An Object-oriented Navigation Strategy for Service Robots Leveraging Semantic Information

Akash Chikhalikar, Ankit A. Ravankar, Jose Victorio Salazar Luces, Seyed Amir Tafrishi and Yasuhisa Hirata

***Abstract*— Simultaneous localization and mapping (SLAM) have been an essential requirement for the autonomous operation of mobile robots for over a decade. However, in the wake of recent developments and successes of deep neural networks and machine learning, the conventional task of SLAM is gradually being replaced by Semantic SLAM. Extracting semantic information (such as object information) from sensory data can enable the robot to distinguish different environmental regions beyond the conventional grid assignments of free and occupied. This level of scene awareness is essential for performing higher-level navigation and manipulation tasks and enhancing human-robot interactions. This paper presents an integrated framework that not only builds such maps of indoor environments but also facilitates the execution of '*Go to object*' tasks with high-level user input. We also present a method to extract meaningful endpoints of navigation based on object class. Our modular stack leverages well-known object detectors (YOLOv3), RGB-D SLAM techniques (RTAB-Mapping) and local navigation planners (TEB) to perform ObjectGoal navigation tasks. We also validate the results of experiments in real environments.**

I. Introduction

Simultaneous localization and mapping (SLAM) has been the key technique to realize autonomous operation of mobile robots for over a decade. Several new and efficient algorithms have been proposed for robot based environment perception and modeling [1]. SLAM has widespread applications in different fields ranging from warehouses, drones, self-driving vehicles, agricultural robotics, to space robotics for exploration and mapping of extra-terrestrial regions [2]–[5]. It is also a prerequisite for upcoming Augmented Reality (AR) scenarios.

Most of the earlier works in SLAM focused on utilising the optimal state estimator filters such as the Extended Kalman Filter (EKF-SLAM) [6] for obtaining the robot's position in the operating space. Due to its inability to handle non-Gaussian noises, EKF was eventually replaced by other probabilistic filters particularly particle filtering techniques such as FastSLAM2.0 [7], [8] or Rao-Blackwellized filter for SLAM [9]. This resolved the problems in mapping using grid, however, it was still inefficient for preparing large scale 3D maps as well as for executing higher-level navigation tasks. Most SLAM applications involving mobile robots mainly focused on mapping and localization in small to medium sized areas that are mostly structured [10]. Mapping and localizing in unstructured environments is still a hard problem such as tunnels and mines, and for exploratory applications in underwater and cluttered environments such as forests. Another challenge is for robots operating in dynamic environment [11]. Particularly environment with moving obstacles such as human causes most SLAM algorithms to fail because of the missed correspondence between moving objects resulting in loop closure failures or unreliable map generation.

The authors are with the Graduate School of Engineering, Department of Robotics, Tohoku University, Sendai 980-8579, Japan. (a.k.chikhalikar, j.salazar, s.a.tafrishi, ankit, hirata@srd.mech.tohoku.ac.jp)

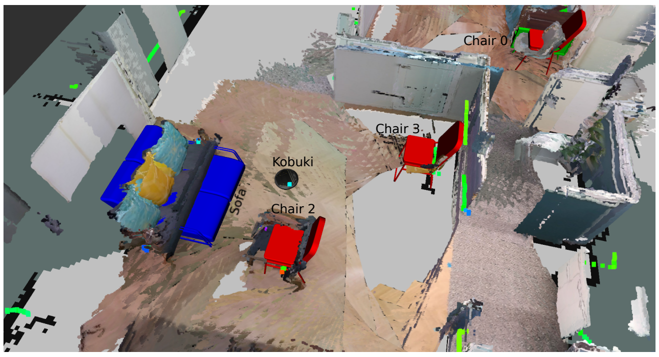


Fig. 1: A partial semantic map of the environment. Markers as STL files are overlaid on the grid map at the location where the object is detected. In this figure, two chairs and a sofa are overlaid by their markers.

On the other hand, humans can seamlessly navigate in any environment. A distinguishing factor is that humans possess *'scene-awareness'* that robots still lack comparatively. For example, a simple task like putting plates in the dishwasher involves understanding *'what is a plate'*, *'what is a dishwasher'*, and *'where is the dishwasher'*. Such reasoning in robot navigation is recently getting much interest. Recent advances in performance of deep learning techniques has enabled real-time object detection [12], [13], instance segmentation [14] and other machine learning based object and place classification methods seems to have answered the first two questions to a large extent. However, Semantic SLAM plays an important role in answering the latter question and is still relatively unsolved. Semantic SLAM is also important to foster human-robot interactions and multi-robot collaboration tasks. A high-level *'scene-awareness'* is critical to deploying service robots in environments like healthcare facilities or restaurants [15], [16]

Earlier approaches in semantic mapping used the 'bag of key points' approach for object detection, and image descriptors such as SIFT for localizing the object in the map [17]. More recently, the Habitat challenge [18], [19] focused on the context-driven exploration of indoor scenes.

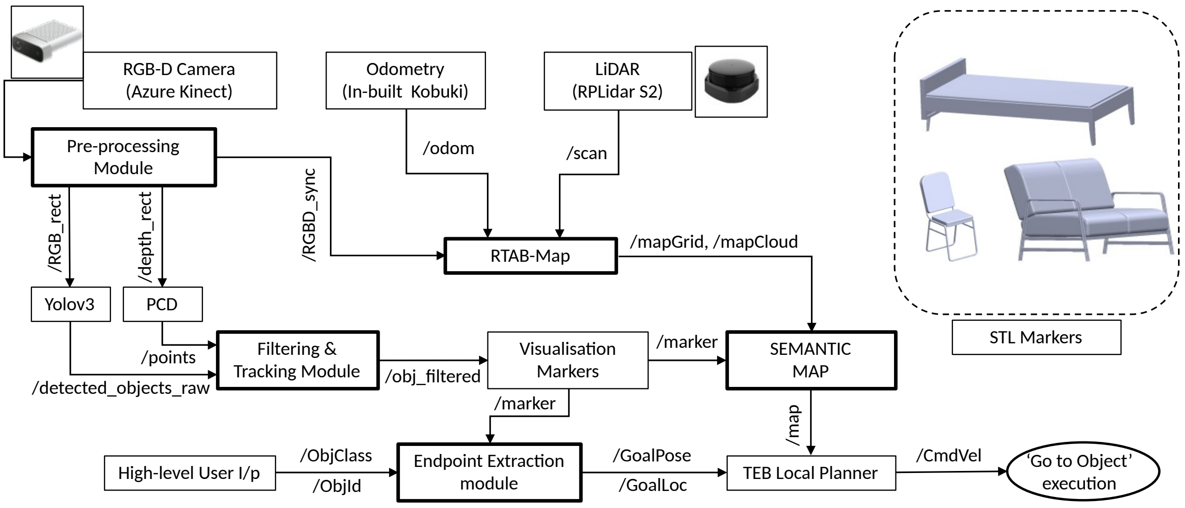


Fig. 2: Overview of proposed framework

One objective of these semantic mapping challenges is 'Go to object' tasks (e.g. 'Go to chair' or 'Go to bed'). While many works targeting the Habitat challenges [20], [21] can understand contextual information in a previously unseen environment and navigate to the desired object, and they are performed in photo-realistic simulators. Martins et al. demonstrated semantic mapping in the real world with off-beat object classes like fire hydrants. However, their approach does not cover 'Go to Object' navigation [22]. Using semantics for path planning is largely an unexplored area of research. Neural networks enriched SLAM algorithms that create pixel-by-pixel encoded semantic maps and give a complete understanding of all the objects and instances in the scene have only recently emerged. However, they do not generate solutions to semantic planning, which is crucial when the robot moves in narrow and constrained spaces inside homes or office-like environments. By considering navigation with semantic understanding, it is possible to segregate spaces into navigable or avoidable regions when planning, e.g. avoiding certain areas of the map when humans are using that region or lowering speed when going over a specific type of flooring or carpets.

This research aims to perform ObjectGoal navigation tasks by providing a framework to navigate to a meaningful endpoint. This endpoint is extracted from a keyword (eg. sofa, chair) input given by the user. We use domain-agnostic modules to ensure that our framework can be effective in different indoor environments. In this work, we leverage well-known object detectors (YOLOv3 [23]), RGB-D SLAM techniques (RTAB-Map [24]) and local navigation planners (TEB [25]). Specifically, we extend existing semantic navigation frameworks to demonstrate the importance of considering object context during navigation.

The rest of the paper is organized as follows. Section II presents our proposed framework with respect to different modules used for the semantic object navigation. Section III describes the hardware configuration, experimental setup, and results of semantic mapping and navigation in indoor test environment. Finally, Section IV concludes the paper with discussion.

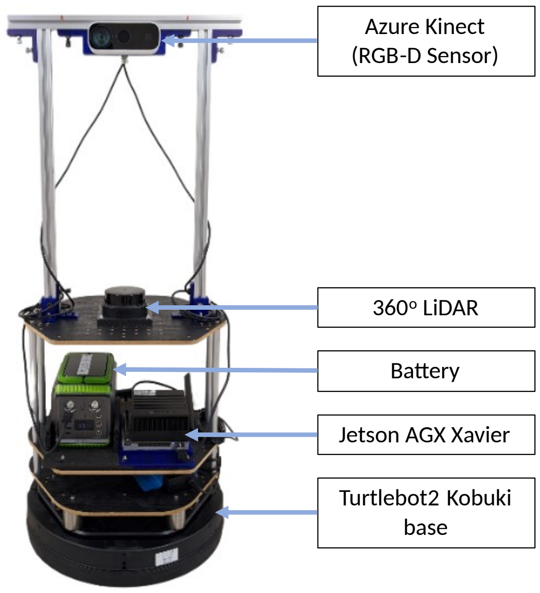


Fig. 3: Hardware configuration of indoor robot (Turtlebot2)

II. System Framework

This section introduces our proposed framework in detail. Firstly, we describe the pre-processing module. This is followed by a discussion on SLAM, filtering and tracking module and describes the steps in preparing the semantic map by overlaying the grid map with object information. Finally, we present our navigation strategy to perform the ObjectGoal navigation. Fig. 2 gives an overview of the proposed framework. The framework is shown as a network of Robot Operating System (ROS) nodes and topics. Fig. 3 shows the hardware configuration for reference. We explain it in detail in Section III

Our framework extensively utilizes the open-source middleware Robot Operating System (ROS) [26] and its associated libraries. For the RGBD sensor, the Azure Kinect DK sensor SDK [27] was used while designing our framework.

A. Pre-processing

The RGB and depth images are acquired from the Azure Kinect sensor at 30 fps. Azure Kinect was chosen as a sensor because of its compatibility with ROS and high resolution camera sensors along with ability to generate high density point cloud. The depth image is registered in the RGB frame since the RGB image has a lower FoV. ROS distributed network is used and the compressed depth and RGB images are transported from the onboard computing system to the master computer. These images are rectified using intrinsic camera calibration parameters provided by the SDK, and finally, a dense point cloud is generated.

B. Mapping

A 3D grid map (octomap) of the environment is first processed from the generated point cloud data. While any kind of 3D map representation can be used for navigation such as the ORB-SLAMv2 [28] and RGBDSLAMv2 [29], we choose RTAB-Map [24] as the preferred method because it is well documented and has good support for different distributions and operating system platforms including embedded systems. Additionally, it uses a graph-based SLAM approach that uses appearance based loop closure in real-time. It uses GTSAM [30] as its graph optimization strategy due to the latter's robustness to multi-session mapping. It also provides the ability to store robot poses as graph nodes that can be used for offline map generation and data post-processing. While mapping, each of the 3D Local occupancy grid is transformed using the robot's pose. If the grid corresponds to a new node, it is appended to the current global map with clearing and adding of obstacles assuming the new node as the truth. However, if the local grid corresponds to a previously visited place using the image correspondence and image feature matching, a loop closure is performed in the back-end allowing the assembly of sub-maps to a globally consistent and updated map.

Three of the most ubiquitous object classes in indoor spaces, namely the ***chair, bed***, and ***sofa*** are taken into consideration for generating the semantic map. During the mapping process, YOLOv3 [23] is running in parallel to detect objects in frames. We choose YOLOv3 due to its low inference times and mean Average Precision (mAP) scores. For each object detected, a 5-D output is generated by the YOLOv3 network, namely, object class and 4 Bounding Box (BB) parameters (centre coordinates: X_c and Y_c, width: W, and height: H). Artificial cropping of roughly 10% of the image width is done before using the BBs to ensure that the object is sufficiently centred and not partially visible. We then randomly sample a pre-determined number of pixels from the detected Bounding Box(15 pixels for bed and 25 pixels for chair, and sofa) and compute the mean X and Y coordinate from the Point Cloud data corresponding to those pixels. It is assumed that this mean equals the centroid of the detected object. These centroid coordinates which are in the camera coordinate frame, are then converted to the global map frame. This is the first of six such measurements carried out before placing an STL marker (3D object model) corresponding to the object class on the map frame. A Kalman filter determines the marker position from the six measurements and the details of which are given in equations 2 and 3. Due to the lack of a reliable, real-time, low computation cost object-pose estimation network, we don't compute the 3D Pose of the object in this work. There are several techniques that can be utilized to correctly estimated the final 3D pose of the detected [31], [32].

After the object is placed on the map as an STL marker object, it still needs to be tracked. If an object is re-observed while the robot is exploring, it is important to determine whether it is a previously detected instance of that object or not. This is not a trivial task as the robot can be at a completely different location and viewing the object from a completely different viewpoint. For tracking the objects in such instances, we firstly maintain a dictionary of $'K'$ previously seen objects of the same class as $\mathbf{P} = \{P_0, P_1, \cdots, P_K\}$. Each element in the dictionary is a 2×1 vector corresponding to x and y locations on the map. If for any acquired frame, $'L'$ objects of the same class are observed, a new list $\mathbf{C} = \{C_0, C_1, \cdots, C_L\}$ is created. We then calculate a cost-association matrix $\mathbf{D}_{k,l}$ between both the sets using Euclidean distance as shown in equation 1 below.

$$\forall(k,l) \in (K,L): \ \mathbf{D}_{k,l} = \sqrt{(\mathbf{P}_k - \mathbf{C}_l)^T(\mathbf{P}_k - \mathbf{C}_l)} \quad (1)$$

After the cost-association matrix is computed, the association between new observations (i.e list $\mathbf{C}$) and previous observations (i.e dictionary $\mathbf{P}$) is determined using the Hungarian algorithm [33] as prescribed in [22]. If the distance corresponding to the association is less than the threshold value α ($\mathbf{D}_{k,l} < \alpha$), it is assumed to correspond to the previously seen object with which it is associated to. Otherwise, it is considered a new instance. The object marker is placed on the map for this new instance, and its location is appended to the dictionary. We conducted numerous experiments by varying the threshold value α and set α as 0.5m for the chair and sofa, and 1m for the bed.

If the object is determined as previously seen, a Kalman filter is then used to combine current observations with prior observations of the same instance over time. For a prior state of the instance 'k' given by $\mathbf{P}_k^{t-1}$ and an associated new observation $\mathbf{C}_l^t$, determined via the Hungarian algorithm, the following computations are carried out:

$$\text{Prediction Step}: \mathbf{P}_k^{\prime t} = \mathbf{A}\mathbf{P}_k^{t-1} \quad (2)$$

$$\text{Correction Step}: \mathbf{P}_k^{t} = \mathbf{P}_k^{\prime t} + \mathbf{K}(t)(\mathbf{C}_l^t - \mathbf{H}\mathbf{P}_k^{\prime t}) \quad (3)$$

where, $\mathbf{A}$ is the 2×2 state transition matrix (set to identity), $\mathbf{H}$ is the measurement matrix (set to identity), and $\mathbf{K}(t)$ is the Kalman gain. The optimal Kalman gain, $\mathbf{K}(t)$, is computed with a diagonal process noise covariance matrix of $0.3\mathbf{I}$ and

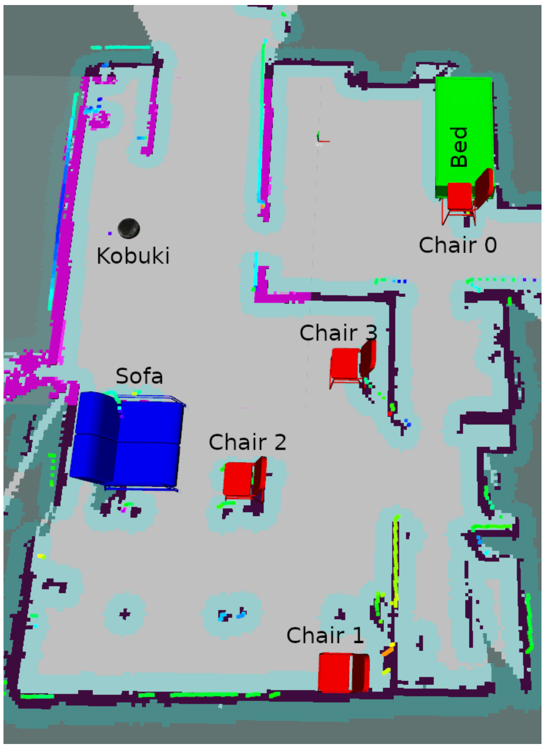


Fig. 4: Semantic layered map of the environment with detected object markers.

a diagonal measurement noise covariance matrix of $0.5\mathbf{I}$. This ensures consistency in object localization as the object classes are relatively static. Fig. 4 shows a semantic map of the environment prepared with this method.

C. Semantic Navigation

After completion of the mapping phase, the navigation module is executed. It is important to note that mapping is a continuous process. The tracking module will continue to incorporate small changes in the location of the objects even during the navigation phase and update the map. Additionally, if a new instance of any object class is discovered during navigation, it is added to the map. This procedure is the same as described in Sec. II-B. This is possible since we maintain a dictionary of previously observed instances of all classes. Any detection during navigation is compared with the dictionary instance to ratify whether it is a new or previously seen instance.

The user input for the object class to navigate is to be provided first. If there are multiple instances of the object class on the map, the instance ID of the object needs to be specified as well. Fig. 4 shows multiple instances of object class ***chair*** in our environment. The instance ID is set according to the order in which the robot discovered the instances (i.e. first chair detected has instance ID = 0, the second chair has one and so on).

Context-driven navigation is used to determine the end-point. In the case of the object class being bed, the logical approach for the robot to approach the object is along the edge of the bed. An example of such a scenario is in a healthcare facility wherein the robot has to deliver medicines to a bed-ridden patient. If the object is sofa or chair the logical approach is to approach from the front. This is meaningful in residential scenarios wherein the user might be relaxing on the sofa and needs an item delivered.

Algorithm 1 Endpoint extraction from user input

```
Require: $Map$, $Markers$, $Teb\_planner$
  Input: $Object$, $Id$, $Goal\_pose$, $Goal\_loc$
  While $Not\_shutdown$ do
      if $Prev\_Goal = Reached$
          $Object \leftarrow User\_Object\_input$
          if $Instances > 1$
              $Id \leftarrow Obj\_Id\_input$
          $Obj\_pose \leftarrow Markers.pose$
          if $Object == Chair\ or\ Sofa$
              $Goal\_pose \leftarrow Rotate\_Obj\_pose\_by\_180^o$
              $Goal\_loc \leftarrow Markers.loc + 0.5 * Obj\_pose$
          if $Goal == Bed$
              $Goal\_pose \leftarrow Rotate\_Obj\_pose\_by\_90^o$
              $Goal\_loc \leftarrow Markers.loc - 0.5 * Goal\_pose$
          $Map \leftarrow Goal\_pose, Goal\_loc$
          $Prev\_Goal = Not\_Reached$
      While $Prev\_Goal = Not\_Reached$ do
          $Run\ Teb\_planner$
      end While
      $Prev\_Goal = Reached$
  end While
```

In both cases, the robot's final pose is desired to be facing toward the object to improve human-robot interaction while maintaining the convenience to the user. The implementation of this algorithm is shown in Pseudocode 1.

III. Experimental Studies

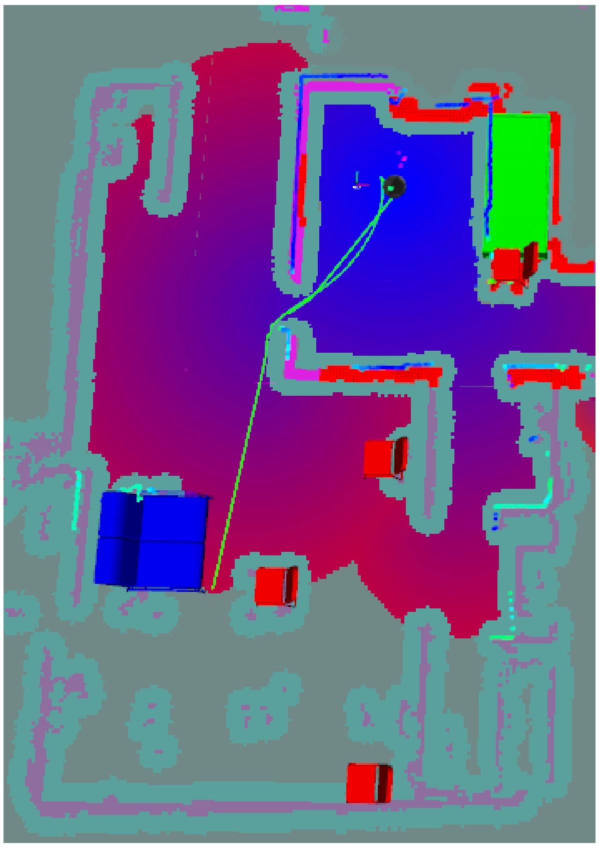
In this section, we first describe the hardware configuration of the robot. Then, we discuss the experiments and results.

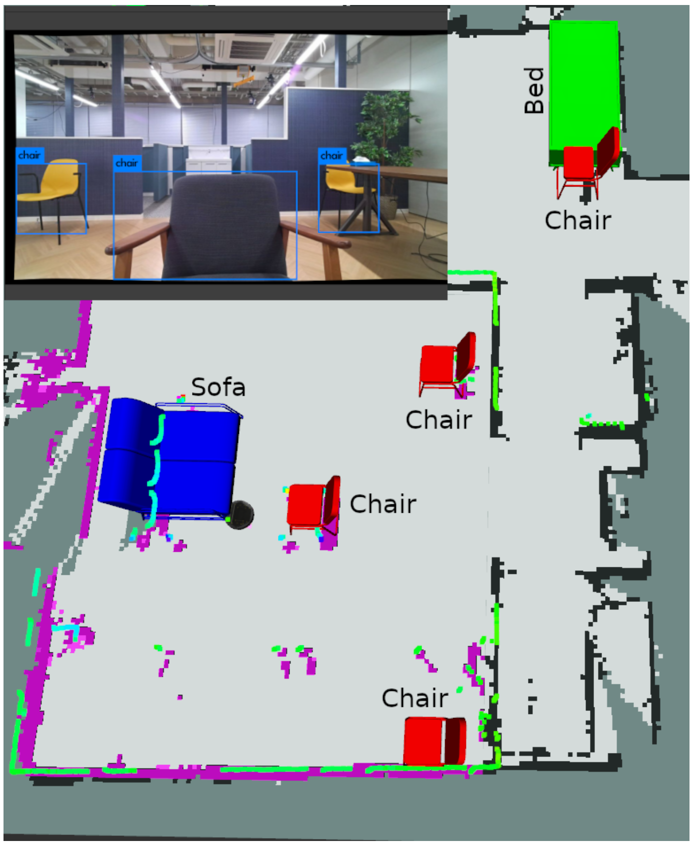
A. Hardware Configuration

As seen in Fig. 3, a modified Turtlebot2 platform with Kobuki base was used to perform the experiments. The Kobuki comes with onboard encoder to calculate the odometry. Additionally, it was fitted with an Azure-Kinect RGB-D camera and an RPLIDAR S2 laser range scanner. For computation, a ROS distributed computing network was setup consisting of NVIDIA Jetson AGX Xavier as an onboard computer mainly for data acquisition and controlling the robot.

The semantic information was generated on a CPU (master) with NVIDIA 3090RTX graphics' unit and i9-12900K processor. The framework was able to sustain implementation at rates higher than 30 fps.

The depth cloud was limited to a range of 3m for reducing the noise generated at higher depth and the scanning LiDAR data to a range of 8m. The field of interest of the rectified RGB-D images was $120^\circ \times 120^\circ$.On the object detection side, a threshold of 0.85 was set for YOLOv3 [23] framework. The


(a)



(b)

Fig. 5: Result of Navigate to chair task (a) Intermediate step with local and global costmaps (b) End-result of semantic navigation

angular tolerance for navigation was 0.25 and the positional tolerance was 0.1m. The maximum forward and reverse movement velocity was 0.55 m/s and 0.2 m/s, respectively. The angular velocity was limited to 0.3 m/s.

B. Mapping and Navigation

The robot was initially teleoperated to obtain the semantic map of the environment by moving inside the area of interest. During the exploration process, minor changes in the object's centroid's coordinates are managed by the filtering and tracking module. After the semantic map is ready, the navigation module is started. For the navigation task, either the user can input the object and the instance ID (if required) to which location the robot is supposed to navigate on the console, or the robot can be given position-based goals. The object extraction algorithm, as described in the Pseudocode 1 takes the input and accordingly publishes a meaningful endpoint position and orientation on the map frame. This activates the move base local planner (TEB), which then prepares a plan based on the endpoint of navigation and the global and local costmaps. The robot is directed to approach the object, maintaining a safe distance of 0.5m to the object in all cases. The intermediate and end result of navigation can be found in Fig. 5a and 5b, respectively. Fig. 5b also shows YOLOv3 output during the navigation. Once the robot is facing the front of the chair, and within the terminal distance, the task of semantic navigation is successfully completed. After the termination criteria are met, the user can provide the following input, and the navigation sequence is repeated. In future, we will extend the semantic navigation node so it can interface with different services like Siri or Alexa [16].

IV. Conclusions and Discussions

This study provided a proof of the concept of semantic navigation in indoor environments. The objective of approaching the user-defined location in a meaningful way was successfully demonstrated. The filtering and tracking module was able to incorporate small changes in the detected location of the object due to differences in depth camera data when the object was viewed from different view points and locations. The end-point extraction algorithm is able to direct the robot to approach the object in a meaningful manner. Our proposed system can currently work in real-time and can perform object-oriented semantic goal tasks in small to medium sized indoor environments such as homes and offices. The proposed semantic navigation framework is flexible and can be easily integrated into existing ROS navigation stack as an additional layer for high-level scene understanding. We are also planning to add additional modularities to our current framework to include high definition 3D maps along with semantic rich information of the environment for performing intelligent tasks. The system can be improved by using instance segmentation techniques for clustering Point Cloud Data instead of vanilla object detection. The framework can be readily incorporated with reliable, real-time object-pose estimation networks to further improve its robustness in complex environments. There is a lot of potential to increase the applicability of the work by adding manipulation tasks using the generated semantic map. This preliminary research showed the necessity of considering the context not only during mapping but also in navigation. Such context-driven navigation is of additional importance in the development of service robots.

Acknowledgment

This work was partially supported by JST Moonshot R&D [Grant Number JPMJMS2034], JSPS Kakenhi [Grant Number JP21K14115], and JST SPRING [Grant Number JPMJSP2114].

References

[1] 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: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] L. Kunze, N. Hawes, T. Duckett, M. Hanheide, and T. Krajník, "Artificial intelligence for long-term robot autonomy: A survey," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4023–4030, 2018.

[3] A. Ravankar, A. A. Ravankar, A. Rawankar, and Y. Hoshino, "Autonomous and safe navigation of mobile robots in vineyard with smooth collision avoidance," *Agriculture*, vol. 11, no. 10, p. 954, 2021.

[4] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[5] E. Narváez, A. A. Ravankar, A. Ravankar, T. Emaru, and Y. Kobayashi, "Autonomous vtol-uav docking system for heterogeneous multirobot team," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–18, 2020.

[6] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[7] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, "Fastslam 2.0 : An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," *Int. Joint Conf. on Artificial Intelligence, 2003*, pp. 1151–1156, 2003.

[8] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[9] G. Grisetti, C. Stachniss, and B. Wolfram, "Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters," *Transactions on Robotics*, vol. 23, no. 1, pp. 34–45, 2007.

[10] A. A. Ravankar, A. Ravankar, T. Emaru, and Y. Kobayashi, "A hybrid topological mapping and navigation method for large area robot mapping," in *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1104–1107, IEEE, 2017.

[11] H. Beomsoo, A. A. Ravankar, and T. Emaru, "Mobile robot navigation based on deep reinforcement learning with 2d-lidar sensor using stochastic approach," in *2021 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*, pp. 417–422, IEEE, 2021.

[12] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[15] A. A. Ravankar, A. Ravankar, C.-C. Peng, Y. Kobayashi, and T. Emaru, "Task coordination for multiple mobile robots considering semantic and topological information," in *2018 IEEE International Conference on Applied System Invention (ICASI)*, pp. 1088–1091, IEEE, 2018.

[16] A. A. Ravankar, S. A. Tafrishi, J. V. S. Luces, F. Seto, and Y. Hirata, "Care: Cooperation of ai-robot enablers to create a vibrant society," *IEEE Robotics and Automation Magazine*, 2023.

[17] I. Jebari, S. Bazeille, E. Battesti, H. Tekaya, M. Klein, A. Tapus, D. Filliat, C. Meyer, S.-H. Sio-Hoï Ieng, R. Benosman, E. Cizeron, J.-C. Mamanna, and B. Pothier, "Multi-sensor semantic mapping and exploration of indoor environments," in *2011 IEEE Conference on Technologies for Practical Robot Applications*, pp. 151–156, 2011.

[18] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans, "ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects," in *arXiv:2006.13171*, 2020.

[19] K. Yadav, S. K. Ramakrishnan, J. Turner, A. Gokaslan, O. Maksymets, R. Jain, R. Ramrakhya, A. X. Chang, A. Clegg, M. Savva, E. Undersander, D. S. Chaplot, and D. Batra, "Habitat challenge 2022." `https://aihabitat.org/challenge/2022/`, 2022.

[20] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 4247–4258, Curran Associates, Inc., 2020.

[21] J. Ye, D. Batra, A. Das, and E. Wijmans, "Auxiliary tasks and exploration enable objectgoal navigation," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16097–16106, 2021.

[22] R. Martins, D. Bersan, M. Campos, and et al., "Extending maps with semantic and contextual object information for robot navigation: a learning-based framework using visual and depth cues," *Journal of Intelligent and Robotic Systems*, vol. 99, p. 555–569, 2020.

[23] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[24] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of field robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[25] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[26] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, Japan, 2009.

[27] "Azure Kinect DK Sensor SDK." `https://docs.microsoft.com/en-us/azure/kinect-dk/sensor-sdk-download`. Accessed: 2022-08-13.

[28] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[29] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, 2014.

[30] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," tech. rep., Georgia Institute of Technology, 2012.

[31] M. Simon, K. Amende, A. Kraus, J. Honer, T. Samann, H. Kaulbersch, S. Milz, and H. Michael Gross, "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[32] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou, "OnePose: One-shot object pose estimation without CAD models," *CVPR*, 2022.

[33] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.