Neural RGB→D Sensing: Depth and Uncertainty from a Video Camera *Supplementary Document*

Chao Liu[1,2*] Jinwei Gu[1,3*] Kihwan Kim[1] Srinivasa G. Narasimhan[2] Jan Kautz[1]
[1]NVIDIA [2]Carnegie Mellon University [3]SenseTime

1. Relation of K-Net to the Kalman filter

The proposed update process defined in Eq. 8 in the main paper using residuals is closely related to Kalman Filter. In Kalman Filter, given the observation x_t at time t and the estimated hidden state h_{t-1} at time $t-1$, the updated hidden state h_t is:

$$h_t = W_t h_{t-1} + K_t(x_t - V_t W_t h_{t-1}) \quad (1)$$

where W_t is the transition matrix mapping the previous hidden state to current state; K_t is the gain matrix mapping the residual in the observation space to the hidden state space. V_t is the measurement matrix mapping the estimation in the hidden state space back to the observation space.

If we assume the measurement matrix is accurate: $x_t = Vh_t$, and the gain and measurement matrices are temporally invariant, we have:

$$\begin{aligned} h_t &= W_t h_{t-1} + K(Vh_t - VW_t h_{t-1}) \\ &= W_t h_{t-1} + KV(h_t - W_t h_{t-1}) \end{aligned} \quad (2)$$

Comparing our proposed update process in Eq. 5, Eq. 8 and Eq. 9 in the main paper and Kalman Filter in Eq.2, in our case the input images correspond to the observations x_t ; the negative-log depth probabilities correspond to the hidden states h_t; the warping operator warp$(\cdot)$ corresponds to the transition matrix W_t; the K-Net $g(\cdot)$ corresponds to the multiplication of the gain and measurement matrices KV in Eq. 2.

2. More Results

2.1. Complete metrics for Comparisons

We show the complete metrics for depth estimation comparisons in Table 1 and Table 2.

2.2. Results on KITTI without GPS or IMU

In Table 3, we show the performance of our method on the KITTI dataset, in case where only the IMU measurement are available (denoted as 'GT R'), and neither IMU nor GPU are available (denoted as 'opt. pose').

*The authors contributed on this work when they were at NVIDIA.

3. Network structures

In this section, we illustrate the network structures used in the pipeline.

3.1. D-Net

We show the structure of the D-Net in Table. 6. In the paper, we set $D = 64$.

3.2. K-Net

We show the structure of the K-Net in Table. 4. In the paper, we set $D = 64$.

3.3. R-Net

We show the structure of the R-Net in Table. 5. In the paper, we set $D = 64$.

Table 1: Comparison of depth estimation over the 7-Scenes dataset [6] with the metrics defined in [2]

	$\sigma < 1.25$	$\sigma < 1.25^2$	$\sigma < 1.25^3$	abs. rel	sq. rel	rmse	rmse log	scale. inv
DeMoN [7]	31.88	61.02	82.52	0.3888	0.4198	0.8549	0.4771	0.4473
MVSNet [8]	54.87	72.60	84.80	0.3481	0.4819	0.8305	0.4470	0.3743
DORN [3]	60.05	87.76	96.33	0.2000	0.1153	0.4591	0.2813	0.2207
Ours	**69.26**	**91.77**	**96.82**	**0.1758**	**0.1123**	**0.4408**	**0.2500**	**0.1899**

Table 2: Comparison of depth estimation over the KITTI dataset [4].

	$\sigma < 1.25$	$\sigma < 1.25^2$	$\sigma < 1.25^3$	abs. rel	sq. rel	rmse	rmse log	scale. inv
Eigen [2]	67.80	88.79	96.51	0.1904	1.263	5.114	0.2758	0.2628
Mono [5]	86.43	97.70	**99.47**	0.1238	0.5023	2.8684	0.1644	0.1635
DORN [3]	92.62	**98.18**	99.35	**0.0874**	**0.4134**	3.1375	0.1337	0.1233
Ours	**93.15**	98.018	99.25	0.0998	0.4732	**2.8294**	**0.1280**	**0.1070**

Table 3: Performance on KITTI dataset without GPS/IMU measurements

	$\sigma < 1.25$	$\sigma < 1.25^2$	$\sigma < 1.25^3$	abs. rel	sq. rel	rmse	rmse log	scale. inv
GT R	89.34	98.30	99.64	0.1178	0.4490	3.2042	0.1514	0.1509
opt. pose	87.78	97.22	99.10	0.1201	0.5763	3.5157	0.1672	0.1665

Table 4: K-Net structure. The operator expand(·) repeat the image intensity in the depth dimension

Name	Components	Input	Output dimension
Input	concat(cost_volume, expand(I_{ref}))		$\frac{1}{4}$H × $\frac{1}{4}$ W × D × 4
conv_0	conv_3d(3×3, ch_in=4, ch_out=32), ReLU conv_3d(3×3, ch_in=32, ch_out=32), ReLU	Input	$\frac{1}{4}$H × $\frac{1}{4}$ W × D × 32
conv_1	[conv_3d(3 ×3, ch_in=32, ch_out=32), ReLU conv_3d(3×3, ch_in=32, ch_out=32)] × 4	conv_0	$\frac{1}{4}$H × $\frac{1}{4}$ W × D × 32
conv_2	conv_3d(3×3, ch_in=32, ch_out=32), ReLU conv_3d(3×3, ch_in=32, ch_out=1)	conv_1	$\frac{1}{4}$H × $\frac{1}{4}$ W × D × 1
Output	Modified cost_volume from the conv_2 layer		$\frac{1}{4}$H × $\frac{1}{4}$ W × D × 1

Table 5: R-Net structure

Name	Components	Input	Output dimension
Input	cost_volume from K-Net		$\frac{1}{4}$H × $\frac{1}{4}$ W × D
conv_0	conv_2d(3×3, ch_in=64+D, ch_out= 64+D), LeakyReLU conv_2d(3×3, ch_in=64+D, ch_out= 64+D), LeakyReLU	concat(Input, fusion in D-Net)	$\frac{1}{4}$H × $\frac{1}{4}$ W × (64+D)
trans_conv_0	transpose_conv(4×4, ch_in=64+D, ch_out=D, stride=2), LeakyReLU	conv_0	$\frac{1}{2}$H × $\frac{1}{2}$ W × D
conv_1	conv_2d(3×3, ch_in=32+D, ch_out=32 + D), LeakyReLU conv_2d(3×3, ch_in=32+D, ch_out=32 + D),LeakyReLU	concat(trans_conv_0, conv_1 in D-Net	$\frac{1}{2}$H × $\frac{1}{2}$ W × (D+32)
trans_conv_1	transpose_conv(4×4, ch_in=32+D, ch_out=D, stride=2), LeakyReLU	conv_1	H × W × D
conv_2	conv_2d(3×3, ch_in=3+D, ch_out=3+D), LeakyReLU conv_2d(3×3, ch_in=3+D, ch_out=D), LeakyReLU conv_2d(3×3, ch_in= D, ch_out=D)	concat(trans_conv_1, I_{ref})	H × W × D
Output	Upsampled and refined cost_volume		H × W × D

Table 6: D-Net structure. The structure is taken from [1]

Name	Components	Input	Output dimension
Input	Input frame		H × W × 3
	CNN Layers		
conv0_1	conv_2d(3×3, ch_in=3, ch_out=32, stride=2), ReLU	Input	$\frac{1}{2}$H × $\frac{1}{2}$ W × 32
conv0_2	conv_2d(3×3, ch_in=32, ch_out=32), ReLU	conv0_1	$\frac{1}{2}$H × $\frac{1}{2}$ W × 32
conv0_3	conv_2d(3×3, ch_in=32, ch_out=32), ReLU	conv0_2	$\frac{1}{2}$H × $\frac{1}{2}$ W × 32
conv1	[conv_2d(3×3, ch_in=32, ch_out=32), ReLU conv_2d(3×3, ch_in=32, ch_out=32)] × 3	conv0_2	$\frac{1}{2}$H × $\frac{1}{2}$ W ×32
conv1_1	conv_2d(3×3, ch_in=32, ch_out=64, stride=2), ReLU	conv1	$\frac{1}{4}$H × $\frac{1}{4}$ W ×64
conv2	[conv_2d(3×3, ch_in=64, ch_out=64), ReLU conv_2d(3×3, ch_in=64, ch_out=64)] × 15	conv1_1	$\frac{1}{4}$H × $\frac{1}{4}$ W ×64
conv2_1	conv_2d(3×3, ch_in=64, ch_out=128), ReLU	conv2	$\frac{1}{4}$H × $\frac{1}{4}$ W ×128
conv3	[conv_2d(3×3, ch_in=128, ch_out=128), ReLU conv_2d(3×3, ch_in=128, ch_out=128)] × 2	conv2_1	$\frac{1}{4}$H × $\frac{1}{4}$ W × 128
conv4	[conv_2d(3×3, ch_in=128, ch_out=128, dila=2), ReLU conv_2d(3×3, ch_in=128, ch_out=128, dila=2)] × 3	conv3	$\frac{1}{4}$H × $\frac{1}{4}$ W × 128
	Spatial Pyramid Layers		
branch1	avg_pool(64×64,stride=64) conv_2d(1×1, ch_in=128, ch_out=32), ReLU bilinear interpolation	conv4	$\frac{1}{4}$H × $\frac{1}{4}$ W × 32
branch2	avg_pool(32 × 32,stride= 32) conv_2d(1×1, ch_in=128, ch_out=32), ReLU bilinear interpolation	conv4	$\frac{1}{4}$H × $\frac{1}{4}$ W × 32
branch3	avg_pool(16 × 16,stride= 16) conv_2d(1×1, ch_in=128, ch_out=32), ReLU bilinear interpolation	conv4	$\frac{1}{4}$H × $\frac{1}{4}$ W × 32
branch4	avg_pool(8 × 8,stride= 8) conv_2d(1×1, ch_in=128, ch_out=32), ReLU bilinear interpolation	conv4	$\frac{1}{4}$H × $\frac{1}{4}$ W × 32
concat	concat(branch1, branch2, branch3, branch4, conv2, conv4)		$\frac{1}{4}$H × $\frac{1}{4}$ W × 320
fusion	conv_2d(3×3, ch_in=320, ch_out=128), ReLU conv_2d(1×1, ch_in=128, ch_out=64), ReLU	concat	$\frac{1}{4}$H × $\frac{1}{4}$ W × 64
Output	The extracted image feature from the fusion layer		$\frac{1}{4}$H × $\frac{1}{4}$ W × 64

References

[1] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018. 3

[2] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 2

[3] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[4] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 2

[5] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[6] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[7] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. DeMoN: Depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[8] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2018. 2