Remote Control Application for "Servosila Engineer" on Android Mobile Devices

Daniel Kiryanov, Roman Lavrenov
Laboratory of Intelligent Robotic Systems (LIRS), Intelligent Robotics Department,
Higher Institute for Information Technology and Intelligent Systems,
Kazan Federal University, 35 Kremlyovskaya street, Kazan, Russia Federation
E-mail: danielkiryanov@gmail.com, lavrenov@it.kfu.ru
http://kpfu.ru/robolab.html

Abstract

Even though modern mobile robots' autonomous navigation capabilities rapidly increase, teleoperation mode is still an important tool, especially in critical domains like rescue or military robotics. This paper presents Android OS based teleoperator control tool for Russian crawler robot Servosila Engineer. We changed the way of data exchange between a robot and its operator, which allows using Wi-Fi data standards in order to simply data transfer from OCU (Operator Control Unit) process to vehicle process. Our application provides robot remote control and video data transfer from robot onboard cameras.

Keywords: crawler robot, OS Android, remote-control tool, UDP, GUI, user interface.

1. Introduction

Today mobile devices are equipped with hardware that is not inferior in performance to computers. For example, smartphones have a wide range of built-in sensors such as an accelerometer, gyroscope, GPS, etc. High performance and multi-threading of mobile processors provide high-speed programs. Increased RAM and physical memory allow processing a large amount of incoming data, while growing capacity of batteries increases devices' autonomy. All together these allowed using mobile devices in many areas of modern information technologies, including robotics.

The common ground between mobile development and robotics can be, for example, augmented reality, robot interaction with a mobile device, robot control via a mobile application, including a remote-control tool via Wi-Fi. In the later case, it's important to consider interaction between the systems.

Nowadays a large number of developments in this area exist. One of them is ABR (Android Based Robotics) project[1]. It has both client and server software. Android app used to connect to a TCP server (ABR_server) and allow streaming of data and remote control. Connection between Android client and the server is via UDP (User Datagram Protocol)[2] protocol. The project demonstrates processes of connecting and receiving data from a TCP server, sending data over UDP sockets, capturing frames from a phone's camera for streaming etc. Server is a QT (C++) project that runs on a PC and can receive connections from multiple phones (ABR_client), display video feedback and sensory information and serve to control robots remotely. Another example is "ROS Control" tool[3]. It is a universal tool using ROS Android. "ROS Control" has multiple teleoperation control options including joystick and tilt control, waypoint planning and navigation, remote camera view, laser scan visualization, GPS data and map view.

In our research, we developed Android OS based remote-control tool for Servosila Engineer robot[4] (Fig.1) using Android SDK only. This was possible due to the software on a vehicle process, which allows communicating between the robot and a smartphone using standard data transmissions[5].

2. Servosila Engineer Robot

The mobile robot Engineer (Fig.1) is developed by Russian company "Servosila" for search and rescue operations in natural and man-made disasters[6]. It is used when receiving remotely a video data from inside a dangerous room is required[7]. Special solutions embedded in the design of the robot chassis allow the robot to enter objects that are located inside industrial and residential buildings. Servosila Engineer is equipped with a powerful manipulator arm, which allows the robot to remove samples of dangerous objects from a disaster area for a detailed analysis or immediately perform a dangerous engineering operation on-site to eliminate or prevent further growth of a man-made disaster. Table 1 briefly describes robot equipment.

Fig. 1. Servosila Engineer servo drive scheme.

3. Android OS

Android operating system is Linux-based and has Google's own Java virtual machine implementation. Applications for Android OS are programs in non-standard byte code for Dalvik virtual machine, for which format of installation packages APK was developed.

Table 1. Robot equipment.

Sensors	Types
Cameras	24x optical camera
	Rear view camera
	Two front cameras for stereo vision
Thermal imager	External
Auto Navigation	Laser scanner
	GPS/GLONASS receiver
	Stereo cameras

Many libraries are available for working with applications: Bionic (library of standard functions), OpenGL (3D graphics engine), SQLite (a lightweight DBMS available for all applications), SSL (protocol for secure data transmission over the network), etc.

4. Existing tools

Programmers can develop software for Android in Java using the SDK or in a native language (C/C++) using the native development kit (NDK). It is also possible for developers to modify the Linux kernel if needed. Implementation of an Android application can be achieved using Android Studio IDE or Eclipse IDE with Android Development Tools (ADT) plug-in. Using this SDK, a developer obtains an easy access to different functionalities of Android phone such as graphical interfaces, multi-threading, networking, data storage, multimedia, sensors, location provider, speech-to-text, text-to-speech, and more. Since Android phones can connect to the Internet, cloud-based applications can also be used when high-performance computing is needed. In robotics, this feature can allow cloud-based robotics applications development. When developing an application that is CPU-intensive but doesn't allocate much memory, an alternative programming option is to use Android NDK. With the NDK, a programmer can create an Android Java application that interacts with native code (C/C++) using Java Native Interface (JNI)[8].

Programming in C/C++ on an Android platform can result in an increase of performance, but also increases complexity. The NDK also enables usage of existing C/C++ libraries. This maked possible exporting popular libraries (e.g., computer vision library OpenCV) to Java so that they can be incorporated in Android applications (OpenCV Android). The robot operating system (ROS) is also available for Android in Java (ROS Java, ROS Android)[9]. An example of SDK is "Pepper SDK"[10], which provides an Android Studio plug-in to develop human-robot interaction tools for Pepper Humanoid Robot. There are many Android applications for robots on Arduino hardware platforms[11]. Android OS mobile devices can also act as a hardware platform due to series of open-source PIC microcontroller-based boards "IOIO"[12]. IOIO allows Android mobile applications to interact with external electronics. Its hardware and

software are entirely open source and enabled the creation of hundreds of DIY robotic projects around the world.

Frame Type ID	Axis #0	Axis #1	...	Axis #15	Button #0	Button #1	...	Button #15	Video Bit Rate Telemetry
1byte	2bytes	2bytes	...	2bytes	1byte	1byte	...	1byte	8bytes
uint8	int16	int16	...	int16	uint8	uint8	...	uint8	double

Fig. 2. Remote control packet structure.

5. Remote control

The above technologies are the result of porting software to Android. In order to interact with the robot in another environment without using third-party libraries, it is necessary to make an interactor on an on-board control computer of the robot. Therefore, we developed a communication system between the vehicle process and OCU process using data transfer over UDP, which was selected because of ease of use. For example, the OCU process sends a request to the vehicle process and hopes to get a response. The client after a certain time may try again if the request or response is lost. It allows developing easier code and reducing a required number of messages in comparison with protocols that request initial configuration. In order for the parties to understand each other, it is also necessary to determine a format of a transmitted message. Figure 2 shows a remote-control packet structure that consists of the following parts[2]:

- frame type id – value indicating a type of message being sent (e. g. remote-control packet)
- axis – fields are used to indicate a speed of servo drives' rotation
- buttons – fields are used for giving an amperage to servo drives
- video bit rate telemetry – section using to set a number of bits used for video data transmission/processing

Due to this structure, the OCU process can control all Engineer's servo drives (Fig. 1) movements. Therefore, any environment can act as a remote-control tool. We used Android OS as one of them.

6. Mobile Application

We designed a simple and intuitive interface when creating this tool. Robot servo drives were divided into groups "Movement" and "Joints" and placed in the separate windows (Fig. 3). There are options to set their

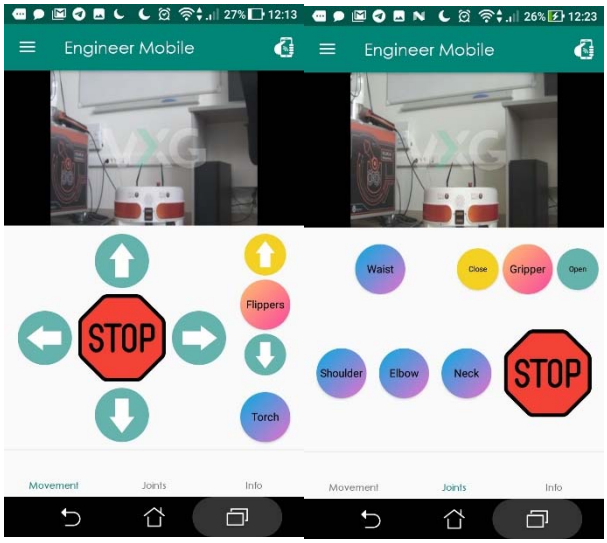

Fig. 3. "Movement" application window (left) and "Joints" application window (right).

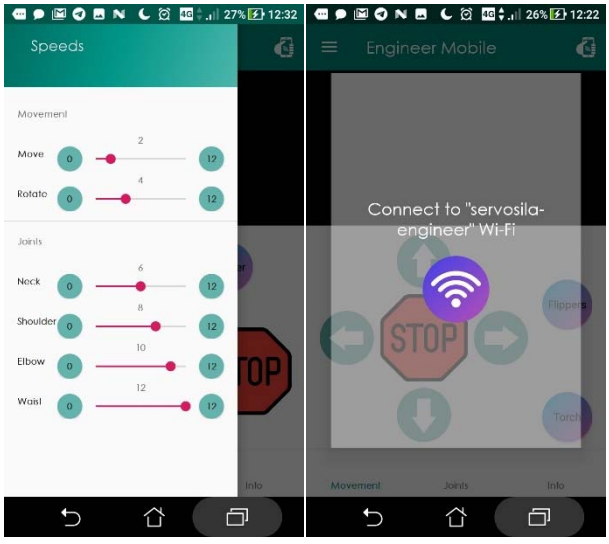

Fig. 4. "Speed" sidebar (left) and "Connection" dialog (right).

speed in the sidebar (Fig. 4). It's useful when it's important to change the speed while the robot is moving. Also, the stop button was provided (Fig. 3). We added video processing from the front camera via RSTP (Rapid Spanning Tree Protocol). The mechanism associated with transmitting the remote-control packet over the network is initialized in a separate process called "Service" (Fig. 5). It allows sending the remote-control packet to the server several times per second and does not affect the UI. The packet is formed inside the PackageManager and transmitted to the UDP client which uses the standard Java mechanism called "DatagramSocket". UI consists of MainActivity and Fragments that can be interchanged. Robot Servosila Engineer has DHCP server for connection, so the app can get its IP, form it in Utils and transfer it to the UDP Client.

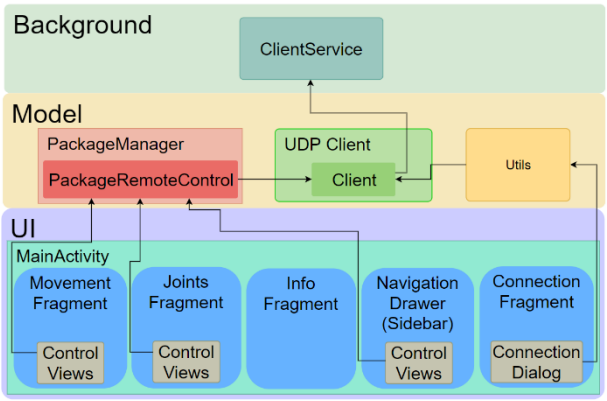

Fig. 5. Application data sharing scheme.

7. Conclusions and Future Work

In this paper, we presented Android OS based teleoperator control tool for Russian crawler robot Servosila Engineer. We developed software for the onboard control computer of the robot. It receives structured remote-control packets by UDP, so we designed a remote-control application using this standard. This tool allows controlling the robot's movement and servo drives. The application receives a video stream from the front camera via RSTP. As a part of our future work, we will supply the application with telemetry data and video streams from other cameras and use Google maps to represent the robot geolocation.

Acknowledgements

This work was supported by the Russian Foundation for Basic Research (RFBR), project ID 19-58-70002.

References

1. Nicolas Oros, Jeffrey L. Krichmar. Android™ Based Robotics: Powerful, Flexible and Inexpensive Robots for Hobbyists, Educators, Students and Researchers. [Web Link].URL:https://www.socsci.uci.edu/~jkrichma/ABR/index.html.
2. Mavrin, I., Lavrenov, R., Magid E. Development of a Graphical User Interface for a Crawler Mobile Robot Servosila Engineer, 11th International Conference on Developments in eSystems Engineering (IEEE, 2018), p. 192-197.
3. I. Rekleitis. Android app for remote controlling a ROS robot. [Web Link]. URL: https://mtbii.github.io/RobotCA
4. Sokolov, M., et al. Modelling a crawler-type UGV for urban search and rescue in Gazebo environment. In International Conference on Artificial Life and Robotics (2017), pp. 360-362.
5. Li, H., et al. Achieving Position Synchronization in Passive Bilateral Teleoperation, IEEE International Conference on Robotics and Biomimetics (IEEE, 2018), pp. 2208-2213.
6. Magid, E., Tsubouchi, T. Static Balance for Rescue Robot Navigation-Translation Motion Discretization Issue within Random Step Environment. In ICINCO v.2, 2010, pp. 415-422.
7. Safin, R., Lavrenov, R., Martínez-García, E. A., Magid, E. ROS-based Multiple Cameras Video Streaming for a Teleoperation Interface of a Crawler Robot. Journal of Robotics, Networking and Artificial Life, **5(3)** (2018), p.184-189.
8. S. Goebel, et al., Using the Android Platform to control Robots, in Proceedings of 2nd International Conference on Robotics in Education, 2011.
9. J. Kerr and K. Nickels, Robot operating systems: Bridging the gap between human and robot, in Southeastern Symposium on System Theory (IEEE, 2012), pp. 99–104.
10. N. Fung, Light weight portable operator control unit using an Android enabled mobile phone, in Proc. SPIE Unmanned Systems Technology XIII, Orlando, FL, 2011.
11. Y. Ji-Dong, et al., Development of Communication Model for Social Robots Based on Mobile Service, in IEEE Second International Conference on Social Computing (2010), pp. 57-64
12. S. Wu, Y. Chen, Remote robot control using intelligent hand-held devices, in Fourth Int. Conf. on Computer and Information Technology, Wuhan, China, 2004.