HHS Public Access

Author manuscript

Proc Hum Factors Ergon Soc Annu Meet. Author manuscript; available in PMC 2021 December 07.

Published in final edited form as:

Proc Hum Factors Ergon Soc Annu Meet. 2017 September ; 61(1): 808–812.

doi:10.1177/1541931213601695

DEVELOPMENT AND USABILITY TESTING OF A REMOTE CONTROL APP FOR AN INTERACTIVE ROBOT

Eric Vasey[1], **Maryam S. FakhrHosseini**[1], **Zhi Zheng**[1], **Chung-Hyuk Park**[2], **Ayanna Howard**[3], **Myounghoon Jeon**[1]

[1]Michigan Tech, Houghton, MI

[2]The George Washington University, Washington, DC

[3]Georgia Institute of Technology, Atlanta, GA

Abstract

Experimenters need robots that are easier to control for experimental purposes. In this paper, we conducted interviews for eliciting interaction requirements for human-robot interaction scenarios. User input was then incorporated into an Android application for remotely controlling an Aldebaran Nao robot for use in Wizard-of-Oz experiments and demos. The app was used in a usability study to compare it with an existing Nao remote control app. Results were positive, highlighting the ease-of-use and organization of the app. Future work includes a more complete usability trial evaluating the unique functionality of the app, as well as a case study of the app in a real Wizard-of-Oz experiment.

INTRODUCTION

When conducting a research study involving robots, we often need to use a Wizard-of-Oz experimental design. This involves a human researcher controlling the actions of a robot from afar, as some robot actions are difficult to perform autonomously, with sufficient robustness and repeatability for user studies. However, creating the actions and motions for the robot requires a degree of programming skill and technical knowledge, which some researchers may not necessarily have.

In this study, we aim to develop a general purpose Android app that will enable the remote operation of an Aldebaran Nao robot, a humanoid robot commonly used in human-robot interaction research. First, we will discuss related work involving the remote control of robots using mobile devices. The procedure section covers our initial requirements elicitation interviews; mocking up, designing, and implementing the app; as well as our usability evaluations that compare our app to an existing Android app for remotely controlling the Nao. We then discuss our findings, as well as our future plans for continuing to improve the app.

Related Works

The remote operation of robots using mobile applications has been successfully implemented in other experiments. Trigo and Brown [10] developed an app for Android and iOS to help teachers remotely control a Nao robot to assist in teaching children

with intellectual disabilities. This app could make Nao change its posture and gesture, walk, ask user-defined questions as well as execute several pre-stored dance routines. An evaluation was done using a think-aloud protocol and a questionnaire with several teachers, and overall feedback was positive. However, this app did not allow for creation of new animations within the app or for teachers to record their own motions (M. Trigo, personal communication, October 31, 2017).

Krofitsch et al. [7] developed an Android app for remote control of a custom Arduino (microcontroller board for rapid prototyping) based robot using a custom communications protocol. This app was intended as an app-based Integrated Development Environment for fast development, deployment, and testing of programs. While a formal evaluation was not completed, the researchers collected usability feedback during demonstrations of the app, and children were able to use the app to control the robot.

In another study, Miskam et al. [8] developed a specialized Android remote control app for the Nao robot to help teach children with autism how to recognize emotions. The app contained pre-defined emotional expressions and poses that could be executed by tapping buttons in the app, as well as a connection to the Nao camera, the ability to sit down or stand up, and to play a random greeting. Evaluation was done with children and their therapists. Children were able to identify the emotions being exhibited by the robot, while therapists were able to easily remotely operate the robot with the app. This app did not allow for the users to modify the emotional expressions or poses – modification of the poses required the intervention of a programmer. This app also lacked general control of poses, gestures, or Text-To-Speech (TTS) control, and did not include the ability to save a set of pre-recorded actions for later execution.

To fill in the gaps in previous research efforts, in this paper we discuss the development of a new remote control Android app designed to help users control Nao more flexibly and thoroughly. The new app is designed to help users overcome some of the limitations found in existing app control methodologies.

PROCEDURE

Requirements Elicitation

Before starting development, a set of vital initial features had to be identified that were represented from the point-of-view of individuals who had previously worked with the Nao robot. To this end, we conducted half-hour long structured interviews with four members of the robotics research laboratory at American Technical University. All participants had prior experience working with the Nao robot through running other experiments, developing software for Nao through Choregraphe (the Aldebaran software used to make custom movement routines), or putting on demos that involved the Nao robot.

The participants were asked to respond to a series of nine prompts designed to directly and indirectly assess typical use cases for the robots in the laboratory, how a hypothetical remote control app could have been used in prior research, and the features such an app would be expected to have (see Table 1).

Based on participant responses, we created a list of essential features, including the ability to change the robot's posture (standing, sitting, crouching, etc.), control the robot's movements using an on-screen joystick control, make the robot speak using Text-To-Speech (TTS), perform hand and arm gestures, remotely execute programs stored on the robot from the mobile device, and view the robot's basic diagnostic and status information (battery charge level, temperatures, messages). In addition to these, we decided to include the ability to create, run, and edit scripts – collections of TTS, poses, and gestures that could be used for Wizard-of-Oz experiments or for demos.

In addition to the design requirements, we incorporated two additional overarching goals when developing the app. First, the app needed to be usable by individuals who were not necessarily computer programmers. Second, the app needed to be responsive – there should be minimal delay between executing an action on the app and having that action executed on the robot.

Apparatus

Nao (Figure 1 is a humanoid robot created by Aldebaran Robotics (now SoftBank Robotics). Nao features 25 degrees of freedom, voice recognition, facial recognition and tracking, text-to-speech, multiple cameras and microphones, and a variety of touch sensors.

Questionnaires

System Usability Scale (SUS).—SUS is a questionnaire with 10 items, each having a five-point Likert scale which ranges from strongly disagree to strongly agree. SUS has been shown to be an effective and reliable tool for measuring the usability of products [2].

Experimenter-developed questionnaire.—We added five additional five-point Likert scale questions ranging from strongly disagree to strongly agree to evaluate other aspects of the application, such as speed of command execution on the robot, visual appeal, overall experience with the app, the intuitiveness of the app, and the organization of the information in the app. There were also five open-form response questions included to evaluate participants' first impressions, likes, dislikes, what they would change, and any additional feedback about each of the apps. A demographics question asking the participants' major was also included.

Design

Based on user requirements, we created mockups of the app interface using Moqups, an online tool for creating wireframes and mockups for websites and mobile apps. It was chosen due to its ease of use and inclusion of different Android views [8]. While creating the design for the app, we attempted to follow the Material Design guidelines set out by Google [5]. A user who had used an Android phone in the past would therefore be able to recognize standard display patterns, enabling them to acclimate to the app more quickly. We chose a tabular navigation pattern for the app, as there are two main components – the remote control and the system information, which would make users aware of the system information aspect of the app while keeping it easily accessible.

Upon opening the app, the user is greeted with a connection dialog, requiring them to enter the Internet Protocol (IP) address of the Nao robot and press the “Connect” button to initiate connection to the robot. Once connected, they are shown two side-swipeable tabs separating the remote control and system information components of the app. We leveraged the Android Action Bar for additional scripting functions in the app, such as loading/saving scripts, executing stored programs, and switching to joystick control of the robot’s movements.

In the remote control tab, tapping anywhere in the bordered box would allow us to type text for the robot to say, and long pressing brings up a dialog for creating a different action. This dialog contains tabs for Volume, Pose, Gesture, Rate, or Pitch actions. Once the user configures the action to their liking, they can tap the “Ok” button, and the action would be shown as a combination of editable text fields for TTS and buttons for other actions. The buttons are tappable to allow for editing, and the text fields can be edited by tapping on them.

The system information tab is a list of values, such as the robot’s battery level, CPU temperature, and temperature of its various motors.

Implementation

Once the mockups were completed, the Android app was developed using the Android Studio Integrated Development Environment. For communication with the robot, we used the NAOqi 2.1 API provided by Aldebaran [1], as it allows for real-time control of the robot without the need of another computer or another program running on the Nao. We used Android layouts, libraries, and frameworks where possible, drawing on the Android support libraries provided by Google for the RecyclerView component that runs the button grid, as well as an open-source implementation of a joystick view [5]. Finally, the feature list was pared down to include only TTS, Volume, Pitch, Rate, Gesture, and Posture actions, the ability to save/load/run scripts, read battery level and CPU temperature, and toggle manual animation and automatic animation (Nao automatically selects gestures based on the text being spoken). Scripts are saved to the phone’s local file system in JavaScript Object Notation (JSON). JSON allows us to save scripts in an easily human-readable format for testing, ease of reading/writing, and is extensible in case we want to save additional data values.

Based on problems and changes that occurred throughout the development process, some of the views changed from mockup to implementation. The only major changes that needed to be made were in the remote control tab (see Figure 2). In this tab, we changed the combination of editable text fields and buttons to a grid of buttons, which could be executed by tapping on them. Long pressing a button in the grid brings up a menu to run the action, edit the action, or delete the action. Not only did this add the ability for users to quickly execute commands using a soundboard-style layout, it eliminated the ambiguity of having multiple types of views (editable text views and buttons) in the same area. It also provided a simple way to edit and remove actions, which the mockup was lacking. Some of the other changes included adding and modifying titles in the dialog for creating/editing actions, as well as miscellaneous text changes and re-ordering of fields.

Usability Testing

To evaluate the usability of the Android app, and to see if it met the stated design goals of responsiveness and ease-of-use by non-programmers, we conducted a pilot study which aimed to compare the ease-of-use of our developed app to an existing app for remote controlling a Nao robot. We selected an appropriate comparison app from the Google Play Store titled *Nao Remote Controller* [3]. This app was the only one we could locate that did not require the Nao robot and the mobile device to be on the same wireless network, that did not rely on an external server to function correctly, and did not require extra code to be deployed to the Nao robot. We only evaluated functionality that was present and functioning in both mobile apps, which restricted us to the connectivity, TTS, and pose functionality.

Considering these limitations, we devised seven usability scenarios for participants to complete (see Table 2). Participants were asked to follow the think-aloud protocol when completing the scenarios. Think-aloud protocol involves participants verbally explaining what they are doing and what they are thinking while completing the scenarios. This allows us to evaluate the paths they take and their reasoning for navigating the app in the way they did. Each scenario was completed with both our remote control app and the comparison app, with the order alternated between participants. The time to complete each scenario was recorded using a stopwatch.

Once the scenarios are completed, participants were asked to fill out a survey using Google Forms asking about their thoughts on the usability of each app. Each app had 20 questions – 15 Likert scale based questions and five open-form questions. Ten of the Likert scale questions were from the System Usability Scale (SUS), a standard tool for investigating the usability of a system [4]. In addition, we collected participants' major and their ultimate choice on app preference, while asking them to justify their choice.

Using these scenarios and post-questionnaire, a usability study was conducted with 19 participants (15 male, 4 female) ages 18 – 22 recruited using the University recruiting system.

RESULTS

The ten usability questions taken from the SUS were scored based on the scoring criteria outlined in [4]. Participants' overall SUS responses were scored for the prototype app ($M = 86.25$, $SD = 11.89$) and the comparison app ($M = 69.44$, $SD = 16.17$). After normalizing the SUS score, we used a Wilcoxon signed-rank. Results showed that there is a significant difference between the two mean SUS scores ($N = 18$, $M_{prototype} = 0.810$, $SD_{prototype} = 0.164$, $M_{comparison} = 0.579$, $SD_{comparison} = 0.223$, $V = 147$, $p < 0.008$).

The mean time to complete the scenarios was analyzed using student's paired t-tests for statistical significance, with five out of seven scenarios showing significant differences. We found that participants connected to the robot in scenario 1 significantly faster in our prototype app ($M_{prototype} = 20.44$, $SD_{prototype} = 4.15$) than the comparison app ($M_{comparison} = 39.75$, $SD_{comparison} = 11.60$, $t = -5.70$, $p < 0.0002$). This pattern was also observed in the initial TTS scenario (scenario 3) ($M_{prototype} = 21.79$, $SD_{prototype} = 6.83$, $M_{comparison} =$

31.79, $SD_{comparison}$ = 5.69, t = −2.92, p < 0.02). In scenario 5, making the robot say “I’m standing!”, participants performed significantly faster with the prototype app ($M_{prototype}$ = 13.44, $SD_{prototype}$ = 0.76), than the comparison app ($M_{comparison}$ = 26.75, $SD_{comparison}$ = 8.06, t = −7.61, p < 1.05E-6). For changing the robot’s pose to crouching (scenario 6), participant’s time to complete the scenario was significantly faster ($M_{prototype}$ = 10.93, $SD_{prototype}$ = 2.41, $M_{comparison}$ = 16.31, $SD_{comparison}$ = 2.35, t = −9.65, p < 4.47E-8). In the final scenario (scenario 7), participants made the robot say “I’m crouching!”, with the data following a similar pattern ($M_{prototype}$ = 11.38, $SD_{prototype}$ = 2.65, $M_{comparison}$ = 19.44, $SD_{comparison}$ = 5.77, t = −10.15, p < 4.13E-8).

For the experimenter developed Likert scale measures, the prototype app ($M_{prototype}$ = 4.37, $SD_{prototype}$ = 0.6), was rated as significantly more intuitive than the comparison app ($M_{comparison}$ = 3.47, $SD_{comparison}$ = 0.90, t = 3.54, p < 0.002). Participants rated the prototype app’s information as being significantly more logically organized than the comparison app ($M_{prototype}$ = 4.37, $SD_{prototype}$ = 0.76, $M_{comparison}$ = 3.37, $SD_{comparison}$ = 1.12, t =3.00, p < 0.008). Participants felt that the prototype app ($M_{prototype}$ = 4.74, $SD_{prototype}$ = 0.45) executed their commands in a timelier manner than the comparison app ($M_{comparison}$ = 3.61, $SD_{comparison}$ = 1.09, t =3.56, p < 0.002). Overall, more participants felt their interaction with the prototype app was better than with the comparison app ($M_{prototype}$ = 4.53, $SD_{prototype}$ = 0.61, $M_{comparison}$ = 3.83, $SD_{comparison}$ = 0.79, t = 0.02, p < 0.02).

DISCUSSION

Results of this study showed that participants rated our app higher on the SUS, compared to the existing app. This suggests our app is easier to use, more user friendly, and has better-integrated functions than an existing app. Participants’ behavioral data (the amount of time they spent completing a task) showed a similar pattern. Users accomplished tasks faster with our app allowing for more efficient interaction with Nao.

Based on the open-form answers in the questionnaire, the participants showed a definite preference for our app as compared to the comparison app. Based on first impressions and positive responses about the app, participants praised the simplicity and organization of the user interface, ease of use, and the grid-based button layout for executing commands. One participant remarked that “Connecting was straight forward and easy, and the command system was fairly simple to get a grip on.” For negative feedback and personal preferences about changes, participants felt that the user interface needed polishing “I did not like how it looks. It looks clunky in my opinion. People like to see easy, minimal, and clean interfaces, and that they had to guess on how to add actions to the app”, “It took a guess to figure out that the + icon needed to be pressed to create a command”. Multiple participants suggested the addition of a tutorial to familiarize users with the app’s functionality, with one participant suggesting “add a tutorial so someone wouldn't get confused”.

Future Work

While the current functionality of the app is sound and the organization of information is good, participants felt that the user interface of the app needed additional work, and the app needed a tutorial to familiarize users with the app’s functions. To address this, we plan to

investigate changes to the layout and color palette, and the addition of a splash screen, help page, or guided tutorial. In addition, we plan to implement some of the additional features that were delayed, such as joystick control and remote execution of stored programs. Once our changes are implemented, we plan to run another usability evaluation covering the unique functionality of this app, since this study focuses on comparison with an existing app. In addition, we aim to investigate the use of the remote control app in the context of a Wizard-of-Oz experiment by observing how the app is used.

CONCLUSION

Based on our results from the usability study, we plan to continue improving and developing this mobile app. This unique combination of a mobile Android app that allows for direct communication and control of a Nao robot, without requiring an external server or program has many useful applications when conducting Wizard-of-Oz experiments or demos. We hope that this app will give researchers a fast, easy, and reliable tool to conduct their experiments.

ACKNOWLEDGMENTS

This project is supported by the National Institutes of Health under grant No. 1 R01 HD082914-01.

REFERENCES

1. Aldebaran. (2015). NAOqi APIs – Aldebaran 2.1.4.13 documentation. Retrieved from http://doc.aldebaran.com/2-1/naoqi/index.html
2. Bangor A, Kortum P, & Miller J (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. Journal of usability studies, 4(3), pp.114–pp.123.
3. Bonnes R (2016). NAO Robot Controller – Android Apps on Google Play. Retrieved from https://play.google.com/store/apps/details?id=com.robinbonnes.naorobotcontroller
4. Brooke J (1998). SUS – A quick and dirty usability scale. Retrieved from https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html
5. erz05. (2016). erz05/JoyStick: Android Library for JoyStick View. Retrieved from https://github.com/erz05/JoyStick
6. Google. (2016). Introduction – Material design – Material design guidelines. Retrieved from https://material.io/guidelines/
7. Krofitsch C, Hinger C, Merdan M and Koppensteiner G, (2013). Smartphone driven control of robots for education and research. 2013 IEEE International Conference on Robotics, Biomimetics, and Intelligent Computational Systems (ROBIONETICS), pp.148–pp.154.
8. Miskam MA, Shamsuddin S, Ridzuan M, Samat A, Yusso H, Ainudin HA, and Omar AR. (2014). Humanoid robot NAO as a teaching tool of emotion recognition for children with autism using the Android app. 2014 International Symposium on Micro-NanoMechatronics and Human Science (MHS), pp.1–pp.5.
9. Moqups. (2016). Online Mockup, Wireframe, and UI Prototyping Tool. Retrieved from https://moqups.com/
10. Trigo MJG and Brown DJ (2014) Remote Operation of Robots via Mobile Devices to Help People with Intellectual Disabilities, 2014 International Conference on Interactive Technologies and Games (iTAG), pp.1–pp.8.

Figure 1:
Nao

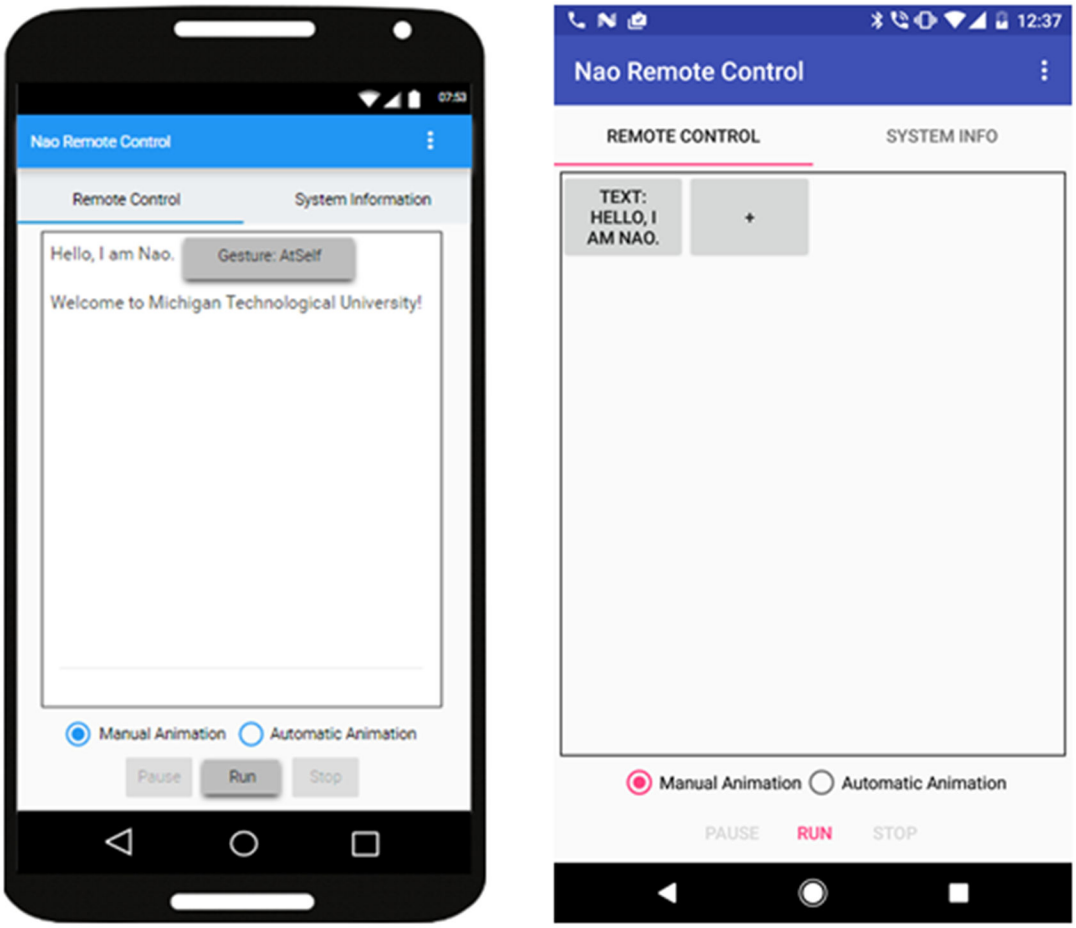

Figure 2:
Changes from mockup (left) to implemented app (right) for remote control tab

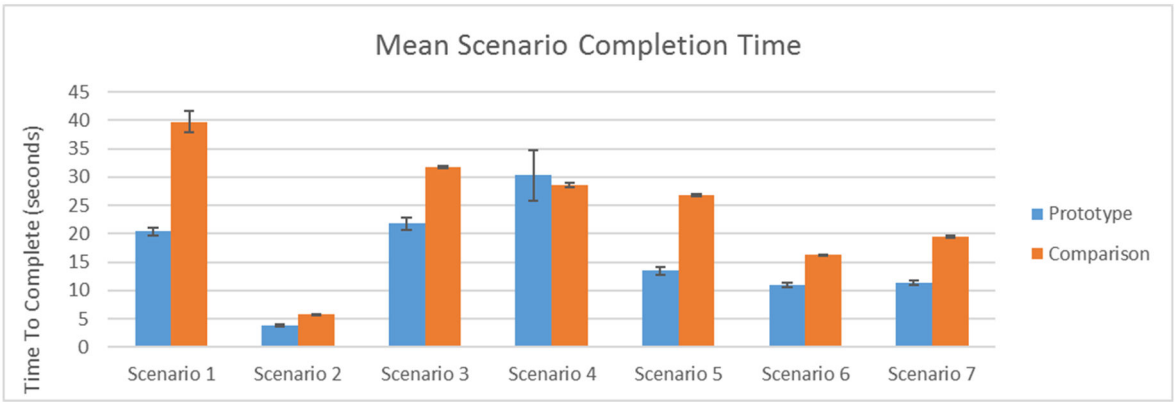

Figure 3:
Mean Time to Completion Graph (Blue is our app, Orange is comparison app)

Table 1:

Requirements Elicitation Interview Prompts

Prompts
How do you prepare for a demo session with a robot?
Walk me through a demo session. What sorts of features would you want to show off?
Describe a typical demo environment. Is power available? Internet? Hazards?
Are others allowed to interact with the robot (physically, verbally, etc.)? Why?
What considerations are made when preparing for an experiment involving robots?
What feature-sets are most often utilized when doing research involving a robot? How are those feature-sets utilized?
How could a remote control app have been used in the Child-Robot Theater project?
What would you expect the abilities of a robot remote control app to be?
What would you personally want to see in a robot remote control app?

Table 2:

Usability Scenarios

Scenario	Description
1	Connect to Nao.
2	Read Battery Level & CPU Temperature.
3	Make Nao say “Welcome to XX University!”
4	Make Nao stand up.
5	Make Nao say “I’m standing!”
6	Make Nao crouch.
7	Make Nao say “I’m crouching!”