Multi-user, Scalable 3D Object Detection in AR Cloud

Siddharth Choudhary, Nitesh Sekhar, Siddharth Mahendran, Prateek Singhal
Magic Leap, CA, USA
{schoudhary,nsekhar,smahendran,psinghal}@magicleap.com

Abstract

As AR Cloud gains importance, one key challenge is large scale, multi-user 3D object detection. Current approaches typically focus on the single-room, single-user scenarios. In this work, we present an approach for multi-user and scalable 3D object detection, based on distributed data association and fusion. We use an off-the-shelf detector to detect object instances in 2D and then combine them in 3D, per object while allowing asynchronous updates to the map. The distributed data association and fusion allows us to scale the detection to a large number of users concurrently, while maintaining a lower memory footprint without loss in accuracy. We show empirical results, where the distributed and centralized approaches achieve comparable accuracy on the ScanNet dataset while reducing the memory consumption by a factor of 15.

1. Introduction

3D object detection is a fundamental problem in computer vision and robotics with modern applications like autonomous driving and augmented reality (AR) where the ability to detect objects of interest quickly and accurately play a very important role. There are two popular approaches to do 3D object detection. One is the *reconstruct-then-recognize* approach, where we reconstruct the entire scene from multiple images [33, 36] and then detect objects of interest in the resulting point-cloud or mesh [5, 12, 13, 14, 25, 26, 27, 28, 35, 39]. This however limits the object resolution because we are now reconstructing an entire scene instead of a single object and we need the entire image sequence as input. A different way is a *recognize-and-reconstruct* approach, where we simultaneously do reconstruction and object recognition [4, 17, 18, 21, 24, 29, 30, 32, 34, 37, 38, 40]. The advantage of such an approach is that we do not have to wait for the entire sequence of images any more but the disadvantage is the object resolution is again limited by voxel resolution of the scene since scene and objects both are reconsructed. Another key limitation of such an approach is that sequence of images input to the algorithm need to be spatially contiguous. In this work, we propose an algorithm that follows the *recognize-and-reconstruct* approach but with objects as the primary output at any desired resolution, without storing the background mesh, without the assumption of spatial contiguity in the input sequence of images and extended to multiple users.

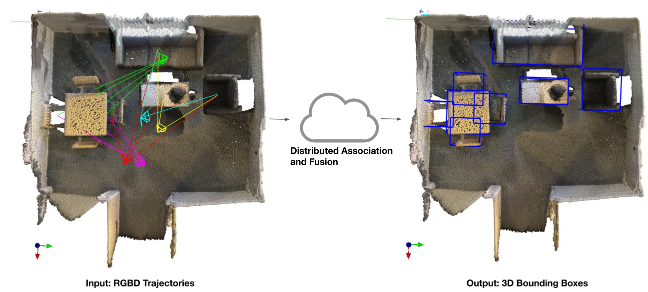

Figure 1: Overview of the problem of multi-user, scalable 3D object detection. (left) Shows multiple RGB-D camera trajectories using different colors. Camera frustum of each trajectory at a particular timestamp is connected using straight lines to the object centers visible from that pose. (center) Poses along with RGB and Depth frames for each trajectory are streamed through our proposed algorithm which estimates the 3D bounding box of each object (right) shown in blue.

Multi-user, scalable 3D object detection is not just an academic one but a real-world problem of current interest. Consider the scenario of multiple users with AR devices in the same room looking at some common objects. Each user now contributes an image containing some objects in the room from different viewpoints which when all grouped together temporally lack spatial contiguity. Given this input sequence of images, the users expect a dense reconstruction of all objects in the room along with their 3D locations in a common world co-ordinate system. Also, this needs to be done not just for a single room with a few users but with much larger environments at a building or city scale with hundreds or thousands of users. We need to be scalable along three important axes: *maps*, *users* and *objects*.

Contributions. Our main contribution in this work is a proposed solution for the above problem of *Multi-user,* *Scalable 3D Object Detection*. We achieve this using *Distributed Association and Fusion* where now instead of matching two consecutive images (assuming spatial contiguity), we match objects detected in the current image with currently detected and reconstructed 3D objects. The distributed data association step matches detections in the current frame with 3D objects visible that frame to find if a new detected corresponds to an existing object. The distributed fusion step then combines the new detection with the matched existing object. The association is distributed at a per-image level and the fusion is distributed at a per-object level allowing for high scalability.

Assumptions. In this work, we assume that the 6DOF poses for each user's trajectory in the same global coordinate frame is available. This is available either using Magic Leap's persistent coordinate frames [15], Azure spatial anchor [19] or decentralized mapping techniques [1, 3, 9, 22].

Related Work. The robotics community has actively worked in building distributed maps [1, 3, 6, 9, 22], where multiple agents communicate to reach consensus on a common map (see [2] for an exhaustive survey). Map API [7] propose a distributed mapping framework for building large scale maps with multiple agents. We build upon similar distributed mapping frameworks, but represent the map as objects achieving higher efficiency and scalability.

Semantic SLAM uses semantic entities like objects and planes as landmarks during the SLAM pipeline [4, 8, 11, 17, 23, 31, 37] and show that it helps improve the accuracy of the estimated trajectory. Fusion++ [17] combines object based SLAM, with object level TSDF for high quality object reconstructions. Panoptic Fusion [21] builds a semantic map of objects and stuff in an online framework. Most of the above approaches store dense scene representation along with objects and are thus not scalable to large maps.

2. Distributed 3D Object Detection

The algorithm for distributed 3D object detection is shown in Algorithm 1. Given an RGB-D input frame $\mathcal{F}$, we first predict and localize objects of interest in the image and their corresponding instance masks $\mathcal{M}$. These instance masks are then voxelized by back-projecting into the world coordinate system under corresponding pose and depth. Next, we find all the existing 3D objects $\mathcal{O}_\mathcal{F}$ visible in the current view frustum and associate the 2D detections $\mathcal{M}$ found in the frame with existing 3D objects. If associated, we fuse the existing 3D objects with new 2D measurements, otherwise we create a new 3D object. Figure 2 shows the corresponding pipeline.

Since the 3D object detection algorithm runs on every frame independently, it can easily be parallelized with each frame being processed on different nodes. Object level locking/unlocking handles any race conditions that might arise and ensures that the same object is not updated at the same time by different nodes.

Algorithm 1: Distributed 3D Object Detection

for all frame $\mathcal{F}$ **do in parallel**
- $\mathcal{M} \leftarrow$ 2D instances of objects of interest in $\mathcal{F}$.
- $\mathcal{O}_\mathcal{F} \leftarrow$ Find all current 3D objects visible in $\mathcal{F}$'s frustum.
- Lock objects $\boldsymbol{O} \in \mathcal{O}_\mathcal{F}$.
- Associate $\mathcal{M}$ with $\mathcal{O}_\mathcal{F}$ using **Distributed Data Association** to get $\mathcal{M}_A \subseteq \mathcal{M}$.
- Fuse $\mathcal{M}_A$ with $\mathcal{O}_\mathcal{F}$ using **Distributed Fusion**.
- Create new 3D objects for $\mathcal{M} - \mathcal{M}_A$.
- Unlock objects $\boldsymbol{O} \in \mathcal{O}_\mathcal{F}$.

end for

Object Representation. We denote the set of objects using $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots\}$. Each object instance is represented using a set of voxels $\mathcal{V} = \{\mathcal{V}^1, \mathcal{V}^2, \ldots\}$ where $\mathcal{V}^i$ represents the i_{th} voxel. Each voxel is represented using $\mathcal{V}^i = \{\mathbf{v}, \boldsymbol{p}\}$ where $\mathbf{v} = (v_x, v_y, v_z)$ represents the 3D location of the voxel and $\boldsymbol{p}$ represents the occupancy weight.

Distributed Data Association. We use Hungarian method to associate detections in the current frame $\mathcal{F}$ with the object $\mathcal{O} \in \mathcal{O}_\mathcal{F}$ visible in the current frame [20]. We find the assignment between detections $\mathcal{M}$ and objects $\mathcal{O}_\mathcal{F}$ by minimizing a cost matrix $\boldsymbol{C}$ where $\boldsymbol{C}(i, j)$ is inversely proportional to the number of common voxels $\mathcal{V}$ between detection $i \in \mathcal{M}$ and object $j \in \mathcal{O}_\mathcal{F}$. The set of detections which gets successfully associated with an existing object is denoted by $\mathcal{M}_A \in \mathcal{M}$. $\mathcal{M} - \mathcal{M}_A$ denotes non-associated detections.

Distributed Fusion. Given the assignment from the Distributed Data Association algorithm, we fuse each object instance $\in \mathcal{M}_A$ by updating the corresponding associated object voxel weights and extracting the updated object instances. We resolve the conflict between neighboring instances by assigning each voxel to only one instance at a particular timestamp. Category label at a particular timestamp is the mode of all detection labels for that voxel until that timestamp. For all non-associated detections $\mathcal{M} - \mathcal{M}_A$, new objects are created and added to $\mathcal{O}$. The resulting objects and their corresponding voxels are refined by removing noisy voxels. Given the refined objects, we estimate the tightest fitting 3D bounding box for that object.

3. Experimental Results

We show results on the ScanNet benchmarking dataset [10]. We use mean Average Precision (mAP) evaluated at an intersection-over-union (IoU) of 0.25 as the metric to evaluate 3D object detection. mAP@0.25 is a standard metric used by recent works to compare 3D object detection approaches [10]. 3D IoU of 0.25 is equivalent to an overlap of 0.75 along all three bounding box axes.

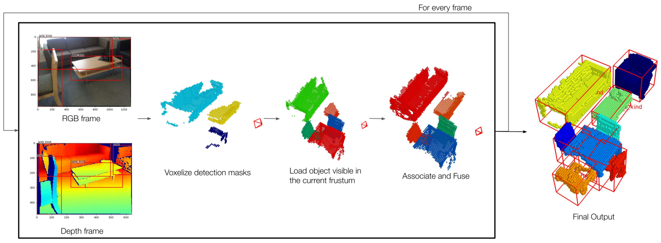

Figure 2: Overview of algorithm for Distributed Association and Fusion. For every frame, we detect objects of interest, voxelize the corresponding depth pixels, load and lock the visible objects, associate the visible objects with the detected objects, fuse into the visible objects given association and finally unlock the locked objects.

Given a ScanNet scene, we have a stream of (pose, RGB, depth) data that is temporally ordered and spatially contiguous. In the *centralized approach*, we directly use this as an input to the standard association and fusion components. This gives us a baseline performance of 3D object detection using mAP@0.25 metric, when the entire scene was recorded by just a single user. For the *distributed approach*, we simulate multiple users by breaking up the scene data into multiple segments which are then shuffled to break inter-segment temporal continuity. Note that the overall data content is identical in both centralized and distributed approaches with the key difference being the manner in which this data is received and processed. We repeat this procedure for $M = 10$ monte-carlo runs and report the mean & variance of these multiple runs. We show results with number of users as $K = 10, 50, 100$. We compare accuracy for the centralized and distributed algorithms.

We use predicted masks generated using an off-the-shelf Mask-RCNN model pre-trained on the COCO dataset [16]. We perform our experiment on 5 object categories: Chair, Table, Laptop, Couch and TV, as these are quite frequently occurring categories and are also common between COCO and ScanNet datasets. The list of scenes used in the experiments is available in the Appendix (§6).

Accuracy. Table 1 shows the performance of the centralized and distributed approaches. We report mAP@0.25 averaged across all the scenes and all the runs in column 3. In column 4, we report the variance across all the monte-carlo runs. From Table 1, we can see that the distributed approach (with more than 1 user) performs similar to the centralized approach (with 1 user). Variance across runs from distributed approach is low which confirms that the distributed association and fusion algorithm is robust to random shuffling of the trajectory. It also shows that RGB, depth frames and poses do not have be time sequenced for the algorithm to work and sensor data from multiple users can be fused in parallel using the distributed association and fusion algorithm. Example scene snapshots are shown in the supplementary.

Approach	#Users	mAP@0.25	Variance
Centralized	1	0.669	**0**
Distributed	10	0.646	0.0027
Distributed	50	**0.672**	0.0025
Distributed	100	0.664	0.0044

Table 1: Comparison between centralised and distributed approaches using predicted masks. Best results in bold.

Metric	Dense Mesh (MB)	Object-level Map & Sparse Map (MB)	Ratio
Mean	201.3	**12.001**	16.77
Median	198.16	**11.63**	17.07

Table 2: Comparison between memory requirement of dense reconstructed mesh and object-level map (including sparse map) (in MB).

The slight difference between mAP performance across the number of users (Table 1) is due to the locally optimal decision taken by the data association algorithm at each frame. A globally optimal association algorithm reduces the system scalability. A more robust yet scalable association algorithm is an appropriate direction for future work.

Memory. Table 2 shows a comparison between the memory requirement for a map that uses a dense reconstructed mesh and object-level map. We also include the memory required to store the sparse map (≈ 10 MB) along with object level map, since sparse map is used to estimate each user's trajectory in a common coordinate frame.

As can be seen, the object-level (+ sparse) map has significantly smaller memory footprint. We report the mean, median and ratio between the two maps averaged across all ScanNet scenes.

Discussion. Table 1 and 2 show that using objects along with the proposed distributed association and fusion algorithm (Algorithm 1) enables scalability along the three important axes: *maps*, *users*, *objects*. **(1)** Our proposed approach requires minimal additional memory storage to store objects on top of existing sparse mapping approaches [15, 19]. Therefore, it can scale to *large maps* as compared to dense reconstructed mesh based approaches. **(2)** Each user's RGB-D frames can be processed in parallel on different nodes without affecting the accuracy (Table 1). Therefore, the number of nodes can be scaled up and down depending on the number of *users*. **(3)** Each object can be updated asynchronously with all the other objects in the map and therefore enables scaling with the number of *objects*. This is in contrast with dense reconstructed mesh based approaches which would require to lock the complete map before the map can be updated. Object level locks enable object level scalability.

4. Conclusion

We presented an approach for scalable 3D object detection for an asynchronous and distributed system running in the cloud. Our empirical results showed almost negligible variance between the proposed distributed and a baseline centralized system, with around 15 times smaller memory footprint.

5. Acknowledgements

This work was possible due to contributions from an extended team of software engineers and researchers including but not limited to Arumugam Kalaikannan, Nitin Bansal, Manushree Gangwar, Shiyu Dong, Divya Ramnath, Khushi Gupta and Adithya Rao.

References

[1] R. Aragues, L. Carlone, G. Calafiore, and C. Sagues. Multi-agent localization from noisy relative pose measurements. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 2

[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age. *IEEE Transactions on Robotics (TRO)*, 2016. 2

[3] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H.I. Christensen, and F. Dellaert. Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models. *The International Journal of Robotics Research*, 2017. 2

[4] S. Choudhary, A. J. B. Trevor, H. I. Christensen, and F. Dellaert. SLAM with object discovery, modeling and mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014. 1, 2

[5] C. Choy, J.Y. Gwak, and S. Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[6] T. Cieslewski, S. Choudhary, and D. Scaramuzza. Data-Efficient Decentralized Visual SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 2

[7] T. Cieslewski, S. Lynen, M. Dymczyk, S. Magnenat, and R. Siegwart. Map API - scalable decentralized map building for robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015. 2

[8] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. M. Montiel. Towards semantic SLAM using a monocular camera. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2011. 2

[9] A. Cunningham, V. Indelman, and F. Dellaert. DDF-SAM 2.0: Consistent distributed smoothing and mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2013. 2

[10] A. Dai, A.X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[11] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. D. Reid. Dense Reconstruction using 3D Object Shape Priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[12] B. Graham, M. Engelcke, and L. van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[13] H. Ji, A. Dai, and M. Nießner. 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[14] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander. Joint 3D Proposal Generation and Object Detection from View Aggregation. *IROS*, 2018. 1

[15] Magic Leap. Magic leap's persistent coordinate frame. `https://developer.magicleap.com/en-us/learn/guides/persistent-coordinate-frames`. 2, 3

[16] F. Massa and R. Girshick. maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. `https://github.com/facebookresearch/maskrcnn-benchmark`, 2018. 3

[17] J. Mccormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric Object-Level SLAM. In *International Conference on 3D Vision (3DV)*, 2018. 1, 2

[18] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. SemanticFusion: Dense 3D semantic mapping with convolutional neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 1

[19] Microsoft. Azure anchors. `https://docs.microsoft.com/en-us/azure/spatial-anchors/overview`. 2, 3

[20] James Munkres. Algorithms for the assignment and transportation problems, 1957. 2

[21] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji. PanopticFusion: Online Volumetric Semantic Mapping at the Level of Stuff and Things. *CoRR arXiv:1903.01177*, 2019. 1, 2

[22] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2009. 2

[23] L. Nicholson, M. Milford, and N. Sünderhauf. QuadricSLAM: Dual Quadrics From Object Detections as Landmarks in Object-Oriented SLAM. *IEEE Robotics and Automation Letters*, 2019. 2

[24] S. Pillai and J. Leonard. Monocular SLAM Supported Object Recognition. In *Proceedings of Robotics: Science and Systems (RSS)*, 2015. 1

[25] C.R. Qi, W. Liu, C. Wu, H. Su, and L.J. Guibas. Frustum PointNets for 3D Object Detection from RGB-D Data. *CoRR arXiv:1711.08488*, 2017. 1

[26] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CoRR arXiv:1612.00593*, 2016. 1

[27] C.R. Qi, L. Yi, H. Su, and L.J. Guibas. Pointnet++: Deep

hierarchical feature learning on point sets in a metric space. *CoRR arXiv:1706.02413*, 2017. 1
[28] Wald J. Sturm J. Navab N. Tombari F. Rethage, D. Fully-Convolutional Point Networks for Large-Scale Point Clouds. In *IEEE European Conference on Computer Vision (ECCV)*, 2018. 1
[29] M. Rünz and L. Agapito. Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple Objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 1
[30] M. Runz, M. Buffier, and L. Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2018. 1
[31] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2
[32] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1
[33] J.L. Schonberger and J.-M. Frahm. Structure-From-Motion Revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1
[34] J. Stückler and S. Behnke. Model learning and real-time tracking using multi-resolution surfel maps. In *AAAI Conference on Artificial Intelligence*, 2012. 1
[35] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 1
[36] R. Szeliski, B. Curless, S. M. Seitz, N. Snavely, Y. Furukawa, and S. Agarwal. Reconstructing Rome. *IEEE Computer*, 2010. 1
[37] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. 1, 2
[38] K. Tateno, F. Tombari, and N. Navab. When 2.5D is not enough: Simultaneous reconstruction, segmentation and recognition on dense SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016. 1
[39] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, and N. Trigoni. Learning Object Bounding Boxes for 3D Instance Segmentation on Point Clouds. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 1
[40] Z. Zeng, Y. Zhou, O. C. Jenkins, and K. Desingh. Semantic mapping with simultaneous object detection and localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. 1

6. Appendix

For experiments with all the scannet scenes, we used the following scenes: scene0025_00, scene0050_00, scene0050_01, scene0050_02, scene0095_01, scene0207_02, scene0231_01, scene0249_00, scene0251_00, scene0314_00, scene0334_00, scene0334_01, scene0334_02, scene0423_00, scene0423_01, scene0423_02, scene0430_00, scene0432_00, scene0432_01, scene0461_00, scene0500_01, scene0518_00, scene0549_01, scene0559_00, scene0559_01, scene0559_02, scene0568_00, scene0568_02, scene0575_02, scene0591_01, scene0609_00, scene0609_02, scene0609_03, scene0647_00, scene0647_01, scene0655_00, scene0655_01, scene0660_00, scene0671_01, scene0701_01, scene0701_02.