

Efficient Object-Level Semantic Mapping with RGB-D Cameras

Benchun Zhou



Contents

- Background
- Literature Review
- Method
- Experiment
- Conclusion

Background and Tasks

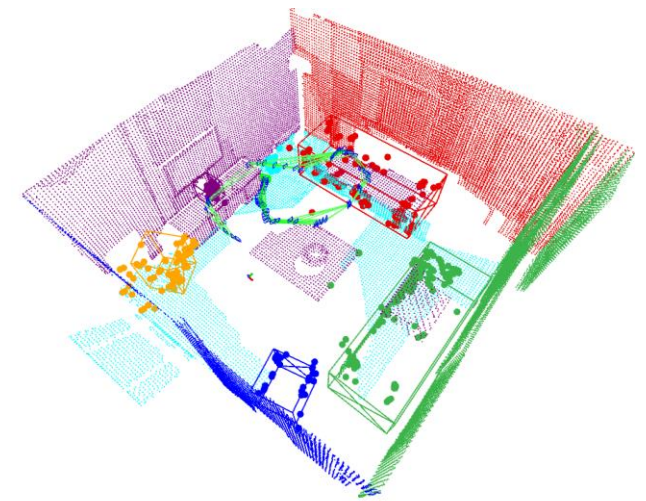
- Scene understanding in unknown environment (3D object segmentation)
- Efficient semantic mapping (voxblox++)
- Field experiment on Agiprobot project



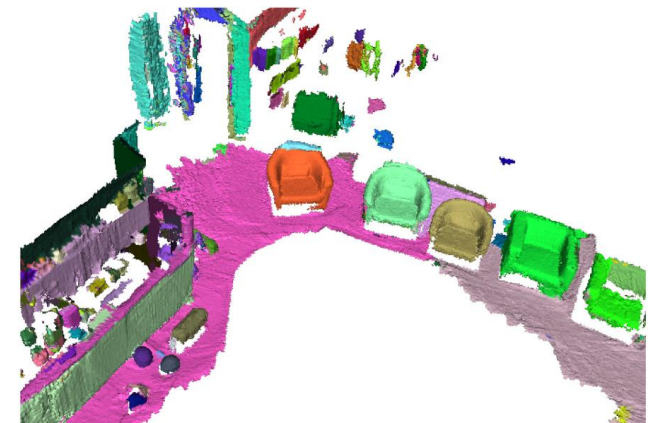
Literature Review

- Semantic mapping
- Object-level semantic mapping

Method	Year	Sensors	Object model	Semantic/object-oriented	Map Type	usage
Zhou et al Structure SLAM	2022	RGB-D	Geometric cuboid	Object-oriented	Feature point map	Localization
Sünderhauf et al. Meaningful map	2014	RGB-D	Point cloud model	Object-oriented	Point cloud map	Scene understanding
Nakajima et al, Efficient	2018	RGB-D	Point cloud model	Semantic	Point cloud map	Scene understanding
McCormac et al. Semantic Fusion	2016	RGB-D	Surfel-based model	semantic	Point cloud map	Scene understanding
Pham et al	2016	RGB-D	Voxel model	Object-oriented	Voxel map	Interactive application, object manipulation and picking
Grinvald et al Voxblox ++	2019	RGB-D	Voxel model	Object-oriented	Voxel map	Scene understanding, navigation
Li et al Incremental	2020	RGB-D	Voxel model	Object-oriented	Voxel map	Navigation and manipulation
Mascaro et al Voxblox fusion	2022	RGB-D LiDAR	Voxel model	Object-oriented	Voxel map	Interactive application

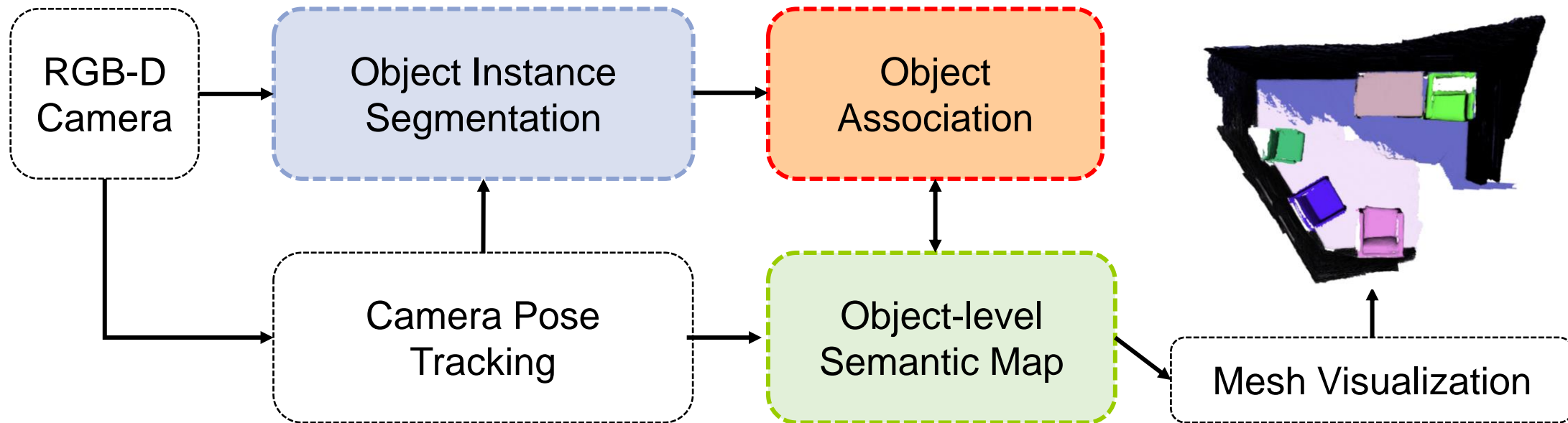


Feature Point Map
Structure SLAM 2019



Voxel object-oriented map
Voxblox++, 2019

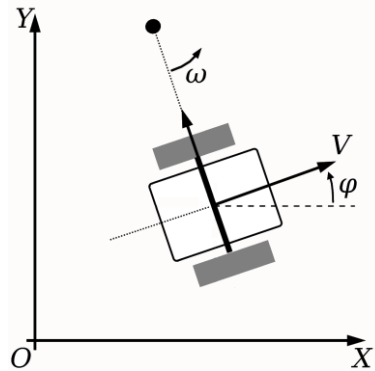
Method: Framework



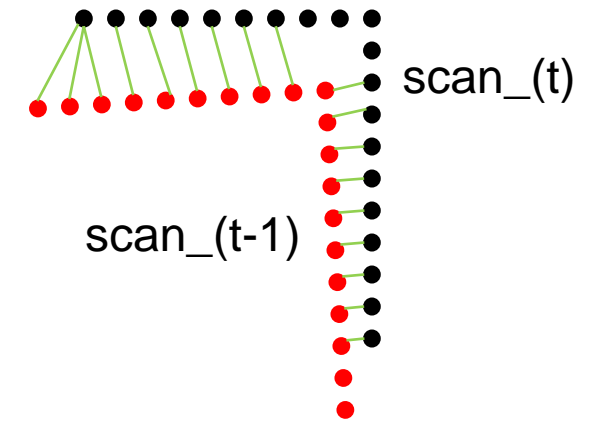
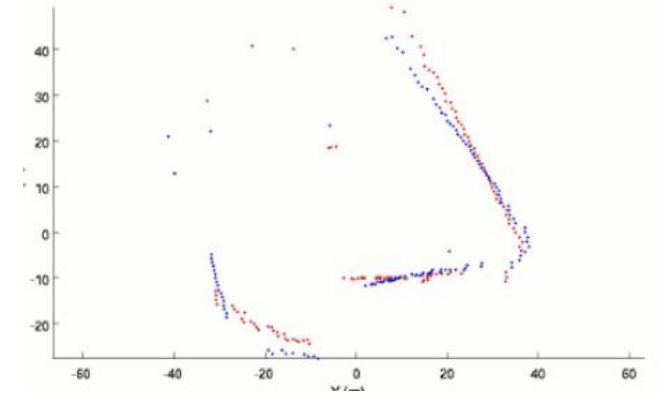
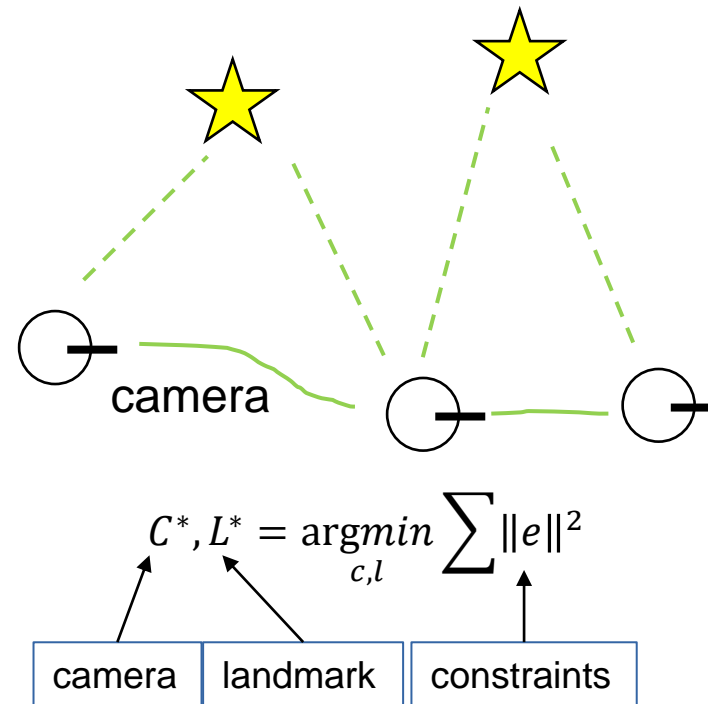
Method: Camera Poses Tracking

■ Camera poses tracking methods:

- Wheel encoder
- Feature tracking
- Laser scan matching



$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \cos \varphi & 0 \\ \sin \varphi & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix}$$

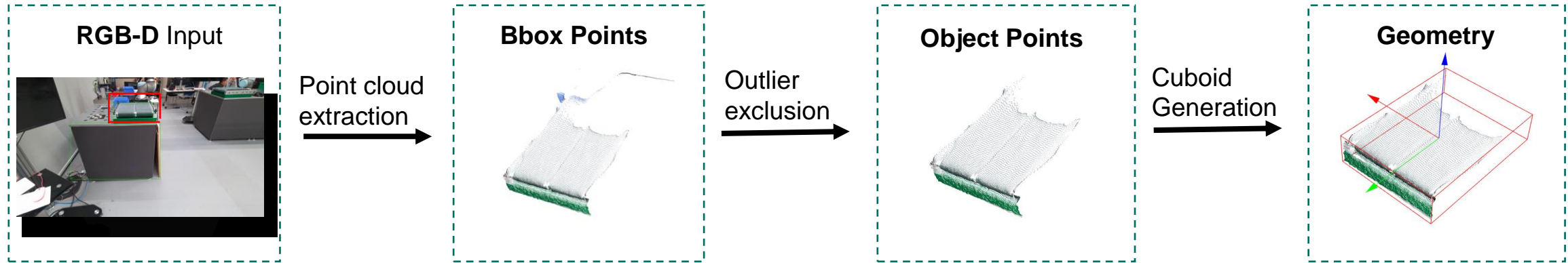


$$C^* = \operatorname{argmin}_{i,j} \sum \|P_i - P_j\|^2$$

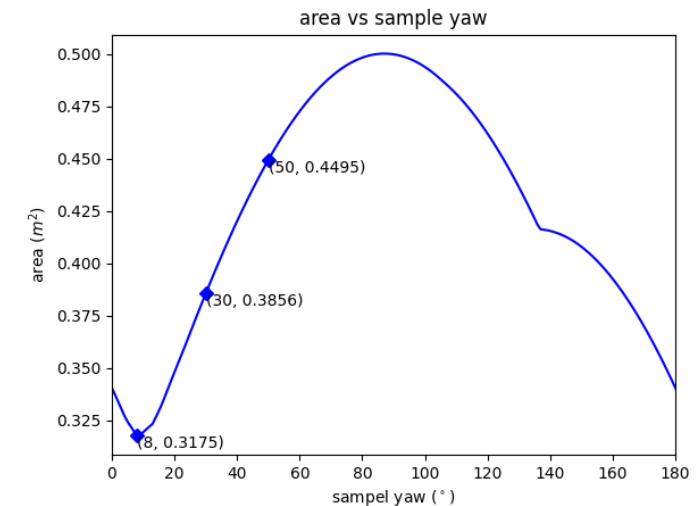
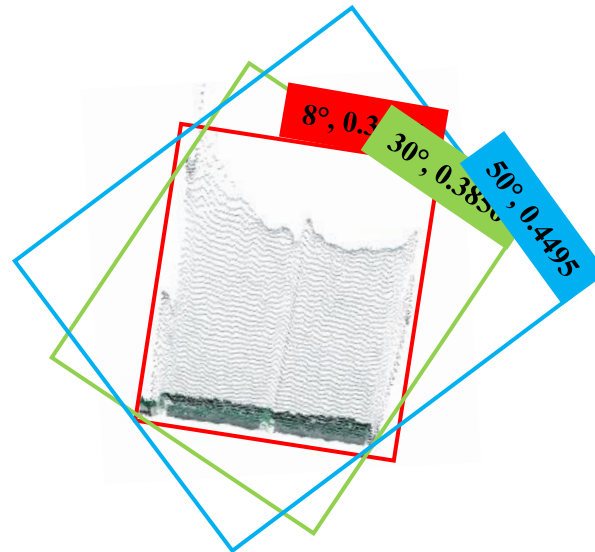
camera

matched points

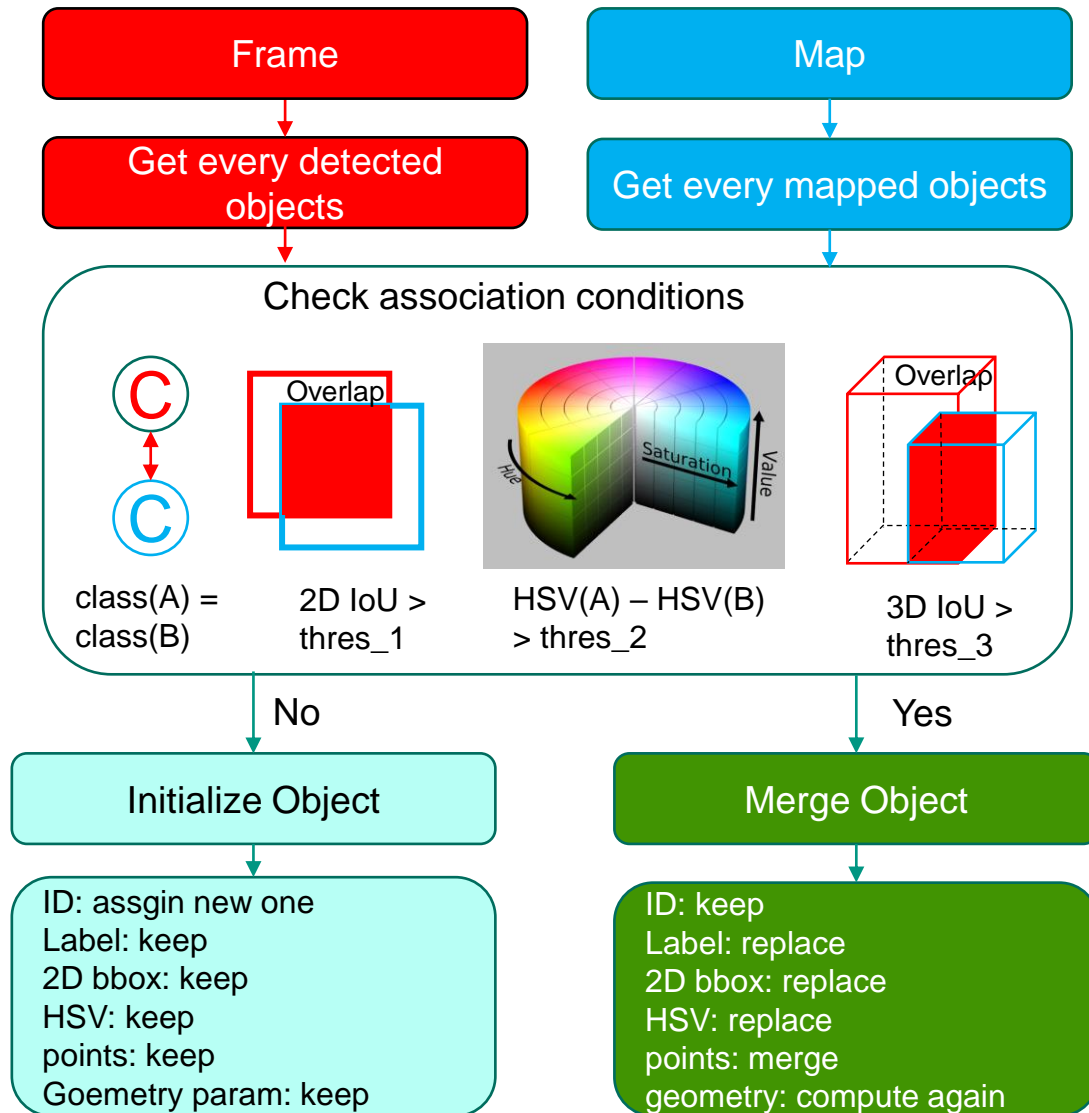
Method: Object Segmentation



Why do we detect conveyor?
>> small, can be fully observed
>> cuboid shape, accurate detection
>> can act as navigation goal



Method: Object Association



■ Association

- Label
- 3D bbox IoU (maybe label is not correct)
- 2D bbox IoU
- HSV
- >> Init frame objects to map objects or update existing objects

■ How to update object information:

- ID -> update related to associated result
- label-> same
- points-> merge, probabilistic
- HSV -> replace
- 9 DoF -> update, calculate again.

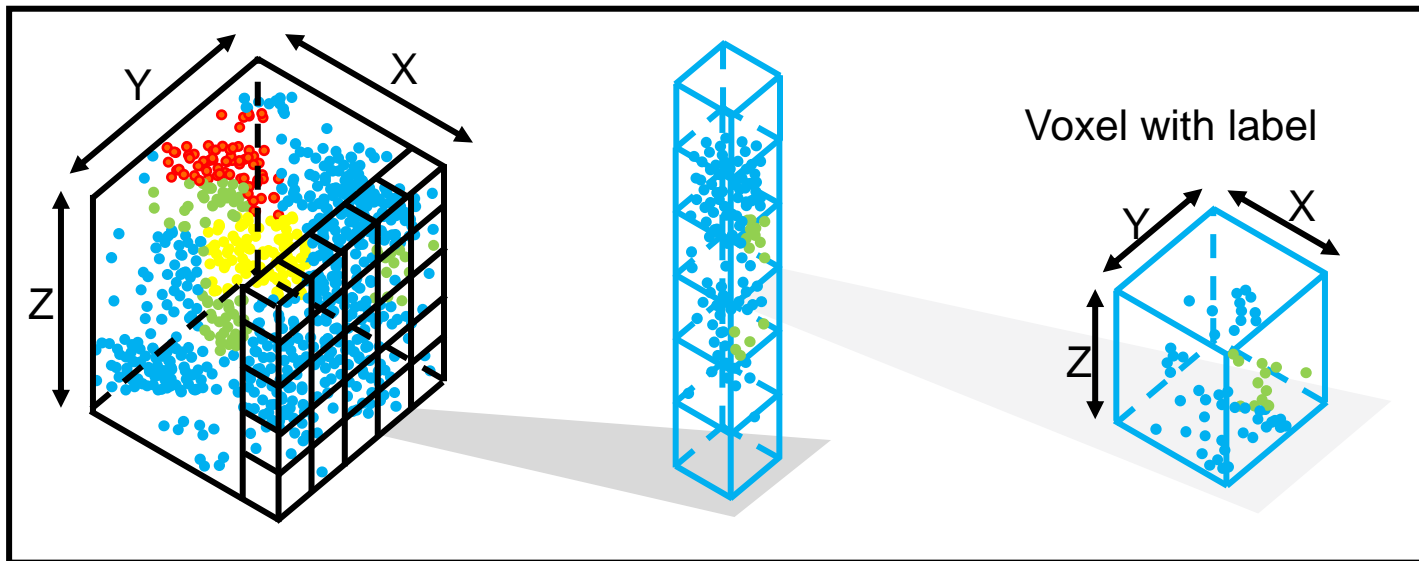
Methods: Object-level Semantic Map

■ Why voxel-based map

- 3D occupancy information: interactive application, such as navigation and picking
- Efficient: CPU real-time solution (from voxblox)

■ Convert object points to voxel-based model

- Voxel size: 2cm
- Voxel param: label info, occupancy info, position info



■ Voxel label update strategy

- Voxel labels collection

$$\varphi(v, l_i) \leftarrow \varphi(v, l_i) + 1$$

- Update voxel label by counting the maximum class

$$(L(v)) = \underset{i}{\operatorname{argmax}} \varphi(v, l_i)$$

Experiments

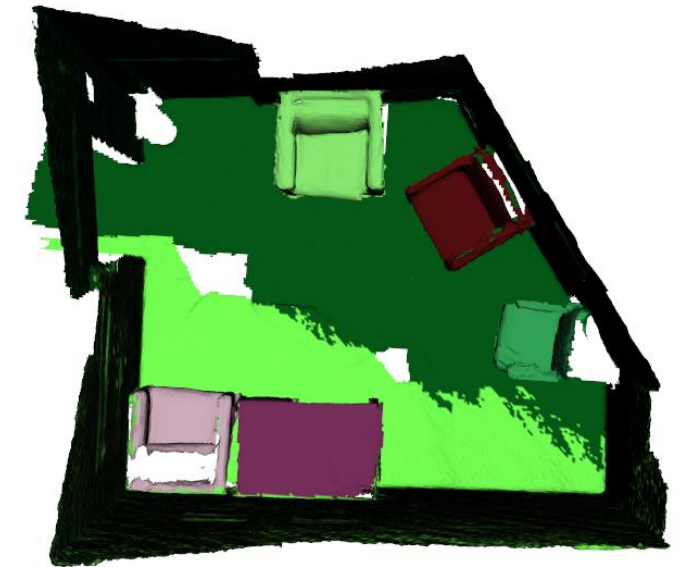
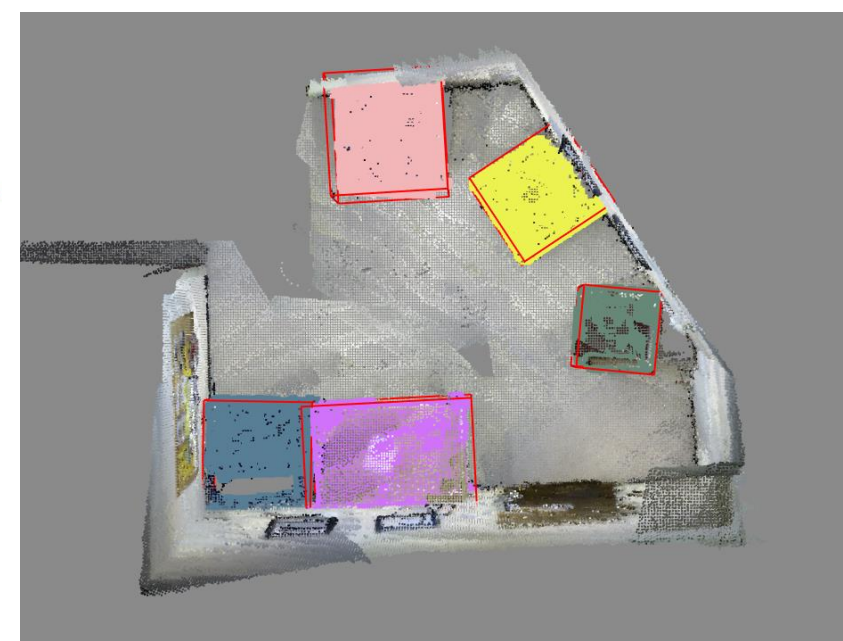
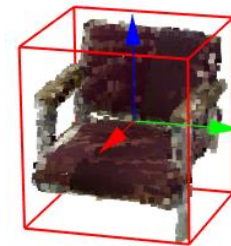
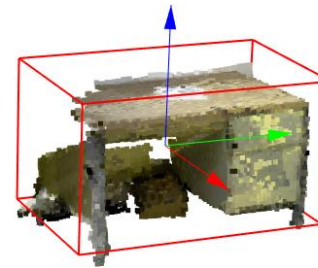
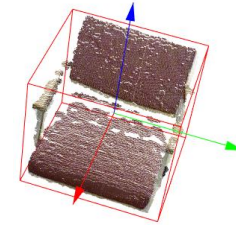
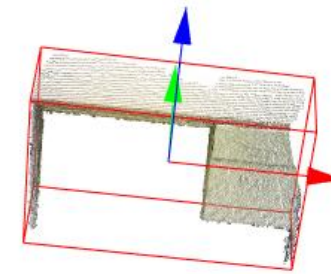
- Indoor dataset: SceneNN dataset
- Logistic dataset: Agiprobot



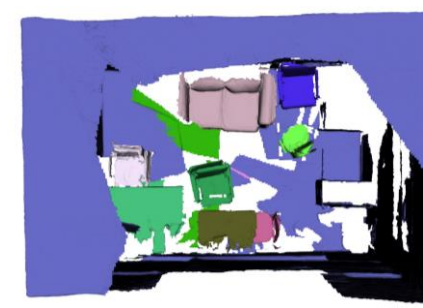
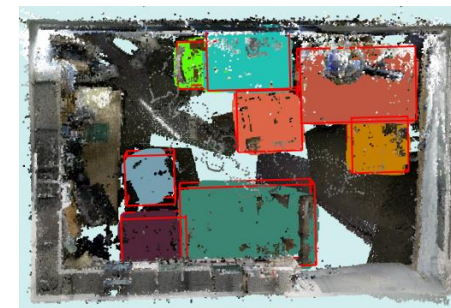
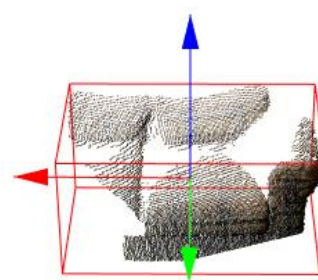
Source: <https://github.com/hkust-vgd/scenenn>

Experiments

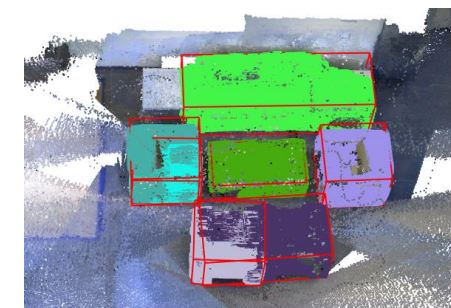
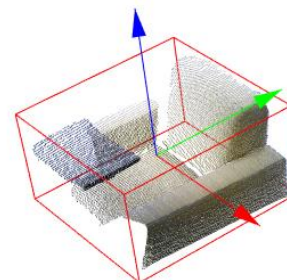
■ Example



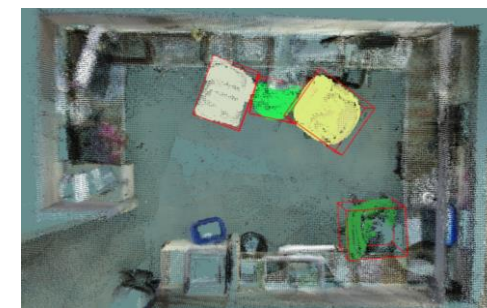
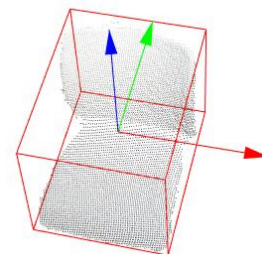
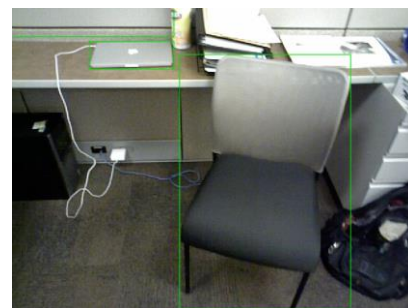
030



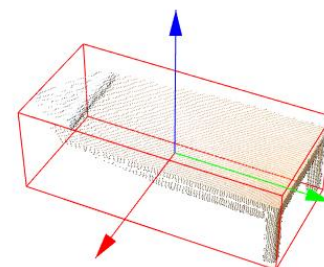
061



086



206



Experiment: SceneNN dataset

Average 3D IoU

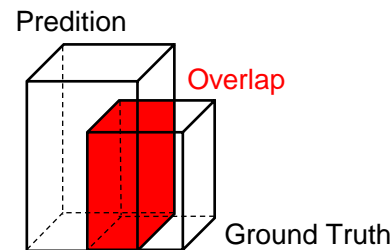
Sequence ID	Bed	Chair	Sofa	Table	Books	refrigerator	Television	Toilet	Bag	Avg.(Ours)
011	-	70.2	70.2	86.3	-	-	-	-	-	78.3
016	51.3	-	71.9	0						41.1
030	-	57.6	80.4	85.7	0	-	-	-	-	57.4
061	-	74.5	62.7	95.1						77.4
078	-	45.9	-	0	0	67.8	-	-	-	13.9
086	-	58.2	-	0	0	-	-	-	53.8	56
096	63.1	60.9	-	0	-	-	32.3	-	0	31.3
206	-	56.5	23.3	65.5	-	-	-	-	29.6	43.7
223	-	63.7	-	69.2	-	-	-	-	-	66.5
255	-	-	-	-	-	55.8	-	-	-	55.8

mAP@0.5

Sequence ID	Bed	Chair	Sofa	Table	Books	refrigerator	Television	Toilet	Bag	Avg.(Ours)
011	-	100	100	100	-	-	-	-	-	100
016	100	-	100	0						66.7
030	-	72	100	66.7	0	-	-	-	-	59.7
061	-	62.5	100	33.3	-	-	-	-	-	65.3
078	-	50	-	0	0	100	-	-	-	37.5
086	-	75	-	0	0	-	-	-	50	31.3
096	100	100	-	0	0	-	0	-	0	33.3
206	-	41	0	40	-	-	-	-	0	20.3
223	-	100	-	50	-	-	-	-	-	75
255	-	-	-	-	-	100	-	-	-	100

3D Intersection over Union (IoU)

$$IoU_{3D} = \frac{V_{overlap}}{V_{gt} + V_{pred} - V_{overlap}}$$



$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{|TP_c|}{|TP_c| + |FP_c|}$$

mAP: mean average precision

TP: True Positive: IoU ≥ 0.5

FP: False Positive: IoU < 0.5

Experiment: SceneNN dataset

runtime performance

Module	Time-CPU (ms)	Time-GPU (ms)
Object Detection	725	32
Object Segmentation	17.85	15
Object Mapping	299	50
Total	1024	102

mAP comparison

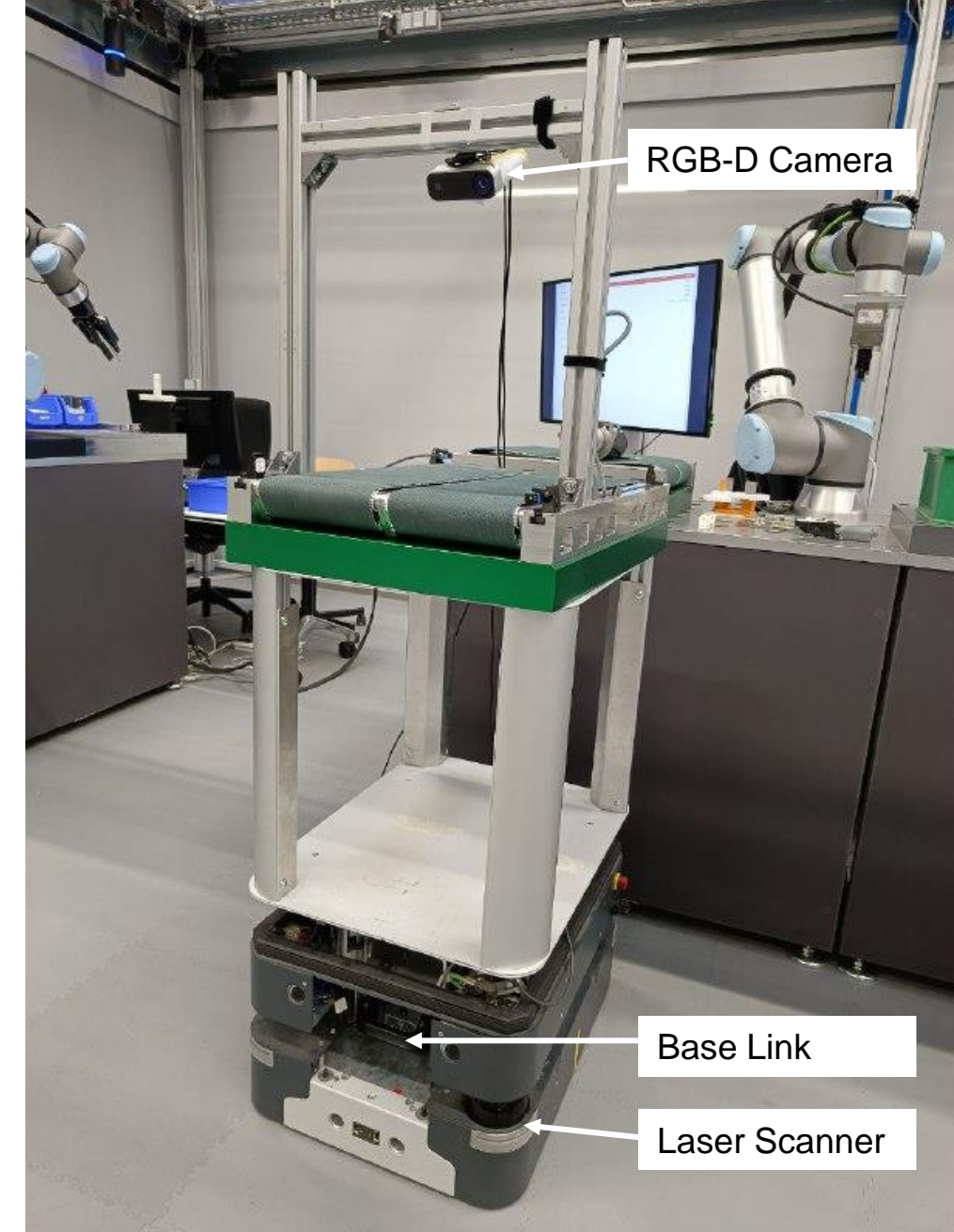
Method	011	016	030	061	078	086	096	206	223	225	Average
Pham et al [1]	52.1	34.2	56.8	59.1	34.9	35.0	26.5	41.7	40.9	48.6	43.0
Grinvald et al [2]	75.0	33.3	56.1	66.7	45.2	20.0	29.2	79.6	43.6	75.0	54.4
Li et al [3]	78.6	25.0	58.6	46.6	69.8	47.2	26.7	78.0	45.8	75.0	55.1
ours	100	66.7	59.7	65.3	37.5	31.3	33.3	20.3	75	100	58.9

runtime comparison

Method	Representation	FPS
Pham et al [1]	Instance-oriented	1 Hz
Grinvald et al [2]	Instance-oriented	1 Hz
Li et al [3]	Instance-oriented	10.8 Hz
Ours-CPU	Instance-oriented	1 Hz
Ours-GPU	Instance-oriented	20 Hz

Experiment: Agiprobot

- Setting:
 - Controller: CPU
 - Camera: Microsoft Azure Kinect
 - Laser scanner: SICK
 - Camera laser calibration:
-
- Scene size: 6x12m
 - Detect objects: conveyor



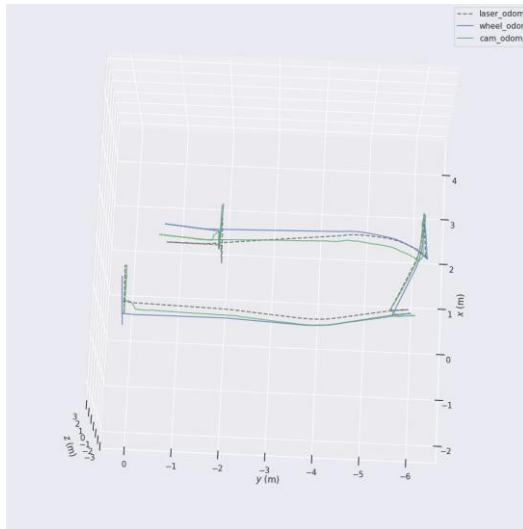
Experiment: Agiprobot

Table: Object IoU

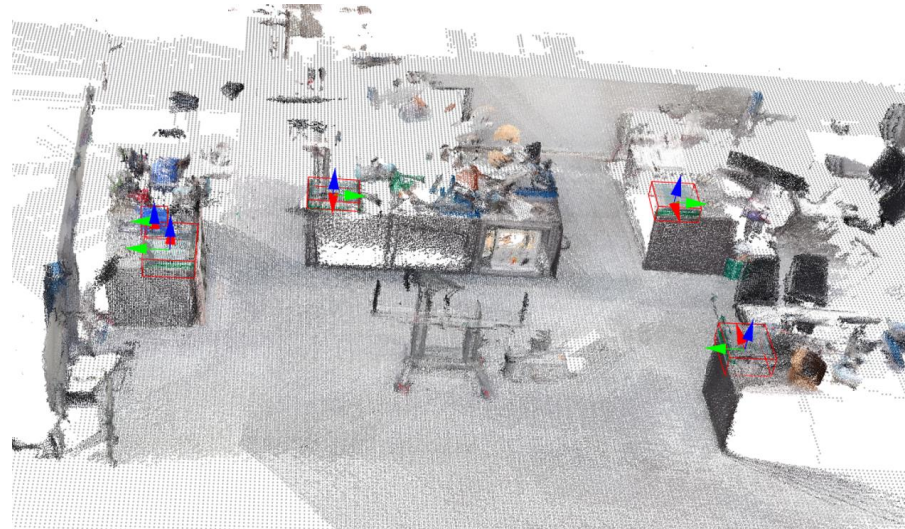
Conveyor	3D IoU	E_Trans(m)	E_rot (°)
1	0.7446	0.058	3.4
2	0.7140	0.060	0.9
3	0.78061	0.029	1.6
4	0.9056	0.035	1.0
Average	0.7925	0.045	1.7

Table: runtime performance

Module	Time-CPU(ms)
Object Detection	725
Object Segmentation	17.85
Object Mapping	299
Total	1024



Camera poses tracking



Point cloud map



Voxel-based map

Conclusion

- What we have done
 - Proposed an efficient object mapping method
 - Evaluate the proposed method on public dataset
 - Evaluate the proposed method on real robotic platform in logistic scene.
- Conclusion
- RQ3: How to efficiently map the environment with semantic object information? (especially in logistic environments)
- we presented an efficient object-level semantic mapping system, which takes RGB-D sequences as input to build a volumetric object-oriented map.
- Experiment on publicly indoor dataset and field test shows that our system has a comparative performance while avoiding high computational cost.
- By employing a fast and stable object detector and TSDF mapping framework, our system can be extended to a CPU-only robotic platform for real-world application.