Algorithm 1: Stitching Video Panoptic Predictions

Input : Panoptic prediction P_t for image t, and panoptic prediction R_t for image $t+1$ when concatenated with image t.

Output: In-place stitched panoptic predictions P_t.

for $t = 1, 2, ..., T-1$ **do**

  Increase all instance IDs of P_{t+1} by the maximum of the instance IDs of P_1 to P_t;

  // Find all overlapping region pairs $\mathbb{C}$ between

  // R_t and P_{t+1} with the same classes.

  Let $\mathbb{S} = [(r, p) \mid r \in R_t \wedge p \in P_{t+1} \wedge r \cap p > 0]$;

  Let $\mathbb{C} = [(r, p) \mid (r, p) \in \mathbb{S} \wedge \text{cls}(r) = \text{cls}(p)]$ where $\text{cls}(\cdot)$ denotes the class;

  Sort $\mathbb{C} = [(r, p)]$ with respect to $\text{IoU}(r, p)$ in ascending order;

  // For each region in R_t, $\mathbb{M}$ stores the region in

  // P_{t+1} having the largest IoU with it. $\mathbb{N}$ stores

  // the opposite direction.

  Let $\mathbb{M}$, $\mathbb{N}$ be empty dictionaries;

  for $(r, p) \in \mathbb{C}$ **do**

    $\mathbb{M}[r] = p$ and $\mathbb{N}[p] = r$;

  for $r \to p \in \mathbb{M}$ **do**

    // Propagate IDs from R_t to P_{t+1} and R_{t+1}

    // if r and p map to each other in $\mathbb{M}$ and $\mathbb{N}$.

    if $r = \mathbb{N}[\mathbb{M}[r]]$ **then**

      if $t < T-1$ **then**

        Assign the region in R_{t+1} that has the ID of p with the ID of r;

      Assign the region of p in P_{t+1} with the instance ID of r in R_t;

A. Stitching Algorithm

Alg. 1 shows the details of the algorithm to stitch video panoptic predictions to form predictions with consistent IDs throughout the entire sequence. We split the panoptic prediction of the concatenated image pair t and $t+1$ in the middle, and use P_t and R_t to denote the left and the right prediction, respectively. This makes P_t the panoptic prediction of image t, and R_t the panoptic prediction of image $t+1$ with instance IDs that are consistent with those of P_t. The objective of the algorithm is to propagate IDs from R_t to P_{t+1} so that each object in P_t and P_{t+1} will have the same ID. The ID propagation is based on mask IoU between region pairs. For each region r in R_t, we find the region p in P_{t+1} that has the same class and the largest IoU with it. We use $\mathbb{M}$ to store this mapping. Similarly, for each region p in P_{t+1}, we also find the region r in R_t that has the same class and the largest IoU with it. We use $\mathbb{N}$ to store this mapping. If a region r of R_t and a region p of P_{t+1} are matched to each other (*i.e.* $\mathbb{M}(r) = p$ and $\mathbb{N}(p) = r$), then we propagate the ID from r to p.

Method	Pedestrians			Cars		
	sMOTSA	MOTSA	IDS	sMOTSA	MOTSA	IDS
TrackR-CNN [77]	46.8	65.1	78	76.2	87.8	93
MOTSNet [64]	54.6	69.3	-	78.1	87.2	-
MOTSFusion [55]	58.9	71.9	36	82.6	90.2	51
PointTrack [92]	62.4	77.3	19	85.5	94.9	22
ViP-DeepLab + KF	68.3	83.2	15	86.0	94.7	52

Table 6: Results on KITTI MOTS validation set.

$\mathcal{L}_{\text{depth}}$ weight	k = 1	k = 2	k = 3	k = 4	VPQ	absRel	DVPQ
0.1	68.9	61.9	58.8	56.5	61.5	9.51	51.3
1.0	69.0	62.0	58.7	56.5	61.6	7.21	55.1
10	67.8	61.1	57.5	55.5	60.5	6.54	54.3

Table 7: ViP-DeepLab trained with different training weights for $\mathcal{L}_{\text{depth}}$ on Cityscapes-DVPS.

B. More Experiments

KITTI MOTS Validation Set We first evaluate ViP-DeepLab on the validation set of KITTI MOTS benchmark [77]. Tab. 6 shows the comparisons between ViP-DeepLab and the previous methods. We adopt the same strategy as we used for training models for KITTI MOTS Leaderboard except that the training data used in here does not include the validation set. As shown in the table, our method equipped with Kalman filter outperforms the previous methods by a large margin.

Effects of Depth Loss Weight Next, we study the effects of different training weights for the depth loss $\mathcal{L}_{\text{depth}}$. In the previous experiments on Cityscapes-DVPS and SemKITTI-DVPS, we use the depth loss defined by Equ. (4), which has a loss weight of 1.0. For the purpose of ablation study, we change the training weight from 1.0 to 10 and 0.1. The results are shown in Tab. 7. From the table we can see that as the $\mathcal{L}_{\text{depth}}$ weight increases, ViP-DeepLab performs better on the sub-task monocular depth estimation (*i.e.* absRel becomes lower), but worse on the sub-task video panoptic segmentation (*i.e.* VPQ becomes lower). This is consistent with our intuition that a larger $\mathcal{L}_{\text{depth}}$ weight makes the model focus more on the task of monocular depth estimation. The metric that matters most here is DVPQ, which unifies the metrics of both sub-tasks. In order to get a high DVPQ score, the predictions must be accurate on both tasks. Therefore, finding a balanced $\mathcal{L}_{\text{depth}}$ weight is critical to get a high DVPQ. As the table shows, setting $\mathcal{L}_{\text{depth}}$ weight to 1.0 achieves the best results among the three choices.

KITTI Depth Validation Set Finally, we show the performance of ViP-DeepLab on the official validation set of

Method	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	absRel$\downarrow$	sqRel$\downarrow$	RMSE$\downarrow$	RMSElog$\downarrow$	SILog$\downarrow$
[20]	95.77	99.21	99.75	6.99	1.27	2.86	0.104	9.73
Ours	96.27	99.41	99.81	5.72	0.96	2.58	0.092	8.47

Table 8: Results on the official KITTI depth validation set. $\uparrow$: The higher the better. $\downarrow$: The lower the better.

Class	HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA	sMOTSA
Cars	76.38	82.70	70.93	88.70	88.77	75.86	86.00	90.75	81.03
Pedestrians	64.31	70.69	59.48	75.71	81.77	67.52	74.92	84.40	68.76

Table 9: ViP-DeepLab performance on the KITTI MOTS test set for the new metrics.

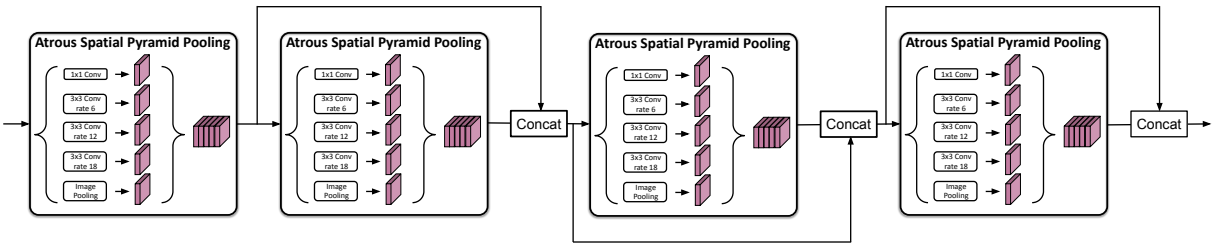

Figure 8: The architecture of Cascade-ASPP, which is employed as *Dense Multi-scale Context* in the next-frame instance branch. It cascades four ASPP modules with the outputs densely connected.

Method	STQ	AQ	SQ
VPSNet [42]	0.50	0.35	0.72
Ours	0.64	0.52	0.78

Table 10: STQ comparison on Cityscapes-VPS.

KITTI depth benchmark [74]. The validation set has 1,000 cropped images. Tab. 8 compares our method with previous methods that report their performances on it. Our method outperforms the previous methods by a large margin on all the metrics.

New Metrics on KITTI MOTS The KITTI MOTS benchmark changed their ranking metrics [56]. Tab. 9 reports the performance of ViP-DeepLab for the new metrics.

STQ on Cityscapes-VPS Table 10 also reports the STQ [83] performance comparison between our ViP-DeepLab and VPSNet [42].

C. Cascade-ASPP

Fig. 8 shows the architecture of Cascade-ASPP. It is used as the module *Dense Multi-scale Context* in the next-frame instance branch shown in Fig. 3. It cascades four ASPP modules with their outputs densely connected. The motivation of Cascade-ASPP is to dramatically increase the receptive field of the next-frame instance branch. As demonstrated in Tab. 3, Cascade-ASPP (*i.e.* DenseContext) improves the performances of video panoptic segmentation on Cityscapes-VPS compared with the single ASPP variant.

D. More Visualizations

We show more prediction visualizations in Fig. 9, Fig. 10, Fig. 11, and Fig. 12. We choose four sequences from 50 validation sequences of Cityscapes-DVPS, and the results are shown in Fig. 9 and Fig. 10. As each sequence contains only 6 frames, the figures show all the frames of the four sequences. Here, the video panoptic predictions demonstrate the results after the stitching algorithm, so each instance has the same instance ID in all the frames. SemKITTI-DVPS results are shown in Fig. 11 and Fig. 12. We present the results on two 16-frame video clips from the validation sequence. From the visualizations we can see that ViP-DeepLab is capable of outputting accurate video panoptic predictions and high quality depth predictions.

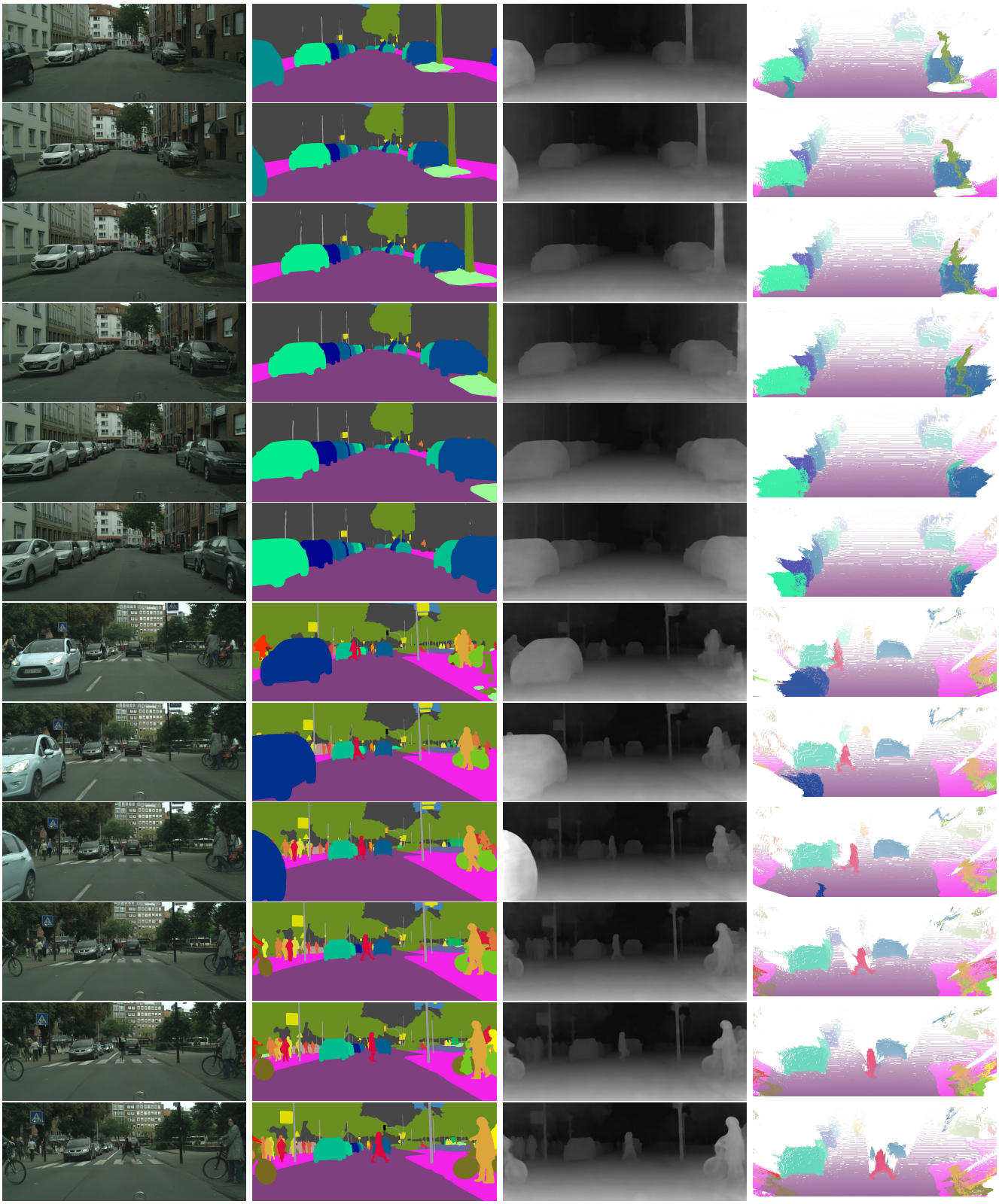

Figure 9: Prediction visualizations on Cityscapes-DVPS. From left to right: input image, temporally consistent panoptic segmentation prediction, monocular depth prediction, and point cloud visualization.

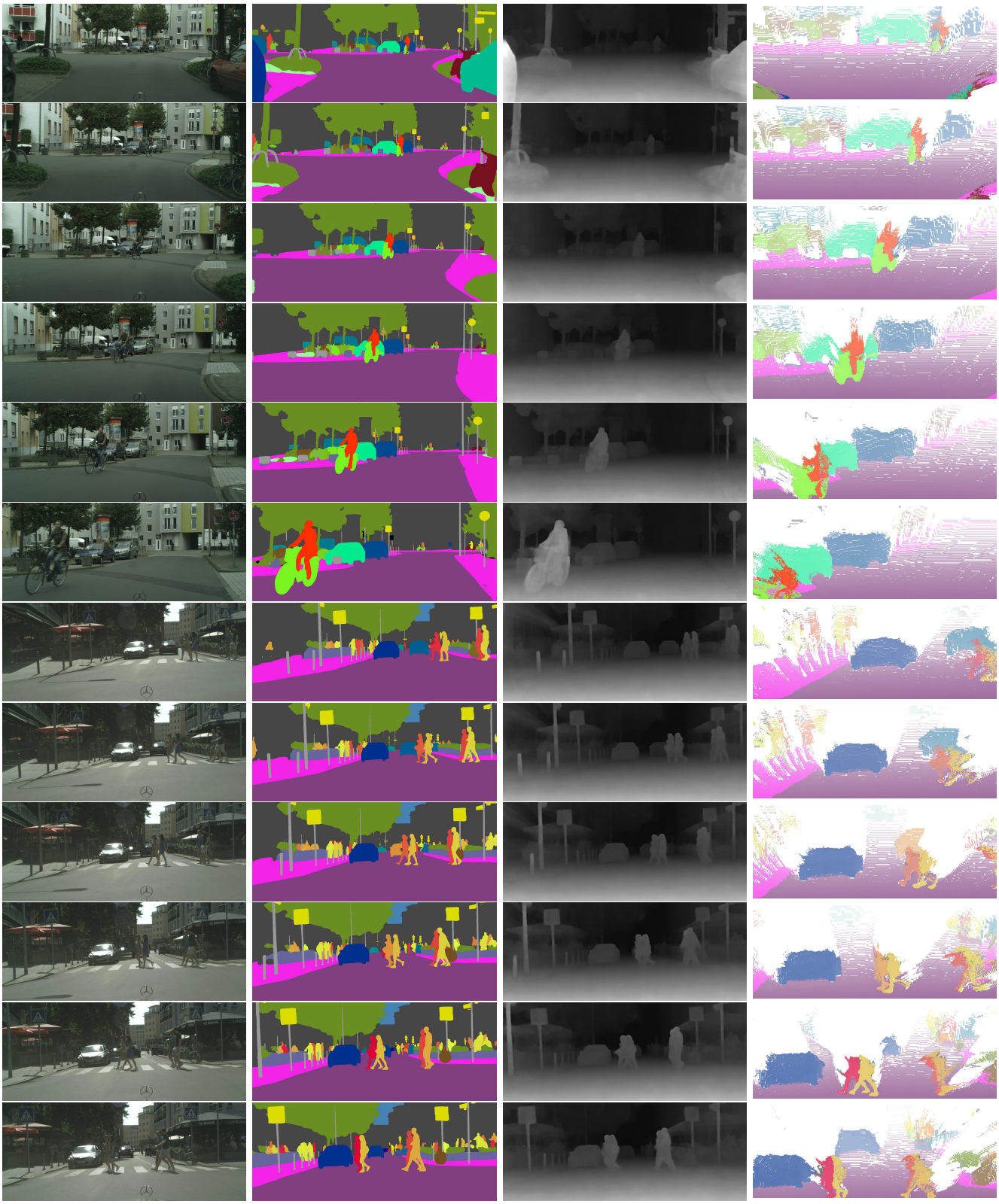

Figure 10: Prediction visualizations on Cityscapes-DVPS. From left to right: input image, temporally consistent panoptic segmentation prediction, monocular depth prediction, and point cloud visualization.

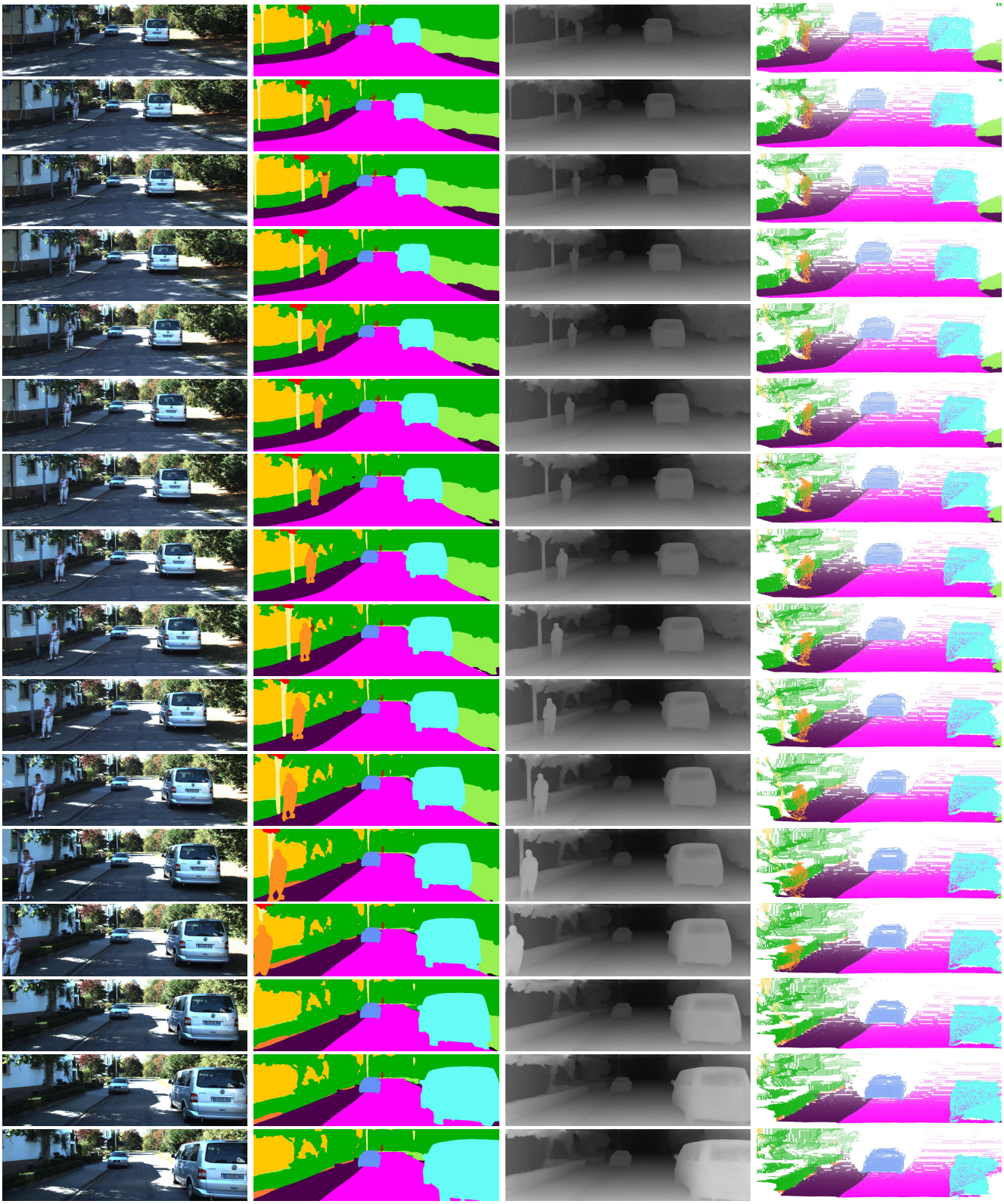

Figure 11: Prediction visualizations on SemKITTI-DVPS. From left to right: input image, temporally consistent panoptic segmentation prediction, monocular depth prediction, and point cloud visualization.

Figure 12: Prediction visualizations on SemKITTI-DVPS. From left to right: input image, temporally consistent panoptic segmentation prediction, monocular depth prediction, and point cloud visualization.