Physical HRI for the Elderly

1 Introduction

1.1 Motivation

Due to the increasing number of elderly individuals in need of care, assisted-living centers are experiencing a shortage in the number of professional caregivers. For example, as a consequence to Japan's low birth rate and long-life expectancy, their elderly population is aging rapidly with 22% over the age of 65 in 2009 and a predicted rise to 34% by 2035 [1]. There is a significant challenge in providing high quality care for the growing population of elderly people. The decrease in skilled workers can negatively impact the quality of professional care, the wellbeing of care-providers, and the integrity of individuals in need of care [2]

1.2 Introduction to Problem

Despite progressive research in physical and social robots for the elderly, few autonomous robots exist due to ethical concerns, existing prejudices around digitalization and automation, and complexity of care ecosystems [2]. Senior citizens need more individualized care and attention [3]. The creation of specialized care robots could potentially decrease feelings of loneliness while improving quality of life [3]. They could also be designed to benefit elderly people experiencing depression or dementia-related conditions which may be difficult for caregivers to manage [3].

We identified one such area to be the simple task of retrieving objects which may be difficult for elderly people to reach or locate. The elderly could have issues in doing this task as there could be physical challenges like body impairment or even the location of the objects, such as high on a shelf or low on the floor. We developed a socially assistive robot algorithm using a TIAGo robot to help the elderly by performing such tasks as retrieving objects through verbal commands.

1.3 Background

Elderly individuals may lack the physical ability to accomplish tasks and would possibly need to be reminded to perform various activities such as taking medication or staying hydrated. They may need to be reminded of the locations of certain items or components of activities they are required to perform for their health. Robots could help overcome mobility problems and reduce elderly individuals' dependence on overworked caregivers [1]. For example, Pepper is a robot especially designed to support older adults and caregivers in assisted-living to increase physical and cognitive activity [2]. Physically assistive robots can help the elderly with various tasks like retrieving items, monitoring activity, providing food, etc. They could aid feelings of loneliness by providing companionship and therapy making depression and dementia-related conditions easier for caregiving employees to manage. The goal for elder-care robots is to work alongside human workers by filling in gaps in care, improving quality of life, promoting independence, and aid in monitoring senior citizens [3].

For this research, we developed an algorithm that would assist elder adults with retrieving objects. The TIAGo robot is able to locate and retrieve items from verbal commands. TIAGo asks the elderly what items it can retrieve for them, listens to the input, locates the item, and brings it back to the elderly person. For this activity, the table consisted of five items: a bottle of water, a pill bottle, a jar of oats, a jar of nuts, and a multi-vitamin bottle. The elderly can request any of these items when prompted by the TIAGo, and the TIAGo will move to the table, search for the location of the requested item, and move the item back to the target table where the elderly person requested the item. To increase the social behavior of the TIAGo, we incorporated social cues to make the TIAGo more interactive for the elderly and increase trust such as looking at the user when communicating. During the interaction, TIAGo politely introduces itself and its capabilities, prompts the user for a task, repeats the prompt after completing each task, and finally bids farewell to the elderly person as it returns to its base position when no tasks are left to complete.

2 Our Approach

2.1 High-Level Overview

Our approach to this problem incorporates four main pipelines: speech, navigation, object detection, and arm movement (pick and place). The high-level overview is detailed in the figure below. First, TIAGo is initialized as it moves to the target table where the user is located and introduces itself and its capabilities. It will then prompt the user for a command. The user has the option to either say "no thank you" or request an object available on the inventory table such as a bottle of water, a pill bottle, a jar of oats, a jar of nuts, and a multi-vitamin bottle. TIAGo will listen for keywords through the speech detection pipeline. If no keywords are detected, TIAGo will notify the user, and ask them to repeat themselves. If the command is "no thank you," the loop will terminate, and TIAGo will move to its home position and shut down. If a keyword is detected, TIAGo will acknowledge the request and confirm the desired item. It will then move to the inventory table, raise its torso, and begin marker detection by retrieving the dictionary of markers and their positions. It can then check if the requested object's marker ID is in the dictionary. If TIAGo cannot find the marker ID, it will move its head to a better position and attempt to detect the marker again. In case TIAGo does not see the marker ID the second time, it will let the user know and move back to the target table. However, if the object is detected with the correct marker ID of the requested item, the robot will acknowledge the object detected and communicate its intent of grabbing the item by moving its arm. Next, it will move the base to where the arm can extend without hitting the table and pick up the object with its arm. TIAGo will grasp the object and communicate to the user that it is moving

Commented [AK1]: Main ethical concerns: potential reduction in amount of human contact, increase in feelings of objectification and loss of control, loss of privacy, loss of personal liberty, deception and infantilization, and circumstances in which elderly people should be allowed to control robots (2)

Commented [AD2]: Added a new part to this

towards the target table. It will then drop the item off at the target table and let the user know the robot has completed the task. Finally, the process restarts where TIAGo will prompt the user for another command. The user can continue to request commands or say "no thank you" in which case the robot will move back to the home position and shut down.

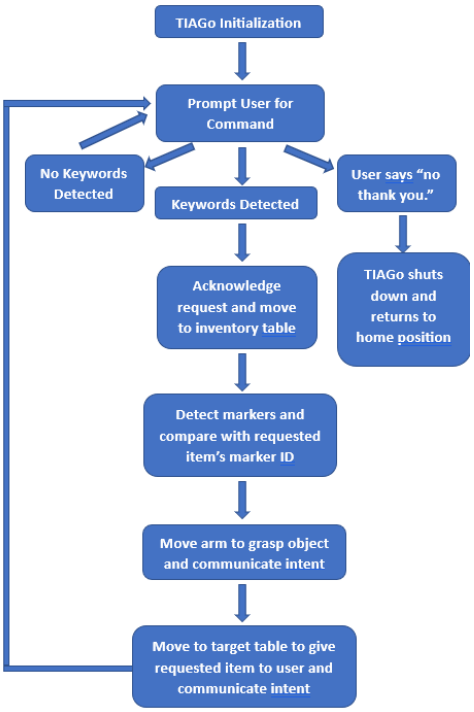

Fig. 1: High-Level Overview of Algorithm

2.2 Speech Pipeline

In the speech pipeline, TIAGo introduces itself and its capabilities of retrieving objects such as water bottles, pill bottles, oats jar, multi-vitamin bottle, and jar of nuts. It will prompt the user to request an item after its introduction using text-to-speech (TTS). When listening for commands, the listener class is activated which uses Google speech recognition to listen for commands from the user. It will then parse through the user input to find keywords using natural language processing with spaCy in Python. The class uses a single list of keywords to look for the command in the raw text and match the keyword to the object name and marker ID which is used for object detection.

Throughout the interaction, the robot is very polite and continuously updates the user about what its intention is before it moves. If the TIAGo prompts the user for an item, and the user replies "no thank you," TIAGo will respond "the pleasure was all mine. I will now go to my starting position for shutdown" and move to the home position. If the user requests an item that is not recognized, TIAGo will respond "I was not able to understand you. Could you please repeat your request?" Finally, if the user requests any item, the robot will acknowledge and confirm the requested item by telling the user it will bring them the object. Before it moves the base to the inventory table, detects the object, picks up the object, moves to the target table, or places the object on the table, TIAGo will communicate its intention to develop trust in the user.

2.3 Navigation Pipeline

In order to navigate between the home position, target table, and inventory table, we used distance-time odometry. The robot was able to track its position using odometry which means it drives in the specified direction for a certain period of time. It uses a function with defined waypoints to represent important locations in the workspace. All move functions use the waypoints such as the move_to_point function which also uses the move_distance function and move_angle to performs distance-time calculations and publishes them for correct durations. Velocity values were constrained for safety purposes since TIAGo was operating in a small space. Twists were generated using linear and angular velocities and published to the mobile base controller to perform movements.

During the interaction, TIAGo moves from the home position to the target table where the user is sitting. It will face the user as it receives a command for object retrieval. It will then move to the inventory table to grasp the item and move back to the target table to place the item and prompt the user for any additional requests. If the user is done requesting items, TIAGo will move back to the home position and shutdown.

2.4 Object Detection Pipeline

To detect objects on the inventory table, we implemented a marker manager class with a dictionary to store marker IDs and positions. These were initialized with hard coded values for testing purposes. The torso is moved up to get a better look at the objects on the table using the torso controller. Object detection was implemented as a separate node (Aruco node) running on the PC. This node reads the camera images and uses CV to calculate the pose of the markers with respect to the 'base_footprint' link. The poses of the markers are published on a rostopic and our class subscribes to this topic and stores the poses + IDs in the dictionary. From the marker list, we can calculate the arm position and orientation relative to the marker. We calculated the offset from the arm using tf which can interpolate transformations.

2.5 Arm Movement (Pick and Place) Pipeline

In order to move the arm to pick and place objects, the x and y distance from the gripper to the object is calculated first. The base is then moved backwards to allow the arm to extend and grasp the object. Play motion is used to extend the right arm. The torso controller is used to center the position of the torso for object pickup. The move form arm to object function maps coordinate frames to move the arm to the correct location with a hard-coded joint position. The gripper then closes to grasp the object using play motion and moves to a safe position before turning towards the target table to move. The torso is lowered to drop the object off. The item is placed on the target table by opening the gripper. Once the object is placed, the right arm is retracted using play motion.

3 Limitations and Future Work

Limitations of the project include minimal experience using ROS, initial difficulties with deploying code, and general issues with Aruco. There was initially no workstation for the TIAGo robot, so students who did not have a Linux computer or virtual machine were at a disadvantage. It would have been helpful to have more testing time or more of an individualized safety tutorial on how to use the TIAGo robot, so students would be allowed to access the robot on their own times. Additionally, the Aruco object detection node consumed robot resources and required the user to manually start and stop the node.

For future work, computer-vision based object detection with template matching could be implemented where the robot is able to scan the room and find the requested items from any location, not just the inventory table. Currently, object detection relies on Aruco markers, but it is not practical to place markers on every object an elderly individual may need. Aruco marker detection is implemented as video feed from a separate node. We could do Aruco marker detection locally on the robot by taking an image of the objects on the table and extracting positions from the picture. We could stop using aruco markers for detection too by using CV for item matching with object dimensions provided and calculate the pose with respect to the robot for cleaner detection. Another future implementation could incorporate object detection through machine learning models that are able to differentiate objects from one another. The robot should also be able to move autonomously without hard-coded arm and base movement commands to locate and retrieve items. It currently uses distance-time odometry for base movement. Instead, we could use the lidar on the base to position the robot more precisely either by building a map and localizing positions or by taking laser distance measurements to position the robot more accurately near the inventory and target tables. Arm movement could also be improved by using moveit to plan a path from the arm tucked position to the object position. Moveit can be given the position of the table and other objects as well for collision avoidance purposes. Finally, to improve human-robot interaction, face detection could be implemented to ensure the robot looks at the user when listening for commands and overall use the robot's gaze to communicate intent more clearly. We did not have enough time to code the extra credit task, but the code has the ability to move the left arm as well. Lastly, other features could have been added like a happy dance, jokes on command, or integration with ChatGPT.

4 Individual Contributions

We worked collaboratively through various aspects of this project from brainstorming initial ideas to the actual implementation. Aditi specifically worked on the speech pipeline method for human-robot interaction and the Aruco demo for object detection. Specifically, she made the robot introduce itself at the beginning of the demo and had the robot prompt the user for a command using text-to-speech (TTS). Using Google speech recognition, she had TIAGo listen for commands and parsed through user input to find keywords by using natural language processing with spaCy in python. Finally, she mapped requested items to specific responses accounting for accurate responses when error occurs. Also, she attempted to get the Aruco demo to work in simulation, and she did the entire report. Vishnu worked on the navigation pipeline and was able to accurately define velocities and generate twists using the mobile base controller publisher to move the robot to home facing the user, to the inventory table, to the target table, and back to the home position to accept more commands. He got the object detection to work using the Aruco marker publisher, and he generated a simple trajectory with two waypoints to move TIAGo's arms with the arm controller action client to get the robot to detect and grasp items. He also got the robot's head and torso to move. He also modularized the code and heavily debugged both on the robot and in simulation. Ahan worked on commenting the code and getting the arm movement to work in simulation. All team members were involved with debugging the code, especially Vishnu.

5 Citations

[1] Amanda Sharkey and Noel Sharkey. 2010. Granny and the Robots: Ethical Issues in Robot Care for the elderly. Ethics and Information Technology 14, 1 (2010), 27–40. DOI:http://dx.doi.org/10.1007/s10676-010-9234-6

[2] Felix Carros et al. 2020. Exploring human-robot interaction with the elderly. Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (2020). DOI:http://dx.doi.org/10.1145/3313831.3376402

[3] James Wright. 2023. Inside Japan's long experiment in automating elder care. (January 2023). Retrieved May 10, 2023 from https://www.technologyreview.com/2023/01/09/1065135/japan-automating-eldercare-robots/