Investigation of Human-Robot Interface Performance in Household Environments

Sven Cremer*[a], Fahad Mirza[a], Anthony Hingeley[a], Rommel Alonzo[a], Yathartha Tuladhar[a], Dan O. Popa[b]

[a]Next Generation Systems, University of Texas at Arlington, Arlington, Texas 76010;
[b]Dept. of Electrical Engineering, University of Louisville, Louisville, KY 40292 USA

ABSTRACT

Today, assistive robots are being introduced into human environments at an increasing rate. Human environments are highly cluttered and dynamic, making it difficult to foresee all necessary capabilities and pre-program all desirable future skills of the robot. One approach to increase robot performance is semi-autonomous operation, allowing users to intervene and guide the robot through difficult tasks. To this end, robots need intuitive Human-Machine Interfaces (HMIs) that support fine motion control without overwhelming the operator. In this study we evaluate the performance of several interfaces that balance autonomy and teleoperation of a mobile manipulator for accomplishing several household task.

Our proposed HMI framework includes teleoperation devices such as a tablet, as well as physical interfaces in the form of piezoresistive pressure sensors arrays. Mobile manipulation experiments are performed with a sensorized KUKA youBot, an omnidirectional platform with a 5 degrees of freedom (DOF) arm. The pick and place tasks involve navigation and manipulation of objects in household environments. Performance metrics include time for task completion and position accuracy.

Keywords: Assistive robotics, human-machine interfaces, mobile manipulation

1. INTRODUCTION

Robotics is currently in its 3rd phase of development. The late 1970s saw the rise of unintelligent, stationary industrial robots. During the 1990s, progress was made in mobility, intelligence, and cooperation to develop “personal” robots in areas such as research, education, and entertainment [1]. Today is the generation of “ubiquitous” robots that will support humans in everyday life. Unlike industrial robots, future co-robots will share their working space with humans and work in household environments. These assistive devices do not necessarily have to be fully autonomous. It has been shown that a human-robot pair can outperform either a human or robot working alone [2].

Therefore, intuitive human-robot interfaces will play a crucial role. Non-expert user must be able to interact and communicate with the robot, which may involve speech, gesture, haptic displays, etc. [3]. Interaction can also be physical, for example a robot can learn a new task from a novice operator via kinesthetic teaching, where the user manually pushes and pulls the manipulator to complete a task [4].

Since physical contact may occur, human-robot interfaces will also play a key role in safety. Safety can be divided into a physical and behavior aspect [5]. When physical contact occurs, the control architecture can limit joint torques and velocities, and take advantage of passive compliance. There are adaptive schemes that can compensate for unknown parameters and disturbances while guaranteeing robust and stable control [6]. In our recent work, we validated a novel neuroadaptive framework that improved physical HRI [7, 8]. These methods use single F/T (force/torque) sensors to estimate the force applied by a human on the robot end-effector. A promising technology is robot skin, which equips the robot with touch and allows precise localization of multiple contact forces [9]. While whole-body, human-like skin yet has to be realized, there already exists pressure sensitive piezo-resistive “taxel” arrays encapsulated in flexible silicone substrates [10].

*{sven.cremer,fahad.mirza34,anthony.hingeley,rommel.alonzo,yathartha.tuladhar}@mavs.uta.edu, dan.popa@louisville.edu; www.uta.edu/ngs

For behavior safety, the robot takes multi-modal cues such as facial expression and body pose to determine human intent. This allows the robot to align its goals with the operator, plan ahead, and adjust its behavior. For example, if a collision is anticipated or a child interacts with the robot, the control scheme could increase its compliance. Feedback from the operator could also allow the robot behave more human-like, making the interface simpler to understand and more intuitive for the operator [11].

The intuitive and safe human-robot interfaces must be facilitated by multi-modal sensor data. The robot can perceive and understand its environment through camera images that are then processed by vision algorithms. For example, the OpenCV library implements several real-time computer vision functionalities, including people detection and face tracking [12]. Additional information can be gained from depth: RGB-D cameras such as the Microsoft Kinect or Asus Xtion have become de-facto standard in robotics [13]. Laser scanners usually provide 2D depth information and are commonly used for mobile navigation. Sonar or ultrasonic sensors are less accurate but relative inexpensive range-finders, and have been deployed on co-robots such as Baxter to detect human presence [14]. Another alternative are thermal sensors, which can detect radiated heat or far-infrared rays from nearby humans. In addition to low cost, processing infrared data is faster compared with image processing to detect humans.

In this paper, a standard robot platform was modified to investigate the performance of several HMIs in a household environment, continuing the work described in [15]. The KUKA youBot, which is commonly used in academia, was transitioned from the “personal” to “ubiquitous” generation of robots by performing several hardware upgrades. The platform was sensorized by installing a RGB-D camera for object and human detection, a laser scanner for fully and semi-autonomous navigation as well as collision detection, robot skin patches consisting of piezo-electric pressure sensors for physical interaction, and thermal sensors to detect human presence. A National Instruments roboRIO was added to provide additional input/output ports and a software architecture was developed to process the data. Networking was improved to allow remote control via a tablet interface and assisted joystick teleoperation. As such, the contribution of this paper is how to sensorize a standard robot platform (both hardware and software) for human-robot interfaces and their expected performance in a household environment.

In the following section, we describe the robot platform and how it was upgraded to meet the HMI requirements. In Section 3, we discuss the sensors used for the multi-modal interfaces and the software architecture in Section 4. The experimental setup is presented in Section 5 and the results in Section 6. Finally, Section 7 concludes the paper and postulates future work.

2. HARDWARE PLATFORM

In this section we describe the robot platform and the necessary hardware sensors and interfaces needed for teleoperation.

The automation company KUKA specifically developed the youBot as a research and application platform for mobile robotics [16]. It has a five degrees of freedoms (DOF) manipulator with a height of 655mm and a workspace of $0.513m^3$. The arm can lift a payload up to 0.5kg and has a position repeatability of 0.1mm. Grasping can be performed with a two-finger gripper that is being powered by 2 independent stepper motors and has a range of 70 mm.

The arm is mounted on a omni-directional base of dimension 580mm (length) by 376mm (width). The four Mecanum wheels allow movement in any direction (x,y) at any orientation (θ). It can reach velocities up to 0.8m/s and carry a payload of 20kg. The mobile base houses a rechargeable battery which allows a runtime of approximately 90 minutes. An onboard mini PC (Intel Atom Dual Core CPU, 2GB RAM) runs the Linux-based operating system Ubuntu. The platform components rely on EtherCAT communication with a 1ms, real-time cycle.

A remote workstation can be connected to the robot via Ethernet cable. This is suitable for running computationally demanding algorithms and heavy graphics processing that would slow down the onboard PC. To create an untethered setup, our first upgrade consisted of mounting a wireless router (ASUS RT-N66U Dual-Band Wireless-N900 Gigabit Router) on top of the base. This router is connected to the onboard PC via Ethernet connection, and acts as a bridge to transmit data to a remote workstation (Fig. 1). This allows an operator to run and troubleshooting processes remotely, view and collect live data, and visualize the robot state in a graphics program. In addition, the youBot WiFi network can be used to connect interface devices such as tablets, phones, or laptops.

The additional sensors for the HMI require Input/Output (I/O) ports for signal acquisition, conditioning, and networking. The National Instrument roboRIO is an advanced robotics controller that features several built-in ports for "I2C, SPI, RS232, USB, Ethernet, PWM, and relays" [17]. Released in 2015, it is part of the reconfigurable I/O (RIO) family and use the Xilinx Zynq chipset. It has a reconfigurable FPGA and is powered by a 667 MHz dual-core Real-Time processor.

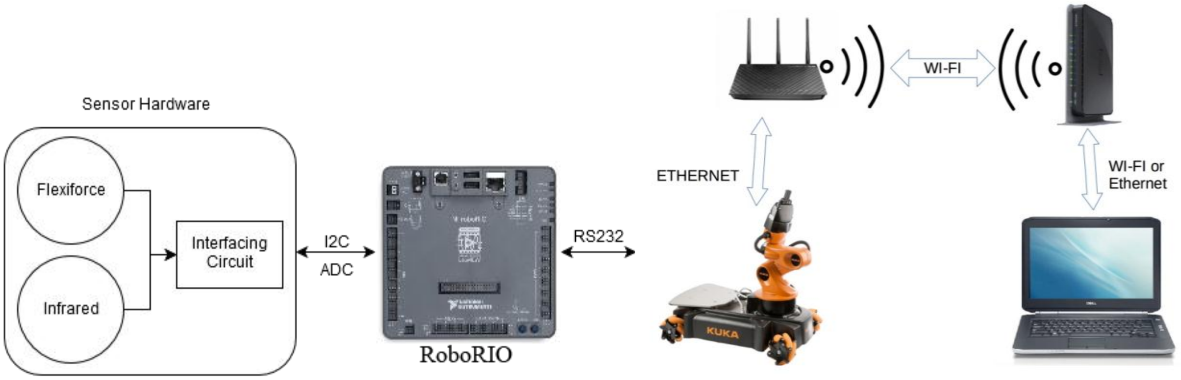


Figure 1. Communication diagram for sensor data acquisition and wireless setup between youBot and remote workstation.

3. SENSORS FOR HMI

3.1 Laser scanner

A laser scanning rangefinder was installed at the front side of the youBot base to detect obstacles. The sensor data can be used for autonomous navigation and assistive teleoperation. We used the Hokuyo URG-04LX-UG-01 model, which has a 240° field of view and a measurements distance of 4m [18].

3.2 Robot Skin

For physical interaction, the youBot end-effector link was outfitted with four skin patches consisting of pressure sensors embedded in P10 polymer as shown in Fig. 2. The Tekscan Flexiforce thin-film sensors are ideal for measuring force between two surfaces. They are cost effective and durable with good sensor characteristics: linearity error within 3%, hysteresis less than 4.5%, drift less than 5%, and low temperature sensitivity (0.36% per °C). The maximum response time is 5 μsec and they handle up to 100lbs (445N). The active sensing area is 9.53mm diameter.

The particular sensors used are piezoresistive: as force increases the resistance decreases from infinity to approximately 300kΩ. A voltage divider circuit with an emitter buffer was designed to measure human forces up to 50lbs (222N). The circuit was implemented on a custom data acquisition board (or MicroBoard), which conditions the pressure data from several sensors (or taxels). The results are read by the roboRIO using its analog input ports.

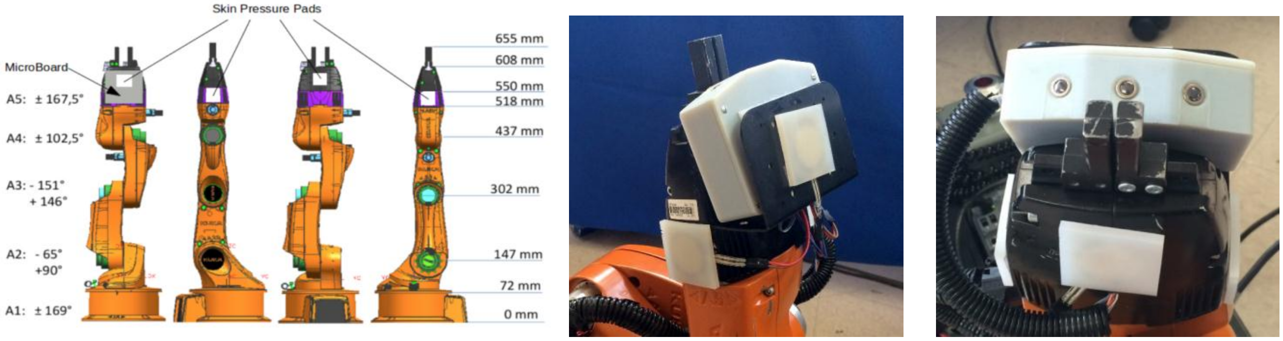


Figure 2. Robot skin patch placement on the KUKA youBot manipulator [16].

3.3 Thermal sensors

Human presence can be detected by using cameras and computer vision algorithms. A simpler and more cost effective approach involves thermal sensors, which detect radiated heat or far-infrared rays of an object. Hence, they are not affected by different lightning conditions like conventional cameras and simple thresholding can be used for detection.

The Omron's MEMS thermal sensor (D6T-44L) consists of a MEMS thermopile sensor chip covered with a silicon lens [18]. The chip measures an electromotive force and an embedded circuit converts the analog signals to digital temperature values. In contrast to conventional pyroelectric sensors, Omron's sensor does not measure a change in signal and continually detects the far-infrared ray of an object. Hence, the sensor is able able to catch a signal of a moving as well as stationary person.

A custom box with three thermal sensors was mounted on the end-effector just below the two finger-gripper. The sensor itself outputs a 4x4 pixel array and has a view angle of 45° [18]. With three sensors, 4x12 pixels cover approximately 120° with some overlapping. The hardware including thermal sensors and wiring were encased in a 3D printed box as shown in Fig. 3. The measured values are transmitted through an I^{2}C bus to the roboRIO.

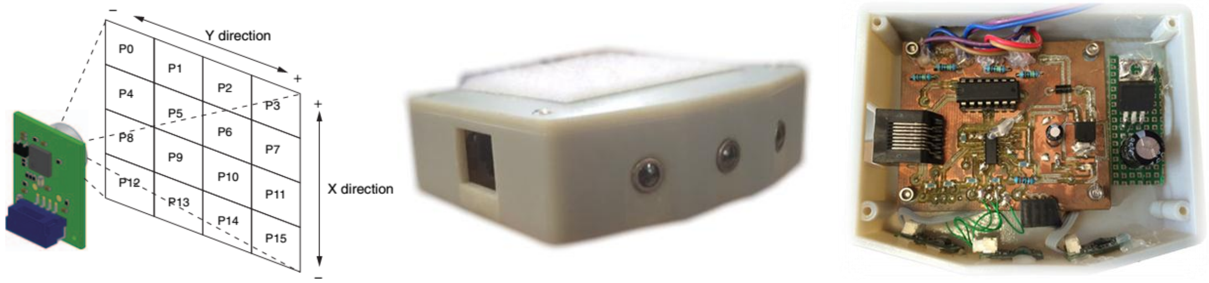


Figure 3. The detection area of the Omron D6T-44L [18]. The 3D printed box and circuitry for three thermal sensors.

4. SOFTWARE ARCHITECTURE

4.1 Robot Operation System

The KUKA youBot runs the linux-based operating system Ubuntu 12.04. The platform is controlled with the open-source Robot Operating System (ROS), a software framework originally developed by Stanford Artificial Intelligence Laboratory in 2007 [19]. ROS has become a de-facto standard in robotics and there is a large collection of software packages developed by the community. Since ROS is programming language and hardware agnostic, these packages can easily be re-used on different platforms. Programs are executed as independent ROS nodes and data is transmitted via ROS messages. This distributed architecture allows several components to run simultaneously, for example sensor data processing, navigation, and perception.

In this spirit, we developed a multi-layered, ROS-based architecture to allow robot autonomy as well as user intervention via several HMIs such as tablet apps or pressure sensors (Fig. 4). Several stand-alone ROS packages were developed, with different functionalities for navigation and manipulation as described in Table 1. The program flow is determined by a cortex node which acts as layer between the interfaces and control modules. It contains a state machine which utilize ROS topics or services to calls other ROS nodes that then executes the necessary code. If anything fails, the cortex node runs contingency plans and is able to restart faulty ROS nodes. This also simplifies data flow making it easier to debug and capture data.

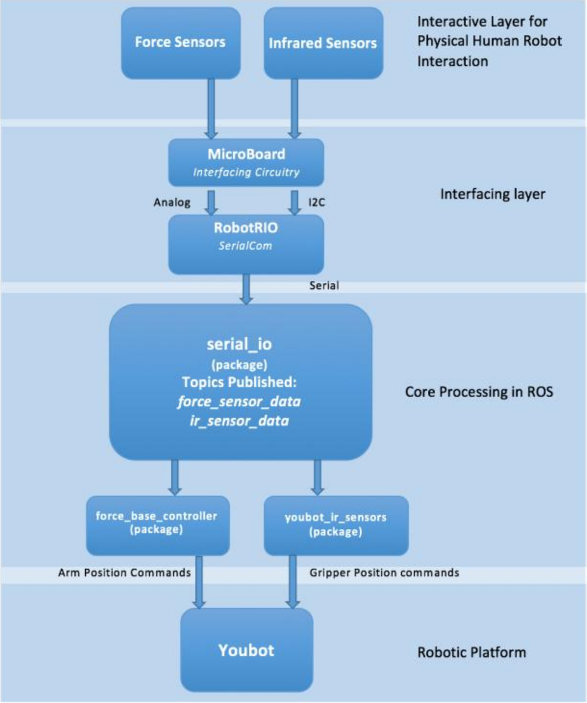


Figure 4. Software architecture.

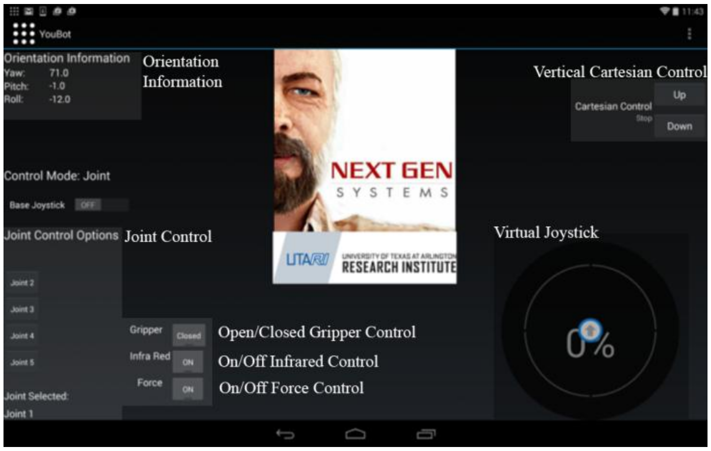
Table 1. Overview of the developed ROS packages.

ROS package	Description
youbot_cortex	Contains state machine nodes which acts as a layer between the data received from the tablet app and the actual arm controllers. It sends service requests to enable/disable different control modes, joints, and sensors.
cortex_msgs	Defines custom ROS messages and services.
serial_io	Reads data from roboRIO via a serial communications channel and publishes it to ROS topics.
youbot_force_sensors	Subscribes to force data and compute the command velocities for the base movement. KDL (Kinematics and Dynamics library) is used for 3D frame and vector transformations.
youbot_ir_sensors	Processes thermal sensor data (filtering and thresholding) to control the gripper.
youbot_description	Contains the youBot robot model.
youbot_navigation	Utilities for mapping, localization, and path planning.
youbot_arm_controllers	Send actual position commands to the Youbot motors using different control algorithms (individual and combined joint control).
youbot_cartesian	Cartesian arm controller using the motion planning software MoveIt! [20].

4.2 Tablet Interface

A youBot tablet application was developed using ROSJava [21] to allow users to control the platform with an easy-to-use GUI interface. *Figure 5. Tablet interface for controlling the youBot.*The app is split up into 3 different classes: ViewController, NodePublisher and the VirtualDivet. The *ViewController* connects the GUI interface (buttons, toggle buttons, switches, text) to backend code. Using ROS messages, the *NodePublisher* communicates user intent to the cortex program which then initiates the robot movement. The *VirtualDivet* class implements the touch-screen joystick in the lower right hand corner. It allows the user to move around a divet inside a circular area, where the off-center distance produces velocity commands between 0 and 100%.

Figure 5 show the layout on a Nexus 10 Android tablet. The top left corner displays the tablet orientation, i.e. the pitch and roll from the gyroscope and the yaw relative to earth from the magnetic compass. The active control mode can either be individual or combined joint control. Joint control mode allows the selection of individual joints which then can be moved by tilting the tablet. The joint velocity is proportional to the pitch of the tablet. In combined joint control, several joints are moved together to move the end-effector in an arc. Joint limits prevent the user from hitting the floor or robot. The Cartesian position of the end-effector can be controlled with buttons in the upper right corner. Vertical Cartesian control moves the end-effector along the y-axis of the base frame. Finally, there are switches for activating different interfaces such as the thermal and pressure sensors. The grippers switch opens or closes the two-finger pincher. The base joystick toggle enables the virtual joystick, which moves omnidirectional base of the robot platform.



Orientation Information
Yaw: 71.0
Pitch: -1.0
Roll: -12.0

Control Mode: Joint

Joint Selected:
Joint 1

Figure 5. Tablet interface for controlling the youBot.

4.3 RoboRio Software

The roboRIO was programmed using the National Instruments LabVIEW. The code is executed in a real-time loop, where pressure sensors are read every 1ms using the analog input ports. The infrared sensors generate data every 30ms and use the I^2C interface. After being captured, the data is relayed via USB to the youBot using the RS232 communication protocol. Collecting sensor information and communicating with ROS are parallel tasks, made possible by the multicore processors on the roboRIO. This way the ROS nodes can access the sensor data at approximately the same rate: 1000Hz for pressure sensor and 33Hz for thermal data.

5. DESCRIPTION OF EXPERIMENTS

Experiments were conducted to test the functionality of the implemented hardware and performance of the proposed HMI schemes. The tasks involved object pick-and-place using the tablet interface and pressure sensors, as well as autonomous navigation. The primary objective was to measure the difference in completion time and accuracy of path trajectory for expert and novice users. Non-expert users are defined as those who had no prior experience with the tablet application, the robot, or its functionality. Both types of users were instructed with a brief overview of the desired tasks for completion for the desired motion. Non-expert users were given a brief introduction to the application and the buttons which control the interface position of the base and motion of the arm. To determine the trajectory error, the youBot was localized using odometry and laser scanner data.

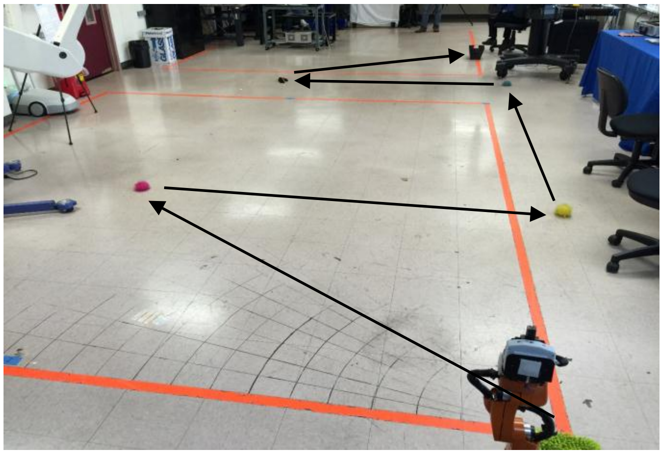


Figure 6. Pick and place experiment on the laboratory floor.

5.1 Pick and Place

In the pick and place task, a user was asked to move several objects between waypoints with the KUKA youBot. To create a rigorous method of testing, the waypoints were marked on the floor as shown in Fig. 6. The path involves straight and diagonal movements, and the distance between each point was varied: 3.05m from point A to point B, 2.76m from point B to point C, 3.68m from point C to point D, 2.80m point D to point E, and 3.07m from point E to point F.

Different interfaces were tested to compare the accuracy and completion time of physical user guidance and tablet teleoperation. Both expert and novice users were asked to pick up 5 objects in 6 trials. This was repeated for two types of interfaces: 1) teleoperation via tablet and 2) direct physical interaction (mannequin mode) via the skin patches.

In Teleoperation mode, the user was required to accomplish the following tasks:

1. Pick up initial object
2. Transport object to waypoint
3. Place object within 2-3 inches from waypoint marker
4. Pick up next object
5. Proceed to next waypoint
6. Deliver final waypoint item into a bin

The motion of the base is controlled by the divet, while the arm can be controlled by joint or Cartesian modes. The user can also open and close the grippers using the button located on the tablet application.

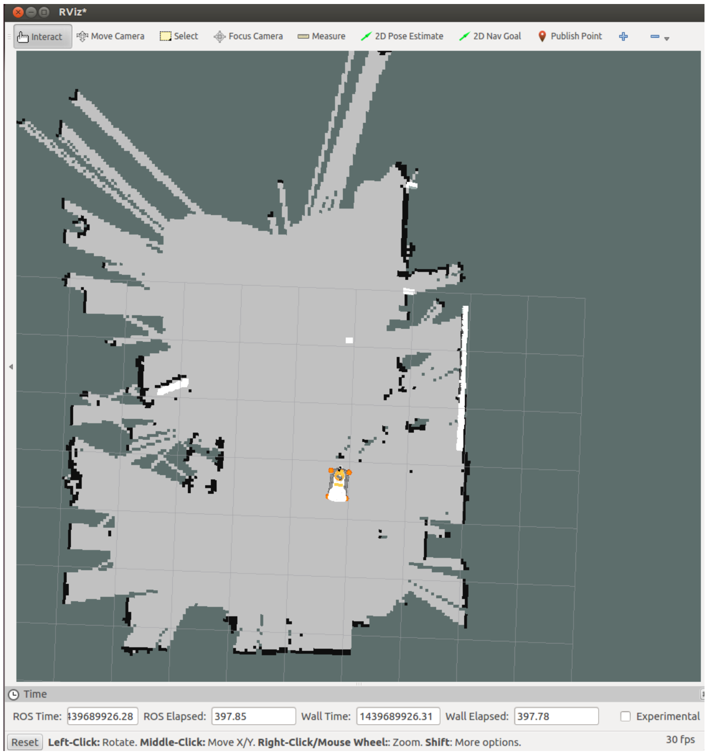
In mannequin mode the tasks were as follows:

1. Pick up initial object
2. Apply a force to the desired direction pressure sensors
3. Guide Youbot to waypoints
4. Use the tablet to control the arm using combined joint angles.
5. Control Gripper grasp using tablet interface
6. Move to next waypoint and deliver current object in close proximity to next target (2-3 inches apart).
7. Final waypoint: object in grasp must be left at final location.

The user must press the pressure sensors to control the robot base to navigate the outlined path to grasp the desired objects. The user must place the object within the grasp as close as possible to the next target (2-3 inches). The reasoning for this method is for stating that although the system may be slower, the accuracy is more precise than that of the tablet interface.

5.2 Path Guidance

In addition to pick and place experiments, in which the user guided the robot by observation, we also carried out several user experiments using the autonomous navigation and obstacle avoidance capabilities of the youBot. For autonomous navigation the youBot recieved goal commands via the ROS Visualization (RViz) interface shown in Fig. 7. This software suite provides the user with a predefined map of the location and obstacle detection marked in white. In the map input via RViz the robot used the onboard LIDAR to map the current location in the room and obstacles encountered that might come within its path. The path taken is provided by the user input in RViz via drag and drop mouse commands and transmitted to the Youbot. The local Youbot controller determines whether or not an obstacle is in the path and will avoid the obstacle using potential field method.



ROS Time: 439689926.28 ROS Elapsed: 397.85 Wall Time: 1439689926.31 Wall Elapsed: 397.78 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options. 30 fps

Figure 7. RViz interface showing the location of the youBot.

6. RESULTS

6.1 Completion times

The timed trial data for each user was tabulated below for a few expert and non-expert users. In teleoperation mode, the common task completion time for an experienced user was 2.5-3 minutes. However, the non-expert users took longer, between 5.7-7 minutes to complete the desired tasks. The time difference for both non-users and expert users required to operate the robot with the two different interfaces is minimal. The disparity in time between users may be attributed to individual skill level of the person. Experimental results and tabulated completion times are shown in Table 2 and Fig. 8.

Table 2. Task completion time (in seconds) for expert and novice users of the tablet and mannequin interfaces. The mean and standard deviation (STD) is computed for 6 trials.

Interface	Expert user 1	Expert user 2	Novice user 1	Novice user 2
Tablet	175	188	395	368
	158	174	388	343
	162	159	382	340
	153	141	370	348
	148	139	372	330
	178	181	409	398
Mean	*162*	*164*	*386*	*355*
STD	*12.0*	*20.7*	*14.7*	*24.7*
Mannequin	169	176	401	382
	171	159	402	389
	173	170	393	381
	166	162	399	395
	159	160	383	346
	171	169	401	389
Mean	*168*	*166*	*397*	*380*
STD	*5.1*	*6.7*	*7.4*	*17.6*

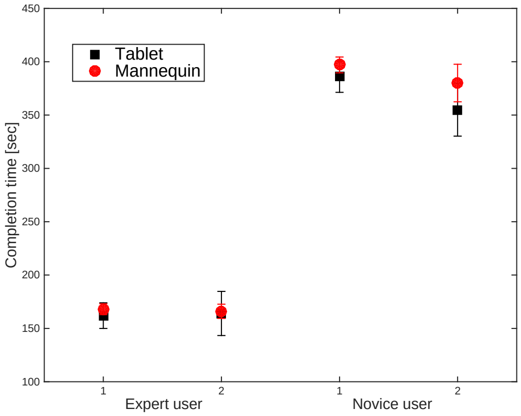


Figure 8. Error plot showing completion times for different users.

6.2 Guidance accuracy

In addition to completion times, we also recorded the location information during interaction with users. The positions of the robot during experiments were updated with respect to the room using the LIDAR sensor and internal odometry. Two expert and two non-expert users were asked to follow a taped path on the lab floor, and each experiment was repeated five times. The figures below summarize the trajectories obtained via manikin and tablet control, respectively.

By general inspection of these graphs, one can infer that the manikin mode is more accurate than teleoperation with a tablet, and that the differences between expert and non-expert users are small.

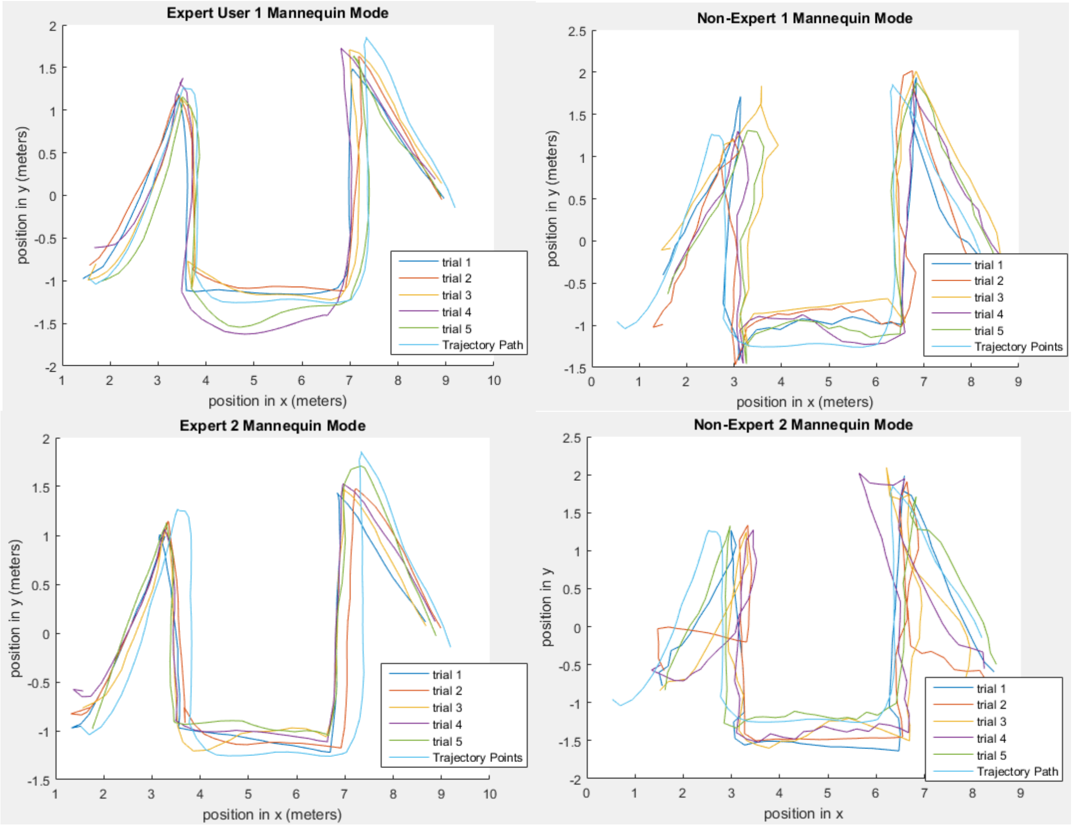


Figure 9. Youbot Cartesian position plots during task execution via mannequin control showing x,y robot coordinates on lab floor in meters for five trials with both expert and non-expert users.

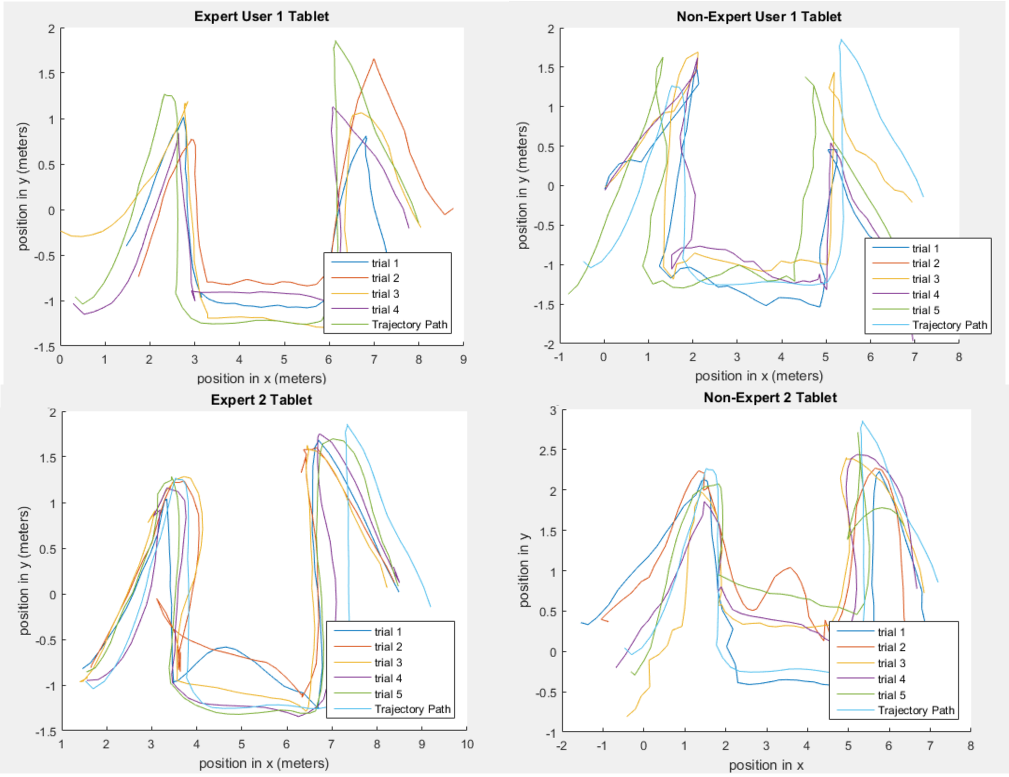


Figure 10. Youbot Cartesian position plots during task execution via tablet control, showing x,y robot coordinates on lab floor in meters for five trials with both expert and non-expert users.

6.3 Autonomous Navigation

Another set of experiments was conducted using the robot autonomous navigation controller, by pointing and clicking between waypoints along a trajectory. In one set of trials, the robot had to move around an obstacle placed in its path, while in another, the robot moved between two successive waypoints in a sharp right corner. The results depicted in the figure below show that in both cases, the resulting trajectories are not qualitatively more accurate than the ones obtained using manikin mode guidance.

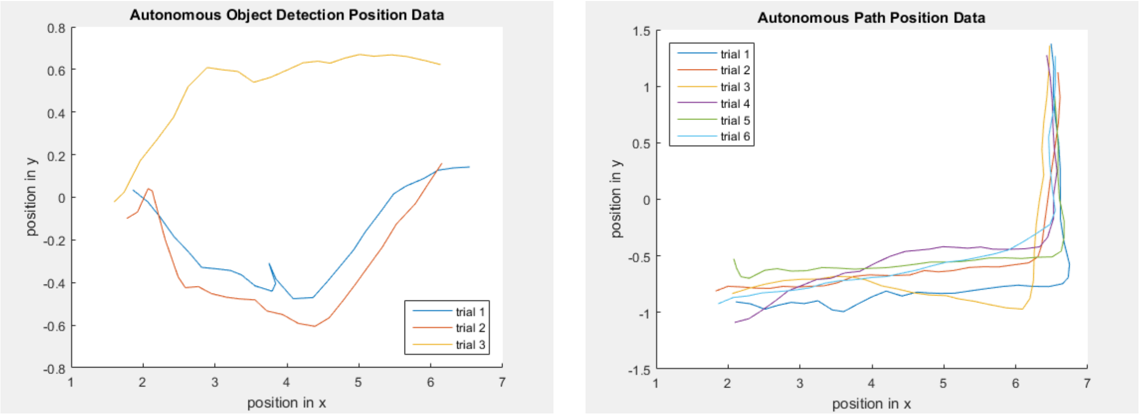


Figure 11. Youbot Cartesian position plots during task execution via autonomous navigation, showing x,y robot coordinates on lab floor in meters for trials involving obstacle avoidance (left) and waypoint navigation with a sharp turn (right).

7. CONCLUSION

In this paper the Kuka youBot mobile manipulator was sensorized for user operation via a Nexus 10 tablet, physical interaction, and gesture. a LIDAR unit, pressure and IR sensors consistent with robotic skin. System integration was accomplished through the addition of a roboRIO controller and computing infrastructure based on the Robot Operating System (ROS). After the HMI and sensor upgrades, the robot can be guided by users via teleoperation with a tablet, direct physical guidance, as well as through a point and click graphical GUI. We demonstrated basic capabilities and acquired data from a few users, which will pave the way for larger experimental studies in the future. Several pick and place tasks for objects placed on the lab floor were completed by two users familiar with the system ("experts"), and two users not familiar with the system ("non-expert"). The results indicate that:

1. The physical interaction (or mannequin) guidance results in the most accurate robot trajectories for both expert and non-expert users. As a result, this interface is intuitive to use in precision guiding tasks.
2. Tablet teleoperation and mannequin guidance result in similar task completion times. Expert users can complete pick and place manipulation and mobility tasks significantly faster than non-expert users. More research is necessary to reduce training time.
3. The autonomous point and click interface had the shortest path time for traversing free and obstacle-filled environments, but it is less accurate than manikin guidance. This interface may be very useful if a map of the environment is available.

ACKNOWLEDGEMENTS

This research was supported by National Science Foundation Grants NRI-1208623 and BIC-1261005720, and Qinetiq-North America Corporation.

REFERENCES

[1] P. Kopacek, "Development trends in robotics," e i Elektrotechnik und Informationstechnik, 130, 42–47 (2013).

[2] H. Ding, M. Schipper, and B. Matthias, "Optimized task distribution for industrial assembly in mixed human-robot environments - Case study on IO module assembly," IEEE International Conference on Automation Science and Engineering (CASE), 19–24 (2014).

[3] K. Dautenhahn and J. Saunders, [New Frontiers in Human-robot Interaction], John Benjamins Publishing Company, Philadelphia, (2011).

[4] S. Cremer, M. Middleton, and D. O. Popa, "Implementation of advanced manipulation tasks on humanoids through kinesthetic teaching," Proc. of the 7th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA), 1–6 (2014).

[5] G. Herrmann and C. Melhuish, "Towards Safety in Human Robot Interaction," Int. J. Soc. Robot., 2, 217–219 (2010).

[6] F. L. Lewis, D. M. Dawson, and C. T. Abdallah, "Robot Manipulator Control," Theory Pract., 656–661 (2004).

[7] I. Ranatunga, S. Cremer, F. L. Lewis, and D. O. Popa, "Neuroadaptive Control for Safe Robots in Human Environments: A Case Study," IEEE International Conference on Automation Science and Engineering (CASE), (2015).

[8] I. Ranatunga, S. Cremer, D. O. Popa, and F. L. Lewis, "Intent aware adaptive admittance control for physical Human-Robot Interaction," IEEE International Conference on Robotics and Automation (ICRA), 5635–5640 (2015).

[9] B. D. Argall and A. G. Billard, "A survey of Tactile Human-Robot Interactions," Robotics and Autonomous Systems, 58, 1159–1176 (2010).

[10] C. Nothnagle, J. R. Baptist, J. Sanford, W. H. Lee, D. O. Popa, and M. B. J. Wijesundara, "EHD printing of PEDOT: PSS inks for fabricating pressure and strain sensor arrays on flexible substrates," Next-Generation

Robotics II; and Machine Intelligence and Bio-inspired Computation: Theory and Applications IX, 9494, 949403 (2015).

[11] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human–robot interaction," Mech. Mach. Theory, 43, 253–270 (2008).

[12] G. Bradski and A. Kaehler, [Learning OpenCV: Computer Vision with the OpenCV Library], O'Reilly Media, (2008).

[13] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D Mapping : Using Depth Cameras for Dense 3D Modeling of Indoor Environments," Int. J. Rob. Res., 31, 647–663 (2012).

[14] C. Fitzgerald, "Developing baxter," IEEE Conference on Technologies for Practical Robot Applications (TePRA), (2013).

[15] R. Alonzo, S. Cremer, F. Mirza, S. Gowda, L. Mastromoro, and D. O. Popa, "Multi-modal sensor and HMI integration with applications in personal robotics," Next-Generation Robotics II; and Machine Intelligence and Bio-inspired Computation: Theory and Applications IX, 8 (2015).

[16] "KUKA youBot User Manual," 6 December 2012, ftp://ftp.youbot-store.com/manuals/KUKA-youBot_UserManual.pdf (5 March 2016).

[17] "NI roboRIO – RIO device for Robotics," 1 July 2015, http://www.ni.com/pdf/manuals/375274a.pdf (5 March 2016).

[18] "D6T MEMS Thermal Sensors," 31 August 2012, http://www.omron.com/ecb/products/sensor/11/d6t.html (5 March 2016).

[19] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, "ROS: an open-source Robot Operating System," ICRA Workshop on Open Source Software, (2009).

[20] I. A. Sucan and S. Chitta, "MoveIt!", http://moveit.ros.org (5 March 2016).

[21] "ROSJava," 1 March 2015, http://wiki.ros.org/rosjava (5 March 2016).