UT Austin Villa@Home
2022 Team Description Paper

Yuqian Jiang Haresh Karnan Minkyu Kim David Balaban
Steven D Patrick Justin Hart Luis Sentis Peter Stone

November 28, 2021

Abstract. UT Austin Villa has participated in five RoboCup@Home competitions, performing respectably in each. What is more exciting, however, is that we have begun a strong program of research that has been in part inspired by our efforts in this competition. It is our intention to build a comprehensive service robot system which is used in our laboratories, in a real-world deployment in our CS Department, and to compete in RoboCup@Home. In this Team Description Paper, you will find the highlights of our efforts and our plans for 2022.

1 Introduction

Using the RoboCup@Home team as a focal point for inter-department and inter-laboratory collaboration, UT Austin Villa@Home has pursued an ambitious research program towards the goal of the development of a comprehensive service robot system. We want to enter RoboCup@Home not with a suite of different programs for each round, but with a single program which is capable of competing and winning.

UT Austin Villa@Home is a collaborative effort between PIs and students in the Computer Science, Mechanical Engineering and Aerospace Engineering departments at the University of Texas at Austin. We have competed in five RoboCup@Home events. In 2007, we took second place. In 2017, we entered into the newly-formed Domestic Standard Platform League (DSPL) and took third place, having received our robot only a couple of months before the competition. In 2018, the team developed a design intended to allow us to develop a single system which would enter into all of the stages of the competition, encompassing knowledge representation, mapping, and architectural aspects. The team advanced to the second stage and was able to score in difficult tasks such as Enhanced General Purpose Service Robot (EGPSR). In 2019, we improved the system with better perception and manipulation modules. During the COVID-19 pandemic, we continued to develop our object recognition and

manipulation capabilities in the HSR simulators, and finished in the 3rd place in the 2021 competition. The progress made would set a good starting point for 2022 and open many research opportunities. Our efforts have resulted in six publications [1,2,3,4,5,6]. Going into 2022, we plan to further improve the core components of our system and develop more rigorous approaches to the tasks.

2 Software and Scientific Contributions

This section describes the component technologies we developed across multiple tasks for our robot architecture, knowledge representation, semantic perception, object manipulation, and person following on top of the HSR software stack. To the extent possible, we built our approach in a manner consistent with our ongoing Building-Wide Intelligence project [7]. While using a different hardware platform, many of the objectives and capabilities are the same. Indeed we have previously designed an underlying architecture that is common to the two platforms [6].

2.1 Robot Architecture

Our architecture is designed for service robots to handle dynamic interactions with humans in complex environments. The three-layer architecture, as shown in Figure 1, outlines integration of the robot's skill components, such as perception and manipulation, with high-level reactive and deliberative controls. The top layer sequences and executes skills, and is reactive during execution to respond to changes. A central knowledge base facilitates knowledge sharing from all the components. The deliberative control layer uses the knowledge base to reason about the environment, and can be invoked to plan for tasks that cannot be statically decomposed. Details on implementation of these layers can be found in our recent paper [6].

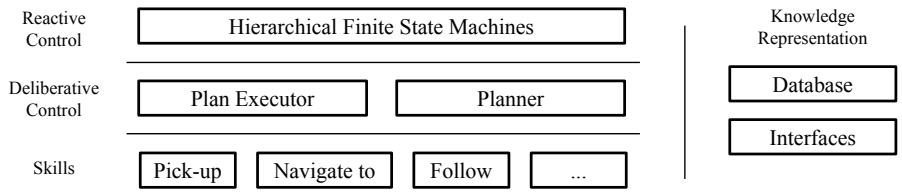

Fig. 1. Implementation of our robot architecture on HSR.

2.2 Knowledge Representation and Planning

Our knowledge representation subsystem stores knowledge in a SQL database in order to allow for fast access and easy querying. For instance, in the Tidy Up task

of the 2021 competition, the knowledge base is used to store object categories and their corresponding placement locations. Queries can be formed using custom C++ and Python libraries. The knowledge base can be interfaced through a simple predicate logic form which can be then imported for task planning. Our task planning module utilizes Answer Set Programming (ASP) to describe the rules for planning and reasoning, and the solver Clingo to generate optimal task plans. Core to our KR subsystem is the ability to reason about hypothetical objects. Details on our knowledge representation and planning system can be found in our recent paper [2].

2.3 Perception

We employ a semantic perception module whose purpose is to process raw video and depth data from the robot's sensors and extract information that can be processed by the manipulation, navigation, and knowledge reasoning modules. The main output representations are a query-able point cloud of objects in the environment and a partial 3D map of the world.

The main input to our semantic perception module is RGBD camera data. Compressed RGB and depth images from the robot are streamed to an offboard computer that runs the perceptual system. This image data is then consumed by finding objects via the YOLO object detection network [8], and constructing a point cloud. Next, semantic information about the world is synthesized in two main ways: a partial 3D environmental map and object cloud. For the former, regions of the point cloud corresponding to detected objects are fused together over time in a probabilistic Octree representation based on Octomap [9], which allows for the realtime construction of a partial 3D map of the world. For the latter, point estimates of the locations of objects are stored in a KD-Tree and wrapped with an efficient querying interface that integrates with our knowledge representation system. The synthesized semantic information is then made available to plugins in an event-based model, where a plugin can request access to semantic information that it wants to operate on. Plugins used include custom RANSAC edge detectors used to detect surfaces, and bounding box fitting on the 3D map for use in manipulation.

A significant limitation is the partial nature of the 3D environmental map. Only a partial map is constructed due to the realtime processing constraint; namely, full views of the world cannot be stitched together at framerate using the Octomap technology. Alternatively, GPU-based techniques for combining full point clouds could potentially overcome this limitation, and thus provides a direction for future development. Benefits of having full 3D environmental maps include the ability to directly localize objects with respect to the robot.

2.4 Manipulation

The purpose of our manipulation system is to enable the pick up and put down of diverse objects of different shapes and sizes. Our manipulation stack consists of

three main components which we describe below: grasp pose generation, parallel motion planning, and closed-loop correction.

First, our semantic perception system provides 3D bounding boxes for objects worth manipulating. Based on these bounding boxes, potential grasp poses are computed that place the gripper on the top of the object as well as on all sides, with multiple possible rotations of the wrist. Of these poses, invalid configurations are filtered out by projecting the gripper onto the object and seeing if there is a collision.

Once grasp poses are determined, motion plans need to be determined in order for the robot to achieve a desired grasp pose. In order to do this quickly, we employ a parallel motion planning architecture built on top of the Moveit framework [10]. Our motion planning architecture is comprised of primary and secondary nodes. The secondary nodes handle generating motion plans for each potential grasp pose, while the primary node coordinates and handles executing motion plans. Specifically, secondary nodes plan in parallel, and the first motion plan found is what is executed. The rationale behind this is that different grasp poses will require different yet unknown amounts of time for finding motion plans. Since motion planning takes a significant amount of time, reducing this bottleneck greatly speeds up the entire manipulation pipeline. Furthermore, the Moveit framework can sometimes crash when trying to find plans. In our setup, this problem is mitigated: If a secondary node dies from such a crash, then the other secondary nodes are still present, allowing the system to continue functioning.

Next, executing a motion plan precisely is usually not feasible. This is because, as the plan is executed, the software solely uses odometry to control its position and the resultant drift can cause errors in how much the robot thinks it has moved. To overcome this obstacle, we slightly modify desired grasp poses by having the gripper be some offset away from the object. This way, after a motion plan is generated and executed, the robot's gripper is close to the object, but there remains a small gap. We take advantage of this small gap by employing a proportional controller based on object detections from the robot's hand camera to correct for odometry drift. This practically means that the robot shifts slightly to align the gripper perfectly with the centroid of the object. The gap is then closed by moving in a straight line towards the object, leading to a successful grasp.

2.5 Person Following

To achieve robust and efficient person-following capabilities, perception, robot gaze control, and navigation must be effectively integrated. Recently, vision-based human recognition has dramatically improved with new software that relies on deep learning-based technologies, but these approaches have a limited range of sight. To resolve this problem, laser-based methods [11][12] and various sensor fusion techniques combining face recognition and leg detection have been employed [13][14]. However, there remain major difficulties include handling

occlusions, identifying target people among crowds, and effectively detecting human faces. To surpass these limitations, new techniques have been devised that rely on extra features, such as the detection of clothes, bags, and shoes [15].

Another problem is due to the use of passive perception techniques where the robot stays stationary, thus losing its target. Therefore, it is highly desirable for robots to achieve active perception such that people can be followed despite their movement. Many researchers have studied this problem within the topic of active perception or visual sensor planning [16]. This kind of problem is usually intractable because there are too many variables. However, using prior knowledge, context, and logical assumptions about the environment, it is possible to find solution approximations. If a robot is aware of the connectivity between spaces, when the target suddenly disappears from the robot's view, one strategy could be to navigate to the anticipated point using the last observed location to look for the target. This space connectivity can be simplified with the use of a topology map or graph . One other key factor is that robot skills should be integrated in harmony with the perceptual processes to improve a robot's ability to adapt to the various dynamic circumstances. For example, actions such as searching for a target, tracking, and navigating should be properly coordinated. To achieve such coordination, we employ the behavior-tree method [17] to sequence skills.

In summary, we develop object tracking capabilities using sensor fusion, active search using trajectory and waypoints predictions, and construct fully autonomous behaviors to follow people including temporary losses of the target being followed. Details on our person following approach can be found in our recent paper [3].

3 2021 Developments and Results

In this section, we discuss our developments in object detection and manipulation for the 2021 competition, successes and limitations of our approach, and directions for future improvements.

3.1 Motivation

The Tidy Up task in RoboCup@Home 2021 requires recognizing and manipulating many objects in a crowded indoor space. Our previous object perception pipeline involves combined manual data collection and labeling to train the YOLO Darknet v2 object detection framework. We aim to first enable automatic training data generation from the simulation environment, and then replace the Darknet framework with the new YOLOv5 architecture.

3.2 Automated Labelling of Object Data in Simulation

Deep neural network based object detection algorithms are data hungry. To train such networks, we need to perform the laborious process of manually annotating images of the objects in the task environment.

To simplify this process, we build an automated labelling procedure utilizing the ground truth object information published by the Gazebo simulator. First, we drive the robot around different object configurations in the simulated apartment in RoboCup@Home 2021. Then, we extract the 3D ground truth bounding boxes of the YCB objects from the gazebo simulator and re-project them into the camera frame to get the 2D bounding boxes for the YCB objects.

Using this automated label generation procedure, we collect about 1M annotated labels for 48 objects in different configurations and different poses of the robot. The Tidy Up challenge also includes detecting non-YCB objects such as handles of it a cupboard for which the gazebo simulator cannot provide groundtruth 3D bounding boxes. To address this problem, we collect a few samples of the non-YCB objects in different viewpoints, we manually segment the objects through our custom labelling tool, perform data augmentation and randomly paste the object into our existing training data.

This two-step procedure of first generating training data through our automated data labelling procedure and then random-pasting of non-YCB objects into the dataset ensures we cover all objects of interest for the RoboCup@Home competition, while also generating plentiful data to train the object detection networks. We train a YOLOv5 network on this dataset.

3.3 Fast YOLOv5 Detections

We build the YOLOv5 model with the TensorRT library for high performance inference. The processing time of one frame is only a few milliseconds on the test hardware. This implementation of the YOLOv5 architecture is integrated with our object tracking and knowledge representation modules developed described in Section 2, so that object information can be easily queried for manipulation tasks, at high frame rates.

The fast YOLOv5 detections also enable real-time, closed-loop grasp adjustments when applied to images from the HSR's hand camera. We use the position of the generated 2D bounding box to align the gripper with the target object. A proportional controller is used to publish a velocity command to the robot base based on the distance between the center of the hand camera image and the center of the bounding box. Once this distance is within a certain tolerance, the hand is directly above the target and the velocity command is set to zero.

One limitation of this approach is that it only runs on GPUs that are compatible with the TensorRT library.

3.4 Results and Future Directions

This method reduced human effort and error in capturing and labeling images of objects. The results show faster and more accurate object detections compared to our previous approach, which led to improved task performance in RoboCup@Home 2021.

In future work, we will study sim-to-real transfer from training with automatically labelled synthetic object data to real-world object recognition and manipulation tasks.

4 Conclusion

UT Austin Villa@Home has been a strong competitor and has a tradition of synergistic research our RoboCup@Home team and our other research efforts. RoboCup@Home has become a driving force in robotics research at UT Austin. We look forward to seeing everyone in Bangkok in summer 2022.

References

1. Rishi Shah, Yuqian Jiang, Haresh Karnan, Gilberto Briscoe-Martinez, Dominick Mulder, Ryan Gupta, Rachel Schlossman, Marika Murphy, Justin Hart, Luis Sentis, and Peter Stone. Solving service robot tasks: Ut austin villa@home 2019 team report. In *AAAI Fall Symposium on Artificial Intelligence and Human-Robot Interaction for Service Robots in Human Environments (AI-HRI 2019)*, November 2019.
2. Yuqian Jiang, Nick Walker, Justin Hart, and Peter Stone. Open-world reasoning for service robots. In *Proceedings of the 29th International Conference on Automated Planning and Scheduling (ICAPS 2019)*, July 2019.
3. Minkyu Kim, Miguel Arduengo, Nick Walker, Yuqian Jiang, Justin W Hart, Peter Stone, and Luis Sentis. An architecture for person-following using active target search. *arXiv e-prints*, pages arXiv–1809, 2018.
4. Justin W. Hart, Rishi Shah, Sean Kirmani, Nick Walker, Kathryn Baldauf, Nathan John, and Peter Stone. Prism: Pose registration for integrated semantic mapping. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2018.
5. Justin Hart, Harel Yedidsion, Yuqian Jiang, Nick Walker, Rishi Shah, Jesse Thomason, Aishwarya Padmakumar, Rolando Fernandez, Jivko Sinapov, Raymond Mooney, and Peter Stone. Interaction and autonomy in robocup@home and building-wide intelligence. In *Proceedings of the AAAI Fall Symposium on Artificial Intelligence and Human-Robot Interaction (AI-HRI)*, October 2018.
6. Yuqian Jiang, Nick Walker, Minkyu Kim, Nicolas Brissonneau, Daniel S Brown, Justin W Hart, Scott Niekum, Luis Sentis, and Peter Stone. Laair: A layered architecture for autonomous interactive robots. In *Proceedings of the AAAI Fall Symposium on Reasoning and Learning in Real-World Systems for Long-Term Autonomy (LTA)*, October 2018.
7. Piyush Khandelwal, Shiqi Zhang, Jivko Sinapov, Matteo Leonetti, Jesse Thomason, Fangkai Yang, Ilaria Gori, Maxwell Svetlik, Priyanka Khante, Vladimir Lifschitz, et al. BWIBots: A platform for bridging the gap between AI and human–robot interaction research. *The International Journal of Robotics Research*, 36(5-7):635–659, 2017.
8. Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.

9. Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 2013. Software available at `http://octomap.github.com`.
10. David Coleman, Ioan Sucan, Sachin Chitta, and Nikolaus Correll. Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*, 2014.
11. Woojin Chung, Hoyeon Kim, Yoonkyu Yoo, Chang-Bae Moon, and Jooyoung Park. The detection and following of human legs through inductive approaches for a mobile robot with a single laser range finder. *IEEE transactions on industrial electronics*, 59(8):3156–3166, 2012.
12. Noriyuki Kawarazaki, Lucas Tetsuya Kuwae, and Tadashi Yoshidome. Development of human following mobile robot system using laser range scanner. *Procedia Computer Science*, 76:455–460, 2015.
13. Matthias Scheutz, John McRaven, and Gy Cserey. Fast, reliable, adaptive, bimodal people tracking for indoor environments. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1347–1352. IEEE, 2004.
14. Nicola Bellotto and Huosheng Hu. Multisensor-based human detection and tracking for mobile service robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):167–181, 2009.
15. Ejaz Ahmed, Michael Jones, and Tim K Marks. An improved deep learning architecture for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3908–3916, 2015.
16. Shengong Chen, Youfu Li, and Ngai M. Kwok. Active vision in robotic systems: A survey of recent developments. *The International Journal of Robotics Research*, 11(30):1343–1377, 2011.
17. Michele Colledanchise. *Behavior Trees in Robotics*. PhD thesis, KTH Royal Institute of Technology, 2017.

HSR Software and External Devices [DSPL]

We use a standard Human Support Robot (HSR) from *Toyota*. No modifications have been applied.

Fig. 2. HSR

Robot's Software Description

We are using the following 3rd party software:

- Object recognition: YOLOv5 and TensorRT
- People and activity recognition: OpenPose
- Manipulation: MoveIt
- Knowledge Base: PostgreSQL
- Planning and reasoning: Clingo
- State Machine: SMACH (ROS)

External Devices

We are using the following external devices:

- Alienware 17 Laptop (Backpack)
- MSI Laptop (Backpack)

Cloud Services

We are using the following cloud services:

- Speech recognition: Google Cloud Speech API