Research Article

Alexander Wilkinson*, Michael Gonzales, Patrick Hoey, David Kontak, Dian Wang, Noah Torname, Sam Laderoute, Zhao Han, Jordan Allspaw, Robert Platt, and Holly Yanco

Design guidelines for human–robot interaction with assistive robot manipulation systems

https://doi.org/10.1515/pjbr-2021-0023
received September 30, 2019; accepted April 23, 2021

Abstract: The design of user interfaces (UIs) for assistive robot systems can be improved through the use of a set of design guidelines presented in this article. As an example, the article presents two different UI designs for an assistive manipulation robot system. We explore the design considerations from these two contrasting UIs. The first is referred to as the *graphical* user interface (GUI), which the user operates entirely through a touchscreen as a representation of the state of the art. The second is a type of novel UI referred to as the *tangible* user interface (TUI). The TUI makes use of devices in the real world, such as laser pointers and a projector–camera system that enables augmented reality. Each of these interfaces is designed to allow the system to be operated by an untrained user in an open environment such as a grocery store. Our goal is for these guidelines to aid researchers in the design of human–robot interaction for assistive robot systems, particularly when designing multiple interaction methods for direct comparison.

Keywords: design guidelines, human–robot interaction, assistive robotics, graphical user interface, tangible user interface, augmented reality

*** Corresponding author: Alexander Wilkinson,** Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: Alexander_Wilkinson@student.uml.edu
Michael Gonzales: Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: Michael_Gonzales@student.uml.edu
Patrick Hoey: Department of Electrical and Computer Engineering, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: Patrick_Hoey1@student.uml.edu
David Kontak: NERVE Center, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01852, United States, e-mail: david.kontak@comcast.net
Dian Wang: Khoury College of Computer Sciences, Northeastern University, Boston, Massachusetts, 02115, United States, e-mail: wang.dian@husky.neu.edu
Noah Torname: Department of Electrical and Computer Engineering, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: Noah_Torname@student.uml.edu
Sam Laderoute: Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: Samuel_Laderoute@student.uml.edu
Zhao Han: Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: zhan@cs.uml.edu
Jordan Allspaw: Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: jallspaw@cs.uml.edu
Robert Platt: Khoury College of Computer Sciences, Northeastern University, Boston, Massachusetts, 02115, United States, e-mail: rplatt@ccs.neu.edu
Holly Yanco: Department of Computer Science, University of Massachusetts Lowell (UML), Lowell, Massachusetts, 01854, United States, e-mail: holly@cs.uml.edu

1 Introduction

Many different types of user interfaces (UIs) exist for assistive robots that are designed to help people with disabilities with activities of daily living (ADLs) while maintaining their sense of independence. For example, Graf et al. [1] presented a touchscreen interface for object selection with an assistive robot manipulator. For direct interaction with the world, Kemp et al. [2] presented a UI that uses a laser interface to select objects, and Gelšvartas et al. [3] presented a projection mapping system that can be used to highlight specific objects in the real world for selection.

For our research on the development of different types of UIs for assistive robotics, we have had to solve the problem of how to directly compare the usability of UI designs despite the modes of interaction between the user and the system being entirely different. Without the ability for a direct comparison, how can we learn the best methods for human–robot interaction (HRI) with assistive robot systems?

In this article, we present guidelines for designing UIs for assistive robots that ensure that any two UIs are directly comparable. We discuss a set of core design prin-

ciples that we have identified as being important for UIs for assistive robots, and the importance of guaranteeing that the underlying process for operating the system is consistent through the use of a state diagram that describes the loop of interactions between human and robot. Finally, we discuss the implementation of these design guidelines for two UIs we have developed for the assistive robotic scooter system presented in [4].

2 Related works

When designing our guidelines, we researched different examples of assistive robotics and their UI in order to determine the design guidelines for our guidelines. A few notable assistive robot systems are Martens et al.'s semi-autonomous robotic wheelchair [5], Grice and Kemp's tele-operated PR2 controlled through a web browser [6], Achic et al.'s hybrid brain–computer interface for assistive wheelchairs [7], and Nicholsen's assistive robot Bestic, designed to help those with motor disabilities [8].

In regard to UI designs, we examined UIs that utilized laser pointing devices and touchscreens along with general pick-and-place interfaces to reference when designing our two test UIs. Some notable touchscreen interfaces include Graf et al.'s touchscreen interface for object selection for general home assistance robots [1], Cofre et al.'s smart home interface designed for people with motor disabilities [9], and Choi et al.'s object selection touchscreen interface designed for amyotrophic lateral sclerosis (ALS) patients [10]. Notable UIs that utilize a laser pointing device include Kemp et al.'s interface that uses a laser pointer to select items [2], Gualtieri et al.'s laser pointer selection interfaces for robotic scooter systems [11], and, again, Choi et al.'s interface for ALS patients suing an ear-mounted laser pointer [10]. In the pick-and-place domain, Quintero et al. used gestures to control the arm [12], Rouhollahi et al.'s UI was written using the Qt toolkit [13], and Chacko and Kapila's augmented reality interface used smartphones as input devices [14].

Expanding upon Gualtieri et al.'s previously mentioned laser pointer selection interface [11], Wang et al. presented an interface comprised of a laser pointer, a screen, and a projector–camera system that enables augmented reality [4]. This interface informed the user of the location of a pick and place procedure by projecting light with a projector. The system used to demonstrate our guidelines in this article expands upon Wang et al.'s improvements.

3 Guidelines for UI development and evaluation for assistive robotics (GUIDE-AR)

In order to create UIs that are as easy as possible to use by people with disabilities and to be able to directly compare different UIs, we created Guidelines for User Interface Development and Evaluation for Assistive Robotics (GUIDE-AR). GUIDE-AR's design principles are those that we deem to be essential for UIs intended for assistive robots. Along with these principles, GUIDE-AR includes the use of a state diagram that represents the process of interactions between the user and the system. This state diagram allows for the direct comparison of different types of UIs and access methods by representing the HRI in a procedural manner. This state diagram allows us to streamline the process of comparing interfaces while also leading to the ability to eliminate unnecessary complexity through the examination of the steps in the HRI process.

3.1 Design principles

When envisioning the design principles for UIs for assistive technology, our high-level goal was the development of a system that would be usable by a novel user in an open environment. We have defined our design principles based upon this goal; the UI of such a system should be:

1. **Understandable.** Because we believe systems should be designed for an untrained user, the UI must offer the user enough feedback so that a novel user can operate the system with little to no instruction. The UI should clearly communicate its current state so that it is never unclear what the system is doing.

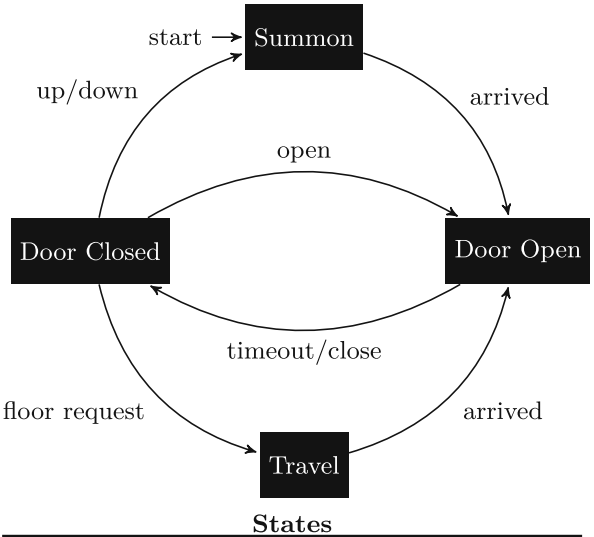
   For systems that are expected to be used long term, allowing the user to select the amount of communication provided by the system will increase its usability.
2. **Reliable.** The real world can be messy and unpredictable, so it is unavoidable that the user or the system might make errors. To mitigate this problem, the system should have robust error handling in the case of either a system failure (i.e., the robot fails to plan a grasp to the selected object) or a user error (i.e., the user selects the wrong object).
3. **Accessible.** The system should be able to be used by people with a wide variety of ability levels, independent of their physical abilities, cognitive abilities, or level of experience with robots. This requirement for accessibility includes the need to design the UI to have

the ability to be adapted to a wide variety of access methods (e.g., switches, sip and puff, joysticks) that are utilized for controlling powered wheelchairs and for using computers.

3.2 Describing the human–robot loop

The interaction process between the user and the robot system can be represented with a state diagram. As an example, Figure 1 shows the state machine for an elevator, an example that should be familiar to all readers. Labels on the nodes (i.e., the black squares in Figure 1) represent states, and labels on the edges (i.e., the arrows) represent changes in state as a result of user action.

The process of creating this state diagram can allow for the process to be streamlined if redundancies or inefficiencies are discovered. Additionally, if we design different UIs that each conform to the same state diagram for the interaction process, we can enforce that they are directly comparable when researching their performance. In this case, the only independent variable in the comparison will be the mode of interaction with the UI. For the elevator example, this type of UI design would mean that you could directly evaluate the performance of a less conventional assistive interface (i.e., voice controls, gestures, etc.) against a more traditional screen- or button-based UI.



States

Summon	The elevator is traveling to the requested floor
Close	The elevator doors are closed
Open	The elevator doors are open
Travel	The elevator is traveling to the requested floor

Figure 1: Example of a state diagram for an elevator UI, describing its UI.

4 Example use of GUIDE-AR

As an example use of the GUIDE-AR guidelines, we present the implementation of two UIs for an assistive robotic manipulator system mounted to a mobility scooter that we have developed in our previous work [4]. The system consists of a mobility scooter with a robotic manipulator that is capable of picking and placing objects. The intended use case is for people with limited mobility to be able to use the robot arm to pick up arbitrary (i.e., unmodeled) items from an open environment such as a grocery store with little to no training.

We have developed two UIs for this system: the *graphical* user interface (GUI), which allows the user to interact with the system entirely through a touchscreen, and the *tangible* user interface (TUI), which makes use of the real world as an interface element, using laser pointers and a projector–camera system to enable augmented reality.

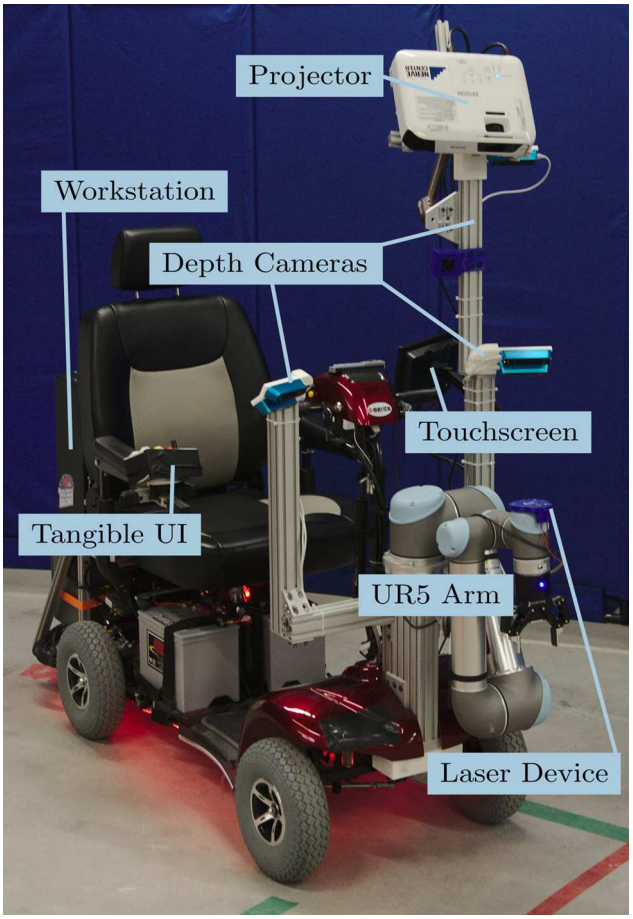
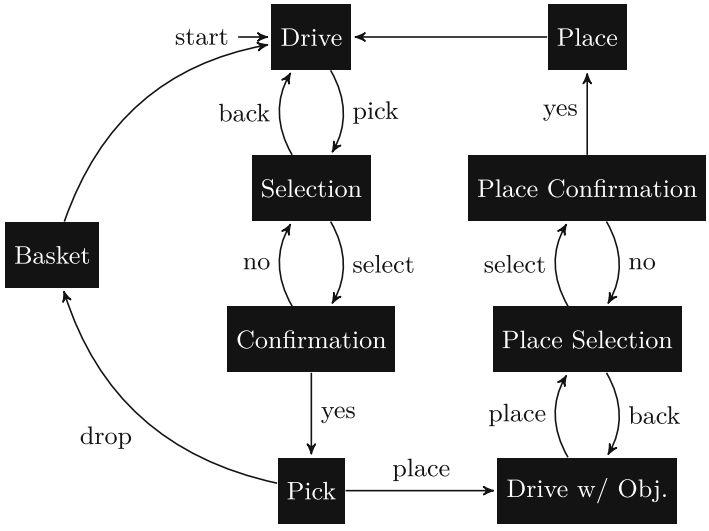


Figure 2: An image of the assistive robot system, with a Universal Robots UR5 robot arm outfitted with a Robotiq 2F-85 gripper mounted on a Merits Pioneer 10 mobility scooter.



States	
Drive	Arm is stowed and the user is free to drive the scooter
Selection	The system is waiting for the user to select an object
Confirmation	The user must confirm that the displayed object is the intended object
Pick	Robot picks object
Basket	Robot drops object in basket
Drive w/ Obj	Arm is stowed with the object held and the user is free to drive the scooter
Place Selection	The user must select a location to place the object
Place Confirmation	The user must confirm that the displayed location is correct
Place	Robot drops object in selected location

Figure 3: State diagram for the robot task. Users of the system select an object for the robot arm to pick up, then decide whether to place the object in a new location or have it dropped into a basket on the scooter.

4.1 Assistive robot system overview

The assistive robot manipulator system consists of a Universal Robots UR5 robotic arm equipped with a Robotiq 2F-85 electric parallel gripper, mounted on a Merits Pioneer 10 mobility scooter, as shown in Figure 2. Five Occipital Structure depth cameras provide perception for the system. A workstation is mounted on the rear of the scooter and is connected to the robot, sensors, and the hardware for the two UIs, the GUI and TUI. The workstation has an Intel Core i7-9700K CPU, 32 GB of memory, and an Nvidia GTX 1080 Ti GPU. The GUI uses a 10-inch touchscreen mounted on one of the armrests in reach of the user. The TUI consists of a box mounted on one of the armrests with six dimmable buttons and a joystick, along with a laser pointing device mounted to the end effector for object selection, and a projector aimed toward the workspace of the arm that enables the system to highlight objects in the world when combined with the depth cameras.

Each subsystem of the software is implemented as a ROS node running on the workstation. The system uses GPD for grasp pose detection, and OpenRAVE and TrajOpt for collision avoidance and motion planning [15–17].

4.2 HRI state diagram

The desired use of our system is picking items in an open environment such as a grocery store with little to no training. The state machine we have designed allows the user to select an object, pick it automatically, and then decide to either drop it in a basket mounted to the scooter or place it in a different location in the world.

Figure 3 is a state diagram that uses actions and resultant states to represent the task being performed by the robot, independent of the UI being utilized. The labels on the nodes (i.e., black rectangles) represent the state of the robot, and the labels on the edges (i.e., arrows) represent the option selected on the touchscreen, the button pressed, or other action taken by the user.

4.3 Design for different interaction methods

As previously mentioned, we have created two different UIs to operate the system, the GUI and TUI. The GUI was designed with the intent to keep the user's focus on a touch screen using an application. In contrast, the TUI

Table 1: Description of how both UIs comply with our design principles

	Graphical	Tangible
Understandable	Descriptive text prompts. Only actions that can be taken from the current state are shown to the user	Descriptive voice prompts. Buttons light up to indicate what actions can be taken from the current state. Eliminates the need for context switch between UI and world
Reliable	User must confirm that the correct object is selected. Error handling for grasp failure	User must confirm that the correct object is selected. Error handling for grasp failure
Accessible	On-screen buttons are large and clearly labelled. Position of touchscreen is adjustable	Laser controlled with joystick instead of handheld, reducing the amount of fine motor skills needed. Position of button box is adjustable

has been designed to keep the user's focus on the world using augmented reality with a custom designed joystick and button box to support the interface. Both UIs comply with the design principles we have outlined, as shown in Table 1.

4.3.1 Touchscreen-based GUI

We consider the GUI to be representative of the state of the art. We designed it to be a suitable control for making comparisons against, under the assumption that no matter how the GUI is implemented, it still requires the user to switch focus from the real world to the virtual representation of the world. This context switch occurs for any type of UI that is based primarily on a screen that provides a representation of the world. We believe that this presents an additional cognitive load on the user, so we designed the GUI as a baseline to study this effect (Figure 4).

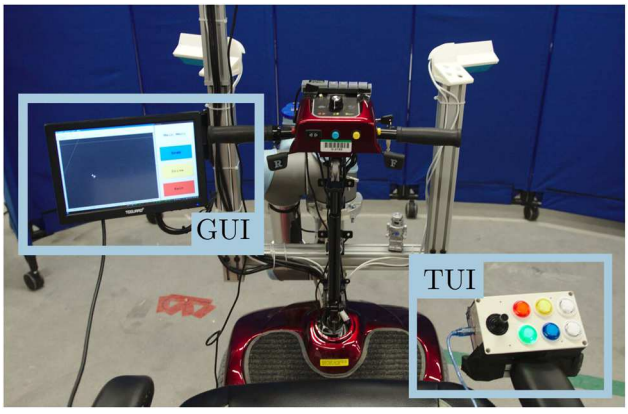


Figure 4: First person point of view from the assistive robot system showing the touchscreen for control of the GUI, and button panel for control of the TUI.

4.3.1.1 User Experience of the GUI

At the start of this project, the assistive manipulation scooter had a single interface, which used a screen, an external four button pad, and a laser pointing device to operate the system. It was also designed to be used by the researchers and designers of the system. Thus, extraneous information about grasping methods and the like would be displayed or prompted while operating the system.

We decided to consolidate the user's area of interaction and the number of steps needed to operate the system to a minimum to lower its cognitive load on the user. This led to the design of the GUI, which allows the user to operate the system entirely through a touch screen. This new GUI removed the need for a laser pointing device and use of the button pad. Selection of objects and operating the pick-and-place features are now all done through one application on a touch screen.

When we were designing the GUI, we wanted it to be accessible to as large of a population as possible. So we took color, cognitive issues, and fine motor issues into consideration for usability. There are at most three buttons on the screen at one time, each colored blue, yellow, or red, as shown in Figure 5. These particular colors were chosen since red, yellow, and blue are less likely to be confused by those with red-green colorblindness, which is the most common form of colorblindness according to the National Eye Institute [18]. We also made the buttons large to account for those with limited fine motor ability so that they would be easily selectable.

The GUI includes a tap to zoom feature, which also assists in making the GUI easy to use for people with motor disabilities by allowing the user's selection area to be smaller (i.e., not requiring large movements of a person's hand or arm) and making the items to be selected larger (i.e., not requiring fine motor control in order to be able to select a small region on the touchscreen). This accessible

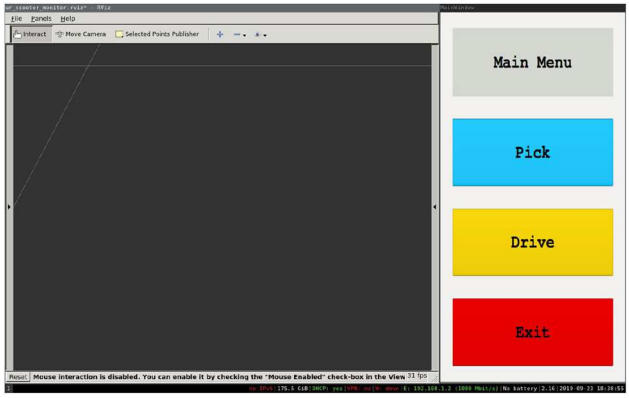


Figure 5: Main Menu of the GUI. From here the user can either drive the scooter or begin a pick process.

design makes it easier for the user to select the desired object.

As mentioned in the system overview, the touch screen is placed along the armrest on side of the user's dominant hand so that it is in the easiest place for the user to access. There are other options that could be considered for accessibility, such as creating a key guard for the screen that provides guidance to the regions of the screen that are supposed to be touched.

4.3.1.2 Building the GUI

To build the GUI, we implemented an RViz window to show the world to the user with a custom made Qt application to operate the pick-and-place system. RViz shows what the five depth sensors are publishing via ROS and displays it to the user as a colorized PointCloud2. To select an object or placement location, the user can tap on the screen to zoom in on the desired object for selection or the desired location's general area for placing an object, then select it with a second tap. The rest of the process, including confirming objects, choosing placement options, prompts, and the like, is in a custom made Qt application. Qt is a commonly used GUI toolkit that runs with C++ or Python, and as most of the system runs on Python, we used Qt to implement the GUI. The final product of the Qt application, tap to zoom and item selection can be seen in Figures 6 and 7.

As previously mentioned, the representation of the world shown to the user (as shown in Figure 6) is stitched from the five depth cameras that are also used for grasp planning. These sensors do not have integrated RGB cameras, so another RGB camera pointed toward the workspace is used to color the pointcloud. First, the RGB image is registered to each of the depth images given the pose of the RGB camera and the depth cameras. Then, an array of points with fields for the position and color is created from the registered depth and RGB images. This array is then published as a ROS PointCloud2 message so that it can be visualized in RViz.

4.3.2 World-based TUI

The context switch discussed previously, in which the user must switch focus between the world and the screen, can be eliminated entirely by building the UI in the world itself. We believe that this will reduce the cognitive load on the user. Toward this goal, we have designed a TUI, which uses a novel combination of technologies that allows the object selection task outlined in Figure 3 to be carried out while keeping all feedback to the user based in the real world (Figure 4).

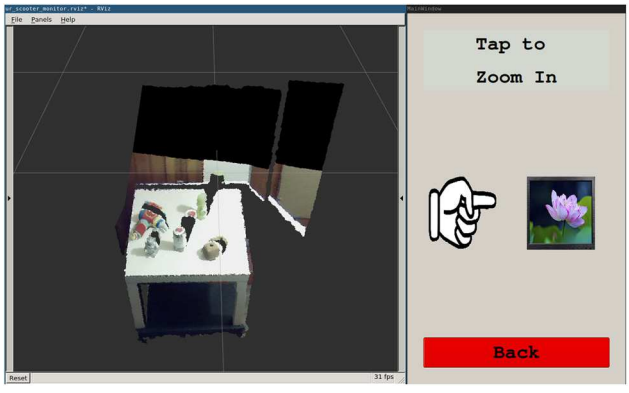


Figure 6: Interactive pointcloud representing the real world. The user can tap on an area to zoom in, and then tap on an object to select it.

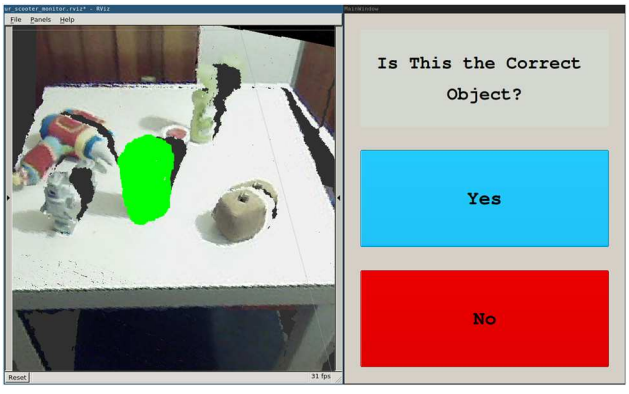


Figure 7: The GUI is showing the selected object in green, and asking the user to confirm that the correct object is selected. This is the *Confirmation* state in Figure 3.

4.3.2.1 Design of the TUI

We designed the TUI to give the user the ability to interact directly with the world around them. The TUI utilizes a laser to select objects with a control panel mounted on the arm of the scooter itself, as shown in Figure 8, alongside multiple methods of user feedback. Feedback is provided using a projector to highlight objects and locations directly in the world, as well as audio to provide status information about the system.

The control panel contains a joystick, a switch, and five buttons that have LEDs within them. Each button can be lit individually. These buttons are utilized to indicate the system's state. A button that is lit a solid color indicates the current system state, and a button that is flashing indicates to the user that they must make a choice (i.e., the *yes* and *no* buttons are flashing when asked if this is the correct object). Outside of these two conditions, the buttons remain unlit.

4.3.2.2 Creating the TUI

The joystick and button box for the TUI are controlled via an Arduino Mega 2560 development board. This board contains an ATmega2560 processor which is the heart of the system. Communication between the microcontroller and the system is done over USB via the ROSSerial Arduino Library. This library allows you to create a ROS node on the Arduino itself, and it allows this node to communicate with the system via a serial node.

The microcontroller communicates the state of the control panel when the system is booted. It sends messages containing the state of each button and the state of the joystick to the system. The system controls the TUI operation state based upon these inputs and publishes messages containing any required updates. The microcontroller processes these messages and will update any of the external circuitry based upon its content (i.e., flash the *yes* and *no* LEDs, enable the lasers, etc.) The computer processes the state of the system and does the computationally intensive tasks. The state machine follows the principles laid out in Figure 3 and calls the proper back-end functionality to allow system operation.

An on-board voice prompting system also provides feedback to the user during their operation of the system via the TUI. These voice prompts are run on a separate thread, implemented via Google's Text-to-Speech API. These voice prompts indicate to the user what state the system is in, and what they must do to utilize the system. They allow the user to be aware of the current state of the system. This is important for cases when the system is stitching pointclouds or computing grasps, as the silence and stillness of the system may worry the user that something is wrong. With a less technically inclined user in mind, these prompts include indications such as the colors of the specific buttons to make the choices as clear as possible.

Item selection is done via the joystick contained within the control panel and a pair of lasers attached to the end effector of the UR5 (as shown in Figure 9). Once the user selects to pick up an object and the system com-

Figure 8: Joystick and panel of buttons used to control TUI. The buttons can light up or flash to indicate actions that the user can take.

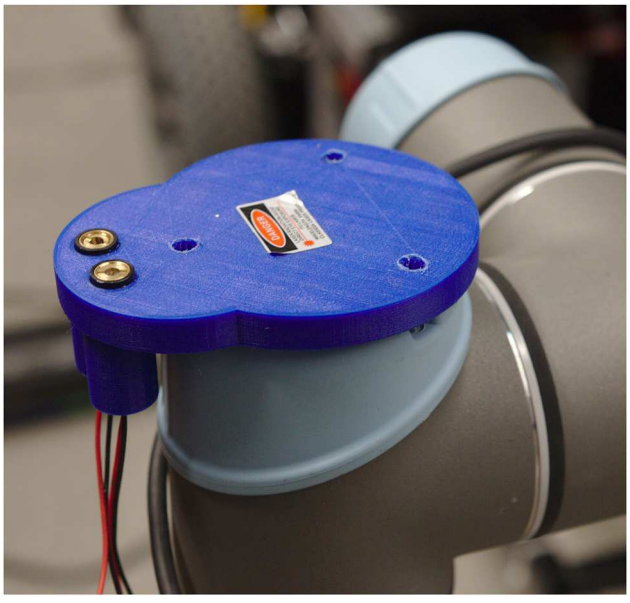


Figure 9: Laser device mounted to the end effector of the UR5. The user can use a joystick to move the lasers to point at objects.

pletes its scan of the environment, the UR5 moves to a position to allow object selection with the laser. The user controls the angle of the arm via the joystick and, in turn, they are able to place both dots of the laser on whatever object they intend to pick up. The system detects the dots shown by both lasers and proceeds to isolate the object. Upon isolating the object, it is highlighted via the mounted projector to allow the user to confirm that the proper item has been selected.

A dynamic spatial augmented reality (DSAR) system is used to highlight objects and locations while operating the TUI. The projector provides feedback to the user about the grasp radius of the arm and object segmentation. Before the user selects an object with the laser, the projector will highlight all surfaces within the robot's reach one color, and all surfaces that are outside the robot's reach another color, as shown in Figure 11. When an object is selected, the projector will highlight the segmented object so that the user can confirm with a button that the system has identified the correct object to pick, as shown in Figure 10. When a location for a place is selected, the location of the place is highlighted as well.

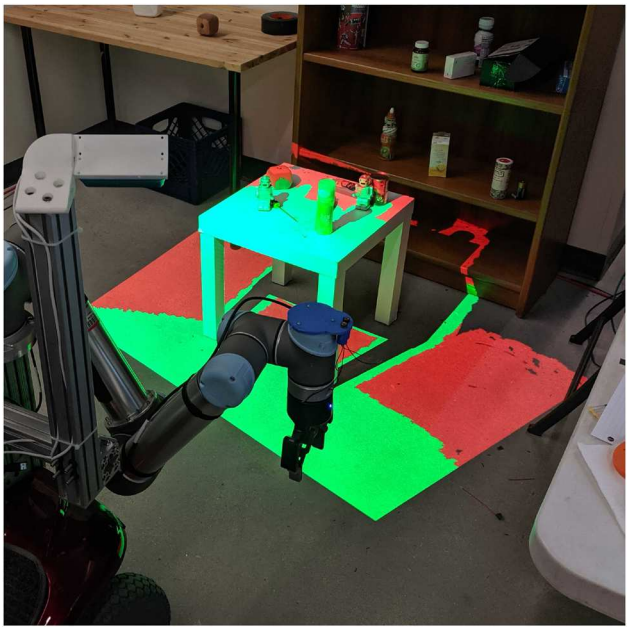


Figure 11: Projection mapping showing grasp radius. Objects that can be grasped are highlighted in green, and objects that are out of reach of the arm are highlighted in red.

4.3.2.3 Augmented reality

The DSAR system highlights objects and places in the world by projecting a visualization of the pointcloud acquired by the depth cameras. This visualization is generated by a virtual camera placed into the 3D environment along with the pointcloud. The virtual camera is then given the same intrinsics and extrinsics as the projector so that when the image from the virtual camera is projected through the projector, each point from the pointcloud is rendered over its location in the real world.

Figure 10: Projector being used to highlight an object in the world. Here it is shown highlighting an object that has been selected by the user.

We determined the intrinsics of the projector lens by mounting the projector a fixed distance from a board orthogonal to the optical axis, projecting the image, and measuring the locations of the corners as described in our previous work [4]. Our methods for projection mapping to highlight objects in the environment are similar to those presented by Gelšvartas, Simutis, and MaskeliÅńnas, except that our system renders a pointcloud instead of a mesh [3].

Let W and H be the width and height of the image respectively, w and h be the width and height of the image in pixels, respectively, X and Y be the distances from the principal point (the intersection of the optical axis and the image) to the origin (top-left) point of the image on the image plane in their respective axes, and Z be the distance from the projector to the image plane. We can determine the focal length f of the projector in pixels using the projective equation: $f_x = f_y = w\frac{Z}{W}$. The principal point (c_x, c_y) is then calculated in pixels: $c_x = w\frac{X}{W}$;

$c_y = h\frac{Y}{H}$. The projector was then modeled as a pinhole camera with an intrinsic camera matrix K:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (1)$$

This camera matrix K is then used as the intrinsics of the virtual camera in the RViz environment so that the camera and the projector have the same intrinsics and extrinsics, allowing the projector to function as a "backwards camera" and all points in the pointcloud to be projected back onto their real-world location.

5 Discussion and future work

We have created guidelines to assist the creation of UIs for assistive robotics entitled GUIDE-AR. These guidelines include design principles and state diagrams to visualize and ensure that each UI is indistinct of each other besides visualization. The GUIDE-AR is the skeleton of all UIs created with it. To demonstrate this, we created a state diagram for pick-and-place procedures for assistive robots and created two UIs with it, the GUI and TUI. Although they are drastically different in interfacing, the skeleton of their code and execution are exactly the same. This is done so that testing different UIs, whether in house or user studies, has the least amount of extraneous variables as possible.

There are several improvements that can be made to the design of the GUI. Currently, the GUI shows two separate windows: the RViz instance for displaying pointclouds, and the Qt application through which buttons and prompts are displayed. We would like to design the GUI so that both windows are integrated into one. This combination would allow us to hide the pointcloud window when it is not necessary, further simplifying the user experience.

We are continuing to improve aspects of the TUI, including improving the DSAR system with a smaller projector with a wider field of view. The registration of the projector can be improved by projecting a graycode pattern over a chessboard and using a registered camera to compute homography. We are also working to integrate the TUI with a Microsoft HoloLens as an alternative way of showing what objects the arm can grab. Along with this, a small prompt will be displayed on the HoloLens to help communicate what the system is doing.

Our work to date has established guidelines for researching different modes of user interaction for our system and other assistive robot systems. In order to determine the best interface design – or to learn which elements of each interface design should be combined into a new interface for controlling the scooter-mounted robot arm – we plan to conduct a user study comparing the two different types of UIs we have designed. The target population of participants to be recruited for our study are people who are at least 65 years old and who are able to get in and out of the seat on the scooter independently.

We plan to use our guidelines to help us develop novel types of UIs for the mobility scooter or other systems. Examples of possible UI designs would be implementing speech-recognition, eye tracking, or sip-and-puff control. Although our focus has been on assistive robots, we believe that the GUIDE-AR guidelines could also be applied to other applications of HRI, particularly when direct comparisons of interfaces are desired.

Funding information: This work has been supported in part by the National Science Foundation (IIS-1426968, IIS-1427081, IIS-1724191, IIS-1724257, IIS-1763469), NASA (NNX16AC48A, NNX13AQ85G), ONR (N000141410047), Amazon through an ARA to Platt, and Google through an FRA to Platt.

Author contributions: AW, MG, PH, NT, and SL developed the GUIDE-AR guidelines as well as the GUI and TUI under the supervision of HY. Specifically, AW focused on the DSAR system, MG focused on development of the GUI, and PH focused on development of the TUI, while NT and SL provided additional system improvements and researched new methods for augmented reality and object selection. DW developed the assistive robotic system that was used as a platform under the supervision of RP as described in ref. [4]. DK provided feedback and guidance on the design of the system. ZH and JA assisted with writing and identifying related works. AW prepared the manuscript with contributions from all coauthors. The authors applied the FLAE approach for the sequence of authors. All authors have accepted responsibility for the entire content of this manuscript and approved its submission.

Conflict of interest: Authors state no conflict of interest.

Data availability statement: Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

References

[1] B. Graf, M. Hans, and R. D. Schraft, "Care-O-bot II—development of a next generation robotic home assistant," *Autonom. Robot.*, vol. 16, no. 2, pp. 193–205, 2004.

[2] C. C. Kemp, C. D. Anderson, H. Nguyen, A. J. Trevor, and Z. Xu, "A point-and-click interface for the real world: laser designation of objects for mobile manipulation," in *2008 3rd ACM/IEEE International Conference on Human-Robot Interaction* (*HRI*), IEEE, 2008, pp. 241–248.

[3] J. Gelšvartas, R. Simutis, and R. Maskeliūnas, "Projection mapping user interface for disabled people," *J. Healthcare Eng.*, vol. 2018, 6916204, 2018.

[4] D. Wang, C. Kohler, A. T. Pas, A. Wilkinson, M. Liu, H. Yanco, et al., "Towards assistive robotic pick and place in open world environments," in *Proceedings of the International Symposium on Robotics Research* (*ISRR*), October 2019, arXiv: http://arXiv.org/abs/arXiv:1809.09541.

[5] C. Martens, N. Ruchel, O. Lang, O. Ivlev, and A. Graser, "A friend for assisting handicapped people," *IEEE Robot. Automat. Magazine*, vol. 8, no. 1, pp. 57–65, 2001.

[6] P. M. Grice and C. C. Kemp, "Assistive mobile manipulation: Designing for operators with motor impairments," in *RSS 2016 Workshop on Socially and Physically Assistive Robotics for Humanity*, 2016.

[7] F. Achic, J. Montero, C. Penaloza, and F. Cuellar, "Hybrid BCI system to operate an electric wheelchair and a robotic arm for navigation and manipulation tasks," in *2016 IEEE Workshop on Advanced Robotics and Its Social Impacts* (*ARSO*), IEEE, 2016, pp. 249–254.

[8] N. C. M. Nickelsen, "Imagining and tinkering with assistive robotics in care for the disabled," *Paladyn, J. Behav. Robot.*, vol. 10, no. 1, pp. 128–139, 2019.

[9] J. P. Cofre, G. Moraga, C. Rusu, I. Mercado, R. Inostroza, and C. Jimenez, "Developing a touchscreen-based domotic tool for users with motor disabilities," in *2012 Ninth International Conference on Information Technology – New Generations*, 2012, pp. 696–701.

[10] Y. S. Choi, C. D. Anderson, J. D. Glass, and C. C. Kemp, "Laser pointers and a touch screen: intuitive interfaces for autonomous mobile manipulation for the motor impaired," in *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, ACM, 2008, pp. 225–232.

[11] M. Gualtieri, J. Kuczynski, A. M. Shultz, A. Ten Pas, R. Platt, and H. Yanco, "Open world assistive grasping using laser selection," in *2017 IEEE International Conference on Robotics and Automation* (*ICRA*), IEEE, 2017, pp. 4052–4057.

[12] C. P. Quintero, R. Tatsambon, M. Gridseth, and M. Jägersand, "Visual pointing gestures for bi-directional human robot interaction in a pick-and-place task," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication* (*RO-MAN*), 2015, pp. 349–354.

[13] A. Rouhollahi, M. Azmoun, and M. T. Masouleh, "Experimental study on the visual servoing of a 4-DOF parallel robot for pick-and-place purpose," in *2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems* (*CFIS*), 2018, pp. 27–30.

[14] S. M. Chacko and V. Kapila, "An augmented reality interface for human–robot interaction in unconstrained environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems* (*IROS*), 2019, pp. 3222–3228.

[15] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, et al., "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, Kobe, Japan, 2009, p. 5.

[16] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *Int. J. Robot. Res.*, vol. 36, no. 13–14, pp. 1455–1473, 2017.

[17] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt, "High precision grasp pose detection in dense clutter," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems* (*IROS*), Oct 2016, pp. 598–605.

[18] National Eye Institute, "Types of color blindness," 2019, https://www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/color-blindness/types-color-blindness [Accessed: May 12, 2021].