3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions APPENDIX

Andy Zeng[1] Shuran Song[1] Matthias Nießner[2] Matthew Fisher[2,4] Jianxiong Xiao[3] Thomas Funkhouser[1]
[1]Princeton University [2]Stanford University [3]AutoX [4]Adobe Systems

http://3dmatch.cs.princeton.edu

1. Supplementary Material

In this section, we present several statistics of the RGB-D reconstruction datasets used to generate training correspondences for 3DMatch, the implementation details of our network, and run-time statistics relevant to the experiments discussed in the main paper.

1.1. RGB-D Reconstruction Datasets

As mentioned in Sec. 3 of the main paper, we use registered depth frames of 62 different real-world scenes collected from Analysis-by-Synthesis [6], 7-Scenes [5], SUN3D [7], RGB-D Scenes v.2 [4], and Halber *et al*. [3], with 54 scenes for training and 8 scenes for testing. For selected scenes of the SUN3D dataset, we use the method from Halber *et al*. to estimate camera poses. For the precise train/test scene splits, see our project webpage [1]. In Fig. 2, we show top-down views of the completed reconstructions. They are diverse in the environments they capture, with local geometries at varying scales, which provide a diverse surface correspondence training set for 3DMatch. In total, there are 214,569 depth frames over the 54 training scenes, most of which are made up of indoor bedrooms, offices, living rooms, tabletops, and restrooms. The depth sensors used for scanning include the Microsoft Kinect, Structure Sensor, Asus Xtion Pro Live, and Intel RealSense.

The size of the correspondence training set correlates with the amount of overlap between visible surfaces from different scanning views. In Fig. 1, we show the average distribution of volumetric voxels (size 0.02^3m) on the surface vs. the number of frames in which the voxels were seen by the depth sensor. We plot this distribution averaged over the 54 training scenes (left) and illustrate a heat map of over two example reconstructions (right), where a warmer region implies that the area has been seen more times. The camera trajectories are plotted with a red line.

1.2. Implementation Details

We implement the 3DMatch network architecture in Marvin [8], a lightweight GPU deep learning framework that supports 3D convolutional neural networks. Weights of the network are initialized with the Xavier algorithm [2], and biases are initialized to 0. We train by SGD with momentum using a fixed learning rate of 10^{-3}, a momentum of 0.99, and weight decay of 5^4. Random sampling matching and non-matching 3D training patches from reconstructions (refer to Sec. 3) is performed on-the-go during network training. We used a batch size of 128, contrastive margin of 1, no batch or patch normalization. Our reference model was trained for approximately eight days on a single NVIDIA Tesla K40c, over 16 million 3D patch pairs, which includes 8 million correspondences and 8 million non-correspondences.

1.3. Run-time Information

The following run-times are reported from the implementations used in our experiments. We did not optimize for speed.
TDF Conversion. As mentioned in Sec 4.1, a local 3D patch around an interest point is represented as a $30 \times 30 \times 30$ voxel grid of TDF values (in all of our experiments, truncation margin is 5 voxels). Converting a point cloud spanning 0.3m^3 of a depth frame into a TDF voxel grid (using CUDA-enabled GPU acceleration) takes anywhere from 3 - 20 milliseconds, depending on the density of the point cloud. We observe an average conversion time of 8.9 milliseconds per patch from the keypoint matching benchmark in Sec. 5.1 on an NVIDIA Tesla K40c.

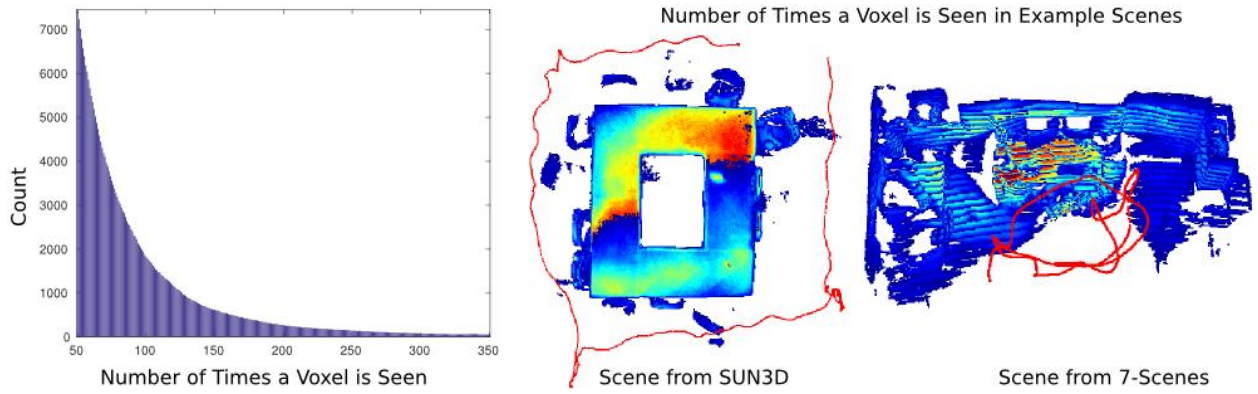

Figure 1. Average distribution (by count) of volumetric voxels (size 0.02^3m) on the surface vs. the number of frames in which the voxels were seen by the depth sensor in each scene (left). We also illustrate a heat map over the reconstructions of example scenes (right), where a warmer region implies that the surface voxels in area has been seen more times. Regions seen less than 50 times are removed from the illustrations. The camera trajectories are drawn in red.

3DMatch Descriptor. Computing a 3DMatch descriptor (*e.g.* ConvNet forward pass) for a single $30 \times 30 \times 30$ TDF voxel grid takes an average of 3.2 milliseconds with Marvin.

Geometric Registration. To register two surfaces, we randomly sample n keypoints per surface. $n = 5000$ for scene fragment surface registration (Sec. 5.2.1 - Sec. 5.2.2) and $n = 1000$ for model alignment in the Amazon Picking Challenge (Sec. 5.3). Finding keypoints with mutually closest 3DMatch descriptors takes 2 seconds or less, while RANSAC for estimating rigid transformation can take anywhere from 2 seconds up to a minute depending on the number of RANSAC iterations and rate of convergence. These numbers are reported using a single CPU thread on an Intel Core i7-3770K clocked at 3.5 GHz.

References

[1] Website for code and data: http://3dmatch.cs.princeton.edu/. 1

[2] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010. 1

[3] M. Halber and T. Funkhouser. Structured global registration of rgb-d scans in indoor environments. *arXiv preprint arXiv:1607.08539*, 2016. 1

[4] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057. IEEE, 2014. 1

[5] 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 *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2930–2937. IEEE, 2013. 1

[6] J. Valentin, A. Dai, M. Nießner, P. Kohli, P. Torr, S. Izadi, and C. Keskin. Learning to navigate the energy landscape, 2016. 1

[7] J. Xiao, A. Owens, and A. Torralba. SUN3D: A database of big spaces reconstructed using SfM and object labels. 2013. 1

[8] J. Xiao, S. Song, D. Suo, and F. Yu. Marvin: A minimalist GPU-only N-dimensional ConvNet framework. 2016. Accessed: 2015-11-10. 1

Figure 2. Visualizing several RGB-D reconstructions of the datasets used to train 3DMatch in our experiments. Each dataset contains depth scans of different environments with different local geometries at varying scales, registered together by different reconstruction algorithms. These datasets provide a diverse surface correspondence training set with varying levels of sensor noise, viewpoint variance, and occlusion patterns. Color for visualization only.