Semantic View Synthesis Supplementary Material

Anonymous ECCV submission

Paper ID 1653

We outline this supplementary material as follows:

1. We show additional visual comparisons with competing baseline methods.
2. We describe the network configuration of the proposed model. The source code and the pre-trained model will be made available to fully reproduce our results in the paper and the supplementary material.
3. We provide implementation details, including training and testing procedures, as well as the implementation of baseline methods.
4. We present runtime analysis and the limitation of our method.

1 Additional visual results

We present additional qualitative results in the attached HTML webpage. Please refer to the file main.html. Our visual results consist of three main components.

(1) Visual Comparison : We compare the novel view videos synthesized by our method and other baseline methods on the ADE20K [8], ADE20K-outdoor, NYU [3] datasets. We demonstrate these results in the *Visual Comparison* webpage.

(2) Background inpainting : We explore alternative methods for handling the dis-occluded regions when rendering the scene at novel views. According to the disparity image, we render the novel view images by standard backward warping the synthesized color image. We then fill the missing pixels using different existing in-painting methods. We experiment with the following three methods. First, we apply simple diffusion-based inpainting for each frame independently. Second, we apply a learning-based image inpainting model (GatedConv) [6] to fill in the dis-occluded regions independently for each frame. Third, we use the 3D Ken Burns [4]. The 3D Ken Burns method inpaints only one single view and may reveal additional dis-occluded regions in other views. The results are shown in the *Background inpainting* webpage.

(3) Number of depth layers in MPI: We present the visual results of using a different number of depth layers (32, 64, 128) in our Multi-plane Image representation in the *Number of depth layers* webpage.

2 Network architecture

We provide the detailed network architectures of our two-stream image/disparity generator in Figure 1. We adopt the design from stereo magnification [9] for the MPI generator, which can be found in Table 1 in [9].

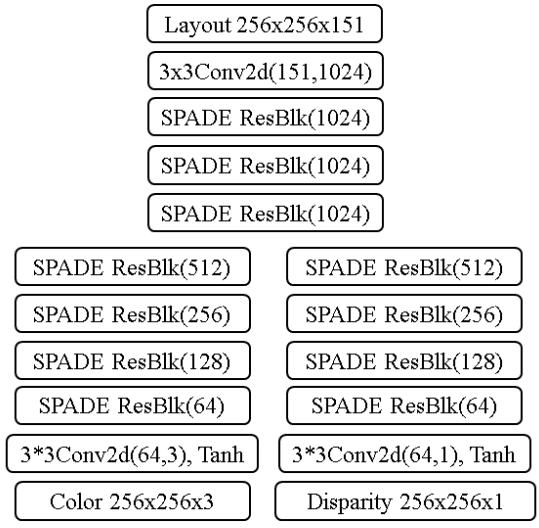

Fig. 1: **Network Architecture of image/disparity generator.** We modify the SPADE model [5] to a two-stream color/disparity generators. One stream serves as the *color generator*, while the other one is the *disparity* generator. The two-stream generators share the first three SPADE-style ResNet blocks.

3 Training

Image/disparity generator. Our training triple for image/disparity generator consists of a semantic layout, a color image, and a disparity image. The semantic layout is a 2D one-hot representation of 151 classes in ADE20K [8] dataset. We generate the (pseudo) ground truth disparity image by the pretrained MiDaS [2] predictor. Since the disparity maps predicted by MiDaS are *relative* prediction with unknown scale and shift, we further use the prediction from DPSNet [1] to transform the relative disparity map into *absolute* disparity. Specifically, for each frame in RealEstate10K, we feed a stereo pair of images with the corresponding camera intrinsic value, and the camera poses to the DPSNet. The DPSNet predicts the absolute disparity map. For each frame, we calculate the scaling and shifting transformation parameters between the MiDaS depth prediction and the DPSNet depth prediction values, and transform the MiDaS prediction to the absolute disparity image as our target disparity. Following DPSNet, the disparity image has a range of 0 to 1, representing the disparity value from 1/2 to 32 equidistantly distributed.

We use a sequential training procedure for the image/disparity generator. The training consists of two steps: (a) semantic image synthesis: we first train the shared blocks and the color stream to generate the color image from the semantic layout; (b) semantic disparity synthesis: we then fix the shared blocks and the color stream, and only train the disparity stream, to generate the disparity image based on the semantic layout. For (a) semantic image synthesis, we borrow the

pretrained model of [5]. The network is trained by 20,000 layout/image tuples in the ADE20K training set, optimized by Adam optimizer for 200 epochs with a learning rate of 0.0001 and the batch size of 32. The training losses include a VGG loss of weight 10, a GAN feature matching loss of weights 10 and a GAN loss of weight 1. For (b) semantic disparity synthesis, we train the network using 150,000 layout/disparity tuples sampled from the RealEstate10K dataset. We use Adam optimizer to train the network with a L_1 loss, 150,000 iterations with a learning rate of 0.0002 and the batch size of 1.

MPI predictor. The MPI predictor takes the source view image and the source view disparity as input, and predicts the MPI, which is then projected to novel views. Our training pair for MPI predictor consists of two images with camera poses sampling from the videos of the RealEstate10K dataset [9]. We follow the sampling method in [9]. We first randomly sample an interval value from 1 to 10, then we sample a 10-frame sequence with the interval value. Finally, we randomly sample 2 frames from the sequence as our training pair and the source view disparity is the same as Section 3. We use Adam optimizer to train the network with a L_1 loss for 150,000 iterations with a learning rate of 0.0002 and the batch size of 1. Our MPI representation consists of 128 depth planes with the depth equally distributed from 1m to 100m according to inverse depth. We discretize the source view disparity to generate the alpha images based on the same depth values.

4 Testing

NYU dataset. As we train our model using the semantic classes from the ADE20K dataset, we are interested in validating how well the model generalizes to different semantic layouts. We produce the testing results on the NYU dataset by mapping the classes of NYU to the classes of ADE. Each class is mapped to the class that has the same class name. For the "objects" class in NYU, we map it to the "box" class in the ADE20K dataset.

RealEstate dataset. We test our model using the testing videos in RealEstate10K [9] dataset. We use a semantic segmentation network PSPNet [7] pretrained on the ADE20K [8] dataset to obtain the testing semantic layout from the images. The results are shown in Figure 1 in the main paper.

5 Baseline Approaches

Direct (U-Net). The model takes the 151-class semantic layout as input, and predicts a 72-layer MPI. The network is following the same U-Net in Zhou et al. [9]. We use Adam optimizer to train the network with a L_1 loss, 300,000 iterations, learning rate 0.0002, and batch size 1. The loss is applied between the projected novel view $\hat{x}^n$ and the ground truth novel view x^n.

Direct (SPADE). The model takes the 151-class semantic layout as input, and predicts a 72-layer MPI. The network follows the design of Park et al. [5]. We

use Adam optimizer to train the network for 100,000 iterations, learning rate 0.0002, and batch size 1. We apply the loss between the projected novel view $\hat{x}^n$ and the ground truth novel view x^n. Specifically, we apply the following loss terms, a hinge GAN loss of weight 1, a GAN feature matching loss of weight 10, a VGG perceptual loss of weight 10, and a KL divergence loss of weight 0.5, following [5]. The main differences between the Direct (SPADE) method and the Direct (U-Net) method are the use of the more advanced generator and training losses for synthesizing photorealistic images.

Cascade (MPI). First, we use the SPADE model [5] to generate the color image using a 151-class semantic layout as input. We use the model architecture from SPADE [5]. Second, we train an MPI generator that takes color images as input, and predicts a 128-layer MPI. The network is following the same U-Net in Zhou et al. [9]. We use Adam optimizer to train the network with a L_1 loss, 100,000 iterations, learning rate 0.0002, and batch size 1. We apply the loss between the projected novel view $\hat{x}^n$ and the ground truth novel view x^n.

Cascade (KB). We use the SPADE model [5] to generate the color image using a 151-class semantic layout as input. We then apply the model provided by Niklaus et al. [4].

Difference between the MPI. Note that the MPIs used in Direct (U-Net), Direct (SPADE), Cascade (MPI) and our model have different settings. Direct (U-Net) and Direct (SPADE) predicts x^s_{FG}, x^s_{BG}, $\{w^s\}$ and $\{\alpha^s\}$. Cascade (MPI) predicts x^s_{BG}, $\{w^s\}$ and $\{\alpha^s\}$, directly using the x^s_{FG} generated by SPADE [5]. Our model only predicts x^s_{BG} and $\{w^s\}$, directly using the x^s_{FG} and d^s as $\{\alpha^s\}$ generated by our two-stream image/disparity generator.

Dis-occlusion handling baselines. We explore alternative methods for handling the dis-occluded regions when rendering at novel views. We applied standard backward warping to project $\hat{x}^s_{FG}$ by $\hat{d}^s$ to the novel view. We then apply Diffusion or GatedConv for inpainting the missing pixels at novel views. For better visual quality, we apply a median filter with window size 19 to the disparity image $\hat{d}^s$. We also set the threshold on the occlusion mask so that a pixel is visible if half of the adjacent pixels are visible in a window of 9.

6 Runtime analysis

In Table 1, we show the comparison of the training time, the inference time, and the rendering time of our model, conducted on a desktop equipped with NVIDIA GTX1080 and Xeon E5-2603 CPU cores.

7 Failure Cases

Our method sometimes may generate unnatural disparity maps, which result in implausible novel view rendering. Figure 2 (top) shows the incorrect generated disparity map and Figure 2 (middle) shows the corresponding generated color image. Our method may also fail due to the incorrect MPI generation (e.g.,

Table 1: Run-time analysis.

Training time of disparity generator (seconds / per iteration)	0.283
Training time of MPI generator (seconds / per iteration)	0.949
Inference time of image and disparity (seconds / per scene)	0.040
Inference time of MPI (seconds / per scene)	9.125
Rendering time of novel view (seconds / per view)	0.443

generate incorrect disoccluded contents). Figure 2 (bottm) shows that novel view reveals clear visual artifacts.

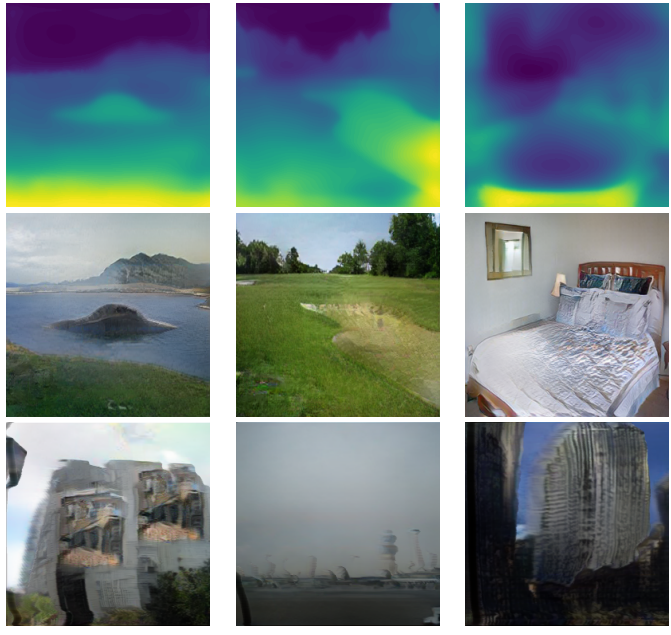

Fig. 2: **Failure Cases.** (a) Our method generates unnatural disparity map *(top, middle)*. (b) Our method generates incorrect disoccluded contents *(bottom)*.

References

1. Im, S., Jeon, H.G., Lin, S., Kweon, I.S.: DPSNet: End-to-end deep plane sweep stereo. In: ICLR (2019) 2
2. Lasinger, K., Ranftl, R., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. In: arXiv:1907.01341 (2019) 2
3. Nathan Silberman, Derek Hoiem, P.K., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV (2012) 1
4. Niklaus, S., Mai, L., Yang, J., Liu, F.: 3d ken burns effect from a single image (2019) 1, 4
5. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: CVPR (2019) 2, 3, 4
6. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: ICCV (2019) 1
7. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017) 3
8. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: CVPR (2017) 1, 2, 3
9. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. In: SIGGRAPH (2018) 1, 3, 4