UNSW RoboCup@Home DSPL Team Description Paper

A. Golding, A. Tran, Z. Li, M. Pagnucco, Y. Song, and C. Sammut

School of Computer Science and Engineering & Creative Robotics Lab,
The University of New South Wales, Sydney, Australia,
`https://robotics.cse.unsw.edu.au/runsweep/`

1 Introduction

This team description paper is accompanied by a video that demonstrates some of our capabilities on the HSR. The videos, taken in our HRI laboratory, shows two stage I tasks "clean the table" and "carry my luggage" and a stage II task "where is this" from the 2019 @Home rule book. The demonstration includes: spoken interaction, world modelling, planning, environmental reasoning, mapping and navigation, SLAM, object recognition in different rooms, manipulation and person recognition.

Much of the current work continues to be focused on developing the robot's world model. This serves as the central repository of the robot's short-term and long-term memory. Somewhat like SOAR [1] and KnowRob [2], the world model is a symbolic representation that provides knowledge to the planner and the spoken dialogue system. The planner is based on a modern version of Nilsson's Teleo-Reactive planner [3]. This work is being done in collaboration with Prof. Keith Clark at Imperial College and Dr. Peter Robinson at the University of Queensland [4].

We use the robot's SLAM system, and episodic memory to give the robot spatio-temporal awareness. This knowledge can be used to assist in language understanding. For example, if the language alone is not sufficient to disambiguate between a reference to an object; proximity, function or recency can be used as reasonable guesses to resolve the reference. To achieve this, the robot requires mapping at several levels of abstraction. The lowest level is the occupancy grid created by SLAM. In addition, we require a topological map to associate spaces to names and relations. These can then be turned into logical predicates and reasoning applied within a logic framework. Connecting spatial reasoning to language understanding is the topic of a current postgraduate research project.

2 Background

We have a substantial code base that includes SLAM and autonomous navigation, topological mapping, multi-modal interaction for conversational agents, a deductive database for world modelling and an episodic memory system. We have added the TeleoR planning language [5], for implementing the robot's behaviours. The remaining components, such as object recognition are derived from existing open source software, especially pre-built ROS packages.

2.1 World Model

The world model has been under development since RoboCup 2023. The world model represents regions, objects and people and stores them in a deductive database, making it easy to access the locations of objects relative to specific regions and other objects. The object database is automatically filled as the robot autonomously explores its surroundings using a vision pipeline with inputs from YOLO. It also detects people in the environment and captures features that allow re-recognition. Meanwhile, the region database is populated from a topological map program that is able to identify regions in the occupancy grid created by SLAM.

The world model is central to the robot's intelligent behaviour as it stores the robot's current beliefs about the world. A current area of research is also exploring the maintenance of beliefs of each person in the environment. These provide the context for reasoning when, for example, the dialogue refers to objects or locations that are not directly visible. The world model is also the source of information for the domain description required by the planner.

2.2 Vision Pipeline

The vision pipeline provides a generalized image processing module. The pipeline is designed to reduce computational overhead by only processing frames that contain objects of interest. The modular plug-and-play approach allows for easy expansion of the core functions enabling research and development of new perception methods. The pipeline consists of object and human pose detection, object annotation including position within the map, size, shape, colour, and person recognition. Specific vision functions such as bag handle detection or empty seat checker can draw from the data generated by the pipeline. Ultralytic's YOLOv10 is used in our vision pipeline to handle object and human pose estimation. The Ultralytics framework provides seamless deployment capability and enable us to swap to newer YOLO models if appropriate.

2.3 Person Re-Identification

Person re-identification is an important part of human-robot interaction. Our person re-identification process *(Figure 1)* extends current state-of-the-art methods of object detection, classification and tracking. If a person is detected in the scene the object track is verified and updated, and the viewpoint of the person from the robot is determined using one of 8 defined viewpoints. Features from the face and head are extracted when available and compared to known people, if the person is recognised the features are updated in the world model, if not, a new person is added to the world model.

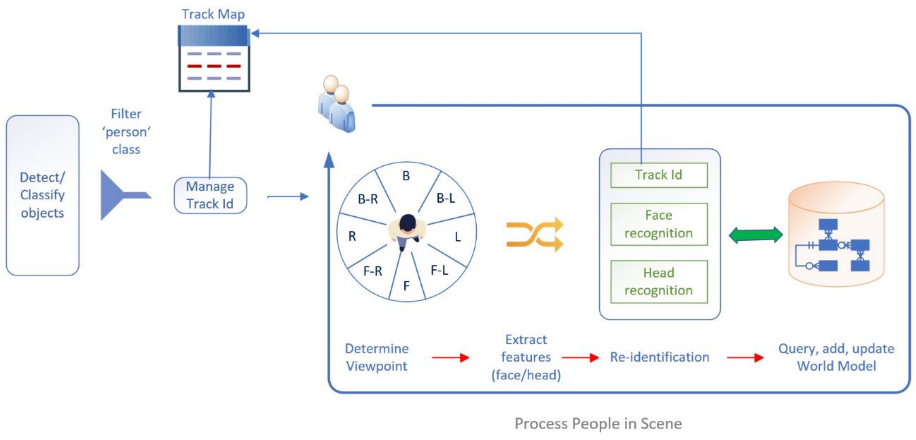

Fig. 1. Person Re-Identification High Level Architecture

2.4 Conversational Agent

The current dialogue system for the robot uses RASA, an open-source language understanding and dialogue management system [6]. RASA Core can be executed on a local host, with a ROS node serving as the interface that connects the RASA model and user input text. The interface executes in a node, constantly passing the input text to the NLP model (converted from speech by VOSK, a speech-to-text system, described in 2.6). The output from the model is converted into goals for the TeleoR planner.

The conversational agent is also connected to the HSR's path planner to be able to output directions for the "where is" task. The parser is augmented by the world model, which has knowledge of relations such as "contained in", so that the robot can correctly respond to utterances like "I want a bag of chips". The system knows that the chips are in the kitchen cupboard so directions are given to the cupboard. The SLAM map is labelled with the locations of objects and spaces, which correspond to the symbolic spatial relations stored in the world model. This provides the information required to turn the path generated by the movement planner into words that make sense to the human user.

2.5 Object Recognition and Grasping

For object recognition we use YOLOv8 [7] to detect objects in the scene, placing bounding boxes around them and then using a point cloud generated from the RGB-D camera to locate the object in space. When attempting to grasp the object, we used ROS packages for finding the grasp points and planning the arm movement.

As described in Section 2.1, once the recognition system has done its job, the object's properties, pose and location are stored in the world model so that this information is available to the language and planning nodes.

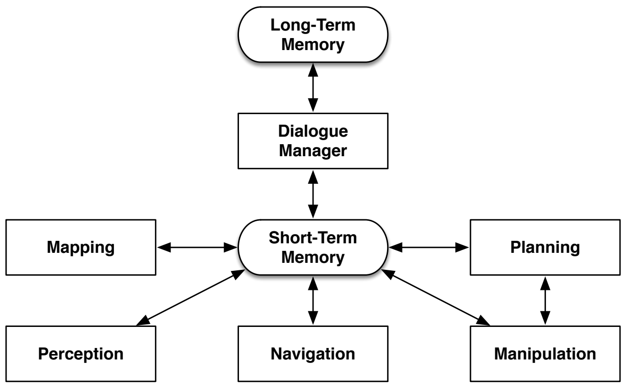

Fig. 2. Software architecture of UNSW@Home

2.6 Externally Available Components

As above, some components are derived from existing open source software, especially ROS packages. We use the MoveIt or Agile Grasp ROS packages for calculating inverse kinematics and performing manipulation tasks. For face recognition and person tracking we use tools in OpenCV 3.0, and the OpenNI/NiTE skeleton tracking library. The speech-to-text voice recognition software currently being used is Vosk. Additionally, we use some off the shelf cloud APIs such as Micorosoft Azure's Face API for gender detection. The TeleoR rules is written in *iProlog*, an experimental Prolog interpreter developed by our team. Later, this will also be ported to QU-Prolog[1]

3 Research

One of the goals of our research is combining high-level reasoning with real-time low-level sensing and control to improve the capabilities of autonomous robots. Our long-term aim is to develop general-purpose intelligent systems that can learn and be taught to perform many different tasks by interacting with their environment. In the course of our research, we have created software that can be ported to the Toyota HSR for the RoboCup@Home competition. Below, we highlight the current focus of our research, and our key innovative technologies and scientific contributions. An overview of the software architecture of our @Home system is shown in Figure 2.

The heart of the system is the world model, implemented using the Postgres database system. Dialogue scripts can access and update the database with knowledge of people, places, objects and actions. This database implements the system's working memory and it's long term memory. Scripts can incorporate

[1] https://staff.itee.uq.edu.au/pjr/

input and output in different modalities, e.g. gestures as well as spoken dialogue. Responses are formulated as goals and passed to the TeleoR planner.

3.1 Human-Robot Interaction and Trust

Human-robot interaction may include speech, sound, music, gestures, body movements, proximity, facial expressions, body language and touch. Poorly designed interactions decrease the willingness of a human to use the robot. Our research aims to improve human-robot interaction by studying two areas, physical elements of human-robot interactions and the ability of the robot to learn from and adapt to new dynamics of the interaction.

The physical components of human-robot interactions we study are touch, gesture, and recognising human emotions through micro and macro human expressions, and the manner in which a robot approaches a human. [8] The goal is to prevent the human from being surprised or fearful of a robot's actions. We use machine learning to alter how the robot behaves and interacts so that the human can teach the robot how they wish to interact, explaining aspects of the interaction they prefer or dislike, find uncomfortable or confronting.

An associated concern is how trustworthy humans regard a robot, especially when they can learn and adapt to new situations. We are studying the change in trust for a mixed initiative task under varying degrees of transparency of the adaptation process. The cognitive architecture mentioned above includes the ability for the robot to adapt to a change. It is implemented on a Baxter robot for a mixed initiative problem solving task where the environment changes, requiring the robot to adapt on the job. This also requires modelling and evaluating the evolving human-robot trust relationship as the robot learns.

For our research in Human-Robot Interaction we have access to a National Facility for Human-Robot Interaction Research. It is a state-of-the-art facility for non-intrusive real-time measurement of the properties that are linked to human affect and intent.

3.2 Robot Learning

UNSW was known for its work in machine learning well before we began working in robotics. In fact, one of the main motivations for entering robotics is that it is such a rich source of data and problems that can be solved by learning.

The most recent work in robot learning employs a scene graph generator (SGG) [9] to learn to recognise components of an object and their relations. The SGG is a deep learning system that generates symbolic representations of these relations. Thus, it provides a bridge between the sub-symbolic perceptual system and the symbolic world model. In addition, a large synthetic data set has been created his Blender to automatically generate many variations in shapes and pose so that the system can learn a robust classifier.

We continue to develop an episodic memory system that enables the robot to recall past events that may be relevant to the interaction or to solving a present problem. This was the subject of a PhD thesis [10]. Event frames are stored in memory with the two primary problems being how to know what events should be remembered or forgotten and what is an appropriate metric to use to determine what a relevant memory is. We have used a novel knowledge acquisition system called "Ripple-Down Rules" to interactively acquire rules for identifying similar events and for determining which past events, stored in the episodic memory, are most relevant to the current situation.

In other work, we have developed methods for learning how to traverse difficult terrain by learning from demonstration and through trial-and-error [11]. We combine learning abstract qualitative models with reinforcement learning, where the abstract layer constrains search in the lower-control layer to greatly reduce the number of trials required.

Another area of interest is giving the robot the ability to learn how to use objects as tools [12]. This uses inductive logic programming to build theories of how objects of different shapes interact with other objects and reasoning how to position and move them so that the object selected as a tool can allow the robot to complete a task that it could not otherwise do, e.g. using an object as a hook to pull another object out of a narrow space. The perceptual system builds models that are imported into a physics simulator, which is used to "visualize" actions before they are executed, thus extending the robot's planning capabilities.

A limitation of previous work was the objects making up the tools had to be simplified for the vision system. The scene graph generator described above enables us to improve tool learning by extending the capabilities of the vision system to recognise more complex objects and their poses.

3.3 Teleo-Reactive Planner

In collaboration with Clark and Robinson, we are implementing a planner that makes use of the Teleo-Reactive (TR) programming paradigm. The main difference between a TR planner and a more traditional PDDL planner is that the TR system attempts to implement a universal plan that is reactive than predictive. A TR program consists of a collection of procedures, each of which contains a set of rules that are implement what Nilsson called "circuit semantics". That is, the system behaves as if all the conditions in all the rules are continuously being evaluated just as in an electronic circuit. A TR procedure has the general form:

```
proc(<arguments>)
    <goal> -> <do nothing>
    <conditionN> -> <actionN>
    ...
    <condition1> -> <action1>
    true -> <default action>
```

where the actions, may be calls to other TR procedures. The condition-action rules are arranged such that the action of a rule lower in the procedure should result in the conditions of a higher rule being satisfied. The lowest level rule is the default rule when no other condition is satisfied and it invokes a default action. The top rule's condition is satisfied when the goal for that procedure has been satisfied. Because the conditions in all rules are, conceptually, being re-evaluated continuously, the planner is very reactive, and thus, capable of responding quickly to changes in the environment.

As an example, a simplified set of TR rules for cleaning the kitchen table is:

```
clean_table :-
    not on_table(_) -> forget(current_task(clean_table));
    holding(X) -> put_in_dishwasher(X);
    on(X, table), near(table) -> pickup(X);
    not near(table) -> goto(table);
    not in(kitchen) -> goto(kitchen).

put_in_dishwasher(X) :-
    not holding(X) -> forget(current_task(put_in_dishwasher));
    near(dish_washer) -> place_in_dishwasher(X);
    true -> goto(dishwasher).
```

Percepts, such as *holding*, are implemented as queries to the world model, which includes the current robot state. Actions, such as *goto* are implemented as messages sent to a ROS action server that communicates to the appropriate nodes, like navigation.

At present, the TR rules must be hand-coded, but a subject of research is how TR can be learned, either from demonstration or by experimentation. The aim is to learn from far fewer examples than required by reinforcement learning.

4 Experiments and Results

The accompanying videos demonstrate results obtained using the Toyota HSR. These include:

- Natural speech interaction using a dialogue manager that understands the context of a conversation and uses the context to disambiguate utterances.
- The robot's sensor's give it an awareness of its surroundings and, coupled with mapping and knowledge contained in the world model, an awareness of space and objects in the environment.
- The vision system is capable of object recognition and tracking, human pose estimation and ray tracing of arms to provide real-world context to dialogue. The robot can differentiate between human users through facial recognition and texture matching.
- A state machine is used as the planner which takes input from the dialogue manager.
- Planning actions combining vision, knowledge from the world model, person recognition, natural language reasoning, path planning and manipulation.
- Manipulation makes use of the Toyota HSRB interface

5 Conclusion

The major thrust of our work in @Home has been developing the world model with its connection to the robot's perceptual system and the episodic memory to enhance the robot's understanding of time and space. The world model serves as the "glue" for the robot's cognitive architecture, so its development is crucial for further improvements in all of the robot's capabilities. With work on the world model mostly completed, incorporation of planning through the TR system is now the major focus of our new development.

References

1. John E Laird, Keegan R Kinkade, Shiwali Mohan, and Joseph Z Xu. Cognitive Robotics Using the Soar Cognitive Architecture. In *8th International Conference on Cognitive Robotics, (Cognitive Robotics Workshop, Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, pages 46–54, Toronto, 2012. ACM Press.
2. Moritz Tenorth and Michael Beetz. KnowRob – knowledge processing for autonomous personal robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4261–4266. IEEE, 2009.
3. N. J. Nilsson. Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1:139–158, 1994.
4. Keith L. Clark and Peter J. Robinson. Robotic agent programming in teleor. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 5040–5047, 2015.
5. Keith L. Clark and Peter J. Robinson. *TeleoR User Guide*, 2023.
6. Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. Rasa: Open source language understanding and dialogue management, 2017.
7. Mohammad Javad Shafiee, Brendan Chywl, Francis Li, and Alexander Wong. Fast YOLO: A fast you only look once system for real-time embedded object detection in video. *CoRR*, abs/1709.05943, 2017.
8. D. Silvera-Tawil, M. Velonaki, and D. Rye. Human-Robot Interaction with Humanoid Diamandini Using an Open Experimentation Method. In *24th IEEE Int. Symp. on Robot and Human Interactive Communication*, pages 425–430, 2015.
9. Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training, 2020.
10. Colm Flanagan. *Episodic Memory for Cognitive Robots in Dynamic, Unstructured Environments.* Phd thesis, School of Computer Science and Engineering, The Uniersity of New South Wales, 2021.
11. Timothy Wiley. *A Planning and Learning Hierarchy for the Online Acquisition of Robot Behaviours.* PhD thesis, School of Computer Science and Engineering, University of New South Wales, 2018.
12. Handy Wicaksono and Claude Sammut. A cognitive robot equipped with autonomous tool innovation expertise. *International Journal of Electrical and Computer Engineering*, 10(2):2200–2207, Jan 2020.

Annex

The foreground software used in 2024 has been made available at `https://robotics.cse.unsw.edu.au/gitlab/toyota-hsr/runsweep2024`. The software is described in the README.md file of the git repository. An excerpt of the readme is provided below.

The ROS packages of the foreground software are described below.

- **manipulation**
  - *hsrb-interfaces-unsw*: A fork of the toyota HSRB interface package
  - *object-grasper*: Package contains services for manipulation actions
- **msgs**
  - *unsw_action_msg*: Custom msgs for action dispatch
  - *unsw_vision_msgs*: Custom msgs for unsw vision pipeline
- **navigation**
  - *gmapping*: Fork of hsrb gmapping package used for mapping
  - *nav_utils*: Contains navigation scripts such as person-following. Intention to refactor to action_servers
  - *rosnav*: Edited fork of HSRB rosnav repository
- **planning**
  - *action_clients*: State machines to dispatch planner outputted actions to action action_servers
  - *action_servers*: Servers for all base actions. Containing manipulation actions at the moment
  - *state_machine*: Package used as proof of concept for continuous feedback using smach library
  - *tr*: In development proof of concept for teleo-reactive planner
- **speech**
  - *sound_stream*: Package to publish microphone stream to ROS topic
  - *speech_to_text*: Vosk based model for speech to speech_to_text
  - NOT INCLUDED:
    - *nlp-rasa*: Package containing scripts for nlp using rasa model
- **tasks**
  - *carry_my_luggage*: State machine/behaviour tree based control for carry my luggage task using services
  - *clean_the_table*: State machine based control for clean the table task, using action_clients and action_servers
  - *inspection_task*: State machine based control for inspection_task
  - *pick_up_apple*: State machine based control for simple integrated vision and manipulation task using services
  - *serve_breakfast*: State machine based control for serve_breakfast written at RoboCup 2024, unchanged and untested since competition. Will be refactored
  - *store_groceries*: State machine based control for store_groceries task written at RoboCup 2024, unchanged untested since competition. Will be refactored

- **tools**
  - *common_utils*: Scripts that don't really belong anywhere such as "door checker" to check if a door has opened
- **vision**
  - *person-recognition*: Person recognition and reidentification inserted in vision pipeline
  - *utils*: Specific vision nodes used for small tasks such as identifying a bag handle or a person pointing
  - *yolov8*: Wrapper of yolov8 for ROS
- **world model**
  - *world-model*: World model and scripts used to maintain belief store

We are using an external device for additional processing, compliant with the rules of the Domestic Standard Platform League: Dell Mobile Precision Workstation (Intel Core Ultra 9, NVIDIA GeForce 4090, 64Gb RAM) connected via Ethernet to the HSR, mounted on the standard backpack mount.

UNSW uses the following third party software and libraries for the competition:

- Vision Processing: YOLO, Face Recognition
- Grasping: MoveIt, Grasp Pose Detection (GPD)
- SLAM and Navigation: GMapping, ROS navigation stack
- Speech: Vosk
- Database: Postgresql

s