Semantic Mapping and Reasoning

A Thesis Presented to the Academic Faculty

Robot Autonomy and Interactive Learning (RAIL) Laboratory
Institute for Robotics and Intelligent Machines (IRIM)
School of Interactive Computing

Kevin Chen
Fall 2019

Faculty Mentor: Dr. Sonia Chernova
2nd Thesis Reader: Dr. Harish Ravichandar

Table of Contents

Summary

Rich, yet efficient knowledge processing is one of the key problems in modern autonomous robotics. The Robot Autonomy and Interactive Learning (RAIL) Lab at the Georgia Institute of Technology has developed a new knowledge processing framework named Robot Common Sense Embedding (RoboCSE), which leverages multi-relational embeddings to learn object affordances, locations, and materials. This project aims to test the capabilities of RoboCSE for household robots by building a perception pipeline, which outputs a semantic map (i.e. map with object labels). The perception pipeline consists of two main components: a Simultaneous Localization and Mapping (SLAM) algorithm to build an occupancy map and two object classifiers to label objects in the map. We hope to integrate the semantic map into RoboCSE to test RoboCSE's ability to perform high-level task planning and knowledge sharing.

Introduction

Common tasks, such as pouring ourselves a cup of coffee, are trivial to us human beings. We do not have to consciously reason and plan each step required to accomplish such tasks. However, for Artificial Intelligence (AI) and robots, such reasoning is not trivial. Even with an accurate representation of the world, robots find it difficult to learn, understand, and utilize new knowledge to accomplish these "simple" tasks. Rich, yet efficient knowledge processing is one of the forefront problems of robotics research, and several works have attempted to delve into this topic with varying degrees of success [1], [2], [3], [4], [5], [6].

One existing approach for knowledge processing is to use Bayesian Logic Networks [1], as seen in [2]. While Bayesian Logic Networks offer precise probability-based predictions, they are not able to scale well to large knowledge graphs [1], [2]. Another solution for knowledge processing is to implement hard-coded rules for robots to learn new concepts and solve tasks in a sensor-equipped environment [3]. Unfortunately, the rules used by the robots to learn new concepts are unable to adapt to uncertainty in the real complex world, which impedes the ability for robots to react to unanticipated scenarios [3]. A third solution to knowledge processing involves training a model on over a thousand manipulation demonstrations across hundreds of different objects and parts [4]. However, this solution is slow to explore new concepts and techniques as it requires a large amount of training data.

A newer method for knowledge processing is inference based on object affordances [5]. This approach relies on using a knowledge base of objects to understand the objects' properties and affordances (i.e. methods of use), from which robots can infer how to accomplish new tasks. The Robot Autonomy and Interactive Learning (RAIL) lab has recently devised a new powerful

knowledge framework named Robot Common Sense Embedding (RoboCSE) for robots to learn and understand such object affordances in their environment [6]. This framework uses semantic reasoning (i.e. conceptual and relational-based reasoning) to offer robots the ability to adapt to new situations, generalize new knowledge, and scale their knowledge to real-world problem sizes. Other existing knowledge frameworks have yet to provide a scalable solution while offering sufficiently rich inference and learning capabilities.

This project aims to explore the ways that RoboCSE can be used to improve the performance of household robots. The robotics platform we have chosen to use for this project is the Fetch robot (pictured right). To test RoboCSE in the environment of the Fetch robot, this project aims to build and connect RoboCSE to a perception pipeline, which is a set of software modules that will enable the robot to sense its environment. The perception pipeline consists of two main components: a Simultaneous Localization and Mapping (SLAM) algorithm and a set of two object classifiers. To build the map, we use an existing open-source SLAM package named *Octomap* and an externally mounted 3D Light Detection and Ranging (LiDAR) sensor [7]. Afterwards, we utilize two object classifiers, PointNet2 [8] and MobileNetV2 [9], to perform semantic segmentation, which is the task of associating each pixel in the map with an object class. The object classifiers use the 3D LiDAR and Red Green Blue-Depth (RGB-D) camera sensors. We plan to integrate this perception pipeline into RoboCSE for the Fetch robot

Figure 1: The Fetch robot is a household robot developed by the manufacturer Fetch Robotics. It is the robotic platform we have used for this project.

to learn and generate semantic representations of its world.

With the semantic representation of the environment, we have two routes we would like to take this project. First, we would like to see how effective RoboCSE is for high-level task planning. An example of such a task is pouring a cup of coffee. There are many hidden constraints and implied tasks involved with this problem. For example, the robot needs to realize that a cup and a coffee maker would most likely be found in the kitchen, and inside the kitchen, the cup is most likely to be found inside a cabinet, cupboard, or dishwasher.

Second, we hope to use RoboCSE to develop a distributed semantic model of the environments for multiple robots. Imagine two robots learning about two different users in two different environments. We could simply have both robots independently learn about their users in their own domain, learning everything from scratch. However, it would more likely prove to be beneficial to have the robots eventually share what they have learned, as there are often some similarities between humans and the robots' environments. Hence, we hope to use this knowledge framework for robots to be able to share knowledge and concepts with each other across multiple domains, which would supplement the knowledge of other robots and allow an even wider range of inferences to be drawn.

Literature Review

Numerous projects within the AI community have sought to develop a reasoning framework to process raw knowledge, both for robots and for other contexts [1], [2], [3], [4], [5], [6]. While general knowledge processing and reasoning is a well-established research field, there has been much less work done to design knowledge reasoning frameworks for the context of robotics control and planning [5]. Autonomous robots require such frameworks in order to

generalize knowledge for new situations. Three prior techniques for knowledge reasoning frameworks include Bayesian Logic Networks [1] used in [2], action models and embedded classifiers in KnowRob [3], and learning from demonstrations in Robobarista [4].

One of the classical approaches for knowledge reasoning is the use of Bayesian Logic Networks [1]. While Bayesian Logic Networks offer precise probabilistic inference and learning, this technique does not scale well to large knowledge graphs [1], [2]. KnowRob tries to overcome some of the scalability issues of Bayesian Logic Networks. KnowRob begins with hard-coded encyclopedic knowledge of certain concepts and specific instances (or concrete examples) of those concepts [3]. Afterwards, it uses an action model and embedded classifier to learn new concepts and new instances of those learned concepts [3]. However, a limitation of this approach is that the action model is fixed (i.e. manually set) as opposed to being learned from the framework itself. Hence, KnowRob is unable to account for the wide complexities and uncertainties of the real world, which impedes the robot's ability to truly adapt to new situations. Robobarista tries to use learning from demonstrations to improve the concepts it learns. Robobarista aggregates manipulation demonstrations that are crowd-sourced from the custom-built web platform [4]. However, Robobarista is slow to explore concepts and techniques as it requires a large amount of training data.

A new framework for knowledge reasoning, RoboCSE, was developed in the RAIL lab to achieve a balance between strong learning capabilities and scalability [5], [6]. RoboCSE leverages multi-relational embeddings to provide a framework that can reason about objects, affordances, locations, and materials, while being scalable to larger problems [6]. One of the most powerful features that RoboCSE offers is the inference of unseen triples, or "triple generalization" [6]. For example, given that a mug is similar to a cup and that cups are fillable,

could a mug be fillable? This framework can answer such questions with a degree of confidence. Based on the test results from several specific datasets, RoboCSE provides better inference capabilities than pre-trained embeddings such as Word2Vec, while using orders of magnitude less memory than the classical approach of a Bayesian Logic Network [1], [6]. RoboCSE's efficient use of memory offers great potential for it to be used in real-time on robotic systems, which usually have limited computational resources.

In this project, we propose to connect RoboCSE with a perception pipeline consisting of an existing SLAM algorithm and a neural network to test the framework's capabilities that are grounded in a robot's real-world scenario. While many SLAM solutions exist in the robot community, many of these solutions have to sacrifice either computational accuracy or efficiency [10]. For example, there exist several offline SLAM packages that offer optimal guesses for all robot states and landmark positions, such as the GraphSLAM algorithm [11]. However, these offline packages often require an extreme amount of computational resources, which is not suitable for performing use cases that we have in mind for the future, such as real-time task planning. We require a SLAM package that is more efficient.

We have considered using the Google *Cartographer* SLAM package for this project [12]. The Google *Cartographer* provides a better balance between accuracy and efficiency, especially for mapping an indoor environment. The *Cartographer* uses a branch-and-bound approach to efficiently compute any loop closures in the robot's path, making it more feasible to run in real-time [12]. The *Cartographer* also has support for integration into Robot Operating System (ROS) projects and for Fetch Robots. However, in practice, the *Cartographer* turns out to be difficult to use on the Fetch robot for several reasons. First, the package required a complicated parameter-tuning procedure that often could not reliably produce high-quality maps, especially

for a wide range of different indoor environments. Second, the package was not designed well to integrate with the other components in our perception pipeline, such as the semantic labelling and RoboCSE knowledge processing framework.

We have had more success with the Octomap framework [7]. This framework uses probabilistic occupancy estimation to model the certainty of each 3D point in space being occupied by any object [7]. The ability to model the occupancy of each point probabilistically helps to develop a solution that is robust to any errors or outliers in the sensor data. Furthermore, this framework uses a novel octree representation of the state space in order to be both memory and space efficient, making this framework suitable for our future use cases of high-level task planning and sharing learned concepts. Because of advantages of using Octomap, we have chosen to integrate Octomap into our perception pipeline.

To perform semantic segmentation on the map generated from Octomap, we have incorporated two object classifiers: PointNet2 and MobileNetV2. PointNet2 is a Deep Learning algorithm based on Deep Convolutional Neural Networks (CNN) that performs 3D semantic segmentation using Point Cloud data [8]. MobileNetV2 is an image classifier that aims to improve the performance of mobile models for efficient object classification [9]. We use the combination of PointNet2 and MobileNetV2 to perform semantic segmentations using the 3D LiDAR sensor and the robot camera sensor, respectively.

The Octomap SLAM algorithm and the two object classifiers create the perception pipeline. Together, the components output a semantic map, or a map with object labels. We hope to use this semantic map to test the capabilities of RoboCSE in a robot's environment with respect to high-level task planning and knowledge sharing between robots.

Methodology

We have deployed the perception pipeline on a Fetch robot. The Fetch robot comes with the ability to drive around an indoor environment and sense the world with a 2D laser on its base and 3D RGB-D camera on its head. Both of these sensors point in the forward direction of the robot. Since the Fetch robot can only sense in the forward direction, we have integrated a 3D Velodyne VLP-16 Laser (LiDAR) sensor, which allows the robot to perceive the environment with a full 360-degree field of view, and 15 degrees up or down from the horizontal. Compared to the sensors that come with the Fetch robot, the 3D Velodyne sensor provides an efficient ability to build a map of the environment.

We have chosen to run the perception pipeline in an environment where there are not a lot of dynamic obstacles (i.e. obstacles that may move while the robot builds the map) as a means of simplification. Hence, we have chosen to run the pipeline in our research lab (the RAIL lab) located at the Georgia Institute of Technology, where the only dynamic obstacle is a few people that occasionally walk into/out of the lab. Once we achieve better results, we plan to expand to the entire floor or potentially map an entire building. The risk of encountering more and more dynamic obstacles increases as we choose to run the perception pipeline in a larger area, so we have begun with a smaller area first and have stayed in indoor environments.

The perception pipeline consists of two main components: the map builder using a SLAM algorithm, and two object classifiers to perform semantic segmentation. Figure 2 below is a software architecture diagram of the perception pipeline, which provides a high-level overview of how the components function together:

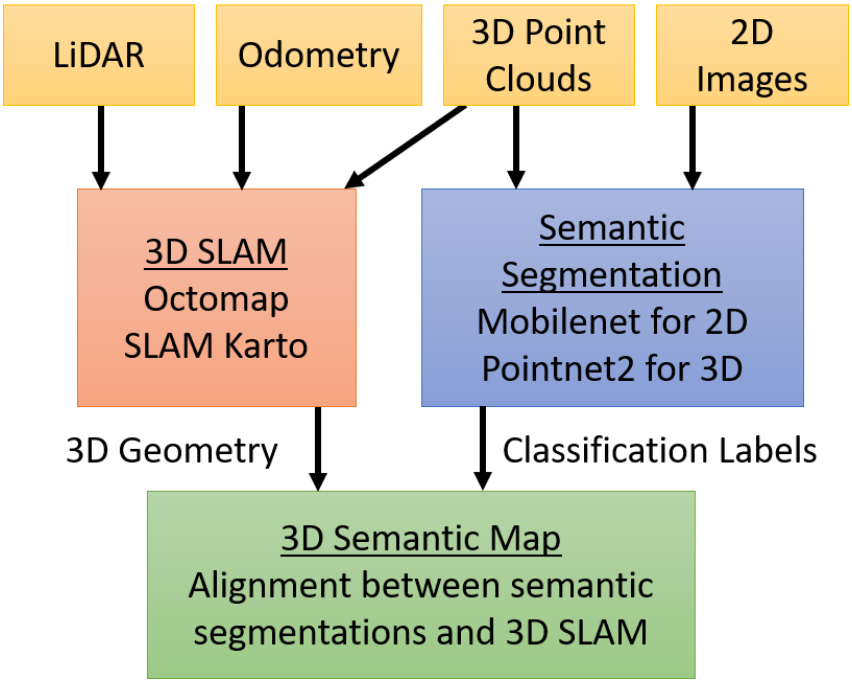

Figure 2: Software architecture of the perception pipeline. Sensor data (shown in the top row) feeds into the two main components of the perception pipeline: 3D SLAM and semantic segmentation. The results of the perception pipeline are combined together to form the semantic map.

For the SLAM algorithm, we have utilized Octomap, as it integrates easily with our sensors and robotics platform [7]. It accepts PointCloud2 messages returned by the 3D LiDAR sensor and data from the odometry sensor, which measures how far the robot has travelled based on wheel rotations. Using the data from these two sensors, Octomap builds a map of the environment over time. The map returned by Octomap simply contains whether a point in space is occupied by an object (i.e. the occupancy of a point), as the PointCloud2 messages returned by the LiDAR only contains occupancy data. In order to leverage knowledge of the objects inside the map, we have integrated two object classifiers.

For the object classifiers, we have used the PointNet2 [8] and MobileNetV2 [9] classifier models. PointNet2 takes in data from the 3D PointCloud2 messages returned by the 3D LiDAR

sensor and performs 3D semantic segmentations [8], while MobileNetV2 takes in 2D images from the camera and performs 2D classification [9]. The outputs of these two models are fused together by selecting the output from the model that was more confident for each specific pixel. These models were trained using a dataset of common indoor objects, such as tables and chairs. We chose to label each object at the pixel level (i.e. for each pixel in space, we store what object class it belongs to), which enables us to represent each object as precisely as possible. This is an important tradeoff: labelling objects at the pixel level is more precise than drawing polygonal regions around each object since drawing polygonal regions will contain small areas that are not part of the object. However, labelling objects at the pixel level uses more computational power. We believe that the extra precision from labelling objects at the pixel level would be needed for RoboCSE to precisely identify object affordances and relations and to successfully accomplish our future goals, such as performing high-level task planning that is grounded in the object instances in the robot's environment.

The object labels returned by the object classifiers are automatically added to the map returned by Octomap in real-time. Each pixel of the map contains a parameter that stores the object label it belongs to. The final result of the perception pipeline is a semantic map, or a map with object labels.

Results

We have tested the perception pipeline by deploying the software onto a Fetch Gazebo simulator. Figure 3 shows the original simulated environment of the robot; Figure 4 shows the output of the SLAM algorithm; and Figure 5 shows the semantic map generated from the entire perception pipeline.

Figure 3: Original simulated environment of the Fetch gazebo environment.

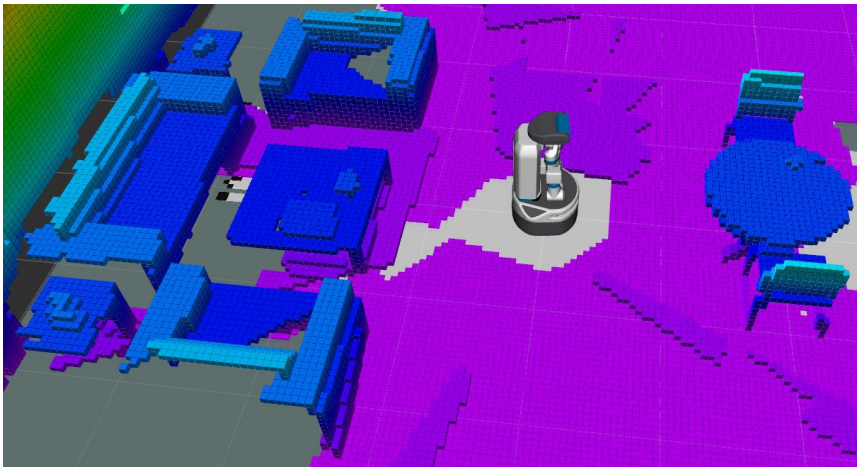

Figure 4: Output of SLAM algorithm: point occupancy. Color is based on pixel height.

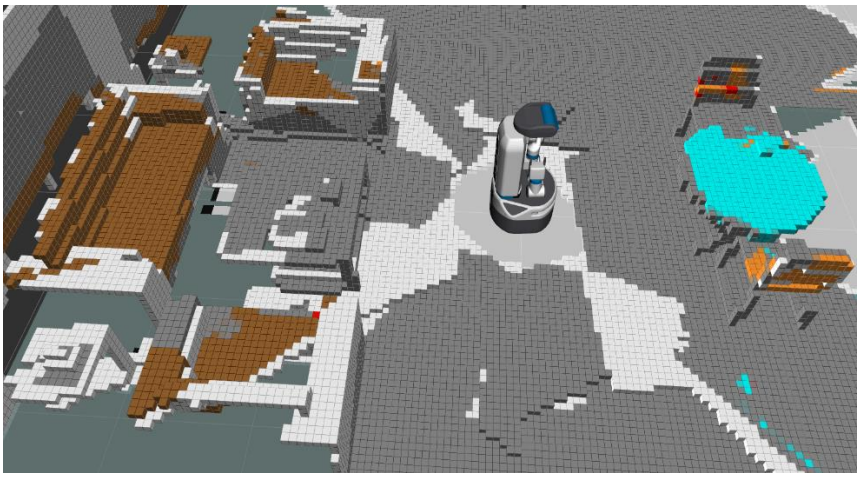

Figure 5: Output of perception pipeline: semantic map. Color is based on object class.

Discussion

We were pleased with the quality of the occupancy map (i.e. whether a point was occupied by an object or not), which was generated by Octomap. Based on visual inspection, we can clearly see the outlines of couches on the left of Figure 4, and two chairs surrounding a small table towards the right of image. However, the object identification and labelling proved to perform much more poorly, as seen in Figure 5. Figure 6 below shows the ground truth of what each object should have been classified as:

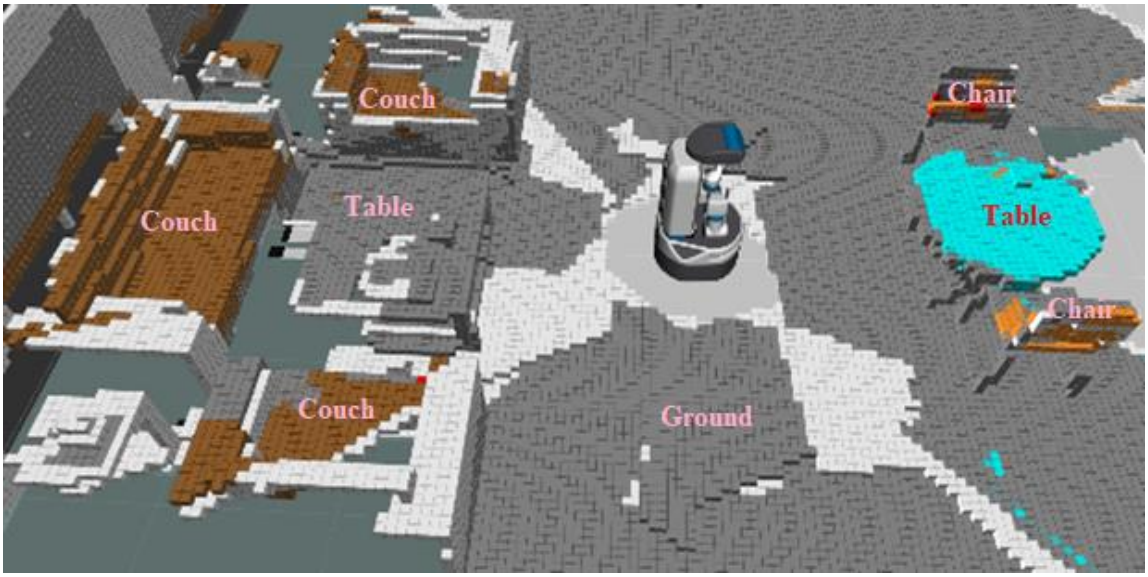

Figure 6: the ground truth of the objects in the semantic map. Couches should be colored brown, ground should be colored dark grey, table should be colored blue, and chair should be colored dark orange. Anything uncertain is light grey.

A substantial portion of the couch was not correctly colored brown, and the table on the left side of the image was not correctly classified at all. Furthermore, only a few spots of the chairs on the right side of the image were correctly colored dark orange. We do not have ground truth at the pixel-level, so the percentage of points classified correctly can only be described qualitatively.

The lack of accuracy for the object classification suggests that future work will need to be

done to improve the object classifiers. Examples of possible areas of improvement for the object classifiers include training on more diverse data sets to reduce any bias that may have been learned by the robots. In particular, many of the examples that the robot has trained on contains the labelled object placed in the center of the image or PointCloud. Unfortunately, when the robot drives around the room, the object is often not positioned well in the image or PointCloud obtained from the robot camera or 3D LiDAR sensor. As a result, the object classifiers' performance may suffer when trying to classify the objects in the robot's images and PointClouds of the environment. To solve this, we should consider training the neural network on images that contain blurry and/or off-centered objects.

Another area that could be improved is the method that the map uses to incorporate semantic labels outputted from the neural network. Each time the robot's camera views a point in space, the object classifiers output an object class (e.g. "chair" or "door") for that point, and the semantic map overwrites the class that it stores for that point. This results in overwriting what the robot had previously thought the object class of that point was. For example, imagine that the camera senses the point with coordinates (25, 62, 17) many times, and the object classifiers say that there is a chair at (25, 62, 17) each time. Further assume that the robot then lurches forward, gets a blurry image that contains point (25, 62, 17), and believes that (25, 62, 17) belongs to a door for this one timeframe. Our semantic map would then store the most recent reading of (25, 62, 17) being a door instead of storing the more probable label for (25, 62, 17) as being a chair. One approach to solve this problem is to develop a probabilistic filter that uses a recent history of readings for a point and gives the most confident class for that point. Only if (25, 62, 17) is believed to be a part a door for multiple sensor readings should the filter believe there is a door at that point.

Future Work

After improving the quality of the semantic map, we have two main extensions that we would like to work on. The first extension is to build another software module that uses the concepts and relations learned from RoboCSE to perform high level task planning. For example, suppose that a human commands the robot to pour them a cup of coffee. The first step for the robot would be to locate a cup. Based on learned knowledge, RoboCSE knows that a cup is likely to be found in a cupboard, so the high-level task planner tells the robot to navigate to the cupboard (whose location is assumed to be known in the map). We assume that there is a separate low-level controller that would actually handle the task of navigating to the cupboard and opening the cupboard. Assuming that there indeed is a cup in the cupboard, RoboCSE identifies the appropriate object relation (i.e. cup at location cupboard) after the semantic map is updated. The high-level task planner, which knows that a cup should be on top of a counter or table before pouring coffee into it, will return the next step of picking up the cup and placing the cup on a counter or table. This loop of calling the high-level task planner, then the low-level trajectory controller, then the perception pipeline, then RoboCSE, and then the high-level task planner again will continually repeat until the task is solved.

The second extension that we would like to work on is sharing knowledge between two robots. Assume that there are two robots situated in two different households. Furthermore, suppose that the human would like the robot to deliver a piece of fruit, but the robot does not know that fruits are usually inside the kitchen (or more specifically, the refrigerator). The robot will then need to learn where the fruit is located, such as by asking the human. After learning where the fruit is located, the robot can share this information with other robots so that other robots would not have to learn this piece of information from scratch. The types of knowledge

that we hope to share among robots is not limited to just location of items, but also new concepts (such as what a fruit is).

Conclusion

Many prior approaches to knowledge reasoning frameworks have certain limitations. While Bayesian Logic Networks [1] can generate accurate probabilistic inferences and predictions, they are not easily scalable. While KnowRob [3] and Robobarista [4] improve upon the scalability of Bayesian Logic Networks, they suffer in their ability to truly learn and generalize knowledge. RoboCSE is a new knowledge reasoning framework that strikes a great balance between the ability to learn knowledge and the ability to scale to large knowledge sets. RoboCSE uses a multi-relational embeddings to reason certain properties about objects, such as their affordances.

The project has developed a perception pipeline to test the capabilities of a new knowledge reasoning framework named RoboCSE [6], which leverages object affordances to draw rich inferences [5], [6]. The perception pipeline consists of two main components: a SLAM package named Octomap [7] and two object classifiers (PointNet2 [8] and MobileNetV2 [9]). Together, the SLAM algorithm and the object classifiers output a semantic map. While the SLAM algorithm gave accurate point occupancy data in the semantic map, the object classifiers did not output accurate object classification. Further work is needed to improve the object labels provided by the object classifiers.

With the semantic map, we hope to test RoboCSE's capabilities for high-level task planning. Many problems, such as pouring a cup of coffee, have hidden constraints and requirements. We hope to leverage RoboCSE's knowledge of object affordances and locations to

infer many of those constraints in high-level task planning. We also plan to use the semantic map to test RoboCSE's capabilities for knowledge sharing between robots. Similar to how humans can often learn faster when they work together, we hope that robots can learn knowledge faster when they are able to share knowledge and concepts among each other.

References

[1] D. Jain, S. Waldherr, and M. Beetz, "Bayesian logic networks," *IAS Group, Fakult¨at f¨ur Informatik, Technische Universit¨at M¨unchen, Tech. Rep*, 2009.

[2] Sonia Chernova, Vivian Chu, Angel Daruna, Haley Garrison, Meera Hahn, Priyanka Khante, Weiyu Liu, and Andrea Thomaz. "Situated bayesian reasoning framework for robots operating in diverse everyday environments".

[3] Moritz Tenorth and Michael Beetz. Knowrob: A knowledge processing infrastructure for cognition-enabled robots. *The International Journal of Robotics Research*, 32(5):566–590, 2013.

[4] Jaeyong Sung, Seok Hyun Jin, and Ashutosh Saxena. "Robobarista: Object Part based Transfer Manipulation Trajectories from Crowd-sourcing in 3D Pointclouds," 2016.

[5] Angel Daruna, Zsolt Kira, and Sonia Chernova. Towards Scalable Semantic Reasoning Frameworks for Robotics Systems, *RSS Workshop*, 2018.

[6] Angel Daruna, Weiyu Liu, Zsolt Kira, and Sonia Chernova. "RoboCSE: Robot Common Sense Embedding". 2018.

[7] Hornung, A., Wurm, K.M., Bennewitz, M. et al. Auton Robot (2013) 34: 189. https://doi.org/10.1007/s10514-012-9321-0

[8] Qi, Charles et al. "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space." Jun 2017, arXiv preprint arXiv:1706.02413v1

[9] Sandler, Mark et al. "MobileNetV2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[10] R. Kummerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner, "On measuring the accuracy of SLAM algorithms," Autonomous Robots, vol. 27, no. 4, pp. 387–407, 2009.

[11] Sebastian Thrun and Michael Montemerlo. The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures. *Stanford AI Lab*, 2004.

[12] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-Time Loop Closure in 2D LIDAR SLAM, *IEEE International Conference on Robotics and Automation*, 2016.