2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

EfficientDPS: Efficient and End-to-End Depth-aware Panoptic Segmentation

Shengkai Wu[1],[2], Liangliang Ren[1], Linfeng Gao[1], Yupeng Li [1], Wenyu Liu[2]

***Abstract*— Depth-aware panoptic segmentation (DPS) combines image segmentation and monocular depth estimation in a single model to achieve semantic and geometry perception simultaneously. DPS task has important applications in the robot area but the previous DPS models are too heavy to be applied. Thus, we propose EfficientDPS, an efficient, end-to-end, and unified model for DPS. In our method, query features extracted with convolution networks are used to represent things/stuff. In this way, different vision tasks such as classification, segmentation, and depth estimation can be realized in a unified manner, leading to a compact and efficient model. EfficientDPS can be trained and tested in an end-to-end manner via bipartite matching and complex post-process is not needed at inference. To enhance the supervision signal, group query representation is proposed, leading to better performance without affecting the inference speed. Extensive experiments on Cityscapes-DPS and SemKITTI-DPS show that EfficientDPS can achieve the best trade-off between speed and accuracy than the state-of-the-art methods.**

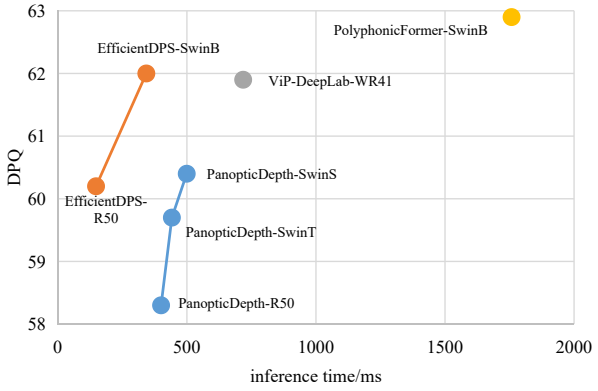

Fig. 1. Speed-and-accuracy Trade-off. Our method achieves the best trade-off between speed and accuracy.

I. INTRODUCTION

For efficient semantic and 3D geometry perception of the environment, depth-aware panoptic segmentation (DPS) [1], [2] is proposed to combine panoptic segmentation and monocular depth estimation into a single model. For panoptic segmentation [3], every pixel in the image is assigned a semantic label and an instance ID, which enables the robots to understand the semantics of things and stuff. Monocular depth estimation [4] aims to estimate the depth via a single camera image and is an important way to perceive the 3D geometry of the environment. The ability to simultaneously perceive semantics and 3D geometry of the environment makes the DPS models have important applications in robots and self-driving vehicles.

However, the previous DPS models are too complex and heavy. Most of the DPS models can be classified into two categories: CNN-based models and Transformer-based models. The CNN-based DPS models such as Vip-deeplab [2], PanopticDepth [1] mostly deal with the things/stuff and different vision tasks via different networks and make the models complex. In addition, these models can't be trained and tested in an end-to-end manner. Thus, complex post-processes such as NMS [2], and kernel fusion [1] are desired. Inspired by DETR [5], the Transformer-based DPS models such as Polyphonicformer [6] unify the representation of things/stuff and the realization of different vision tasks. These models can be trained and tested in an end-to-end manner. However, the heavy utilization of Transformers [7] to extract representative query features makes these models inefficient and difficult to be deployed in embedded AI chips such as NPU and DPU, which are widely used in robots. Thus, to be applied to devices with limited computing resources, the DPS models should be simple, efficient, and deployment-friendly.

In this paper, we propose an efficient and end-to-end depth-aware panoptic segmentation model (EfficientDPS) based on convolutional networks. EfficientDPS represents things and stuff with unified query features that are extracted by the query network. Specifically, the query network first predicts the query activation maps to represent the location of things and stuff in the image. Then the query features are extracted by adaptively aggregating the query feature map with the corresponding query activation maps. Based on the unified representation of things/stuff, different vision tasks can be easily unified into a single network. We utilize dynamic convolution between refined query features and semantics/depth embedding maps to realize query-wise segmentation and depth estimation.

During training, the label assignment between targets and predictions is formulated as a bipartite matching problem similar to DETR [5]. Hungarian algorithm [8] is utilized to compute the optimal one-to-one assignment between targets and predictions. To enhance the supervision signal for better performance, we propose group query representation in the query network. The targets are assigned to each group of queries during training and only one group of queries is kept during inference without affecting the inference speed.

Extensive experiments on Cityscapes-DPS and SemKITTI-DPS demonstrate that EfficientDPS can

[1]CVTE, China, wushengkai@cvte.com
[2]Huazhong University of Science and Technology, China liuwy@hust.edu.cn

achieve the best trade-off between speed and accuracy with an efficient, deployment-friendly model architecture. The inference speed of our method is $3.0 \sim 5.1$ times faster compared with the state-of-the-art methods.

The main contribution of our work can be summarized as follows.

- We propose EfficientDPS, an efficient and deployment-friendly DPS model, which achieves the best trade-off between speed and accuracy.
- We demonstrate that things and stuff can be represented in a unified way based on the convolution networks.
- We conduct extensive experiments to demonstrate the effectiveness of EfficientDPS.

II. Related Work

Panoptic Segmentation. Traditional panoptic segmentation methods [3], [9], [10] tackle instance segmentation and semantic segmentation separately. For semantic segmentation, these methods generally adopt an encoder-decoder framework. For instance segmentation, Panoptic FPN [3] uses RoI-Pooling [11] to extract region feature maps for the instance segmentation. Panoptic Deeplab [10] and RealTimePan [9] detect object centers and then group the instance embeddings. Thus, realizing the two tasks with different designs makes the models complex. Recently, inspired by DETR [5], many methods such as Max-DeepLab [12], K-Net [13], MaskFormer [14], and SegFormer [14] propose to solve the semantic and instance segmentation in a unified manner. These methods represent things and stuff with unified query features extracted with multiple Transformer decoder layers, which are important for the representation ability of query features. Finally, these query features are used for classification and segmentation. These models can be trained and tested in an end-to-end manner, and complex post-processes are not desired. However, the heavy utilization of Transformers makes these models inefficient and difficult to be deployed in the embedding system. Similarly, our method also represents things and stuff with unified queries. However, the query features are extracted with convolutional layers inspired by SparseInst [15] which makes our method efficient and deployment-friendly.

Depth-aware Panoptic Segmentation. The previous works [16], [17] have explored the combination of semantic/instance segmentation and depth estimation to achieve the DPS task. Vitor et al [16] proposes to leverage semantic structure to guide geometric representation learning in the self-supervised monocular depth estimation. SDC-Depth [17] leverages instance/semantic segmentation to improve depth estimation. However, the different tasks are not unified in the model design and evaluation metric. Recently, Vip-Deeplab [2] proposes the DPS datasets and evaluation metric. Vip-Deeplab separately solves the segmentation and depth estimation problems by combining Panoptic-Deeplab [10] with a dense depth estimation head. Complex post-processes such as non-maximum suppression and instance embedding grouping are required. Based on PanopticFCN [18] PanopticDepth [1] proposes to unify depth estimation and panoptic segmentation by dynamic convolution. However, this method designs multiple position branches to explicitly predict the positions of things and stuff. Thus, complex kernel selection and grouping strategies must be applied. PolyphonicFormer [6] represents things and stuff with queries, and the different vision tasks are realized in the same manner. However, the query features must be refined by multi-stage Transformer-based modules, which make the model extremely heavy. Differently, our method extracts the query features with convolution networks and can achieve the best trade-off between speed and accuracy compared with these methods.

III. EfficientDPS

As shown in Fig. 2, EfficientDPS consists of a feature extractor to extract a contextual feature map, a query network to extract query features for things/stuff, and a task network for classification, segmentation, and depth estimation.

A. Feature Extractor

The feature extractor consists of a backbone and a feature fusion module. We use ResNet [19] and Swin Transformer [20] to extract the multi-level features ($C3$, $C4$, $C5$). Then, PPM [21] is utilized to extract the global contextual information from the last-layer feature map $C5$. Then, FPN [22] is used to generate feature pyramids from $C3$, $C4$, $C5$. Finally, the feature pyramids are fused together by interpolation and concatenation to generate the contextual feature map. In addition, we concatenate the normalized coordinates to the contextual feature map to generate the localization-aware contextual feature map similar to CoordConv [23].

B. Task-decoupled Group Query Network

In this work, queries are adopted to unify the representation of things/stuff and the realization of different vision tasks. To generate representative queries efficiently, we propose the task-decoupled group query network based on the convolutional network as shown in Fig. 2 and Fig. 3.

Task-decoupled. Query map module is utilized for generating query maps. In order to relieve the competition of gradients between recognition and depth estimation tasks during training, task-decoupled query maps are extracted. Specifically, we utilize two branches of stacked convolution layers to predict the semantic query map and depth query map respectively. Query activation maps are used to represent the positions of things and stuff that are highly correlated with the semantics. Thus, on the semantic query map, a convolution layer followed by a sigmoid layer is used to predict the query activation maps for different queries. Finally, the query activation maps adaptively gather the semantic query map and depth query map by inner product to generate the semantic query features and depth query features. As a result, each thing or stuff in the image is represented by a semantic query and a depth query which contain the semantic information and 3D geometry information respectively.

Group Query Representation. As indicated in the previous object detection models such as FCOS [24], ATSS [25] and Group DETR [26], one-to-many label assignment

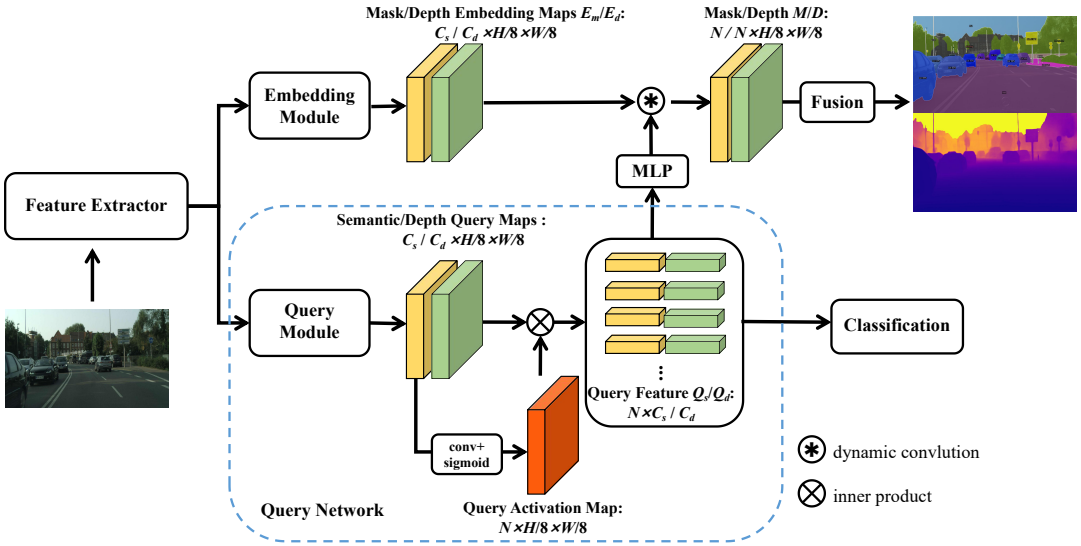

Fig. 2. The architecture of EfficientDPS. H and W are the height and width of the input image respectively. N represents the number of queries. C_s and C_d are the channel number of the semantic query map and depth query map respectively.

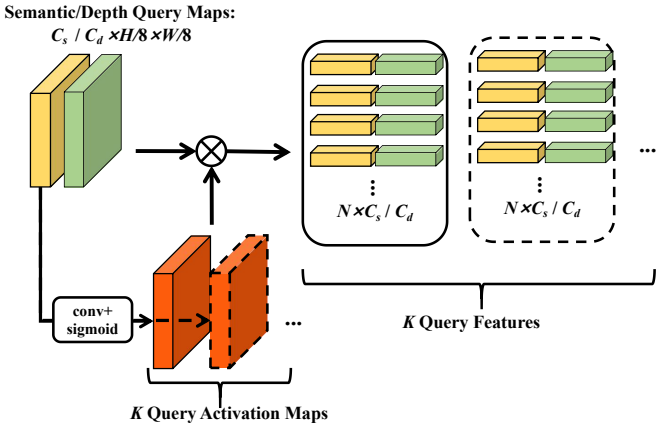

Fig. 3. Group Query Representation. The modules with the dashed lines are only used during training time.

is an important way to improve the performance of object detection by enhancing the supervision signals from target boxes. Inspired by this idea, we propose group query representation as shown in Figure 3. Specifically, during training, we attach multiple convolutional branches to predict multiple groups of query activation maps. Then, multiple groups of semantic/depth query features are generated by inner product between the query activation maps and semantic/depth query maps. The group query representation is only used during training and only one group of query features is kept during inference, which will not hurt the inference speed.

C. Task Network

Based on the query representation, we can use a unified framework to realize different vision tasks.

Classification. Accepting the semantic query features Q_s as input, the classification head implemented with MLP predicts the classification score $P \in R^{N*C_{cls}}$ for the corresponding thing or stuff as shown in Eq 1. f_{cls} represents MLP for classification.

$$P = \text{sigmoid}(f_{cls}(Q_s)) \tag{1}$$

Embedding Module. On the localization-aware contextual feature map, the embedding module extracts the embedding map for segmentation and depth estimation. For decoupling the learning of segmentation and depth estimation tasks, we attach two branches of stacked convolution layers to predict mask embedding map $E_m \in R^{C_s*H/8*W/8}$ and depth embedding map $E_d \in R^{C_d*H/8*W/8}$ respectively. The mask embedding map encodes the semantic information for each pixel while the depth embedding map encodes the geometry information for each pixel.

Segmentation and Depth Estimation. For the segmentation task, as shown in Eq 2, the semantic query features Q_s are refined by MLP to generate the mask kernels and then dynamic convolution followed by sigmoid function between mask kernels and mask embedding map is performed to generate the segmentation $M \in R^{N*H/8*W/8}$ for all the queries. For the depth estimation task, as shown in Eq 3, accepting the depth query features Q_d as input, MLP f_{depth} generates the depth kernels and then dynamic convolution followed by sigmoid function between depth kernels and depth embedding map is performed to predict the normalized depth. Finally, the normalized depth is multiplied with D_{max} to recover the absolute depth $D \in R^{N*H*W}$.

$$M = \text{sigmoid}(f_{seg}(Q_s) * E_m) \tag{2}$$

$$D = \text{sigmoid}(f_{depth}(Q_d) * E_d) \cdot D_{max} \tag{3}$$

Fusion. During inference, argmax() along the query dimension is applied to segmentation masks to generate the

non-overlapped panoptic segmentation. For dense depth prediction, the predicted masks after the softmax operation are multiplied with query-wise depth maps and then summation along the query dimension is performed to generate the final dense depth map as shown in Eq 4.

$$D_{dense}(x,y) = \sum_{i=1}^{N} \frac{exp(M_i) \cdot D_i(x,y)}{\sum_{j=1}^{N} exp(M_j(x,y))} \tag{4}$$

D. Optimization

Label Assignment. During training, EfficientDPS outputs a group fixed number of predictions $Z = (P_i, M_i, D_i)_{i=1}^{N}$, where $P_i \in R^{C_{cls}}$ represents the probability distribution over C_{cls} categories, $M_i \in \{0,1\}^{H \times W}$ is the binary mask, and $D_i \in [0, D_{max}]^{H \times W}$ is the estimated depth map. Assuming the target sets are represented by $Z^{gt} = (C_i^{gt}, M_i^{gt}, D_i^{gt})_{i=1}^{N^{gt}}$ where $C_i^{gt} \in \{1, ..., C_{cls}\}$ is the class label, $M_i^{gt} \in \{0,1\}^{H \times W}$ is the binary mask, and $D_i^{gt} \in [0, D_{max}]^{H \times W}$ is the target depth map. To train the model, the matching between the prediction sets Z and target sets Z^{gt} is desired.

Inspired by DETR [5], the bipartite matching is adopted. The matching score between each pair of prediction and target is computed based on the multiplication of classification score and dice score [27] as shown in Eq 5. The optimal matching is computed by Hungarian algorithm [8].

$$C(i,j) = P_i(C_j^{gt})^{\alpha} \cdot \text{dice}(M_i, M_j^{gt})^{1-\alpha} \tag{5}$$

During training, group query representation as shown in Figure 3 generates multiple groups of query features so as to output multiple groups of predictions. The optimal matching between each group of prediction set and target set is computed respectively. The individual matching for each prediction group is important for the model's performance which will be demonstrated in the following experiments.

Loss Function. Given the optimal matching between the prediction set and target set, we adopt the multi-task losses to train the model as shown in Eq 6. λ_{cls}, λ_{mask} and λ_{depth} are the loss weights. Focal loss [28] is adopted as the classification loss L_{cls}. Due to the imbalance between the foreground and background region, L_{mask} is the linear combination of binary cross-entropy loss and dice loss [27]. For the depth loss, we adopt the combination of scale-invariant logarithmic error [29] and relative squared error [30] as shown in Eq 7.

$$\mathcal{L} = \lambda_{cls} \cdot L_{cls} + \lambda_{mask} \cdot L_{mask} + \lambda_{depth} \cdot L_{depth} \tag{6}$$

Given the predicted depth value d and the depth label d^{gt}, the loss function of the depth branch L_{depth} is as follows:

$$L_{depth} = \lambda_{log}\mathbb{D}(\log \frac{d}{d^{gt}}) + \lambda_{rel}\mathbb{E}(\|1 - \frac{d}{d^{gt}}\|_2) \tag{7}$$

where $\mathbb{D}(\cdot)$ and $\mathbb{E}(\cdot)$ refer to variance and expectation respectively, which aim to minimize the distribution distance between the d^{gt} and d.

IV. Experiments

Datasets. Our model is evaluated on Cityscapes [31], Cityscapes-DPS [2] and SemKITTI-DPS[2]. Cityscapes is a challenging dataset for image segmentation. It consists of 2975, 500 and 1425 images with a resolution of 1024×2048 for training, validation, and testing respectively. In Cityscapes-DPS, there are 2400 and 300 images annotated for training and validation respectively. The depth annotations are computed based on the disparity map of stereo images. The SemKITTI-DPS contains 19130, 4071 and 4342 images for training, validation, and testing respectively. The annotations of panoptic segmentation and depth are sparse.

Evaluation Metrics. For panoptic segmentation, Panoptic Quality (PQ) [37] is proposed to evaluate the models' performance. PQ is defined as:

$$\text{PQ} = \text{SQ} \cdot \text{RQ} = \frac{\sum_{(M,M^{gt}) \in TP} \text{IoU}\left(M, M^{gt}\right)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \tag{8}$$

where M, M^{gt} represent the predicted and target masks respectively. $|TP|$, $|FP|$, and $|FN|$ represent the number of matched pairs of predicted and target masks ($\text{IoU}\left(M, M^{gt}\right) > 0.5$), unmatched predicted masks and unmatched target masks respectively. SQ and RQ represent segmentation quality and recognition quality respectively. For depth-aware panoptic segmentation, Vip-DeepLab [2] proposes DPQ to evaluate segmentation and depth estimation simultaneously. DPQ^{λ} is defined as:

$$\text{DPQ}^{\lambda}\left(M, M^{gt}\right) = \text{PQ}\left(M^{\lambda}, M^{gt}\right) \tag{9}$$

where $M^{\lambda} \in \{0,1\}^{H \times W}$ is defined as: $M^{\lambda} = M \cap \mathbb{I}(|d(x,y) - d^{gt}(x,y)| \leq \lambda d^{gt}(x,y))$ and $\mathbb{I}$ is the Indicative function. DPQ is the average of DPQ^{λ} with $\lambda \in \{0.1, 0.25, 0.5\}$. The inference time is measured on RTX 2080ti if not specified.

A. Implementation Details.

EfficientDPS is built upon PyTorch [38] and Detectron2 [39]. ResNet50 [19] is used as the backbone by default. We first train panoptic segmentation on Cityscapes and then finetune depth-aware panoptic segmentation on Cityscapes-DPS and SemKITTI-DPS. The training details are as follows.

Training Details of PS Model. We train the panoptic model with three steps. We first pre-train the model on MS COCO [40] with 64 images per mini-batch. AdamW optimizer [41] with weight decay 0.05 is adopted. The models are trained for 50 epochs. The initial learning rate is set 5e-5 and divided by 10 at 40 epochs and 46 epochs respectively. Random flipping and scale jitter are adopted. Then, we fine-tune the model on Cityscapes with 32 images per mini-batch. The images are resized with random factors in [0.5, 2.0] and then randomly cropped into 512×1024. Color augmentation [42] and horizontal flipping are adopted. Thirdly, we fine-tune the model for 10k iterations with images randomly scaled by [1.0, 1.5] and then cropped into 1024×2048.

TABLE I

EXPERIMENTAL RESULTS ON CITYSCAPES VALIDATION SET. '*' MEANS MODELS ARE PRETRAINED ON MAPILLARY VISTAS. 'P' MEANS THE NUMBER OF PARAMETERS. 'F' MEANS FLOATING POINT OPERATIONS.

Method	Backbone	PQ	SQ	RQ	PQ_{th}	PQ_{st}	P/M	F/G	GPU	Time/ms
UPSNet [32]	R50	59.3	79.7	73.0	54.6	62.7	45	487	V100	140
Seamless [33]	R50	59.8	-	-	54.6	63.6	51	514	V100	150
UPSNet [32]	R50	60.5	80.9	73.5	57.0	63.0	45	487	RTX	202
RealPS [34]	R50	58.8	-	-	52.1	63.7	-	-	V100	99
LPSNet [35]	R50	59.7	79.9	73.6	54.0	63.9	-	-	-	130
Panoptic-DeepLab [10]	R50	59.7	-	-	-	-	-	398	V100	117
Panoptic-DeepLab [10]	Xcp-71	63.0	-	-	-	-	46	547	V100	175
EfficientPS [36]	EfficientNet-B5	63.9	81.5	77.1	60.7	66.2	40	433	RTX	166
EfficientPS* [36]	EfficientNet-B5	66.1	82.5	78.9	62.7	68.5	40	433	RTX	166
PanopticDepth [1]	R50	64.1	82.5	76.9	58.8	68.0	38	570	2080ti	258
EfficientDPS	R50	64.4	82.3	77.3	57.0	69.7	33	394	2080ti	121
EfficientDPS*	R50	66.3	83.4	78.2	58.9	71.8	33	394	2080ti	121

TABLE II

EXPERIMENTAL RESULTS ON CITYSCAPES-DPS VALIDATION SET. '*' MEANS MODELS ARE PRE-TRAINED ON MAPILLARY VISTAS. 'P' MEANS THE NUMBER OF PARAMETERS. 'F' MEANS FLOATING POINT OPERATIONS.

Method	$DPQ_{0.5}$	$DPQ_{0.25}$	$DPQ_{0.1}$	DPQ / DPQ_{th} / DPQ_{stuff}	P/M	F/G	Time/ms
ViP-DeepLab-WR41* [2]	68.7	66.5	50.5	61.9 / 55.9 / 66.3	171	-	718
PanopticDepth-R50 [1]	66.7	63.1	45.0	58.3 / 52.1 / 62.8	42	610	400
PanopticDepth-SwinT [1]	66.5	64.1	48.6	59.7 / 55.2 / 63.0	46	641	442
PanopticDepth-SwinS [1]	67.4	65.0	48.8	60.4 / 56.0 / 63.6	67	829	499
PolyphonicFormer-SwinB* [6]	70.6	67.8	50.2	62.9 / 55.8 / 68.0	106	1675	1758
EfficientDPS-R50	66.8	64.1	47.8	59.6 / 51.8 / 65.2	37	547	148
EfficientDPS-R50*	67.5	64.8	48.2	60.2 / 52.5 / 66.1	37	547	148
EfficientDPS-SwinB	68.7	65.6	49.5	61.3 / 53.8 / 67.0	99	1062	343
EfficientDPS-SwinB*	69.4	66.1	50.3	62.0 / 54.5 / 68.5	99	1062	343

TABLE III

EXPERIMENTAL RESULTS ON SEMKITTI-DPS VALIDATION SET. '*' MEANS MODELS ARE PRE-TRAINED ON MAPILLARY VISTAS.

Method	DPQ	DPQ_{th}	DPQ_{stuff}	Time/ms
ViP-DeepLab-WR41* [2]	48.9	42.0	53.9	172.7
PanopticDepth-R50 [1]	46.9	46.0	47.6	105.7
PolyphonicFormer-SwinB* [6]	52.2	50.1	53.8	420.5
EfficientDPS-SwinB*	51.0	49.1	52.0	85.2

Training Details of DPS Model. EfficientDPS is trained on Cityscapes-DPS for 10k iterations. We randomly resize the images with scale factors $s \in [0.8, 1.2]$ and the corresponding depth annotations are scaled by $1/s$. Then, the images are center-cropped into 1024×2048. Color augmentation, horizontal flipping, and rotation are adopted. λ_{rel}, λ_{log}, λ_{bce}, λ_{dice} and λ_{cls} are set to 1.0, 5.0, 5.0, 2.0, 2.0 respectively. On SemKITTI-DPS, the images are cropped into 384×1280 at both pretraining and finetuning stages. Other settings remain the same.

B. Main Results

Panoptic Segmentation. On Cityscapes datasets, as shown in Table I, with only 33.25 M parameters and 394.70 GFLOPs, our method achieves PQ of 64.4%. With pre-trained on Mapillary Vistas, our model can achieve PQ of 66.3%. The inference time per image with a resolution of 1024*2048 is only 121 ms. In contrast, Panoptic-DeepLab [10], EfficientPS [36], and PanopticDepth [1] achieve inferior performance with inference time of 175 ms, 166ms, and 258 ms respectively, which are greatly slower than our method.

Depth-aware Panoptic Segmentation. On Cityscapes-DPS, as shown in Table II, our model with Resnet50 achieves 59.6% average DPQ with only 36.5 M parameters and 547.0 GFLOPs. With pre-trained on Mapillary Vista, our model can achieve 60.2% average DPQ. The inference speed is 148 ms per frame with a resolution of 1024*2048. In contrast, the recently proposed PanopticDepth [1] with Resnet50 achieves an average DPQ of 58.3% with an inference time of 400ms per frame, which is 2.7 times our method. Compared with ViP-DeepLab [2] that costs 718 ms per frame for inference, our model with Swin-B [20] achieves slightly better DPQ of 62.0% with only 343 ms per frame. PolyphonicFormer [6] with Swin-B achieves higher accuracy than our model with Swin-B, but the inference time per frame is 1758 ms which is 5.1 times our model's inference time. As shown in Fig. 1, compared with all the other models, our method achieves the best trade-off between speed and accuracy. As shown in Table III, on SemKITTI-DPS, our model achieves DPQ of 51.0% with speed of 85.2 ms per frame, leading to the same conclusions as that on cityscapes-DPS.

C. Ablation Studies

Group Query Representation. As shown in Table IV, when the group number of query activation maps increases from 1 to 8, the one-to-many label assignment strategy can improve the performance of PQ by 2.1%, PQ_{th} by 2.6%, PQ_{st} by 1.4%. This demonstrates that group query representation

TABLE IV

GROUP QUERY REPRESENTATION. ALL THE MODELS ARE EVALUATED ON COCO VALIDATION DATASETS. 'NUMBER' MEANS THE NUMBER OF GROUPS OF QUERY ACTIVATION MAPS. '*' MEANS ALL THE GROUPS ADOPT THE SAME BIPARTITE MATCHING RESULTS.

Number	Epoch	PQ	SQ	RQ	PQ_{th}	PQ_{st}
1	145	42.6	80.0	52.1	46.8	37.1
1	12	38.5	79.4	47.4	40.9	34.8
4	12	39.4	79.6	48.6	41.8	35.9
6	12	40.1	79.0	49.4	42.9	35.9
8	12	40.6	79.6	49.9	43.5	36.2
10	12	40.1	78.9	49.3	42.6	36.1
8	36	43.3	80.4	52.8	46.9	37.9
8	50	44.1	80.7	53.7	48.0	38.4
8*	50	42.4	79.6	51.8	45.6	37.6

TABLE V

ABLATION STUDIES. 'QDE' MEANS QUERY-WISE DEPTH ESTIMATION. 'TDN' MEANS TASK-DECOUPLED QUERY NETWORK. 'GQR' MEANS GROUP QUERY REPRESENTATION.

QDE	TDN	GQR	DPQ	DPQ_{th}	DPQ_{stuff}
		✓	58.9	50.2	65.2
✓			58.5	49.8	64.9
✓	✓		57.7	47.4	65.2
✓		✓	58.7	49.9	65.1
✓	✓	✓	59.6	51.8	65.2

is important for both thing classes and stuff classes. Compared with the baseline trained for 145 epochs, our method trained with 34% training time (50 epochs) can achieve higher performance and improve PQ by 1.5%, showing that our method can improve the model's convergence speed. When all the prediction groups adopt the same bipartite matching result, the PQ is reduced by 1.7% compared with that different prediction groups match with the targets individually. This demonstrates the matching noises introduced by the individual matching process between different query groups are beneficial for the model's training.

Task-decoupled Query Network. As shown in Table V, compared with the model that has query-wise depth estimation and group query representation, adding the task-decoupled design can improve DPQ by 0.9% demonstrating the task-decoupled query network can improve the model's performance by relieving the competition of gradients between panoptic segmentation and depth estimation. In addition, compared with the model with only query-wise depth estimation, adding task-decoupled design decreases DPQ from 58.5% to 57.7%, but adding group query representation can largely improve DPQ from 57.7% to 59.6%. This demonstrates that group query representation is important for learning a good task-decoupled query network.

Query-wise Depth Estimation. As shown in Table V, compared with image-wise depth estimation, query-wise depth estimation can achieve 59.6% DPQ and 51.8% DPQ_{th}, which is 0.7% and 1.6% higher. The DPQ_{stuff} is not improved. This shows query-wise depth estimation is beneficial to the depth-aware panoptic segmentation of thing classes.

Visualization. Fig 4 shows the visualization of the query activation map and the corresponding segmentation mask. For thing classes, the query activation map activates in the central region like a Gaussian map. For stuff classes, the distribution of the activated positions is dispersed, due to that the stuff classes don't need to distinguish different instances. These observations demonstrate the query network has learned to locate the discriminative positions for things and stuff based on a single convolution layer, which is more efficient than Transformer-based models. Fig. 5 shows the depth-aware panoptic segmentation from EfficientDPS.

Fig. 4. Visualization of Query Activation Map. The first and third rows show the query activation maps while the second and fourth rows show the corresponding masks.

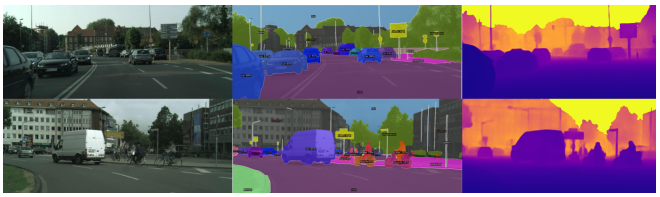

Fig. 5. Prediction Visualization of EfficientDPS on Cityscapes-dps val.

V. CONCLUSION

This paper proposes an efficient and end-to-end depth-aware panoptic segmentation method EfficientDPS based on convolutional networks. In this method, the query network based on convolution is proposed to efficiently extract queries to unify the representation of things and stuff. Based on query representation, the classification, segmentation, and depth estimation tasks are unified in an efficient manner. In addition, group query representation is adopted to accelerate convergence and improve performance with no harm to inference speed. Extensive experiments on Cityscapes-DPS and SemKITTI-DPS demonstrate that with a simple, deployment-friendly model architecture, our method can achieve the best trade-off between speed and accuracy, making it easily applied to resource-limited devices such as robots.

REFERENCES

[1] N. Gao, F. He, J. Jia, Y. Shan, H. Zhang, X. Zhao, and K. Huang, "Panopticdepth: A unified framework for depth-aware panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1632–1642.

[2] S. Qiao, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3997–4008.

[3] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6399–6408.

[4] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[5] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[6] H. Yuan, X. Li, Y. Yang, G. Cheng, J. Zhang, Y. Tong, L. Zhang, and D. Tao, "Polyphonicformer: unified query learning for depth-aware video panoptic segmentation," in *European Conference on Computer Vision*. Springer, 2022, pp. 582–599.

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

[8] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2325–2333.

[9] R. Hou, J. Li, A. Bhargava, A. Raventos, V. Guizilini, C. Fang, J. Lynch, and A. Gaidon, "Real-time panoptic segmentation from dense detections," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[10] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, and L.-C. Chen, "Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 475–12 485.

[11] 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.

[12] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Max-deeplab: End-to-end panoptic segmentation with mask transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5463–5474.

[13] W. Zhang, J. Pang, K. Chen, and C. C. Loy, "K-net: Towards unified image segmentation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 10 326–10 338, 2021.

[14] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 864–17 875, 2021.

[15] 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, pp. 4433–4442.

[16] V. Guizilini, R. Hou, J. Li, R. Ambrus, and A. Gaidon, "Semantically-guided representation learning for self-supervised monocular depth," *arXiv preprint arXiv:2002.12319*, 2020.

[17] L. Wang, J. Zhang, O. Wang, Z. Lin, and H. Lu, "Sdc-depth: Semantic divide-and-conquer network for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 541–550.

[18] Y. Li, H. Zhao, X. Qi, L. Wang, Z. Li, J. Sun, and J. Jia, "Fully convolutional networks for panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 214–223.

[19] 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.

[20] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[21] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[22] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[23] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," *Advances in neural information processing systems*, vol. 31, 2018.

[24] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9627–9636.

[25] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9759–9768.

[26] Q. Chen, X. Chen, J. Wang, H. Feng, J. Han, E. Ding, G. Zeng, and J. Wang, "Group detr: Fast detr training with group-wise one-to-many assignment," *arXiv preprint arXiv:2207.13085*, vol. 1, no. 2, 2022.

[27] 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.

[28] 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.

[29] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[31] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[32] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun, "Upsnet: A unified panoptic segmentation network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8818–8826.

[33] L. Porzi, S. R. Bulo, A. Colovic, and P. Kontschieder, "Seamless scene segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8277–8286.

[34] R. Hou, J. Li, A. Bhargava, A. Raventos, V. Guizilini, C. Fang, J. Lynch, and A. Gaidon, "Real-time panoptic segmentation from dense detections," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8523–8532.

[35] W. Hong, Q. Guo, W. Zhang, J. Chen, and W. Chu, "Lpsnet: A lightweight solution for fast panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 746–16 754.

[36] R. Mohan and A. Valada, "Efficientps: Efficient panoptic segmentation," *International Journal of Computer Vision*, vol. 129, no. 5, pp. 1551–1579, 2021.

[37] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9404–9413.

[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[39] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[40] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[41] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[42] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.