Smartphone-Controlled Telerobotic Systems

Hyemin Ahn, Hyunjun Kim, Yoonseon Oh, and Songhwai Oh

***Abstract*—This paper proposes a telerobotic system based on a smartphone and Nao, a humanoid robot from Aldebaran Robotics. A user can control the robot using her smartphone and interact with people and surroundings around the robot in a remote location. The overall system includes two servers to facilitate the connection between the user's smartphone and the robot. We have particularly focused on providing a user-friendly interface such that a user who is unfamiliar with the robot platform can control the robot intuitively.**

***Index Terms*—Telerobotics, Humanoid robot, Smartphones**

I. Introduction

A telerobotic system, which allows human operators to extend their sensing and manipulation capability to a remote environment [1], is a field of robotics that has received attention in recent years.

There exists a number of telerobotic systems for enabling people to feel the presence at a remote location. Commercial robots, such as QB from Anybots [2] and VGo from VGo Communication [3], enable users to attend a conference in a remote location. These wheeled robots help communication between a user and people in the vicinity of the robot by providing a video feed from the robot's perspective to the user and displaying the user's face on the robot. Paulos and Canny demonstrated a telerobotic system which allows remote scholars to view museum artifacts and specimens on demand [4]. It assists multiple users to explore a remote environment by controlling a robot arm with an attached camera.

A humanoid robot is also used for telerobotics. Nao, a humanoid robot from Aldebaran Robotics [5], has been used as a telerobotic platform in [6], [7]. According to [8], the appearance of a robot is one of the key factors that help a robot to be recognized as a human-like entity. Hence, a humanoid robot can be a highly effective platform for telerobotics when its user tries to start an interaction with people nearby the robot.

This paper presents a telerobotic system for controlling a humanoid robot Nao using a portable smartphone. Figure 1 shows an overview of the proposed system. A user of this system can manipulate the robot using an Android smartphone by observing the robot's surroundings from a video feed played on the smartphone display. The main strengths of the proposed telerobotic system are a human-like remote robot platform and the portability and availability of the smartphone-based controller. A humanoid robot can increase the familiarity with people in the vicinity of the robot. A remote user can access the system at any time using her smartphone.

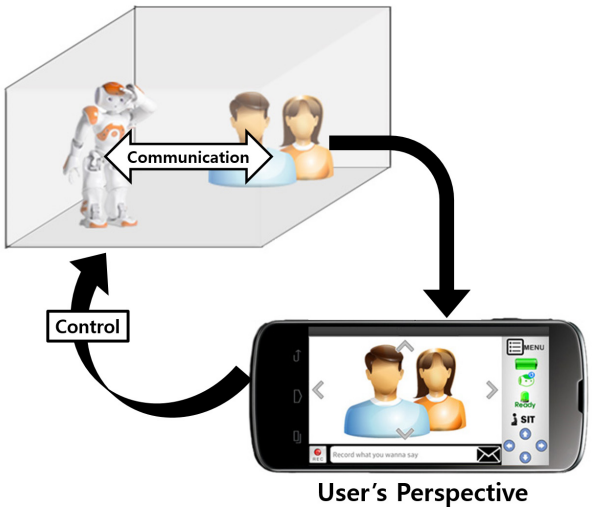


Fig. 1. An overview of the proposed telerobotic system.

The remainder of this paper is organized as follows. After discussing desirable features of the target system in Section II, we provide an overview of the system in Section III. In Section IV, we describe the Android application developed for controlling a robot and its user-friendly interface features. A few implementation issues are discussed in Section V.

II. Smartphone-Controlled Telerobotic Systems

The following are some desirable features of a smartphone-controlled telerobotic system:

- a low latency of data transfer between the robot and the user,
- portability and availability of a remote controller, and
- a user-friendly interface designed for a smartphone.

Since a user controls a robot relying only on the streamed video and sensor data from the robot, a low latency is critical for a successful telerobotic system. The experience of a telerobotic system can be improved if users can access the telerobotic system regardless of their location, hence, the portability of a remote robot controller is important. In addition, the robot controller's interface needs to be intuitive such that a user who is not familiar with a robot can operate easily.

This work was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2009348).

H. Ahn, H. Kim, Y. Oh, and S. Oh are with CPSLAB and ASRI, Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea (e-mails: {hyemin.ahn, hyunjun.kim, yoonseon.oh, songhwai.oh}@cpslab.snu.ac.kr).

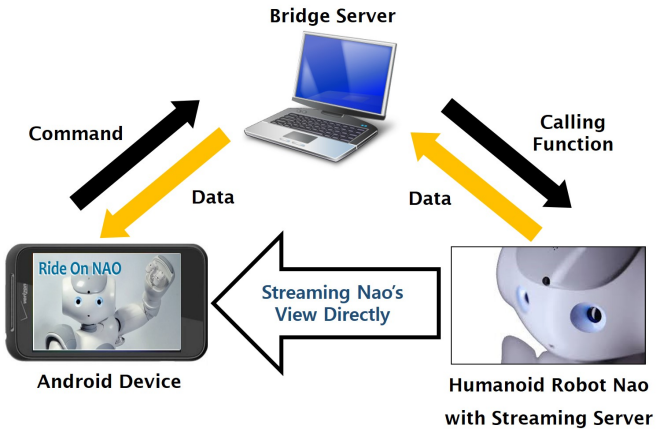


Fig. 2. The system architecture of the proposed telerobotic system.

III. Overview of the System

In this section, we describe the overall structure of the proposed telerobotic system and communication mechanisms between components of the system.

A. System Architecture

The system architecture of the proposed telerobotic system is shown in Figure 2. The system consists of an Android smartphone, a humanoid robot, a bridge server, and a video streaming server inside a humanoid robot. The use of a smartphone in our system makes the overall system more portable, accessible, and available. In our system, a humanoid robot from Aldebaran Robotics [5], called Nao, is used as a remote robot platform.

Currently, a direct communication between an Android device and a Nao robot is not possible. We address this issue by introducing a bridge server between Nao and an Android device for transferring data and commands between them. In addition, an independent server in Nao is implemented in order to provide real-time video streaming to the user.

B. Communication

The overall communication flow of the proposed system can be seen from Figure 2. Various features of the Android application are mapped to API functions provided in [9]. When user commands are entered into the Android application, the encoded commands are sent to the bridge server. The bridge server interprets and serializes the user commands. The ordered commands in the form of API function calls are sent to Nao for execution. If a return value is expected after the execution, the server waits for data from Nao before transferring the returned data back to the Android application.

The second server embedded inside a Nao robot is designed for video streaming only. While it is possible to stream a video from Nao to the Android application using NAOqi [9], [10] and the bridge server, the frame rate was too slow to be used for a telerobotic system. It was even below two frames per second for the worst case and caused disturbing latency problems. Therefore, an independent server is introduced for video streaming.

When the Android application is connected to the bridge server, the bridge server passes the IP address of the Android device to Nao. After receiving the IP address, the video streaming server at Nao starts streaming video captured from the camera of the robot to the Android device.

C. System Specification

In our implementation, a Google Nexus 4 smartphone is used as an Android device. The bridge server is implemented on a desktop using Winsock [11]. The server is developed using C++ for its high compatibility with the NAOqi SDK [9]. The video streaming server is embedded in Nao using the real-time transport protocol (RTP) provided by Gstreamer [12] in order to stream images captured by Nao to the Android device in real time. The video is streamed with a resolution of 320×240 pixels at the frame rate of 24 frames per second. On the Android device, the Gstreamer SDK for Android [12] is used to display the received video feed to the user.

IV. Android Application

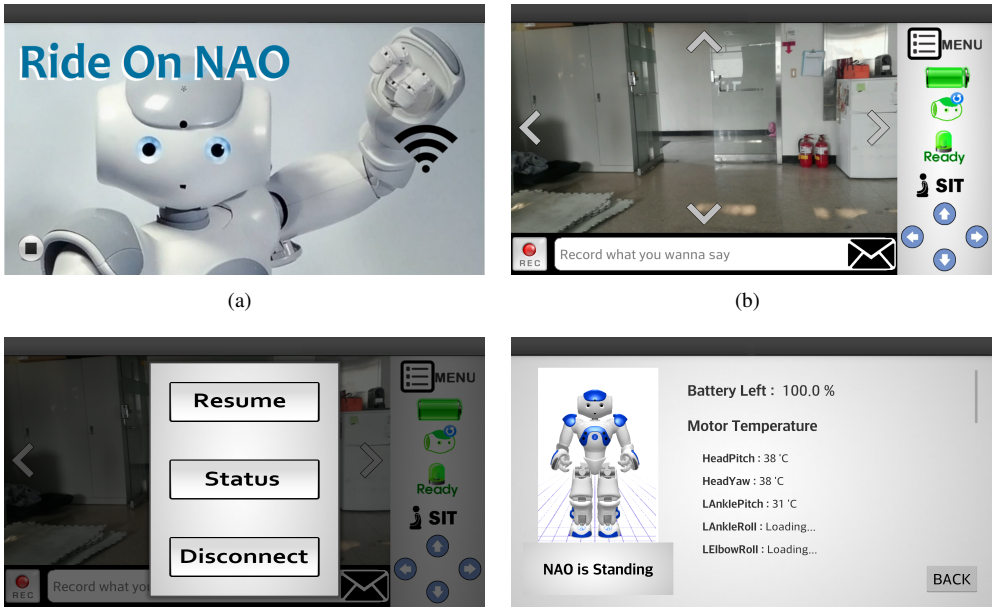
We first describe the Android application for the proposed telerobotic system and then discuss major features of the application and their usability.

Some screenshots of the developed Android application are shown in Figure 3. When a user opens the application, the opening screen shown in Figure 3(a) appears first. If a user touches the `Wi-Fi` button in the right, a popup menu shows up for entering the information such as the IP address of the bridge server in order to access a Nao robot.

The main screen for controlling a Nao robot is shown in Figure 3(b). From this main screen, the user can view what the Nao robot is viewing in real time. At the same time, the user can move the robot and also change the robot's gaze directions. In addition, the user can send a message to the robot such that people near the robot can hear the message in a synthesized voice. Note that Nao provides an API for converting text to voice [13].

When a user presses the `Menu` button on the right of the main screen, a popup menu shows up as shown in Figure 3(c). This menu provides choices such as returning to the main screen (`Resume`), checking the current status of the robot (`Status`), and disconnecting from the robot (`Disconnect`). If the user presses the `Status` button, a status screen, such as Figure 3(d) will be displayed to the user. On the left of the status screen, the current pose of the robot is displayed. On the right of the status screen, current robot's sensor values are displayed. This panel can be scrolled down to read sensor values, such as the battery level and temperatures of Nao's 26 motors.

If the user presses the `Disconnect` button from the menu shown in Figure 3(b), the application disconnects from Nao and returns to the startup screen.



(a) (b) (c) (d)

Fig. 3. Screenshots of the developed Android application. (a) A startup screen. (b) The main screen of the application. The live view from the robot is shown in the middle. A display for the robot status and control commands are on the right of the screen. A user can also enter texts or record a speech into the bottom to be recited by the robot. (c) A popup display when the `Menu` button is pressed. (d) A status screen of the robot when the robot is in the standing position.

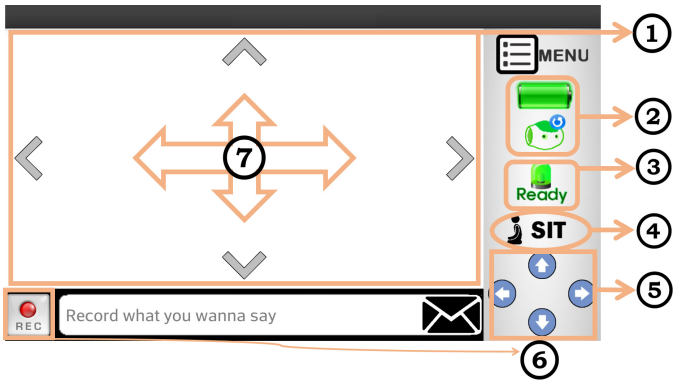


Fig. 4. Components of the main screen of the Android application.

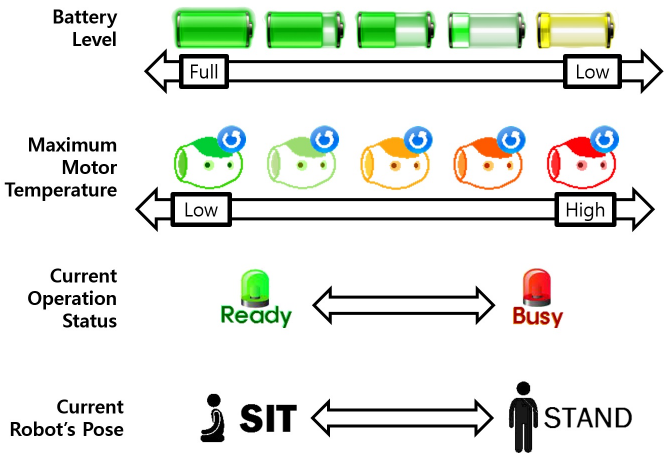


Fig. 5. Icons used in the main screen.

A. User Interface Design

We have determined a number of design principles for designing a user-friendly interface as follows:

- To make an important section occupy a large space and locate it in the center.
- To use graphics rather than text to convey information to assist users to understand the current status of the robot intuitively.
- To display only necessary information in the main screen. Too much information only distracts users.

We have focused on making the main screen as user-friendly as possible since this is the territory users will occupy more often. Figure 4 shows the main screen in detail. Partition 1 shows the streamed video from the robot's camera and takes the largest part of the screen. This is because a user interacts with the robot through the view of the robot and the current view of the robot is the most important aspect of the application.

Partitions 2 and 3 in Figure 4 shows a quick summary of the status of the robot. Rather than using text or sensor values, we have adopted graphical icons. With these icons, users can recognize the current status of the robot instantaneously. The details of these icons are summarized in Figure 5. Two icons included in Partition 2 of Figure 4 are graphically representing the battery level and the maximum temperature of Nao motors. A variable color scheme is applied to indicate the levels of sensor values as shown in Figure 5. Sensor values are regularly updated. The icon in Partition 3 of Figure 4 indicates whether the Nao robot is ready for performing user commands. If the

robot is in the middle of previously commanded operations, the icon's color turns red and the text below the icon will be changed to `Busy` as shown in Figure 5. In addition, all buttons for operating Nao are disabled while Nao is executing the current command. The `Sit` button in Partition 4 is to change Nao's current pose from the standing position to the sitting position. Once Nao is sitting down on the floor, the button will be changed to `Stand` for standing up.

B. Control Commands

The control commands provided in the main screen can be grouped into two types:

- Commands for controlling the pose and position of the robot.
- Commands for sending a message to people nearby the robot.

The first type of commands can change the pose and position of the robot. A user can change the viewpoint by rotating the robot head and this can be done by pressing an arrow shown in Partition 7 of the main screen (see Figure 4). If a user presses the arrow pointing to the left, the robot will turn its head to the left direction. For other directions of the arrow, the robot turns its head into the corresponding direction and the viewpoint changes accordingly. To prevent the collision between Nao's head and body, we restrict the range of its head rotation angle in both vertical and horizontal directions. A user can also move the robot in all four directions by pushing buttons located in Partition 5 of the main screen as shown in Figure 4. In our implementation, the forward direction is defined as the direction of the gaze of the robot rather than the orientation of the body of the robot in order to provide a more intuitive control of the robot with respect to the user's view.

The second type of commands are related to sending a message to people near the robot. The `REC` button in Partition 6 of the main screen shown in Figure 4 is used to record a voice message from the user. When the button is pressed, the application asks the user to speak a message then the message is recorded. Then the Google STT (speech to text) module is activated to convert the voice message to a text message. The text message is sent to the Nao robot when the user pushes the envelope shaped button. Once a text message is received by Nao, Nao vocalizes the message using its TTS (text to speech) module [13]. A user can also type in a text message and the typed message will be vocalized by Nao in a similar way. This function enables the user to interact with people in the vicinity of Nao. Currently, the system supports only English but we plan to support other languages in the future.

V. Implementation Issues

We have discovered a few unexpected problems while testing the proposed system. The first problem was the battery level of Nao. Even with 30 percent battery remaining, the Nao robot automatically shuts down its power. We addressed this problem by displaying a low battery warning notification when the battery level is below 40 percent.

Another problem occurred when we tried to stream the audio from Nao. Even if we used the platform provided by Gstreamer [12], direct audio streaming was not possible. Nao's onboard soundcard does not allow an access by another program when Nao's TTS module is used. We expect that the problem can be solved by authorizing the control of Nao's soundcard for audio streaming.

VI. Conclusions

In this paper, we have developed a telerobotic system using a smartphone and a humanoid robot. Since there is no framework available for a direct communication between a smartphone and a humanoid robot, we have introduced two servers in order to exchange data and user commands. A separate video streaming server was necessary for real-time streaming of the video. An Android application is developed to provide a user-friendly interface for the users of the system. The user can view what the robot is seeing through the Android application. The application also allows a user to control the robot and monitor the status of the robot and various sensor values. We plan to enhance the system by enabling the audio streaming capability. In addition, a direct communication without a bridge server is under consideration. With these improved features, we envision that the proposed system can be an attractive solution for many telerobotic applications.

References

[1] T. B. Sheridan, "Telerobotics," *Automatica*, vol. 25, no. 4, pp. 487–507, 1989.

[2] "Anybots." [Online]. Available: http://www.anybots.com/

[3] "Vgo." [Online]. Available: http://www.vgocom.com/

[4] E. Paulos and J. Canny, "Delivering real reality to the world wide web via telerobotics," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 1996.

[5] Aldebaran Robotics. "Nao robot: intelligent and friendly companion". [Online]. Available: http://www.aldebaran.com/en/humanoid-robot/nao-robot

[6] M. Markovich, "Nao robot as remote avatars," 2011. [Online]. Available: https://apps.foe.auckland.ac.nz/p4p/ece/downloads/Part4Reports2011/IntelligentSystems/project-24-mmar149.pdf

[7] R. Brown, H. Helton, A. Williams, M. Shrove, M. Milosevic, E. Jovanov, D. Coe, and J. Kulick, "Android control application for Nao humanoid robot," in *Proc. of the International Conference on Frontiers in Education: Computer Science and Computer Engineering*, 2013.

[8] E. Park, H. Kong, H.-t. Lim, J. Lee, S. You, and A. P. del Pobil, "The effect of robot's behavior vs. appearance on communication with humans," in *Proc. of the ACM/IEEE International Conference on Human-Robot Interaction*, 2011.

[9] Aldebaran Robotics. "NAOqi Framework - NAO Software 1.14.5 documentation". [Online]. Available: https://community.aldebaran-robotics.com/doc/1-14/naoqi/index.html

[10] ——. "ALVideoDevice API - NAO Software 1.14.5 documentation". [Online]. Available: https://community.aldebaran-robotics.com/doc/1-14/naoqi/vision/alvideodevice-api.html

[11] B. Quinn, *Windows sockets network programming*. Addison-Wesley Longman Publishing Co., 1998.

[12] "GStreamer: Open Source Multimedia Framework." [Online]. Available: http://gstreamer.freedesktop.org

[13] Aldebaran Robotics. "ALTextToSpeech API - NAO Software 1.14.5 documentation". [Online]. Available: https://community.aldebaran-robotics.com/doc/1-14/naoqi/audio/altexttospeech-api.html