Accessible Interfaces for Robot Assistants

Daniel A. Lazewatsky[1] and William D. Smart[2]

Abstract— **Currently, high-level task control of robots is generally performed by using a graphical interface on a desktop or laptop computer. This type of mediated interaction is not natural, and can be problematic and cumbersome for persons with certain types of motor disabilities, and for people interacting with the robot when there are no computer displays present. In this work, we present a framework which enables the removal of such obvious intermediary devices and allows users to assign tasks to robots using interfaces embedded directly in the world, by projecting these interfaces directly onto surfaces and objects. We describe the implementation of the projected interface framework, and give several examples of tasks which can be performed with such an interface.**

I. Introduction

The idea of a robot assistant has been around for a long time. However, recent advances in perception, manipulation, and autonomy are bringing this vision closer to reality. There are already many tasks that robots can perform autonomously, from picking up and delivering household objects [1], to opening doors, drawers, cabinets, and containers [2], to cooking complete meals [3]. These advances have the potential to make robot assistants truly useful, especially for persons who cannot perform some of these actions for themselves. They also enable the sorts of tasks often performed by home automation to be used in uninstrumented environments such as public spaces. However, the question of how best to direct the robot to perform these tasks is still an open one.

Interfaces to assign tasks to these mobile manipulation robots typically involve either physical gestures interpreted by the robot (for example [4]) or a custom-designed graphical interface displayed on a computer screen (for example [5], [6]). While these approaches have been shown to work well in a number of systems, they make some implicit assumptions about the person directing the robot. To use physical gestures, a person must be able to move their arms. To use a graphical interface, a person must have a computer in front of them and be able to use it.

Both of these assumptions limit the usefulness of a robot assistant for persons with severe motor disabilities who, otherwise, might most benefit from such a system. If a person is unable to effectively move their arms, they cannot use a gestural interface. While many persons with physical disabilities can use a computer through the use of alternative input devices, a human assistant often has to bring this computer to them. This raises the question: if a human assistant has to bring the computer, can't they also help with the task, rather than having the robot do it? Even for persons with normal physical abilities, requiring a computer to interact with the robot is a limitation we would like to remove.

We describe a system that allows a person with limited physical abilities to assign tasks to a complex mobile manipulation robot in a home setting. For the work reported in this paper, we assume that the person has good control of their head position. We present a system for use with a variety of household tasks that uses only the motion of a user's head and a single click to interact with task-specific interfaces projected into the world or onto relevant objects. To do so, the system *a*) uses the robot's on-board sensors to estimate the head pose of the person; *b*) combines this with information from the world models maintained by the robot to generate context-sensitive interface elements; and *c*) projects these interface elements directly onto the world, allowing the person to interact with them using a cursor controlled by the user's head motion.

II. Related Work

Before going on to describe our system in detail, we first discuss some of the related work in interfaces and robot control.

Graphical user interfaces typically rely on some sort of pointing device [7]. The mouse, or some similar device, is by far the most common device although pen-based devices, first seen in Sutherland's Sketchpad [8] are relatively widespread. However, mouse and pen interfaces are often difficult for persons with motor disabilities to use. Alternative input devices such as eye tracking [9] or other "mouse replacement" devices such as TrackerPro® [10] have been developed to emulate mouse input to enable use of standard graphical interfaces.

There are a few examples of systems that project interface elements into the world. SixthSense [11] uses a wearable device incorporating a computer, projector, and camera, to project interactive interfaces onto the world. These interfaces can be informational (projecting departure gate information onto an airline ticket), or functional (a working calculator projected onto one's hand). While SixthSense is very similar in spirit to the work reported in this paper, it lacks the rich sensor information and world models available to our robot and the ability to move around (and effect changes in) the world independent of its wearer.

PICOntrol [12] uses a handheld projector and sensor package along with small sensor units on devices to allow users to send simple commands to devices. Cao et al. use a

Daniel A. Lazewatsky and William D. Smart are with the School of Mechanical, Industrial, and Manufacturing Engineering, Oregon State University, Corvallis, OR 97331, USA

This work was funded in part by a research gift from Willow Garage.

[1]lazewatd@engr.oregonstate.edu
[2]bill.smart@oregonstate.edu

handheld projector to enable users to explore virtual spaces, and interact with virtual objects using a pen, and movement of the projector itself [13].

Projected interfaces, using static projectors, have been used with fixed industrial robot arms [14]. Again, this is similar in spirit to our system, although it is in a fixed location, and uses a custom-designed handheld interaction device for user input. Sasai and colleagues [15] have demonstrated a system that projects a control interface for a simple mobile robot onto the floor that allows users to direct the robot using foot-taps on the interface. This work is similar to ours, although it is designed for single type of interaction (direction-giving), and the projection assumes a clear, open floor in a known position with respect to the robot. Sato and Sakane use a fixed projector and robot arm to project onto a workspace and perform simple pick and place tasks [16].

Gesture interfaces, tracing their history back to Bolt's Put-that-There system [17] allow a user to use pointing gestures to interact with an interface. Some of these systems, such as XWand [18] use custom interaction devices, while others interpret natural human body gestures.

Gesture-based interfaces on robots have enjoyed less success than those aimed at interacting with fixed displays. Kemp's Clickable World [19] is a notable exception that uses a standard laser pointer to designate objects for a mobile manipulation robot to fetch. Looper and colleagues [20] describe a system that interprets and responds to a limited set of stylized human gestures (military hand signals).

III. Implementation

Our system is currently implemented on a PR2 mobile manipulation robot using the ROS software infrastructure [21]. ROS is free and open-source, and provides a simple and standard way of interacting with sensors and actuators. The system comprises three main components; a model of the world, maintained by the robot; a pointing input, generated by tracking the user's head pose; and a projected interface, that allows the user to task the robot. We describe each of these four components in turn.

A. The Robot

For all interactions requiring a robot, we use a Willow Garage PR2 robot. PR2 has a quasi-holonomic base, two 7 degree-of-freedom arms, and a movable head containing a variety of sensors, including two pairs of stereo cameras, a textured light projector, a high-resolution camera, and a Microsoft Kinect. Additionally, PR2 has a planar laser rangefinder on the base, and another planar laser rangefinder mounted on a tilting platform on the torso which can create 3d world models.

B. The World Model

The robot builds and maintains a 3d model of the world with its sensors. For the work reported here, however, we only use part of this model. We extract the plane, corresponding to the surface onto which the robot will project the interface. This plane is represented by its normal vector, and a point on the plane, in the robot's coordinate frame. Using a simple parameterized model for the plane allows us to perform fast intersection calculations to determine where the user is looking. It would be equally easy, however, to use a more complex, non-planar world model generated by the robot, such as a 3d mesh; the only difference would be in the computational cost of the intersection calculations.

The planar model parameters can be estimated in two ways. We can add markers, in the form of augmented reality (AR) tags [22] to the relevant surfaces, and use a monocular camera (such as a webcam) to determine the 3d location and orientation of the tag (and, hence, the surface). We can also use the more advanced sensors mounted on the robot, which generate clouds of 3d points corresponding to objects in the world. A planar model can then be fit to the point cloud using the Random sample consensus (RANSAC) algorithm [23]. This algorithm works by successively selecting a random subset of the data as inliers, and testing how well those data fit the given (planar) model. Once a set of inliers has been chosen, the algorithm then estimates the model parameters using those inliers.

C. The Pointing Input

The system incorporates user input in the form of a 3d vector that "points" at objects in the world. The intersection of this vector and the world model allows us to determine the 3d point in the world that the user is attending to. Although this vector can be estimated from a number of input sources, for the work reported in this paper, we use an estimate of the user's head pose, both position and orientation, for the pointing input.

When using a planar world model, the point can be found using the simple plane-ray intersection calculation. This is only valid if the ray is already known to intersect the plane somewhere, and is not contained within (parallel to) the plane. The first condition is satisfied by assuming the projection surface is an infinite plane. The second property holds because the camera used to track the user is always pointing approximately away from the projection surface and can only track faces from a frontal view.

When using a point cloud representation of the world, the point is found by intersecting the ray with the point cloud. This can be performed efficiently using an octree representation of the point cloud, which enables expected $O(\log n)$ ray tracing operations. When ray tracing, we can return the intersected point closest to the user because any points father away would be occluded from the user's view. However, this intersection calculation is still slower than the constant-time plane-ray calculation, and scales (albeit logarithmically) with the size of the world model.

1) Head Pose Estimation

In the current system, head pose estimation is performed in real time using depth data collected from a Microsoft Kinect sensor. The estimation is performed using the system described by Fanelli *et al.* [24]. This technique takes noisy depth data and produces a 6 degree-of-freedom pose estimate containing the 3d position of the head as well as the head's

orientation, an example of which is shown in figure 1. Although we use the Kinect sensor for the work reported here, any source of 3d point data would work equally well.

Fig. 1. A point cloud view of a user showing the user's head pose estimate as a vector.

This estimate is quite noisy. With the user at approximately 1m from the Kinect, the standard deviation in the roll, pitch and yaw angles was found to be 0.62rad, 0.12rad, and 1.37rad respectively. At 2m from the projection surface, this translates to the cursor from a stationary user being within a circle of diameter approximately 9.521cm with 95% confidence. This problem only gets worse as the distance to the projection surface increases, or the obliqueness of the angle increases.

We have previously evaluated the Kinect as a pointing device, and found that despite the noise, novice users are able to effectively use it in object designation tasks [25].

2) Mouse Clicks

Our system relies on the user being able to perform actions analogous to mouse clicks. This can be done with a traditional computer mouse, if the user is physically able to operate one well enough to simply click one of the buttons, even if they cannot move the mouse on a surface. This is sometimes the case, even for people with severe motor disabilities. It can also be done any one of a variety of augmentative and assistive communication (AAC) devices, such a special-purpose switches, or sip-puff devices. If we want to avoid additional hardware, other events can trigger a mouse click. For example, the Kinect sensor we use to estimate head position could detect when the user opens their mouth, and use this to initiate a mouse click. For the rest of this paper, the term "mouse click" should be taken to mean a discrete signal that the user can send to communicate to the system that the cursor is currently over the object of interest.

D. System Calibration

In order to be able to accurately project onto locations in the world, and to determine the relationship between head orientation and objects in the world, an initial calibration step is required. Because we are using a PR2 robot, we can assume that all of the sensors and actuators are already calibrated, so the only additional calibration step is to find the relationship between 3d world locations and projected pixels. This relationship is a homography (a linear mapping) between pixels in the camera used to model the world and projected pixels, denoted by the matrix H. H can be found, using standard techniques, by projecting a known calibration pattern, and detecting it with the camera. To find H, we need at least four points whose locations are known in both the project's pixel coordinates, and the camera's pixels coordinates. To project onto any 3d location, we can now project the 3d location into pixels in the camera's coordinates, and then use H to find the corresponding projector pixels. This calibration is very similar to system presented in [16]. One advantage that naturally falls out of this type of calibration is the elimination of any need for explicit keystone, pincushion or any other sort of distortion correction.

IV. Interaction Methods

In this section, we describe the basic building blocks of our projected interface, and how they fit together.

A. Interface Elements

All projected interfaces are built from a small set of simple polygonal elements, which can be annotated with text (see figure 2 for some representative examples). A cursor is overlaid on the interface, providing the user with feedback on where the system thinks they are pointing. If the cursor is within an interface element's selection space, the selected element is highlighted to indicate that it is active. With an active element, a click from the user will change the highlight color to indicate that the click has been received, and will dispatch a message to the control software containing the ID of the interface element which the user has selected. Additionally, if the cursor location is outside of the projectable area, a bar is displayed on the edge of the projectable area indicating the direction of the off-screen cursor. Previous results from [25] indicated that providing feedback in this situation is extremely helpful for users.

Because the mapping between real world coordinates and projected coordinates is known, we have fine control over the geometry of the projected interface. Interfaces can be composed in the real world, positioning elements with respect to objects or markers in the world, and dimensions can be specified in meaningful units such as meters. This makes it easy to design interfaces that fit with the objects they control, and ensures that angles and measurements are reproduced accurately, for example, guaranteeing that elements which should be rectilinear, are rectilinear regardless of the placement or orientation of the projector with respect to the projection surface.

B. Interaction Styles

We are interested in enabling interactions which require a user simply to walk up to the robot to begin interacting with it. However, these interactions will always be embedded in some context, which will allow us to simplify and specialize the interface elements dynamically.

External context is supplied by *where* and *when* the interaction takes place. The range of robot tasks in the kitchen, for example, will be different from those in the

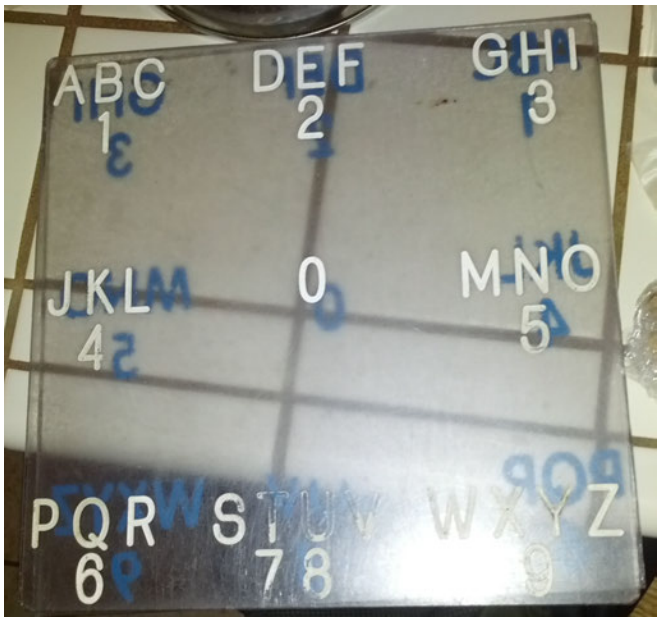

Fig. 3. An example of a letterboard used for alternative communication. Users communicate by looking at letter groups, sequences of which are interpreted by someone experienced with the system (such as a caregiver, family or friend).

dining room, and this will let us specialize the interfaces to make the interaction more efficient. Similarly, the tasks that the user assigns the robot in the morning might be different from those assigned in the evening. Since the robot is capable of estimating its position and its physical environment, we can use this to infer the appropriate context of the interaction.

Task context is context that can be inferred, or learned, from the task itself. For example, if the user always has a particular brand of cereal in the morning, the interface can be specialized to place that choice in a prominent location in the interface. This preference could either be pre-supplied to the robot or, potentially, learned over time through repeated interactions.

We can use both the external and the task context to modify the interface presented, with the goal of making the interaction as efficient as possible. We give some examples of this in the next section.

V. Example Applications

A. Letterboard

To illustrate a simple interaction with our system, we created a projected version of a standard augmentative and alternative communication (AAC) device: a gaze-based letterboard. The particular letter board (shown in figure 3) is one used by a colleague of ours who has quadriplegia and is mute. Use of the physical version of this letterboard involves an able-bodied "listener" holding the board between themselves and the AAC user. The AAC user spells out words by using eye gaze to indicate letter groups to the listener. The listener must correctly identify the letter that the AAC user is looking at, and then uses the letter groups to infer what the AAC user is saying, asking for confirmation along the way. In our reimplementation, we use head pose as a proxy for eye-gaze.

Our interface is context-free in that it does not rely on any objects or properties of the physical world (other than a usable projection surface). It can, however, be made more efficient by adding context to interactions by using a language model to perform text prediction. We have implemented this by creating a scored set of bigrams from one of the standard linguistics corpora [26], and ordering word suggestions based on their scores from the previous word and current partial word. This can be extended even further by learning a language model for each user seeded by, for example, all of their sent emails, and updated as they use the interface.

The robot detects the wall, estimates a parametric planar model of it, and projects the interface onto the surface taking this model into account. In our interface, shown in figure 2a, the letters and numbers from the original are presented statically, comprising most of the area of the interface. A dynamic list of predicted words appears down the right-hand side, and the current sentence is shown along the bottom. In the figure, the user is pointing at "today" with their head pose, and this interface element is highlighted in green. Clicking the mouse button will select it and add it to the sentence. When they are finished, clicking on the completed sentence causes the robot to speak it, using a standard text-to-speech system. We note that this interface is a particularly simplistic virtual version of the physical letterboard. Our intention in showing it is only to illustrate a simple use-case of our system. However, even with this simple system, the AAC user is able to directly communicate with anyone, not just those able to interpret physical letterboard gazes, in any location, as long as there is a flat surface (and they are accompanied by their robot).

B. Television

In addition to simple communication interfaces, the system can also be used to interact with objects in the real world. Many objects already have affordances for changing their state, either on the device, or on external control devices, such as a TV remote control. These types of interfaces present two challenges. First, decoupling the interface from the device requires users to divide their attention [12]. Second, devices such as remote controls often have an abundance of options which can be difficult even for able-bodied users, and impossible for persons with motor disabilities or visual impairments.

Pairing embedded, projected interfaces with existing device controls has the potential to enable powerful yet simple interactions. Using an infrared transceiver module, we have built an interface that enables users to control TV functions with simple head movements. Since the location of the television is known to the robot in the world model that it maintains, the TV can be turned on and off by the user facing the TV and clicking (again with the projector on the robot providing a cursor for feedback). More complicated functions such as changing channels are possible by creating simple interfaces with buttons for these functions. The interfaces

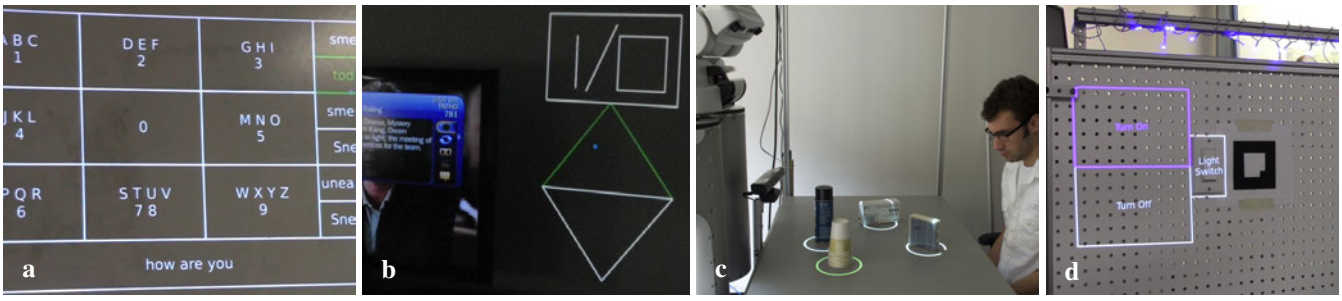

Fig. 2. Several example applications: a) Letterboard interface, in the process of saying "how are you today;" b) TV interface showing a user selecting *channel up*; c) tabletop interface showing several detected objects; d) an interface for controlling a light switch. The "Turn On" button is highlighted, indicating that the switch is currently on.

can either be projected, which will not interfere with normal use of the TV, or by using the TV as the interface display device. If projected, the small controls on the TV remote can be made arbitrarily large, within the limits of the projection system, and unnecessary controls can be left out, affording those with visual impairments improved access to the controls.

In the interface shown, controls for on/off and channel up and down are displayed. When the TV is off, only the on/off button is displayed, and turning on the TV causes extra controls to be displayed. Additional controls can be easily added to the interface, and controls can be hidden or displayed based on the state of the TV.

Accessibility can be further increased by incorporating more task context into the interaction. Standard TV remotes are dumb devices (with a few exceptions). They know nothing about the user, or their preferences. Since our interface is mediated by a robot that is connected to the Internet, we can display interfaces that give program listings, show names, or other contextually-appropriate selection options. If we assume that a person will use our system for an extended period of time, we can learn (or have programmed in) their preferences. If they always watch the channel 9 news at 9pm, we can adjust the interface presentation accordingly, since we know the time.

C. Tabletop Manipulation

A common task for manipulation robots involves moving objects around on a tabletop. This is an important ability for a variety of useful tasks that the robot might perform under the direction of a person. Presented in detail in [25], an interface for directing a robot in pick-and-place tasks can easily be created. In this interface, all objects with which the robot is able to interact are highlighted by drawing circles around them. A cursor, representing the point where the ray from the user's head pose intersects the world model, is projected onto the work surface, which both shows the user where the system believes them to be pointing, and also indicates where the robot is able to pick up or place objects.

Object detection, grasp planning, and execution [27] are all performed by modules which are core packages within ROS. The system runs a simple two-state finite state machine, the state of which depends on whether or not the robot is currently grasping an object. When no object is being grasped, a click on a valid object directs the robot to pick up that object. When an object is being grasped, a click anywhere in the workspace directs the robot to put down the object at the indicated location. This interface can be augmented with other task-dependent elements. For example, for a sorting task, areas can be projected onto the workspace for each category, assisting in object placement. An example of this is shown in figure 2c, with several manipulatable objects circled on a table.

D. Light Switch

Users should also be able to control the physical infrastructure in their environments. As an example of this, we have created an interface that allows the user to turn on and off a light switch. The robot first detects and categorizes the light switch, and places an interface element that says "light switch" over it. Clicking on this element causes a context-sensitive menu to be displayed, as shown in figure 2d. This menu enumerates all of the physical manipulations that can be performed on the light switch. Clicking on "turn on," for example, will cause the robot to move over to the light switch and actuate it with its gripper.

Once a device is detected and classified, it can be stored in the world model maintained by the robot. This allows the device to be used in the future without the detection and classification step. The locations and types of switches and other infrastructure elements could even be entered into a persistent world model by a human, to remove the need for recognition and classification completely.

VI. Future Work and Discussion

Ultimately, we envision these projected interfaces as one piece in a larger system for enabling anyone, but especially users with physical disabilities and visual impairments, to control mobile robots in a variety of tasks in their homes. Central to this will be a rich, persistent model of the world, where the robot can store information about the environment. This information can be used to give context to the interactions, and will allow us to make interfaces that take advantage of this context.

Some of the tools necessary for such models already exist such as the ability to build semantic maps, which can provide much richer world models than those we have presented. Semantic maps store meaningful information about objects and locations, which could include data such as locations

of light switches or of objects with which the robot knows it can interact, or users are interested in interacting with. Projects such as RoboEarth [28] could also be leveraged for information about recognizing and interacting with previously unknown objects.

In this paper, we presented a framework for embedded interfaces for use with mobile manipulation robots. The system is designed to be usable by persons with severe motor disabilities by using only simple motions as input. It additionally removes the need for interactions to be mediated by a traditional personal computer and monitor, moving interactions out into the real world. We additionally presented several illustrative use-cases and discussed how different types of contexts affect the interaction. The framework is quite general, and will work with any input device that can generate a pointing vector. Our implementation uses visual head pose estimation, but a laser pointer, orientation sensors in Google Glass, or some other device could be used with no modifications to the framework.

As robots become more capable of performing useful work in people's homes, the interfaces to support that must become more integrated with the environments in which the tasks take place. By moving interfaces from computer screens to the objects to be manipulated themselves, we hope we have taken a step in that direction.

VII. Acknowledgments

We would like to thank the members of the Robots for Humanity project [29], both at Willow Garage, and at the Healthcare Robotics Lab at Georgia Tech, as well as Henry and Jane Evans.

References

[1] K. Hsiao, M. Ciocarlie, and P. Brook, "Bayesian grasp planning," in *Proceedings of the International Conference on Robotics and Automation Workshop on Mobile Manipulation: Integrating Perception and Manipulation*, 2011.

[2] T. Rühr, J. Sturm, D. Pangercic, D. Cremers, and M. Beetz, "A generalized framework for opening doors and drawers in kitchen environments," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, (St. Paul, MN, USA), May 14–18 2012.

[3] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth, "Robotic Roommates Making Pancakes," in *Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots*, (Bled, Slovenia), October, 26–28 2011.

[4] S. Waldherr, S. Thrun, and R. Romero, "A gesture-based interface for human-robot interaction," *Autonomous Robots*, vol. 9, no. 2, pp. 151–173, 2000.

[5] A. Leeper, K. Hsiao, M. Ciocarlie, L. Takayama, and D. Gossow, "Strategies for human-in-the-loop robotic grasping," in *Proceedings of the 3rd International Conference on Human Robot Interaction (HRI)*, (Boston, MA), pp. 1–8, 2012.

[6] H. Nguyen, M. Ciocarlie, K. Hsiao, and C. Kemp, "Ros commander: Flexible behavior creation for home robots," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, In Press.

[7] D. Engelbart, "Augmenting human intellect: A conceptual framework," 2001.

[8] I. E. Sutherland, "Sketchpad: A man-machine graphical communication system," in *Proceedings of the SHARE Design Automation Workshop*, pp. 6–329, ACM, 1964.

[9] D. Rasmusson, R. Chappell, and M. Trego, "Quick glance: Eye-tracking access to the Windows 95 operating environment," in *Proceedings of the Fourteenth International Conference on Technology and Persons with Disabilities (CSUN)*, 1999.

[10] Madentec Inc, "Trackerpro." http://www.madentec.com/products/tracker-pro.php.

[11] P. Mistry and P. Maes, "Sixthsense: A wearable gestural interface," in *ACM SIGGRAPH ASIA 2009 Sketches*, p. 11, ACM, 2009.

[12] D. Schmidt, D. Molyneaux, and X. Cao, "PICOntrol: Using a handheld projector for direct control of physical devices through visible light," in *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 379–388, ACM, 2012.

[13] X. Cao and R. Balakrishnan, "Interacting with dynamically defined information spaces using a handheld projector and a pen," in *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 225–234, ACM, 2006.

[14] G. Reinhart, W. Vogl, and I. Kresse, "A projection-based user interface for industrial robots," in *Proceedings of the IEEE Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS)*, pp. 67–71, 2007.

[15] T. Sasai, Y. Takahashi, M. Kotani, and A. Nakamura, "Development of a guide robot interacting with the user using information projection — basic system," in *Proceedings of the International Conference on Mechatronics and Automation (PICMA)*, pp. 1297–1302, 2011.

[16] S. Sato and S. Sakane, "A human-robot interface using an interactive hand pointer that projects a mark in the real work space," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, pp. 589–595 vol.1, 2000.

[17] R. A. Bolt, ""Put-That-There": Voice and gesture at the graphics interface," in *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 262–270, ACM, 1980.

[18] A. Wilson and S. Shafer, "Xwand: UI for intelligent spaces," in *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, pp. 545–552, ACM, 2003.

[19] H. Nguyen, A. Jain, C. Anderson, and C. Kemp, "A clickable world: Behavior selection through pointing and context for mobile manipulation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 787–793, September 2008.

[20] M. M. Loper, N. P. Koenig, S. H. Chernova, C. V. Jones, and O. C. Jenkins, "Mobile human-robot teaming with environmental tolerance," in *Proceedings of the 4rd International Conference on Human Robot Interaction (HRI)*, pp. 157–164, ACM, 2009.

[21] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *Proceedings of the IEEE International Conference on Robotics and Automation Workshop on Open Source Software*, 2009.

[22] H. Kato, , M. Billinghurst, and I. Poupyrev, "ARtoolkit user manual, version 2.33," 2000. Human Interface Technology Lab, University of Washington.

[23] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381–395, June 1981.

[24] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real time head pose estimation from consumer depth cameras," in *33rd Annual Symposium of the German Association for Pattern Recognition (DAGM'11)*, September 2011.

[25] D. A. Lazewatsky and W. D. Smart, "Context-sensitive in-the-world interfaces for mobile manipulation robots," in *Proceedings of the 21st International Symposium on Robot and Human Interactive Communication (Ro-Man)*, pp. 989–994, 2012.

[26] W. N. Francis and H. Kucera, "The brown corpus: A standard corpus of present-day edited american english," 1979. Brown University Liguistics Department.

[27] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Sucan, "Towards reliable grasping and manipulation in household environments," *Proceedings of the International Symposium on Experimental Robotics (ISER)*, 2010.

[28] M. Waibel, M. Beetz, J. Civera, R. D"Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. M. M. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle, and R. van de Molengraft, "Roboearth," *Robotics Automation Magazine*, vol. 18, no. 2, pp. 69–82, 2011.

[29] T. L. Chen, M. Ciocarlie, S. Cousins, P. M. Grice, K. Hawkins, K. Hsiao, C. C. Kemp, C.-H. King, D. A. Lazewatsky, A. E. Leeper, H. Nguyen, A. Paepcke, C. Pantofaru, W. D. Smart, and L. Takayama, "Robots for humanity: Using assistive robotics to empower people with disabilities," *Robotics and Automation Magazine*, vol. 20, pp. 30–39, March 2013.