Perceiving the Invisible: Proposal-Free Amodal Panoptic Segmentation

Rohit Mohan and Abhinav Valada

Abstract—Amodal panoptic segmentation aims to connect the perception of the world to its cognitive understanding. It entails simultaneously predicting the semantic labels of visible scene regions and the entire shape of traffic participant instances, including regions that may be occluded. In this work, we formulate a proposal-free framework that tackles this task as a multi-label and multi-class problem by first assigning the amodal masks to different layers according to their relative occlusion order and then employing amodal instance regression on each layer independently while learning background semantics. We propose the PAPS architecture that incorporates a shared backbone and an asymmetrical dual-decoder consisting of several modules to facilitate within-scale and cross-scale feature aggregations, bilateral feature propagation between decoders, and integration of global instance-level and local pixel-level occlusion reasoning. Further, we propose the amodal mask refiner that resolves the ambiguity in complex occlusion scenarios by explicitly leveraging the embedding of unoccluded instance masks. Extensive evaluation on the BDD100K-APS and KITTI-360-APS datasets demonstrate that our approach set the new state-of-the-art on both benchmarks.
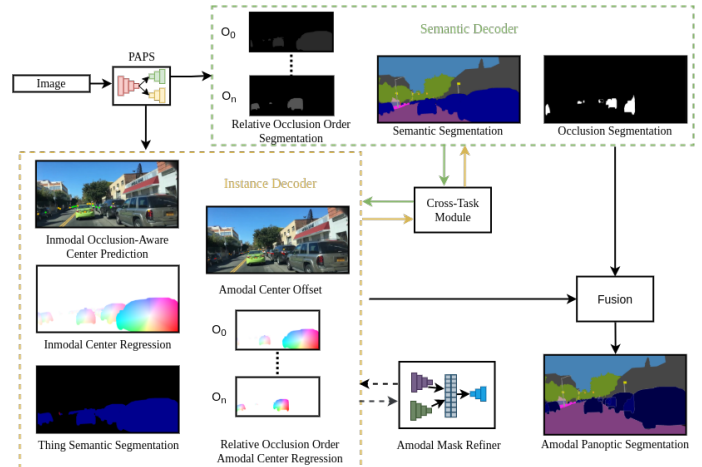
Fig. 1: Overview of our proposed PAPS architecture for amodal panoptic segmentation. Our model predicts multiple outputs from both the semantic and instance decoder. We then fuse the instance-agnostic semantic labels and foreground masks obtained from the segmentation heads with class-agnostic amodal instances that are obtained from the rest of the heads by grouping and majority voting to yield the final amodal panoptic segmentation output.

I. INTRODUCTION

The ability to perceive the entirety of an object irrespective of partial occlusion is known as amodal perception. This ability enables our perceptual and cognitive understanding of the world [1]. The recently introduced amodal panoptic segmentation task [2] seeks to model this ability in robots. The goal of this task is to predict the pixel-wise semantic segmentation labels of the visible amorphous regions of *stuff* classes (e.g., road, vegetation, sky, etc.), and the instance segmentation labels of both the visible and occluded countable object regions of *thing* classes (e.g., cars, trucks, pedestrians, etc.). In this task, each pixel can be assigned more than one class label and instance-ID depending on the visible and occluded regions of objects that it corresponds to, i.e. it allows multi-class and multi-ID predictions. Further, for each segment belonging to a *thing* class, the task requires the knowledge of its visible and occluded regions.

The existing amodal panoptic segmentation approach [2] and baselines [2] follow the proposal-based architectural topology. Proposal-based methods tend to generate overlapping inmodal instance masks as well as multiple semantic predictions for the same pixel, one originating from the instance head and the other from the semantic head, which gives rise to a conflict when fusing the task-specific predictions. This problem is typically tackled using cumbersome heuristics for fusion, requiring multiple sequential processing steps in the pipeline which also tends to favor the amodal instance

segmentation branch. On the other hand, proposal-free methods have been more effective in addressing this problem in the closely related panoptic segmentation task [3]–[5] by directly predicting non-overlapping segments. In this work, we aim to alleviate this problem by introducing the first proposal-free framework called Proposal-free Amodal Panoptic Segmentation (PAPS) architecture to address the task of amodal panoptic segmentation. Importantly, to facilitate multi-class and multi-ID predictions, our PAPS decomposes the amodal masks of objects in a given scene into several layers based on their relative occlusion ordering in addition to conventional instance center regression for visible object regions of the scene referred to as inmodal instance center regression. Hence, the network can focus on learning the non-overlapping segments present within each layer. Fig. 1 illustrates an overview of our approach.

Further, amodal panoptic segmentation approaches tend to predict the amodal masks of *thing* class objects by leveraging occlusion features that are conditioned on features of the visible regions. Although it is effective when objects are only partially occluded, it fails in the presence of heavy occlusion as the area of the visible region is reduced. Motivated by humans whose amodal perception is not only based on visible and occlusion cues but also their experience in the world, we propose the amodal mask refiner module to model this capability using explicit memory. This module first predicts an embedding that represents the unoccluded object regions and correlates it with the amodal features generated using either a proposal-free or proposal-based method to complement the lack of visually conditioned occlusion features. We also demonstrate that our amodal mask refiner can be readily incorporated into a variety

of existing architectures to improve performance.

An interesting aspect of proposal-free methods is that the two sub-tasks, namely, semantic segmentation and instance center regression, are complementary in nature. We leverage this to our benefit and propose a novel cross-task module to bilaterally propagate complementary features between the two sub-tasks decoders for their mutual benefit. Moreover, as rich multi-scale features are important for reliable instance center prediction, we propose the context extractor module that enables within-scale and cross-scale feature aggregation. Finally, to exploit informative occlusion features that play a major role in the amodal mask segmentation quality [2], [6], we incorporate occlusion-aware heads in our PAPS architecture to capture local pixel-wise and global instance-level occlusion information. We present extensive quantitative and qualitative evaluations of PAPS on the challenging BDD100K-APS and KITTI-360-APS datasets, which shows that it achieves state-of-the-art performance. Additionally, we present comprehensive ablation studies to demonstrate the efficacy of our proposed architectural components and we make the models publicly available at http://amodal-panoptic.cs.uni-freiburg.de.

II. Related Work

Although the amodal panoptic segmentation task [2] is relatively new, the inmodal variant called panoptic segmentation has been extensively studied. We first briefly discuss the methods for panoptic segmentation followed by amodal panoptic segmentation.

Panoptic Segmentation: We can categorize existing methods into top-down and bottom-up approaches. Top-down approaches [7]–[10] follow the topology of employing task-specific heads, where the instance segmentation head predicts bounding boxes of objects and its corresponding mask, while the semantic segmentation head outputs the class-wise dense semantic predictions. Subsequently, the outputs of these heads are fused by heuristic-based fusion modules [9], [11]. On the other hand, bottom-up panoptic segmentation methods [4], [5] first perform semantic segmentation, followed by employing different techniques to group [12]–[14] *thing* pixels to obtain instance segmentation. In this work, we follow the aforementioned schema with instance center regression to obtain the panoptic variant of our proposed architecture. Our proposed network modules enrich multi-scale features by enabling feature aggregation from both within-scales and cross-scales. Additionally, our cross-task module facilitates the propagation of complementary features between the different decoders for their mutual benefit.

Amodal Panoptic Segmentation: Mohan *et al.* [2] propose several baselines for amodal panoptic segmentation by replacing the instance segmentation head of EfficientPS [9], a top-down panoptic segmentation network, with several existing amodal instance segmentation approaches. EfficientPS employs a shared backbone comprising of an encoder and the 2-way feature pyramid in conjunction with a Mask R-CNN based instance head and a semantic segmentation head, whose outputs are fused to yield the panoptic segmentation prediction. The simple baseline, Amodal-EfficientPS [2], extends EfficientPS

with an additional amodal mask head and relies implicitly on the network to capture the relationship between the occluder and occludee. ORCNN [15] further extends it with an invisible mask prediction head to explicitly learn the feature propagation from inmodal mask to amodal mask. Subsequently, ASN [6] employs an occlusion classification branch to model global features and uses a multi-level coding block to propagate these features to the individual inmodal and amodal mask prediction heads. More recently, Shape Prior [16] focuses on leveraging shape priors using a memory codebook with an autoencoder to further refine the initial amodal mask predictions. Alternatively, VQ-VAE [17] utilizes shape priors through discrete shape codes by training a vector quantized variational autoencoder. BCNet [18] seeks to decouple occluding and occluded object instances boundaries by employing two overlapping GCN layers to detect the occluding objects and partially occluded object instances. The most recent, APSNet [2] which is the current state-of-the-art top-down approach focuses on explicitly modeling the complex relationships between the occluders and occludees. To do so, APSNet employs three mask heads that specialize in segmenting visible, occluder, and occlusion regions. It then uses a transformation block with spatio-channel attention for capturing the underlying inherent relationship between the three heads before computing the final outputs. In this work, we present the first bottom-up approach that learns the complex relationship between the occluder and occludee by focusing on learning the relative occlusion ordering of objects. We also employ an occlusion-aware head to explicitly incorporate occlusion information and an amodal mask refiner that aims to mimic the ability of humans by leveraging prior knowledge on the physical structure of objects for amodal perception.

III. Methodology

In this section, we first describe our PAPS architecture and then detail each of its constituting components. Fig. 2 illustrates the network which follows the bottom-up topology. It consists of a shared backbone followed by semantic segmentation and amodal instance segmentation decoders. The outputs of the decoders are then fused during inference to yield the amodal panoptic segmentation predictions. PAPS incorporates several novel network modules to effectively capture multi-scale features from within-layers and cross-layers, to enable bilateral feature propagation between the task-specific decoders and exploit local and global occlusion information. Further, it incorporates our amodal mask refiner that embeds unoccluded inmodal instance masks to refine the amodal features.

A. PAPS Architecture

1) Backbone: The backbone is built upon HRNet [19] which specializes in preserving high-resolution information throughout the network. It has four parallel outputs with a scale of $\times 4$, $\times 8$, $\times 16$ and $\times 32$ downsampled with respect to the input, namely, B_4, B_8, B_{16}, and B_{32}, as shown in Fig. 2. We then upsample the feature maps to $\times 4$ and concatenate the representations of all the resolutions resulting in C_4, followed by reducing the channels to 256 with a 1×1 convolution. Lastly, we aggregate multi-scale features by downsampling high-resolution representations to multiple levels and process each level with a 3×3 convolution layer (P_4, P_8, P_{16}, P_{32}).
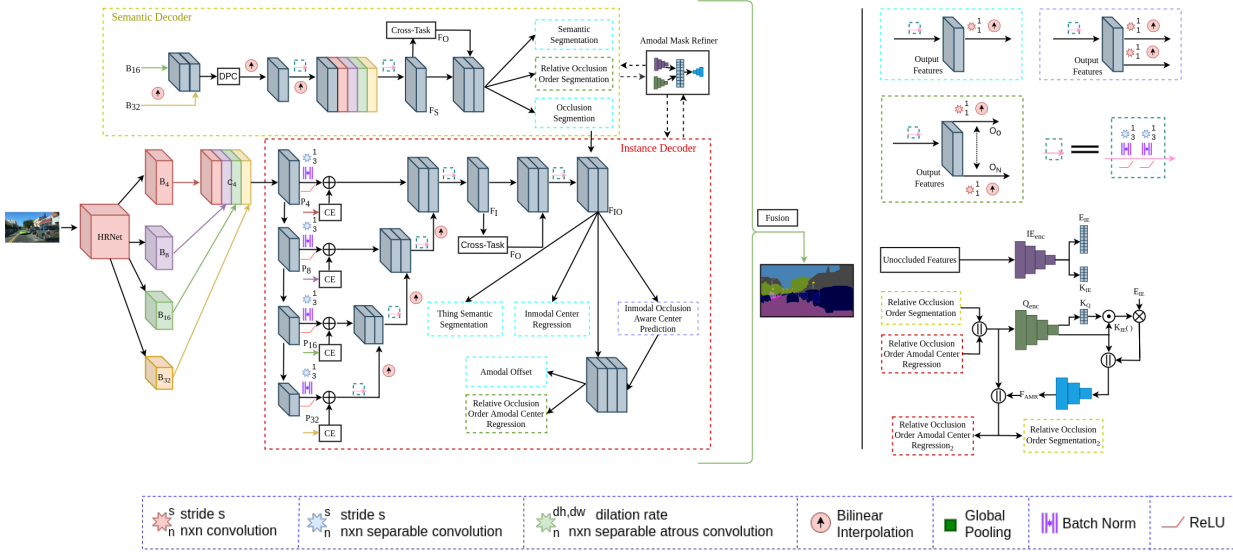
Fig. 2: Illustration of our proposed PAPS architecture consisting of a shared backbone and asymmetric dual-decoder followed by a fusion module that fuses the outputs of the multiple heads of the decoder to yield the amodal panoptic segmentation output. The semantic decoder (yellow-green) and the instance decoder (dark-red) boxes show the topologies of the dual-decoder employed in our architecture. The black-box shows the architecture of our proposed context extractor module. The amodal mask refiner module exploits features from both the decoders to improve amodal masks with embedding correlation.

2) Context Extractor: The multi-scale representations from the backbone are computed over all four scales which we refer to as cross-scale features. The way these cross-scale features are computed (concatenation, reduction, and downsampling) leads to a limited exploration for multi-scale features at a given individual scale resolution. Since rich multi-scale representations are crucial for the instance decoder's performance, we seek to enhance the cross-scale features with within-scale contextual features. To do so, we design a lightweight module called the context extractor which is based on the concept of spatial pyramid pooling and is known for efficiently capturing multi-scale contexts from a fixed resolution. We use the context extractor module at each scale (B_4, B_8, B_{16}, B_{32}) , and add its output to P_4, P_8, P_{16}, and P_{32}, respectively. The proposed context extractor module shown in Fig. S.1 in the supplementary material, employs two 1×1 convolutions, two 3×3 depth-wise atrous separable convolutions with a dilation rate of $(1, 6)$ and $(3, 1)$, respectively, and a global pooling layer. The output of this module consists of 256 channels, where 128 channels are contributed by the 1×1 convolution and four 32 channels come from each of the two 3×3 depth-wise atrous separable convolutions and its globally pooled outputs. We evaluate the benefits of the aforementioned module in the ablation study presented in Sec. IV-D1.

3) Cross-Task Module: The sub-tasks, semantic segmentation and amodal instance center regression, are both distinct recognition problems and yet closely related. The intermediate feature representations of each task-specific decoder can capture complementary features that can assist the other decoder to improve its performance. We propose the cross-task module to enable bilateral feature propagation between the decoders to mutually benefit each other. Given feature inputs F_I and F_S from the two decoders, we fuse them adaptively by employing cross-attention followed by self-attention as

$$F_R = (1 - g_1(F_S)) \cdot F_I + (1 - g_2(F_I)) \cdot F_S, \quad (1)$$

$$F_O = g_3(F_R) \cdot F_R, \quad (2)$$

where $g_1(\cdot)$, $g_2(\cdot)$, and $g_3(\cdot)$ are functions to compute feature confidence score of F_S and F_I. These functions consist of a global pooling layer, followed reducing the channels from 256 to 64 using a 1×1 convolution. Subsequently, we employ another 1×1 convolution with 256 output channels to remap from the lower dimension to a higher dimension and apply a sigmoid activation to obtain the feature confidence scores. F_O is the output of the cross-task module. The cross-attention mechanism in this module enables F_I and F_S to adaptively complement each other, whereas the following self-attention mechanism enables enhancing the highly discriminative complementary features. The ablation study presented in Sec. IV-D1 shows the influence in performance due to this module.

4) Semantic Decoder: The semantic decoder takes B_{32}, B_{16}, C_4 feature maps and the output of cross-task module as its input. First, the B_{32} feature maps are upsampled ($\times 16$) and concatenated with B_{16} and are fed to the dense prediction cell (DPC) [20]. The output of DPC is then upsampled ($\times 8$) and passed through two sequential 3×3 depth-wise separable convolutions. Subsequently, we again upsample ($\times 4$) and concatenate it with C_4. We then employ two sequential 3×3 depth-wise separable convolutions and feed the output (F_S) to the cross-task module. Further, we concatenate F_S with the output of the cross-task module (F_O) and feed it to the multiple heads in the semantic decoder.

We employ three heads, namely, relative occlusion order segmentation (L_{roo}), semantic segmentation (L_{ss}), and occlusion segmentation (L_{os}), towards the end of our semantic decoder. The relative occlusion order segmentation head predicts foreground mask segmentation for O_N layers. The masks of each layer are defined as follows: All unoccluded class-agnostic *thing* object masks belong to layer 0 (O_0). Next, layer 1 (O_1) comprises amodal masks of any occluded object that are occluded by layer 0 objects but not occluded by any other occluded object. Next, layer 2 (O_2) consists of amodal masks of any occluded object, not in the previous layers that

are occluded by layer 1 objects but not occluded by any other occluded objects that are not part of previous layers and so on. Fig. 3 illustrates the separation of *thing* amodal object segments into relative occlusion ordering layers. This separation ensures each *thing* amodal object segment belongs to a unique layer without any overlaps within that layer. We use the binary cross-entropy loss (L_{roo}) to train this head. Next, the semantic segmentation head predicts semantic segmentation of both *stuff* and *thing* classes, and we employ the weighted bootstrapped cross-entropy loss [21] (L_{ss}) for training. Lastly, the occlusion segmentation head predicts whether a pixel is occluded in the context of *thing* objects and we use the binary cross-entropy loss (L_{occ}) for training. The overall semantic decoder loss is given as

$$L_{sem} = L_{ss} + L_{os} + L_{roo}. \tag{3}$$

The predictions from all the heads of the semantic decoder are used in the fusion module to obtain the final amodal panoptic segmentation prediction.

5) Instance Decoder: The instance decoder employs a context encoder at each scale (B_{32}, B_{16}, B_8, B_4) and adds the resulting feature maps to P_{32}, P_{16}, P_8, and P_4, respectively. Then, beginning from ($\times 32$), the decoder repeatedly uses a processing block consisting of two sequential 3×3 depthwise separable convolutions, upsamples it to the next scale ($\times 16$), and concatenates with the existing features of the next scale until $\times 4$ feature resolution is obtained (F_I). The F_I is then fed to the cross-task module. The cross-task output F_O is concatenated with F_I and is processed by two sequential 3×3 depth-wise separable convolutions. Subsequently, the features from the occlusion segmentation head of the semantic decoder are concatenated to incorporate explicit pixel-wise local occlusion information referred to as F_{IO} features.

The instance decoder employs five prediction heads. The inmodal occlusion-aware center prediction head consists of two prediction branches, one for predicting the center of mass heatmap of inmodal *thing* object instances (L_{icp}) and the other for predicting whether the heatmap is occluded (L_{ico}). For the former, we use the Mean Squared Error (MSE) loss (L_{icp}) to minimize the distance between the 2D Gaussian encoded groundtruth heatmaps and the predicted heatmaps, for training. For the latter, we use binary cross-entropy loss (L_{ico}) for training. Following, the *thing* semantic segmentation (L_{tss}) head predicts $N_{thing}+1$ classes, where N_{thing} is the total number of *thing* semantic classes and the '+1' class predicts all *stuff* classes as a single class. This head is trained with the weighted bootstrapped cross-entropy loss [21] (L_{tss}). Next, the inmodal center regression (L_{icr}) head predicts the offset from each pixel location belonging to *thing* classes to its corresponding inmodal object instance mass center. We use the L_1 loss for training this head (L_{icr}). All the aforementioned heads take F_{IO} features as input.

The remaining heads of the instance decoder are referred to as the amodal center offset (L_{aco}) and relative occlusion order amodal center regression (L_{rooacr}). The amodal center offset head predicts the offset from each inmodal object instance center to its corresponding amodal object instance center. Whereas, the relative occlusion ordering amodal center regression head, for each relative occlusion ordering layer, predicts the offset from each pixel location belonging to *thing*
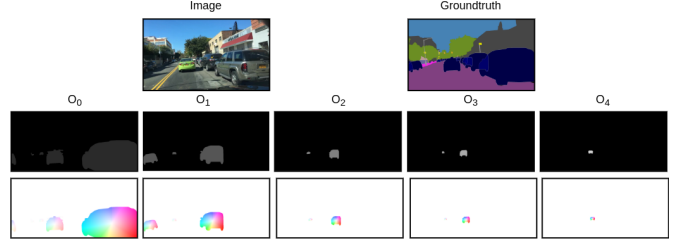

Fig. 3: Groundtruth examples for relative occlusion order segmentation (top-row) and instance center regression (bottom-row) consisting of layer from O_0 to O_5. Best viewed at $\times 4$ zoom.

classes of the layer to its corresponding amodal object instance mass center. Here, the layers of relative occlusion ordering are defined similarly as in the semantic decoder. Further, we concatenate F_{IO} with features of inmodal occlusion-aware center prediction head to incorporate object-level global occlusion features before feeding it to the aforementioned heads. Finally, we use L_1 loss to train both the heads (L_{aco}, L_{rooacr}). The overall loss for the instance decoder is

$$L_{inst} = L_{tss} + L_{ico} + \alpha L_{icp} + \beta(L_{icr} + L_{aco} + L_{rooacr}), \tag{4}$$

where the loss weights $\alpha = 200$ and $\beta = 0.01$.

Note that we learn amodal center offset instead of the amodal center itself to have a common instance-ID that encapsulates both the amodal and inmodal masks.

6) Amodal Mask Refiner: We propose the amodal mask refiner module to model the ability of humans to leverage priors on complete physical structures of objects for amodal perception, in addition to visually conditioned occlusion cues. This module builds an embedding that embeds the features of the unoccluded object mask and correlates them with the generated amodal features to complement the lack of visually conditioned occlusion features. The amodal mask refiner shown in Fig. 2 consists of two encoders, unoccluded feature embeddings, and a decoder. We employ the RegNet [22] topology with its first and last stages removed as the two encoders with feature encoding resolution of $\times 16$ downsampled with respect to the input. The two encoders are an inmodal embedding encoder ($IE_{enc} \in \mathbb{R}^{(H/16) \times (W/16) \times C}$) that encodes unoccluded objects features and a query encoder ($Q_{enc} \in \mathbb{R}^{(H/16) \times (W/16) \times C}$) that encodes the amodal features, where H and W are the height and width of the input image and C is the feature dimension which is set to 64. Subsequently, an embedding matrix $E_{IE} \in \mathbb{R}^{N \times D}$ embeds the IE_{enc} encoding to create the embedding of unoccluded object masks. Further, to extract the mask embedding information from E_{IE}, we compute two key matrices, namely, $K_{IE} \in \mathbb{R}^{N \times D}$ matrix and $K_Q \in \mathbb{R}^{1 \times D}$ matrix, from IE_{enc} and Q_{enc} encodings, respectively. Here, $N = 128$ and $D = [(H/16) \times (W/16) \times C]$.

Next, we compute the inner product of K_{IE} and K_Q followed by a softmax and take the inner product of the resulting probability and E_{IE}. We then rearrange this output into $(H/16) \times (W/16) \times C$ shape and concatenate it with Q_{enc} and feed it to the decoder. The decoder employs repeated blocks of two 3×3 depth-wise separable convolutions, followed by a bilinear interpolation to upsample by a factor of 2 until the upsampled output resolution is $\times 4$ downsampled with respect to the input. We refer to this output as F_{AMR}. The resulting features enrich the amodal features of occluded objects with similar

unoccluded object features, thereby enabling our network to predict more accurate amodal masks.

The amodal mask refiner takes two inputs, namely, the amodal features and the features of the unoccluded objects. The input amodal features are obtained by concatenating the output features (Fig. 2) of relative occlusion ordering heads of the semantic and instance decoders. To compute the features of the unoccluded object, we first perform instance grouping using predictions of the inmodal occlusion-aware, inmodal center regression, and thing semantic segmentation heads to obtain the inmodal instance masks. We then discard all the occluded inmodal instances to generate an unoccluded instance mask. Next, we multiply the aforementioned mask with the output of the second layer of the inmodal center regression head to compute the final unoccluded object features. Finally, the amodal mask refiner outputs F_{AMR} which is then concatenated with the amodal features. We employ two similar heads as relative occlusion ordering amodal center regression and segmentation that takes the aforementioned concatenated features as input. We use the same loss functions and loss weights for training the heads as described in Sec. III-A5.

7) Inference: We perform a series of steps during inference to merge the outputs of the semantic and instance decoders to yield the final amodal panoptic segmentation. We begin with computing the semantic segmentation prediction and the *thing* foreground mask. To do so, we duplicate the void class logit of the *thing* semantic segmentation head logits N_{stuff}-times, such that its number of channels transforms from $1 + N_{thing}$ to $N_{stuff} + N_{thing}$. We then add it to the logits of the semantic segmentation head and employ a softmax followed by an argmax function to obtain the final semantic segmentation prediction. Subsequently, we assign 0 to all the *stuff* classes and 1 to all the *thing* classes to obtain the *thing* foreground mask. Next, we obtain the inmodal center point predictions by employing a keypoint-based non-maximum suppression [5] and confidence thresholding (0.1) to filter out the low confidence predictions while keeping only the top-k (200) highest confidence scores on the heatmap prediction of inmodal occlusion-aware center prediction head. We then obtain the amodal center points predictions by applying the corresponding offsets from the amodal instance head to the inmodal center point predictions. We obtain the class-agnostic instance-IDs and the inmodal instance mask using simple instance grouping [5] with the inmodal center prediction and the *thing* foreground mask. Further, we compute semantic labels for each instance-ID by the majority vote of the corresponding predicted semantic labels with its inmodal instance masks.

Now, for each instance-ID, we have its semantic label, inmodal mask, and the amodal center prediction. We compute the relative occlusion order segmentation masks for each layer by applying a threshold of 0.5 on the outputs of the relative occlusion ordering segmentation head connected to the amodal mask refiner. We then assign the instance-ID to its corresponding relative occlusion ordering layer by checking if the corresponding amodal center lies within the segmentation mask of the layer in question. Finally, we again use the simple instance grouping at each of the relative occlusion ordering layers. For all instance-IDs belonging to a layer, we apply the instance grouping using its amodal instance center and

regression along with the corresponding segmentation mask to compute the amodal mask. In the end, for each *thing* object, we have its unique instance-ID, semantic label, inmodal, and amodal mask along with *stuff* class semantic predictions from the semantic segmentation prediction. We obtain the visible attribute of the amodal mask directly from the inmodal mask and obtain the occluded attributes of the amodal mask by removing the inmodal mask segment from the amodal mask.

IV. EXPERIMENTAL EVALUATION

In this section, we describe the datasets that we benchmark on in Sec. IV-A and the training protocol in Sec. IV-B. We then present extensive benchmarking results in Sec. IV-C, followed by a detailed ablation study on the architectural components in Sec. IV-D and qualitative comparisons in Sec. IV-E. We use the standard Amodal Panoptic Quality (APQ) and Amodal Parsing Coverage (APC) metrics [2] to quantify the performance.

A. Datasets

KITTI-360-APS [2] provides amodal panoptic annotations for the KITTI-360 [23] dataset. It consists of 9 sequences of urban street scenes with annotations for 61,168 images. The sequence numbered 10 of the dataset is treated as the validation set. This dataset comprises 7 *thing* classes, namely, car, pedestrians, cyclists, two-wheeler, van, truck, and other vehicles. Further, the dataset consists of 10 *stuff* classes. These stuff classes are road, sidewalk, building, wall, fence, pole, traffic sign, vegetation, terrain, and sky.

BDD100K-APS [2] extends the BDD100K [24] dataset with amodal panoptic annotations for 15 of its sequences consisting of 202 images per sequence. The training and validation set consists of 12 and 3 sequences, respectively. Pedestrian, car, truck, rider, bicycle, and bus are the 6 *thing* classes. Whereas, road, sidewalk, building, fence, pole, traffic sign, fence, terrain, vegetation, and sky are the 10 *stuff* classes

B. Training Protocol

All our models are trained using the PyTorch library on 8 NVIDIA TITAN RTX GPUs with a batch size of 8. We train our network in two stages, with a crop resolution of 376×1408 pixels and 448×1280 pixels for the KITTI-360-APS and BDD100K-APS datasets, respectively. For each stage, we use the Adam optimizer with a poly learning rate schedule, where the initial learning rate is set to 0.001. We train our model for 300K iterations for the KITTI-360-APS dataset and 70K iterations for the BDD100K-APS dataset, while using random scale data augmentation within the range of $[0.5, 2.0]$ with flipping for each stage. We use $N = 8$ for relative occlusion order layers. We first train the model without the amodal mask refiner, followed by freezing the weights of the architectural components from the previous stage and train only the amodal mask refiner.

C. Benchmarking Results

In this section, we present results comparing the performance of our proposed PAPS architecture against current state-of-the-art amodal panoptic segmentation approaches. We report the

TABLE I: Comparison of amodal panoptic segmentation benchmarking results on the KITTI-360-APS and BDD100K-APS validation set. Subscripts S and T refer to *stuff* and *thing* classes respectively. All scores are in [%].

Model	KITTI-360-APS						BDD100K-APS					
	APQ	APC	APQ_S	APQ_T	APC_S	APC_T	APQ	APC	APQ_S	APQ_T	APC_S	APC_T
Amodal-EfficientPS	41.1	57.6	46.2	33.1	58.1	56.6	44.9	46.2	54.9	29.9	64.7	41.4
ORCNN [15]	41.1	57.5	46.2	33.1	58.1	56.6	44.9	46.2	54.9	29.9	64.7	41.5
BCNet [18]	41.6	57.9	46.2	34.4	58.1	57.6	45.2	46.4	55.0	30.7	64.7	42.1
VQ-VAE [17]	41.7	58.0	46.2	34.6	58.1	57.8	45.3	46.5	54.9	30.8	64.7	42.2
Shape Prior [16]	41.8	58.2	46.2	35.0	58.1	58.2	45.4	46.6	55.0	31.0	64.8	42.6
ASN [6]	41.9	58.2	46.2	35.2	58.1	58.3	45.5	46.6	55.0	31.2	64.8	42.7
APSNet [2]	42.9	59.0	46.7	36.9	58.5	59.9	46.3	47.3	55.4	32.8	65.1	44.5
PAPS (Ours)	**44.6**	**61.4**	**47.5**	**40.1**	**59.2**	**64.7**	**48.7**	**50.4**	**56.5**	**37.1**	**66.4**	**51.6**

APQ and APC metrics of the existing state-of-the-art methods directly from the published manuscript [2]. Tab. I presents the benchmarking results on both datasets.

We observe that our proposed PAPS architecture achieves the highest APQ and APC scores compared to APSNet and other baselines on both datasets. The improvement of 1.7%-2.7% in both the metrics can be attributed to the various proposed components of our architecture. For *stuff* segmentation, the complementary features from the cross-task module aid in better distinguishing *stuff* and *thing* classes, while the high resolution features with the long-range contextual features help in finer segmentation of the boundaries. Consequently, we observe an improvement of 0.7%-1.3% in the *stuff* components of the metrics for both datasets. The *thing* components of the metrics achieve an improvement of 3.2%-7.1% which can be attributed to the synergy of several factors. The context extractor and the cross-task modules provide richer multi-scale representations along with complementary semantic decoder features. This enables reliable segmentation of far-away small-scale instances. Further, the incorporation of local and object-level global occlusion information from the instance and semantic decoder heads enables explicit amodal reasoning capabilities. We also believe that the relative occlusion ordering layers force the network to capture the complex underlying relationship of objects to one another in the context of occlusions. Lastly, the amodal mask refiner module with its transformation of amodal features with unoccluded object mask embeddings improves the quality of large occlusion area segmentation as observed from the higher improvement in APC than the APQ metric. Overall, PAPS establishes the new state-of-the-art on both the amodal panoptic segmentation benchmarks.

D. Ablation Study

In this section, we first study the improvement due to the various architectural components that we propose in our PAPS and study the generalization ability of the amodal mask refiner by incorporating it in various proposal-based methods. We then evaluate the performance of PAPS for panoptic segmentation and amodal instance segmentation tasks.

1) Detailed Study on the PAPS Architecture: In this section, we quantitatively evaluate the influence of each proposed architectural component in PAPS, on the overall performance. Here, the addition of modules to the architecture of the base model M1 in the incremental form is performed according to their description in Sec. III. Tab. II presents results from this experiment. We begin with the model M1 which employs

TABLE II: Evaluation of various architectural components of our proposed PAPS model. The performance is shown for the models trained on the BDD100K-APS dataset and evaluated on the validation set. Subscripts S and T refer to *stuff* and *thing* classes respectively. All scores are in [%].

Model	APQ	APC	APQ_S	APQ_T	APC_S	APC_T
M1	45.6	46.9	55.8	30.4	65.7	42.2
M2	45.9	47.1	55.8	31.0	65.7	42.7
M3	46.1	47.2	55.9	31.3	65.8	42.9
M4	46.3	47.3	55.9	31.9	65.8	43.3
M5	46.7	47.7	56.3	32.4	66.2	43.9
M6	47.4	48.5	**56.5**	33.7	**66.4**	45.8
M7 (PAPS)	**48.7**	**50.4**	**56.5**	**37.1**	**66.4**	**51.6**

a semantic decoder as described in Sec. III-A4 without any cross-task module and occlusion segmentation head and is similar to [5] with amodal capabilities. For the instance decoder, it employs the aforementioned semantic decoder with the heads described in Sec. III-A5 without occlusion-awareness of center and *thing* semantic segmentation. In the M2 model, we replace the instance decoder architecture with that described in Sec. III-A5 without the cross-task module and the same heads as the M1 model. The improvement in performance shows the importance of multi-scale features from cross-layers for amodal instance center regression. In the M3 model, we add the *thing* segmentation head to the instance decoder whose output is used during inference as described in Sec. III-A7. The improvement achieved indicates that the two decoders capture diverse representations of *thing* classes which further improves the performance.

In the M4 model, we add the context extractor module. The higher increase in APQ_T compared to APC_T indicates that the multi-scale features obtained from the aggregation of within-scales and cross-scales layers are much richer in the representation capacity, thereby improving the detection of small far away objects. Building upon M4, in the M5 model, we add the cross-task module. The increase in both *stuff* and *thing* components of the metrics demonstrates that the two decoders learn complementary features which when propagated bidirectionally is mutually beneficial for each of them. In the M6 model, we add the occlusion segmentation head and occlusion awareness to the inmodal center prediction head. We observe an improvement of 1.3%-1.9% in *thing* components of the metrics demonstrating that the incorporation of occlusion information is integral for good amodal mask segmentation. Lastly, in the M7 model, we add the amodal mask refiner. The substantial improvement of 3.4% and 5.8% in APQ_T and APC_T, respectively, demonstrates the efficacy of our proposed module. We note that the improvement in APC_T is higher than APQ_T indicating

TABLE III: Evaluation of various propsal-based amodal panoptic segmentation approaches with our proposed amodal mask refiner. The performance is shown for the models trained on the BDD100K-APS dataset and evaluated on the validation set. Subscript T refer to *thing* classes. All scores are in [%].

Model	Amodal Mask Refiner	APQ	APC	APQ$_T$	APC$_T$
ORCNN [15]		44.9	46.2	29.9	41.4
BCNet [18]		45.2	46.4	30.7	42.1
ASN [6]		45.5	46.6	31.2	42.7
APSNet [2]		46.3	47.3	32.8	44.5
ORCNN [15]	✓	45.3	46.6	30.9	42.8
BCNet [18]	✓	46.3	47.8	33.2	46.4
ASN [6]	✓	46.7	48.1	34.4	47.1
APSNet [2]	✓	**47.5**	**48.9**	**35.9**	**49.2**

TABLE IV: Performance comparison of panoptic segmentation on the Cityscapes validation set. − denotes that the metric has not been reported for the corresponding method. All scores are in [%].

Network	PQ	SQ	RQ	PQ$_T$	PQ$_S$	AP	mIoU
Panoptic FPN [25]	58.1	−	−	52.0	62.5	33.0	75.7
UPSNet [11]	59.3	79.7	73.0	54.6	62.7	33.3	75.2
DeeperLab [21]	56.3	−	−	−	−	−	−
Seamless [7]	60.3	−	−	56.1	63.3	33.6	77.5
SSAP [4]	61.1	−	−	55.0	−	−	−
AdaptIS [3]	62.0	−	−	58.7	64.4	36.3	79.2
Panoptic-DeepLab [5]	63.0	−	−	−	−	35.3	80.5
EfficientPS [9]	63.9	81.5	77.1	**60.7**	66.2	**38.3**	79.3
PAPS (ours)	**64.3**	**82.1**	**77.3**	60.1	**67.3**	37.2	**80.8**

that the increase in segmentation quality of objects with larger occlusion areas is relatively higher than the smaller areas. This result precisely demonstrates the utility of our proposed amodal mask refiner, validating our idea of using embeddings of non-occluded object masks to supplement the amodal features with correlation for mid-to-heavy occlusion cases.

2) Generalization of amodal mask refiner: In this section, we study the generalization ability of our proposed amodal mask refiner by incorporating it in existing proposal-based amodal panoptic segmentation approaches. To do so, we adapt the amodal mask refiner by removing all downsampling layers in the encoders and upsampling layers from its decoder, to make it compatible with proposal-based approaches. We add an occlusion classification branch in the amodal instance head of all the proposal-based methods similar to ASN [6] and add another identical amodal mask head. The output of the fourth layer of the amodal mask head of each method is considered as the amodal features input. For the non-occluded object features, we multiply the output of the occlusion classification branch with the output of the fourth layer of the inmodal mask head. We feed the amodal features and non-occluded object features to the amodal mask refiner, followed by concatenating its output with the amodal features. Subsequently, we feed these concatenated features to the newly added amodal mask head. To train the networks, we use the same two-stage procedure described in Sec. IV-B and the training protocol described in [2].

Tab. III presents the results from this experiment. We observe a considerable improvement in the performance of all the proposal-based methods demonstrating the effectiveness and the ease of integration into existing architectures. Moreover, the improvement achieved for APSNet is higher than ORCNN indicating that the performance can vary depending on the quality of the inmodal and amodal feature representations in the network.

TABLE V: Amodal instance segmentation results on the KINS dataset. All scores are in [%].

Model	Amodal$_{AP}$	Inmodal$_{AP}$
ORCNN [15]	29.0	26.4
VQ-VAE [17]	31.5	−
Shape Prior [16]	32.1	29.8
ASN [6]	32.2	29.7
APSNet [2]	35.6	32.7
PAPS (Ours)	**37.4**	**33.1**

3) Panoptic Segmentation Results on Cityscapes Dataset: In this section, we evaluate the performance of our proposed PAPS for panoptic segmentation on the Cityscapes [26] dataset. In the architecture, we remove the amodal mask refiner, occlusion segmentation, amodal center offset, relative occlusion order segmentation, and amodal center regression heads as they only contribute to obtaining the amodal masks. We train our network with a learning rate $lr = 0.001$ for 90K iterations using the Adam optimizer. We report the Panoptic Quality (PQ), Segmentation Quality (SQ) and Recognition Quality (RQ) metrics on the validation set of Cityscapes for single-scale evaluation in Tab. IV. For the sake of completeness, we also report the Average Precision (AP), and the mean Intersection-over-Union (mIoU) scores. We observe that PAPS achieves the highest PQ score of 64.3% which is 1.3% and 0.4% higher than the state-of-the-art Panoptic-DeepLab and EfficientPS, respectively. The improvement achieved over Panoptic-DeepLab demonstrates the efficacy of our proposed modules and architectural design choices.

4) Performance on KINS Dataset: We benchmark the performance of our proposed PAPS architecture on the KINS [6] amodal instance segmentation benchmark. This benchmark uses the Average Precision (AP) metric for evaluating both amodal and inmodal segmentation. We train our network with a learning rate $lr = 0.001$ for 40K iterations using the Adam optimizer. We use the same validation protocols as [6]. Tab. V presents results in which our proposed PAPS outperforms the state-of-the-art APSNet by 1.8% and 0.4% for amodal AP and inmodal AP, respectively, establishing the new state-of-the-art on this benchmark. The large improvement in the Amodal$_{AP}$ compared to the Inmodal$_{AP}$ indicates refining amodal masks with unoccluded object embeddings is an effective strategy.

E. Qualitative Evaluations

In this section, we qualitatively compare the amodal panoptic segmentation performance of our proposed PAPS architecture with the previous state-of-the-art APSNet. Fig. 4 presents the qualitative results. We observe that both approaches are capable of segmenting partial occlusion cases. However, our PAPS outperforms APSNet under moderate to heavy occlusion cases such as cluttered cars and pedestrians. In Fig. 4(a) the faraway cars on the right are detected more reliably by our network along with their amodal mask segmentations demonstrating the positive effects of within-scales and cross-scales multi-scale features and the occlusion aware heads. In Fig. 4(b), our model successfully predicts the amodal masks of heavily occluded pedestrians and cars. This demonstrates the utility of our amodal mask refiner module. By relying on the unoccluded mask features, PAPS is able to make a coarse estimate of

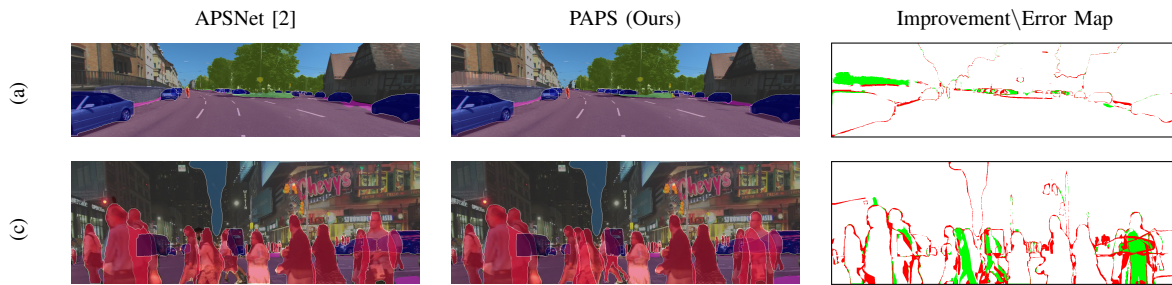| APSNet [2]	PAPS (Ours)	Improvement\Error Map

Fig. 4: Qualitative amodal panoptic segmentation results of our proposed PAPS network in comparison to the state-of-the-art APSNet [2] on (a) KITTI-360-APS and (b) BDD100K-APS datasets. We also show the Improvement\Error Map which denotes the pixels that are misclassified by PAPS in red and the pixels that are misclassified by APSNet but correctly predicted by PAPS in green.

the object's amodal masks. Furthermore, PAPS achieves more accurate segmentation of the challenging thin *stuff* classes such as poles and fences.

V. CONCLUSION

In this work, we presented the first proposal-free amodal panoptic segmentation architecture that achieves state-of-the-art performance on both the KITTI-360-APS and BDD100K-APS datasets. To facilitate learning proposal-free amodal panoptic segmentation, our PAPS network learns amodal center offsets from the inmodal instance center predictions while decomposing the scene into different relative occlusion ordering layers such that there are no overlapping amodal instance masks within a layer. It further incorporates several novel network modules to capture within-layer multi-scale features for richer multi-scale representations, to enable bilateral propagation of complementary features between the decoders for their mutual benefit, and to integrate global and local occlusion features for effective amodal reasoning. Furthermore, we proposed the amodal mask refiner module that improves the amodal segmentation performance of occluded objects for both proposal-free and proposal-based architectures. Additionally, we presented detailed ablation studies and qualitative evaluations highlighting the improvements that we make to various core network modules of our amodal panoptic segmentation architectures. Finally, we have made the code and models publicly available to accelerate further research in this area.

REFERENCES

[1] B. Nanay, "The importance of amodal completion in everyday perception," *i-Perception*, vol. 9, no. 4, 2018.

[2] R. Mohan and A. Valada, "Amodal panoptic segmentation," *arXiv preprint arXiv:2202.11542*, 2022.

[3] K. Sofiiuk, O. Barinova, and A. Konushin, "Adaptis: Adaptive instance selection network," in *Int. Conf. on Computer Vision*, 2019, pp. 7355–7363.

[4] N. Gao, Y. Shan, Y. Wang, X. Zhao, Y. Yu, M. Yang, and K. Huang, "Ssap: Single-shot instance segmentation with affinity pyramid," in *Int. Conf. on Computer Vision*, 2019.

[5] 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 *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2020.

[6] L. Qi, L. Jiang, S. Liu, X. Shen, and J. Jia, "Amodal instance segmentation with kins dataset," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 3014–3023.

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

[8] N. Gosala and A. Valada, "Bird's-eye-view panoptic segmentation using monocular frontal view images," *IEEE Robotics and Automation Letters*, 2022.

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

[10] K. Sirohi, R. Mohan, D. Büscher, W. Burgard, and A. Valada, "Efficientlps: Efficient lidar panoptic segmentation," *IEEE Transactions on Robotics*, 2021.

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

[12] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *Workshop on statistical learning in computer vision, ECCV*, 2004.

[13] J. Uhrig, E. Rehder, B. Fröhlich, U. Franke, and T. Brox, "Box2pix: Single-shot instance segmentation by assigning pixels to object boxes," in *IEEE Intelligent Vehicles Symposium*, 2018, pp. 292–299.

[14] F. R. Valverde, J. V. Hurtado, and A. Valada, "There is more than meets the eye: Self-supervised multi-object detection and tracking with sound by distilling multimodal knowledge," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 612–11 621.

[15] P. Follmann, R. König, P. Härtinger, M. Klostermann, and T. Böttger, "Learning to see the invisible: End-to-end trainable amodal instance segmentation," in *IEEE Winter Conference on Applications of Computer Vision*, 2019, pp. 1328–1336.

[16] Y. Xiao, Y. Xu, Z. Zhong, W. Luo, J. Li, and S. Gao, "Amodal segmentation based on visible region segmentation and shape prior," in *AAAI Conference on Artificial Intelligence*, 2021.

[17] W.-D. Jang, D. Wei, X. Zhang, B. Leahy, H. Yang, J. Tompkin, D. Ben-Yosef, D. Needleman, and H. Pfister, "Learning vector quantized shape code for amodal blastomere instance segmentation," *arXiv preprint arXiv:2012.00985*, 2020.

[18] L. Ke, Y.-W. Tai, and C.-K. Tang, "Deep occlusion-aware instance segmentation with overlapping bilayers," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2021, pp. 4019–4028.

[19] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[20] L.-C. Chen, M. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, and J. Shlens, "Searching for efficient multi-scale architectures for dense image prediction," in *Advances in Neural Information Processing Systems*, 2018, pp. 8713–8724.

[21] T.-J. Yang, M. D. Collins, Y. Zhu, J.-J. Hwang, T. Liu, X. Zhang, V. Sze, G. Papandreou, and L.-C. Chen, "Deeperlab: Single-shot image parser," *arXiv preprint arXiv:1902.05093*, 2019.

[22] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 10 428–10 436.

[23] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *arXiv preprint arXiv:2109.13410*, 2021.

[24] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2636–2645.

[25] A. Kirillov, R. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 6399–6408.

[26] 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 *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

Perceiving the Invisible: Proposal-Free Amodal Panoptic Segmentation

- Supplementary Material -

Rohit Mohan and Abhinav Valada

In this supplementary material, we provide additional ablation studies on the proposed architectural components and the illustration of the context extractor module.

S.1. ABLATION STUDY

In this section, we first study the importance of the various components of our proposed cross-task module. Subsequently, we study the influence of the number of relative occlusion ordering layers on the performance of our network. For all the experiments, we train our PAPS network without the amodal mask refiner on the BDD100K-APS dataset and evaluate it on the validation set. We use APQ and APC metrics as the principal evaluation criteria for all the experiments performed in this section.

A. Evaluation of the Cross-Task Module

In this section, we evaluate our proposed architecture of the cross-task module to enable bilateral propagation of features between the task-specific decoders. For this experiment, we use the PAPS architecture without the amodal mask refiner, similar to model M6 in Sec. IV-D. Tab. S.1 presents results from this experiment. We begin with model M61 which does not use the cross-task module. In model M62, we concatenate outputs of the opposite decoder as F_O. For the instance decoder, $F_O = F_S$ where F_S are the output features of the semantic decoder. For the semantic decoder, $F_O = F_I$ where F_I are the output features of the semantic decoder. The improvement in the performance shows the utility of propagating features between the task-specific decoders. In the model M63, we define F_O as the summation of the task-specific decoder features given as

$$F_O = F_I + F_S. \tag{1}$$

We observe a drop in performance for model M63 compared to both model M61 and model M62 indicating that the use of summation fails to capture complementary features and at the same time affects learning the relevant primary features of the decoders themselves. In model M64, we employ self-attention given by

$$F_R = F_I + F_S, \tag{2}$$
$$F_O = g_3(F_R) \cdot F_R, \tag{3}$$

where $g_3(\cdot)$ is the function to compute the confidence scores of F_R. This model achieves improved performance over both model M62 and model M63 demonstrating that the attention

TABLE S.1: Ablation study on various configurations of our proposed cross-task head. The performance is shown for the models trained on the BDD100K-APS dataset and evaluated on the validation set. Subscripts S and T refer to *stuff* and *thing* classes respectively. All scores are in [%].

Model	APQ	APC	APQ_S	APQ_T	APC_S	APC_T
M61	46.9	48.1	56.1	33.2	66.0	45.2
M62	47.0	48.1	56.2	33.3	66.1	45.3
M63	46.7	48.0	55.9	32.9	65.9	45.1
M64	47.1	48.2	56.3	33.4	66.3	45.4
M65	47.0	48.1	56.2	33.3	66.1	45.3
M66	47.1	48.2	56.3	33.4	66.3	45.4
M67	**47.4**	**48.5**	**56.5**	**33.7**	**66.4**	**45.8**

mechanisms are beneficial for learning complementary features. As a next step, we employ self-attention to each individual task-specific decoder features in model M65 and define F_O as

$$F_O = g_1(F_I) \cdot F_I + g_2(F_S) \cdot F_S, \tag{4}$$

where $g_1(\cdot)$ and $g_2(\cdot)$ are the functions to compute the confidence scores. Model M65 achieves a score lower than Model M64 and similar to Model M62. This indicates that applying self-attention to each input of the cross-task module effectively reduces them to be similar to a summation operation. Hence, in Model M66, we employ cross-attention in F_O as follows

$$F_O = (1 - g_1(F_S)) \cdot F_I + (1 - g_2(F_I)) \cdot F_S. \tag{5}$$

This model achieves a performance similar to Model M64 demonstrating that cross-attention is equally important as self-attention. Lastly, we use our proposed cross-attention followed by self-attention cross-task configuration (Eq. (1) and Eq. (2)), which yields the highest overall improvement. Consequently, from this experiment, we infer that cross-attention enables learning of adaptive complementary decoder features, whereas the following self-attention enables enhancement of these highly discriminative complementary features.

B. Detailed Study on the Relative Occlusion Ordering Layers

In this section, we study the effects of the number of relative occlusion ordering layers on the performance of our proposed architecture. Similar to Sec. S.1-A, for this experiment we use the PAPS architecture without the amodal mask refiner module. Tab. S.2 shows results from this experiment. We begin with $N = 4$ where N is the number of relative occlusion ordering layers. The model achieves an improved score of 45.4% and 46.6% in APQ and APC, respectively compared to the baselines. This indicates that with four relative occlusion ordering layers, we can encapsulate sufficient object instances present in a given scene. Next, we use $N = 6$ and obtain a

TABLE S.2: Influence on varying the number of layers of the relative occlusion ordering layers. The performance is shown for the models trained on the BDD100K-APS dataset and evaluated on the validation set. N is the number of layers, subscripts S and T refer to *stuff* and *thing* classes respectively. All scores are in [%].

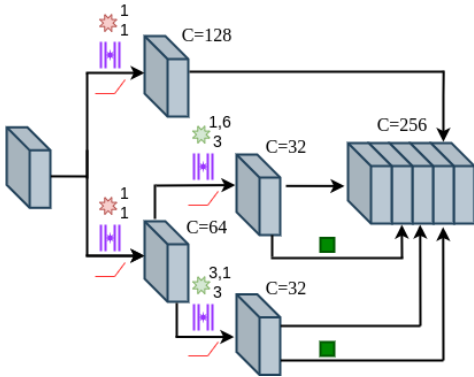| N	APQ	APC	APQ_S	APQ_T	APC_S	APC_T
4	45.4	46.6	56.1	29.3	65.9	41.1
6	46.8	47.8	56.3	32.6	66.2	44.3
8	**47.4**	**48.5**	**56.5**	**33.7**	**66.4**	**45.8**
10	**47.4**	**48.5**	**56.5**	**33.7**	**66.4**	**45.8**
12	**47.4**	**48.5**	**56.5**	**33.7**	**66.4**	**45.8**

Fig. S.1: Topology of our proposed context extractor module.

significant improvement in the *thing* components of the metrics. Subsequently, we train the model with $N = 8$ which yields a lower performance in the metrics compared to $N = 6$. This indicates that $N = 6$ covers the majority of object instances in a given scene throughout the dataset. We then train the network with $N = 10$ and $N = 12$. These models do not achieve any improvement over the model with $N = 8$ layers demonstrating that with eight relative occlusion ordering layers, we can encapsulate the maximal number of object instances in the dataset.

S.2. CONTEXT EXTRACTOR

Our proposed context extractor module enriches cross-scale features with within-scale contextual features, resulting in a rich multi-scale representation. This yields an improvement in performance for the instance decoder of our PAPS architecture as shown in Sec. IV-D-B. Fig. S.1 illustrates the architecture of the context extractor module. It splits the input into two parallel branches and employs two 1×1 convolutions. One of the branches is further subdivided into two parallel branches. Here, each branch uses a 3×3 depth-wise atrous separable convolutions with a dilation rate of $(1, 6)$ and $(3, 1)$, respectively. These branches are again subdivided into two parallel branches each. In each of these two parallel branches, one branch employs a global pooling layer. Finally, all the outputs of the remaining parallel branches are concatenated. Please note that each of the convolutions is followed by batch normalization and ReLU activation function.