Supplementary Material for
SMD-Nets: Stereo Mixture Density Networks

Fabio Tosi[1] Yiyi Liao[2] Carolin Schmitt[2] Andreas Geiger[2]
[1]Department of Computer Science and Engineering (DISI), University of Bologna
[2] Autonomous Vision Group, MPI-IS / University of Tübingen
[1]{fabio.tosi5}@unibo.it [2]{firstname.lastname}@tue.mpg.de

Abstract

In this supplementary document, we first provide details of the new high-resolution stereo datasets in Section 1. Next, we present implementation details regarding training protocols and network architectures in Section 2. Finally, we show additional qualitative and quantitative results as well as computational cost analysis in Section 3.

1. Dataset Description

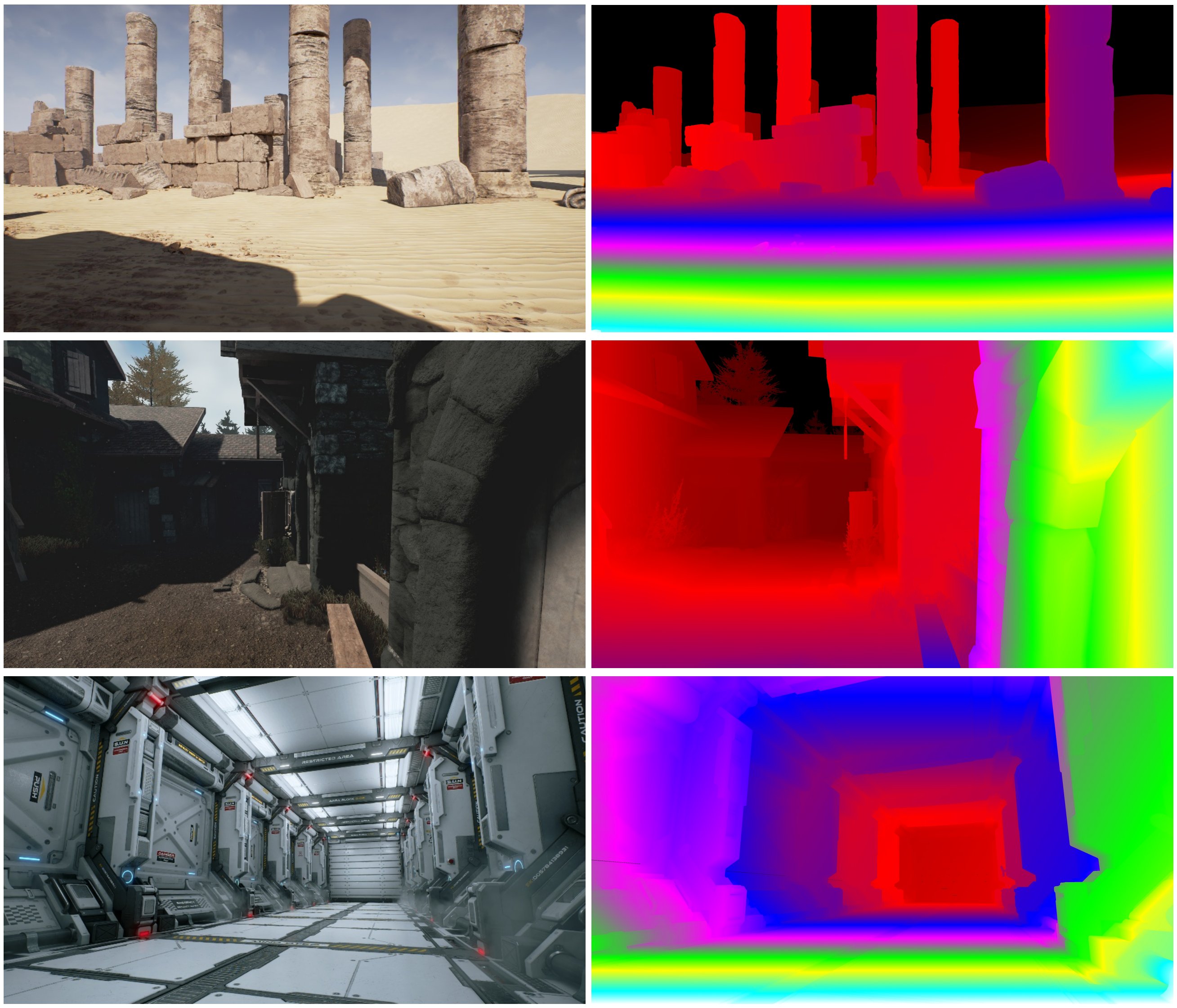
In this section, we provide details of the proposed high-resolution stereo datasets, including the synthetic dataset UnrealStereo4K and the real-world active dataset RealActive4K.

1.1. UnrealStereo4K

We create our synthetic dataset, UnrealStereo4K, using the popular game engine Unreal Engine combined with the open-source plugin UnrealCV [9].

Binocular Stereo: Our binocular UnrealStereo4K dataset consists of images of 8 scenes, including indoor and outdoor environments. We set the baseline to 20cm for indoor environments and 50cm for outdoor environments. We render 1000 stereo pairs at 3840×2160 resolution for each scene and divide them into 7720 training pairs, 80 validation pairs and 200 *in-domain* test pairs. We further render 200 *out-of-domain* test pairs from an unseen scene to evaluate the generalization ability of our method. For each stereo pair, we randomly sample a camera pose disabling the camera in-plane rotation and perform a sanity check to reject invalid camera poses. Specifically, we filter out images according to a minimum depth of 0.25m to avoid viewpoint obstruction. We also filter out invalid stereo pairs if the RGB images are too dark according to their average intensity. We further check the gradient of the disparity map to keep a stereo pair only if it contains non-trivial geometry structure. Fig. 1 shows some examples of our binocular UnrealStereo4K dataset corresponding to different rendered scenes.

Active Depth: For the active monocular UnrealStereo4K dataset we render 4 scenes using the intrinsic matrix of the IR camera on our structured light sensor. As our sensor is designed to be operated in indoor environments with a close working range, we randomly sample camera poses such that the synthetic active dataset contains larger disparity values. We then warp the reference dot pattern to each image to simulate the IR camera. We further convert the original RGB image to gray-scale and combine it with the warped dot pattern to simulate the ambient illumination, following [10]. Both our sensor and the reference dot pattern image are described in Sec. 1.2. The active dataset consists of 3856 training images, 40 validation images, 100 test images. Fig. 2 shows an example of the synthetic active image, the reference dot pattern and the ground truth disparity map.


(a) *In-domain* scenes

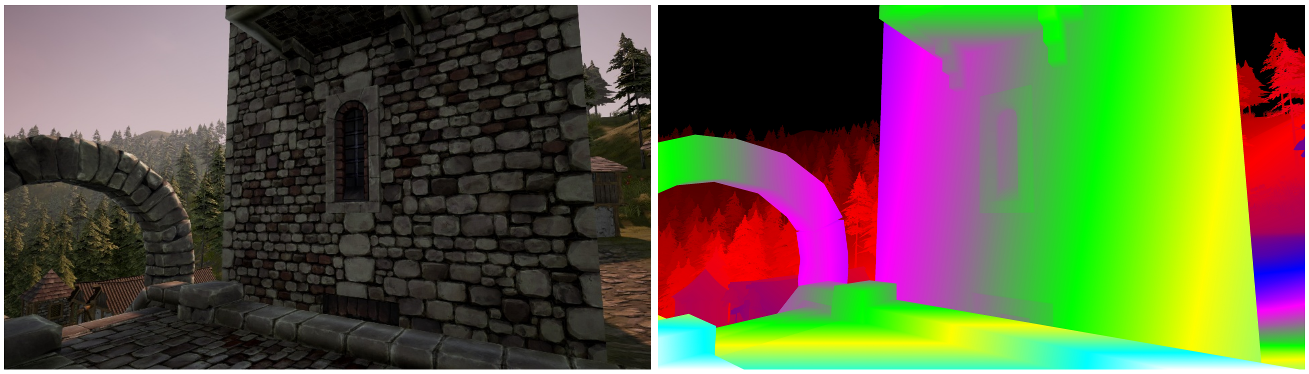

(b) *Out-of-domain* scene

Figure 1: **Examples of the Binocular UnrealStereo4K Dataset.** We show the left image of each stereo pair on the left and the corresponding ground truth disparity map on the right.

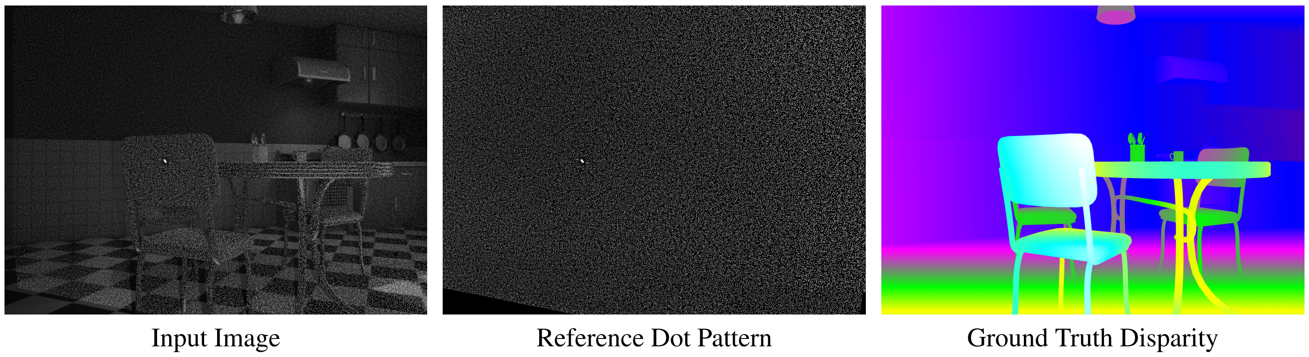


Figure 2: **Example of the Active UnrealStereo4K Dataset.** We warp the reference dot pattern to the reference image and combine it with the gray-scale image to simulate the IR camera. The reference dot pattern is a constant image of the laser projector and serves as the right image. The ground truth disparity map is in the coordinate of the reference image.

1.2. RealActive4K

Our real-world active dataset consists of 2570 images of an indoor room captured with a Kinect-like structured light sensor. The sensor has two main components. The first is a high-resolution monochrome camera (4112×3008 pixels) with an IR filter in front which we refer to as "IR camera". The second is a powerful IR laser projector with a *Diffractive Optical Element* (DOE) that casts $200,000$ random points into the scene. We calibrate our system geometrically by estimating the intrinsic camera matrix, lens distortion and the baseline between camera and projector. Further, we need to capture the reference dot pattern of the projector. For that purpose, we model the laser projector as a virtual camera that observes a constant dot pattern image which we can calculate from observations of a flat wall. Hereby, we assume the virtual camera to share the intrinsic matrix of the IR camera for simplicity.

We create the dataset by undistorting and rectifying the raw captures and also adjusting the intensity distribution of each image to fit the image statistics of the synthetic active UnrealStereo4K dataset. To obtain pseudo-ground truth as co-supervision during training, i.e. for adapting the features of the stereo network to the real captures, we perform Block Matching with left-right consistency check. We divide the resulting real-world active dataset into 2500 images for training, 20 for validation and 50 for testing. An example of the real active capture is shown in Fig. 3. For our experiments on real data, we jointly train on real and synthetic data as described below.

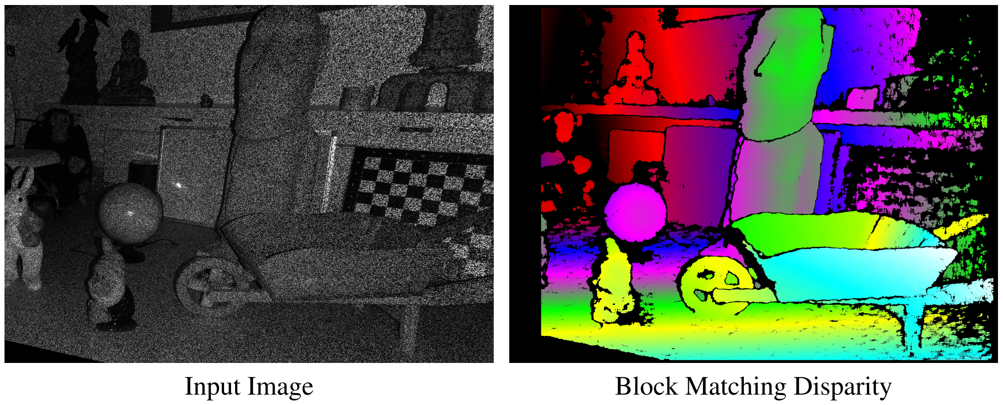


Figure 3: **Example of the RealActive4K Dataset.** On the left is the captured active image and the right shows the Block Matching disparity map used as pseudo-ground truth for training.

2. Implementation Details

We present details regarding the sampling strategy, stereo super-resolution, architecture designs and training/inference choices in this section.

2.1. Sampling Strategy

In Fig. 4, we show the effect of different dilation kernel sizes $\rho \times \rho$ applied to the boundary mask of a ground truth disparity map. We sample points based on the dilated boundary masks when the *Depth Discontinuity Aware (DDA)* sampling strategy is used during training.

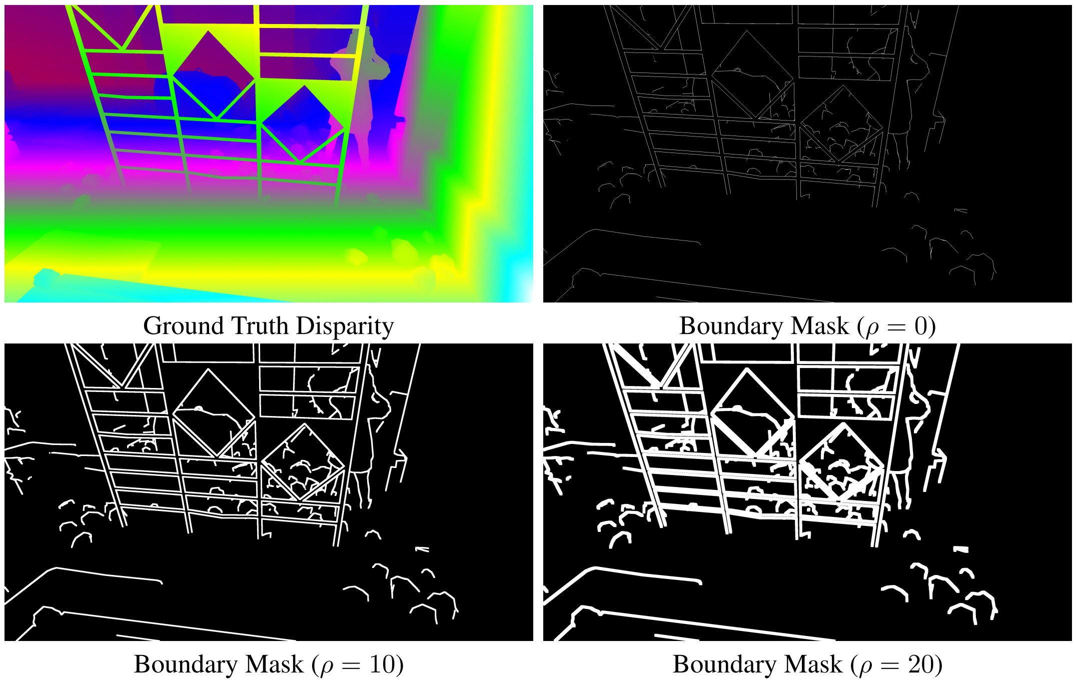

Ground Truth Disparity
Boundary Mask ($\rho = 0$)
Boundary Mask ($\rho = 10$)
Boundary Mask ($\rho = 20$)

Figure 4: **Boundary Masks on the UnrealStereo4K Dataset** using different size of dilation kernels. During training, given the total number of training points N, we randomly and uniformly select $N/2$ points from the domain of all pixels belonging to depth discontinuity regions (white) and $N/2$ points uniformly from the continuous domain of all remaining pixels (black).

2.2. Stereo Super-Resolution

Our continuous formulation allows us to exploit ground truth at higher resolution than the input $\mathbf{I}$, which we refer to as stereo super-resolution. Specifically, let us assume that a ground truth disparity map at resolution $W' \times H'$ pixels is available for the input image $\mathbf{I}$ with size $W \times H$, where $W' = sW$, $H' = sH$ and $s >= 1$. Given a 2D location $\mathbf{x}$ sampled from the continuous image domain $[0, W-1] \times [0, H-1]$, we retrieve its disparity at $s\mathbf{x}$ from the ground truth disparity map using nearest neighbor interpolation. We scale the interpolated disparity down by a factor of s such that the disparity range aligns with the input image.

2.3. Architecture

We now provide additional architectural details regarding the proposed SMD-Nets on different 2D and 3D stereo backbones. In particular, we follow the original implementation of three stereo networks: U-Net [3], PSM [1] and HSM [12]. While in the main paper we describe a stereo backbone that generates a feature map at $W \times H \times D$ resolution for simplicity, our model is able to exploit features from different resolutions/layers in practice. In our experiments, we adopt a different combination of feature maps depending on the specific backbone ψ_θ in order to capture both local and global context.

PSM: For the PSM [1] stereo backbone, we use feature maps of the reference image computed using Spatial Pyramid Pooling (SPP) modules as well as stereo matching cost probabilities extracted from stacked hourglass 3D convolutions on the constructed cost volume. More specifically, we perform bilinear interpolation on convolutional features of size $\frac{1}{4}W \times \frac{1}{4}H \times 32$ and the matching cost probabilities of dimension $\frac{1}{4}W \times \frac{1}{4}H \times 64$. Such design choice results into $D = 96$, the input dimension of our SMD Head. Notice that the softargmax operation on the cost volume, adopted as the last step of the original PSM network to obtain the final disparity value, is not needed as the proposed SMD Head directly regresses the parameters of the disparity distribution.

HSM: For the HSM [12] stereo backbone, we use multi-level features extracted from the reference image by the Feature Pyramid Encoder, which consists of an encoder-decoder architecture with skip connections and multiple SPP modules. Similarly to PSM, we concatenate features with stereo matching cost probabilities computed on a single low resolution cost volume. Specifically, we interpolate features at $\frac{1}{2}W \times \frac{1}{2}H \times 32$, $\frac{1}{8}W \times \frac{1}{8}H \times 64$, $\frac{1}{16}W \times \frac{1}{16}H \times 128$, $\frac{1}{32}W \times \frac{1}{32}H \times 128$ resolution and matching cost probabilities of dimension $\frac{1}{8}W \times \frac{1}{8}H \times 16$. This results into $D = 368$.

U-Net: For the U-Net [3] stereo and monocular backbone, we simply concatenate features of the last 4 levels of the decoder in order to exploit coarse-to-fine information. In particular, we bilinear interpolate feature maps of size $W \times H \times 16$, $\frac{1}{2}W \times \frac{1}{2}H \times 32$, $\frac{1}{4}W \times \frac{1}{4}H \times 64$ and $\frac{1}{8}W \times \frac{1}{8}H \times 128$. This results into $D = 240$.

2.4. Training

Here, we report the training details for each dataset and different models. We implement our framework in PyTorch [8] and use Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as optimizer [5]. We train all models using a single NVIDIA V100 GPU. We scale the ground truth disparity to $[0, 1]$ for each dataset for numerical stability. We predict μ_1, μ_2 in the range of $[0, 1]$ accordingly. π is in $[0, 1]$ by definition. We also predict b_1, b_2 in $[\epsilon, 1]$, where we set $\epsilon = 1e^{-2}$. Therefore we adopt a sigmoid function as activation of the last layer of our SMD Head and clip b_1 and b_2 to be larger or equal to ϵ.

UnrealStereo4K: For the UnrealStereo4k dataset, we use the entire training set (7720 stereo pairs) for training and evaluate on both *in-domain* (200 stereo pairs) and *out-domain* (200 stereo pairs) test sets. We scale the ground truth disparity to $[0, 1]$ dividing by D_{max} for numerical stability, where we set $D_{max} = 256$. We train all networks end-to-end from scratch for 200 epochs using a batch size 12 for HMS and U-Net backbones, and 60 epochs with batch size 4 for PSM due to memory limitations. We experimentally find that PSM converges faster and thus requires less training epochs.

During training, we use random crops of size 512×384 from input images at 960×540 while keeping the ground truth at the original full resolution 3840×2160. From each crop we sample $50,000$ training points, regardless of the sampling strategy. Moreover, we perform chromatic augmentations on the fly, including random brightness, gamma and color shifts sampled from uniform distributions ($[0.5, 1.5]$ for brightness, $[0.8, 1.1]$ for gamma and $[0.8, 1.2]$ for contrast) to both the reference and the target images. We further augment the training procedure by flipping the two images horizontally and vertically while adapting the ground truth disparities (left and right) accordingly.

For the active monocular dataset we need to adjust some training details since the projected dots get indistinguishable from one another or dissolve to noise when using a low-resolution input (e.g. 960×540). Therefore, we use random crops of 1024×768 pixels from input images of size 2056×1504, set D_{max} to 786 and choose batch size 4 to stay in memory limits. We also do not use any chromatic augmentations on the monochrome input images. Furthermore, we follow [10] and remove intensity variations caused by different reflection properties of the materials in the scene. Specifically, we apply *local contrast normalization* (LCN) to the monochrome input images and concatenate both the original and the LCN image as input left image for the network. The right image is given by the pre-calibrated constant reference dot pattern image. See Fig. 2 for an example. Apart from those adjustments, the training and testing procedure is the same as for the binocular stereo dataset.

KITTI 2015: For KITTI, we fine-tune the models pre-trained on UnrealStereo4K on 160 stereo pairs of the KITTI 2015 training set [7]. For this, we use random crops of size 512×256 from the original KITTI resolution while scaling the ground truth using $D_{max} = 256$ as scale factor. Since the provided ground truth disparities are sparse, we rely on the naïve random sampling strategy to train our models. Similarly to UnrealStereo4K, we set 12 and 4 as batch size for HSM and PSM respectively and train them for 1000 and 500 epochs. Then, we validate all models on the remaining 40 images of the KITTI 2015 training set. Similarly to [2], we use the model based on the PSM backbone with the best performance on validation set for submission to the official online benchmark.

RealActive4K: We fine-tune active depth models jointly on the real-world RealActive4K captures and the synthetic images of the active monocular UnrealStereo4K data. This allows our network to (i) accurately reason about object boundaries present in the synthetic data, (ii) adapt its features to both synthetic and real inputs and (iii) learn from a larger and more diverse set of training images compared to the relatively small amount of real images in the RealActive4K dataset.

Since the pseudo-ground truth obtained from Block Matching does not provide reliable samples near disparity edges, we avoid any possibly wrong disparity estimates by eroding the image areas with positive Block Matching predictions by a conservative margin. Then we use the naïve random sampling strategy on the real-world captures. To contrast this lack of samples near disparity boundaries, we use DDA sampling on all synthetic input images. We fine-tune all networks for 100 epochs.

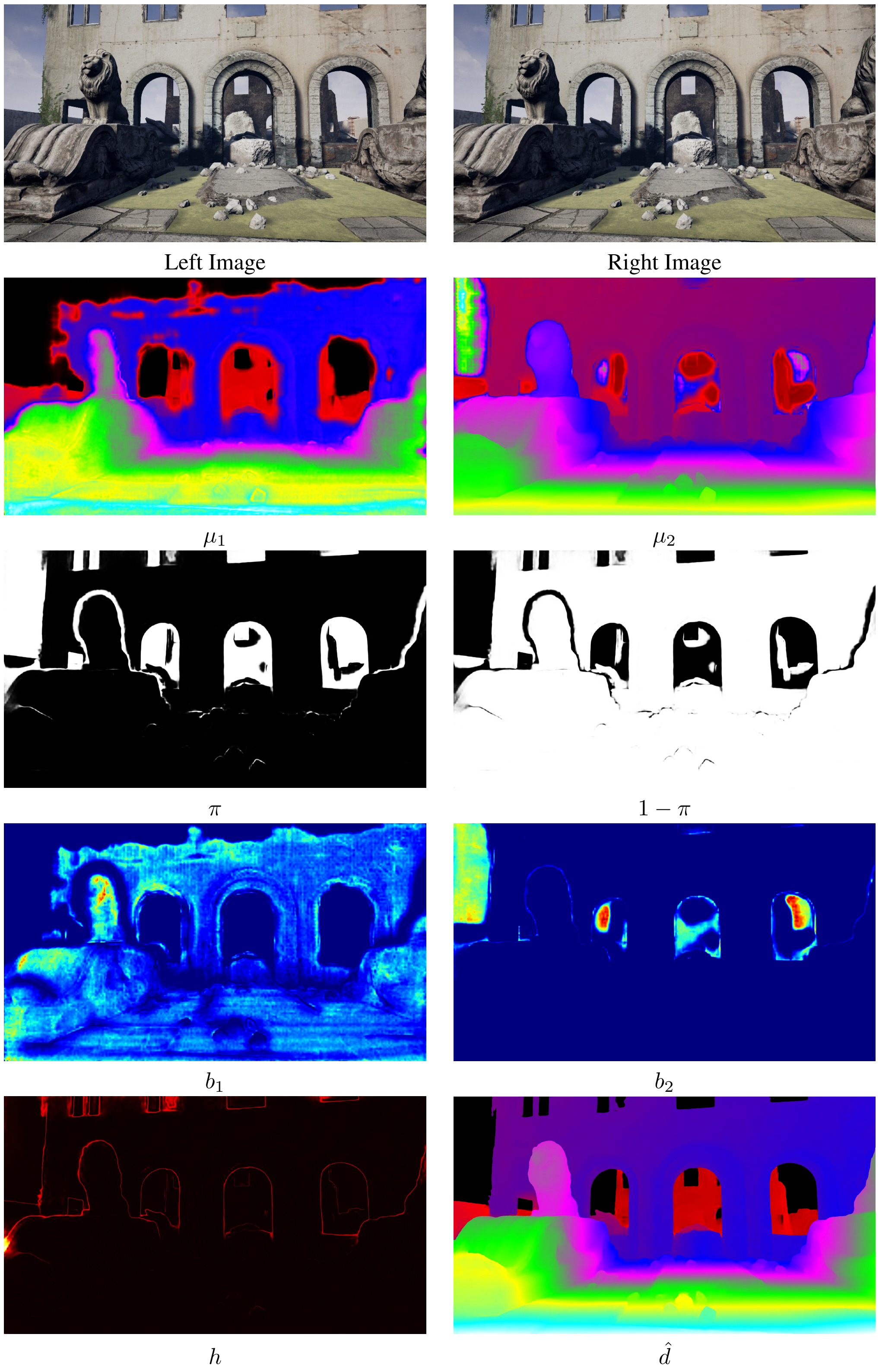


Figure 5: **Bimodal Output Representation**. We visualize a stereo pair of the proposed UnrealStereo4K dataset, the estimated five-dimensional vector (π, μ_1, b_1, μ_2, b_2), the uncertainty h computed by evaluating the differential entropy of the continuous mixture distribution and the final disparity map $\hat{d}$.

2.5. Inference

During testing, we select the mode with the highest density value as the predicted disparity $\hat{d}$, which is scaled back to the original disparity range by multiplying D_{max}. Given an (arbitrary) output resolution, we leverage our model to predict disparity values at each pixel. In Fig. 5 we visualize all the estimated quantities of our bimodal formulation.

3. Additional Experimental Results

In this section, we first present additional quantitative results using the HSM backbone, for both the ablation study of the output representation and the evaluation on the KITTI 2015 dataset. Next, we provide extensive qualitatively evaluations on all datasets considered in the main paper. We further analyze the computational cost of our method.

3.1. Quantitative Results

Output Representation: Tab. 1 is provided as a complement to Tab. 1 in the main paper for an extensive evaluation. More specifically, we report results on the binocular UnrealStereo4K stereo dataset using different output representations, adopting HSM [12] as backbone. Note that our bimodal representation outperforms both the direct regression and the unimodal formation using the HSM backbone. In combined with Tab. 1 in the main paper, the results suggest that our SMD Head can be advantageously combined with a wide class of stereo backbones to improve the disparity prediction at object boundaries.

Dim.	*SEE*$_3$			*SEE*$_5$			*EPE*	
	Avg	$\sigma(1)$	$\sigma(2)$	Avg	$\sigma(1)$	$\sigma(2)$	Avg	$\sigma(3)$
1	1.85	37.59	20.49	1.73	35.55	19.27	1.11	5.47
2	2.63	43.49	27.12	2.51	41.66	25.98	2.07	8.90
5	**1.40**	**26.78**	**12.65**	**1.26**	**24.70**	**11.40**	**1.00**	**4.42**

Table 1: **Output Representation** analysis on the binocular UnrealStereo4K test set. "Dim." refers to the output dimension of the SMD Head where 1 indicates the point estimate d, 2 the unimodal output representation (μ, b) [4] and 5 our bimodal formulation $(\pi, \mu_1, b_1, \mu_2, b_2)$.

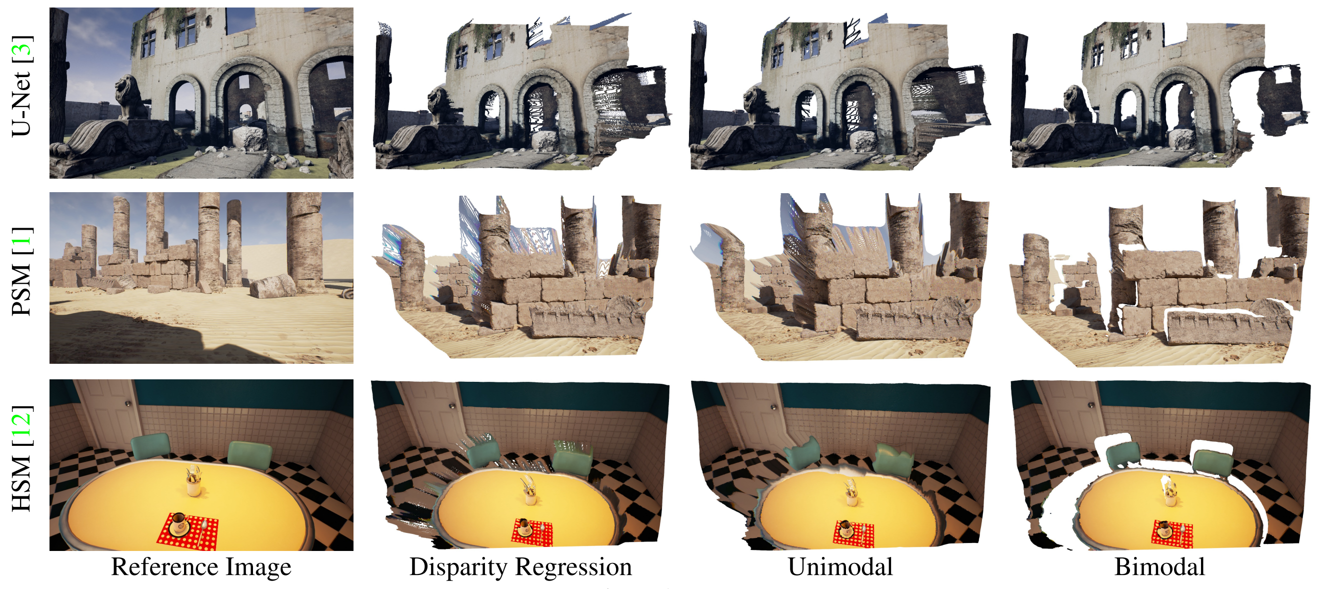
KITTI 2015: Tab. 2 is provided as a complement to Tab. 5 in the main paper. In particular, we report results on the KITTI 2015 validation set adopting HSM [12] as stereo backbone. Here, we compare our model with the original HSM [12] stereo network and HSM [12] trained following [2]. Consistent with results using the PSM backbone, our method built on the HSM backbone leads to better *SEE* and *EPE* compared to both the original HSM backbone and [2].

Method	*SEE*$_3$			*SEE*$_5$			*EPE*	
	Avg	$\sigma(1)$	$\sigma(2)$	Avg	$\sigma(1)$	$\sigma(2)$	Avg	$\sigma(3)$
HSM [12]	1.37	29.13	14.80	1.25	25.83	13.63	0.79	3.25
HSM [12] + CE + SM [2]	1.46	31.48	15.63	1.33	27.86	14.49	0.85	3.55
HSM [12] + Ours	**1.18**	**20.30**	**9.71**	**1.04**	**17.23**	**8.86**	**0.74**	**2.67**

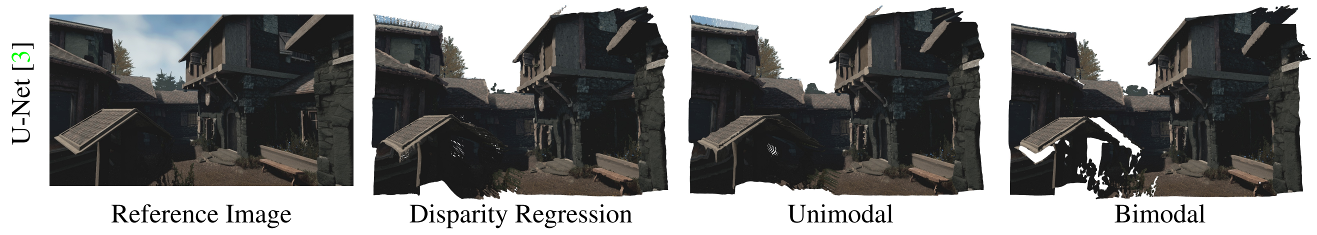
Table 2: **Comparison on the KITTI 2015 Validation Set** using boundaries extracted from instance segmentation masks to evaluate at depth discontinuities (given the sparse KITTI ground truth, depth discontinuities cannot be derived from the ground truth depth maps).

3.2. Qualitative Results

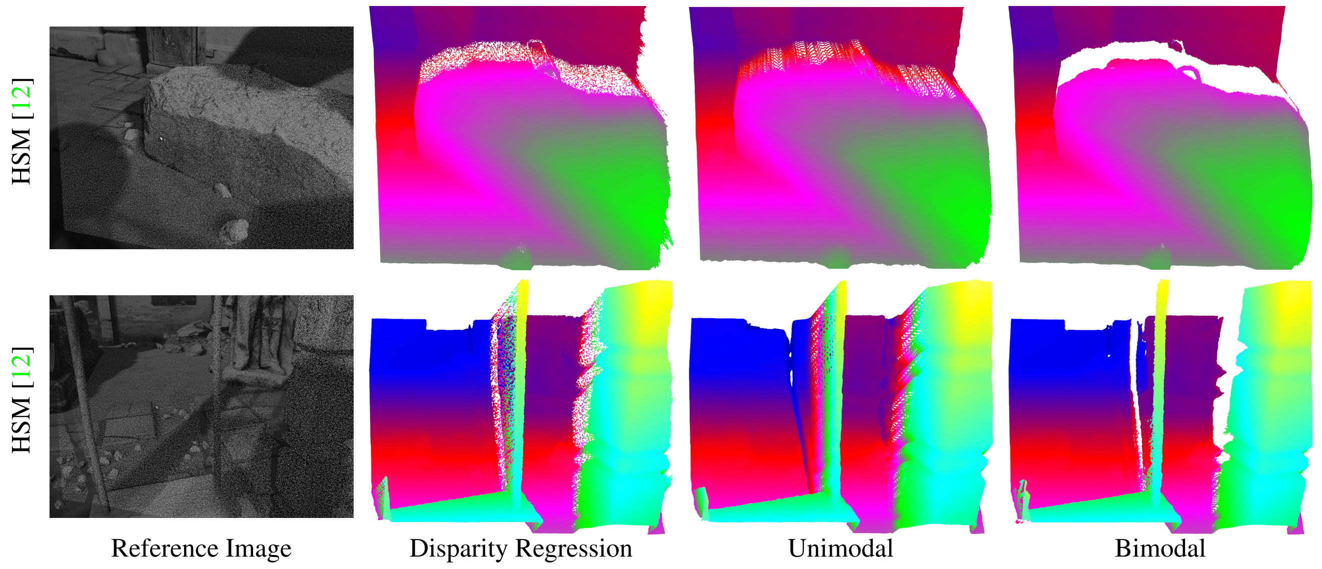
Output Representation: In Fig. 6 we complement Tab. 1 in the main paper with point cloud visualizations on the UnrealStereo4K dataset. Specifically, we show results obtained using different backbones on passive stereo, monocular and active depth estimation tasks. For each backbone, we compare standard disparity regression (i.e., disparity point estimate), a unimodal Laplacian distribution and our bimodal Laplacian mixture distribution.



(a) Binocular Stereo



(b) Monocular Depth



(c) Active Depth

Figure 6: **Point Cloud Comparison of Output Representations** on the UnrealStereo4K dataset. We show outcomes of different 2D and 3D backbones on (a) binocular stereo, (b) monocular depth and (c) active depth. Note that our bimodal representation notably alleviates bleeding artifacts near object boundaries compared to both disparity regression and the unimodal formulation.

UnrealStereo4K: We present Fig. 7 as a complement to Fig. 5 in the main paper. Specifically, Fig. 7 compares the predicted disparity maps of different strategies which aim to address the over-smoothing problem.

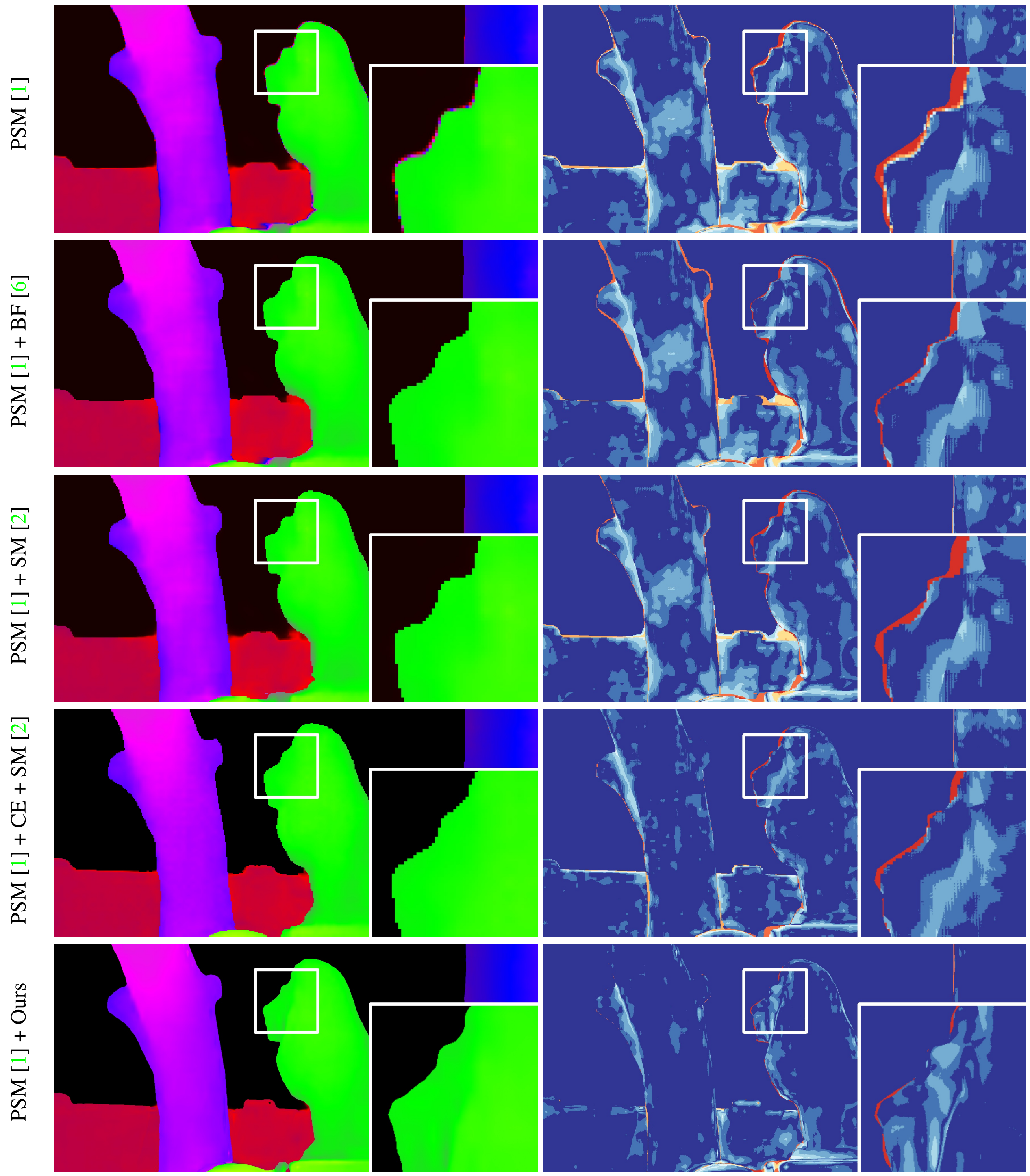


Figure 7: **Qualitative Results on UnrealStereo4K.** The left column shows the predicted disparity maps while the right depicts the corresponding error maps. We zoom-in a patch in all images to better perceive details near depth boundaries.

KITTI 2015: We also report qualitative results on stereo pairs belonging to the KITTI 2015 dataset [7]. More specifically, in Fig. 8 we show disparity maps predicted by our mixture density formulation using PSM [1] as backbone on testing images of the online KITTI 2015 benchmark. Fig. 9, instead, depicts point cloud comparisons of the same model with respect to the original PSM [1] network and [2]. It can be observed how our model allows to notably alleviate bleeding artifacts compared to the original PSM [1] network and to obtain more accurate results in distant regions compared to [2].

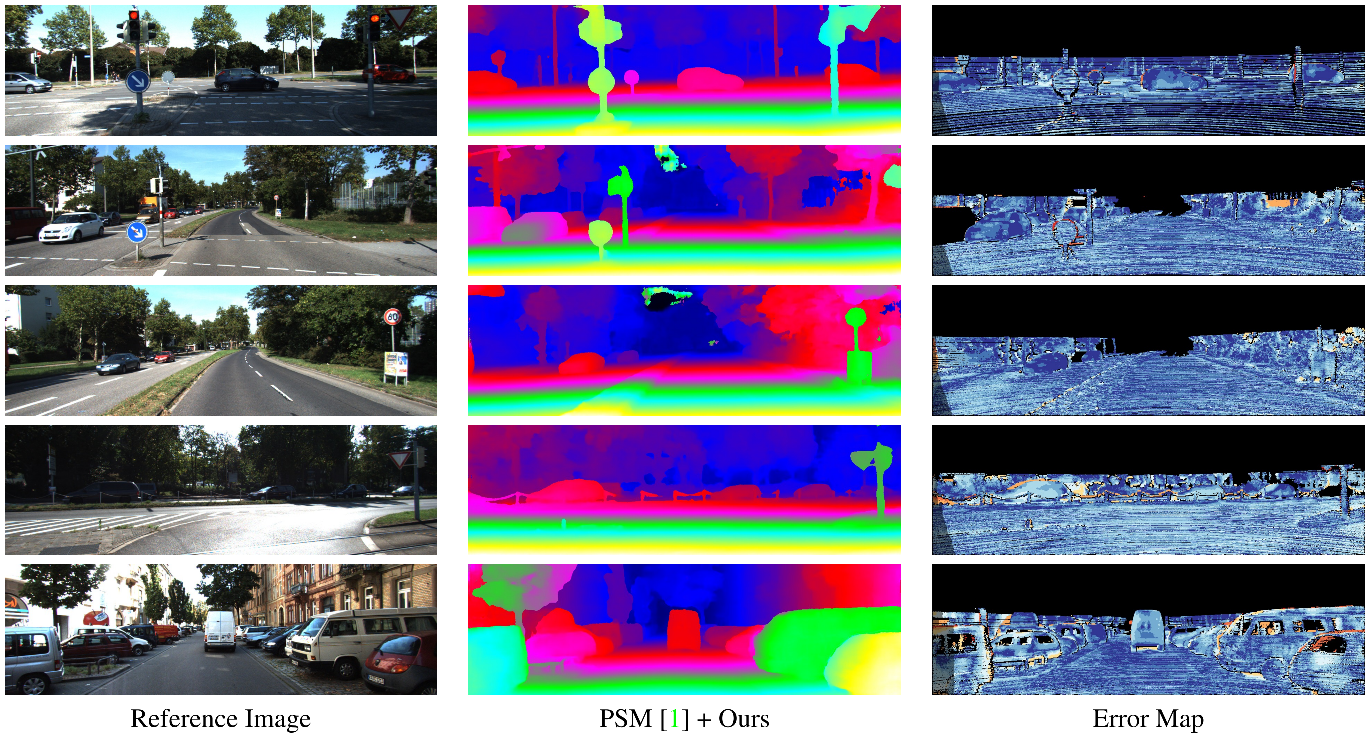


Figure 8: **Qualitative Results on KITTI 2015 Test Set.** Error maps are downloaded from the online KITTI 2015 benchmark.

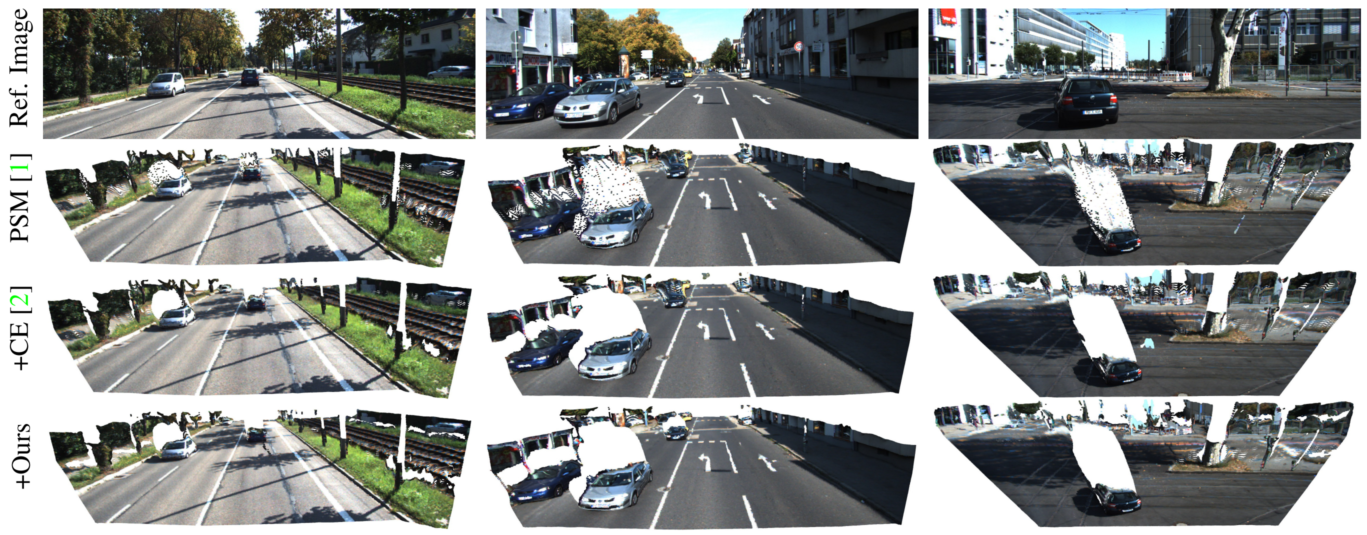


Figure 9: **Point Cloud Comparison on KITTI 2015 Validation Set.** The first row shows the reference image and the rest shows the point cloud comparison. Note that [2] produces bleeding artifacts at far regions (e.g., black car at far distance in the middle column) while our method predicts sharp object boundaries at both near and far regions. Please zoom in for details.

Middlebury v3: We also show qualitative results concerning generalization capability on the Middlebury v3 dataset [11] in Fig. 10. In particular, we compare the original PSM [1] network to our bimodal mixture representation using PSM as backbone. It can be clearly perceived that our strategy recovers more accurate and sharper details on boundary regions.

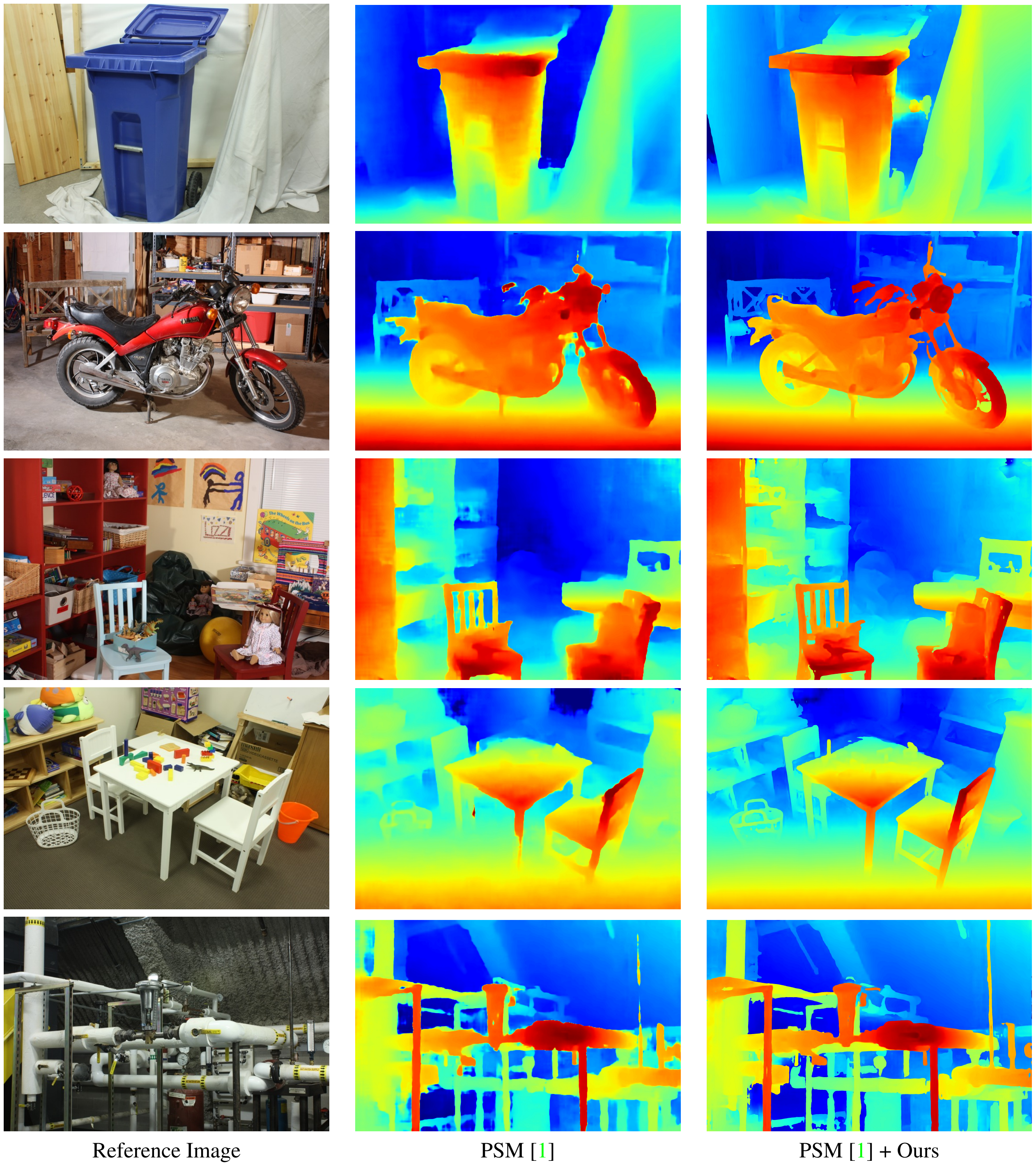


Figure 10: **Generalization on Middlebury v3** [11] using the original PSM [1] and our SMD-Net, both trained on stereo pairs of our UnrealStereo4K training set. Following the common pratice for the Middlebury dataset, we adopt the colormap 'jet' for disparity visualization.

RealActive4K: Fig. 11 demonstrates more qualitative results on the RealActive4K dataset. We show the projected disparity map that we use for co-supervision during training and a point cloud comparison of our bimodal output representation and standard disparity regression.

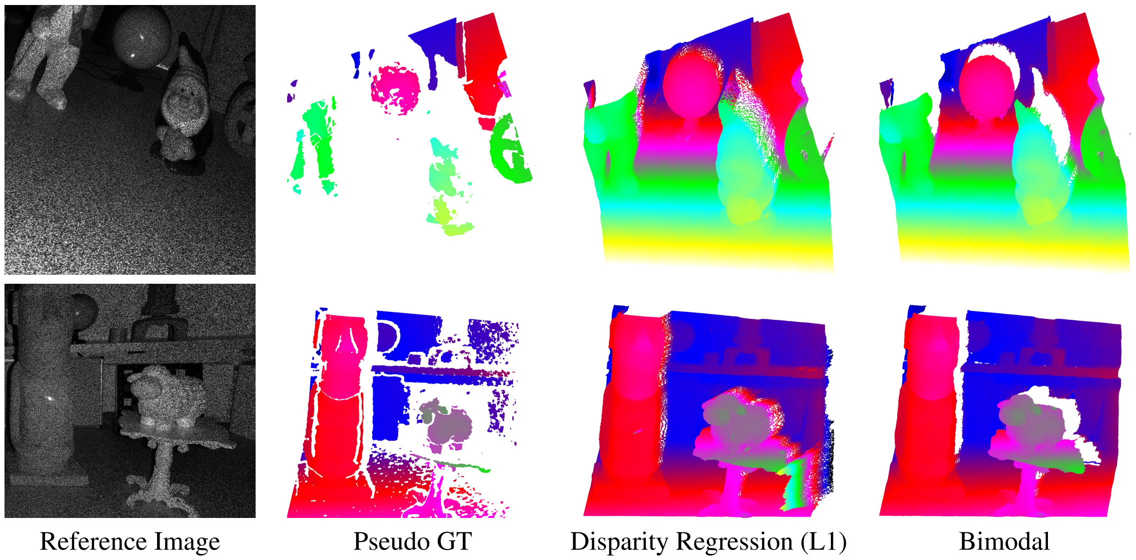


Figure 11: **Generalization on RealActive4K.** Note that despite the pseudo-ground truth disparity not providing information on disparity edges, our model is able to predict sharp object boundaries for real-world input images while at the same time inferring plausible disparity values in regions where the classical method used for supervision fails entirely.

3.3. Computational Cost Analysis

Training Time and GPU Memory: We compare the training time and memory footprint of our SMD-Nets to the original backbones and [2] in Tab. 3. Specifically, we evaluate the memory consumption on a single NVIDIA V100 GPU with 32 GB GPU RAM and the average time of a single iteration (forward + backward) during training. As can be seen, the cross entropy loss (CE) [2] noticeably increases the memory footprint and training time on both, PSM and HSM backbones. The main reason is that the CE loss requires the network to trilinearly interpolate a matching cost probability distribution to the full resolution $W \times H \times D_{max}$ with D_{max} denoting the maximum disparity. In contrast, our simple bimodal representation predicts a compact five-dimensional vector at every point and thus is more efficient to train. Note that on the PSM backbone our bimodal formulation requires only a marginal increase of the memory and training time compared to the original PSM, while it is more expensive on the HSM backbone. This is due to the different feature dimension D we use for the different backbones ($D = 96$ for PSM and $D = 368$ for HSM). We remark that our method performs favorably with a relatively small D. We leave the selection of a good D to balance performance and training time for future research.

Method	Memory (GB)	Time (s)	Method	Memory (GB)	Time (s)
PSM [1]	23.80	2.854	HSM [12]	17.65	0.578
PSM [1] + CE [2]	31.73	4.835	HSM [12] + CE [2]	27.11	7.096
PSM [1] + Ours	24.78	2.889	HSM [12] +Ours	27.54	0.983

Table 3: **Memory Usage and Time Consumption During Training.** We present the memory footprint in Gigabytes (GB) and the average time for one iteration (forward + backward) in seconds.

Inference Time: We now analyze the average time for estimating a single disparity map using the proposed SMD-Nets. Tab. 4 compares the inference time of our SMD-Net at different output resolutions based on the HSM backbone. Note that the inference time of the backbone remains the same as the input resolution is fixed. Though the evaluation time of the SMD Head increases wrt. the output resolution, our method is still able to estimate a disparity map at 3840×2160 resolution in about 3.6 seconds. Tab. 5 shows the inference time of predicting an output disparity map at 3840×2160 resolution using different backbones. Here the inference time depends on the input dimension D of the SMD Head, where a smaller D leads to faster inference.

Output Resolution	T(Stereo Backbone)	T(SMD Head)
960×540	0.030	0.206
1920×1080	0.030	0.856
3840×2160	0.030	3.524

Table 4: **Inference Time** using the HSM backbone at different output resolutions. $T(\cdot)$ denotes the average time in seconds.

Stereo Backbone	T(Stereo Backbone)	T(SMD Head)
U-Net [3]	0.012	3.070
PSM [1]	0.575	2.402
HSM [12]	0.030	3.524

Table 5: **Inference Time** at 3840×2160 output resolution using different stereo backbones. $T(\cdot)$ denotes the average time in seconds.

References

[1] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 8, 9, 10, 11, 12, 13

[2] Chuangrong Chen, Xiaozhi Chen, and Hui Cheng. On the over-smoothing problem of cnn based disparity estimation. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 5, 7, 9, 10, 12

[3] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 4, 5, 8, 13

[4] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 7

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 5

[6] Ziyang Ma, Kaiming He, Yichen Wei, Jian Sun, and Enhua Wu. Constant time weighted median filtering for stereo matching and beyond. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, December 2013. 9

[7] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5, 10

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 5

[9] Weichao Qiu, Fangwei Zhong, Yi Zhang, Siyuan Qiao, Zihao Xiao, Tae Soo Kim, Yizhou Wang, and Alan Yuille. Unrealcv: Virtual worlds for computer vision. *ACM Multimedia Open Source Software Competition*, 2017. 1

[10] Gernot Riegler, Yiyi Liao, Simon Donne, Vladlen Koltun, and Andreas Geiger. Connecting the dots: Learning representations for active monocular depth estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 5

[11] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 11

[12] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 5, 7, 8, 12, 13