PolyphonicFormer: Unified Query Learning for Depth-aware Video Panoptic Segmentation

Xiangtai Li[1*], Haobo Yuan[2*], Yibo Yang[3], Lefei Zhang[2], Yunhai Tong[1], Dacheng Tao[3],

[1] Key Laboratory of Machine Perception (MOE), Peking University

[2] School of Computer Science, Wuhan University [3] JD Explore Academy

Abstract

Depth-aware Video Panoptic Segmentation (DVPS) aims to predict panoptic segmentation results and depth maps in a video which is a challenging scene understanding problem. In this report, we present PolyphonicFormer, a vision transformer to unify all the sub-tasks for DVPS task. Our method explores the relation between depth prediction and panoptic segmentation via query based learning. In particular, we design three different queries including thing query, stuff query and depth query. Then we propose to learn the correlation among these queries via dynamic convolution. From the experiments, we prove the benefits of our design from both depth and panoptic segmentation aspects. Since each thing query also encodes the instance-wise information, it is natural to perform tracking via cropping instance mask features with appearance learning. Our method outperforms the state-of-the-art method, ViP-Deeplab. Ablation studies are reported to show how we improve the performance. No external data is used.

1. Introduction

Depth-aware Video Panoptic Segmentation (DVPS) [13] is a challenging computer vision task which extends the video panoptic segmentation [8, 18] with monocular depth estimation. Achieving accurate and robust DVPS methods in real world scenarios can greatly promote the development of video analysis applications including auto-driving. Recently, transformers [1,4] make a great process in computer vision. Object queries have been shown to be very effective for detection and segmentation task where they play central roles for modeling instance and stuff in the scene. Furthermore, one core advantage in transformers lies on modeling multi modal modeling. Motivated by these two points, we aim to design a unified model to solve the DVPS via a transformer architecture.

We propose PolyphonicFormer to jointly model the panoptic segmentation and depth via object queries. In particular, we represent thing, stuff and depth as queries. Since depth is not countable, we associate the thing and stuff queries with depth and make it into patched based prediction. In particular, we broadcast thing and stuff queries into initial depth query and then we perform dynamic convolution and self-attention among these queries. For thing and stuff queries, we follow the recent work [20] design to refine thing and stuff queries. Moreover, motivated by recent works [15,20], our model avoids the heavy transformer encoder by interactively refining the object queries in the decoder part. The backbone of our model can be a convolution network or a vision transformer. Our experiment shows that joint depth, thing and stuff queries learning can boost both results of panoptic segmentation and depth estimation on the strong baselines. Thus we name our method as PolyphonicFormer (Polyphonic Transformer) where different queries come from different sources (depth or panoptic) but both can benefit each other which is just like polyphony used in music since late Middle Ages. Our method achieves 63.6 DSTQ on KITTI-DVPS test set and suppress the previous work Vip-Deeplab. To summarize, our contributions are as follows.

- PolyphonicFormer is the first transformer-like architecture with unified query modeling to tackle the DVPS task. It can be trained end-to-end without extra region proposal network for box detection.
- PolyphonicFormer models the depth prediction, panoptic segmentation prediction and tracking using the object query and link both semantics in panoptic segmetnation and geometric in depth with query interaction.
- PolyphonicFormer outperforms previous ViP-Deeplab on KITTI-DVPS without the extra posting for tracking.

2. Method

In this section, we explain the architecture design of PolyphonicFormer in detail. Fig. 1 gives an overall illustration of the proposed method. Our PolyphonicFormer contains three parts: (1) an encoder network to extract features

*Indicates Equal Contribution. Emails: lxtpku@pku.edu.cn; yuanhaobo@whu.edu.cn

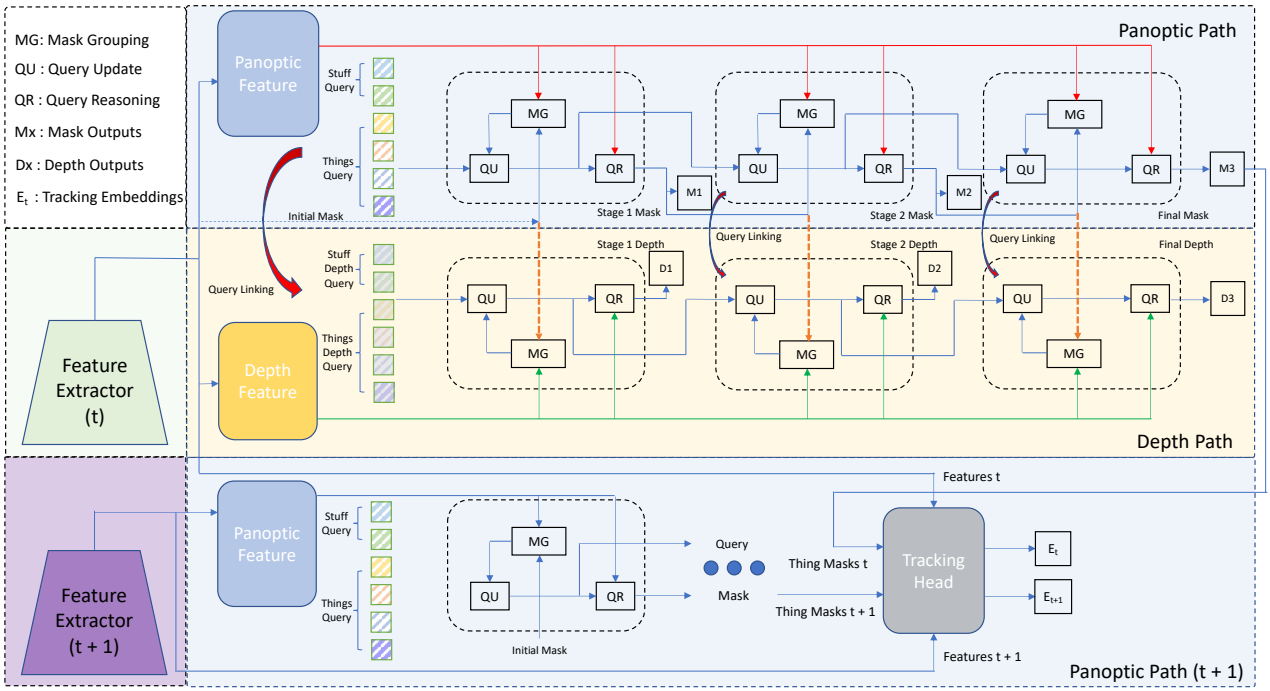

Figure 1. An illustration of our proposed PolyphonicFormer. Our method contains three parts: (1) Encoder Network as Feature Extractor to obtain two parallel features for depth prediction and panoptic segmentation(bottom left). (2) Decoder Network to refine the all the queries at the same time (middle). (3) Tracking Head to learn the feature embedding among the frames (bottom).

for each frames. (2) a cascaded decoder which takes three different types of queries and backbone features as inputs and outputs panoptic segmentation results and depth maps. It contains two paths: depth path and panoptic path. (3) a tracking head with several convolution layers to learn the instance-wised tracking embeddings for each thing query.

Encoder Network This part extract image features for each input image. It contains a backbone network(Convolution Network [7] or Swin Transformer [10]) with Feature Pyramid Network as neck. We adopt semantic FPN design to fuse features in a top-down manner. In particular, we adopt two parallel fusion to obtain two features for depth (X_d) and panoptic segmentation (X_p) respectively.

Thing, Stuff and Depth as Queries Beyond previous works [1, 3], our model takes thing, stuff and depth as input queries to directly obtain the final panoptic segmentation and depth maps. Following previous works [15, 20], the initial weights of these queries are directly copied from the first stage weights of initial heads. For things and stuff mask predictions, we use 1×1 convolution layer to obtains the outputs of both things and stuff masks. For initial depth prediction, we also use 1×1 convolution layer to prediction dense depth maps. All these predictions are directly supervised with corresponding ground truth. Using such initial heads can avoid heavier transformer encoder layers and save the time for learning correspondence between object queries and input features. Since depth prediction is not countable, we broadcast both thing query and stuff query into the initial depth query via sum operation. Then we obtain the same number of depth queries with both thing and stuff queries. We name this process Query Linking (QL).

Joint Panoptic and Depth Interaction The decoder takes previous mask predictions, previous object queries and shared features as inputs and outputs current refined mask predictions, refined object queries and refined depth predictions. As shown in Fig. 1, it contains two path: panoptic path and depth path where depth queries are obtained directly from the thing and stuff queries for each interaction (shown in red arrow in Fig. 1). In particular, for both things and stuff queries, following previous work [20], we first obtained feature queries via grouping between previous mask predictions and input features (X_p) where we term this process as MG (Mask Groping). Then we perform dynamic convolution to refine input thing and stuff queries with such learned feature queries. In particular, we adopt the same design [20] by learning the gating function to update the refine queries. We term this process Query Update (QU). After that, we adopt self attention layers to learn the correspondence among each queries to obtain the updated queries. We name this operation Query Reasoning (QR)

to get the full correlation among queries. Finally refined masks are obtained via dot product between refined queries and X_p which will be used for the next stage input. For depth queries, we adopt the same strategies since the number of depth queries are same with thing and stuff queries. We obtained the refined depth queries and depth prediction which have the same shapes as the thing stuff queries and mask predictions. The main differences are two points: (1) We obtain query level depth prediction via thing and stuff masks from panoptic path. (2) The stuff depth query and things depth query is refined according to thing and stuff masks from panoptic path which means we obtain depth query features via shared mask predictions. This is shown in Fig 1(arrows with orange color). Since our method operates on query level and this can avoid heavier computation cost on feature level compared with origin detection vision transformer [1, 3].

Tracking with Thing Query For tracking part, we adopt previous work design [11] and add a tracking head, where we directly learn appearance embedding among the different frames. During training, we first match predicted thing masks from panoptic path to ground truth thing mask according to mask iou. To improve location cues of masks, we adopt corresponding ground truth boxes as supervision signals to train these tracking embeddings. During the inference, we use the thing masks which are generated from thing query to obtain final tracking embedding for tracking which is shown in bottom part of Fig. 1.

Loss Function All the outputs are encoded via queries, we need to assign ground truth according to cost. In particular, we mainly follow the design of MaskFormer and MaxDeeplab [3, 16]. We use the classification and mask based cost. For depth prediction, we use mask based bipartite matching to assign depth ground truth for each query. Following the previous works [5, 13], we adopt absolute relative loss and square relative loss. For panoptic segmentation, we adopt focal loss to learn the mask prediction and label classification. For tracking heads, we adopt the loss proposed in [11]. All the weights are set to 1 by default.

Inference We directly get the outputs of panoptic masks and depth maps from the corresponding queries according to their sorted scores. For final panoptic segmentation results, we adopt the method used in Panoptic-FPN [9] to merge panoptic mask. For final depth prediction, since each depth query encodes the instance-wised information, we first filter each depth map via corresponding mask prediction and then merge the such depth prediction into the initial depth map. For tracking, we mainly use the previous tracker [11]. We first obtain learned tracking feature embedding via final thing mask prediction. We adopt ROI Align [6] to crop image features directly from the backbone features. To assign each thing mask a id, we calculate the similarities among the learned embeddings and assign track id in a online manner.

Method	Backbone	PQ	PQ_{th}	PQ_{stuff}	$abs\ rel$
Panoptic baseline	ResNet-50	61.8	55.2	66.5	-
Depth baseline	ResNet-50	-	-	-	0.105
Our	ResNet-50	63.9	57.5	68.6	0.089
Panoptic baseline	Swin-base	66.0	60.5	71.3	-
Depth baseline	Swin-base	-	-	-	0.086
Our	Swin-base	67.8	61.0	72.4	0.072
Our+RFP [12]	Swin-base	68.8	62.0	73.1	0.069

Table 1. Ablation Studies on image baseline on Cityscapes-VPS validation set.

Method	backbone	$DSTQ$	AQ
PolyphonicFormer + DeepSort	Swin-b	51.8	25.9
PolyphonicFormer + Unitrack	Swin-b	49.3	22.5
PolyphonicFormer + QuaniDenseTrack	Swin-b	63.6	46.2

Table 2. Ablation Studies on tracking part using KITTI-DVPS.

Method	backbone	$DSTQ$
Vip-Deeplab	WiderResNet-41	63.3
rl-lab	unknown	54.8
yang26	unknown	55.6
PolyphonicFormer(HarborY)	Swin-b	**63.6**

Table 3. Performance Comparison on ICCV-2021 KITTI-DVPS challenge.

3. Experiment

Experimental Setup We carry out ablation studies on Cityscapes-DVPS datasets for evaluating panoptic segmentation results and depth results. Then we verify the tracking parts of our model on the KITTI-DVPS test set. We implement our models with Pytorch. We follow the same training settings from Panoptic Deeplab where we first pretrain our model on both Mapillary and Cityscapes datasets for Panoptic Segmentation and then we fine-tune the model with our depth query on Cityscapes-DVPS and KITTI-DVPS. During pretraining, we randomly resize the origin images with scale from 1.0 to 2.0 and then we perform random crops with size 1024×2048. We pretrain our model on Mapillary dataset by 240 epochs and Cityscape dataset by 64 epochs. The batch size is set to 16 and we adopt the synchronized batch normalization during the training. For video training settings on KITTI-DVPS, we use full image inputs and randomly sample one nearly frame to learn the tracking embeddings. All the models use the *single scale inference*. For image models, we report Panoptic Quality (PQ) and absolute relative depth errors (abs rel). For video models, we report Depth-aware Segmentation and Tracking Quality (DSTQ) on KITTI-DVPS [18]. We also report Association Quality (AQ) to evaluate tracking performance.

Ablation Studies Following previous work [13], our im-

age baseline model is firstly pretrained on Mapillary and datasets. Since there are no depth annotation, we only train our panoptic path of our model. Then we apply both panoptic path and depth path to train the image baseline model on Cityscape-DVPS. We adopt two different backbone models including ResNet-50 and Swin-base. For depth baseline, we adopt dense prediction-like model [14] as baseline. As shown in Tab. 1, compared with strong baseline, our methods get consistent gains(about 2.0 PQ gains and 0.015 abs rel error drops) among different backbone network. In Tab. 2, we explore three different tracking methods including DeepSort [19], Unitrack [17] and Quansidense [11]. All these tracking methods are modified into our framework. We found the Quansidense tracker works best which is chosen as the final tracking head. The results show that appearance modeling is more important than motion modeling for tracking. Note that our result only uses the tracking embedding and we believe there are still a large space to improve for tracking which will be our future work.

Submitted Entry to Challenge On the KITTI DVPS-test set, we use the strong backbone to train our model with tracking heads. We finetune our model which is trained from Cityscape-DVPS on KITTI-DVPS train set. As shown in Tab. 3, our outperform previous strong dense prediction baseline [13]. Note that Vip-Deeplab uses more extra engineering tricks including TTA (Test Time Augmentation), AutoAug and Semi-Supervised Learning [2]. We believe our method can still gain the performance by applying these tricks and this will be our future work.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 2, 3

[2] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D Collins, Ekin D Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Naive-student: Leveraging semi-supervised learning in video sequences for urban scene segmentation. In *ECCV*, 2020. 4

[3] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *arXiv*, 2021. 2, 3

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[5] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *NIPS*, 2014. 3

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 3

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[8] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 1

[9] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 3

[10] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 2

[11] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, 2021. 3, 4

[12] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *CVPR*, 2021. 3

[13] Siyuan Qiao, Yukun Zhu, H. Adam, A. Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. *CVPR*, 2021. 1, 3, 4

[14] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021. 4

[15] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, and Ping Luo. SparseR-CNN: End-to-end object detection with learnable proposals. *CVPR*, 2021. 1, 2

[16] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. *arXiv preprint arXiv:2012.00759*, 2020. 3

[17] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip HS Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? *arXiv preprint arXiv:2107.02156*, 2021. 4

[18] M. Weber, J. Xie, M. Collins, Yukun Zhu, P. Voigtlaender, H. Adam, B. Green, A. Geiger, B. Leibe, D. Cremers, Aljosa Osep, L. Leal-Taixé, and Liang-Chieh Chen. Step: Segmenting and tracking every pixel. *ArXiv*, abs/2102.11859, 2021. 1, 3

[19] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017. 4

[20] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. *CoRR*, abs/2106.14855, 2021. 1, 2