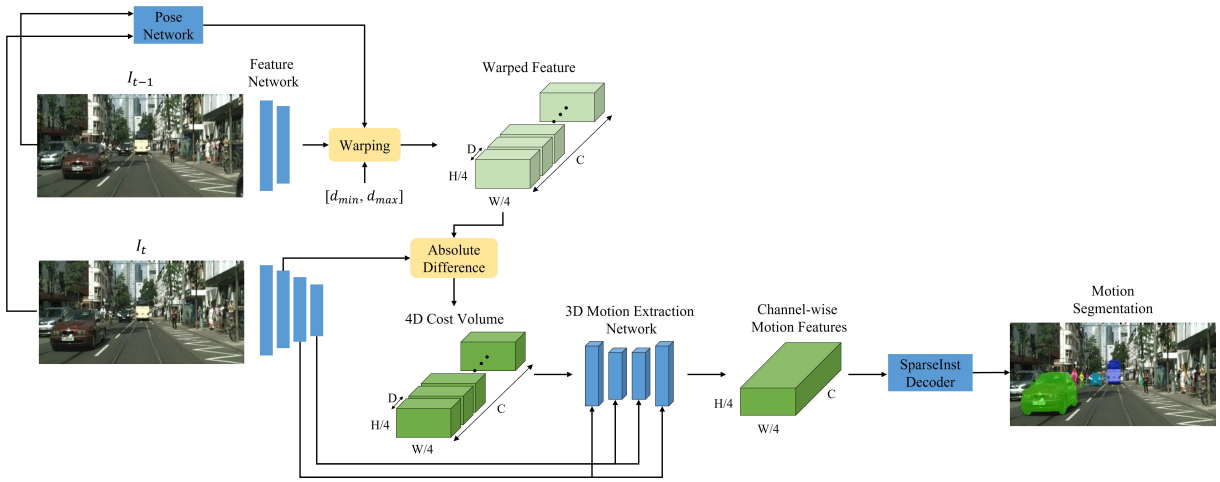


Fig. 3. Overview of our proposed model. From two consecutive frames, I_{t-1} and I_t, the Pose Network estimates the camera's ego-motion. Then, the Feature Network, which shares weights, extracts features from each of the two frames separately. By utilizing this estimated motion along with a predefined set of depth values, we warp the feature map F_{t-1} to align it with the perspective of I_t. The 3D Motion Extraction Network then transforms the 4D Cost Volume($D \times C \times H \times W$) into Channel-wise Motion Features($C \times H \times W$), enabling the extraction of motion information for each channel. Finally, leveraging the instance activation maps, the SparseInst Decoder estimates the segmentation masks.

has only low accuracy in motion segmentation. On the other hand, by incorporating our proposed Channel-wise Motion Features into SparseInst, we can improve the performance of motion segmentation.

III. METHOD

In this section, we first describe the overall architecture of our proposed model, then discuss each component of detail.

A. Overall Architecture

As shown in Fig. 3, our network architecture is built upon the structure of SparseInst [7].

Given the consecutive frames I_{t-1}, $I_t \in \mathbb{R}^{3\times H\times W}$, we first apply the Pose Network following [6] and predict relative camera pose $T_{t\to t-1}$. We apply a Feature Network with shared weights to consecutive frames separately, obtaining a feature map F^1_{t-1} from I_{t-1}, and three feature maps F^1_t, F^2_t, F^3_t from I_t. The Feature Network is composed of the first four layers of ResNet50 [28], and the resolution of F^1_t, F^2_t, and F^3_t are $1/4, 1/8$, and $1/16$ of the input image, respectively. The warped feature $F^1_{t-1\to t} \in \mathbb{R}^{D\times C\times H\times W}$ is obtained by warping F^1_{t-1} to the viewpoint of I_t using $T_{t\to t-1}$, with the depth range represented by d_{min} and d_{max} which are manually set. The methodology for determining d_{min}, d_{max} is explained in Section III-C. We then build a 4D Cost Volume calculating the absolute difference between the $F^1_{t-1\to t}$ and F^1_t. Channel-wise Motion Features, which aggregate 3D information for each channel, are generated by passing the 4D Cost Volume and F^2_t, F^3_t through a 3D Motion Extraction Network. In the end, we estimate the instance motion segmentation mask by feeding the Channel-wise Motion Features into an IAM-based Segmentation Decoder [7].

B. Channel-wise Motion Features

Recently proposed instance segmentation models have been successful in using the activation of feature maps [7], [27]. Similarly, in our model, we believe that the Feature Network can extract features regarding activation for each instance in the input image. A key idea in our method is computing and aggregating three-dimensional matching between instance activation features of different frames obtained from the Feature Network, enabling the extraction of motion information of instances. Below we provide details of our proposed method: Channel-wise Motion Features.

1) Construction of the 4D Cost Volume: Cost volume captures the correspondence between pixels in the different views. Our method constructs a 4D cost volume following the multi-frame depth estimation technique [6].

In this setup, the I_{t-1} serves as the source image, and I_t as the target image. The relative camera pose between these two frames is determined through the Pose Network. We generate the warped feature volume $F_{t-1\to t} \in \mathbb{R}^{D\times C\times H\times W}$ by warping the source features F_{t-1} to the viewpoint of the target image I_t using a plane of depth value d_i, linearly sampled within the depth range from d_{min} to d_{max} along with the estimated relative camera pose. The warped feature volume is computed as follows:

$$F_{t-1\to t}(d_i) = F_{t-1}\langle proj(d_i, T_{t\to t-1}, K)\rangle \tag{1}$$

where $proj()$ denotes projection function provides the 2D coordinates of the sampled depth value d_i in F_{t-1} and $\langle\rangle$ is the sampling operator. $K \in \mathbb{R}^{3\times 3}$ denotes a camera intrinsics.

Subsequently, the element of the 4D Cost Volume $4DCV$ at depth d_i is given by:

$$4DCV(d_i, c, x, y) = |F_{t-1\to t}(d_i, c, x, y) - F_t(c, x, y)| \tag{2}$$

where $||$ denotes the absolute difference, x and y are spatial coordinates, and c represents the channel dimension. This

computation is performed for each depth, spatial location, and channel, yielding a 4D representation.

2) Channel-wise Motion Features Extraction: We present a detailed methodology for transforming the 4D Cost Volume into Channel-wise Motion Features. Channel-wise Motion Features aggregates 3D information per channel and computes motion information between consecutive frames. In multi-frame depth estimation [6], [12], the 4D Cost Volume is aggregated along the channel direction to construct a 3D Cost Volume with $D \times H \times W$. On the other hand, our approach aggregates information along the depth direction, resulting in a three-dimensional representation of $C \times H \times W$.

For the transformation of the 4D Cost Volume, we employ a network similar to the 3D Regularization Network presented in [13]. We made the 3D Regularization Network more lightweight for computational efficiency and repurposed it as a feature extractor for motion detection. Throughout this paper, we will refer to our adapted version of the 3D Regularization Network as the 3D Motion Extraction Network. The architecture of the 3D Motion Extraction Network is illustrated in Fig. 4. The 3D Motion Extraction Network comprises two down-sampling blocks and two up-sampling blocks. Each down-sampling block contains two 3D convolutions of size $3 \times 3 \times 3$. On the other hand, the up-sampling block consists of a block designed with a single $4 \times 4 \times 4$ 3D transposed convolution, and a block designed with a $4 \times 4 \times 4$ 3D transposed convolution and two $3 \times 3 \times 3$ 3D convolutions. The 3D Regularization Network adopts guided cost volume excitation [29], which enhances geometric features by applying spatially varying updates to the cost volume feature map. Transform the feature map, which matches the scale of the cost volume, to align with its channel dimensions and employ it as weights. Utilizing these feature map weights, the guided cost volume excitation at scale i can be represented as follows:

$$w = \sigma(f(F_i)) \tag{3}$$

$$C_i' = w \odot C_i \tag{4}$$

where f denotes 2D point-wise convolution, σ is the sigmoid function, and $\odot$ is the Hadamard product. In the final layer of the 3D Motion Extraction Network, 3D transposed convolution is applied to reduce the dimension in the D direction to 1, we obtain Channel-wise Motion Features of size $C \times H \times W$.

C. Setting Depth Range

When constructing the Cost Volume, selecting the optimal depth range $[d_{min}, d_{max}]$ is important. If the chosen range is excessively broad beyond what is necessary for depth estimation, it leads to sparse depth value sampling, compromising the accuracy of dense matching. Conversely, increasing the depth samples for detailed matching amplifies computational demands. For efficient dense matching with minimal samples and computational cost, choosing the right depth range is essential.

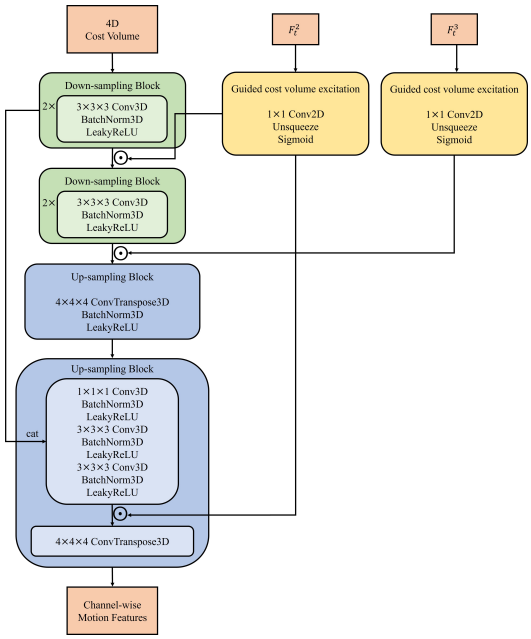


Fig. 4. 3D Motion Extraction Network. *Cat* denotes a concatenation operation. The multiplication of numbers described before convolution represents the size of the convolution kernel. In the $1 \times 1 \times 1$ 3D convolution of the second down-sampling block, we adjust the dimensions in the depth direction.

In our work, we introduce modifications to the ManyDepth framework [6], which leverages outputs from single-image Depth Networks. Our key adaptation is tailored for motion segmentation, wherein the cost volume computation specifically hones in on the depth range containing moving objects. In ManyDepth, the values for d_{min} and d_{max} are determined as follows:

$$d_{min} = 0.99 * d_{min} + 0.01 * \hat{d}^i_{min} \tag{5}$$

$$d_{max} = 0.99 * d_{max} + 0.01 * \hat{d}^i_{max} \tag{6}$$

where $\hat{d}^i_{min}$ and $\hat{d}^i_{max}$ denote the average of the minimum and maximum depth values output by the Depth Network at iteration i, respectively. However, as shown in Fig. 5, the Depth Network occasionally produces output errors. Moreover, while Equation (5) and Equation (6) are applied across the entire image, they include depth values from the background regions. Yet, for constructing a Cost Volume for moving object detection, only the depth range where the moving object exists is crucial. We therefore determined the depth range by considering only the regions where moving objects exist in the depth values output by the Depth Network and by excluding outliers. Fig. 6 shows the depth value distribution for moving objects in the training data for the VCAS-Motion Dataset [1]. The Depth range set by the ManyDepth manner is [0.114, 17.95], whereas our method sets the depth range to [0.091, 2.646]. We set d_{min} and d_{max} based on the 1st and 99th percentiles of the depth distribution of moving objects in each dataset. The depth value distribution is constructed by taking the average depth

(a) Input image (b) Depth estimation error

Fig. 5. Estimation error by Depth Network. Even with a trained Depth Network, sometimes estimates a miss value. In this example, the maximum depth value for the car outlined in red is 58.61. This value is notably high in the context of the depth distribution and can negatively affect the depth range settings.

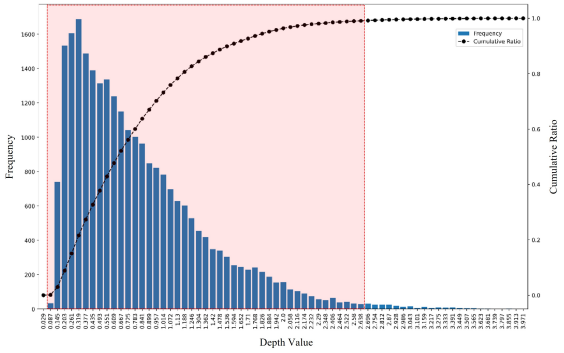


Fig. 6. Distribution of the average depth values for each moving object in the VCAS-Motion Dataset. This graph visualizes the distribution of depth values within the range of 0.0 to 4.0. The red region includes the depth range of $d_{min} = 0.091$ to $d_{max} = 2.646$ that was used in our experiments.

value within the ground truth mask region of the moving object as its depth value. Notably, the learning of depth is conducted for the pre-training of the backbone and for determining the depth range, but the Depth Network is not utilized during the motion segmentation training.

D. Training Loss

Following [7], the training loss is defined as:

$$L = L_M + \lambda_C L_C + \lambda_S L_S \tag{7}$$

where L_M is mask loss composed by dice loss [30] L_D and pixel-wise binary cross entropy loss L_P:

$$L_M = \lambda_D L_D + \lambda_P L_P \tag{8}$$

L_C is classification loss for estimated object class and uses focal loss [31] and L_S denotes binary cross entropy loss for the objectness. The λ in each equation represents a coefficient, we set $\lambda_C = 2.0$, $\lambda_S = 3.0$, $\lambda_D = 2.0$, and $\lambda_P = 5.0$.

IV. EXPERIMENTS

A. Dataset and Evaluation Metrics

KITTI [32] is a real-world dataset for autonomous driving. This dataset is annotated with masks for moving cars in 200 images. Following prior research, we report the F-measure [33] as an evaluation metric for instance segmentation masks of moving objects, and the IoU [5], [34] for the non-moving background regions mask. We utilize the pre-trained weights from the Depth Network of the ManyDepth framework for the Backbone.

VCAS-Motion [1] is a dataset that has been annotated with motion segmentation labels for Cityscapes-VPS [35] and KITTIMOTS [36]. In this dataset, annotation masks are provided for moving objects within eight categories across 11,008 images. Following previous works [1], we adopt class-agnostic quality as our evaluation metric, defined by the following equations:

$$SQ = \frac{\sum_{p,g \in TP} IoU(p,g)}{|TP|} \tag{9}$$

$$RQ = \frac{|TP|}{|TP| + |FN|} \tag{10}$$

$$CAQ = SQ \cdot RQ \tag{11}$$

TP and FN represent the true positive and false negative, respectively. An instance mask is deemed a true positive only if its intersection over union with the ground-truth exceeds 0.5. We utilize the pre-trained weights from the Depth Network of the ManyDepth framework for the Backbone.

B. Implementation Details

We implemented our model using PyTorch [37] and trained it on two NVIDIA RTX A6000 GPUs. For inference speed, we measured frames per second (FPS) using a single NVIDIA RTX 3080Ti Mobile GPU for all models used in our experiments. All training employed AdamW as an optimizer, with β_1=0.9, β_2=0.999, weight decay of 0.001, and scheduled by cosine learning rate decay scheduler [38]. The detailed hyperparameters for each training session are provided in Table I. For all training phases, we utilize data augmentation techniques, including horizontal flips and random variations in brightness, contrast, saturation, and hue. The respective ranges are ± 0.2 for brightness, contrast, and saturation, and ± 0.1 for hue. We set the number of depth bins for Cost Volume construction at 64. We employed the ManyDepth framework for our Pose Network and trained it using all accessible data from KITTI and Cityscapes, or a combination of both, excluding their test sets. During this process, the depth distribution of moving objects from the Depth Network was utilized to define the depth range for the Cost Volume. It's important to note that the Depth Network was not utilized during the learning or inference phases for motion segmentation.

In the VCAS-Motion Dataset, Cityscapes is sampled every 5 frames, and we adopted a 2-frame interval for both training and evaluation to match the frame rate with KITTI. Additionally, due to the resolution discrepancies between KITTI and Cityscapes, the size of the Cost Volume differ. To address this, both datasets were first trained jointly at a resolution of 320×960. Subsequent training was conducted on each dataset at its corresponding resolution.

TABLE I: Details for training hyperparameters. The gradient accumulation column indicates how many iterations are performed before updating the model.

Training Session	Epochs	Learning Rate	Batch Size / GPU	Gradient Accumulation	Depth Range
Depth and Pose (ManyDepth)	20	1×10^{-4}	14	-	-
KITTI Dataset	20	backbone: 5×10^{-5}, other: 1×10^{-4}	2	4	[0.090, 2.465]
VCAS-Motion 320×960	15	backbone: 5×10^{-5}, other: 1×10^{-4}	6	4	[0.091, 2.646]
VCAS-Motion KITTI	5	backbone: 1×10^{-5}, other: 5×10^{-5}	2	8	[0.081, 2.424]
VCAS-Motion Cityscapes	5	backbone: 1×10^{-5}, other: 5×10^{-5}	2	8	[0.101, 2.444]

TABLE II: Results on the KITTI Dataset. Obj F represents the F-measure for the moving object masks. Bg IoU denotes the Intersection over Union (IoU) for the background, which excludes the moving objects. Evaluations of models other than SparseInst and TMO are described in accordance with the assessments reported in [3]. TMO is a state-of-the-art model in the video object segmentation tasks, utilizing RGB images and optical flow as inputs to produce segmentation masks. For optical flow estimation, we utilize RAFT [16]. Similar to our proposed model, SparseInst has been adapted to process two consecutive frames as input. It dependently extracts features from each frame using the backbone, followed by concatenating these features before feeding them into the context encoder.

Method	Params↓	FPS↑	Obj F (%) ↑	Bg IoU (%) ↓
competitive collaboration [5]	5.22M	196.08	50.87	85.50
COSNet [39]	81.24M	4.09	66.67	93.03
MAT-Net [40]	142.69M	3.75	68.40	93.08
FusionSeg [41]	85.19M	4.61	85.08	96.27
TMO [42]	38.85M	10.55	80.75	98.16
Rigidmask [3]	138.52M	4.92	90.71	97.05
SparseInst [7]	35.73M	43.48	65.15	98.17
Ours	35.71M	19.12	84.62	98.40

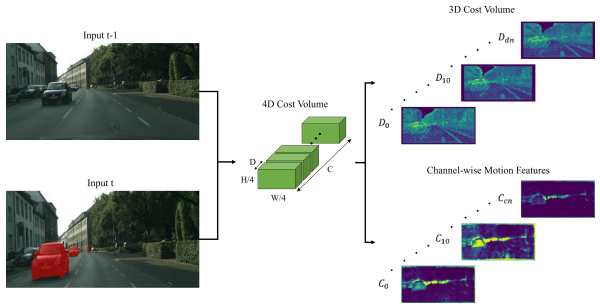


Fig. 7. Visualization of Feature Maps for Channel-wise Motion Features and 3D Cost Volume. Channel-wise Motion Features shows activation for each moving instance, whereas 3D Cost Volume shows activation across the entire image. In the input image t, the red mask indicates moving objects. Channel-wise Motion Features here is represented as a feature map that has undergone 2D convolution without changing the dimension of the feature volume after the 3D Motion Extraction Network.

C. Results

Following prior studies, we compare our proposed method with other models. To show that our model is both efficient and effective in motion segmentation, we evaluate the FPS and the number of model parameters, in addition to the metrics presented in Section IV-A.

Table II shows the results for the KITTI dataset. Compared to the state-of-the-art method, Rigidmask, our model achieves approximately four times the FPS and operates with roughly one-quarter of the parameters. Yet, the disparity in F-measure for moving objects is 6.09%. Additionally, our model exceeds Rigidmask in IoU for the background. We also observed that our model consistently outperforms other models in the balance between efficiency and precision. Compared to the base model SparseInst, our model, while having a similar number of parameters, achieves 19.47% improvement in F-measure.

In Table III, we show results for the VCAS-Motion Dataset. Compared to the state-of-the-art method, VCANet, our proposed model achieves over four times the FPS and less than one-quarter of the Motion Segmentation Network parameters, while maintaining nearly equivalent accuracy on Cityscapes. Furthermore, considering the need for VCANet to estimate ego-flow suppression as an input, the difference is believed to be even more pronounced. On the other hand, in the KITTI, our model underperforms compared to VCANet in CAQ by 18.22%. Compared to the base model SparseInst, our model outperforms by 20.03% in RQ and 15.73% in CAQ on the KITTI dataset. Similarly, on the Cityscapes dataset, it outperforms by 34.31% in RQ and 28.70% in CAQ.

D. Ablation Analysis

To demonstrate the effectiveness of Channel-wise Motion Features, we conduct two ablation analyses:

1. Comparison of Channel-wise Motion Features and 3D Cost Volume. Channel-wise Motion Features not only captures the property of the 3D Cost Volume that takes the "inconsistency value of the moving object region" but also aggregates information in the depth direction for each channel. We verify its effects based on experiments.
2. Comparison regarding depth range settings. We examine to what extent the depth range set for constructing the Cost Volume affects the accuracy of motion segmentation.

Comparison between Channel-wise Motion Features and 3D Cost Volume.

In this experiment, the 3D Cost Volume is constructed by taking the average of the channel direction of the 4D Cost Volume, the same as ManyDepth.

Fig. 7 illustrates the differences in the feature maps of Channel-wise Motion Features and the 3D Cost Volume. Channel-wise Motion Features aggregates information in the Depth direction for each channel of the feature maps between consecutive frames, and it exhibits activations to moving objects. This indicates that it is capable of capturing information regarding the motion of each instance. Furthermore, as shown in Table IV, on the VCAS-Motion Dataset, the use of Channel-wise Motion Features yields better results compared to the use of 3D Cost Volume.

Effects of setting d_{min} and d_{max} for motion segmentation.

We compare the depth range setting method we proposed with the method proposed in ManyDepth. In Table V, we show the results obtained when applying both depth range

TABLE III: Results on the VCAS-Motion Dataset. We report the SQ, RQ, and CAQ metrics for KITTI and Cityscapes, respectively. The FPS was measured on images with a resolution of 320×960. The $+\alpha$ in VCANet indicates the inference time of Ego-Flow Suppression in terms of FPS. VCANet consists of the Motion Segmentation Network with 92.08M parameters and the network for Ego-flow Suppression estimation with 190.35M parameters. The parameter breakdown for the Ego-flow Suppression estimation is as follows: FlowNet2 [43] utilized for optical flow estimation has 162.51M, MonoDepth2 [44]'s Depth Network comprises 14.84M, and its Pose Network contains 13.00M, culminating in a total of 190.35M. The number of parameters in our model is 22.71M for Motion Segmentation Network and 13.00M for the Pose Network.

Method	Params↓	FPS↑	KITTI			Cityscapes		
			SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑	SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑
VCANet [1]	92.08M+190.35M	6.23+α	82.4	75.4	62.1	78.0	56.2	43.8
SparseInst [7]	35.73M	47.62	79.96	35.21	28.15	68.52	21.34	14.62
Ours	22.71M + 13.00M	26.89	79.44	55.24	43.88	77.85	55.65	43.32

TABLE IV: Comparison between using 3D Cost Volume and Channel-wise Motion Features on the VCAS-Motion Dataset.

Method	KITTI			Cityscapes		
	SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑	SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑
3D Cost Volume	73.08	49.70	36.32	76.08	49.13	37.38
Our setting methods	79.44	55.24	43.88	77.85	55.65	43.32

TABLE V: Comparison of depth range settings. In the ManyDepth approach, the depth range for VCAS-Motion 320 × 960 is [0.114, 17.95], that for KITTI is [0.105, 8.471], and that for Cityscapes is [0.105, 27.05]. Our settings are the same as in Table I.

Depth Range	KITTI			Cityscapes		
	SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑	SQ (%) ↑	RQ(%) ↑	CAQ(%) ↑
ManyDepth manner	77.89	49.86	38.84	77.33	54.53	42.17
Our setting methods	79.44	55.24	43.88	77.85	55.65	43.32

settings to our model. Our method outperforms the setting approach of ManyDepth across all evaluation metrics, indicating that limiting the range setting to the moving object detection area is effective for motion segmentation.

V. CONCLUSION

We proposed Channel-wise Motion Features, a novel motion feature representation for motion segmentation. We tackled the previously unattained trade-off between inference speed and accuracy, demonstrating the efficiency and effectiveness of our proposed model. Experiments showed that our model could achieve approximately four times the FPS and reduce parameters by about 25%, with only a 6.09% F-measure drop on the KITTI Dataset and a 0.48% CAQ decrease on the VCAS-Motion Cityscapes Dataset compared to state-of-the-art models. Furthermore, through ablation analysis, we showed that Channel-wise Motion Features effectively captures features of moving objects. We expect that our findings will lead to advancements in motion segmentation and applications such as autonomous driving.

Limitations. Similar to previous methods based on instance activation [7], [27], our model exhibits a weakness in detecting smaller objects. Additionally, distant moving objects, which exhibit subtle apparent motions, are not detected. Potential solutions include employing high-resolution feature maps when constructing the Cost Volume. However, such approaches significantly increase the computational cost. Additionally, since processing the 4D Cost Volume with 3D convolutions requires a large amount of GPU memory, it is necessary to explore practical solutions to address this issue. We intend to address these challenges in our future research.

REFERENCES

[1] M. Siam, A. Kendall, and M. Jagersand, "Video class agnostic segmentation benchmark for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2825–2834.

[2] M. Neoral, J. Šochman, and J. Matas, "Monocular arbitrary moving object discovery and segmentation," 2021.

[3] G. Yang and D. Ramanan, "Learning to segment rigid motions from two frames," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1266–1275.

[4] C. Homeyer and C. Schnörr, "On moving object segmentation from monocular video with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 880–891.

[5] A. Ranjan, V. Jampani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black, "Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 240–12 249.

[6] J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The temporal opportunist: Self-supervised multi-frame monocular depth," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1164–1174.

[7] T. Cheng, X. Wang, S. Chen, W. Zhang, Q. Zhang, C. Huang, Z. Zhang, and W. Liu, "Sparse instance activation for real-time instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[8] J. Vertens, A. Valada, and W. Burgard, "Smsnet: Semantic motion segmentation using deep convolutional neural networks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 582–589.

[9] P. Tokmakov, C. Schmid, and K. Alahari, "Learning to segment moving objects," *International Journal of Computer Vision*, vol. 127, pp. 282–301, 2019.

[10] H. Rashed, M. Ramzy, V. Vaquero, A. El Sallab, G. Sistu, and S. Yogamani, "Fusemodnet: Real-time camera and lidar based moving object detection for robust low-light autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[11] V. Guizilini, R. Ambruș, D. Chen, S. Zakharov, and A. Gaidon, "Multi-frame self-supervised depth with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 160–170.

[12] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6112–6122.

[13] G. Xu, X. Wang, X. Ding, and X. Yang, "Iterative geometry encoding volume for stereo matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 919–21 928.

[14] L. Lipson, Z. Teed, and J. Deng, "Raft-stereo: Multilevel recurrent field transforms for stereo matching," in *2021 International Conference on 3D Vision (3DV)*. IEEE, 2021, pp. 218–227.

[15] T.-W. Hui and C. C. Loy, "Liteflownet3: Resolving correspondence ambiguity for more accurate optical flow estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer, 2020, pp. 169–184.

[16] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[19] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9799–9808.

[20] C. Tang, H. Chen, X. Li, J. Li, Z. Zhang, and X. Hu, "Look closer to segment better: Boundary patch refinement for instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 926–13 935.

[21] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[22] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang *et al.*, "Hybrid task cascade for instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4974–4983.

[23] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, and P. Luo, "Polarmask: Single shot instance segmentation with polar representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 193–12 202.

[24] Z. Tian, C. Shen, and H. Chen, "Conditional convolutions for instance segmentation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer, 2020, pp. 282–298.

[25] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "Solo: Segmenting objects by locations," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*. Springer, 2020, pp. 649–665.

[26] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "Solov2: Dynamic and fast instance segmentation," *Advances in Neural information processing systems*, vol. 33, pp. 17 721–17 732, 2020.

[27] J. He, P. Li, Y. Geng, and X. Xie, "Fastinst: A simple query-based model for real-time instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 663–23 672.

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

[29] A. Bangunharcana, J. W. Cho, S. Lee, I. S. Kweon, K.-S. Kim, and S. Kim, "Correlate-and-excite: Real-time stereo matching via guided cost volume excitation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3542–3548.

[30] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*. Ieee, 2016, pp. 565–571.

[31] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

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

[33] P. T. Achal Dave and D. Ramanan, "Towards segmenting anything that moves," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[34] Z. Lv, K. Kim, A. Troccoli, D. Sun, J. M. Rehg, and J. Kautz, "Learning rigidity in dynamic scenes with a moving camera for 3d motion field estimation," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 468–484.

[35] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon, "Video panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9859–9868.

[36] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "Mots: Multi-object tracking and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7942–7951.

[37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[38] I. Loshchilov and F. Hutte, "Sgdr: Stochastic gradient descent with warm restarts," in *International Conference on Learning Representations*, 2017.

[39] X. Lu, W. Wang, C. Ma, J. Shen, L. Shao, and F. Porikli, "See more, know more: Unsupervised video object segmentation with co-attention siamese networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3623–3632.

[40] T. Zhou, S. Wang, Y. Zhou, Y. Yao, J. Li, and L. Shao, "Motion-attentive transition for zero-shot video object segmentation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 13 066–13 073.

[41] S. Dutt Jain, B. Xiong, and K. Grauman, "Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3664–3673.

[42] S. Cho, M. Lee, S. Lee, C. Park, D. Kim, and S. Lee, "Treating motion as option to reduce motion dependency in unsupervised video object segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5140–5149.

[43] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.

[44] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.