Supplementary Material for "A Confidence-based Iterative Solver of Depths and Surface Normals for Deep Multi-view Stereo"

Wang Zhao[1,3*] Shaohui Liu[2,3*] Yi Wei[1] Hengkai Guo[3] Yong-Jin Liu[1,4]
[1]Tsinghua University [2] ETH Zurich [3] ByteDance Inc. [4] JCMV
b1ueber2y@gmail.com, {zhao-w19, y-wei19}@mails.tsinghua.edu.cn,
guohengkai@bytedance.com, liuyongjin@tsinghua.edu.cn

In this document, we provide a list of supplementary materials that accompany the main paper.

A. Detailed Derivations of the Proposed Solver

A.1. Preliminaries

As discussed in the main paper, we solve the depth map and normal map with two separate suboptimization steps with respect to the total energy. Each step contains a plane-based propagation with slanted planes. Recall that $P(x, d, n)$ denotes the slanted plane at pixel coordinate x generating by spanning a plane from the corresponding 3D points recovered from d and x and its surface normal n. In practice we parameterize the normal to be $n = (a, b, -1)$, which enables closed-form computation in the normal update step (N-step). Let (p, q, z) denotes the 3D coordinate of the points recovered from d and $x = (u, v)^T$ at the frame coordinate system:

$$\begin{bmatrix} p \\ q \\ z \end{bmatrix} = K^{-1} \begin{bmatrix} x \\ 1 \end{bmatrix} d = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ 1 \end{bmatrix} d, \tag{1}$$

where K is the camera intrinsic parameter, $x = (u, v)^T$ is the 2D pixel coordinate and $(\tilde{u}, \tilde{v}, 1)^\top = K^{-1}(u, v, 1)^\top$ is the normalized homogeneous coordinate. Then, the plane equation of $P(x_i, d_i, n_i)$ with $n_i = (a_i, b_i, -1)$ and the recovered 3D points (p_i, q_i, z_i) can be written as follows:

$$a_i(p - p_i) + b_i(q - q_i) - (z - z_i) = 0. \tag{2}$$

At plane-based propagation, the propagated depth $d_{i\to j}$ ($d_{j\to i}$) is computed by projecting the slanted plane at i (j) onto the pixel j (i). We give the derivation of $d_{i\to j}$ here:

$$a_i(p_j - p_i) + b_i(q_j - q_i) - (z_j - z_i) = 0 \tag{3}$$

$$\Leftrightarrow a_i(\tilde{u}_j d_{i\to j} - p_i) + b_i(\tilde{v}_j d_{i\to j} - q_i) - (d_{i\to j} - z_i) = 0 \tag{4}$$

$$\Leftrightarrow d_{i\to j} = \frac{a_i p_i + b_i q_i - z_i}{a_i \tilde{u}_j + b_i \tilde{v}_j - 1} = \frac{a_i \tilde{u}_i + b_i \tilde{v}_i - 1}{a_i \tilde{u}_j + b_i \tilde{v}_j - 1} d_i \tag{5}$$

Here $d_{i\to j}$ is non-linearly dependent upon the depth d_i and the surface normal $n_i = (a_i, b_i, -1)$. As there exist second-order terms $a_i d_i$ and $b_i d_i$ in the denominator, a quadratic energy over both d_i and n_i is infeasible even when the algebraic formulation is employed. Thus, closed-form solution cannot be acquired when the data term and plane-based structural term $E_{i\to j}$[1] are jointly optimized over d_i and n_i. This motivates us to employ iterative suboptimization in the solver to acquire close-form solution, which can further benefit our deep MVS system with end-to-end joint training, as discussed in the main paper.

Before introducing the details of the two update steps of the proposed solver, let us take a step further on the formulation of jointly solving depths and surface normals. We want to note that it is possible to formulate closed-form solution by substitution of variables when only the surface normal data term is employed. This can be achieved by parameterizing the plane equation in Eq. (2) as $a_i p + b_i q - (z - t_i) = 0$, where $t_i = z_i - a_i p_i - b_i q_i = (1 - a_i \tilde{u}_i - b_i \tilde{v}_i) d_i$. When the depth data term is not included, by employing the algebraic form we can get a 3x3 linear system with respect to a_i, b_i and t_i. However, we empirically observe that the depth data term is extremely beneficial in practice.

A.2. Closed-form Solution

As discussed in the main paper, we employ suboptimization over the depth map and the surface normal map iteratively. This enables closed-form solution in both steps.

[1] $E_{j\to i}$ does not include the surface normal n_i in its formulation, and thus can only be used when only depth map is required to be solved.

Depth Update (D-step). At the depth update step (D-step), we fix the surface normal map and solve for the optimal depth map d^*. L2 distance between the optimized depth and the propagated depth $d_{j\to i}$ from neighboring pixels are used in the plane-based structural term. The objective is written as follows (Eqs. (4)(5) in the main paper):

$$\min_d E_{total} = \min_d E_d \tag{6}$$

$$E_d = \alpha \sum_i c_i (d_i - \hat{d}_i)^2 + \sum_i \sum_{j \in N(i)} c_j w_{ij} (d_i - d_{j\to i})^2. \tag{7}$$

As discussed in the main paper, we assume fixed neighborhoods to enable parallelization of the solver. Thus, the propagated depth $d_{j\to i}$ is the projection of the plane $P(x_j, \hat{d}_j, \hat{n}_j)$ at pixel i:

$$d_{j\to i} = \frac{\hat{a}_j \tilde{u}_j + \hat{b}_j \tilde{v}_j - 1}{\hat{a}_j \tilde{u}_i + \hat{b}_j \tilde{v}_i - 1} \hat{d}_j \tag{8}$$

Set the first-order derivative to zero we can easily derive the optimal depth d_i^* for each pixel:

$$d_i^* = \frac{\alpha c_i \hat{d}_i + \sum_{j \in N(i)} c_i w_{ij} d_{j\to i}}{\alpha c_i + \sum_{j \in N(i)} c_i w_{ij}}. \tag{9}$$

Surface Normal Update (N-step). At the surface normal update step, we fix the depth map and solve for the optimal surface normal n^*. The objective is written as follows (Eqs. (6)(7) in the main paper):

$$\min_n E_{total} = \min_n E_n \tag{10}$$

$$\begin{aligned} E_n = {} & \alpha \sum_i c_i ||n_i - \hat{n}_i||^2 \\ & + \sum_i \sum_{j \in N(i)} c_j w_{ij} D_n(d_j, P(x_i, d_i, n_i)). \end{aligned} \tag{11}$$

Here D_n is a distance function defined over d_j and the plane $P(x_i, d_i, n_i)$ being optimized. Note that because $d_{i\to j}$ is non-linearly dependent over n_i as shown in Eq. (5) in this supplementary material, we cannot directly use L2 distance as in the D-step. Instead, we employ the algebraic formulation of the plane equation and directly formulate the distance function D_n as the square of the LHS of Eq. (3) in this supplementary material:

$$D_n(d_j, P(x_i, d_i, n_i)) = [a_i(p_j - p_i) + b_i(q_j - q_i) - (z_j - z_i))]^2. \tag{12}$$

Because the depth map is fixed in the N-step, the only two unknown variables are a_i and b_i, which represent the surface normal $n_i = (a_i, b_i, -1)$. Note that this parameterization is feasible because all visible surfaces are facing the position where the camera center locates. However, numerical problems may occur when there exist ill-posed cases with surfaces that are nearly parallel to the corresponding camera rays. Thus, we clip the absolute value of the solved a_i and b_i with a threshold 20.0. This operation is empirically crucial to stabilize the end-to-end training process.

By setting the first-order derivatives to zero we can get a 2x2 linear system over a_i^* and b_i^*, where the optimal surface normal n_i^* is parameterized with $n_i^* = (a_i^*, b_i^*, -1)$. The coefficients are listed as follows:

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} a_i^* \\ b_i^* \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{13}$$

$$A_{11} = \alpha c_i + \sum_{j \in N(i)} c_j w_{ij} (p_j - p_i)^2 \tag{14}$$

$$A_{22} = \alpha c_i + \sum_{j \in N(i)} c_j w_{ij} (q_j - q_i)^2 \tag{15}$$

$$A_{12} = A_{21} = \sum_{j \in N(i)} c_j w_{ij} (p_j - p_i)(q_j - q_i) \tag{16}$$

$$B_1 = \alpha c_i \hat{a}_i + \sum_{j \in N(i)} c_j w_{ij} (p_j - p_i)(z_j - z_i) \tag{17}$$

$$B_2 = \alpha c_i \hat{b}_i + \sum_{j \in N(i)} c_j w_{ij} (q_j - q_i)(z_j - z_i) \tag{18}$$

In the final step, normalization is applied on the output surface normal to get the final prediction with standard parameterization:

$$n = \left(\frac{a}{\sqrt{a^2 + b^2 + 1}}, \frac{b}{\sqrt{a^2 + b^2 + 1}}, -\frac{1}{\sqrt{a^2 + b^2 + 1}}\right). \tag{19}$$

A.3. Extension: Incorporating Customized Depth Confidence and Normal Confidence

While we present all the derivations with a single confidence map c_i for simplicity, a separate depth confidence map c_d and surface normal confidence map c_n can be used in practice to further improve the solver. Employing those separate customized confidence maps will change the data term (Eq. (2) in the main paper) into the formulation below:

$$E_{data} = \sum_i c_{d,i} (d_i - \hat{d}_i)^2 + \sum_i c_{n,i} ||n_i - \hat{n}_i||^2, \tag{20}$$

where $c_{d,i}$ and $c_{n,i}$ denotes the per-pixel depth and surface normal confidence respectively. Furthermore, the confidence used in the plane-based structural term can also be monitored by the separate confidence maps.

Separate Confidence Maps used in the D-step. When the depth confidence and normal confidence are used jointly, the final formulation of E_d becomes the form as follows:

$$E_d = \alpha \sum_i c_{d,i}(d_i - \hat{d}_i)^2 + \sum_i \sum_{j \in N(i)} c_{d,j} c_{n,j} w_{ij} (d_i - d_{j \to i})^2. \quad (21)$$

Note that both the depth confidence $c_{d,j}$ and normal confidence $c_{n,j}$ are used in the plane-based structural term. This is due to the fact that the quality of the propagated depth $d_{j \to i}$ depends on both $\hat{d}_j$ and $\hat{n}_j$.

Separate Confidence Maps used in the N-step. As for the surface normal update step, because $E_{i \to j}$ is employed, $\hat{n}_j$ is non-relevant to the computation so the plane-based structural term will not be affected by $c_{n,j}$. However, since the depth map is fixed here and the computation of $d_{i \to j}$ depends on the quality of $\hat{d}_i$, we multiply the plane-based structural term with the depth confidence of the studied pixel $c_{d,i}$. The final formulation of E_n is written as follows:

$$E_n = \alpha \sum_i c_{n,i} ||n_i - \hat{n}_i||^2 + \sum_i c_{d,i} \sum_{j \in N(i)} c_{d,j} w_{ij} D_n(d_j, P(x_i, d_i, n_i)). \quad (22)$$

Closed-form solution can be easily derived for the modified objectives in Eq. (21) and Eq. (22) following the previous discussion.

In our implementation, the depth confidence map and normal confidence map are jointly predicted by the cost-volume based neural networks. The supervision is acquired by computing the relative depth error and normal angle error, as described in the main paper. At inference, the hybrid confidence map combining the deep depth confidence and the geometric confidence is used as the depth confidence map, while the normal confidence only employs the deep normal confidence prediction.

B. Network Architectures

We follow prior works [2, 3] to design the initial depth, surface normal, and confidence estimation networks.

For the depth and normal branches, we use the same network architectures as [3] without the consistency module. Specifically, the target and reference images are first encoded to get the feature maps. Then we use plane sweeping with 64 hypothesis planes to build the feature cost volume. The 3D CNNs and 2D context CNNs are applied on the cost volume to aggregate and regularize the cost information. The soft argmin is used to regress the final depth values from the final cost volume. The cost volume information is also utilized for the multi-view surface normal estimation. The intermediate cost volume features are concatenated with the world coordinates of every voxel, and then transformed by several 3D CNNs to get 8 cost volume slices. Each slice is processed by 7 shared layers of 2D convolutions of dilated 3×3 kernels. The output of all slices are summed and normalized to get the final surface normal. We recommend the reader to refer to [3] for more details.

For the confidence branch, we input multiple sources to the network to better estimate the confidence. To predict the depth confidence, we utilize target image features, homography-warped reference image features using currently predicted depth, cost volume features before softmax, and the predicted depth. As illustrated in Figure 1, these inputs are processed by three mini-branches, and each mini-branch contains two or three layers of 3×3 convolutions. To be specific, the first branch (target image feature + predicted depth) consists of two 3×3 convolution layers whose output channels are both 16. The second branch (target image feature + warped reference image feature) has also two 3×3 convolution layers whose output channels are both 32. The third branch (cost volume feature) consists of three 3×3 convolution layers with [64, 32, 1] as each layer's output channel number. The outputs from three mini-branches are then concatenated, and five dilated 3×3 convolution layers with output channels [64, 64, 64, 32, 1] and dilations [1, 2, 4, 1, 1], followed by the final sigmoid activation, are applied to jointly predict the depth confidence. For surface normal confidence, we use target image features, intermediate features from the previously discussed 8 cost volume slices, and the predicted surface normal, as shown in Figure 2. These inputs are processed by two mini-branches, then concatenated and used to jointly predict the surface normal confidence. The first branch consists of two 3×3 convolution layers with both 16 output channels, and the second branch has also two 3×3 convolution layers with both 32 output channels. Then the outputs from these two mini-branches are concatenated and processed by five dilated 3×3 convolution layers with output channels [64, 64, 64, 32, 1] and dilations [1, 2, 4, 1, 1], plus final sigmoid activation, to get the confidence map for surface normal prediction.

C. System Training

Our proposed deep MVS system is trained with two stages. In the first stage, we train the initial depth, normal, and confidence estimation network for 15 epochs. The training loss for the initial depth, normal and confidence estimation, denoted as L_{net}, includes three parts: $L_{net} = w_d L_d + w_n L_n + w_c L_c$, where w_d, w_n, w_c are three hyper-parameters to balance different loss terms. L_d and L_n are formulated as smoothed L1 loss, and L_c employs

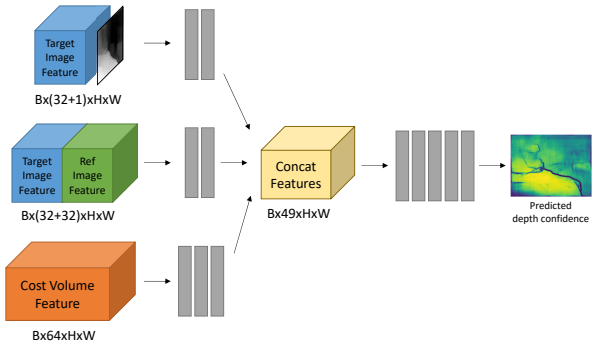


Figure 1: The inputs are processed with three mini-branches (with 2 or 3 layers of 3x3 convolutions), and then jointly fed into 5 dilated convolutions followed by final sigmoid activation to regress the depth confidence map.

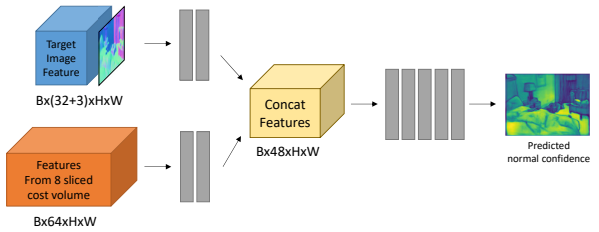


Figure 2: The inputs are processed with two mini-branches (with 2 layers of 3x3 convolutions), and then jointly fed into 5 dilated convolutions followed by final sigmoid activation to regress the normal confidence map.

cross-entropy loss. L_c consists of depth confidence loss L_{cd} and surface normal confidence loss L_{cn}, and is computed as $L_c = L_{cd} + L_{cn}$.

In the second stage, we finetune the network jointly with the proposed iterative depth normal solver for 10 epochs. During the training, We iteratively refine the depth map and surface normal map for 5 times to balance the computation and effectiveness. We apply the depth and surface normal loss on both the initial predictions and the final solved geometry. Thus the total loss for end-to-end training is $L_{total} = \lambda L_{net} + L_{solver}$, where $L_{solver} = w_{d'} L_{d'} + w_{n'} L_{n'}$ is the depth and normal loss for the solved geometry. $L_{d'}$ and $L_{n'}$ are smoothed L1 loss for the refined depth and normal, respectively. $w_{d'}$ and $w_{n'}$ are hyper-parameters. λ is the hyper-parameter for weighting these two losses L_{net} and L_{solver}. We choose to include the L_{net} in the end-to-end training to regularize the initial depth and normal predictions, which in practice stabilizes the joint training.

D. Implementation Details

The hyperparameters are heuristically selected without much tuning. We set loss weights $[\lambda, w_d, w_n, w_c, w_{d'}, w_{n'}]$ to [0.7, 1.0, 3.0, 0.2, 1.0, 3.0]. The scaling factors γ_1, γ_2 for depth and normal confidence groundtruth are both set to 5.0 in the training. The spatial and color weights for bilateral affinity σ_x^2, σ_c^2 are set to 2.5 and 25.0, and the weight α for depth and normal data term of the energy is set to 1.0. These hyper-parameters are fixed during both training and inference.

E. Notations

We provide a notation cheat sheet in Table 1, which describes the relevant notations used in the main paper and this supplementary material.

F. Additional Visualization

We provide additional visualizations for both depth and surface normal estimation in Figure 3 and 4. All samples are from the official test split of ScanNet [1].

References

[1] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 4, 6, 7

[2] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. In *ICLR*, 2019. 3, 6

[3] Uday Kusupati, Shuo Cheng, Rui Chen, and Hao Su. Normal assisted stereo depth estimation. In *CVPR*, pages 2189–2199, 2020. 3, 6, 7

[4] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb (r) d sensing: Depth and uncertainty from a video camera. In *CVPR*, pages 10986–10995, 2019. 6

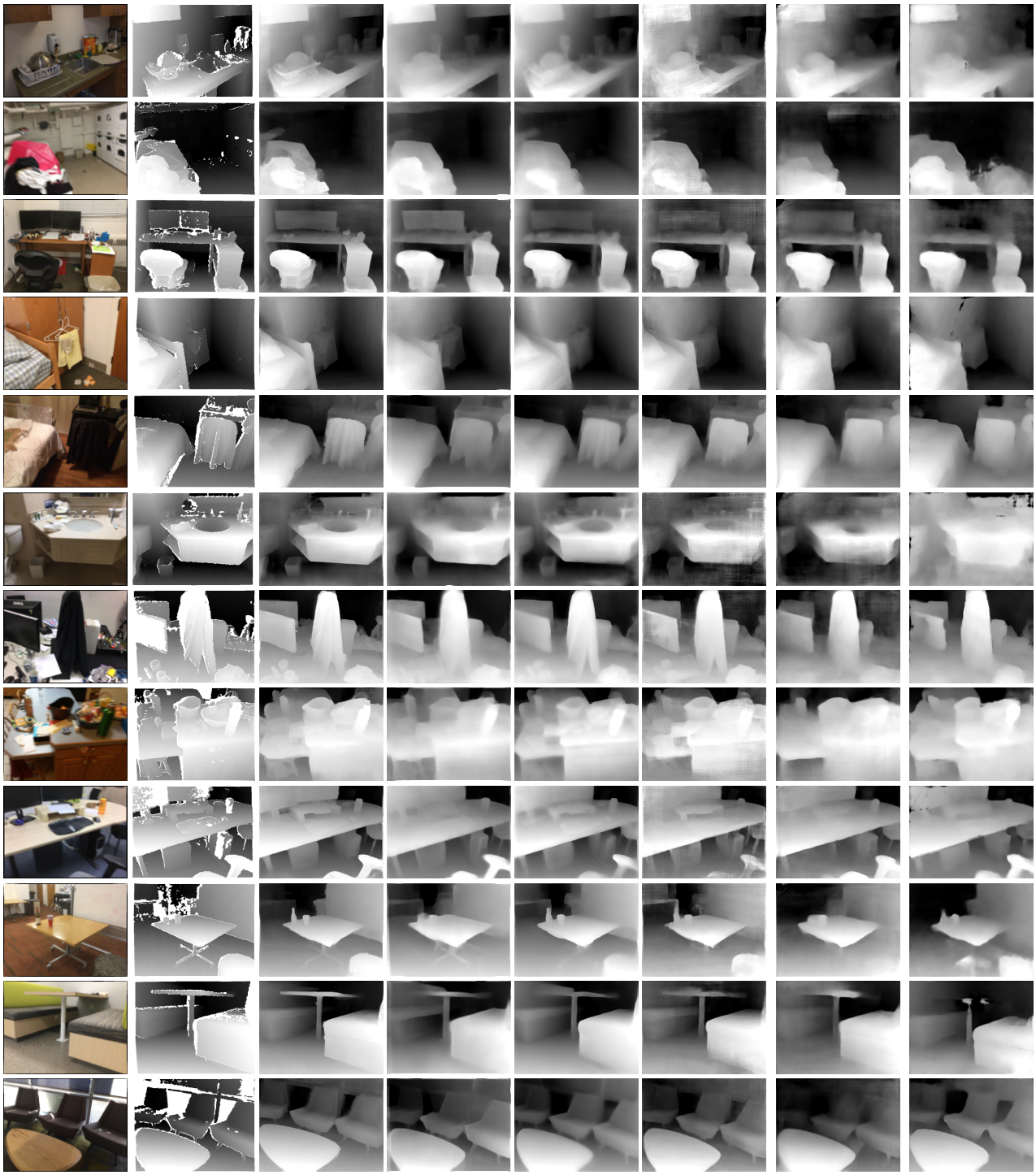
[5] Xiaoxiao Long, Lingjie Liu, Christian Theobalt, and Wenping Wang. Occlusion-aware depth estimation with adaptive normal constraints. In *ECCV*, pages 640–657. Springer, 2020. 7

[6] Ayan Sinha, Zak Murez, James Bartolozzi, Vijay Badrinarayanan, and Andrew Rabinovich. Deltas: Depth estimation by learning triangulation and densification of sparse points. In *ECCV*, 2020. 6

[7] Kaixuan Wang and Shaojie Shen. Mvdepthnet: Real-time multiview depth estimation neural network. In *3DV*, pages 248–257. IEEE, 2018. 6

Notations	Descriptions	Appearances
E_{total}	The total energy potential for the solver	Sec. 3.1, Sec. 3.2, Sec. A.2
E_{data}	The data term in the total energy potential	Sec. 3.1, Sec. A.3
E_{plane}	The plane-based structural term in the total energy potential	Sec. 3.1
α	The hyperparameter to balance the data term and structural term	Sec. 3.1, Sec. 3.2, Sec. A.2, Sec. A.3, Sec. E
x, d, n, c	2D coordinate, depth, surface normal, and confidence	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.2, Sec. A.3
x_i, d_i, n_i, c_i	Per-pixel 2D coordinate, depth, surface normal, and confidence	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.2
$\hat{d}_i, \hat{n}_i$	Initial per-pixel depth and surface normal	Sec. 3.1, Sec. 3.2, Sec. A.2, Sec. A.3
$P(x, d, n)$	The plane generated by current 2D coordinate x, depth d, and normal n	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.2, Sec. A.3
$E_{j\to i}, E_{i\to j}$	The plane-based structural term defined in two directions	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.3
$d_{j\to i}$	The projection of the plane $P(x_j, d_j, n_j)$ at pixel i	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.2, Sec. A.3
$d_{i\to j}$	The projection of the plane $P(x_i, d_i, n_i)$ at pixel j	Sec. 3.1, Sec. 3.2, Sec. A.1, Sec. A.2, Sec. A.3
w_{ij}	Edge-aware bilateral affinity between pixel i and j	Sec. 3.1, Sec. 3.2, Sec. A.2, Sec. A.3
I_i	RGB value at pixel i	Sec. 3.1
σ_x, σ_c	The hyperparameters for the spatial term and color term in the bilateral affinity	Sec. 3.1, Sec. E
E_d	The minimized objective in the D-step	Sec. 3.2, Sec. A.2, Sec. A.3
$N(i)$	The defined neighborhoods of pixel i	Sec. 3.2, Sec. A.2, Sec. A.3
E_n	The minimized objective in the N-step	Sec. 3.2, Sec. A.2, Sec. A.3
D_n	The distance function between the depth d and slanted plane P used in the N-step	Sec. 3.2, Sec. A.2, Sec. A.3
a, b	Components of the parameterized surface normal, $n = (a, b, -1)$	Sec. 3.2, Sec. A.1, Sec. A.2
c_{dgt}, c_{ngt}	Groundtruth (GT) confidence maps for depth and surface normal	Sec. 4.1
e_{rel}, e_{ang}	Relative depth error and normal angle error between predictions and groundtruths	Sec. 4.1
γ_1, γ_2	Hyperparameters used in the computation of the GT depth and normal confidence	Sec. 4.1, Sec. E
p, q, z	The 3D coordinate of the unprojected point	Sec. A.1, Sec. A.2
K	Camera intrinsic matrix	Sec. A.1
u, v	2D pixel coordinates, $x = (u, v)^T$	Sec. A.1
$\tilde{u}, \tilde{v}$	$(\tilde{u}, \tilde{v}, 1)^T = K^{-1}(u, v, 1)^T$	Sec. A.1
d^*	The optimal depth map in the D-step	Sec. A.2
n^*	The optimal surface normal map in the N-step	Sec. A.2
a^*, b^*	Components of the optimal surface normal n^*, $n^* = (a^*, b^*, -1)$	Sec. A.2
$A_{11}, A_{12}, A_{21}, A_{22}$, B_1, B_2	Coefficients used in the N-step computation	Sec. A.2
c_d, c_n	The depth confidence map and surface normal confidence map	Sec. A.3
$c_{d,i}$, $c_{n,i}$	Per-pixel depth confidence and surface normal confidence	Sec. A.3
L_{net}	Training loss for the initial depth, surface normal, and confidence network	Sec. D
L_d, L_n, L_c	Losses of initial depth, surface normal and confidence	Sec. D
w_d, w_n, w_c	Loss weights of initial depth, surface normal and confidence	Sec. D, Sec. E
L_{cd}, L_{cn}	Confidence loss for depth and normal, $L_c = L_{cd} + L_{cn}$	Sec. D
L_{total}	Total training loss in the end-to-end training	Sec. D
L_{solver}	The Loss defined over the solved geometry $L_{solver} = w_{d'}L_{d'} + w_{n'}L_{n'}$	Sec. D
$L_{d'}, L_{n'}$	Losses of the solved depth and surface normal	Sec. D
$w_{d'}, w_{n'}$	Loss weights of the solved depth and surface normal	Sec. D, Sec. E
λ	The hyperparameter used to balance L_{net} and L_{solver}	Sec. D, Sec. E

Table 1: Notations used in the main paper and supplementary material.



(a) Image (b) Groundtruth (c) Ours (d) DELTAS [6] (e) NAS [3] (f) DPSNet (FT) [2] (g) MVDepth (FT) [7] (h) N-RGBD [4]

Figure 3: More qualititative results of depth estimation on ScanNet [1]. **Better viewed when zoomed in.**

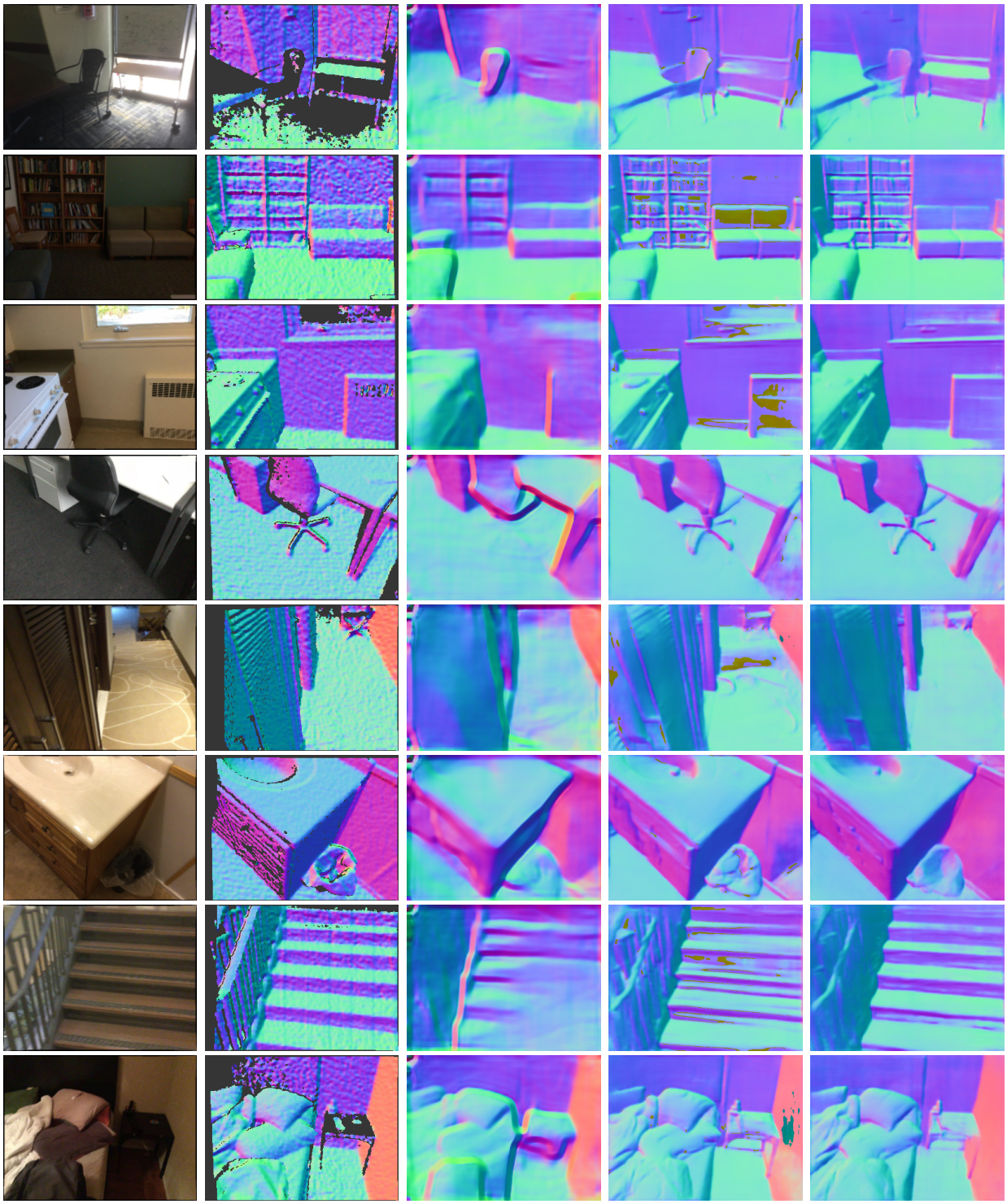


Figure 4: More qualitative results of surface normal estimation on ScanNet [1]. **Better viewed when zoomed in.**