# Supplemental Document for
# Temporally Consistent Online Depth Estimation Using Point-Based Fusion

## A1. Extended Results for CVD [1]

Further evaluation of our method and CVD on the MPI Sintel dataset is presented in Fig. A2 and Table A1. As CVD uses the Mannequin Challenge (MC) [1] network, we use it as the backbone for our method too for fair evaluation. CVD runs offline, fine-tuning the backbone network for ∼20 minutes at inference-time. Consequently, its performance is state-of-the-art and represents the high bar for temporal consistency. Our method achieves a good portion of the gains of CVD while being generalizable and online. Furthermore, the constraints for optimizing CVD at inference-time are obtained from COLMAP [4, 5] for static regions of the scene only. As a result CVD can sometimes filter moving objects as seen in the first row of Fig. A2.

## A2. Blender Dataset

Figure A1 shows samples from the custom Blender dataset we use for fine-tuning the temporal fusion network. The scene consists of organically moving and deforming humanoids in an indoor setting. We generate RGB images, ground truth depth, and camera poses for 50 stereo sequences of 60 frames each ($50 \times 2 \times 60 = 6000$ frames at 1280×720 pixels). The camera poses are manually initialized but follow random trajectories. We use both left and right cameras to generate samples and use data augmentation to increase the diversity of the training data.

## A3. Data Augmentation

We augment both the Blender dataset, and the FlyingThings3D [3] dataset described in Sec. 4 of the paper by applying random perturbations to the hue, saturation, brightness and contrast of the RGB images. We also randomly add camera shot noise to simulate real-world capture settings. Further, we generate training samples as random crops of 500×500 and 320×320 pixels respectively from the FlyingThings3D and Blender dataset. We encourage both networks to learn a depth scale invariant representation by randomly scaling the input depth maps by $s \in [0.2, 2]$.

## A4. Network Architectures

The detailed architectural specifications of the temporal and spatial fusion networks are provided in Tables A2 and A3 respectively. Both networks are based on a four-layer U-Net architecture, with the temporal fusion network having two additional sets of three residual blocks for preprocessing the color and depth branches.

Figure A1. Samples from the custom Blender dataset we rendered to train the temporal fusion network. It consists of 50 60-frame sequences with human-like motion in an indoor setting. We augment the dataset using the techniques described in Sec. A3

.

## References


[1] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T Freeman. Learning the depths of moving people by watching frozen people. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4521–4530, 2019. 1, 2

[2] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (ToG)*, 39(4):71–1, 2020. 2

[3] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134. 1

[4] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[5] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

| | MPI Sintel Dataset | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OPW↓ | SC↓ | RTC↑ | TCM↑ | TCC↑ | SD(L1)↓ | RAE↓ | RMS↓ | RMS(log)↓ | $\delta_1 \uparrow$ | $\delta_2 \uparrow$ | $\delta_3 \uparrow$ |
| MC [1] | 0.553 | 0.608 | 0.233 | 0.423 | 0.402 | 0.619 | 0.371 | 3.771 | 0.517 | 0.462 | 0.690 | 0.820 |
| CVD-MC [2] | 0.226 -59.1% | 0.245 -59.7% | 0.402 +72.5% | 0.515 +21.7% | 0.460 +14.4% | 0.589 -4.85% | 0.338 -8.89% | 3.078 -18.4% | 0.494 -4.66% | 0.493 +6.71% | 0.702 +1.74% | 0.819 -0.12% |
| Ours-MC | 0.377 -31.8% | 0.399 -34.4% | 0.360 +54.5% | 0.443 +4.73% | 0.423 +5.22% | 0.583 -5.82% | 0.361 -2.70% | 3.711 -1.59% | 0.506 -2.13% | 0.466 +0.87% | 0.696 +0.87% | 0.823 +0.37% |

Table A1. Comparing our approach to CVD using the Mannequin Challenge (MC) backbone for both methods. The percentage change in each metric over the baseline (MC) is listed in color. CVD fine-tunes the backbone for each scene (∼20 mins) and runs offline.

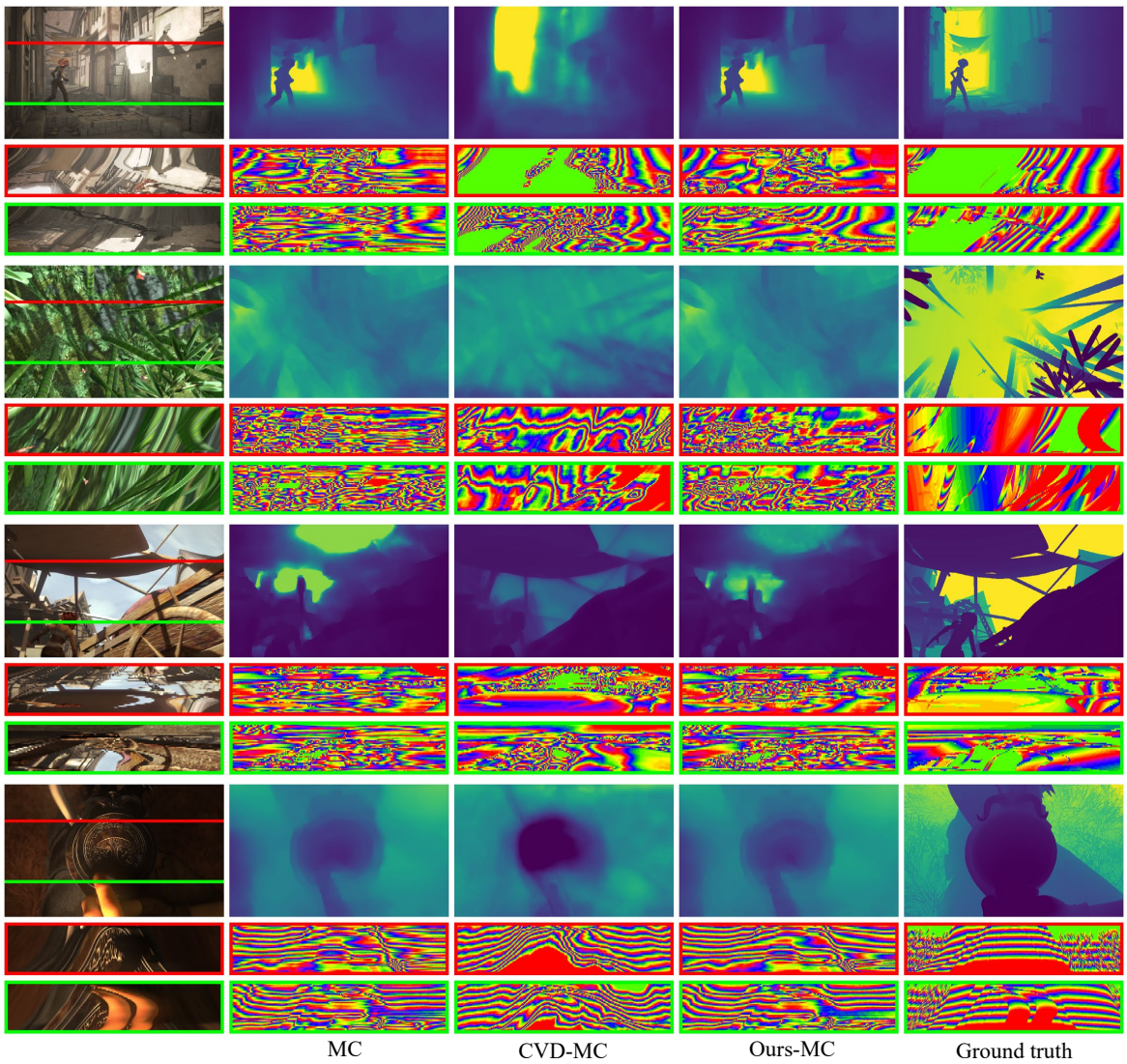


Figure A2. Qualitative comparison of our method with CVD [2] on the MPI Sintel dataset. We use an MC [1] backbone for both methods. The green and red boxes show an angular slice along the temporal dimension.

| Temporal Fusion Network | | | | | |
|---|---|---|---|---|---|
| Layer | **k** | **s** | **p** | Channels | Input |
| down1-1 | | | | 2/2 | $d^t \oplus d^t_{\mathrm{p}}$ |
| down1-2 | | | | 6/6 | $c^t \oplus c^t_{\mathrm{p}}$ |
| resblock2-1 | 5 | 1 | 2 | 2/8 | down1-1 |
| resblock2-2 | 3 | 1 | 1 | 8/16 | resblock2-1 |
| resblock2-3 | 3 | 1 | 1 | 16/24 | resblock2-2 |
| resblock2-4 | 5 | 1 | 2 | 6/8 | down1-2 |
| resblock2-5 | 3 | 1 | 1 | 8/16 | resblock2-4 |
| resblock2-6 | 3 | 1 | 1 | 16/24 | resblock2-5 |
| up3-1 | | | | 24/24 | resblock2-3 |
| up3-2 | | | | 24/24 | resblock2-6 |
| conv1-1 | 3 | 1 | 1 | 52/24 | up3-1 $\oplus$ up3-2 $\oplus$ $d^t \oplus c^t$ |
| conv1-2 | 3 | 1 | 1 | 24/24 | conv1-1 |
| maxpool1-1 | 3 | 1 | 1 | 24/24 | conv1-2 |
| conv2-1 | 3 | 1 | 1 | 24/48 | maxpool1-1 |
| conv2-2 | 3 | 1 | 1 | 48/48 | conv2-1 |
| maxpool2-1 | 3 | 1 | 1 | 48/48 | conv2-2 |
| conv3-1 | 3 | 1 | 1 | 48/96 | maxpool2-1 |
| conv3-2 | 3 | 1 | 1 | 96/96 | conv3-1 |
| maxpool3-1 | 3 | 1 | 1 | 96/96 | conv3-2 |
| conv4-1 | 3 | 1 | 1 | 96/192 | maxpool3-1 |
| conv4-2 | 3 | 1 | 1 | 192/192 | conv4-1 |
| maxpool4-1 | 3 | 1 | 1 | 192/192 | conv4-2 |
| conv5-1 | 3 | 1 | 1 | 192/384 | maxpool4-1 |
| conv5-2 | 3 | 1 | 1 | 384/384 | conv5-1 |
| up6-1 | | | | 384/384 | conv5-2 |
| conv6-2 | 3 | 1 | 1 | 576/192 | conv4-2 $\oplus$ up6-1 |
| conv6-3 | 3 | 1 | 1 | 192/192 | conv6-2 |
| up7-1 | | | | 192/192 | conv6-3 |
| conv7-2 | 3 | 1 | 1 | 288/96 | conv3-2 $\oplus$ up7-1 |
| conv7-3 | 3 | 1 | 1 | 96/96 | conv7-2 |
| up8-1 | | | | 96/96 | conv7-3 |
| conv8-2 | 3 | 1 | 1 | 144/48 | conv2-2 $\oplus$ up8-1 |
| conv8-3 | 3 | 1 | 1 | 48/48 | conv8-2 |
| up9-1 | | | | 48/48 | conv8-3 |
| conv9-2 | 3 | 1 | 1 | 72/24 | conv1-2 $\oplus$ up9-1 |
| conv9-3 | 3 | 1 | 1 | 24/1 | conv9-2 |

Table A2. The network architecture for the temporal fusion network $\Theta(\cdot)$ (Sec. 3.1, main paper); **k**, **s**, **p** denote the kernel size, stride and padding respectively, $\oplus$ is concatenation along the channel dimension, and "down"/ "up" is $\times 2$ bilinear down/up-sampling. All convolutional layers in the lower section except **conv9-3** are followed by a ReLU activation and instance norm; **conv9-3** has a sigmoid activation. All convolutions are 2D with a dilation of one.

| Spatial Fusion Network | | | | | |
|---|---|---|---|---|---|
| Layer | **k** | **s** | **p** | Channels | Input |
| conv1-1 | 3 | 1 | 1 | 4/24 | $c^t \oplus d^t$ OR $c^t \oplus d^t_{\mathrm{f}}$ |
| conv1-2 | 3 | 1 | 1 | 24/24 | conv1-1 |
| maxpool1-1 | 3 | 1 | 1 | 24/24 | conv1-2 |
| conv2-1 | 3 | 1 | 1 | 24/48 | maxpool1-1 |
| conv2-2 | 3 | 1 | 1 | 48/48 | conv2-1 |
| maxpool2-1 | 3 | 1 | 1 | 48/48 | conv2-2 |
| conv3-1 | 3 | 1 | 1 | 48/96 | maxpool2-1 |
| conv3-2 | 3 | 1 | 1 | 96/96 | conv3-1 |
| maxpool3-1 | 3 | 1 | 1 | 96/96 | conv3-2 |
| conv4-1 | 3 | 1 | 1 | 96/192 | maxpool3-1 |
| conv4-2 | 3 | 1 | 1 | 192/192 | conv4-1 |
| maxpool4-1 | 3 | 1 | 1 | 192/192 | conv4-2 |
| conv5-1 | 3 | 1 | 1 | 192/384 | maxpool4-1 |
| conv5-2 | 3 | 1 | 1 | 384/384 | conv5-1 |
| up6-1 | | | | 384/384 | conv5-2 |
| conv6-2 | 3 | 1 | 1 | 576/192 | conv4-2 $\oplus$ up6-1 |
| conv6-3 | 3 | 1 | 1 | 192/192 | conv6-2 |
| up7-1 | | | | 192/192 | conv6-3 |
| conv7-2 | 3 | 1 | 1 | 288/96 | conv3-2 $\oplus$ up7-1 |
| conv7-3 | 3 | 1 | 1 | 96/96 | conv7-2 |
| up8-1 | | | | 96/96 | conv7-3 |
| conv8-2 | 3 | 1 | 1 | 144/48 | conv2-2 $\oplus$ up8-1 |
| conv8-3 | 3 | 1 | 1 | 48/48 | conv8-2 |
| up9-1 | | | | 48/48 | conv8-3 |
| conv9-2 | 3 | 1 | 1 | 72/48 | conv1-2 $\oplus$ up9-1 |
| conv9-3 | 3 | 1 | 1 | 48/24 | conv9-2 |
| conv9-4 | 3 | 1 | 1 | 24/1 | conv9-3 |

Table A3. The network architecture for the spatial fusion network $\Phi(\cdot)$ (Sec. 3.2, main paper); **k**, **s**, **p** denote the kernel size, stride and padding respectively, $\oplus$ is concatenation along the channel dimension, and "up" is $\times 2$ bilinear up-sampling. All convolutional layers have a ReLU activation, with hidden layers using instance normalization; All convolutions are 2D with a dilation of one.