Supplementary Material
DeepMapping: Unsupervised Map Estimation From Multiple Point Clouds

Li Ding* †
l.ding@rochester.edu

Chen Feng* ‡ §
cfeng@nyu.edu

†University of Rochester ‡NYU Tandon School of Engineering §Mitsubishi Electric Research Laboratories (MERL)

1. Ablation Studies

In this section, we conduct several ablation studies to investigate the effects of various network architectures in DeepMapping using the AVD [2]. For quantitative comparison, we choose absolute trajectory error (ATE) as metrics and include the ATE from baseline multiway registration method [3].

Feature extraction module in the L-Net: we compare the effects of feature extraction module, i.e., CNN-based architecture and PointNet-based architecture [6]. The CNN-based network consists of C(64)-C(128)-C(256)-C(1024)-AM(1), where C(n) denotes 2D atrous convolutions that have kernel size 3, dilation rate of 2 and n-channel outputs, AM(1) denotes 2D adaptive max-pooling layer.. The PointNet-based architecture is FC(64)-FC(128)-FC(256)-FC(1024)-AM(1) , where FC(n) denotes fully-connected layer with n-channel output.

The box plot in Figure 1 depicts the quantitative results of the ATE. As shown, CNN-based architecture achieves better performance with a median error of 134.07mm than PointNet-based architecture that has a median error of 207.84mm. This is not supervising because CNN is able to explore local structure information from neighborhood pixels while PointNet is a per-point function performing on each point independently.

Architecture of the M-Net: the proposed DeepMapping uses MLP in the M-Net to predict the occupancy status in the global coordinates. We compare this architecture with ResMLP that integrate the idea of deep residual networks [5]. ResMLP consists of a stack of basic residual blocks where each residual block, denoted as RB(n), contains two fully-connected layers with the same number of output nodes n. The detailed ResMLP architecture can be described as RB(64)-RB(64)-RB(64)-RB(128)-RB(128). As shown in Figure 2, MLP has a marginal improvement over ResMLP in terms of the ATE and therefore is adopted in the proposed DeepMapping.

*This work was partially done while the authors were with MERL. And Chen Feng is the corresponding author.

Depth and width of MLP in the M-Net: the depth and width of MLP are defined as the number of layers in the MLP and the number of output nodes from each layer. To investigate the influence of layer depth, we fixed the layer width to 64 and test MLP with depths 4, 5, 6, and 7. Figure 3 show the corresponding results of the ATE. As shown, increasing MLP depth is beneficial to reducing the ATE. For example, MLP with depth 6 has a lower error than those with depth 4 and 5. However, deeper networks may deteriorate the performance and make it difficult to optimize. To compare the effect of MLP width, we fixed the depth to 4 and choose MLP with width 32, 64, 96, and 128. As shown in Figure 4. MLP with a width of 128 achieves the best performance.

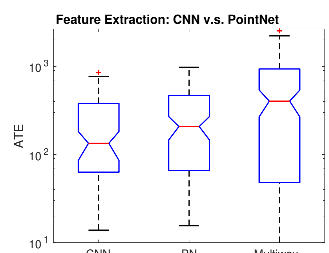

Figure 1. Quantitative comparison of the ATE between CNN-based and PointNet-based architectures in the L-Net, tested on the AVD [2].

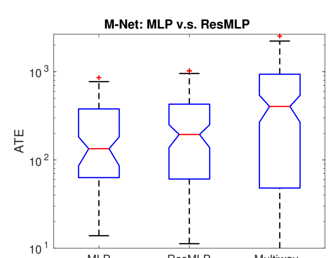

Figure 2. Quantitative comparison of the ATE between MLP and ResMLP in the M-Net, tested on the AVD [2].

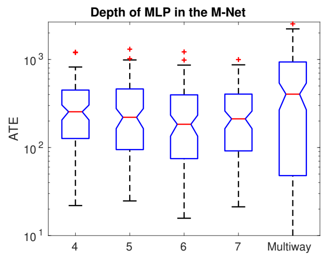

Figure 3. Quantitative comparison of different depths of MLP in the M-Net, tested on the AVD [2]. The layer width is fixed to 64.

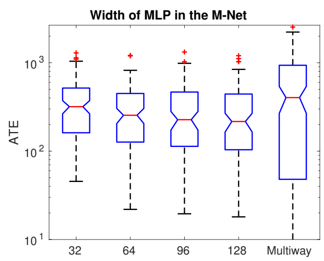

Figure 4. Quantitative comparison of different width of MLP in the M-Net, tested on the AVD [2]. All MLP have the same depth of 4 layers.

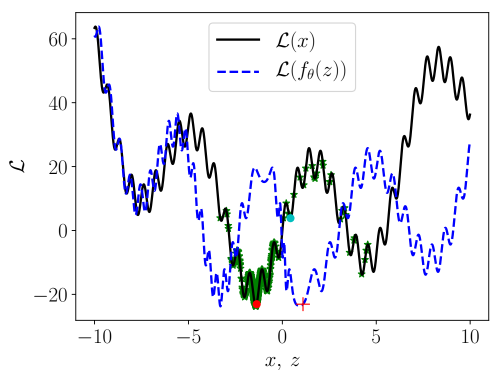

Figure 5. A 1D example to show the effectiveness of our neural-network-based conversion of a optimization problem into a higher dimension one. Red point shows the optimal solution found in the converted problem, while the cyan point shows the gradient descent optimum in the original problem. Please refer to Section 4 for a detailed explanation of the figure. Best viewed in color.

2. More Results on 2D Simulated Point Cloud

Figure 6 shows additional qualitative comparisons of registration results on the 2D simulated dataset. As shown, both the direct optimization and the incremental ICP with point-to-point metric fails to register all point clouds. The proposed DeepMapping, however, is more robust and accurate than baseline methods. The last two rows in Figure 6 show two cases where all methods fail to find correct registration.

Table 1 reports the average execution time and the success rate for different methods to register 128 point clouds. We run DeepMapping and the direct optimization for 3000 epochs. A registration of multiple point clouds is considered successful if the ATE is less than a threshold of 20 pixel, which is about 2% of the image size (1024×1024). The success rate is then defined as the ratio of the number of successful registration to the total number of test cases. All methods are tested on a machine with a 3.3GHz Intel® Core™ i9-7900X CPU. We use an Nvidia TITAN XP for "training" DeepMapping and the direct optimization. While DeepMapping seems slow, the method in fact converges very quickly: within 500 epochs (4.8 minutes), our ATE error is already smaller than of baseline methods.

	DeepMapping	Direct Opt.	ICP (Point)	ICP (Plane)
Runtime	29min	17min	6.48s	12.35s
Succ. Rate	84.2%	31.5%	36.0%	53.3%

Table 1. Average runtime for 3000 epochs and success rate for different methods tested on the 2D simulated dataset.

We also test two initialization methods, random initialization and zero initialization, for the direct optimization. Both methods have worse performances than the initialization which is the same as DeepMapping.

3. More Results on the Active Vision Dataset

Figure 7 shows additional visual comparison tested on the AVD [2]. The black ellipse highlights the region corresponding to misaligned parts from baseline methods. Table 2 lists the average execution time for 3000 epochs and the success rate to register 16 point clouds from the AVD. In this experiment, A registration is considered to be successful if the ATE is less than $450mm$. The hardware configuration is identical to those in Section 2. As shown, the success rate from DeepMapping is higher than the rate from multiway registration [3].

	DeepMapping	Direct Opt.	Multiway [3]
Runtime	24min	20min	42.49s
Succ. Rate	80.0%	77.1%	58.1%

Table 2. Average runtime for 3000 epochs and success rate for different methods tested on the AVD.

4. Interpretation of Our Method

Given the differences between the problem formulations in (1) and (2) (in the main paper), it is natural to ask why we use the neural network f_θ to estimate the sensor poses $\mathbf{T}$ instead of directly optimizing them. In this section, we attempt to provide a simple potential interpretation of the benefit introduced by our formulation.

The basic inspiration comes from an optimization technique known as changing variables [1] that can convert an originally non-convex optimization problem to an equivalent convex one. In their example, a geometric program can be converted to a linear program by substituting exponential functions as original variables. In our formulation, we combine this idea with neural networks by replacing the optimization variables $\mathbf{T}$ with $f_\theta(\mathbf{S})$ and transforming the objective function from (1) to (2) (in the main paper). While we do not expect that the replacement of variables $\mathbf{T}$ with neural network parameters θ yields a convex problem, we observe that this transformation is beneficial to finding the optimal solution to the original problem.

We conduct a simple 1D experiment to illustrate this observation. Consider a problem of finding the optimal value of $x \in \mathbb{R}$ that minimizes $\mathcal{L}(x)$, a non-convex objective function with multiple local minima shown as the black line in Figure 5. Specifically, the objective function is defined as

$$\mathcal{L}(x) = \frac{1}{2}x^2 + 5\sin(10x) + 20\sin(x).$$

In this experiment, we compare two optimization methods, i.e., the proposed network-based optimization and the direct optimization. For network-based optimization, we introduce an MLP, f_θ, which consists of FC(10)-FC(20)-FC(30)-FC(40)-FC(1). Each MLP layer is followed by an ELU [4] activation function except for the output layer. The MLP has one node in the input and the output layer to replace the variable x with $f_\theta(z)$, resulting in another problem with optimization variable z. To ensure the same starting point, the direct optimization is initialized with $x_0 = f_{\theta_0}(z_0)$ where θ_0 and z_0 are the initial values of network parameters θ and variable z, respectively. We use gradient descent with a learning rate of 2×10^{-4} and run 1000 iterations. For the network-based optimization, we jointly update the network parameters θ and z.

The cyan point shows the result using gradient descent optimization that is performed directly on $\mathcal{L}(x)$, which is trapped in a local minimum. The function $\mathcal{L}(f_{\theta^\star}(z))$ with the optimal $\theta^\star$ found in the network-based optimization is plotted as the blue dash line in Figure 5. We take $f_{\theta^\star}(z^\star)$ to retrieve the optimal point $x^\star$ for $\mathcal{L}(x)$. The red plus and red circle in Figure 5 correspond to $z^\star$ and $x^\star$, respectively. The green star symbols show the values of x during the 1000 gradient-descent iterations.

Notice the distribution of the green star symbols, visualizing the "sampled locations" in the domain, x, of the original problem. It is interesting to see that our conversion leads to a wider search range in the original problem domain, while keeping the same number of function evaluations of the original problem $\mathcal{L}(\cdot)$ as in direct gradient descent.

References

[1] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd. A rewriting system for convex optimization problems. *J. Control and Decision*, 5(1):42–60, 2018. 2

[2] P. Ammirato, P. Poirson, E. Park, J. Kosecka, and A. C. Berg. A dataset for developing and benchmarking active vision. In *Proc. the IEEE Intl. Conf. on Robotics and Auto.*, 2017. 1, 2, 5

[3] S. Choi, Q. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *IEEE Intl. Conf. Comp. Vision and Pattern Recog.*, June 2015. 1, 2

[4] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). In *Intl. Conf. Learning Representations*, 2015. 3

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Intl. Conf. Comp. Vision and Pattern Recog.*, June 2016. 1

[6] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *IEEE Intl. Conf. Comp. Vision and Pattern Recog.*, July 2017. 1

Figure 6. Additional visual comparisons of multiple point clouds registration from the 2D simulated dataset. The black lines are the trajectories of sensor. The third column shows occupancy maps that are estimated by the M-Net. The black, while, and gray pixels show the occupied, unoccupied, and unexplored locations, respectively. Note that the results of each trajectory cam be defined in arbitrary coordinate systems and do not necessarily aligned with ground truth. The last two rows show the failure cases. Best viewed in color.

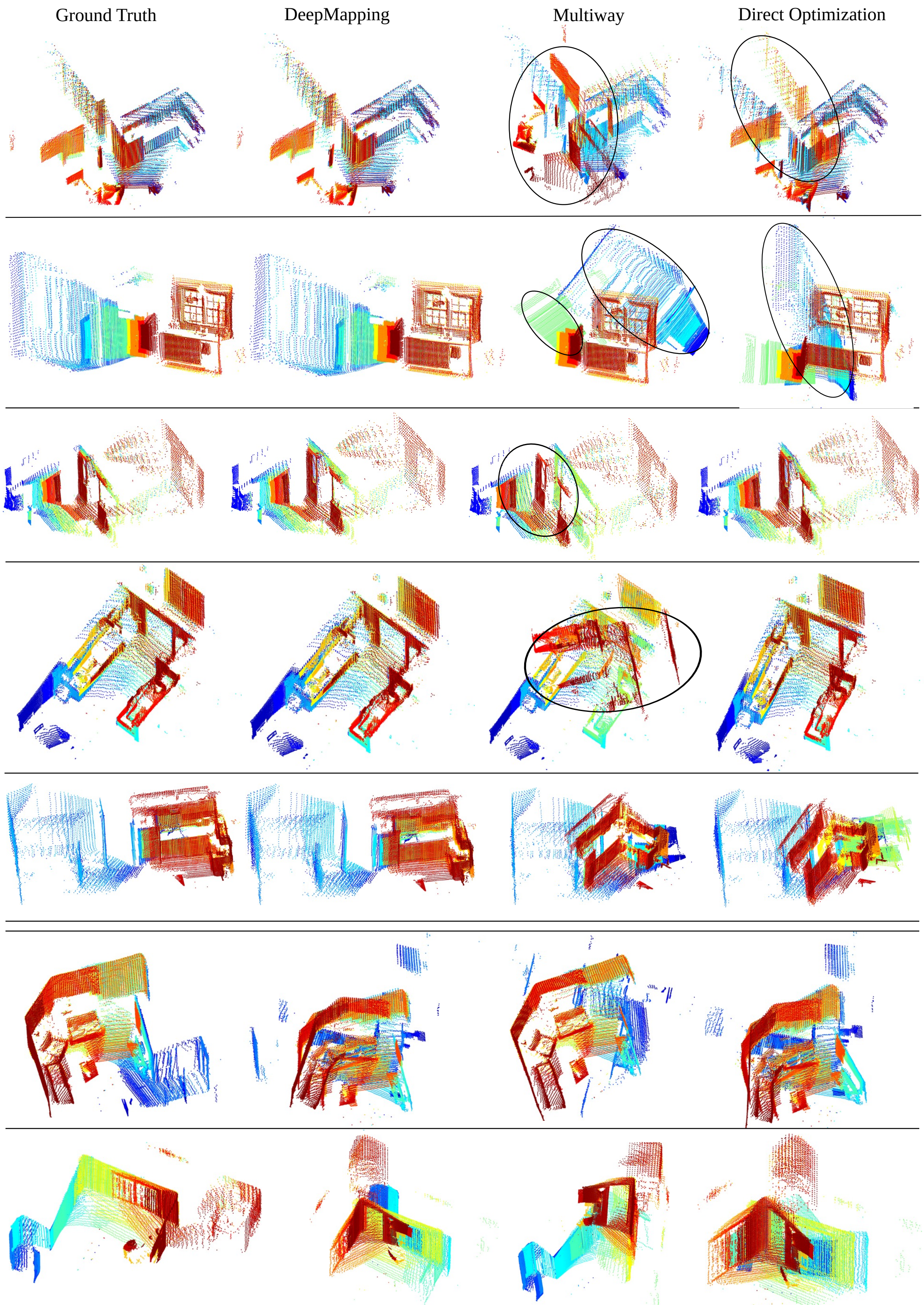

Figure 7. Additional visual comparisons from the AVD [2]. The black ellipses highlight the misaligned parts in baselines. Each color represents one point cloud. The last two rows show the failure cases. Best viewed in color.